Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Μηχανικών Πληροφορικής

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη εφαρμογής ηλεκτρονικού εμπορίου και μηχανογράφησης με τεχνολογίες Java EE

Γεώργιος Γενετζάκης (Α.Μ. 2709)

Νίκη Μαλλιά (Α.Μ. 2799)

Επιβλέπων Καθηγητής: Νικόλαος Παπαδάκης

Abstract

Java Enterprise Edition is comprises of a number of libraries and frameworks that have become a de facto standard for the development of enterprise applications. In these thesis we detail the process of the creation of an online store using J2EE as well as the Vaadin framework for the administration backoffice.

Σύνοψη

Η Java Enterprise Edition είναι ένα σύνολο βιβλιοθηκών και προδιαγραφών που χρησιμοποιείται ευρέως σε servers και εφαρμογές που έχουν ιδιαίτερες απαιτήσεις επεκτασιμότητας, σταθερότητας και ταχύτητας. Σε αυτή την εργασία εξετάζουμε την δημιουργία μίας εφαρμογής ηλεκτρονικού καταστήματος με την χρήση αυτών των τεχνολογιών σε συνδυασμό με το Vaadin framework για την δημιουργία του περιβάλλοντος διαχείρισης της εφαρμογής.

Περιεχόμενα

1 Εισαγωγή

1.1 Περίληψη

Σε αυτή την εργασία παρουσιάζουμε την διαδικασία που ακολουθήσαμε για την ανάπτυξη ενός ηλεκτρονικού καταστήματος. Η εφαρμογή αποτελείται από δύο διακριτά μέρη, την σελίδα με την παρουσίαση του καταλόγου του καταστήματος που είναι προσβάσιμη σε όλους τους επισκέπτες (με περισσότερες δυνατότητες για τα εγγεγραμμένα μέλη) και την περιοχή διαχείρισης στην οποία έχουν πρόσβαση μόνο οι διαχειριστές του συστήματος.

1.2 Κίνητρο για την διεξαγωγή της εργασίας

Μία τυπική εφαρμογή ηλεκτρονικού εμπορίου χρησιμοποιεί την γλώσσα HTLM (σε συνδυασμό με CSS και Javascript) για την δημιουργία των σελίδων που βλέπουν οι χρήστες. Ο server χρησιμοποιεί κάποια γλώσσα προγραμματισμού για να δημιουργήσει δυναμικά τις σελίδες (τυπικά PHP, .Net ή Java σε συνδυασμό με MVC) και για την επικοινωνία με την βάση δημιουργούνται δυναμικά εντολές SQL.

Θέλοντας να εξετάσουμε τις δυνατότητες που δίνουν οι τεχνολογίες την Java EE επιλέξαμε να χρησιμοποιήσουμε την HTML μόνο για την δημιουργία της σελίδας του καταστήματος καθώς, κυρίως για λόγους συμβατότητας, είναι εξαιρετικά σπάνιο να χρησιμοποιηθεί κάποια άλλη τεχνολογία. Σε αυτόν τον τομέα επιλέξαμε να μην χρησιμοποιήσουμε την τεχνολογία JSP αλλά το Velocity template engine, με το οποίο ο κώδικας που παράγεται είναι απλούστερος.

Πέρα από αυτό για την δημιουργία του backend χρησιμοποιήσαμε το Vaadin framework, με το οποίο μπορούμε να χρησιμοποιήσουμε την γλώσσα Java για την συγγραφή του περιβάλλοντος και της συμπεριφοράς του. Το Vaadin αναλαμβάνει να μετατρέψει αυτόματα τα απαραίτητα κομμάτια κώδικα σε HTML (η οποία και παρουσιάζεται στον χρήστη), όμως κατά την ανάπτυξη χρησιμοποιείται μόνο η γλώσσα Java.

Για την επικοινωνία με την βάση χρησιμοποιήσαμε το JPA framework, στο οποίο περιγράφονται οντότητες Java, οι οποίες έπειτα αυτόματα μετατρέπονται σε πίνακες στην βάση.

Τέλος για την δημιουργία των δυναμικών σελίδων HTML χρησιμοποιήθηκε το Velocity template engine.

1.3 Σκοπός και στόχοι εργασίας

Ο στόχος της εργασίας ήταν μέσα από την δημιουργία της εφαρμογής να εξετάσουμε τα πλεονεκτήματα και τις διαφορές της χρήσης των τεχνολογιών που χρησιμοποιήσαμε σε σχέση με τον παραδοσιακό τρόπο ανάπτυξης.

Για αυτό τον λόγο οι τεχνολογίες που χρησιμοποιήσαμε επιλέχθηκαν με κριτήριο την ευρεία χρήση τους σε περιβάλλοντα εφαρμογών επιχειρήσεων.

1.4 Δομή εργασίας

Η εργασία είναι χωρισμένη στα εξής κεφάλαια:

- Κεφάλαιο 1: Αυτό το κεφάλαιο.
- Κεφάλαιο 2: Γενικές έννοιες σχετικά με πληροφοριακά συστήματα.
- Κεφάλαιο 3: Βασικά εισαγωγικά και ιστορικά στοιχεία σχετικά με τις τεχνολογίες που χρησιμοποιήθηκαν.

- Κεφάλαιο 4: Ανάλυση στοιχείων της εφαρμογής και παρουσίαση της βάσης δεδομένων.
- Κεφάλαιο 5: Παρουσίαση λειτουργιών του συστήματος.
- Κεφάλαιο 6: Συμπεράσματα και μελλοντικές επεκτάσεις.
- Κεφάλαιο 7: Βιβλιογραφία.

2 Γενικές και Βασικές Έννοιες

2.1 Μηχανογράφηση

Ως ορισμό μπορούμε να δεχτούμε την χρησιμοποίηση μηχανών σύγχρονης τεχνολογίας στην οργάνωση επιχειρήσεων και οργανισμών, για τη συγκέντρωση, ταξινόμηση και επεξεργασία στοιχείων.

Στην εποχή που ζούμε, η σπουδαιότητα μηχανογράφησης παίζει καθοριστικό ρόλο ως ανταγωνιστικό πλεονέκτημα απέναντι τόσο μέσα στην ίδια την εταιρία είτε οργανισμό όσο και εξωτερικά. Η σωστή διαχείριση της πληροφορίας αλλά και ο συνδυασμός της φύσης μας με την τεχνολογία μπορούν να δώσουν το καλύτερο αποτέλεσμα σχετικά με το πλάνο πορείας οργάνωσης και εξέλιξης. Η τεχνολογία προχωράει με γοργούς ρυθμούς και δεν επιτρέπει λάθος χειρισμούς τόσο για την σωστή ενημέρωση όσο και για την σωστή επεξεργασία των πόρων που υπάρχουν. Δεν χρειάζεται να γίνεται σπατάλη πόρων αλλά και καταπόνηση πνευματικής υγείας.

Η σωστή λύση είναι η σωστή μηχανογράφηση με την σωστή μηχανοργάνωση κάθε φορέα, οργανισμού και εταιρείας.

2.2 Πληροφοριακό Σύστημα

Πληροφοριακά συστήματα (*Information Systems* ή *IS*) ονομάζεται ένα σύνολο διαδικασιών, ανθρώπινου δυναμικού και αυτοματοποιημένων υπολογιστικών συστημάτων, που προορίζονται για τη συλλογή, εγγραφή, ανάκτηση, επεξεργασία, αποθήκευση και ανάλυση πληροφοριών. Τα συστήματα αυτά μπορούν να περιλαμβάνουν λογισμικό, υλικό και τηλεπικοινωνιακό σκέλος.

Τα πληροφοριακά συστήματα αποτελούν το μέσο για την αρμονική συνεργασία ανθρώπινου δυναμικού, δεδομένων, διαδικασιών και τεχνολογιών πληροφορίας και επικοινωνιών. Προέκυψαν ως γέφυρα μεταξύ των πρακτικών εφαρμογών της επιστήμης υπολογιστών και του επιχειρηματικού κόσμου. Κάθε ειδικό

πληροφοριακό σύστημα έχει ως στόχο την υποστήριξη των επιχειρήσεων, τη διαχείριση και λήψη αποφάσεων. Σε μια ευρεία έννοια, ο όρος χρησιμοποιείται για να αναφερθεί όχι μόνο στην τεχνολογία της πληροφορίας και της επικοινωνίας (ΤΠΕ), που ένας οργανισμός χρησιμοποιεί, αλλά στο τρόπο με τον οποίο οι άνθρωποι αλληλεπιδρούν με αυτή την τεχνολογία για την υποστήριξη των επιχειρηματικών διαδικασιών.

Ως εκ τούτου, τα πληροφοριακά συστήματα σχετίζονται με τα συστήματα διαχείρισης βάσης δεδομένων από τη μία πλευρά και με τα συστήματα δραστηριότητας από την άλλη. Ένα πληροφοριακό σύστημα είναι μια μορφή επικοινωνίας του συστήματος στο οποίο τα δεδομένα αντιπροσωπεύουν και υποβάλλονται σε επεξεργασία ως μια μορφή κοινωνικής μνήμης. Ένα πληροφοριακό σύστημα μπορεί επίσης να θεωρηθεί ως ημι-επίσημη γλώσσα που υποστηρίζει τις ανθρώπινες λήψεις αποφάσεων και δράσης.

2.3 Βάση Δεδομένων

Βάση δεδομένων ονομάζουμε μία συλλογή από συστηματικά μορφοποιημένα σχετιζόμενα δεδομένα στα οποία είναι δυνατή η ανάκτηση δεδομένων μέσω αναζήτησης κατ' απαίτηση. Ο Αμερικανός επιστήμονας υπολογιστών Τζιμ Γκρέϊ (Jim Gray) έχει γράψει για τις βάσεις δεδομένων: «Όταν οι άνθρωποι χρησιμοποιούν τις λέξεις βάση δεδομένων, διατυπώνουν στην ουσία ότι τα δεδομένα πρέπει να αναπροσδιορίζονται και να έχουν μια σχηματική δομή. Αυτό ακριβώς περιγράφουν οι λέξεις βάση δεδομένων».

Ειδικότερα, στην επιστήμη της πληροφορικής και στην καθημερινή χρήση των ηλεκτρονικών υπολογιστών, με τον όρο βάσεις δεδομένων αναφερόμαστε σε οργανωμένες, διακριτές συλλογές σχετιζόμενων δεδομένων ηλεκτρονικά και ψηφιακά αποθηκευμένων, στο λογισμικό που χειρίζεται τέτοιες συλλογές (Σύστημα Διαχείρισης Βάσεων Δεδομένων, ή DBMS) και στο γνωστικό πεδίο που το μελετά. Πέρα από την εγγενή της ικανότητα να αποθηκεύει δεδομένα, η βάση δεδομένων παρέχει μέσω του σχεδιασμού και του τρόπου ιεράρχησης των δεδομένων, τα αποκαλούμενα συστήματα διαχείρισης περιεχομένου, δηλαδή τη δυνατότητα γρήγορης άντλησης και ανανέωσης των δεδομένων.

2.4 Ηλεκτρονικό κατάστημα **(e-shop)**

E-shop είναι ο όρος που χρησιμοποιείται για να αναφερθεί κάποιος σε ένα Διαδικτυακό τόπο (site) μέσω του οποίου πραγματοποιούνται πωλήσεις διαφόρων ειδών. Ορισμένες πλατφόρμες δημοπρασιών προσφέρουν και συστήματα ηλεκτρονικών καταστημάτων.

Η Επανάσταση της Πληροφορικής άλλαξε σημαντικά τον τρόπο ζωής των πολιτών, επιφέροντας μια σειρά αλλαγών, που επηρεάζουν και τις εμπορικές επιχειρήσεις. Παλαιότερα, η ενσωμάτωση αυτή περιλάμβανε μόνο την ηλεκτρονική παρουσίαση των καταστημάτων, όχι όμως και όλων των ειδών τους και, πολύ περισσότερο, δεν υπήρχε δυνατότητα άμεσης παραγγελίας κάποιου είδους.

Το ηλεκτρονικό επιχειρείν (E-Business) αναφέρεται στην πραγματοποίηση επιχειρηματικών συναλλαγών μέσω του Internet και είναι η προσαρμογή του κλασικού επιχειρηματικού μοντέλου στην νέα ηλεκτρονική πραγματικότητα ή την ανάπτυξη νέου επιχειρηματικού μοντέλου με αντικείμενο μόνο το Διαδίκτυο.

Έτσι, αναπτύσσονται διεθνώς τα ηλεκτρονικά καταστήματα, που προσφέρουν ημερησίως χιλιάδες προϊόντα που υπόσχονται χαμηλότερες τιμές. Ανάλογα με τα προσφερόμενα είδη, ο μελλοντικός πελάτης μπορεί να αναζητήσει ανάμεσα σε πολλά ομοειδή το συγκεκριμένο είδος που επιθυμεί, να μάθει την τιμή και τον χρόνο αποστολής (εάν το παραγγείλει), να το δει σε εικόνες (ορισμένες φορές και σε βίντεο) και να κάνει και σχετικές συγκρίσεις τιμών. Οι τιμές στα ηλεκτρονικά καταστήματα είναι φθηνότερες, γιατί ένα τέτοιο κατάστημα δεν διατηρεί σημεία πώλησης με υψηλό ενοίκιο, δεν απασχολεί αριθμητικά το ίδιο προσωπικό με ένα συμβατικό και παραμένει "ανοικτό" σε 24ωρη βάση και για 365 μέρες ετησίως. Ο μέλλων πελάτης μπορεί ακόμη να βρει και να παραγγείλει είδη που δεν υπάρχουν στα συμβατικά καταστήματα της πόλεως ή της χώρας του και μπορεί να πληρώσει μέσω της πιστωτικής του κάρτας ή με την χρήση της αντικαταβολής ή Paypal.

Συνήθως οι πληροφορίες που πρέπει οπωσδήποτε να παρέχει ένα ηλεκτρονικό κατάστημα στους καταναλωτές του ώστε να θεωρείται αξιόπιστο θα πρέπει να περιλαμβάνουν:

- Ταυτότητα του εμπόρου
- Τρόποι επικοινωνίας με το e-κατάστημα τόσο με ηλεκτρονικό όσο και με συμβατικό τρόπο
- Τελική τιμή του προϊόντος ή της υπηρεσίας συμπεριλαμβανομένου ΦΠΑ και εξόδων αποστολής
- Εγγύηση του προϊόντος και υποστήριξη μετά την πώληση (After Sales Service)
- Χρόνος παράδοσης του προϊόντος
- Τρόπους πληρωμής και παράδοσης
- Τρόπους ακύρωσης της παραγγελίας σε περίπτωση λάθους ή αλλαγής γνώμης
- Πληροφορίες για την προστασία των προσωπικών ευαίσθητων δεδομένων

2.5 Δυναμικές ιστοσελίδες

Με τον όρο δυναμική ιστοσελίδα περιγράφουμε μία ιστοσελίδα της οποίας το περιεχόμενο δεν είναι προκαθορισμένο (όπως συμβαίνει στις στατικές ιστοσελίδες) αλλά δημιουργείται κάθε φορά που την επισκεπτόμαστε. Ο λόγος για αυτό είναι συνήθως ότι το περιεχόμενο που παρουσιάζει η σελίδα (μπορεί) να αλλάζει συνέχεια, οπότε χρειάζεται να δείχνει κάθε φορά την νεώτερη έκδοση. Ένα τυπικό παράδειγμα είναι ένα ειδησεογραφικό site, στο οποίο προστίθενται νέες ειδήσεις ή ανανεώνονται οι υπάρχοντες.

Όπως γίνεται κατανοητό από τα παραπάνω μία σελίδα δεν βρίσκεται αποθηκευμένη στο σύστημα όπως να παρουσιάζεται κάθε φορά. Αντίθετα η εφαρμογή βρίσκει το περιεχόμενο που πρέπει να παρουσιάσει (πχ στο προηγούμενο παράδειγμα τις πιο πρόσφατες ειδήσεις) και επίσης ξέρει την μορφή που πρέπει να έχει κάθε κομμάτι περιεχομένου (πχ πως σε μία λίστα με ειδήσεις κάθε είδηση έχει τον τίτλο της και μία μικρή φωτογραφία). Έπειτα συνδυάζει το περιεχόμενο με τους κατάλληλους κανόνες ώστε να δημιουργηθεί το επιθυμητό αποτέλεσμα, το οποίο βλέπει ο χρήστης.

2.5.1 Ροή δεδομένων σε μία δυναμική ιστοσελίδα

2.5.1.1 Αποστολή αιτήματος

Για να δει ένας χρήστης μία δυναμική ιστοσελίδα χρειάζεται να χρησιμοποιήσει οπωσδήποτε έναν web browser. Συγκεκριμένα πρέπει να γράψει την διεύθυνση της σελίδας ώστε ο browser να αποστείλει ένα αίτημα (request) στον server, από τον οποίο θα ζητήσει αυτή τη σελίδα.

2.5.1.2 Διευθύνσεις HTTP και web servers

Ένας server είναι ένα πρόγραμμα που περιμένει συνδέσεις σε μία συγκεκριμένη θύρα. Όταν ένας web browser αποστέλλει ένα request σε έναν server

τότε συνδέεται σε μία συγκεκριμένη διεύθυνση που προσδιορίζεται από το πρώτο τμήμα της διεύθυνσης της σελίδας. Χρησιμοποιείται πάντα το πρωτόκολλο HTTP, το οποίο καθορίζει πως εξ' ορισμού τέτοια αιτήματα στέλνονται στη θύρα 80. Ένας HTTP server (ή αλλιώς web server) ακούει στη θύρα 80 και περιμένει να εξυπηρετήσει αιτήματα που γίνονται σε αυτή.

Ένας web server μπορεί να τρέχει είτε στον υπολογιστή που γίνεται η αίτηση είτε σε κάποιον άλλο. Όταν ένας χρήστης ζητάει μία σελίδα στο internet τότε χρειάζεται να συνδεθεί στον υπολογιστή της εταιρείας που τρέχει τον web server, στον οποίο συνήθως αναφερόμαστε σαν server. Αντίθετα, κατά την διαδικασία ανάπτυξης μίας εφαρμογής ο web server τρέχει στον υπολογιστή που χρησιμοποιεί ο προγραμματιστής. Και στις δύο περιπτώσεις ο browser λειτουργεί με τον ίδιο ακριβώς τρόπο, απλά για σύνδεση σε έναν απομακρυσμένο server παρέχεται η διεύθυνση του server ενώ για σύνδεση σε έναν server που βρίσκεται στον ίδιο υπολογιστή παρέχεται η διεύθυνση του τοπικού υπολογιστή (localhost ή 127.0.0.1).

Εκτός από τη διεύθυνση στην οποία βρίσκεται ο server σε ένα URL συνήθως προσδιορίζεται και η σελίδα που θέλουμε να δούμε. Σε αρκετές περιπτώσεις δίνεται απλά η διεύθυνση του server

Παρακάτω χρησιμοποιούμε την λέξη server για να αναφερθούμε στο πρόγραμμα και όχι στον υπολογιστή στον οποίο τρέχει.

2.5.1.3 Επεξεργασία αιτήματος

Όταν ένας web server δεχθεί ένα αίτημα για μία σελίδα ελέγχει αν αυτή υπάρχει. Αν όχι επιστρέφει ένα λάθος πως τέτοιο αρχείο δεν βρέθηκε (με κωδικό 404). Σε περίπτωση που υπάρχει το τι γίνεται εξαρτάται από τον τύπο της σελίδας που ζητήθηκε και τις ρυθμίσεις του server. Διακρίνουμε δύο τυπικές περιπτώσεις.

Η πρώτη περίπτωση είναι να έχει ζητηθεί ένα αρχείο που έχει στατικό περιεχόμενο, δηλαδή περιεχόμενο το οποίο δεν χρειάζεται να εκτελεστεί από τον server. Τέτοια είδη περιεχομένου είναι αρχεία HTML, CSS, JavaScript και εικόνες. Τότε ο server απλά στέλνει το αρχείο που ζητήθηκε στον χρήστη.

Η δεύτερη περίπτωση είναι το αρχείο που ζητήθηκε να περιέχει δυναμικό περιεχόμενο. Ο server συνήθως καταλαβαίνει το είδος του αρχείου από την κατάληξή του και εφόσον είναι εκτελέσιμο το τρέχει. Η διαδικασία είναι παρόμοια με την εκτέλεση ενός προγράμματος με την διαφορά πως το αποτέλεσμα πρέπει να έχει την μορφή κειμένου που αποστέλλεται στον χρήστη. Γενικά δεν υπάρχει κανένας περιορισμός για το περιεχόμενο που παράγεται αλλά συνήθως είναι κώδικας HTML

Αν η εφαρμογή χρησιμοποιεί μία βάση δεδομένων τότε κατά την εκτέλεση του αρχείου μπορεί να γίνεται μία σύνδεση με αυτή ώστε να ζητηθούν ή να τροποποιηθούν δεδομένα. Μία βάση δεδομένων είναι ένας άλλος server, που όπως και ένας web server μπορεί να τρέχει στον ίδιο υπολογιστή με τον web server ή σε ένα άλλο μηχάνημα.

2.5.1.4 Επιστροφή δεδομένων στον client

Αφού τελειώσει η εκτέλεση του αρχείου, επιστρέφονται τα περιεχόμενα που παρήγαγε στον browser που έκανε το αρχικό αίτημα. Το αποτέλεσμα που βλέπει ο client είναι πάντα τύπου HTML, CSS, JavaScript ή εικόνες. Δεν μπορεί να καταλάβει αν αυτό το περιεχόμενο έχει δημιουργηθεί δυναμικά ή αν ήταν στατικό.

3 Οι Τεχνολογίες Που χρησιμοποιήθηκαν

Οι τεχνολογίες που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής που μας ζητήθηκε παρουσιάζονται παρακάτω και είναι οι εξής:

3.1 HTML

Αρχικά η δομή της σελίδας που βλέπει ο χρήστης περιγράφεται με την γλώσσα HTML. Ο στόχος της HTML είναι να περιγράψει κάθε ένα στοιχείο που περιέχει η σελίδα βάση του τύπου του, δηλαδή αν είναι μία επικεφαλίδα, μία παράγραφος ή ένα πεδίο κειμένου. Τα στοιχεία είναι ιεραρχικά τοποθετημένα, δηλαδή ένα στοιχείο μενού (nav) μπορεί να περιέχει μία επικεφαλίδα (που θα περιγράφει το είδος του μενού) και μία λίστα με επιλογές.

Πέρα από μία βασική δομή που πρέπει να έχουν όλες οι σελίδες και κάποιους γενικούς κανόνες δεν υπάρχει κάποιος περιορισμός στη σειρά των στοιχείων ή στο ποιο στοιχείο μπορεί να περιέχεται μέσα σε κάποιο άλλο. Αυτό εξαρτάται από το πώς θέλουμε να φαίνεται και τι να περιέχει η σελίδα. Για παράδειγμα το περιεχόμενο του στοιχείου header περιγράφει τα περιεχόμενα ενός άλλου στοιχείου. Το header μπορεί να βρίσκεται μέσα στο στοιχείο που περιγράφει, να προηγείται ή να έπεται.

Επιπλέον το ίδιο το περιεχόμενο του header δεν είναι δεδομένο. Μπορεί να περιέχει ένα στοιχείο με κείμενο επικεφαλίδας h1, το οποίο περιέχει το μεγαλύτερο κείμενο τίτλου. Όμως αν το header αναφέρεται σε κάποιο εσωτερικό στοιχειό της σελίδας μπορεί αντί για h1 να περιέχει h2. Επίσης μπορεί να υπάρχει ένα στοιχείο img που δείχνει μία εικόνα, πχ το λογότυπο της σελίδας.

Από τα παραπάνω φαίνεται πως δεν υπάρχει σωστός ή λάθος τρόπος για την συγγραφή του κώδικα HTML, απλά ακολουθείται η λογική. Αυτό αναφέρεται βέβαια στη δομή της σελίδας και τη διάρθρωση των στοιχείων, καθώς οι συντακτικοί κανόνες είναι απαραίτητο να ακολουθούνται.

Τέλος, τα πλεονεκτήματα της HTML είναι η ευκολία της στη χρήση, υποστηρίζεται από κάθε πρόγραμμα περιήγησης, είναι δωρεάν, δεν χρειάζεται η αγορά κάποιου λογισμικού, είναι εύκολο στη μάθηση και στη δημιουργία κώδικα, χρησιμοποιείται ευρέως.

3.1.1 HTML5

Κατά την ανάπτυξη της εφαρμογής χρησιμοποιήσαμε αρκετές δυνατότητες που προσφέρονται από την έκδοση 5 της γλώσσας HTML.

Γενικά οι εκδόσεις των γλωσσών προγραμματισμού καθορίζονται από παγκόσμια πρότυπα. Ένα πρότυπο περιέχει λεπτομερείς οδηγίες σχετικά με τις δυνατότητες και τον τρόπο λειτουργίας της γλώσσας, το οποίο συνήθως γίνεται αποδεκτό και παγιώνεται έπειτα από κοινή συμφωνία των κυριότερων φορέων της βιομηχανίας που αφορά η γλώσσα. Έπειτα τα προγράμματα τα οποία χρησιμοποιούν τη γλώσσα (συνήθως compilers, interpreters ή στην περίπτωση της HTML οι browsers) τροποποιούνται για να υποστηρίζουν τις νέες δυνατότητες.

3.1.2 Λειτουργία

Οι ετικέτες HTML λειτουργούν ανά ζεύγη όπως για παράδειγμα **<h1>**Επικεφαλίδα1**</h1>**, με την πρώτη ετικέτα να ονομάζεται ετικέτα έναρξης και τη δεύτερη ετικέτα λήξης. Ανάμεσα στις ετικέτες τοποθετούνται αρχεία όπως κείμενα, βίντεο, εικόνες κτλ όπου ο browser ερμηνεύει το περιεχόμενο, χωρίς όμως να εμφανίζει τις ετικέτες

HTML. Οι σελίδες που γράφονται σε αυτή τη μορφή είναι κείμενα και μπορούν να διαβαστούν από οποιονδήποτε χρησιμοποιεί απλό κείμενο. Δεν επηρεάζονται οι εντολές από το αν έχουν γραφτεί με κεφαλαία ή πεζά γράμματα. Τα αρχεία αυτά αποθηκεύονται ως *.html ή *.htm**.**

3.1.3 Μερικές συντάξεις και εντολές των στοιχείων της HTML

- Περιγραφή της ιστοσελίδας: <**html**>**…**</**html**>
- Κεφαλίδα εγγράφου HTML: <**head**>**...**</**head**>
- Καθορισμός του τίτλου της ιστοσελίδας, ο οποίος εμφανίζεται στο πάνω μέρος του browser:<**title**>....</**title**>
- Κύριο μέρος του αρχείου:<**body**>**…** </**body**>
- Επικεφαλίδες:<**h1**>Επικεφαλίδα1</**h1**><**h2**>Επικεφαλίδα2</**h2**><**h3**>Επικεφαλίδα3 </**h3**>
- Πίνακες δεδομένων:

  <**table**>**…**Οι γραμμές του πίνακα καθορίζονται από την ετικέτα:
  <**tr**>**…** </**tr**>Οι στήλες του πίνακα καθορίζονται από την ετικέτα:

  <**th**> **…** Τα δεδομένα στηλών καθορίζονται από την ετικέτα:

  <**td**>**…** </**td**>

  </**th**>

  </**table**>

- Πλαίσια (frames): <**frame**>**…** </**frame**>
- Μη – αριθμημένη λίστα:<**ul**>

  <**li**> Πρώτο στοιχείο

  <**li**> Δεύτερο στοιχείο

  </**ul**>

- Αριθμημένη λίστα:<**ol**>

  <**li**> Πρώτο στοιχείο

  <**li**> Δεύτερο στοιχείο

  </**ol**>

- Φόρμα: <**form**>**…**

  Σε περίπτωση εισαγωγής κειμένου:
  <**input** name="**..**" ...>
  Σε περίπτωση αποστολής δεδομένων συμπλήρωσης φόρμας μέσω κουμπιού:
  <**input** type="**submit**" value="**....**">
  </**form**>

3.1.4 Συμβατότητα μίας εφαρμογής με την HTML5

Η HTML5 δεν υπάρχει σαν ένα ολοκληρωμένο πρότυπο. Όμως αρκετά τμήματά του προτύπου είναι ήδη ολοκληρωμένα και έχουν υλοποιηθεί στους σύγχρονους browsers. Μία εφαρμογή που είναι γραμμένη σε HTML5 πρακτικά είναι γραμμένη σε HTML καθώς η νέα έκδοση είναι σε μεγάλο βαθμό συμβατή με την προηγούμενη και χρησιμοποιεί κάποιες από τις επιπλέον δυνατότητες.

Με αυτή τη λογική η εφαρμογή μας είναι γραμμένη σε HTML5. Τα στοιχεία της HTML5 που χρησιμοποιούμε είναι τα παρακάτω:

- Κάποια tag που ορίστηκαν στην Πέμπτη έκδοση της γλώσσας, όπως τα header, footer, nav.
- Κάποιες ιδιότητες που χρησιμοποιούμε στο αρχείο CSS, όπως το min-height.
- Κάποιες δυνατότητες της JavaScript που χρησιμοποιούνται κυρίως εσωτερικά από το JQuery.

3.2 CSS

Η CSS (Cascading Style Sheets) είναι μια γλώσσα φύλλων στυλ που χρησιμοποιείται για την περιγραφή της παρουσίασης ενός εγγράφου που είναι γραμμένο σε μια γλώσσα σήμανσης. Συνήθως χρησιμοποιείται για τον έλεγχο της εμφάνισης εγγράφων που έχουν γραφτεί στις γλώσσες HTML και XHMTL. Είναι ουσιαστικά σαν να έχει τον ρόλο του ζωγράφου που αποφασίζει με τι χρώματα, τι γράμματα, τι εμφάνιση θέλει να έχει στο πορτραίτο του και σε περίπτωση που θέλει να τα αλλάξει να μπορεί δημιουργήσει στον ίδιο καμβά το πορτραίτο του αλλαγμένο απλά πειράζοντας τους παραμέτρους που όρισε.

Η CSS έχει σχεδιαστεί για να επιτρέπει κυρίως τον διαχωρισμό του περιεχομένου ενός εγγράφου από την παρουσίαση του εγγράφου και για να προσφέρει περισσότερες δυνατότητες για μορφοποίηση από την HTML, συμπεριλαμβανομένων στοιχείων, όπως η διάταξη, τα χρώματα και τις γραμματοσειρές. Η πιο κοινή της εφαρμογή είναι σε ιστοσελίδες γραμμένες σε HTML και XHTML, αλλά η CSS μπορεί επίσης να εφαρμοστεί σε οποιοδήποτε είδος εγγράφου XML, συμπεριλαμβανομένου του απλού XML, SVG και XUL.

Η CSS δημιουργεί έναν δομημένο τρόπο παρακολούθησης του εγγράφου και των τροποποιήσεων της παρουσίασης του. Η χρήση της CSS και ο διαχωρισμός που προκύπτει του περιεχομένου από τη μορφοποίηση, παρέχει μεγαλύτερη ευελιξία και έλεγχο στην εξειδίκευση των χαρακτηριστικών παρουσίασης, βελτιώνει την προσβασιμότητα του περιεχομένου, προσφέρει τη δυνατότητα πολλές σελίδες να μοιράζονται την ίδια ακριβώς μορφοποίηση και μειώνει την πολυπλοκότητα και τις επαναλήψεις στον κώδικα που αφορά τη δόμηση του εγγράφου – ιστοσελίδας.

Ένα φύλλο στυλ αποτελείται από μια λίστα κανόνων. Κάθε κανόνας ή σετ κανόνων αποτελείται από έναν ή περισσότερους επιλογείς και ένα μπλοκ δήλωσης. Οι επιλογείς χρησιμοποιούνται για να δηλώσουν σε ποιο μέρος της σήμανσης εφαρμόζεται το στυλ. Το

τμήμα δήλωσης αποτελείται από μια λίστα δηλώσεων οι οποίες περικλείονται σε αγκύλες. Κάθε δήλωση αποτελείται από μια ιδιότητα, άνω και κάτω τελεία (:) και μια τιμή. Εάν υπάρχουν πολλές δηλώσεις σε ένα μπλοκ τότε εισάγεται ένα ερωτηματικό για το διαχωρισμό κάθε δήλωσης.

Η CSS επιτρέπει στην ίδια σελίδα σήμανσης (πχ μια HTML σελίδα) σε μια ιστοσελίδα να παρουσιαστεί με διαφορετικό τρόπο ανάλογα με το μέγεθος της οθόνης ή τη συσκευή στην οποία προβάλλεται να παρουσιάζεται με διαφορετικό στυλ ανάλογα με τη μέθοδο επεξεργασίας, όπως σε οθόνη υπολογιστή, σε συσκευές αφής ή σε έντυπη μορφή. Επιτρέπει επίσης.

Ο δημιουργός ενός εγγράφου σήμανσης συνήθως συνδέει το έγγραφο με ένα αρχείο CSS, όμως όσοι το διαβάζουν μπορούν να χρησιμοποιήσουν ένα διαφορετικό φύλλο στυλ και να παρακάμψουν αυτό που έχει καθορίσει ο δημιουργός.

Τα φύλλα στυλ ονομάστηκαν επικαλυπτόμενα ή αλλιώς διαδοχικά (cascading style sheets) γιατί η CSS έχει ορίσει ένα σύστημα προτεραιότητας για να καθορίζει ποιος κανόνας στυλ πρέπει να εφαρμοστεί αν περισσότεροι από ένας κανόνας ταιριάζουν με ένα συγκεκριμένο στοιχείο. Προτεραιότητες ή αλλιώς βάρη υπολογίζονται και εκχωρούνται στους κανόνες έτσι ώστε να υπάρχει μια σειρά και να μπορεί να προβλεφθεί η εμφάνιση της σελίδας.

Καλό θα ήταν να αναφέρουμε κάποιους περιορισμούς αλλά και πλεονεκτήματα από την άλλη που έχει αυτή την στιγμή η γλώσσα αυτή.

Ένα βασικό μειονέκτημα αποτελεί ότι οι επιλογείς δεν είναι σε θέση να καταλάβουν την ιεραρχία που προσφέρει η γλώσσα καθώς δεν υπάρχει τρόπος για να επιλεχθεί ένα γονέα ή πρόγονο ενός στοιχείου που πληρεί ορισμένα κριτήρια. Έχουν απορριφθεί διάφορες προτάσεις εξαιτίας ότι προκαλούν ζητήματα απόδοσης. Πιο εξειδικευμένοι επιλογείς χρησιμοποιούνται από την γλώσσα XPath για να υποστηρίζουν πιο πολύπλοκα έγγραφα.

Υπάρχουν κάθετοι περιορισμοί ελέγχου στην τοποθέτηση στοιχείων καθώς είναι δύσκολο να διαχειριστούν είτε αδύνατον ορισμένες φορές ενώ αντιθέτως η οριζόντια τοποθέτηση είναι διαχωρίσιμη. Απλές εργασίες όπως κεντράρισμα είναι ορισμένες φορές ανώφελο να ορισθούν μέσα από την γλώσσα για το έγγραφο.

Δυστυχώς εξαιτίας ότι δεν έχουν ακόμα οριστεί από το πρότυπο εκφράσεις δεν μπορούν να υποστηριχτούν και απουσιάζουν εντελώς ακόμα και απλές εκφράσεις όπως το περιθώριο κέρδους. Ενέργειες που είχαν γίνει σχετικά με αυτό σταμάτησαν να υποστηρίζονται γιατί πρέπει να υπάρχει συνάφεια των πρωτοτύπων με τους browsers ώστε να μην υπάρχουν προβλήματα επίδοσης και ασφάλειας.

Η έλλειψη της δήλωσης στήλη είναι ένα άλλο θέμα που ταλαιπωρεί τον σχεδιασμό και αν και είναι δυνατόν στη σημερινή CSS 3 (με τη χρήση της μονάδας στήλης-count), σχεδιαγράμματα με πολλαπλές στήλες μπορεί να είναι περίπλοκο να εφαρμοστεί στην CSS 2.1. Στην CSS 2.1, η διαδικασία γίνεται συχνά με τη χρήση πλωτών στοιχείων, τα οποία συχνά αποδίδονται με διαφορετικό τρόπο από διαφορετικούς browsers, διαφορετικά σχήματα ανάλογα με την οθόνη του υπολογιστή και τις αναλογίες της.

Δεν μπορεί να δηλωθεί καινούργιο πεδίο, ανεξάρτητα από τη θέση του.

Δεν γίνεται υποστήριξη και έλεγχος ψευδό- δυναμικής συμπεριφοράς. Η CSS υλοποιεί ψευδό-κλάσεις που επιτρέπουν ως ένα βαθμό τα σχόλια των χρηστών με όρους εφαρμογής εναλλακτικών μορφών. Όπως για παράδειγμα, μια CSS ψευδό-class, «: hover», είναι δυναμική (ισοδύναμη με το «onmouseover» της JavaScript) και έχει τη δυνατότητα

κατάχρησης- υπερκάλυψης (π.χ., στην εφαρμογή αναδυόμενα παράθυρα), αλλά η CSS δεν έχει καμία δυνατότητα για έναν πελάτη για να το απενεργοποιήσει. Δεν υπάρχει δυνατότητα απενεργοποίησης ή να περιορίσει τα αποτελέσματά της.

Δεν επιτρέπεται να γίνει ονομασία κανόνων και δεν υφίσταται να συμπεριληφθεί κάποιο στυλ από έναν κανόνα σε έναν άλλο κανόνα. Η επανάληψη αρκετών κανόνων σε style CSS πρέπει να πραγματοποιηθεί ώστε να επιτευχθεί το επιθυμητό αποτέλεσμα, προκαλώντας πρόσθετη συντήρηση και απαίτηση ενός ενδελεχή έλεγχου.

Από την άλλη πλευρά καλό είναι να αναλυθούν και τα πλεονεκτήματα της γλώσσας και απλά στα μειονεκτήματα να δείξουμε λίγη υπομονή μέχρι να υποστηριχθούν σιγά-σιγά.

Αρχικά, επιτυγχάνεται ο διαχωρισμός του περιεχομένου από την παρουσίαση. Η CSS διευκολύνει τη δημοσίευση του περιεχομένου σε πολλαπλές μορφές παρουσίασης με βάση τις ονομαστικές παραμέτρους. Μερικές από τις ονομαστικές παραμέτρους αποτελούν οι προτιμήσεις του χρήστη, τα διαφορετικά προγράμματα περιήγησης στο Web, ο τύπο της συσκευής που χρησιμοποιείται για να δει το περιεχόμενο είτε μέσω ενός επιτραπέζιου υπολογιστή είτε ενός κινητού), η γεωγραφική θέση του χρήστη και πολλές άλλες μεταβλητές.

Προσφέρει συνοχή αφενός με την χρήση κύριων άρθρων όπου ο διαχωρισμός της παρουσίασης, του περιεχομένου και των style sheets για τον σχεδιασμό ιστοσελίδων είναι ευδιάκριτος και αφετέρου όταν η CSS χρησιμοποιεί αποτελεσματικά, την έννοια της κληρονομικότητας και υπερχείλισης. Ένα παγκόσμιο φύλλο στυλ μπορεί να χρησιμοποιηθεί για να επηρεάσει τα στοιχεία style σε όλη την εμβέλεια. Εάν σύμφωνα με την περίσταση, προκύπτει ότι το style sheet των στοιχείων θα πρέπει να αλλάξει ή να προσαρμοστεί, κάτι τέτοιο πριν την CSS, ήταν πιο δύσκολο, δαπανηρό και χρονοβόρο, τώρα όμως εύκολο, γρήγορο και ευέλικτο.

Παρέχει την δυνατότητα εύκολης και γρήγορης επεξεργασίας και αλλαγής παρουσίασης της ιστοσελίδας. Με μια απλή αλλαγή σε μία γραμμή, ένα διαφορετικό stylesheet μπορεί να χρησιμοποιηθεί για την ίδια σελίδα. Αυτό έχει πλεονεκτήματα σχετικά με την προσβασιμότητα αλλά παρέχει και τη δυνατότητα να προσαρμόζει μια σελίδα ή ένα ολόκληρο site σε διαφορετικές συσκευές.

Το βασικότερο όλων όμως είναι η προσβασιμότητα καθώς χωρίς το CSS, οι σχεδιαστές ιστοσελίδων έπρεπε τυπικά να ορίζουν τις σελίδες τους με τεχνικές όπως πίνακες HTML που εμπόδιζαν την προσβασιμότητα σε χρήστες με προβλήματα όρασης χρήστες, πράγμα το οποίο είναι τώρα εφικτό.

3.2.1 Προγραμματιστικός Συνδυασμός HTML και CSS

Ο συνδυασμός HTML και CSS**,** αν και γίνεται είναι γενικά πολύ κακή πρακτική. Γενικός στόχος είναι να ξεχωρίζουμε το περιεχόμενο από την εμφάνιση, ώστε οι αλλαγές στο ένα να μην επηρεάζουν το άλλο. Για αυτό το λόγο τυπικά οι οδηγίες CSS δίνονται σε ένα ξεχωριστό αρχείο CSS, το οποίο συμπεριλαμβάνουμε στη σελίδα με την κατάλληλη εντολή. Για να εισάγουμε ένα αρχείο CSS προσθέτουμε μέσα στο στοιχείο head μία γραμμή που καθορίζει το αρχείο που θέλουμε να χρησιμοποιηθεί.

```
<!doctype html>

<html>

<head>

        <meta charset=”utf-8”>
```

```
<title> Εφαρμογή Πτυχιακής</title>

<link href= “style.css” rel = “stylesheet” type=”text/css”>
```

Ο παραπάνω κώδικας αποτελεί την αρχή μίας σελίδας HTML που χρησιμοποιεί ένα αρχείο CSS.

Το παραπάνω παράδειγμα δείχνει ξεκάθαρα τον ρόλο του CSS σε αντιδιαστολή με την HTML. Το αρχείο style.css που εισήγαμε έχει μέγεθος περίπου διακόσιες γραμμές και αλλάζει ριζικά τον τρόπο εμφάνισης της σελίδας. Όμως όλες οι πληροφορίες για την εμφάνιση είναι περιορισμένες σε αυτό, και για την χρήση του δεν χρειάζεται καμία επιπλέον αλλαγή στην HTML. Ο κώδικας της σελίδας παραμένει ξεκάθαρος και διαβάζοντάς τον είναι εύκολο να καταλάβουμε το περιεχόμενό της και τον ρόλο του κάθε στοιχείου.

3.3 Η γλώσσα προγραμματισμού JavaScript

Η JavaScript (JS) είναι διερμηνευμένη γλώσσα προγραμματισμού για ηλεκτρονικούς υπολογιστές.] Αρχικά αποτέλεσε μέρος της υλοποίησης των browser, ώστε τα client-side scripts να μπορούν να επικοινωνούν με τον χρήστη, να ανταλλάσσουν δεδομένα ασύγχρονα και να αλλάζουν δυναμικά το περιεχόμενο του εγγράφου που εμφανίζεται.

Η JavaScript είναι μια γλώσσα σεναρίων που βασίζεται στα πρωτότυπα (prototype-based), είναι δυναμική, με ασθενείς τύπους και έχει συναρτήσεις ως αντικείμενα πρώτης τάξης. Η σύνταξή της είναι επηρεασμένη από τη C. Η JavaScript αντιγράφει πολλά ονόματα και συμβάσεις ονοματοθεσία από τη Java, αλλά γενικά οι δύο αυτές γλώσσες δε σχετίζονται και έχουν πολύ διαφορετική σημασιολογία. Οι βασικές αρχές σχεδιασμού της JavaScript προέρχονται από τις γλώσσες προγραμματισμού Self και Scheme. Είναι γλώσσα βασισμένη σε διαφορετικά προγραμματιστικά παραδείγματα (multi-paradigm), υποστηρίζοντας αντικειμενοστραφές, προστακτικό και συναρτησιακό στυλ προγραμματισμού.

Η JavaScript χρησιμοποιείται και σε εφαρμογές εκτός ιστοσελίδων — τέτοια παραδείγματα είναι τα έγγραφα PDF, οι εξειδικευμένοι browser (site-specific browsers) και οι μικρές εφαρμογές της επιφάνειας εργασίας (desktop widgets). Οι νεότερες εικονικές μηχανές και πλαίσια ανάπτυξης για JavaScript (όπως το Node.js) έχουν επίσης κάνει τη JavaScript πιο δημοφιλή για την ανάπτυξη server-side εφαρμογών ..

3.3.1 Λειτουργίες

Παραπάνω είδαμε πως η δομή μιας σελίδας περιγράφεται από τη γλώσσα HTML και η εμφάνισή της από τη γλώσσα CSS. Είναι φανερό πως και οι δύο γλώσσες είναι στατικές, δηλαδή απλά δίνουν οδηγίες για το τι και πώς θα φαίνεται, και όχι για το πώς θα λειτουργεί.

Καθώς οι σελίδες HTML παρουσιάζουν περιεχόμενο στον χρήστη, αρκετές φορές αυτό αρκεί. Όμως υπάρχουν περιπτώσεις που θέλουμε να κάνουμε μία σελίδα να αντιδράει στις επιλογές του χρήστη με διάφορους τρόπους. Κάτι τέτοιο δεν είναι απαραίτητο, όμως μπορεί να κάνει την σελίδα πολύ πιο εύκολη στη χρήση.

Ένα παράδειγμα έχει να κάνει με τον έλεγχο στοιχείων μίας φόρμας. Ένα τυπικό σενάριο είναι μία φόρμα εγγραφής νέου χρήστη. Σε αυτή ο χρήστης συμπληρώνει τα στοιχεία του και υποβάλει την φόρμα, δηλαδή στέλνει αυτά τα στοιχεία στον server.

Πριν γίνει η αποθήκευση των στοιχείων του νέου χρήστη χρειάζεται να γίνει έλεγχος των στοιχείων που στάλθηκαν, για παράδειγμα αν έχουν συμπληρωθεί όλα τα υποχρεωτικά πεδία και αν έχουν δοθεί έγκυρα στοιχεία, πχ σε ένα πεδίο που ζητάει το email του χρήστη. Αν τα στοιχεία δεν είναι έγκυρα τότε δεν μπορεί να γίνει αποθήκευση, οπότε είναι

απαραίτητο να εμφανιστεί πάλι η φόρμα και να εμφανιστεί το κατάλληλο μήνυμα που να ειδοποιεί τον χρήστη για τα λάθη.

Παρόλο που η παραπάνω προσέγγιση λειτουργεί, είναι κουραστική και δυσνόητη για τον χρήστη. Αφού αυτός συμπληρώνει την φόρμα εγγραφής περιμένει να δει το αποτέλεσμά της ενέργειάς του. Το να εμφανίζεται πάλι η ίδια φόρμα μπορεί να τον αποπροσανατολίσει, ειδικά αν δεν έχει εμπειρία, ενώ χρειάζεται να καταβάλει προσπάθεια για καταλάβει τι λάθος έκανε.

Στο παραπάνω παράδειγμα η φόρμα θα μπορούσε να ελέγχεται με JavaScript πριν την υποβολή της. Με αυτόν τον τρόπο τα λάθη θα εμφανίζονταν μόλις ο χρήστης πατούσε το κουμπί υποβολής, χωρίς να ξαναφορτωθεί η σελίδα. Με αυτόν τον τρόπο μπορεί να καταλάβει αμέσως τι έχει συμβεί, αφού το λάθος εμφανίστηκε σαν αντίδραση στο πάτημα του κουμπιού, κάνοντας την εφαρμογή πιο φιλική.

Αξίζει να σημειωθεί πως αν και με τη JavaScript είναι δυνατόν να τροποποιηθεί το περιεχόμενο μιας σελίδας, υπάρχουν κάποιοι βασικοί περιορισμοί. Ο πρώτος είναι πως για την εκτέλεση πολύπλοκων λειτουργιών ο κώδικας JavaScript που απαιτείται είναι αρκετά πιο μπερδεμένος σε σχέση με κάποιες παραδοσιακές γλώσσες προγραμματισμού. Επιπλέον υπάρχουν αρκετές μικρές διαφοροποιήσεις στον ακριβή τρόπο λειτουργίας της JavaScript ανάμεσα σε διαφορετικούς browsers, κάτι το οποίο κάνει απαραίτητο τον έλεγχο της σελίδας σε κάθε έναν από αυτούς.

Ο βασικότερος περιορισμός της JavaScript είναι πως, αν και είναι δυνατή η επικοινωνία με τον server μέσω AJAX, δεν γίνεται με κανέναν τρόπο να επικοινωνήσει με την βάση δεδομένων, εκτός και αν παρεμβάλλεται μία σελίδα γραμμένη σε κάποια γλώσσα που εκτελείται στον server. Έτσι ο ρόλος της είναι βοηθητικός, στο να προσφέρει κάποια επιπλέον λειτουργικότητα, αλλά η δημιουργία της βασικής σελίδας γίνεται στον server.

3.3.2 JavaScript Frameworks

Όπως αναφέραμε προηγουμένως κάθε browser έχει διαφορετικές υλοποιήσεις της JavaScript, οι οποίες μπορούν να διαφέρουν μεταξύ τους. Για να εξασφαλιστεί η απρόσκοπτή λειτουργία μίας σελίδας που χρησιμοποιεί JavaScript χρειάζεται να ελεγχθεί σε όλους τους πιθανούς browsers που θα την τρέξουν, κάτι το οποίο είναι εξαιρετικά επίπονο και πολλές φορές πρακτικά αδύνατο, καθώς δεν γίνεται να καλυφθούν όλες οι εκδόσεις του κάθε browser, ιδιαίτερα αν λάβουμε υπόψη μας και τους διαφορετικούς browsers που υπάρχουν σε ένα πλήθος κινητών τηλεφώνων και tablets.

Αν και υπάρχουν μικρές ασυμβατότητες στην υλοποίηση και τον τρόπο εμφάνισης των στοιχείων HTML και των κανόνων CSS, το πρόβλημα είναι μεγαλύτερο στην JavaScript γιατί πχ, μία διαφορά σε έναν κανόνα CSS συνήθως οδηγεί σε διαφορετική απεικόνιση ενός στοιχείου, ενώ ένα κομμάτι κώδικα JavaScript μπορεί να πάψει να λειτουργεί. Γι’ αυτόν τον λόγο σπάνια γίνεται άμεσα χρήση εντολών μόνο της JavaScript που περιέχεται σε έναν browser, αλλά συνήθως χρησιμοποιείται ένα επιπλέον Framework.

Ένα framework είναι μία βιβλιοθήκη που περιέχει κάποια ήδη ορισμένα σύμβολα (μεταβλητές, κλάσης και συναρτήσεις) τα οποία παρέχουν κάποιες λειτουργίες, οι οποίες μπορεί να αντικαθιστούν τις λειτουργίες που περιέχει η JavaScript ή και να προσφέρουν επιπλέον δυνατότητες. Είναι και αυτό γραμμένο σε JavaScript, όμως με τέτοιο τρόπο ώστε να εγγυάται την συμβατότητα με αρκετούς browsers και πολλές φορές να προσφέρει έτοιμες κάποιες ενέργειες, συνήθως με την μορφή έτοιμων συναρτήσεων, οι οποίες αν και μπορούν να υλοποιηθούν με JavaScript είναι είτε τετριμμένες είτε αρκετά δύσκολες στην υλοποίηση.

Ένα βασικό framework μπορεί να προσφέρει κάποιες έτοιμες δυνατότητες, όμως αρκετά από αυτά είναι αρκετά πιο εξελιγμένα και έχουν συγκεκριμένους κανόνες χρήσης. Η εκμάθηση ενός framework χρειάζεται κάποια προσπάθεια, όμως αποδίδει γιατί στην πορεία μειώνει τον κόπο για την υλοποίηση των επιμέρους λειτουργιών που χρησιμοποιούνται και καθιστά περιττό τον έλεγχο σε κάθε browser.

Το βασικό μειονέκτημα που έχει η χρήση ενός framework, πέρα από τον κόπο που χρειάζεται για την εκμάθησή του, είναι πως τυπικά για την εκτέλεση μίας λειτουργίας απαιτείται η χρήση πιο πολύπλοκου κώδικα, τον οποίον ο προγραμματιστής δεν βλέπει. Αυτό μπορεί να έχει επιπτώσεις στην ταχύτητα εκτέλεσης των λειτουργιών σε μία σελίδα.

Η πτώση των επιδόσεων ήταν παλιότερα μεγάλο πρόβλημα, όμως με το πέρασμα του χρόνου τα περισσότερα frameworks έχουν βελτιστοποιηθεί αρκετά σε αυτόν τον τομέα, και σε συνδυασμό με τη συνεχή αύξηση των επιδόσεων των υπολογιστών έχει πάψει να είναι αποτρεπτικός παράγοντας.

3.3.3 jQuery

Για την ανάπτυξη της εφαρμογής μας επιλέξαμε να χρησιμοποιήσουμε το framework jQuery. Αν και υπάρχουν πάρα πολλές παρόμοιες λύσεις το jQuery συνδυάζει μικρό μέγεθος, πάρα πολλές δυνατότητες και αρκετά καλές επιδόσεις, με αποτέλεσμα να είναι ένα από τα ποιο δημοφιλή frameworks. Είναι χαρακτηριστικό πως χρησιμοποιείται στο 65% των 10000 σελίδων με τις περισσότερες επισκέψεις. Πρόκειται για ένα λογισμικό ανοικτού κώδικα, που συντηρείται από μία ομάδα εθελοντών.

Η βασική διαφορά που έχει από την απλή JavaScript είναι πως αλλάζει αρκετά το βασικό συντακτικό, από την άποψη ότι κάνει χρήση μίας συνάρτησης που ονομάζεται jQuery και συνήθως χρησιμοποιείται για αυτή το σύμβολο του δολαρίου ($).

Παρακάτω θα δώσουμε ένα παράδειγμα στο οποίο φαίνεται η διαφορά του jQuery από την απλή JavaScript, όπως φαίνεται μέσα από μία κλήση AJAX. Για να κάνουμε κάτι τέτοιο με απλή JavaScript απαιτείται ο παρακάτω κώδικας.

```
function loadXMLDoc() {

   var xmlhttp;



   // code for IE7+, Firefox, Chrome, Opera, Safari

   if (window.XMLHttpRequest) {

      xmlhttp=new XMLHttpRequest();

   } else { // code for IE6, IE5

      xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");

   }

   xmlhttp.onreadystatechange=function() {

      if (xmlhttp.readyState==4 && xmlhttp.status==200) {

         document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
```

```
        }
    }
    xmlhttp.open("POST","ajax_info.txt",true);
    xmlhttp.send();
}
```

Όπως φαίνεται στην αρχή χρειάζεται να πάρουμε περιπτώσεις για διαφορετικούς browsers, ενώ μετά ορίζεται μία μέθοδος που θα καλεστεί αφού έρθει η απάντηση ή αν αποτύχει το αίτημα στον server. Τέλος ορίζεται η διεύθυνση και το τρόπος αποστολής του αιτήματος και αποστέλλονται οι πληροφορίες.

Για να γίνει η ίδια διαδικασία με το jQuery χρειάζεται ο παρακάτω κώδικας:

```
$.ajax({ url: "ajax_info.txt", success: function(data) {
        $("#myDiv").html(data);
    }
});
```

Απευθείας φαίνεται η σαφώς μικρότερη έκταση που καταλαμβάνει ο κώδικας, καθώς επίσης και η παράλειψη των περιπτώσεων που έχουν να κάνουν με το είδος και την έκδοση του browser. Επίσης φαίνεται η απλοποίηση των περιπτώσεων, δηλαδή ο έλεγχος σχετικά με το αν η απάντηση είναι σωστή έχει αντικατασταθεί με τον ορισμό μίας μεθόδου στην ιδιότητα success.

Ο κώδικας του jQuery είναι αρκετά πυκνός, πράγμα το οποίο μειώνει αρκετά την έκταση, αλλά την ίδια στιγμή είναι πιο εύκολο να γίνει κάποιο λάθος. Παραπάνω έχουμε μία συνάρτηση που παίρνει για παράμετρο έναν πίνακα – αντικείμενο javascript, ένα από τα στοιχεία του είναι μία συνάρτηση. Τέτοιες κλήσεις είναι αρκετά συνηθισμένες και απαιτούν από τον χρήστη να είναι αρκετά προσεκτικός.

3.4 MySQL

Οι σχεσιακές βάσεις δεδομένων, δηλαδή αυτές που χρησιμοποιούν την γλώσσα SQL για την επεξεργασία και την ανάπτυξη δεδομένων είναι η προφανής επιλογή για την αποθήκευση μίας πληθώρας δεδομένων, διευκολύνοντας την ανάκτηση τους ανάλογα με πολλά κριτήρια συνδυάζοντας δεδομένα από διαφορετικούς πίνακες.

Σαν βάση δεδομένων χρησιμοποιήσαμε την MySQL. Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων (Relational Database Management System - RDBMS), δηλαδή ένας server που μπορεί να περιέχει πολλές διαφορετικές μεταξύ τους

βάσεις δεδομένων. Κάθε βάση έχει ένα μοναδικό όνομα και μπορεί να περιέχει πολλούς πίνακες, ενώ τα περιεχόμενα δύο διαφορετικών βάσεων δεν σχετίζονται μεταξύ τους.

Πρόκειται για την περισσότερο διαδεδομένη βάση δεδομένων, με εκατομμύρια εγκαταστάσεις σε όλο τον κόσμο. Προσφέρει αρκετά καλές επιδόσεις και υποστηρίζει ένα μεγάλο εύρος της γλώσσας SQL, αν και έχει κάποιες ελλείψεις σε σχέση με άλλες βάσεις, με πιο εμφανή στο θέμα της υποστήριξης των ξένων κλειδιών.

Συγκεκριμένα με τις κατάλληλες εντολές συνδεόμαστε στη βάση και μετά χρησιμοποιούμε κάποιες συναρτήσεις που παίρνουν για παραμέτρους ερωτήματα SQL σαν string. Οποιοδήποτε ερώτημα μπορεί να εκτελεστεί, είτε πρόκειται απλή επερώτηση είτε εάν σχετίζεται με δημιουργία νέων εγγραφών, τροποποίηση υπαρχόντων ή διαγραφή.

Το πρόγραμμα τρέχει έναν εξυπηρετητή (server) ο οποίος παρέχει πρόσβαση πολλών χρηστών σε ένα σύνολο βάσεων δεδομένων. Είναι γνωστή για διαδικτυακά προγράμματα και ιστοσελίδες και χρησιμοποιείται στις πιο διαδεδομένες ιστοσελίδες όπως το Twitter, το Google, τη Wikipedia και το YouTube. Μία βάση επιτρέπει να αποθηκεύσουμε, να ταξινομήσουμε και να αναζητήσουμε αποτελεσματικά τα δεδομένα. Η MySQL ελέγχει την πρόσβαση στα δεδομένα μας ώστε να μπορούν να δουλεύουν πολλοί χρήστες ταυτόχρονα, αλλά και να ελέγχει ότι μόνο πιστοποιημένοι χρήστες θα μπορούν να έχουν πρόσβαση.

3.4.1 Πλεονεκτήματα και Μειονεκτήματα:

Τέλος, επισημαίνουμε τα πλεονεκτήματα της MYSQL είναι τα εξής είναι πολύ γρήγορο και δυνατό σύστημα διαχείρισης βάσεων δεδομένων, μπορούν πολλές συνδέσεις με τη βάση να υπάρχουν ταυτόχρονα χωρίς να υπάρχουν πολλαπλά αντίγραφα της, η απόδοση της είναι καλύτερη σε μεγαλύτερο όγκο βάσεων δεδομένων, είναι οικονομική, είναι λογισμικό ανοιχτού κώδικα, είναι γρήγορη στην ανάκτηση δεδομένων, παρέχει ευκολίες στο backup. Επίσης καλό είναι να αναφερθούμε σ τα μειονεκτήματα της MySQL που είναι ότι οι συναλλαγές δεν αντιμετωπίζονται αρκετά αποτελεσματικά, δεν υποστηρίζει ένα μεγάλο μέγεθος της βάσης δεδομένων και δεν υποστηρίζει ξένα κλειδιά.

3.5 Java

Η Java είναι μια αντικειμενοστραφής γλώσσα προγραμματισμού που σχεδιάστηκε από την εταιρεία πληροφορικής Sun Microsystems. Ένα από τα βασικά πλεονεκτήματα της Java έναντι των περισσότερων άλλων γλωσσών είναι η ανεξαρτησία του λειτουργικού συστήματος και πλατφόρμας. Τα προγράμματα που είναι γραμμένα σε Java τρέχουνε ακριβώς το ίδιο σε Windows, Linux, Unix και Macintosh (χωρίς να χρειαστεί να ξαναγίνει μεταγλώττιση (compiling) ή να αλλάξει ο πηγαίος κώδικας για κάθε διαφορετικό λειτουργικό σύστημα.

Για να επιτευχθεί όμως αυτό χρειαζόταν κάποιος τρόπος έτσι ώστε τα προγράμματα γραμμένα σε Java να μπορούν να είναι «κατανοητά» από κάθε υπολογιστή ανεξάρτητα του είδους επεξεργαστή (Intel x86, IBM, Sun SPARC, Motorola) αλλά και λειτουργικού συστήματος (Windows, Unix, Linux, BSD, MacOS). Ο λόγος είναι ότι κάθε κεντρική μονάδα επεξεργασίας κατανοεί διαφορετικό κώδικα μηχανής. Ο συμβολικός κώδικας (assembly) που μεταφράζεται και εκτελείται σε Windows είναι διαφορετικός από αυτόν που μεταφράζεται και εκτελείται σε έναν υπολογιστή Macintosh. Η λύση δόθηκε με την ανάπτυξη της Εικονικής Μηχανής (Virtual Machine).

3.5.1 Τα χαρακτηριστικά της Java

Ένα από τα βασικά πλεονεκτήματα της Java εν συγκρίσει με των υπολοίπων περισσοτέρων γλωσσών είναι η ανεξαρτησία που διαθέτει ως προς το λειτουργικό σύστημα και την πλατφόρμα. Αρχικά τα προγράμματα που είναι γραμμένα σε Java τρέχουν με τον ίδιο

ακριβώς τρόπο τόσο σε Windows, Linux, Unix όσο και σε Macintosh (σύντομα θα τρέχουν και σε Playstation καθώς και σε άλλες κονσόλες παιχνιδιών) χωρίς να χρειαστεί να ξαναγίνει μεταγλώττιση (compiling) ή ακόμα πιο σημαντικό να αλλάξει ο πηγαίος κώδικας για κάθε διαφορετικό λειτουργικό σύστημα. Για να επιτευχθεί ο στόχος αυτός όμως, χρειάστηκε να βρεθεί κάποιος τρόπος έτσι ώστε τα προγράμματα που ήταν γραμμένα σε Java να μπορούν να είναι «κατανοητά» από κάθε υπολογιστή ανεξάρτητα από το είδος του επεξεργαστή (Intel x86, IBM, Sun SPARC, Motorola) αλλά και του λειτουργικού συστήματος (Windows, Unix, Linux, BSD, MacOS). Ο λόγος που συμβαίνει αυτό είναι ότι κάθε κεντρική μονάδα επεξεργασίας κατανοεί διαφορετικό κώδικα μηχανής. Κατά συνέπεια ο συμβολικός κώδικας (assembly) που μεταφράζεται και εκτελείται σε Windows είναι διαφορετικός από αυτόν που μεταφράζεται και εκτελείται σε έναν υπολογιστή Macintosh. Η λύση προήλθε με την ανάπτυξη της Εικονικής Μηχανής (VirtualMachine ή VM ή EM στα ελληνικά).Σύμφωνα με την Εικονική Μηχανή όταν γραφτεί κάποιο πρόγραμμα σε Java, στη συνέχεια μεταγλωττίζεται μέσω του μεταγλωττιστή javac, ο οποίος παράγει έναν αριθμό από αρχεία .class (κώδικας byte ή bytecode). Ο κώδικας byte είναι η μορφή που παίρνει ο πηγαίος κώδικας της Java όταν μεταγλωττιστεί. Όταν χρειαστεί να εκτελεστεί η εφαρμογή σε ένα μηχάνημα, το JavaVirtualMachine που πρέπει να είναι εγκατεστημένο σε αυτό θα αναλάβει να διαβάσει τα αρχεία .class. Στη συνέχεια τα μεταφράζει σε γλώσσα μηχανής που να μπορούν να υποστηριχτεί από το λειτουργικό σύστημα και τον επεξεργαστή, έτσι ώστε να εκτελεστεί (να σημειωθεί εδώ ότι αυτό συμβαίνει με την παραδοσιακή Εικονική Μηχανή (VirtualMachine). Οι πιο σύγχρονες εφαρμογές της Εικονικής Μηχανής μπορούν και μεταγλωττίζουν εκ των προτέρων τμήματα bytecode απευθείας σε κώδικα μηχανής (εγγενή κώδικα ή nativecode) με αποτέλεσμα να βελτιώνεται η ταχύτητα. Χωρίς αυτό δε θα ήταν δυνατή η εκτέλεση λογισμικού γραμμένου σε Java. Αξίζει να σημειωθεί ότι η JVM είναι λογισμικό που εξαρτάται από την πλατφόρμα, δηλαδή για κάθε είδος λειτουργικού συστήματος και αρχιτεκτονικής επεξεργαστή υπάρχει διαφορετική έκδοση του. Έτσι υπάρχουν διαφορετικές JVM για Windows, Linux, Unix, Macintosh, κινητά τηλέφωνα, παιχνιδομηχανές κλπ.

Οτιδήποτε θέλει να κάνει ο προγραμματιστής (ή ο χρήστης) γίνεται μέσω της εικονικής μηχανής. Εξαιτίας αυτού υπάρχει μεγαλύτερη ασφάλεια στο σύστημα γιατί η εικονική μηχανή είναι υπεύθυνη για την επικοινωνία χρήστη - υπολογιστή. Γενικά ισχύει ότι ο προγραμματιστής δεν μπορεί να γράψει κώδικα ο οποίος θα έχει καταστροφικά αποτελέσματα για τον υπολογιστή γιατί η εικονική μηχανή θα τον ανιχνεύσει και δε θα επιτρέψει να εκτελεστεί. Παρομοίως, στην άλλη μεριά ούτε ο χρήστης μπορεί να κατεβάσει «κακό» κώδικα από το δίκτυο και να τον εκτελέσει. Αυτό είναι ιδιαίτερα χρήσιμο για μεγάλα κατανεμημένα συστήματα όπου πολλοί χρήστες χρησιμοποιούν το ίδιο πρόγραμμα συγχρόνως.

Η ύπαρξη του συλλέκτη απορριμμάτων (GarbageCollector) αποτελεί ακόμα μία ιδέα που βρίσκεται πίσω από τη Java. Συλλογή απορριμμάτων είναι μία κοινή ονομασία που χρησιμοποιείται στον τομέα της πληροφορικής για να δηλώσει την ελευθέρωση τμημάτων

μνήμης από δεδομένα που δε χρειάζονται και δε χρησιμοποιούνται άλλο. Στη Java η απελευθέρωση μνήμης είναι αυτόματη και γίνεται μέσω του συλλέκτη απορριμμάτων. Η εικονική μηχανή είναι και πάλι υπεύθυνη για αυτό και μόλις «καταλάβει» ότι ο σωρός (heap) της μνήμης (στη Java η συντριπτική πλειοψηφία των αντικειμένων αποθηκεύονται στο σωρό σε αντίθεση με τη C++ όπου αποθηκεύονται κυρίως στη στοίβα) κοντεύει να γεμίσει ενεργοποιεί το συλλέκτη απορριμμάτων. Συνεπώς ο προγραμματιστής δε χρειάζεται να ανησυχεί για το πότε και αν θα ελευθερώσει ένα συγκεκριμένο τμήμα της μνήμης, ούτε και για σφάλματα δεικτών. Αυτό είναι ιδιαίτερα σημαντικό γιατί είναι κοινά τα σφάλματα προγραμμάτων που οφείλονται σε λανθασμένο χειρισμό της μνήμης.

Παρόλο που η εικονική μηχανή προσφέρει όλα αυτά τα πλεονεκτήματα, η Java αρχικά ήταν πιο αργή σε σχέση με άλλες προγραμματιστικές γλώσσες υψηλού επιπέδου (high-level) όπως η C και η C++. Στο παρελθόν εμπειρικές μετρήσεις είχαν δείξει ότι η C++ μπορούσε να είναι αρκετές φορές γρηγορότερη από την Java. Παρόλα αυτά γίνονται προσπάθειες από τη Sun για τη βελτιστοποίηση της εικονικής μηχανής, ενώ υπάρχουν και άλλες υλοποιήσεις της εικονικής μηχανής από διάφορες εταιρίες (όπως της IBM), οι οποίες μπορεί σε κάποια σημεία να προσφέρουν καλύτερα και σε κάποια άλλα χειρότερα αποτελέσματα. Επιπλέον με την καθιέρωση των μεταγλωττιστών JIT (JustInTime), οι οποίοι μετατρέπουν τον κώδικα byte απευθείας σε γλώσσα μηχανής, η διαφορά ταχύτητας από τη C++ έχει μικρύνει κατά πολύ.

Οι τελευταίες εκδόσεις του javac με τη χρήση της τεχνολογίας HotSpot έχουν καταφέρει αξιόλογες επιδόσεις που πλησιάζουν ή και ξεπερνούν σε μερικές περιπτώσεις τον εγγενή κώδικα.

3.5.2 Java servlet

Ένα Java servlet είναι ένα πρόγραμμα γραμμένο στην γλώσσα προγραμματισμού Java το οποίο επεκτείνει τις δυνατότητες ενός server. Αν και οι server μπορούν να ανταποκρίνονται σε πολλούς τύπους αιτημάτων- απαιτήσεων, μπορούν γενικότερα να υλοποιούν εφαρμογές που φιλοξενούνται σε servers διαδικτυακούς. Τα servlets παρέχουν την δυνατότητα στην java να μπορέσει να υλοποιήσει και να ανταγωνιστεί τις υπόλοιπες γλώσσες προγραμματισμού και διαδικτύου σε τεχνολογίες όπως την PHP και ASP.NET.

3.5.2.1 Εισαγωγή στα Servlets

Τα Servlets είναι ευρέως γνωστά για την χρήση τους σε διάφορους τομείς για τις εξής ικανότητες τους:

Για την επεξεργασία και αποθήκευση μιας Java class σε Java EE η οποία επιβεβαιώνεται και καθορίζεται μέσω λειτουργιών και μεθόδων που παρέχονται από το εκάστωτε Java Servlet API που αποτελεί μια στάνταρ αλληλουχία διαδικασιών για την υλοποίηση Java κλάσεων που απαιτείται να ανταποκρίνονται σε αιτήματα. Τα Servlets μπορούν κατά κόρων να επικοινωνούν πάνω σε πρωτόκολλα client-server, αλλά και να χρησιμοποιούνται ευρέως σε HTTP πρωτόκολλα. Επομένως, εν συντομία ένα "servlet" χρησιμοποιείται σαν μια συντόμευση ονόματος για το "HTTP servlet". Κατά συνέπεια, ένας προγραμματιστής λογισμικού μπορεί να χρησιμοποιήσει ένα servlet για την προσθήκη δυναμικού περιεχομένου σε ένα server διαδικτύου που χρησιμοποιεί και προγραμματίζει σε Java platform. Το παραγόμενο περιεχόμενο που επιστρέφει είναι κατά κύριο λόγο HTML αλλά μπορεί να δώσει και αποτελέσματα δεδομένων και σε άλλες μορφές όπως είναι σε XML. Τα Servlets μπορούν να διατηρούν την κατάσταση σε επίπεδο session μεταβλητών καθώς εξελίσσονται συναλλαγές – δοσοληψίες μεταξύ των server με την χρήση HTTP cookies, η επαναπληκτρολόγιση διευθύνσεων.

Η χρήση ενός διαδικτυακού container είναι απαραίτητο για το στήσιμο, τον προγραμματισμό, τον σχεδιασμό και το τρέξιμο ενός servlet. Το διαδικτυακό container (γνωστό και ως servlet container) είναι ουσιαστικά ένα εξάρτημα- τμήμα του διαδικτυακού server που αλληλεπιδρά με τα servlets και είναι υπεύθυνο για να διαχειριστεί τον κύκλο ζωής των servlets καθώς και να χαρτογραφήσει ένα URL σε ένα συγκεκριμένο servelt και να έχει την δυνατότητα να επιβεβαιώσει στον server ότι το αίτημα για το εκάστοτε URL έχει πρόσβαση με σωστά και επαληθευμένα δικαιώματα.

Το Servlet API περιέχεται ως ένα επιπλέον package της Java με ιεραρχία javax.servlet, το οποίο ορίζει τις απαιτούμενες μεθόδους για την δυνατότητα υλοποίησης σωστών αλληλεπιδράσεων μεταξύ των διαδικτυακών container και των servlet.

Ένα Servlet είναι ένα αντικείμενο το οποίο λαμβάνει αιτήματα και αποστέλλει απαντήσεις βάση του αιτήματος που έλαβε. Το βασικό Servlet package ορίζει αντικείμενα της Java που αναπαριστούν τα αιτήματα και τις απαντήσεις των servlet με τέτοιο τρόπο ώστε να ανταποκρίνονται και να αναπαριστούν σωστές παραμετροποιήσεις σε περιβάλλοντα εκτέλεσης. Το pakage javax.servlet.http ορίζει συγκεκριμένες υποκλάσεις του HTTP που παράγουν στοιχεία τύπου generic servlet, τα οποία περιλαμβάνουν αντικείμενα διαχείρισης session που ανιχνεύουν πολλαπλές αιτήσεις και απαντήσεις μεταξύ server διαδικτύου και client. Τα Servlets μπορούν να πακεταριστούν σε ένα WAR αρχείο ως μια εφαρμογή διαδικτύου

Τα servlets μπορούν να παραχθούν αυτόματα από τις σελίδες τύπου server της Java, τα λεγόμενα Java Server Pages (JSP) από τον JavaServer Pages compiler. Η διαφορά μεταξύ servlets και JSP είναι ότι τυπικά τα servlets εμπεριέχουν HTML μέσα σε κώδικα Java, ενώ τα JSPs εμπεριέχουν κώδικα Java code σε HTML. Ενώ η άμεση χρησιμότητα των servlets να παράγουν HTML μπορεί να δημιουργείται σπάνια, το πιο υψηλό επίπεδο MVC web framework σε Java EE (JSF) ακόμα χρησιμοποίει την τεχνολογία των servlet για χαμηλού επιπέδου αιτήματα και απαιτήσεις που χειρίζονται μέσω FacesServlet. Μια παλιότερη χρησιμότητα είναι η χρήση των servlets σε συνδυασμό με JSPs σε ένα μοτίβο γνωστό και ως "Model 2", το οποίο αποτελεί μια μικρή γεύση του model–view–controller.

3.5.2.2 Ιστορία

Η δημιουργία εξειδίκευσης των Servlet δημιουργήθηκε από την SUN Microsystems, με την έκδοση 1.0 και οριστικοποιήθηκε τον Ιούνιο του 1997. Ξεκινώντας με την έκδοση 2.3, εξειδίκευση αναπτύχθηκε κάτω από την επίβλεψη της Κοινότητα Επεξεργασίας της Java. Η JSR 53 ορίστηκε από τις εξειδικεύσεις τόσο των Servlet 2.3 όσο και των JavaServer Page 1.2 εξειδικεύσεων. Η έκδοση JSR 154 εξειδικεύεται στα Servlet 2.4 και 2.5 εξειδικεύσεις. Η τρέχουσα έκδοση των servlets τρέχει από τον Μάρτιο του 2010 ως Servlet εξειδίκευση 3.0.

Ο Jim Driscoll ήταν ο πρώτος που σκέφτηκε και ανέλυσε λεπτομερώς την τεχνολογία των servlet. Ο James Gosling ήταν αυτός που προσπάθησε να προωθήσει και να υλοποιήσει σε πράξη την θεωρία των servlets αλλά αυτό το κατάφερε και το έκδωσε σαν προϊόν στην αγορά η Sun ως Java Web Server

Servlet API history

Servlet API version	Released	Platform	Important Changes
Servlet 3.1	May 2013	JavaEE 7	Non-blocking I/O, HTTP protocol upgrade mechanism (WebSocket)[5]
Servlet 3.0	December 2009	JavaEE 6, JavaSE 6	Pluggability, Ease of development, Async Servlet, Security, File Uploading
Servlet 2.5	September 2005	JavaEE 5, JavaSE 5	Requires JavaSE 5, supports annotation
Servlet 2.4	November 2003	J2EE 1.4, J2SE 1.3	web.xml uses XML Schema
Servlet 2.3	August 2001	J2EE 1.3, J2SE 1.2	Addition of `Filter`
Servlet 2.2	August 1999	J2EE 1.2, J2SE 1.2	Becomes part of J2EE, introduced independent web applications in .war files
Servlet 2.1	November 1998	Unspecified	First official specification, added `RequestDispatcher`, `ServletContext`
Servlet 2.0		JDK 1.1	Part of Java Servlet Development Kit 2.0
Servlet 1.0	June 1997		

3.5.2.3 Servlets σε σύγκριση με άλλα μοντέλα εφαρμογών διαδικτύου

Τα πλεονεκτήματα χρήσης των Servlets είναι η γρήγορη επίδοση τους και η ευκολία χρήσης σε συνδυασμό με περισσότερη δυναμική πέρα από τις παραδοσιακές CGI διεπαφές (Common Gateway Interface) τους. Παραδοσιακά σενάρια CGI γραμμένα σε Java δυστυχώς έχουν έχει μια σειρά από μειονεκτήματα απόδοσης που είναι τα ακόλουθα:

Όταν ένα αίτημα HTTP πραγματοποιηθεί, μια νέα διαδικασία δημιουργείται κάθε φορά που το σενάριο CGI καλείται. Η κλήση του σχετίζεται με τη δημιουργία της διαδικασίας που μπορεί να δημιουργήσει καθυστερήσεις στον φόρτο εργασίας και δυνατοτήτων της εφαρμογής, ιδίως όταν το σενάριο κάνει σχετικά γρήγορα πράξεις. Επομένως, η δημιουργία της διαδικασίας θα χρειαστεί περισσότερος χρόνος για την εκτέλεση του script CGI παρά την σχετική απόδοση και λειτουργικότητα που μπορεί να παρέχει. Αντίθετα, για τα servlets, κάθε αίτηση γίνεται από ένα ξεχωριστό νήμα Java στο πλαίσιο της διαδικασίας του web server, αποφεύγοντας έτσι την επιβάρυνση που συνδέεται με την δημιουργία εικονικών διεργασιών πανομοιότυπες με την ίδια μέσα στον δαίμονα HTTP.

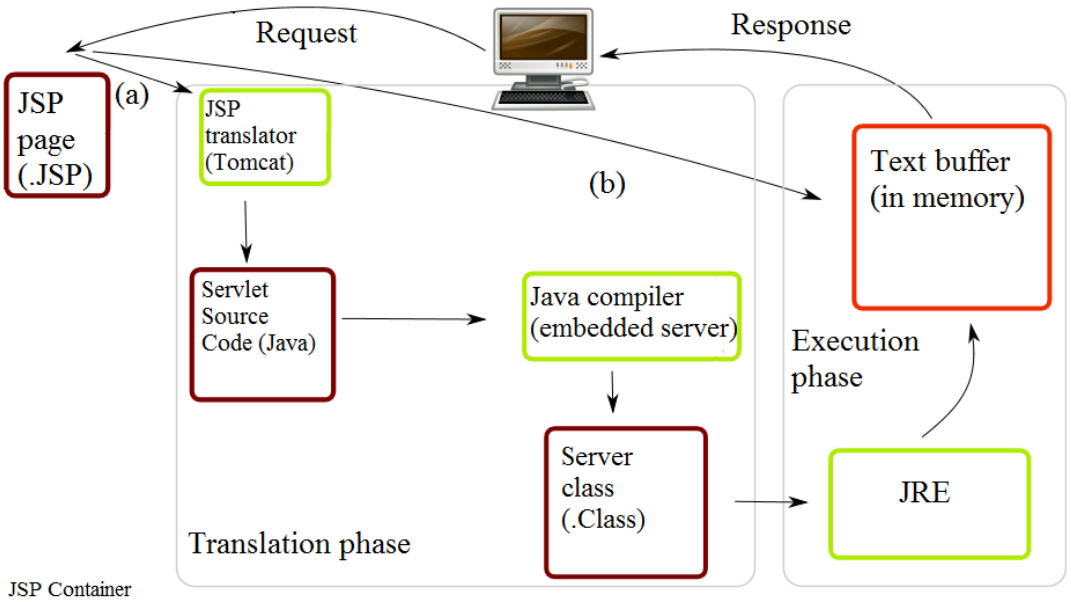
Η διαδικασία ταυτόχρονης αίτησης CGI είναι χρονοβόρα και βαριά καθώς θα πρέπει να φορτώσει το σενάριο CGI που θα αντιγραφεί στη μνήμη μία φορά ανά αίτηση ενώ με τα servlets, υπάρχει μόνο ένα αντίγραφο, που επιμένει σε όλη αιτήματα και μοιράζεται μεταξύ των νημάτων που ενεργούν για την ολοκλήρωση των αιτημάτων προς απάντηση από τον server διαδικτύου.

Η αναπαράσταση του αιτήματος με ένα μόνο στιγμιότυπο δίνει την δυνατότατα απάντησης σε όλα τα αιτήματα ταυτόχρονα. Αυτό έχει ως αποτέλεσμα να μειώνει τη χρήση της μνήμης και διευκολύνει τη διαχείριση των δεδομένων.

Ένα servlet μπορεί να εκτελεστεί από ένα servlet container σε ένα περιοριστικό περιβάλλον, που ονομάζεται sandbox. Αυτό είναι παρόμοιο με ένα applet που τρέχει στο sandbox του web browser. Αυτό επιτρέπει περιορισμένη χρήση των δυνητικά επιβλαβών servlets. Τα CGI προγράμματα μπορούν από μόνα τους να αποτελέσουν sandbox και τα ίδια, δεδομένου ότι είναι απλά OS διαδικασίες.

Τεχνολογίες όπως η FastCGI και τα παράγωγά του (συμπεριλαμβανομένης της SCGI, WSGI) δεν παρουσιάζουν τα μειονεκτήματα των επιδόσεων των CGI που προκύπτουν από τη συνεχή επεξεργασία. Είναι, ωστόσο, κατά προσέγγιση τόσο απλό όπως τα CGI. Επομένως, είναι σε μόνιμη αντίθεση με τα servlets τα οποία είναι ουσιαστικά πιο περίπλοκα.

3.5.2.4 Ο κύκλος ζωής ενός servlet



(a) Translation occurs at this point, if JSP has been changed or is new.
(b) If not, translation is skipped.

Τρεις μέθοδοι παίζουν καθοριστικό ρόλο για τη διάρκεια του κύκλου ζωής ενός servlet. Αυτές είναι η init(), service(), και destroy(). Χρησιμοποιούνται σε κάθε εφαρμογή από ένα servlet και γίνεται επίκληση αυτών των μεθόδων σε συγκεκριμένες ώρες από το διακομιστή - server.

Κατά το πρώτο στάδιο αρχικοποίησης του κύκλου ζωής ενός servlet, το container διαδικτύου προετοιμάζει ένα στιγμιότυπο του servlet, καλώντας τη μέθοδο init (), περνώντας ένα αντικείμενο που υλοποιεί το javax.servlet.ServletConfiginterface. Αυτό το αντικείμενο επιτρέπει την διαμόρφωση του servlet για να αποκτήσει πρόσβαση σε παραμέτρους αρχικοποίησης από την διαδικτυακή εφαρμογή.

Μετά την αρχικοποίηση , το στιγμιότυπου του servlet μπορεί να εξυπηρετήσει τα αιτήματα των πελατών - client. Κάθε αίτηση εξυπηρετείται στο δικό της ξεχωριστό νήμα. Το web container καλεί τη μέθοδο service () του servlet για κάθε αίτηση. Η μέθοδος service () προσδιορίζει το είδος του αιτήματος που γίνεται και τις αποστολές του με ένα κατάλληλο τρόπο έτσι ώστε να γίνει η σωστή διαχείριση της αίτησης. Ο προγραμματιστής που γράφει το servlet πρέπει να παρέχει μια υλοποίηση υποστήριξης αυτών των μεθόδων. Αν μια αίτηση γίνεται για μια μέθοδο που δεν υλοποιείται από το servlet, τυπικά η μέθοδος της γονικής κλάσης αναλαμβάνει ώστε ως αποτέλεσμα να επιστραφεί ένα σφάλμα στον αιτούντα και ο προγραμματιστής να μεριμνήσει για αυτό σχετικά με το error που παρουσιάστηκε.

Τέλος, το container διαδικτύου καλεί τη μέθοδο destroy () που θέτει το servlet εκτός λειτουργίας μετά την ολοκλήρωση του αιτήματος που είχε να ασχοληθεί. Η μέθοδος destroy (), όπως και η init (), καλείται μόνο μία φορά κατά τη διάρκεια του κύκλου ζωής ενός servlet.

Το ακόλουθο είναι ένα τυπικό σενάριο χρήσης των μεθόδων αυτών.

1. Ας υποθέσουμε ότι ένας χρήστης ζητά να επισκεφθεί ένα URL.

   a. Το πρόγραμμα περιήγησης στη συνέχεια, δημιουργεί μια αίτηση HTTP για αυτό το URL.
   b. Το αίτημα αυτό αποστέλλεται στη συνέχεια στο κατάλληλο διακομιστή.
2. Η αίτηση HTTP λαμβάνεται από τον server διαδικτύου και προωθείται στο container με τα Servlet.
   a. Το container αντιστοιχίζει αυτό το αίτημα σε ένα συγκεκριμένο servlet ανάλογα με το αίτημα που ζητήθηκε.
   b. Το servlet δυναμικά ανακτάται και τοποθετείται στον ανάλογο χώρο διευθύνσεων του conatiner.
3. Το container επικαλείται τη μέθοδος init () του servlet.
   a. Η μέθοδος αυτή επικαλείται μόνο όταν το servlet παραφορτωθεί στη μνήμη.
   b. Υπάρχει η περίπτωση φόρτωσης ορισμάτων-παραμέτρων κατά την αρχικοποίηση του Servlet που του παρέχει την δυνατότητα να μπορεί να αυτορυθμιστεί.
4. Το δοχείο επικαλείται τη μέθοδο service () του servlet.
   a. Αυτή η μέθοδος καλείται για να επεξεργαστεί την αίτηση HTTP.
   b. Το servlet μπορεί να διαβάσει τα δεδομένα που έχουν παραχθεί στην αίτηση HTTP.
   c. Το servlet μπορεί επίσης να διατυπώνει μια απόκριση HTTP για τον πελάτη.
5. Το servlet παραμένει στο χώρο διευθύνσεων του container και είναι διαθέσιμο για την επεξεργασία οποιωνδήποτε άλλων αιτήσεων HTTP που λαμβάνονται από τους πελάτες.
   a. Η μέθοδος service () καλείται για κάθε αίτηση HTTP.
6. Το δοχείο μπορεί, οποιαδήποτε χρονική στιγμή επιθυμεί, να αποφασίσει να ξεφορτώσει το servlet από τη μνήμη του.
   a. Οι αλγόριθμοι με τους οποίους γίνεται η παρούσα απόφαση είναι ειδικές για κάθε δοχείο.
7. Το δοχείο καλεί τη μέθοδο του servlet destroy () ώστε να παραιτηθεί από τυχόν πόρους, όπως χειρισμού αρχείων που διατίθενται για την servlet
8. Η μνήμη που διατίθεται για το Servlet και τα αντικείμενά του μπορεί στη συνέχεια αυτοκαταστραφούν

3.5.3 JDBC

Το JDBC είναι μια τεχνολογία Java για την σύνδεση με βάσεις δεδομένων από την Oracle Corporation. Αυτή η τεχνολογία είναι ένα API για τη γλώσσα προγραμματισμού Java που ορίζει τον τρόπο με τον οποίο ο πελάτης μπορεί να έχει πρόσβαση σε μια βάση δεδομένων . Παρέχει μεθόδους για την αναζήτηση και την ενημέρωση των δεδομένων σε μια βάση δεδομένων . JDBC συγκεκριμενοποιείται σε σχεσιακές βάσεις δεδομένων . Μια γέφυρα JDBC -προς- ODBC επιτρέπει συνδέσεις σε οποιαδήποτε πηγή δεδομένων ODBC προσβάλαμε στο περιβάλλον υποδοχής JVM.

3.5.3.1 Ιστορία και υλοποίηση

Η Sun Microsystems κυκλοφόρησε την JDBC ως μέρος του JDK 1.1 στις 19 Φεβρουαρίου , 1997. Από τότε είναι μέρος της Java Standard Edition .

Οι JDBC κλάσεις περιέχονται στο pakage java.sql και javax.sql Java .

Ξεκινώντας με την έκδοση 3.1 ,η JDBC έχει αναπτυχθεί στο πλαίσιο της κοινότητας Επεξεργασίας της Java . JSR 54 εξειδικεύεται στην JDBC 3.0 (που περιλαμβάνεται στο J2SE 1.4) , JSR 114 καθορίζει τις προσθήκες των JDBC rowset , και η JSR 221 αποτελεί τον προσδιορισμός του JDBC 4,0 (συμπεριλαμβανομένου της Java SE 6)

Η JDBC 4.1 , προσδιορίζεται με δελτίο συντήρησης 1 του JSR 221 και περιλαμβάνεται στην Java SE 7.

Η τελευταία έκδοση , JDBC 4.2 , προσδιορίζεται με δελτίο συντήρησης 2 του JSR 221 [5] και περιλαμβάνεται στην Java SE 8. [6]

3.5.3.2 Λειτουργικότητα

JDBC επιτρέπει σε πολλαπλές υλοποιήσεις να υπάρχουν και να χρησιμοποιούνται από την ίδια εφαρμογή. Το API παρέχει ένα μηχανισμό για τη δυναμική φόρτωση των σωστών πακέτων Java και την εγγραφή τους με το πρόγραμμα JDBC Driver Manager. Ο Manager Driver χρησιμοποιείται ως ένα εργοστάσιο σύνδεσης για τη δημιουργία JDBC συνδέσεις.

Οι JDBC συνδέσεις υποστηρίζουν τη δημιουργία και την εκτέλεση δηλώσεων. Αυτά μπορεί να είναι δηλώσεις όπως SQL CREATE, INSERT, UPDATE και DELETE, ή μπορεί να είναι δηλώσεις ερώτημα, όπως SELECT. Επιπλέον, αποθηκευμένες διαδικασίες μπορεί να επανακληθουν μιας σύνδεσης JDBC. Η JDBC αντιπροσωπεύει δηλώσεις που χρησιμοποιούν τις ακόλουθες κλάσεις:

• Statement - η δήλωση αποστέλλεται στον διακομιστή της βάσης δεδομένων κάθε φορά.

• PreparedStatement - η δήλωση είναι προσωρινά αποθηκευμένη και στη συνέχεια η πορεία εκτέλεσης της προκαθορίζεται στο διακομιστή της βάσης δεδομένων που επιτρέπει να εκτελεστεί πολλές φορές με αποτελεσματικό τρόπο.

• CallableStatement - που χρησιμοποιούνται για την εκτέλεση αποθηκευμένων διαδικασιών στη βάση δεδομένων.

Οι Δηλώσεις Ενημέρωσης όπως INSERT, UPDATE και DELETE επιστρέφουν ένα καταμετρητή ενημερωμένη έκδοσης που δείχνει πόσες γραμμές επιλέγηκαν στη βάση δεδομένων. Αυτές οι δηλώσεις δεν επιστρέφουν οποιαδήποτε άλλη πληροφορία.

Οι Δηλώσεις ερωτημάτων QUERY επιστρέφουν ένα σύνολο αποτελεσμάτων στην JDBC. Οι μεμονωμένες στήλες σε μια σειρά ανακτώνται είτε με βάση το όνομα ή τον αριθμό της στήλης. Μπορεί να υπάρχει οποιοσδήποτε αριθμός των γραμμών στο σύνολο αποτελεσμάτων. Το σύνολο των αποτελεσμάτων έχει τα μεταδεδομένα που περιγράφουν τα ονόματα των στηλών και τους τύπους τους.

Υπάρχει μια επέκταση του βασικού JDBC API στην javax.sql.

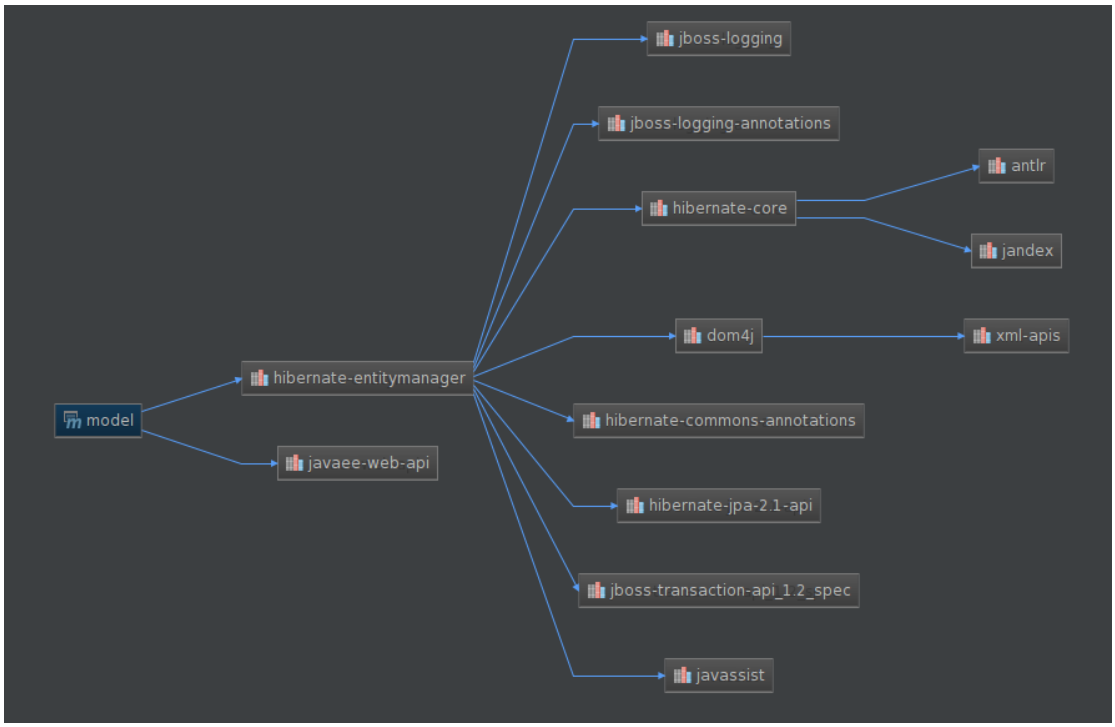
4 Δομή project

4.1 Ρυθμίσεις maven

Το project χρησιμοποιεί το εργαλείο maven για την διαδικασία συγκέντρωσης όλων των απαραίτητων dependencies και το compile. Αυτή η διαδικασία θα ήταν εξαιρετικά δύσκολη χωρίς το maven για δύο λόγους.

Αφ' ενός κάνουμε χρήση ενός αρκετά μεγάλου αριθμού βιβλιοθηκών, κάθε μία από τις οποίες απαιτεί επιπλέον βιβλιοθήκες. Με την χρήση του maven αρκεί να ορίσουμε τις βιβλιοθήκες που χρησιμοποιούμε με μία συγκεκριμένη δομή και αυτό αναλαμβάνει αυτόματα να συγκεντρώσει τα dependencies κάθε μίας από αυτές (τα οποία πολλές φορές έχουν με τη σειρά τους επιπλέον dependencies κοκ).

Παρακάτω φαίνονται τα dependencies για ένα από τα project που συνθέτουν την εφαρμογή, το οποίο είναι και το πιο περιορισμένο σε έκταση και πλήθος dependencies.



Το παραπάνω δέντρο προέκυψε με τις δύο παρακάτω γραμμές στο αρχείο pom.xml του project:

```
<dependencies>
    <dependency>
        <groupId>org.hibernate</groupId>
        <artifactId>hibernate-entitymanager</artifactId>
    </dependency>
    <dependency>
        <groupId>javax</groupId>
        <artifactId>javaee-web-api</artifactId>
        <version>7.0</version>
    </dependency>
</dependencies>
```

Όπως φαίνεται παραπάνω έχουμε ζητήσει μόνο τις δύο βιβλιοθήκες και το maven αυτόματα έχει συγκεντρώσει τα dependencies τους. Επιπλέον για την πρώτη βιβλιοθήκη δεν έχουμε ορίσει κάποια έκδοση σε αυτό το project, γιατί την έχουμε ορίσει κεντρικά στον πατέρα του. Έτσι δεν έχουμε προβλήματα επειδή μπορεί να συμπεριλαμβάνουμε τις ίδιες βιβλιοθήκες με διαφορετικές εκδόσεις.

Αφ' ετέρου η διαδικασία του compile δεν είναι απλή. Κάποιες από τις βιβλιοθήκες που χρησιμοποιούμε χρειάζεται να υπάρχουν κατά το compile, όμως είναι λάθος να συμπεριλαμβάνονται στο τελικό αρχείο γιατί υπάρχουν ήδη ενσωματωμένες στον server και η ύπαρξη τους δύο φορές θα οδηγούσε σε συγκρούσεις.

Επιπλέον για να δημιουργηθεί το τελικό αρχείο war της εφαρμογής είναι απαραίτητο να αντιγραφούν τα παραγόμενα αρχεία και οι βιβλιοθήκες σε μία συγκεκριμένη ιεραρχία φακέλων, καθώς και να τρέξουν κάποια εξωτερικά εργαλεία για να παράξουν κάποια έξτρα αρχεία που χρειάζονται, κυρίως για το compile του Vaadin. Αυτές τις διαδικασίες τις αναλαμβάνουν αυτόματα κάποια maven plugins που έχουμε ρυθμίσει.

```
<plugin>
	<groupId>com.vaadin</groupId>
	<artifactId>vaadin-maven-plugin</artifactId>
	<version>${vaadin.plugin.version}</version>
	<configuration>
		<extraJvmArgs>-Xmx512M -Xss1024k</extraJvmArgs>
	<webappDirectory>${basedir}/target/classes/VAADIN/widgetsets</webapp
Directory>
		<draftCompile>false</draftCompile>
		<compileReport>false</compileReport>
		<style>OBF</style>
		<strict>true</strict>
	</configuration>
	<executions>
		<execution>
			<goals>
				<goal>compile</goal>
			</goals>
		</execution>
	</executions>
</plugin>
```

Για παράδειγμα το παραπάνω plugin αναλαμβάνει το compile του Vaadin.

4.2 Maven modules

Το project αποτελείται από ένα βασικό maven project, το οποίο περιέχει δύο maven modules.

Το αρχείο ρυθμίσεων pom.xml του βασικού project περιέχει κάποιες ρυθμίσεις που είναι κοινές για όλα τα project, όπως τις εκδόσεις των dependencies που χρησιμοποιούνται.

Για να αποφύγουμε να έχουμε όλα τα αρχεία σε ένα πολύ μεγάλο project έχουμε κάνει έναν βασικό διαχωρισμό σε δύο επιμέρους modules, που χρησιμοποιούν τις ρυθμίσεις του βασικού project.

```
<groupId>gr.eshop7</groupId>
<artifactId>eshop-parent</artifactId>
<packaging>pom</packaging>
```

```
<version>1.0-SNAPSHOT</version>
<modules>
	<module>eshop</module>
	<module>model</module>
</modules>
```

Το πρώτο module είναι το model, το οποίο περιέχει τα αρχεία που περιγράφουν την δομή της βάσης και το δεύτερο είναι το eshop, το οποίο περιέχει τα αρχεία για την λειτουργία του site και του admin. Και τα δύο αναφέρονται στο βασικό project, για παράδειγμα το module model έχει την παρακάτω δομή:

```
<parent>
	<artifactId>eshop-parent</artifactId>
	<groupId>gr.eshop7</groupId>
	<version>1.0-SNAPSHOT</version>
</parent>
<modelVersion>4.0.0</modelVersion>
<artifactId>model</artifactId>
```

Εναλλακτικά θα μπορούσαμε να είχαμε επιπλέον σπάσει το module eshop σε τρία project, ένα για το site, ένα για το admin και ένα τρίτο που θα περιείχε τα κοινά τους στοιχεία. Αυτό θα έδινε την δυνατότητα να δημιουργούμε αρχεία war που θα περιείχαν μόνο το site ή μόνο το admin, όμως αυτή η δυνατότητα δεν ήταν χρήσιμη για την εφαρμογή μας.

4.3 Ρυθμίσεις βάσης δεδομένων

Για την επικοινωνία με την βάση δεδομένων χρησιμοποιήσαμε το πρότυπο JPA, το οποίο μας απάλλαξε από την χρήση της γλώσσας SQL. Σύμφωνα με τον τρόπο εργασίας που επιβάλλεται από το JPA δημιουργούμε αντικείμενα Java τα οποία έχουν συγκεκριμένα annotations που περιγράφουν πώς αποθηκεύονται στην βάση.

Αυτή η προσέγγιση έχει το πλεονέκτημα ότι η εφαρμογή μπορεί να μεταφερθεί σε διαφορετικές βάσης δεδομένων και να λειτουργεί με τον ίδιο τρόπο. Ακόμη μπορεί να γίνει χρήση των οντοκεντρικών δυνατοτήτων που προσφέρει η java για την αναπαράσταση διάφορων αντικειμένων, χωρίς να επαναλαμβάνονται τα ίδια στοιχεία σε πολλά παρόμοια αντικείμενα.

Παρακάτω φαίνεται ένα απόσπασμα από το αντικείμενο IdObject, το οποίο περιέχει μόνο ένα id, δηλαδή το μοναδικό αναγνωριστικό των αντικειμένων.

```
@MappedSuperclass
public class IdObject implements Serializable {
	@Id
	@GeneratedValue(strategy= GenerationType.IDENTITY)
	private long id;

	@Override
	public boolean equals(Object o) {
		if (this == o) return true;
		if (!(o instanceof IdObject)) return false;

		IdObject idObject = (IdObject) o;

		return getId() == idObject.getId();
	}
```

```
	@Override
	public int hashCode() {
		return (int) (getId() ^ (getId() >>> 32));
	}
}
```

Περιλαμβάνει έναν ακέραιο id, το οποίο είναι μοναδικό αναγνωριστικό κάθε οντότητας και έχει το annotation @Id. Το annotation @GeneratedValue λέει στο JPA ότι το id πρέπει να δημιουργείται αυτόματα και το GenerationType.IDENTITY λέει ότι η δημιουργία του θα γίνεται από την βάση δεδομένων.

Η παραπάνω οντότητα δεν αντιστοιχεί σε έναν πίνακα αλλά κληρονομείται από άλλες οντότητες οι οποίες έχουν τον δικό τους πίνακα, οπότε έχει το annotation @MappedSuperclass.

Οι μέθοδοι equals και hashCode χρησιμοποιούνται από την Java για να ελέγξει αν δύο αντικείμενα είναι όμοια. Τυπικά, όταν δεν δίνονται, δύο οντότητες θεωρούνται όμοιες όταν έχουν την ίδια κλάση και όλες οι μεταβλητές τους έχουν την ίδια τιμή. Στην περίπτωσή μας, καθώς αναφερόμαστε σε αντικείμενα που αποθηκεύονται σε μία βάση δεδομένων δύο αντικείμενα συμπίπτουν όταν έχουν το ίδιο id, οπότε γράψαμε αυτές τις μεθόδους ώστε να λαμβάνουν υπόψη τους μόνο αυτό το id. Αυτή η λειτουργικότητα θα υπάρχει και σε όλες τις οντότητες που θα κληρονομήσουν το IdObject.

Μία οντότητα που το κληρονομεί είναι το CustomerOrder, που αναπαριστά μία παραγγελία. Σε αυτό θα μπορούσαμε να είχαμε χρησιμοποιεί σαν αναγνωριστικό το σύνθετο κλειδί που προκύπτει από τον πελάτη, το προϊόν και την ημερομηνία της παραγγελίας, όμως προτιμήσαμε να χρησιμοποιήσουμε σαν αναγνωριστικό έναν ακέραιο γιατί απλοποιούσε αρκετά τον κώδικα και γιατί ενδείκνυται για λόγους επιδόσεων.

Παρακάτω φαίνεται ένα απόσπασμα του CustomerOrder, το οποίο δεν περιλαμβάνει όλες τις μεταβλητές του:

```
@Entity
public class CustomerOrder extends IdObject {

	@Enumerated(EnumType.STRING)
	private OrderStatus orderStatus;

	@ManyToOne
	@JoinColumn(name = "customerId")
	private Customer customer;

	@ManyToOne
	@JoinColumn(name = "productId")
	private Product product;

	private float price;
	private String ref;
}
```

Το CustomerOrder έχει το annotation @Entity, το οποίο σημαίνει ότι θα δημιουργηθεί ξεχωριστός πίνακας για αυτό. Αφού κληρονομεί το IdObject θα πάρει από αυτό το πεδίο Id σαν αναγνωριστικό και τις μεθόδους equals και hashCode.

Το πεδίο orderStatus είναι ένα enum:

```
public enum OrderStatus {
	SUBMITTED,
	IN_PROGRESS,
	DELIVERED,
	CANCELLED
}
```

Σε αυτό λέμε να αποθηκεύεται στην βάση σαν string, με το όνομά του. Εναλλακτικά θα μπορούσαμε να ζητήσουμε να αποθηκεύεται με τον αριθμό του κάθε στοιχείου στο enum ή σαν enum της βάσης, όμως με την αποθήκευση σαν string κάνουμε τον πίνακα να διαβάζεται πιο εύκολα και μπορούμε να προσθέσουμε επιπλέον στοιχεία στο enum χωρίς να επηρεάζονται τα δεδομένα που έχουν ήδη αποθηκευτεί.

Η παραγγελία γίνεται από έναν πελάτη, με τον οποίο έχει μία σχέση πολλά προς ένα, που υποδηλώνεται από το annotation @ManyToOne. Αυτό σημαίνει ότι θα υπάρχει ένα ξένο κλειδί σε κάθε παραγγελία που θα ανήκει στον πίνακα Customer, και με το annotation @JoinColumn προσδιορίζουμε το όνομα της στήλης που θα περιέχει το ξένο κλειδί. Επιπλέον κάθε παραγγελία έχει μία παρόμοια σχέση με ένα προϊόν (και πιθανόν με ένα κατάστημα, αν και δεν φαίνεται στο παραπάνω απόσπασμα).

Οι μεταβλητές String ref και float price αυτόματα εισάγονται σαν στήλες στον πίνακα της παραγγελίας με τον αντίστοιχό τύπο.

Μία άλλη οντότητα στην οποία πρέπει να γίνει αναφορά είναι το PublishedObject, που φαίνεται παρακάτω:

```
@MappedSuperclass
public class PublishedObject extends IdObject {

	@Column(columnDefinition = "boolean NOT NULL DEFAULT 0 ")
	private boolean published = true;
}
```

Σε μία κανονική εφαρμογή δεν σβήνονται σχεδόν ποτέ δεδομένα και ένας από τους λόγους είναι ότι αρκετά δεδομένα εξαρτώνται από πολλά άλλα. Για παράδειγμα αν σβηστεί ένα προϊόν τότε όλες οι παραγγελίες που αναφέρονται σε αυτό παύουν να έχουν νόημα.

Για να αντιμετωπίσουμε τέτοιες καταστάσεις γενικά δεν σβήνουμε εγγραφές από την βάση αλλά της κάνουμε ανενεργές. Αυτό το συμβολίζουμε με την οντότητα PublishedObject, την οποία κληρονομούν σχεδόν όλες οι υπόλοιπες οντότητες.

Το PublishedObject περιέχει μία μεταβλητή published, που ανάλογα με το αν είναι true ή false συμβολίζει αν μία οντότητα θα φαίνεται ή όχι στο site. Είναι @MappedSuperclass, οπότε δεν έχει δικό της πίνακα, αλλά όλες οι οντότητες που την κληρονομούν έχουν μία στήλη published, καθώς και την στήλη id από το IdObject (το οποίο κληρονομεί το PublishedObject).

4.3.1 JPA και SQL

Αξίζει να παρατηρήσουμε ότι το JPA δεν αντικαθιστά την γλώσσα SQL. Κάτι τέτοιο δεν θα ήταν δυνατό, καθώς εν τέλει τα δεδομένα αποθηκεύονται σε μία σχεσιακή βάση. Η βασική λειτουργία του είναι να κρύβει την SQL από τον προγραμματιστή, δίνοντας του

επιπλέον την δυνατότητα να χρησιμοποιήσει την οντοκεντρική προσέγγιση της Java σαν επιπλέον εργαλείο στην σχεδίαση την βάσης του.

Όμως εν τέλει για την σωστή σχεδίαση των οντοτήτων που θα αποθηκευτούν στην βάση είναι απαραίτητο ο προγραμματιστής να έχει επίγνωση του πώς τα διαφορετικά πεδία που δημιουργούνται μέσα από τα αντικείμενα θα αναπαρασταθούν τελικά στην βάση. Το γεγονός ότι ένα ξένο κλειδί ορίζεται βάζοντας ένα annotation σε μία μεταβλητή δεν απαλλάσσει τον προγραμματιστή από τις γνώσεις που χρειάζονται για τα ξένα κλειδιά, γιατί κατά την δημιουργία της βάσης θα δημιουργηθούν τα απαραίτητα ξένα κλειδιά και τα σχετικά constrains για αυτά.

4.3.2 Δομή βάσης δεδομένων

Παρακάτω παρουσιάζουμε τους πίνακες στους οποίους αποθηκεύονται τα δεδομένα στην βάση. Θεωρούμε ότι είναι πιο κατανοητό να παρουσιάσουμε την δομή των πινάκων παρά τα αρχεία JPA από τα οποία παράγονται.

4.3.2.1 Πίνακας Category

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**id**	bigint(20)			No	*None*	AUTO_INCREMENT
2	**name**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
3	**published**	tinyint(1)			No	0	

Σε αυτόν τον πίνακα αποθηκεύονται οι κατηγορίες των προϊόντων.

Περιλαμβάνει τα παρακάτω πεδία:

- Id, το μοναδικό αναγνωριστικό της εγγραφής.
- Name, το όνομα της κατηγορίας.
- Published, το αν το στοιχείο είναι δημοσιευμένο ή όχι.

4.3.2.2 Πίνακας Manufacturer

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**id**	bigint(20)			No	*None*	AUTO_INCREMENT
2	**published**	tinyint(1)			No	0	
3	**name**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	

Σε αυτόν τον πίνακα αποθηκεύονται οι κατασκευαστές των προϊόντων.

Περιλαμβάνει τα παρακάτω πεδία:

- Id, το μοναδικό αναγνωριστικό της εγγραφής.
- Name, το όνομα της κατηγορίας.
- Published, το αν το στοιχείο είναι δημοσιευμένο ή όχι.

4.3.2.3 Πίνακας Customer

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**id**	bigint(20)			No	*None*	AUTO_INCREMENT
2	**published**	tinyint(1)			No	0	
3	**address**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
4	**admin**	tinyint(1)			No	0	
5	**email**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
6	**name**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
7	**password**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
8	**phone**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	

Σε αυτόν τον πίνακα αποθηκεύονται οι χρήστες του συστήματος.

Περιλαμβάνει τα παρακάτω πεδία:

- Id, το μοναδικό αναγνωριστικό της εγγραφής.
- Published, το αν το στοιχείο είναι δημοσιευμένο ή όχι.
- Address, την διεύθυνση του χρήστη.
- Admin, το αν ο χρήστης έχει πρόσβαση στο περιβάλλον διαχείρισης.
- Email, η διεύθυνση ηλεκτρονικού ταχυδρομείου του χρήστη.
- Name, το όνομα του χρήστη.
- Password, ο κωδικός του χρήστη.
- Phone, το τηλέφωνο του χρήστη.

Δεν αποθηκεύονται στοιχεία πληρωμής με πιστωτική κάρτα, γιατί στην πραγματικότητα τέτοιου είδους στοιχεί πολύ σπάνια κρατούνται στο σύστημα.

4.3.2.4 Πίνακας Store

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**id**	bigint(20)			No	*None*	AUTO_INCREMENT
2	**name**	text	latin1_swedish_ci		Yes	*NULL*	
3	**address**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
4	**phone**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
5	**published**	tinyint(1)			No	0	

Σε αυτόν τον πίνακα αποθηκεύονται τα στοιχεία των καταστημάτων.

Περιλαμβάνει τα παρακάτω πεδία:

- Id, το μοναδικό αναγνωριστικό της εγγραφής.

- Name, το όνομα του καταστήματος.
- Published, το αν το κατάστημα φαίνεται στο site ή όχι.
- Phone, το τηλέφωνο του καταστήματος.
- Address, η διεύθυνση του καταστήματος.

Δεν βάλαμε στοιχεία διευθυντή γιατί δεν είχαμε σημείο που να τα παρουσιάσουμε στο site.

4.3.2.5 Πίνακας Product

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**id**	bigint(20)			No	*None*	AUTO_INCREMENT
2	**description**	text	latin1_swedish_ci		Yes	*NULL*	
3	**image**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
4	**name**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
5	**price**	float			No	*None*	
6	**categoryId**	bigint(20)			Yes	*NULL*	
7	**published**	tinyint(1)			No	0	
8	**manufacturerId**	bigint(20)			Yes	*NULL*	

Σε αυτόν τον πίνακα αποθηκεύονται τα στοιχεία των προϊόντων.

Περιλαμβάνει τα παρακάτω πεδία:

- Id, το μοναδικό αναγνωριστικό της εγγραφής.
- Name, το όνομα του προϊόντος.
- Description, η περιγραφή του προϊόντος.
- Published, το αν το προϊόν θα φαίνεται στο site ή όχι.
- Image, το όνομα της φωτογραφίας του προϊόντος.
- Price, η τιμή του προϊόντος.
- categoryId, ξένο κλειδί στην κατηγορία στην οποία ανήκει το προϊόν.
- manufacturerId, ξένο κλειδί στον κατασκευαστή του προϊόντος.

4.3.2.6 Πίνακας CustomerOrder

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**id**	bigint(20)			No	*None*	AUTO_INCREMENT
2	**address**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
3	**deliveryDate**	datetime			Yes	*NULL*	
4	**name**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
5	**orderPayment**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
6	**orderShipping**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
7	**orderStatus**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
8	**paymentInfo**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
9	**phone**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
10	**price**	float			No	*None*	
11	**ref**	varchar(255)	latin1_swedish_ci		Yes	*NULL*	
12	**submissionDate**	datetime			Yes	*NULL*	
13	**customerId**	bigint(20)			Yes	*NULL*	
14	**productId**	bigint(20)			Yes	*NULL*	
15	**storeId**	bigint(20)			Yes	*NULL*	

Σε αυτόν τον πίνακα αποθηκεύονται τα στοιχεία των παραγγελιών.

Περιλαμβάνει τα παρακάτω πεδία:

- Id, το μοναδικό αναγνωριστικό της εγγραφής.
- Address, η διεύθυνση που θα παραδοθεί η παραγγελία αν γίνει αποστολή με κούριερ.
- Name, το όνομα στο οποίο θα αποσταλεί η παραγγελία.
- Phone, το τηλέφωνο του παραλήπτη της παραγγελίας.
- Price, η τιμή στην οποία αγοράστηκε το προϊόν. Κρατείται επιπλέον του προϊόντος γιατί η τιμή του προϊόντος μπορεί να αλλάξει.
- Ref, ένας τυχαίος κωδικός που παράγεται για κάθε παραγγελία.
- SubmissionDate, η ημερομηνία που έγινε η παραγγελία.
- DeliveryDate, η ημερομηνία που παραδόθηκε η παραγγελία.
- PaymentInfo, στοιχεία σχετικά με τον τρόπο πληρωμής, τα στοιχεία της πιστωτικής κάρτας αν η παραγγελία έχει γίνει με αυτόν τον τρόπο.
- OrderPayment, αν η παραγγελία θα πληρωθεί με μετρητά ή με πιστωτική.
- OrderShipping, αν η παρραγελία θα αποσταλεί με κούριερ ή θα παραληφθεί από το κατάστημα.
- OrderStatus, η κατάσταση της παραγγελίας.
- customerId, ξένο κλειδί στον χρήστη που έκανε την παραγγελία.
- productId, ξένο κλειδί στο προϊόν που παραγγέλθηκε.
- storeId, ξένο κλειδί στο κατάστημα από το οποίο θα παραληφθεί η παραγγελία αν δεν αποσταλεί με κούριερ.

4.3.2.7 Πίνακας Stock

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**productId**	bigint(20)			No	0	
2	**storeId**	bigint(20)			No	0	
3	**stock**	int(11)			Yes	*NULL*	

Σε αυτόν τον πίνακα αποθηκεύεται το στοκ που υπάρχει για κάθε προϊόν σε κάθε κατάστημα. Υπάρχει μία εγγραφή ανά προϊόν ανά κατάστημα.

Ο πίνακας έχει ένα σύνθετο κλειδί που αποτελείται από το προϊόν και το κατάστημα στα οποία ανήκει η εγγραφή.

Περιλαμβάνει τα παρακάτω πεδία:

- productId, ξένο κλειδί, το κλειδί του προϊόντος.
- storeId, ξένο κλειδί, το κλειδί του καταστήματος.
- Stock, το πλήθος των τεμαχίων του συγκεκριμένου προϊόντος που υπάρχουν στο συγκεκριμένο κατάστημα.

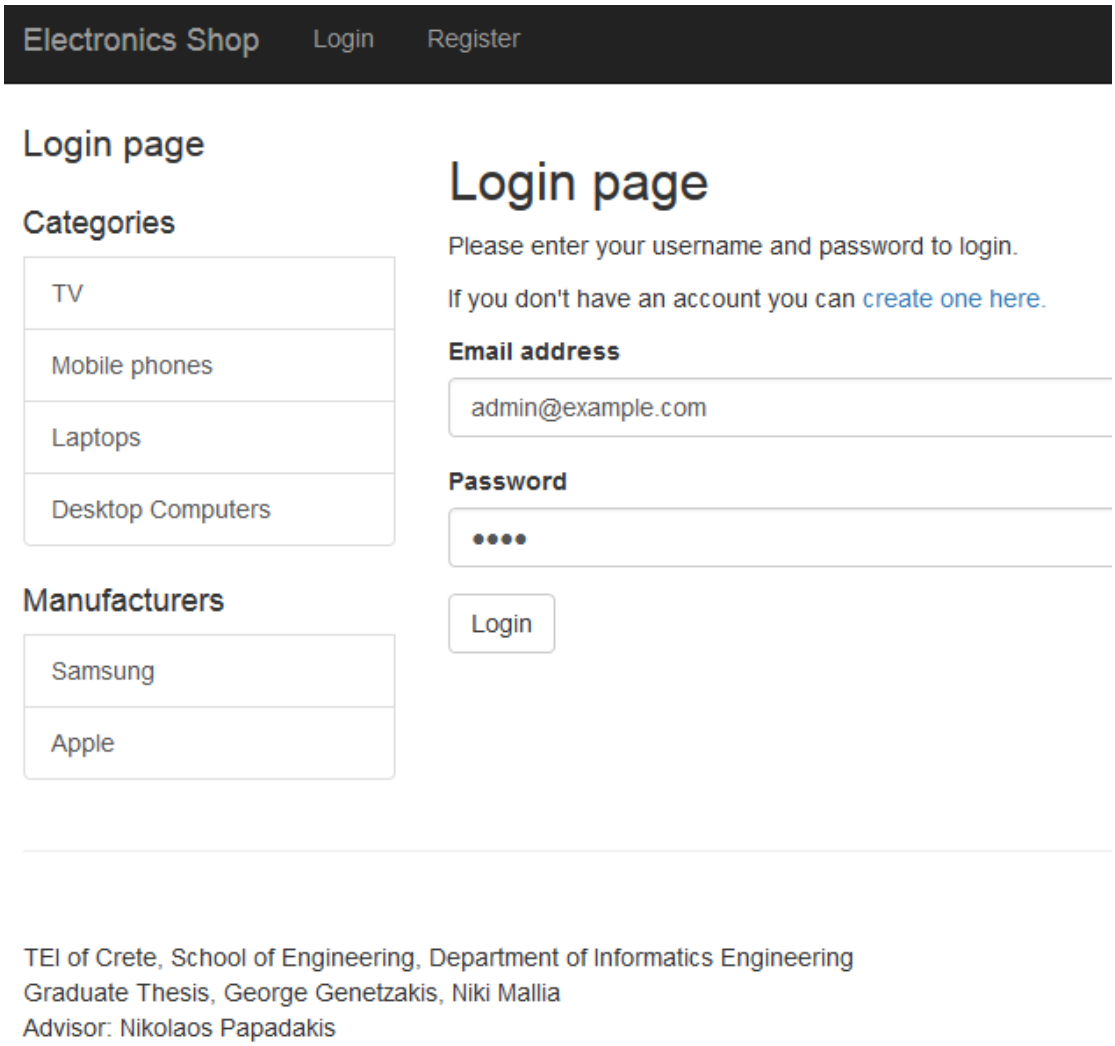
5 Οδηγίες χρήσης

5.1 Περιοχή διαχειριστή

Παρακάτω θα παρουσιάσουμε τις οδηγίες χρήσης για τον διαχειριστή της εφαρμογής.

5.1.1 Είσοδος

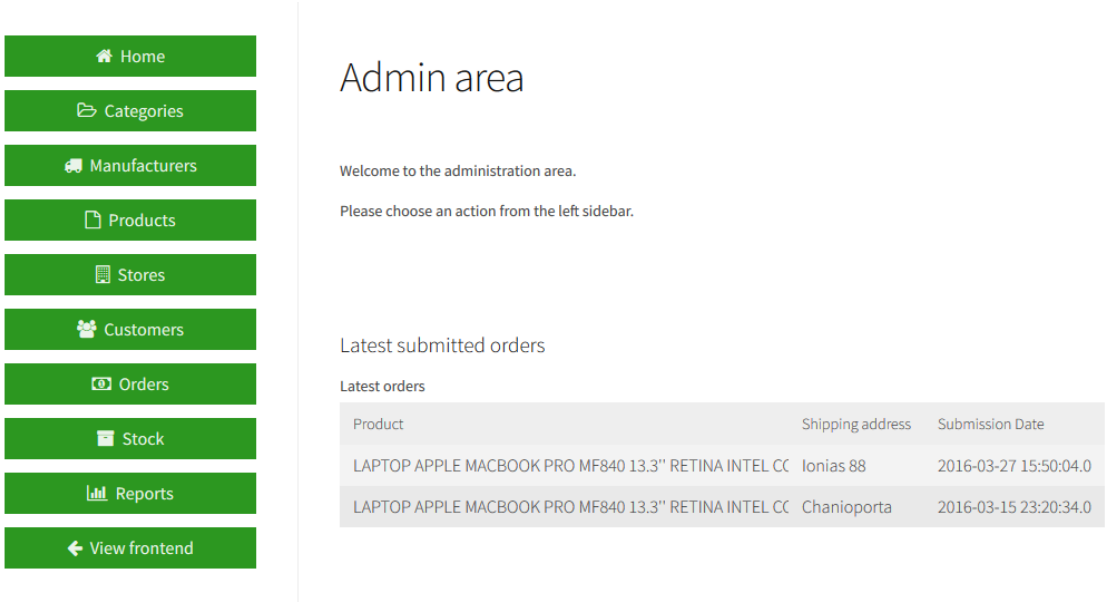
Ο διαχειριστής μπαίνει στο σύστημα από την φόρμα που μπαίνουν και οι απλή χρήστες, συμπληρώνοντας τα στοιχεία του.



Το σύστημα αναγνωρίζει ότι πρόκειται για λογαριασμό διαχειριστή και τον προωθεί στη περιοχή διαχείρισης.

5.1.2 Κεντρική σελίδα

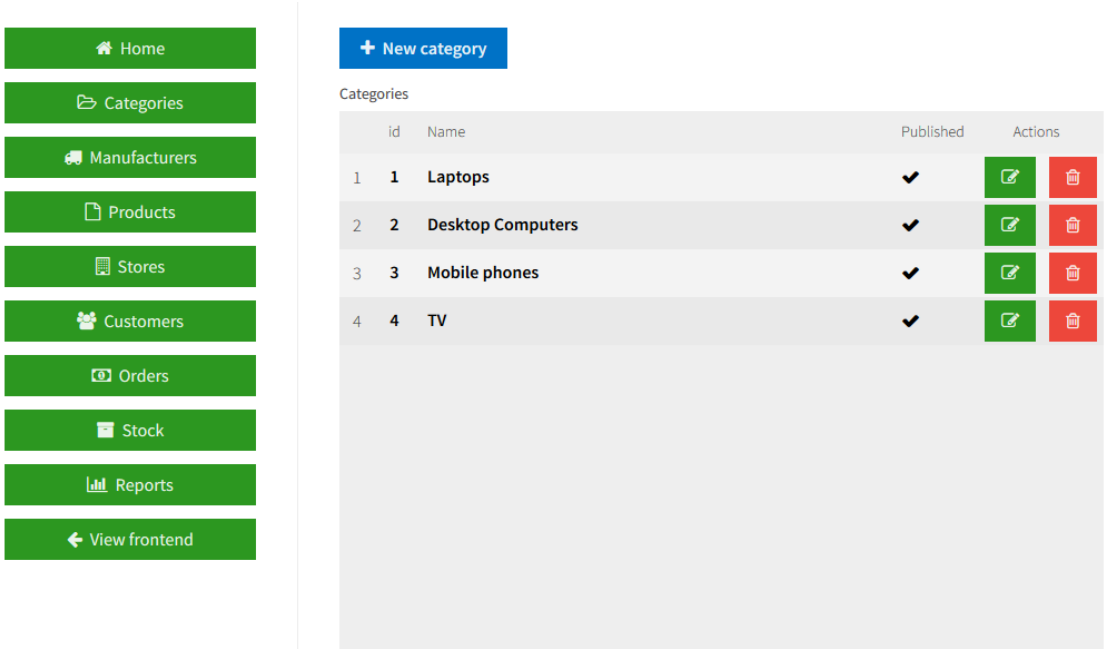
Στην σελίδα διαχείρισης φαίνονται οι εκκρεμείς παραγγελίες, ενώ στην αριστερή στήλη φαίνονται οι λειτουργίες του καταστήματος.



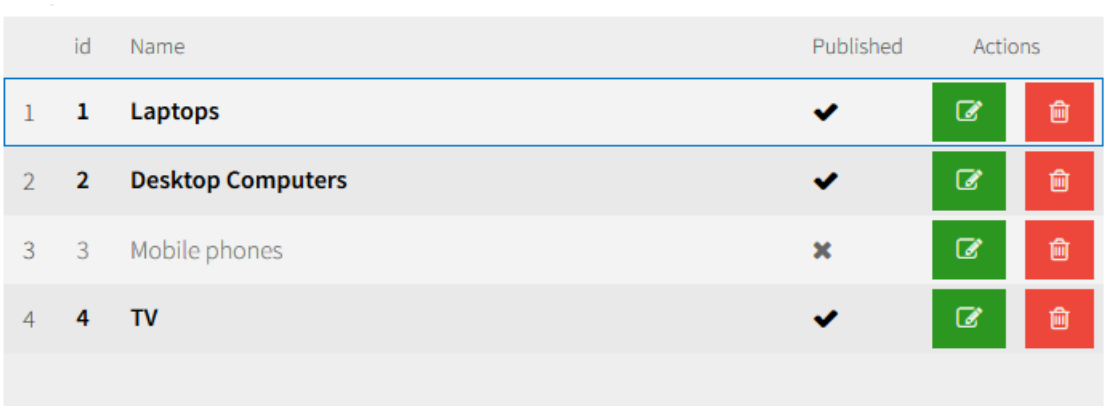
Επιλέγοντας μία λειτουργία από τα αριστερά μπορεί να πλοηγηθεί στις υπόλοιπες λειτουργίες.

5.1.3 Λίστα κατηγοριών

Πατώντας στο κουμπί κατηγοριών εμφανίζεται μία λίστα με όλες τις κατηγορίες προϊόντων.



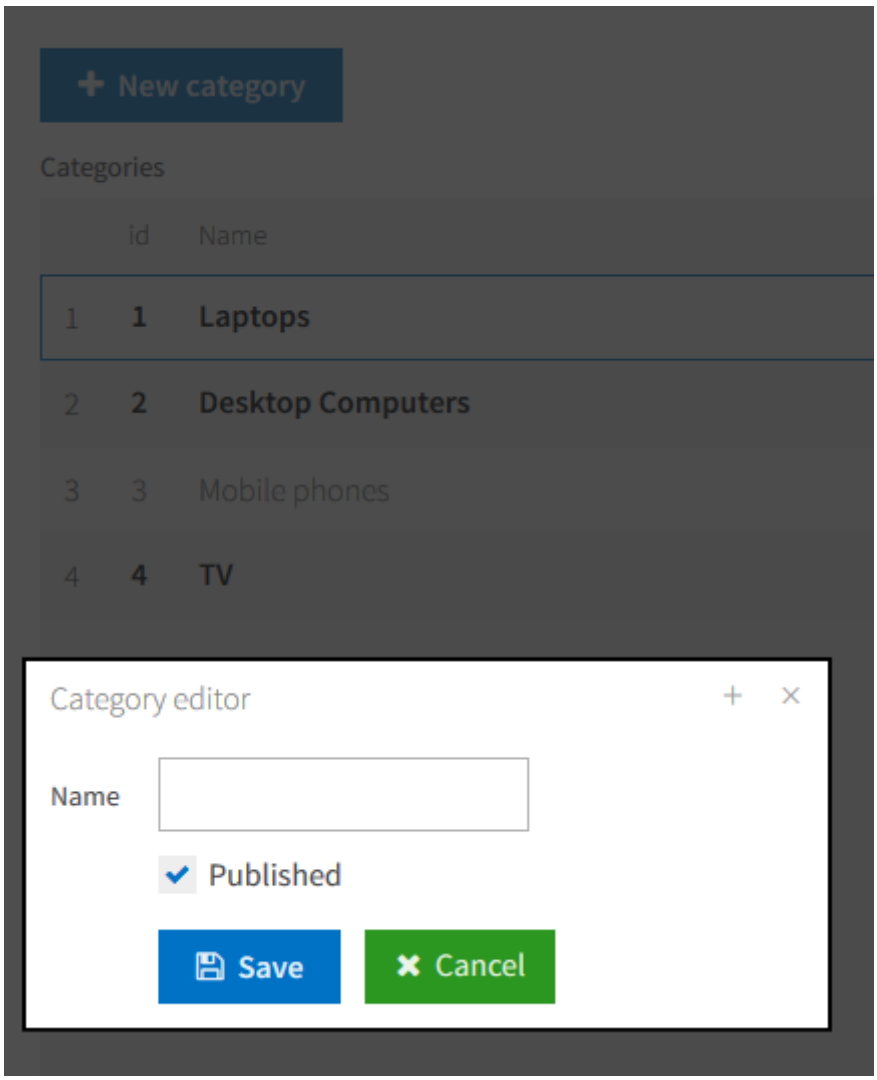
Εδώ μπορεί να κάνει μία κατηγορία μη δημοσιευμένη, πατώντας στο κόκκινο κουμπί.



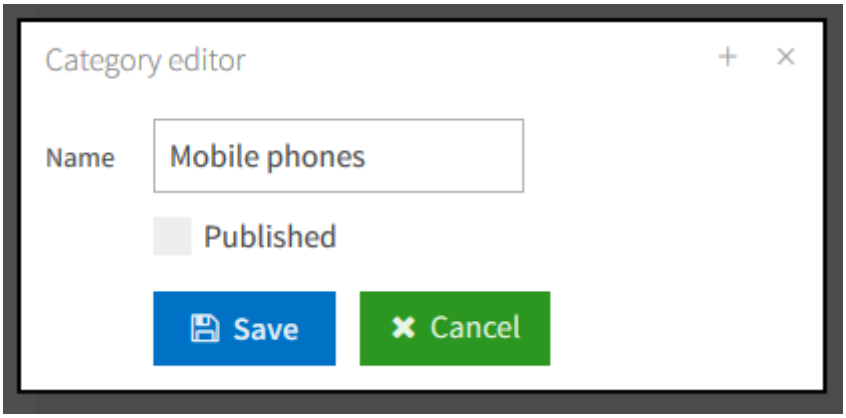
Επιπλέον μπορεί να δημιουργήσει μία νέα κατηγορία ή να επεξεργαστεί μία από τις υπάρχουσες πατώντας στο αντίστοιχο κουμπί στις κατηγορίες ή στο κουμπί νέας κατηγορίας.

5.1.3.1 Φόρμα κατηγορίας

Στην φόρμα με τα στοιχεία μίας κατηγορίας μπορεί να βάλει το όνομά της και αν θα εμφανίζεται στο site.

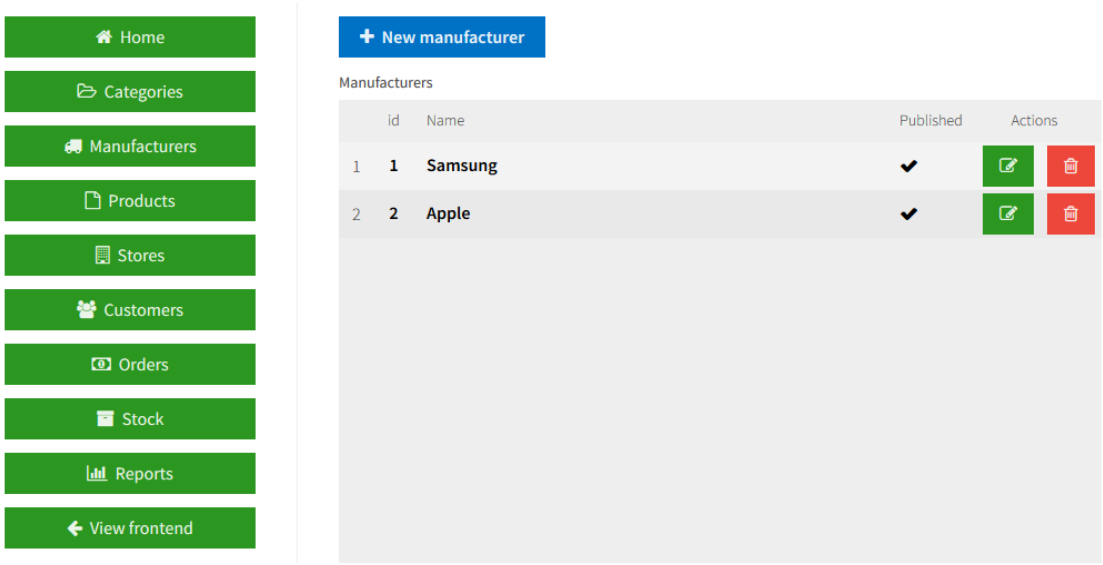


Αν τροποποιεί μία υπάρχουσα κατηγορία, τότε η φόρμα εμφανίζει τα στοιχεία της.



5.1.4 Λίστα κατασκευαστών

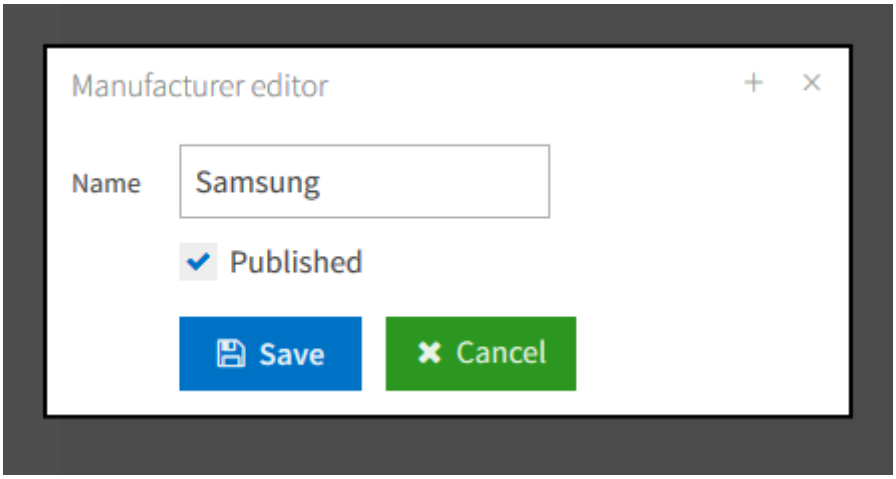
Παρόμοια με την λίστα κατηγοριών είναι η οθόνη με την λίστα των κατασκευαστών.



Πάλι μπορεί να κάνει έναν κατασκευαστή ανενεργό, να τον τροποποιήσει ή να δημιουργήσει έναν νέο.

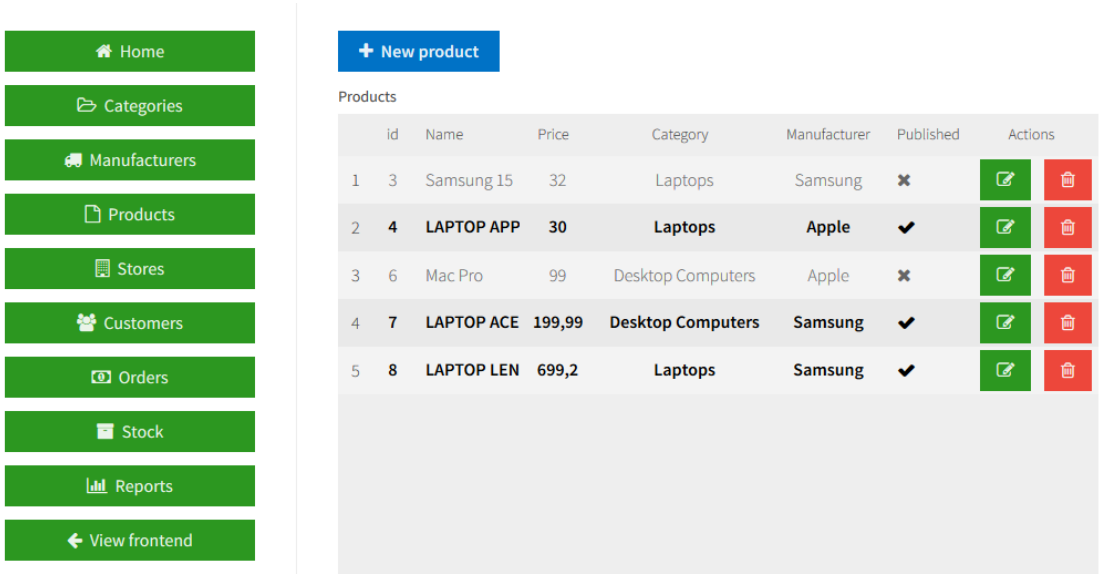
5.1.4.1 Φόρμα κατασκευαστή

Στη φόρμα κατασκευαστεί μπορεί να τροποποιηθεί ένας κατασκευαστής και να ρυθμιστεί αν θα εμφανίζεται ή όχι.



5.1.5 Λίστα προϊόντων

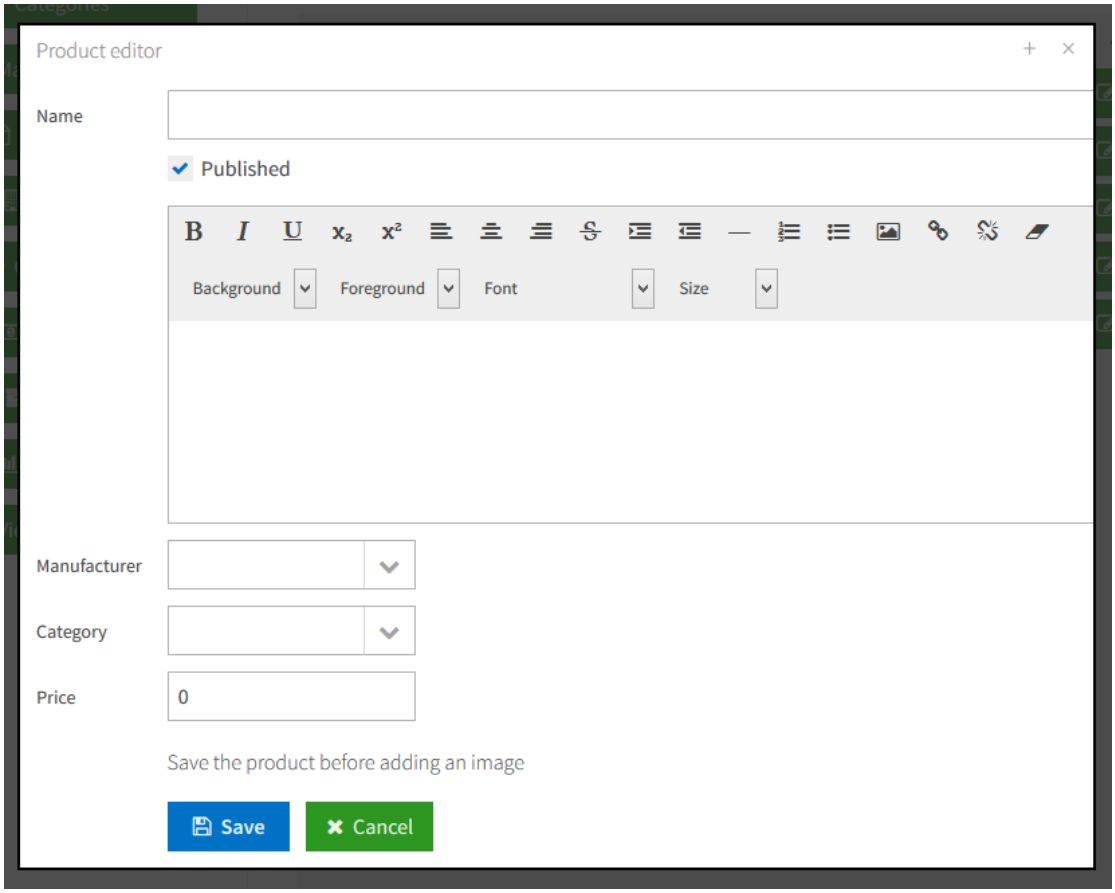
Πατώντας στην επιλογή Products εμφανίζεται η λίστα με τα προϊόντα.



Όπως και στα προηγούμενα αντικείμενα ένα προϊόν μπορεί να γίνει ανενεργό ή να τροποποιηθεί. Εναλλακτικά μπορεί να δημιουργηθεί ένα νέο.

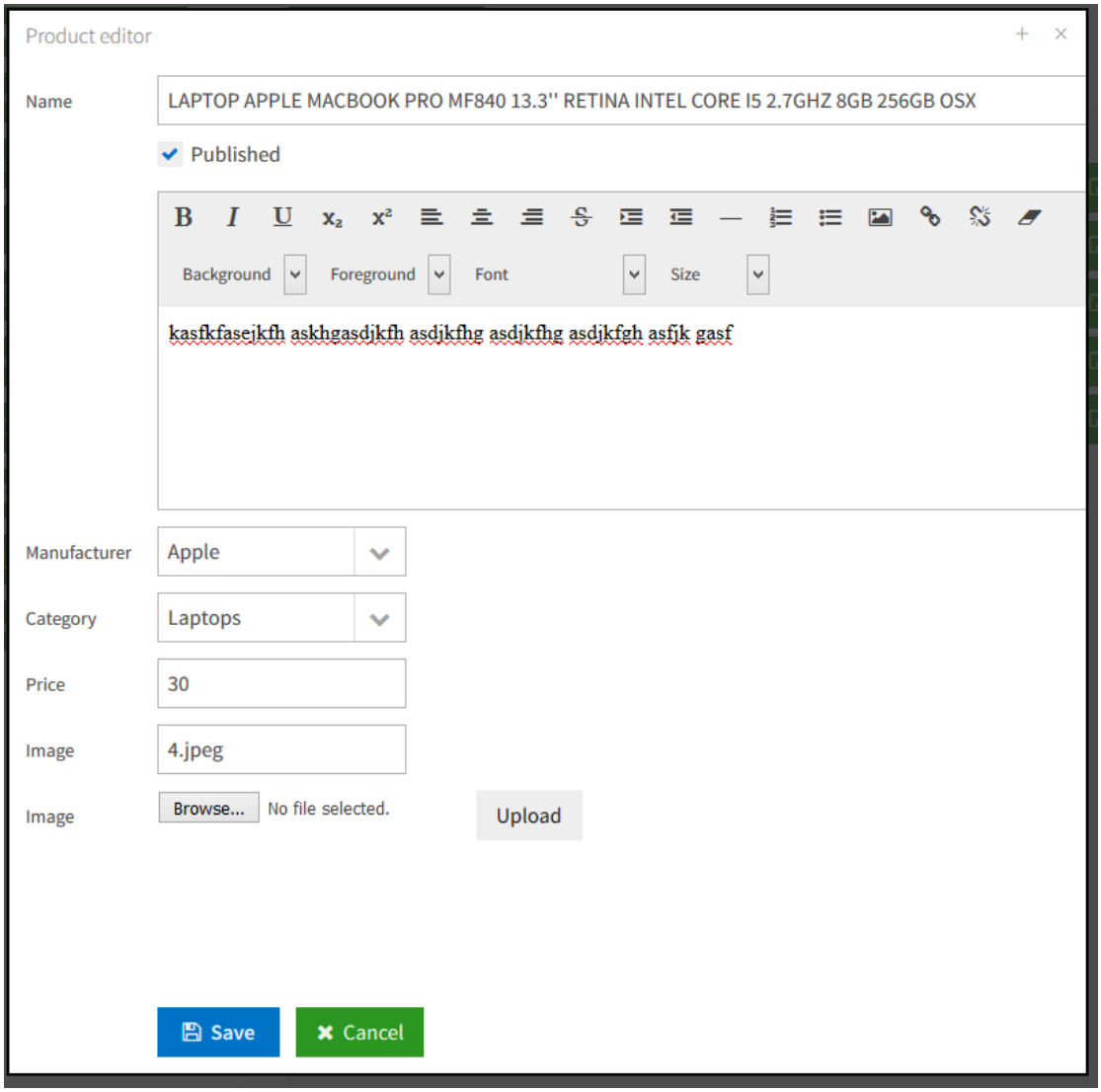
5.1.5.1 Φόρμα προϊόντος

Στη φόρμα προϊόντος εισάγονται τα στοιχεία του νέου προϊόντος ή τροποποιούνται τα υπάρχοντα.



Από τα μενού μπορεί να επιλεχθεί ο κατασκευαστής και η κατηγορία του.

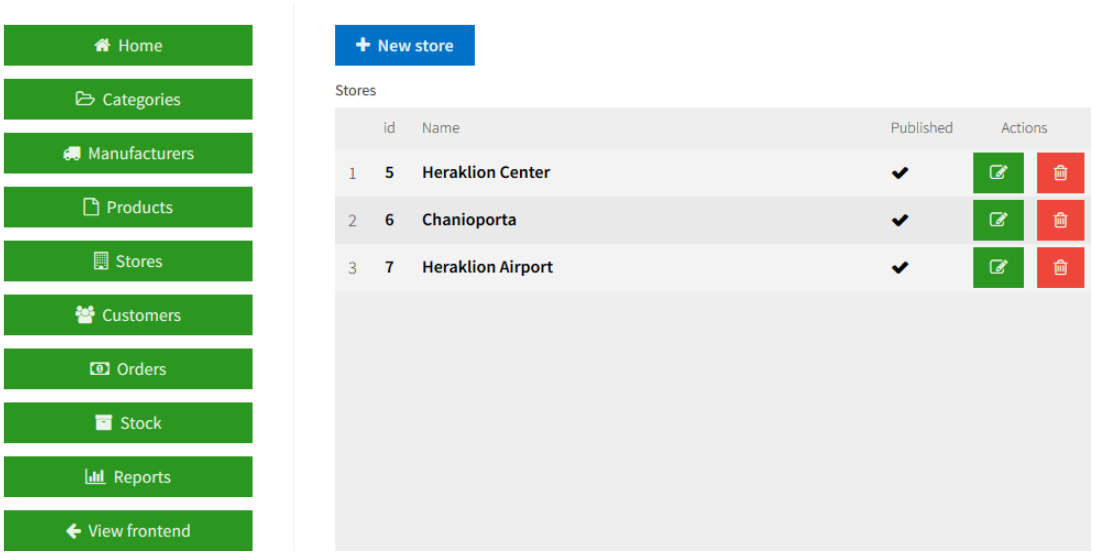
Η φωτογραφία δεν μπορεί να εισαχθεί σε ένα νέο προϊόν. Για να έχουμε μοναδικά ονόματα προϊόντος χρησιμοποιούμε το id τους, και ένα προϊόν που δεν έχει σωθεί στην βάση δεν έχει id.



Σε ένα προϊόν που έχει σωθεί μπορεί με το κουμπί Browse και Upload να επιλεχθεί μία κατηγορία και αν εισαχθεί στην βάση.

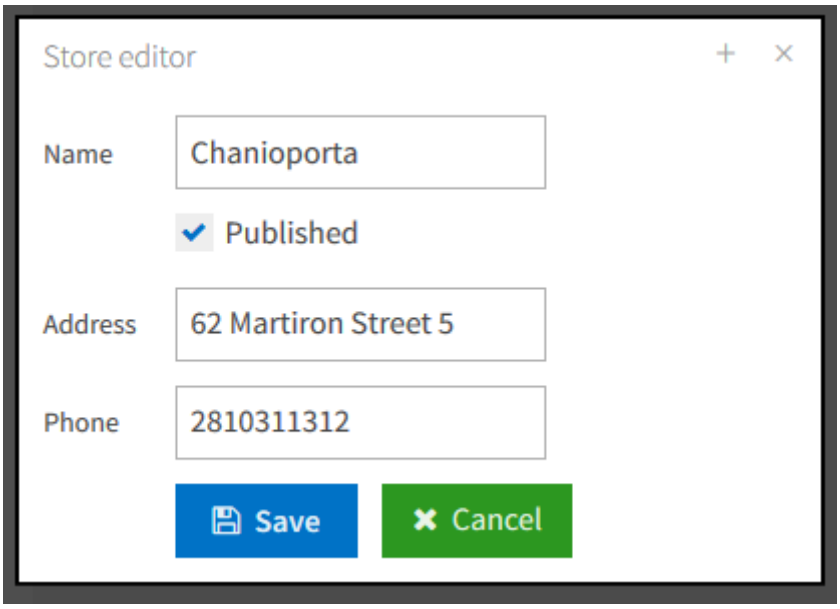
5.1.6 Λίστα καταστημάτων

Η λίστα καταστημάτων δίνει παρόμοιες λειτουργίες με τις προηγούμενες σελίδες, επιτρέποντας την δημιουργία, την απόκρυψη και την τροποποίηση ενός καταστήματος.



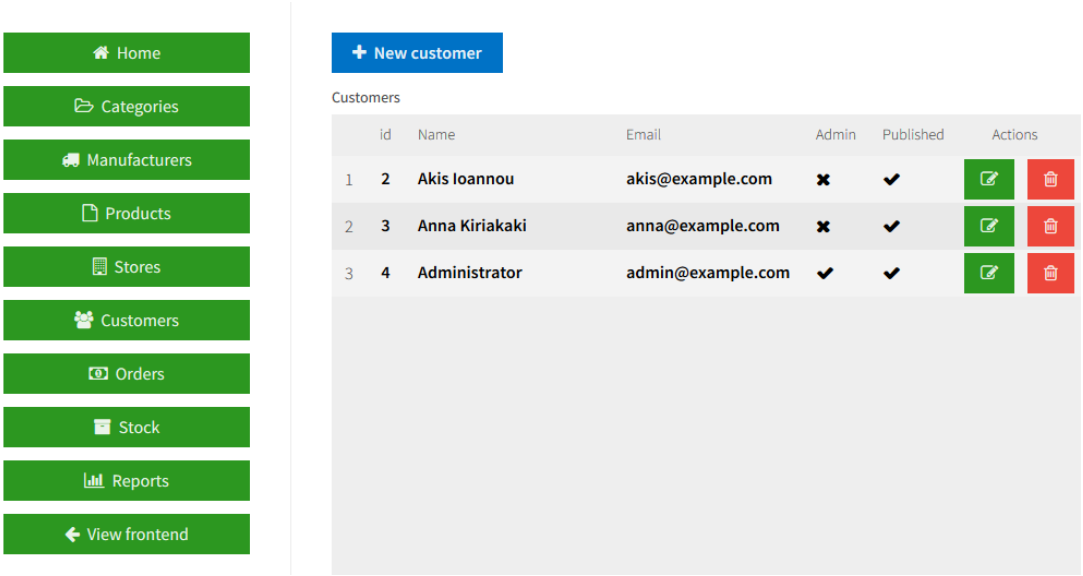
5.1.6.1 Φόρμα καταστημάτων

Όπως και παραπάνω η φόρμα καταστημάτων επιτρέπει την εισαγωγή και την τροποποίηση των καταστημάτων.



5.1.7 Λίστα πελατών

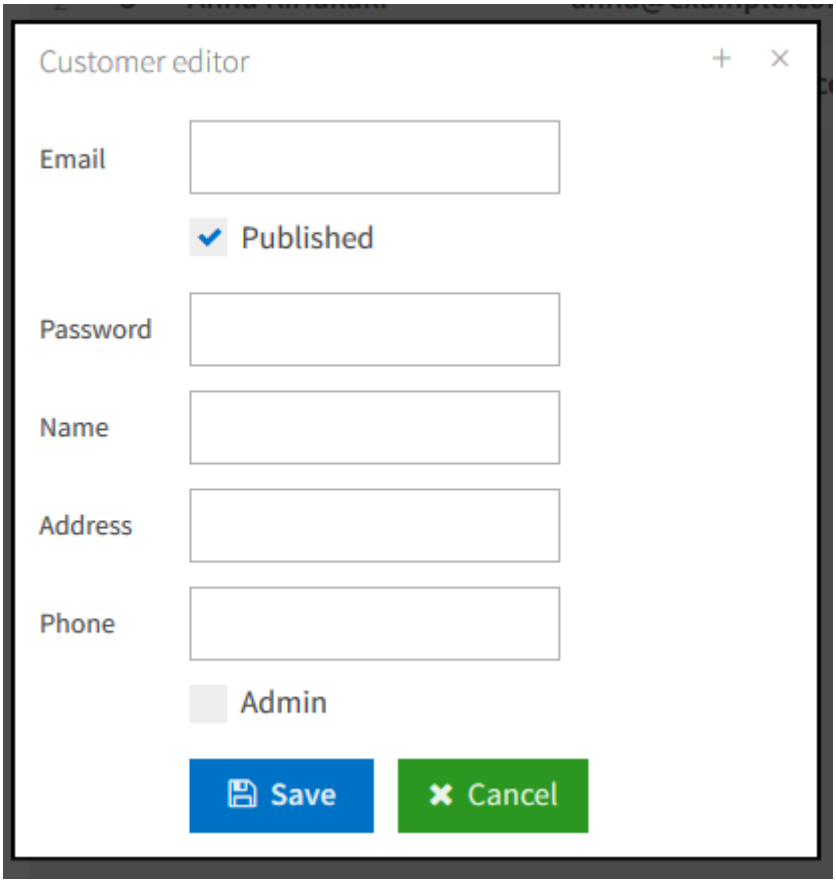
Η λίστα πελατών προσφέρει την δυνατότητα προβολής των χρηστών του συστήματος.



Για κάθε έναν υπάρχουν οι τυπικές λειτουργίες της απενεργοποίησης (δεν θα μπορεί να κάνει login), της τροποποίησης των στοιχείων του και της δημιουργίας νέου.

5.1.7.1 Φόρμα πελατών

Η φόρμα πελατών παρέχει πρόσβαση στα στοιχεί του χρήστη. Μεταξύ των επιλογών υπάρχει και το admin, που καθορίζει το αν είναι διαχειριστής.



5.1.8 Λίστα παραγγελιών

Η λίστα παραγγελιών παρέχει πρόσβαση στις παραγγελίες.

Orders

	Code	Cu	Price	Product	Submitted	Status	Delivered	Store	Actions
1	JN-7I3I	Aki	30	LAPTOP APPLE MACBOOK PRO MF840 13.3" RETINA INTEL CORE I5 2.7GHZ 8GB 256GB OSX	14/03/2016, 19:43	Cancelled			
2	1E-3EEB3	Aki	199,99	LAPTOP ACER ICONIA W510 10.1" INTEL DUAL CORE Z2760 64GB WIFI BT DUAL CAMERA WIN 8 + KEYBOARD DOCK	14/03/2016, 19:43	Delivered	14/03/2016, 19:53		
3	2G-2AJ1A	Aki	199,99	LAPTOP ACER ICONIA W510 10.1" INTEL DUAL CORE Z2760 64GB WIFI BT DUAL CAMERA WIN 8 + KEYBOARD DOCK	14/03/2016, 19:44	Delivered	15/03/2016, 22:30	Heraklion Airport	
4	VF-3D0B6	Aki	30	LAPTOP APPLE MACBOOK PRO MF840 13.3" RETINA INTEL CORE I5 2.7GHZ 8GB 256GB OSX	14/03/2016, 23:07	Cancelled		Chanioporta	
5	VB-2EC59	Ant	30	LAPTOP APPLE MACBOOK PRO MF840 13.3" RETINA INTEL CORE I5 2.7GHZ 8GB 256GB OSX	15/03/2016, 22:34	Delivered	15/03/2016, 22:35		
6	BS-3CB5G	Ant	30	LAPTOP APPLE MACBOOK PRO MF840 13.3" RETINA INTEL CORE I5 2.7GHZ 8GB 256GB OSX	15/03/2016, 23:20	Submitted		Chanioporta	
7	JP-24BDI	Aki	30	LAPTOP APPLE MACBOOK PRO MF840 13.3" RETINA INTEL CORE I5 2.7GHZ 8GB 256GB OSX	27/03/2016, 15:50	Submitted			
8	R-33I1A	Aki	30	LAPTOP APPLE MACBOOK PRO MF840 13.3" RETINA INTEL CORE I5 2.7GHZ 8GB 256GB OSX	27/03/2016, 15:52	Delivered	06/04/2016, 22:29	Chanioporta	
9	1J-2EFF5	Aki	699,2	LAPTOP LENOVO 100-15IBD 80QQ006RPB 15.6" INTEL CORE I5-5200U 4GB 500GB FREE DOS	27/03/2016, 15:52	Cancelled			
10	P9-5G9F8	Aki	699,2	LAPTOP LENOVO 100-15IBD 80QQ006RPB 15.6" INTEL CORE I5-5200U 4GB 500GB FREE DOS	27/03/2016, 15:52	In progress		Heraklion Airport	

Κάθε γραμμή δείχνει μία παραγγελία, τον πελάτη που την έκανε, το προϊόν και τις ημερομηνίες κατάθεσης και ολοκλήρωσης. Μία παραγγελία που μόλις κατατέθηκε μπορεί πατώντας το φορτηγό να μπει σε κατάσταση in progress και έπειτα πατώντας το πράσινο κουμπί ολοκλήρωσης να χαρακτηριστεί ολοκληρωμένη ή πατώντας το κόκκινο κουμπί να χαρακτηριστεί ακυρωμένη.

5.1.9 Απόθεμα προϊόντος

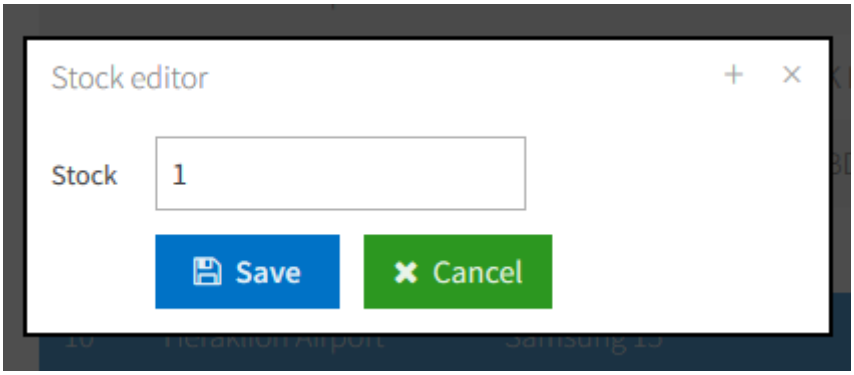
Επιλέγοντας το stock εμφανίζεται μία οθόνη με τα αποθέματα προϊόντος.

Stores

	Store	Product	Stock	Actions
1	Chanioporta	LAPTOP ACER ICONIA W510 10.1'' INTEL DUAL	1	
2	Chanioporta	LAPTOP APPLE MACBOOK PRO MF840 13.3'' RI	7	
3	Chanioporta	LAPTOP LENOVO 100-15IBD 80QQ006RPB 15.6	0	
4	Chanioporta	Mac Pro	2	
5	Chanioporta	Samsung 15	4	
6	Heraklion Airport	LAPTOP ACER ICONIA W510 10.1'' INTEL DUAL	3	
7	Heraklion Airport	LAPTOP APPLE MACBOOK PRO MF840 13.3'' RI	1	
8	Heraklion Airport	LAPTOP LENOVO 100-15IBD 80QQ006RPB 15.6	0	
9	Heraklion Airport	Mac Pro	2	
10	Heraklion Airport	Samsung 15	1	
11	Heraklion Center	LAPTOP ACER ICONIA W510 10.1'' INTEL DUAL	0	
12	Heraklion Center	LAPTOP APPLE MACBOOK PRO MF840 13.3'' RI	0	
13	Heraklion Center	LAPTOP LENOVO 100-15IBD 80QQ006RPB 15.6	0	
14	Heraklion Center	Mac Pro	0	
15	Heraklion Center	Samsung 15	0	

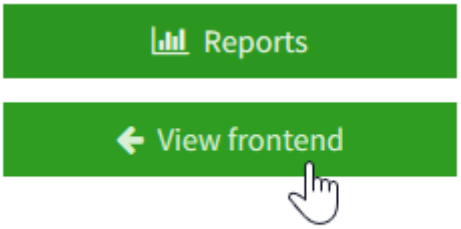
Για κάθε κατάστημα για κάθε προϊόν υπάρχει μία εγγραφή με την ποσότητα που υπάρχει σε απόθεμα.

Προφανώς δεν έχει νόημα να σβηστεί κάποια εγγραφή, αλλά μπορεί να τροποποιηθεί η ποσότητα.



5.1.10 Επιστροφή στο κατάστημα

Πατώντας το κουμπί View frontend το σύστημα προωθεί τον διαχειριστή στην σελίδα του καταστήματος.



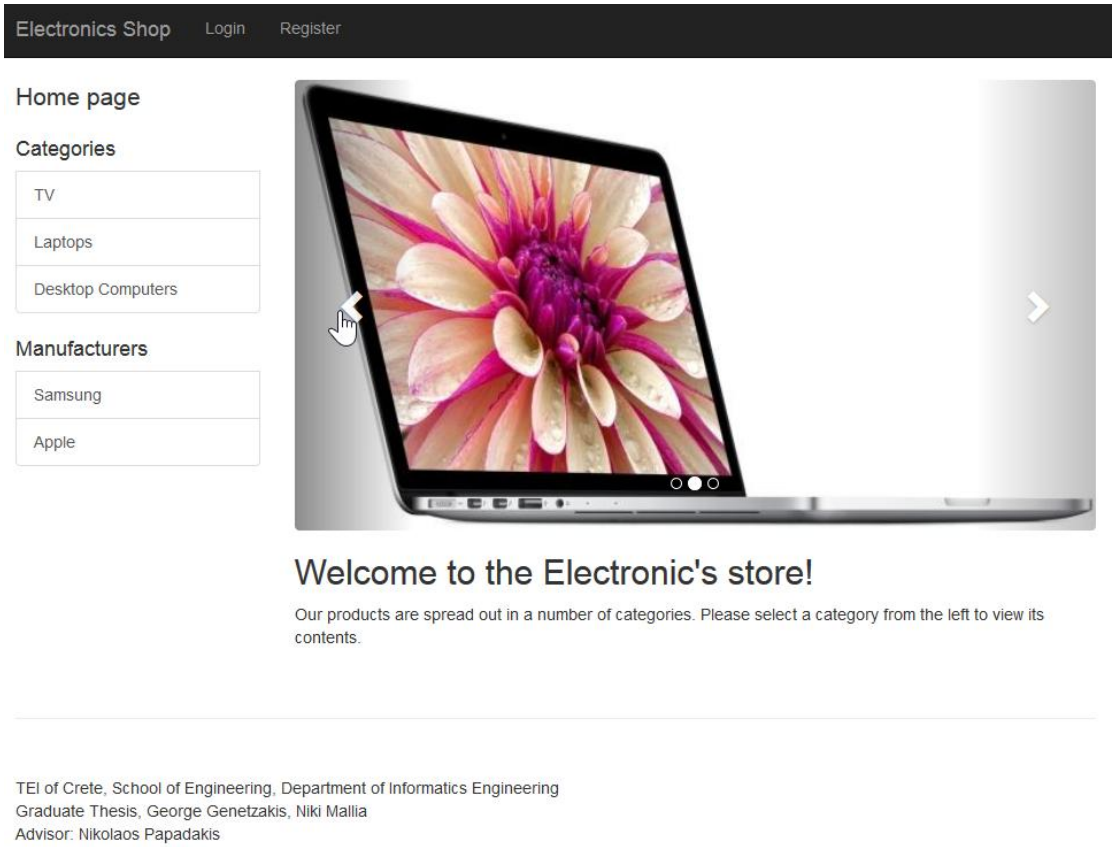
5.2 Περιοχή χρηστών

Παρακάτω παρουσιάζονται οι λειτουργίες του site.

Αυτές διαφέρουν σε διάφορα σημεία ανάλογα με το αν ο χρήστης έχει κάνει login ή όχι, κυρίως στην δυνατότητα να αγοράσει προϊόντα.

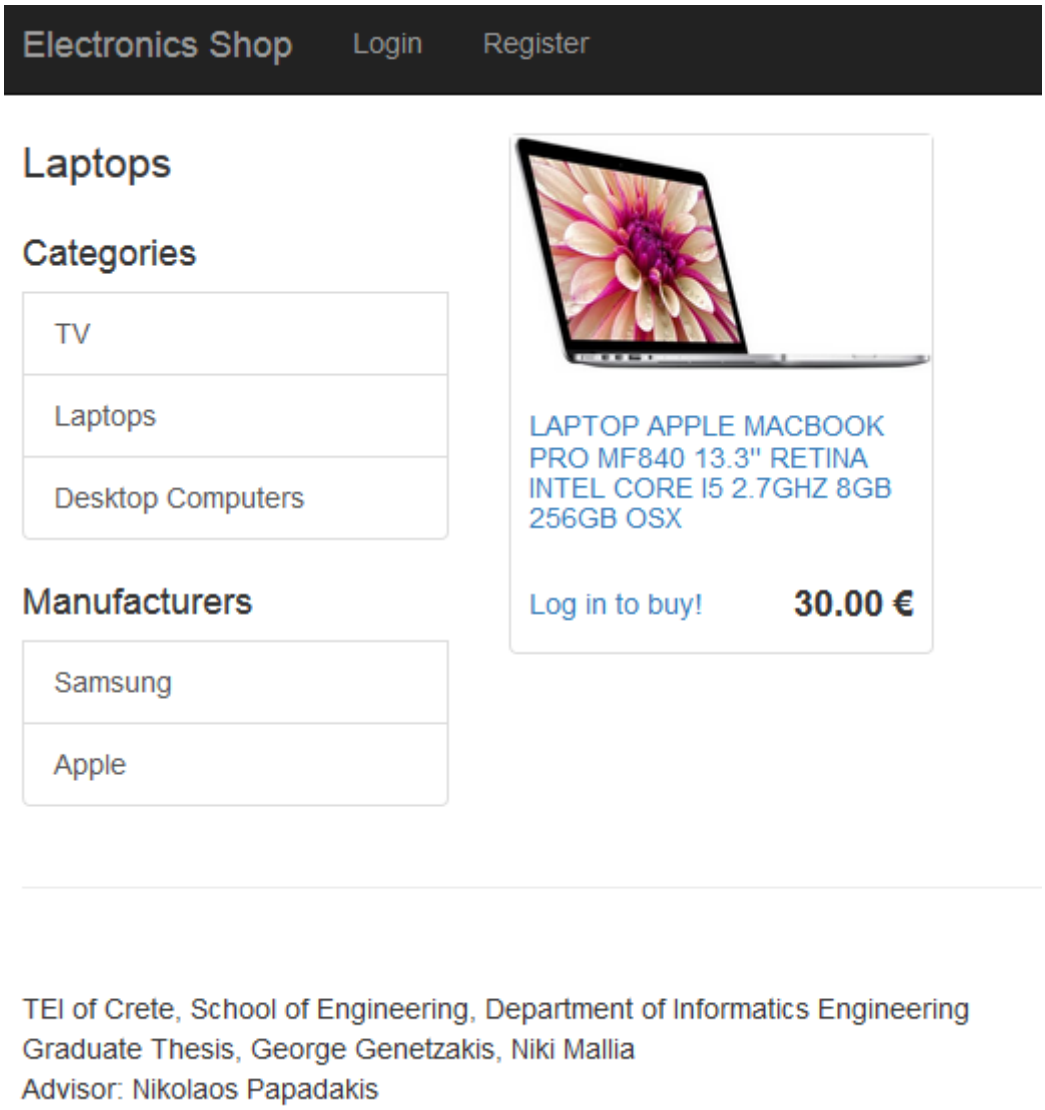
5.2.1 Αρχική σελίδα

Στην αρχική σελίδα φαίνονται οι κατασκευαστές και οι κατηγορίες στα αριστερά, και οι λειτουργίες χρήστη στο επάνω μέρος.



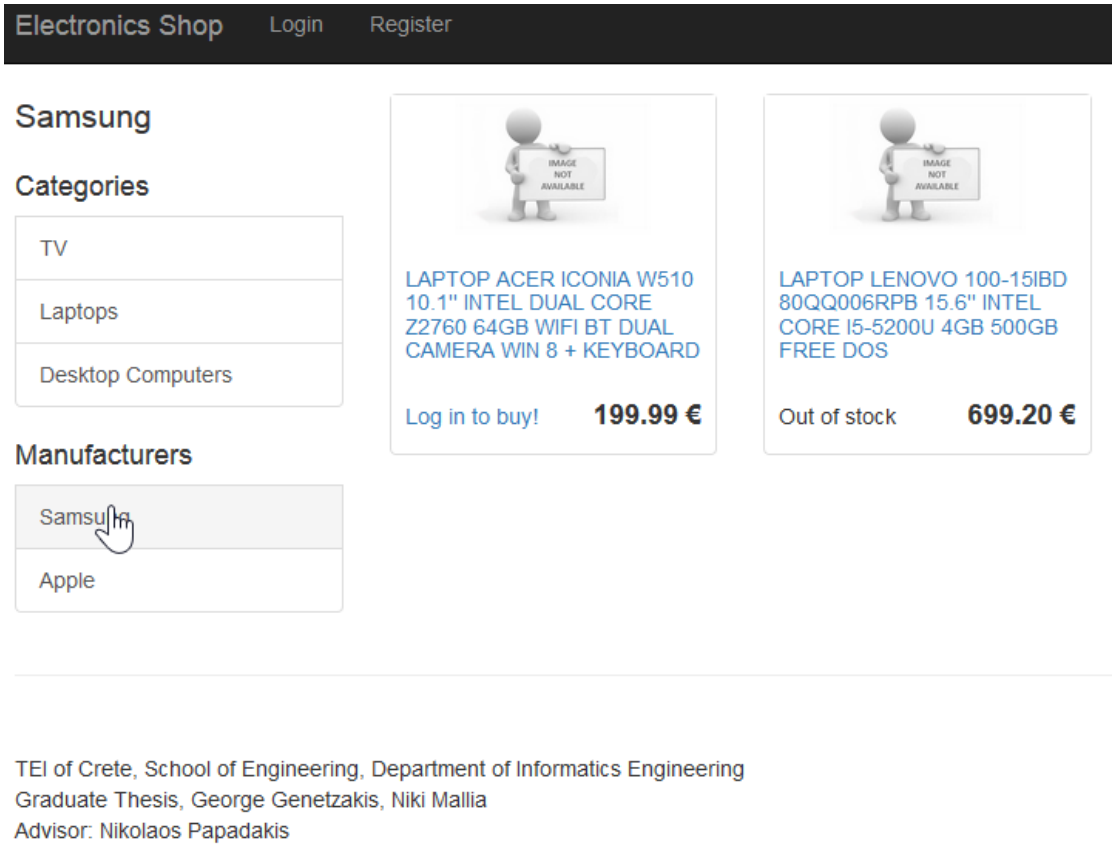
5.2.2 Λίστα προϊόντων ανά κατηγορία

Πατώντας σε μία κατηγορία ο χρήστης μπορεί να δει όλα τα προϊόντα αυτής της κατηγορίας.



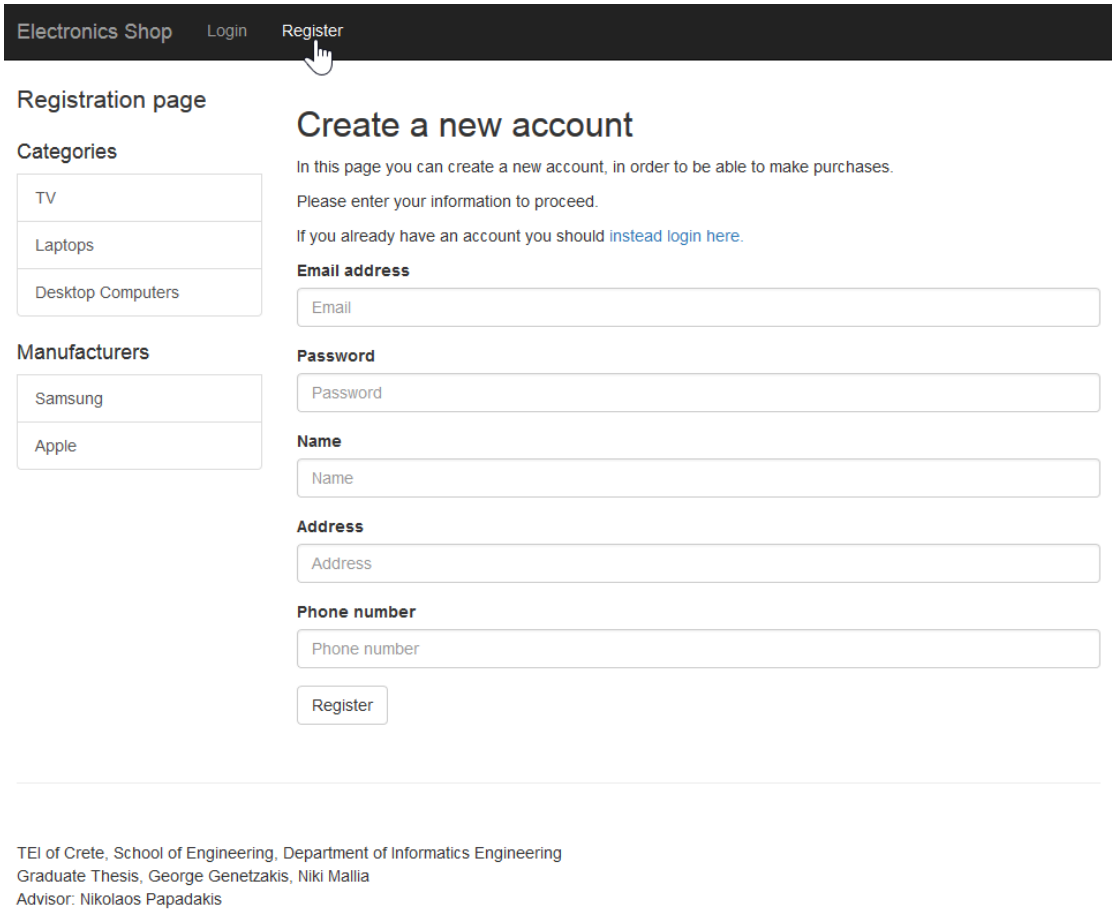
5.2.3 Λίστα προϊόντων ανά κατασκευαστή

Αντίστοιχα επιλέγοντας έναν κατασκευαστή φαίνονται όλα τα προϊόντα του.



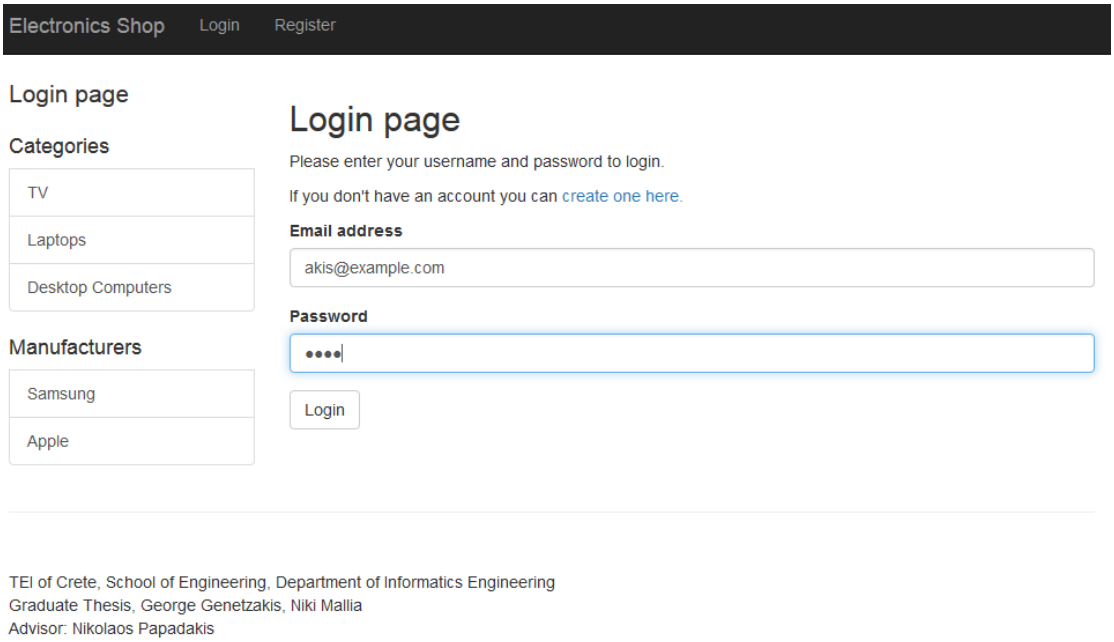
5.2.4 Δημιουργία λογαριασμού

Ένας χρήστης μπορεί να εισάγει τα στοιχεία του για να δημιουργήσει λογαριασμό στο σύστημα. Προφανώς δεν είναι διαχειριστής, καθώς αυτό αλλάζει μόνο από το περιβάλλον διαχείρισης.



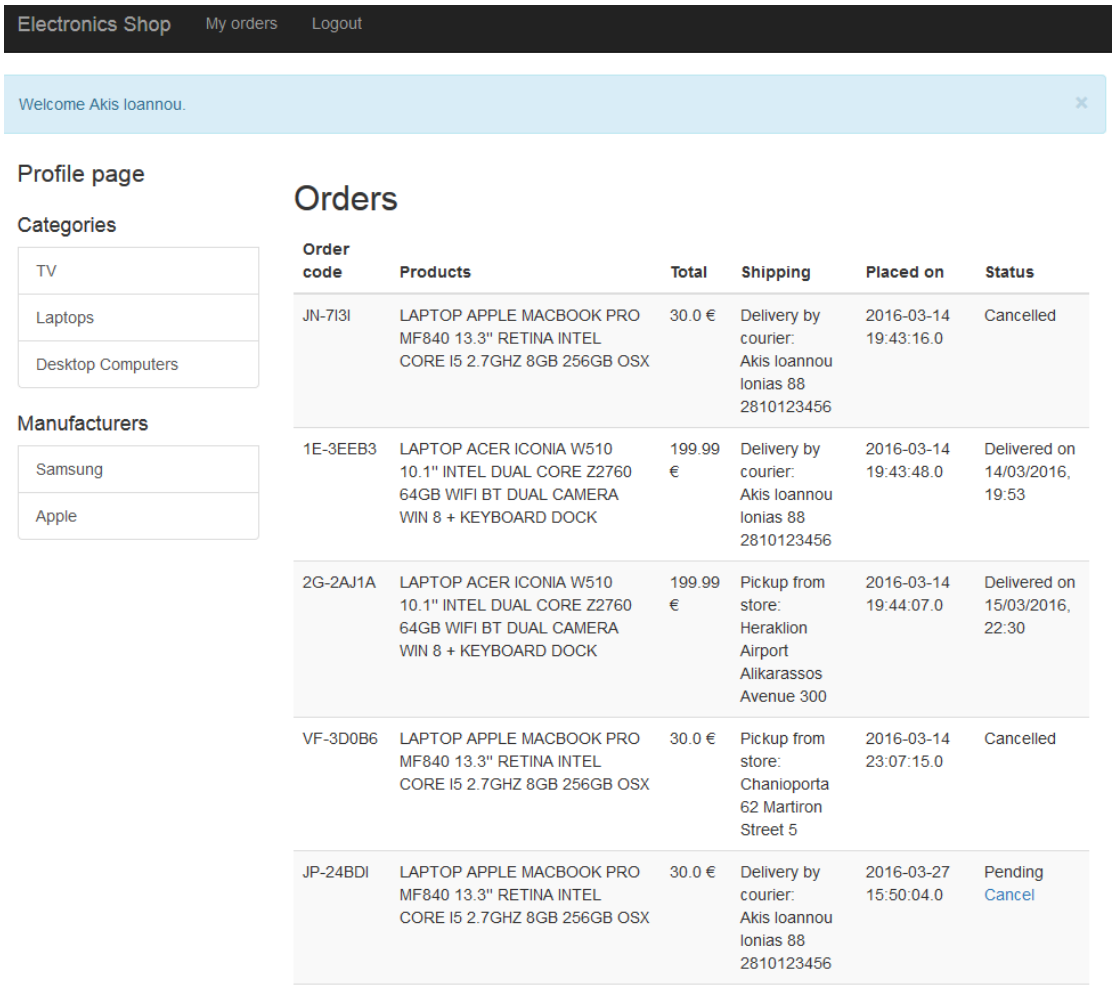
5.2.5 Είσοδος στο σύστημα

Η οθόνη εισόδου είναι κοινή με τους διαχειριστές όπως παρουσιάστηκε παραπάνω.



5.2.6 Λίστα παραγγελιών

Μπαίνοντας στο σύστημα ή πατώντας την επιλογή My orders αφού έχει κάνει login, ένας χρήστης μπορεί να δει τις παραγγελίες του.



Για όσες εκκρεμούν μπορεί να τις ακυρώσει.

5.2.7 Οθόνη προϊόντος

Πατώντας επάνω σε ένα προϊόν από οποιαδήποτε λίστα ο χρήστης μπορεί να δει αυτό το προϊόν. Αν έχει κάνει login μπορεί να το αγοράσει.

5.2.8 Οθόνη παραγγελίας

Αν ο χρήστης επιλέξει να αγοράσει ένα προϊόν πρέπει να συμπληρώσει την φόρμα παραγγελίας.

Αρχικά συμπληρώνει τα στοιχεία του.

Checkout page

You are going to purchase this product:

Product:

Name	Price
LAPTOP APPLE MACBOOK PRO MF840 13.3" RETINA INTEL CORE I5 2.7GHZ 8GB 256GB OSX	30.00 €

Please fill in the details and review the following info.

Once you are ready, press the Purchase button to submit your order.

Email address

akis@example.com

Name

Akis Ioannou

Address

Ionias 88

Phone number

2810123456

Έπειτα επιλέγει αν θέλει να παραλάβει το προϊόν με κούριερ ή από το κατάστημα.

Delivery method

- ○ Send by courier
- ◉ Store pickup

Pickup store

Chanioporta (62 Martiron Street 5) selected

Chanioporta (62 Martiron Street 5) selected

Heraklion Airport (Alikarassos Avenue 300)

Αν επιλέξει παραλαβή από το κατάστημα πρέπει να διαλέξει ένα από τα καταστήματα που παρουσιάζονται. Αυτά είναι μόνο όσα έχουν το προϊόν σε στοκ.

Τέλος επιλέγει πληρωμή με πιστωτική κάρτα ή με μετρητά.

Payment method

○ Cash

◉ Credit card

Credit card type

Visa

Credit card number

XXXX-XXXX-XXXX-XXXX

Holder name

eg. JOHN SMITH

CCV

XXX

Purchase

Τέλος με το κουμπί purchase ολοκληρώνει την παραγγελία του.

6 Συμπεράσματα

6.1 Γενικά

Όταν είχαμε αρχικά διερευνήσει τις τεχνολογίες που χρησιμοποιήσαμε, η γενικότερη ιδέα που μας είχε δημιουργηθεί ήταν πώς κάνουν την ανάπτυξη των εφαρμογών αρκετά απλούστερη, κρύβοντας το πλήθος των γλωσσών που χρησιμοποιούνται για την ανάπτυξη μίας τυπικής σελίδας πίσω από την γλώσσα Java. Όμως η εμπειρία που αποκτήσαμε κατά την χρήση τους μας έδειξε ότι η αλήθεια είναι αρκετά διαφορετική.

Το γεγονός ότι ο προγραμματιστής χρησιμοποιεί την γλώσσα Java και αποφεύγει να γράψει SQL ή HTML είναι όντως ένα πλεονέκτημα, γιατί με αυτόν τον τρόπο ο κώδικας που παράγεται μπορεί να ελεγθεί άμεσα, ενώ η δυναμική δημιουργία εντολών SQL (ή σελίδων HTML, αν και τη χρησιμοποιήσαμε στο front end) μπορεί να δημιουργήσει σε λάθη που δεν είναι εμφανή μέχρι να τρέξει η εφαρμογή. Σε αυτόν τον τομέα υπάρχει ένα εμφανές πλεονέκτημα, που θεωρητικά επιταχύνει την ανάπτυξη.

Όμως το γεγονός ότι δεν χρησιμοποιείται άμεσα SQL δεν σημαίνει ότι δεν χρειάζονται γνώσεις SQL. Όταν χρησιμοποιείται μία οντότητα που αναπαριστά δεδομένα στην βάση είναι αναγκαίο να γνωρίζουμε πώς αυτή έχει αναπαρασταθεί στην βάση, πώς οι λειτουργίες που εκτελούνται πάνω σε αυτήν θα μεταφραστούν σε SQL και τι αντίκτυπο θα έχουν στα δεδομένα των πινάκων.

Ένα παράδειγμα έχει να κάνει με τα προϊόντα και τις κατηγορίες: Κάθε προϊόν ανήκει σε μία κατηγορία, δηλαδή ο πίνακας του προϊόντος έχει ένα ξένο κλειδί με το id της κατηγορίας στην οποία ανήκει. Τυπικά για να αλλάξει κατηγορία αρκεί να του δώσουμε την νέα κατηγορία. Όμως κάθε αντικείμενο κατηγορίας έχει και μία λίστα με προϊόντα. Τυπικά η αλλαγή κατηγορίας δεν δημιουργεί προβλήματα, όμως αν στο ίδιο κομμάτι κώδικα χρησιμοποιείται και η παλιά κατηγορία του, τότε αυτή περιέχει μία λίστα με όλα τα προϊόντα της. Αν το προϊόν που τροποποιήθηκε δεν βγει από αυτή τη λίστα μπορεί, ανάλογα με την περίπτωση να έχουμε λάθος δεδομένα ή το πρόγραμμα να πετάξει Exception. Τέτοιου είδους λάθη δεν είναι πάντα προφανή, είναι χρονοβόρο να διορθωθούν και η επίλυσή τους πολλές φορές χρειάζεται αλλαγές σε πολλά σημεία του κώδικα (στο παραπάνω παράδειγμα δεν χρειάζεται να υπάρχει λίστα με τα προϊόντα της κατηγορίας μέσα σε αυτή γιατί χρησιμοποιείται σπάνια, οπότε όταν είναι αναγκαίο να τα ανακτήσουμε μπορούμε να τα πάρουμε από τη βάση).

Πέρα από το κομμάτι του JPA, το Vaadin γενικά λειτουργεί αρκετά καλά και μειώνει την πολυπλοκότητα της ανάπτυξης, όμως για την σωστή χρήση του ο προγραμματιστής χρειάζεται να μάθει πώς λειτουργεί, το οποίο πρακτικά σημαίνει ότι πρέπει να εξοικειωθεί με ένα αρκετά μεγάλο πλήθος εννοιών και αντικειμένων.

Τέλος το Velocity επιτελεί πρακτικά την ίδια λειτουργία με τα αρχεία JSP με λιγότερο και πιο κατανοητό κώδικα.

Πέρα από την δυσκολία στην χρήση των παραπάνω τεχνολογιών, ένα άλλο αρκετά χρονοβόρο τμήμα της διαδικασίας ανάπτυξης ήταν η δημιουργία ενός project που να τις συνδυάζει. Αν και υπάρχουν γενικές οδηγίες για το πώς μπορεί να χρησιμοποιηθεί κάθε τεχνολογία από μόνη της, αυτές καλύπτουν πολύ βασικές περιπτώσεις και ο συνδυασμός τους είναι αρκετά δύσκολος, κυρίως εξαιτίας της έλλειψης υλικού. Το σύστημα maven δεν παρέχει απλά την ευκολία του να μην συγκεντρώνουμε χειροκίνητα τα αρχεία που χρειάζονται σε συγκεκριμένους φακέλους, αλλά είναι απαραίτητο για να γίνουν διάφορα τμήματα του compile, κάτι το οποίο όμως δεν είναι προφανές από τις οδηγίες.

6.2 Μελλοντική μελέτη

Διάφορα τμήματα της εφαρμογής προσφέρονται για επέκταση και μελλοντική μελέτη. Πέρα από την επέκταση των δυνατοτήτων της, υπάρχουν ευκαιρίες για έρευνα πάνω στις τεχνολογίες που χρησιμοποιήθηκαν.

Στο τμήμα του JPA χρησιμοποιήσαμε την βάση MySQL, αλλά θεωρητικά μπορεί να χρησιμοποιηθεί οποιαδήποτε συμβατή βάση δεδομένων. Θα ήταν ενδιαφέρον να γίνει μία σύγκριση για το αν η βάση μπορεί να αντικατασταθεί με κάποια άλλη όπως η PostgreSQL ή ο Microsoft SQL Server και αν θα χρειαστούν αλλαγές.

Καθώς το JPA κάνει αρκετά πράγματα αυτόματα, καταλήγει να κάνει πολύ περισσότερα πράγματα από ότι χρειάζονται (για παράδειγμα αν θέλουμε την τιμή ενός προϊόντος τυπικά παίρνουμε το προϊόν, δηλαδή φορτώνεται όλη η εγγραφή από την βάση και από αυτό παίρνουμε την τιμή). Υποτίθεται ότι το σύστημα είναι αρκετά βελτιστοποιημένο, όμως θα ήταν ενδιαφέρον να γίνουν μετρήσεις για τις επιπτώσεις που έχει αυτό στις επιδόσεις του συστήματος, αντικαθιστώντας διάφορα τμήματα με εντολές SQL.

Ένας άλλος τομέας που μπορεί να διερευνηθεί έχει να κάνει με την μετατροπή της διεπαφής, τόσο του HTML front end όσο και του Vaadin back end, ώστε να υποστηρίζει συσκευές με μικρές οθόνες.

7 Βιβλιογραφία

EE Subsystem Configuration . [Ηλεκτρονικό] https://docs.jboss.org/author/display/WFLY8/EE+Subsystem+Configuration.

Apache Velocity. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/Apache_Velocity.

Beginning Apache Velocity. [Ηλεκτρονικό] http://edwin.baculsoft.com/2011/06/beginning-apache-velocity-creating-a-simple-web-application/.

Book of Vaadin. [Ηλεκτρονικό] https://vaadin.com/book.

Data form validation. [Ηλεκτρονικό] https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Forms/Data_form_validation.

DataSource configuration. [Ηλεκτρονικό] https://docs.jboss.org/author/display/WFLY8/DataSource+configuration.

Full-stack Java web dev with Vaadin. [Ηλεκτρονικό] https://www.ibm.com/developerworks/library/j-full-stack-java-web-dev-vaadin/index.html.

Hibernate EntityManager. [Ηλεκτρονικό] https://docs.jboss.org/hibernate/entitymanager/3.6/reference/en/html_single/.

Hibernate ORM documentation (5.1). [Ηλεκτρονικό] http://hibernate.org/orm/documentation/5.1/.

Html5 and CSS3 Tutorial. [Ηλεκτρονικό] http://www.html5andcss3.org/.

Java Persistence API. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/Java_Persistence_API.

Java Persistence/ManyToMany. [Ηλεκτρονικό] https://en.wikibooks.org/wiki/Java_Persistence/ManyToMany.

Java Persistence/OneToMany. [Ηλεκτρονικό] https://en.wikibooks.org/wiki/Java_Persistence/OneToMany.

Java Transaction API. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/Java_Transaction_API.

Java Transaction API (JTA). [Ηλεκτρονικό] http://www.oracle.com/technetwork/java/javaee/jta/index.html.

JBoss vs. Tomcat: Choosing A Java Application Server. [Ηλεκτρονικό] https://www.futurehosting.com/blog/jboss-vs-tomcat-choosing-a-java-application-server/.

JPA 2 Tutorial – Many To Many with Self. [Ηλεκτρονικό] http://blog.jbaysolutions.com/2013/06/06/jpa-2-tutorial-many-to-many-with-self-2/.

JPA 2 Tutorial – Relationships – Many To Many. [Ηλεκτρονικό] http://blog.jbaysolutions.com/2012/12/17/jpa-2-relationships-many-to-many/.

JPA Annotations for Relationships. [Ηλεκτρονικό] http://www.objectdb.com/api/java/jpa/annotations/relationship.

JPA Tutorial. [Ηλεκτρονικό] http://www.coderpanda.com/jpa-tutorial/.

Management API reference. [Ηλεκτρονικό] https://docs.jboss.org/author/display/WFLY8/Management+API+reference.

Persistence. [Ηλεκτρονικό] https://docs.oracle.com/javaee/7/tutorial/partpersist.htm.

The Apache Velocity Project. [Ηλεκτρονικό] https://velocity.apache.org/.

Understanding JPA, Part 2: Relationships the JPA way. [Ηλεκτρονικό] http://www.javaworld.com/article/2077819/java-se/understanding-jpa-part-2-relationships-the-jpa-way.html.

Vaadin. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/Vaadin.

Vaadin Tutorial. [Ηλεκτρονικό] https://vaadin.com/docs/-/part/framework/tutorial.html.

Velocity - An Introduction. [Ηλεκτρονικό] http://www.javaranch.com/journal/2004/03/Velocity-AnIntroduction.html.

VelocityViewServlet . [Ηλεκτρονικό] https://velocity.apache.org/tools/devel/view.servlet.html.

WildFly is a flexible, lightweight, managed application runtime . [Ηλεκτρονικό] http://wildfly.org/.

8 Παράρτημα: Κώδικας εφαρμογής

Αρχείο eshop\pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <parent>
        <artifactId>eshop-parent</artifactId>
        <groupId>gr.eshop7</groupId>
        <version>1.0-SNAPSHOT</version>
    </parent>
    <modelVersion>4.0.0</modelVersion>

    <artifactId>eshop</artifactId>
    <packaging>war</packaging>
    <name>eshop</name>

    <repositories>
        <repository>
            <id>vaadin-addons</id>
            <url>http://maven.vaadin.com/vaadin-addons</url>
        </repository>
        <repository>
            <id>vaadin-snapshots</id>
            <url>https://oss.sonatype.org/content/repositories/vaadin-
snapshots/</url>
            <releases>
                <enabled>false</enabled>
            </releases>
            <snapshots>
                <enabled>true</enabled>
            </snapshots>
        </repository>
    </repositories>

    <dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>com.vaadin</groupId>
                <artifactId>vaadin-bom</artifactId>
                <version>${vaadin.version}</version>
                <type>pom</type>
                <scope>import</scope>
            </dependency>
        </dependencies>
    </dependencyManagement>

    <dependencies>

        <dependency>
            <groupId>gr.eshop7</groupId>
            <artifactId>model</artifactId>
            <version>1.0-SNAPSHOT</version>
        </dependency>

        <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>javax.servlet-api</artifactId>
        </dependency>

        <dependency>
            <groupId>javax.inject</groupId>
```

```
    <artifactId>javax.inject</artifactId>
</dependency>

<dependency>
    <groupId>javax.enterprise</groupId>
    <artifactId>cdi-api</artifactId>
</dependency>

<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-entitymanager</artifactId>
</dependency>


<dependency>
    <groupId>com.vaadin</groupId>
    <artifactId>vaadin-server</artifactId>
</dependency>

<dependency>
    <groupId>com.vaadin</groupId>
    <artifactId>vaadin-push</artifactId>
</dependency>

<dependency>
    <groupId>com.vaadin</groupId>
    <artifactId>vaadin-themes</artifactId>
</dependency>

<dependency>
    <groupId>com.vaadin</groupId>
    <artifactId>vaadin-client</artifactId>
</dependency>

<dependency>
    <groupId>com.vaadin</groupId>
    <artifactId>vaadin-client-compiler</artifactId>
    <exclusions>
        <exclusion>
            <groupId>org.eclipse.jetty</groupId>
            <artifactId>jetty-annotations</artifactId>
        </exclusion>
        <exclusion>
            <groupId>org.eclipse.jetty</groupId>
            <artifactId>jetty-servlets</artifactId>
        </exclusion>
        <exclusion>
            <groupId>org.eclipse.jetty</groupId>
            <artifactId>jetty-util</artifactId>
        </exclusion>
    </exclusions>
</dependency>
<dependency>
    <groupId>com.vaadin</groupId>
    <artifactId>vaadin-cdi</artifactId>
</dependency>
<dependency>
    <groupId>com.vaadin.addon</groupId>
    <artifactId>jpacontainer</artifactId>
</dependency>
<dependency>
    <groupId>org.vaadin</groupId>
    <artifactId>viritin</artifactId>
</dependency>
       <dependency>
              <groupId>javax</groupId>
```

```
                    <artifactId>javaee-web-api</artifactId>
            </dependency>

            <dependency>
                    <groupId>org.apache.velocity</groupId>
                    <artifactId>velocity</artifactId>
            </dependency>
            <dependency>
                    <groupId>org.apache.velocity</groupId>
                    <artifactId>velocity-tools</artifactId>
            </dependency>

            <!--
              Needed when using the widgetset optimizer (custom
ConnectorBundleLoaderFactory).

              For widgetset compilation, vaadin-client-compiler is automatically
added on the
              compilation classpath by vaadin-maven-plugin so normally there is
no need for an
              explicit dependency.
            -->
        <!--
            <dependency>
                    <groupId>com.vaadin</groupId>
                    <artifactId>vaadin-client-compiler</artifactId>
                    <scope>provided</scope>
            </dependency>
            -->

    </dependencies>

    <build>
        <resources>
            <resource>
                <directory>src/main/resources</directory>
                <filtering>true</filtering>
            </resource>
            <resource>
                <directory>src/main/webapp</directory>
                <filtering>true</filtering>
            </resource>
        </resources>
        <plugins>

            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-war-plugin</artifactId>
                <version>2.6</version>
                <configuration>
                    <archive>
                        <manifestEntries>
                            <Vaadin-Package-Version>1</Vaadin-Package-Version>
                            <Vaadin-Widgetsets>gr.eshop7.MyAppWidgetset</Vaadin-
Widgetsets>
                        </manifestEntries>
                    </archive>
                    <failOnMissingWebXml>false</failOnMissingWebXml>
                    <packagingExcludes>
                        WEB-INF/classes/VAADIN/gwt-unitCache/**,
                        WEB-INF/classes/VAADIN/widgetsets/WEB-INF/**
                    </packagingExcludes>
                    <webResources>
                        <resource>
                            <directory>${basedir}/src/main/webapp</directory>
                            <filtering>true</filtering>
```

```
                            <excludes>
                                <exclude>**/VAADIN/**</exclude>
                            </excludes>
                        </resource>
                    </webResources>
                </configuration>
            </plugin>

            <plugin>
                <groupId>com.vaadin</groupId>
                <artifactId>vaadin-maven-plugin</artifactId>
                <version>${vaadin.plugin.version}</version>
                <configuration>
                    <extraJvmArgs>-Xmx512M -Xss1024k</extraJvmArgs>

<webappDirectory>${basedir}/target/classes/VAADIN/widgetsets</webappDirectory>
                    <draftCompile>false</draftCompile>
                    <compileReport>false</compileReport>
                    <style>OBF</style>
                    <strict>true</strict>
                </configuration>
                <executions>
                    <execution>
                        <goals>
                            <!--<goal>update-widgetset</goal>-->
                            <goal>compile</goal>
                            <!-- disabled by default to use on-the-fly theme
compilation -->
                            <!-- <goal>compile-theme</goal> -->
                        </goals>
                    </execution>
                </executions>
            </plugin>

            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-clean-plugin</artifactId>
                <version>2.6.1</version>
                <!-- Clean up also any pre-compiled themes -->
                <configuration>
                    <filesets>
                        <fileset>
                            <directory>src/main/webapp/VAADIN/themes</directory>
                            <includes>
                                <include>**/styles.css</include>
                                <include>**/styles.scss.cache</include>
                            </includes>
                        </fileset>
                    </filesets>
                </configuration>
            </plugin>

        </plugins>

    </build>

</project>
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\AbstractAdminView.java

```
package gr.eshop7.eshop.admin;

import com.vaadin.navigator.View;
import com.vaadin.server.FontAwesome;
import com.vaadin.ui.Button;
```

```
import com.vaadin.ui.Component;
import com.vaadin.ui.HorizontalSplitPanel;
import com.vaadin.ui.Label;
import com.vaadin.ui.Panel;
import com.vaadin.ui.VerticalLayout;
import com.vaadin.ui.themes.ValoTheme;
import gr.eshop7.eshop.admin.util.MenuItem;
import gr.eshop7.eshop.admin.util.MetroTheme;
import gr.eshop7.eshop.admin.util.UIUtils;
import gr.eshop7.eshop.admin.util.Views;

import javax.annotation.PostConstruct;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
public abstract class AbstractAdminView extends VerticalLayout implements View {
	private final MenuItem[] items = new MenuItem[]{
			new MenuItem("Home", FontAwesome.HOME, Views.MAIN),
			new MenuItem("Categories", FontAwesome.FOLDER_OPEN_O,
Views.CATEGORY),
			new MenuItem("Manufacturers", FontAwesome.TRUCK,
Views.MANUFACTURER),
			new MenuItem("Products", FontAwesome.FILE_O, Views.PRODUCT),
			new MenuItem("Stores", FontAwesome.BUILDING_O, Views.STORE),
			new MenuItem("Customers", FontAwesome.USERS, Views.CUSTOMER),
			new MenuItem("Orders", FontAwesome.MONEY, Views.ORDER),
			new MenuItem("Stock", FontAwesome.ARCHIVE, Views.STOCK),
			new MenuItem("Reports", FontAwesome.BAR_CHART_O,
Views.STATISTICS),
	};

	private String title;

	public AbstractAdminView(String title) {
		this.title = title;

	}

	@PostConstruct
	public void postConstruct() {
		initUI();
	}

	private void initUI() {
		setSizeFull();

		addComponent(initHeader());

		HorizontalSplitPanel mainLayout = new HorizontalSplitPanel();
		addComponent(mainLayout);

		mainLayout.addComponent(initMenu());
		mainLayout.addComponent(initContent());
		mainLayout.setSplitPosition(300, Unit.PIXELS);

		setExpandRatio(mainLayout, 1.0f);

		addComponent(initFooter());
	}

	protected Component initHeader() {
		Panel panel = new Panel();
		panel.setIcon(UIUtils.themeImg(MetroTheme.LOGO_SMALL));
```

```
		panel.setCaption(title);
		panel.addStyleName(MetroTheme.PANEL_CAPTION_TEXT_LARGE);

		return panel;
	}

	protected Component initFooter() {
		Label footerLabel = new Label("Graduate Thesis, George Genetzakis,
Niki Mallia");
		footerLabel.addStyleName(ValoTheme.LABEL_SMALL);
		footerLabel.setHeight("30px");

		return footerLabel;
	}

	private Component initMenu() {
		VerticalLayout layout = new VerticalLayout();
		layout.setSpacing(true);
		layout.setMargin(true);
		layout.setWidth(300, Unit.PIXELS);

		for(final MenuItem item: items) {
			Button b = new Button(item.getName());
			b.setWidth(100, Unit.PERCENTAGE);
			b.setIcon(item.getIcon());
			b.addStyleName(ValoTheme.BUTTON_FRIENDLY);

			if(item.hasAction()) {
				b.addClickListener(e -> item.action());
			}

			layout.addComponent(b);
		}

		Button exitButton = new Button("View frontend");
		exitButton.setWidth(100, Unit.PERCENTAGE);
		exitButton.setIcon(FontAwesome.ARROW_LEFT);
		exitButton.addClickListener(e ->
getUI().getPage().setLocation("http://localhost:8080/eshop7/"));
		exitButton.addStyleName(ValoTheme.BUTTON_FRIENDLY);
		layout.addComponent(exitButton);

		return layout;
	}

	protected abstract Component initContent();
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\AbstractEditor.java

```
package gr.eshop7.eshop.admin;

import com.vaadin.addon.jpacontainer.EntityItem;
import com.vaadin.data.fieldgroup.FieldGroup;
import com.vaadin.server.FontAwesome;
import com.vaadin.shared.ui.window.WindowMode;
import com.vaadin.ui.FormLayout;
import com.vaadin.ui.Window;
import com.vaadin.ui.themes.ValoTheme;
import org.vaadin.viritin.button.MButton;
import org.vaadin.viritin.layouts.MHorizontalLayout;

/**
 * Created on 19/4/2015.
 *
```

```
 * @author george
 */
public abstract class AbstractEditor extends Window {
	protected EntityItem item;
	protected ItemEditorContainer itemEditorContainer;
	protected FieldGroup fieldGroup;

	public AbstractEditor(String title, EntityItem item, ItemEditorContainer
itemEditorContainer) {
		super(title);
		this.item = item;
		this.itemEditorContainer = itemEditorContainer;

		initLayout();

		setModal(true);
		setWindowMode(WindowMode.NORMAL);
	}

	private void initLayout() {
		FormLayout formLayout = new FormLayout();
		formLayout.setMargin(true);

		fieldGroup = new FieldGroup(item);

		initComponents(formLayout);

		MButton saveButton = new MButton("Save", e -> onSave())
				.withIcon(FontAwesome.SAVE)
				.withStyleName(ValoTheme.BUTTON_PRIMARY);

		MButton cancelButton = new MButton("Cancel", e -> close())
				.withIcon(FontAwesome.TIMES)
				.withStyleName(ValoTheme.BUTTON_FRIENDLY);

		formLayout.addComponent(new MHorizontalLayout(saveButton,
cancelButton));

		setContent(formLayout);
	}

	protected abstract void initComponents(FormLayout formLayout);

	protected void onSave() {
		try {
			fieldGroup.commit();

			if(itemEditorContainer != null) {
				itemEditorContainer.onSave(item);
			}
			close();
		} catch (FieldGroup.CommitException e) {
			e.printStackTrace();
		}

	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\AdminMainView.java

```
package gr.eshop7.eshop.admin;

import com.vaadin.cdi.CDIView;
import com.vaadin.navigator.ViewChangeListener;
import com.vaadin.ui.Component;
```

```
import com.vaadin.ui.Label;
import com.vaadin.ui.VerticalLayout;
import gr.eshop7.eshop.admin.util.TableHelpers;
import gr.eshop7.eshop.admin.util.Views;
import gr.eshop7.eshop.services.StatisticsService;
import org.vaadin.viritin.label.Header;

import javax.inject.Inject;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
@CDIView(Views.MAIN)
public class AdminMainView extends AbstractAdminView {

	@Inject
	private StatisticsService statisticsService;

	public AdminMainView() {
		super("Administration - Main Page");
	}

	@Override
	public void enter(ViewChangeListener.ViewChangeEvent viewChangeEvent) {
	}

	@Override
	protected Component initContent() {
		VerticalLayout layout = new VerticalLayout();
		layout.setWidth(100, Unit.PERCENTAGE);
		layout.setMargin(true);

		Header header = new Header("Admin area");
		layout.addComponent(header);

		Label label = new Label();
		label.setCaptionAsHtml(true);
		label.setCaption("<p>Welcome to the administration area.</p><p>Please
choose an action from the left sidebar.</p><br>");
		layout.addComponent(label);

		Header header2 = new Header("Latest submitted
orders").setHeaderLevel(3);
		layout.addComponent(header2);

		TableHelpers.addLatestOrders(layout, statisticsService);

		return layout;
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\category\CategoryEditor.java

```
package gr.eshop7.eshop.admin.category;

import com.vaadin.addon.jpacontainer.EntityItem;
import com.vaadin.ui.CheckBox;
import com.vaadin.ui.FormLayout;
import com.vaadin.ui.TextField;
import gr.eshop7.eshop.admin.AbstractEditor;
import gr.eshop7.eshop.admin.ItemEditorContainer;

/**
```

```
 * Created on 19/4/2015.
 *
 * @author george
 */
public class CategoryEditor extends AbstractEditor {

	public CategoryEditor(EntityItem item, ItemEditorContainer
itemEditorContainer) {
		super("Category editor", item, itemEditorContainer);

		setWidth(400, Unit.PIXELS);
	}

	protected void initComponents(FormLayout formLayout) {
		TextField nameField = ((TextField) fieldGroup.buildAndBind("name"));
		nameField.setNullRepresentation("");
		formLayout.addComponent(nameField);

		CheckBox publishedField = ((CheckBox)
fieldGroup.buildAndBind("published"));
		formLayout.addComponent(publishedField);
	}


}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\category\CategoryView.java

```
package gr.eshop7.eshop.admin.category;

import com.vaadin.addon.jpacontainer.EntityItem;
import com.vaadin.addon.jpacontainer.EntityItemProperty;
import com.vaadin.addon.jpacontainer.JPAContainer;
import com.vaadin.addon.jpacontainer.provider.jndijta.CachingMutableEntityProvider;
import com.vaadin.cdi.CDIView;
import com.vaadin.navigator.ViewChangeListener;
import com.vaadin.server.FontAwesome;
import com.vaadin.ui.Alignment;
import com.vaadin.ui.Button;
import com.vaadin.ui.Component;
import com.vaadin.ui.Table;
import com.vaadin.ui.UI;
import com.vaadin.ui.themes.ValoTheme;
import gr.eshop7.eshop.admin.AbstractAdminView;
import gr.eshop7.eshop.admin.ItemEditorContainer;
import gr.eshop7.eshop.admin.util.ActionsColumnsGenerator;
import gr.eshop7.eshop.admin.util.BooleanColumnGenerator;
import gr.eshop7.eshop.admin.util.PublishedCellStyleGenerator;
import gr.eshop7.eshop.admin.util.Views;
import org.eshop7.model.Category;
import org.vaadin.viritin.layouts.MHorizontalLayout;
import org.vaadin.viritin.layouts.MVerticalLayout;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
@CDIView(Views.CATEGORY)
public class CategoryView extends AbstractAdminView implements ItemEditorContainer
{
	private JPAContainer<Category> container;
	private Table table;

	public CategoryView() {
```

```
		super("Administration - Categories");
	}

	@Override
	protected Component initContent() {
		container = new JPAContainer<>(Category.class);
		container.setEntityProvider(new
CachingMutableEntityProvider<>(Category.class));

		table = new Table("Categories", container);
		table.setImmediate(true);
		table.setSizeFull();
		table.setSelectable(true);
		table.setMultiSelect(false);
		table.setSortEnabled(true);
		table.setNullSelectionAllowed(true);
		table.setCellStyleGenerator(new PublishedCellStyleGenerator());
		table.setVisibleColumns("id", "name", "published");
		table.addGeneratedColumn("actions", new
ActionsColumnsGenerator(this));
		table.addGeneratedColumn("published", new BooleanColumnGenerator());
		table.setColumnHeaders("id", "Name", "Published", "Actions");
		table.setRowHeaderMode(Table.RowHeaderMode.INDEX);
		table.setColumnAlignments(Table.Align.CENTER, Table.Align.LEFT,
Table.Align.CENTER, Table.Align.CENTER);
		table.setColumnExpandRatio("name", 1.0f);

		Button addButton = new Button("New category", FontAwesome.PLUS);
		addButton.addStyleName(ValoTheme.BUTTON_PRIMARY);
		addButton.addClickListener(event -> {
			CategoryEditor editor = new
CategoryEditor(container.createEntityItem(new Category()), this);
			UI.getCurrent().addWindow(editor);
		});

		MVerticalLayout layout = new MVerticalLayout()
				.add(new MHorizontalLayout(addButton)
						.withFullWidth())
				.expand(table)
				.withAlign(table, Alignment.MIDDLE_CENTER)
				.withFullHeight()
				.withSpacing(true);

		return layout;
	}

	@Override
	public void enter(ViewChangeListener.ViewChangeEvent event) {

	}

	@Override
	public void onSave(EntityItem item) {
		container.addEntity((Category)item.getEntity());
	}

	@Override
	public void edit(Object itemId) {
		CategoryEditor editor = new CategoryEditor(container.getItem(itemId),
this);
		UI.getCurrent().addWindow(editor);
	}

	@Override
	public void delete(Object itemId) {
```

```
                EntityItemProperty published =
container.getItem(itemId).getItemProperty("published");
                published.setValue(!(Boolean) published.getValue());
                table.refreshRowCache();
        }
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\customer\CustomerEditor.java

```
package gr.eshop7.eshop.admin.customer;

import com.vaadin.addon.jpacontainer.EntityItem;
import com.vaadin.ui.CheckBox;
import com.vaadin.ui.FormLayout;
import com.vaadin.ui.PasswordField;
import com.vaadin.ui.TextField;
import gr.eshop7.eshop.admin.AbstractEditor;
import gr.eshop7.eshop.admin.ItemEditorContainer;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
public class CustomerEditor extends AbstractEditor {

        public CustomerEditor(EntityItem item, ItemEditorContainer
itemEditorContainer) {
                super("Customer editor", item, itemEditorContainer);

                setWidth(400, Unit.PIXELS);
        }

        protected void initComponents(FormLayout formLayout) {
                TextField emailField = ((TextField)
fieldGroup.buildAndBind("email"));
                emailField.setNullRepresentation("");
                formLayout.addComponent(emailField);

                CheckBox publishedField = ((CheckBox)
fieldGroup.buildAndBind("published"));
                formLayout.addComponent(publishedField);

                PasswordField passwordField = fieldGroup.buildAndBind("Password",
"password", PasswordField.class);
                passwordField.setNullRepresentation("");
                formLayout.addComponent(passwordField);

                TextField nameField = ((TextField) fieldGroup.buildAndBind("name"));
                nameField.setNullRepresentation("");
                formLayout.addComponent(nameField);

                TextField addressField = ((TextField)
fieldGroup.buildAndBind("address"));
                addressField.setNullRepresentation("");
                formLayout.addComponent(addressField);

                TextField phoneField = ((TextField)
fieldGroup.buildAndBind("phone"));
                phoneField.setNullRepresentation("");
                formLayout.addComponent(phoneField);

                CheckBox adminField = ((CheckBox) fieldGroup.buildAndBind("admin"));
                formLayout.addComponent(adminField);
```

```
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\customer\CustomerView.java

```
package gr.eshop7.eshop.admin.customer;

import com.vaadin.addon.jpacontainer.EntityItem;
import com.vaadin.addon.jpacontainer.EntityItemProperty;
import com.vaadin.addon.jpacontainer.JPAContainer;
import com.vaadin.addon.jpacontainer.provider.jndijta.CachingMutableEntityProvider;
import com.vaadin.cdi.CDIView;
import com.vaadin.navigator.ViewChangeListener;
import com.vaadin.server.FontAwesome;
import com.vaadin.server.Page;
import com.vaadin.ui.*;
import com.vaadin.ui.themes.ValoTheme;
import gr.eshop7.eshop.admin.AbstractAdminView;
import gr.eshop7.eshop.admin.ItemEditorContainer;
import gr.eshop7.eshop.admin.util.ActionsColumnsGenerator;
import gr.eshop7.eshop.admin.util.BooleanColumnGenerator;
import gr.eshop7.eshop.admin.util.PublishedCellStyleGenerator;
import gr.eshop7.eshop.admin.util.Views;
import org.eshop7.model.Customer;
import org.vaadin.viritin.layouts.MHorizontalLayout;
import org.vaadin.viritin.layouts.MVerticalLayout;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
@CDIView(Views.CUSTOMER)
public class CustomerView extends AbstractAdminView implements ItemEditorContainer
{
	private JPAContainer<Customer> container;
	private Table table;

	public CustomerView() {
		super("Administration - Customers");
	}

	@Override
	protected Component initContent() {
		container = new JPAContainer<>(Customer.class);
		container.setEntityProvider(new
CachingMutableEntityProvider<>(Customer.class));

		table = new Table("Customers", container);
		table.setImmediate(true);
		table.setSizeFull();
		table.setSelectable(true);
		table.setMultiSelect(false);
		table.setSortEnabled(true);
		table.setNullSelectionAllowed(true);
		table.setCellStyleGenerator(new PublishedCellStyleGenerator());
		table.setVisibleColumns("id", "name", "email", "admin", "published");
		table.addGeneratedColumn("actions", new
ActionsColumnsGenerator(this));
		table.addGeneratedColumn("published", new BooleanColumnGenerator());
		table.addGeneratedColumn("admin", new BooleanColumnGenerator());
		table.setColumnHeaders("id", "Name", "Email", "Admin", "Published",
"Actions");
```

```
		table.setRowHeaderMode(Table.RowHeaderMode.INDEX);
		table.setColumnAlignments(Table.Align.CENTER, Table.Align.LEFT,
Table.Align.LEFT, Table.Align.CENTER, Table.Align.CENTER, Table.Align.CENTER);
		table.setColumnExpandRatio("name", 1.0f);

		Button addButton = new Button("New customer", FontAwesome.PLUS);
		addButton.addStyleName(ValoTheme.BUTTON_PRIMARY);
		addButton.addClickListener(event -> {
			CustomerEditor editor = new
CustomerEditor(container.createEntityItem(new Customer()), this);
			UI.getCurrent().addWindow(editor);
		});

		MVerticalLayout layout = new MVerticalLayout()
				.add(new MHorizontalLayout(addButton)
						.withFullWidth())
				.expand(table)
				.withAlign(table, Alignment.MIDDLE_CENTER)
				.withFullHeight()
				.withSpacing(true);

		return layout;
	}

	@Override
	public void enter(ViewChangeListener.ViewChangeEvent event) {

	}

	@Override
	public void onSave(EntityItem item) {
		container.addEntity((Customer)item.getEntity());
	}

	@Override
	public void edit(Object itemId) {
		CustomerEditor editor = new CustomerEditor(container.getItem(itemId),
this);
		UI.getCurrent().addWindow(editor);
	}

	@Override
	public void delete(Object itemId) {
		EntityItemProperty admin =
container.getItem(itemId).getItemProperty("admin");
		if(!((Boolean) admin.getValue())) {
			EntityItemProperty published =
container.getItem(itemId).getItemProperty("published");
			published.setValue(!(Boolean) published.getValue());
			table.refreshRowCache();

		} else {
			new Notification("Operation refused",
						"<br/>You cannot unpublished administrators.",
						Notification.Type.ERROR_MESSAGE, true)
						.show(Page.getCurrent());
		}
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\ItemEditorContainer.java

```
package gr.eshop7.eshop.admin;

import com.vaadin.addon.jpacontainer.EntityItem;
```

```
/**
 * Created on 19/4/2015.
 *
 * @author george
 */
public interface ItemEditorContainer {
	void onSave(EntityItem item);
	void edit(Object itemId);
	void delete(Object itemId);

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\manufacturer\ManufacturerEditor.java

```
package gr.eshop7.eshop.admin.manufacturer;

import com.vaadin.addon.jpacontainer.EntityItem;
import com.vaadin.ui.CheckBox;
import com.vaadin.ui.FormLayout;
import com.vaadin.ui.TextField;
import gr.eshop7.eshop.admin.AbstractEditor;
import gr.eshop7.eshop.admin.ItemEditorContainer;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
public class ManufacturerEditor extends AbstractEditor {

	public ManufacturerEditor(EntityItem item, ItemEditorContainer
itemEditorContainer) {
		super("Manufacturer editor", item, itemEditorContainer);

		setWidth(400, Unit.PIXELS);

	}

	protected void initComponents(FormLayout formLayout) {
		TextField nameField = ((TextField) fieldGroup.buildAndBind("name"));
		nameField.setNullRepresentation("");
		formLayout.addComponent(nameField);

		CheckBox publishedField = ((CheckBox)
fieldGroup.buildAndBind("published"));
		formLayout.addComponent(publishedField);
	}


}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\manufacturer\ManufacturerView.java

```
package gr.eshop7.eshop.admin.manufacturer;

import com.vaadin.addon.jpacontainer.EntityItem;
import com.vaadin.addon.jpacontainer.EntityItemProperty;
import com.vaadin.addon.jpacontainer.JPAContainer;
import com.vaadin.addon.jpacontainer.provider.jndijta.CachingMutableEntityProvider;
import com.vaadin.cdi.CDIView;
import com.vaadin.navigator.ViewChangeListener;
import com.vaadin.server.FontAwesome;
```

```
import com.vaadin.ui.*;
import com.vaadin.ui.themes.ValoTheme;
import gr.eshop7.eshop.admin.AbstractAdminView;
import gr.eshop7.eshop.admin.ItemEditorContainer;
import gr.eshop7.eshop.admin.util.ActionsColumnsGenerator;
import gr.eshop7.eshop.admin.util.BooleanColumnGenerator;
import gr.eshop7.eshop.admin.util.PublishedCellStyleGenerator;
import gr.eshop7.eshop.admin.util.Views;
import org.eshop7.model.Manufacturer;
import org.vaadin.viritin.layouts.MHorizontalLayout;
import org.vaadin.viritin.layouts.MVerticalLayout;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
@CDIView(Views.MANUFACTURER)
public class ManufacturerView extends AbstractAdminView implements
ItemEditorContainer {
	private JPAContainer<Manufacturer> container;
	private Table table;

	public ManufacturerView() {
		super("Administration - Manufacturers");
	}

	@Override
	protected Component initContent() {
		container = new JPAContainer<>(Manufacturer.class);
		container.setEntityProvider(new
CachingMutableEntityProvider<>(Manufacturer.class));

		table = new Table("Manufacturers", container);
		table.setImmediate(true);
		table.setSizeFull();
		table.setSelectable(true);
		table.setMultiSelect(false);
		table.setSortEnabled(true);
		table.setNullSelectionAllowed(true);
		table.setCellStyleGenerator(new PublishedCellStyleGenerator());
		table.setVisibleColumns("id", "name", "published");
		table.addGeneratedColumn("actions", new
ActionsColumnsGenerator(this));
		table.addGeneratedColumn("published", new BooleanColumnGenerator());
		table.setColumnHeaders("id", "Name", "Published", "Actions");
		table.setRowHeaderMode(Table.RowHeaderMode.INDEX);
		table.setColumnAlignments(Table.Align.CENTER, Table.Align.LEFT,
Table.Align.CENTER, Table.Align.CENTER);
		table.setColumnExpandRatio("name", 1.0f);

		Button addButton = new Button("New manufacturer", FontAwesome.PLUS);
		addButton.addStyleName(ValoTheme.BUTTON_PRIMARY);
		addButton.addClickListener(event -> {
			ManufacturerEditor editor = new
ManufacturerEditor(container.createEntityItem(new Manufacturer()), this);
			UI.getCurrent().addWindow(editor);
		});

		MVerticalLayout layout = new MVerticalLayout()
				.add(new MHorizontalLayout(addButton)
						.withFullWidth())
				.expand(table)
				.withAlign(table, Alignment.MIDDLE_CENTER)
				.withFullHeight()
				.withSpacing(true);
```

```
		return layout;
	}

	@Override
	public void enter(ViewChangeListener.ViewChangeEvent event) {

	}

	@Override
	public void onSave(EntityItem item) {
		container.addEntity((Manufacturer)item.getEntity());
	}

	@Override
	public void edit(Object itemId) {
		ManufacturerEditor editor = new
ManufacturerEditor(container.getItem(itemId), this);
		UI.getCurrent().addWindow(editor);
	}

	@Override
	public void delete(Object itemId) {
		EntityItemProperty published =
container.getItem(itemId).getItemProperty("published");
		published.setValue(!(Boolean) published.getValue());
		table.refreshRowCache();
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\order\OrderView.java

```
package gr.eshop7.eshop.admin.order;

import com.vaadin.addon.jpacontainer.JPAContainer;
import com.vaadin.addon.jpacontainer.provider.jndijta.CachingMutableEntityProvider;
import com.vaadin.cdi.CDIView;
import com.vaadin.data.util.converter.StringToDateConverter;
import com.vaadin.navigator.ViewChangeListener;
import com.vaadin.ui.Alignment;
import com.vaadin.ui.Component;
import com.vaadin.ui.Notification;
import com.vaadin.ui.Table;
import gr.eshop7.eshop.admin.AbstractAdminView;
import gr.eshop7.eshop.admin.OrderContainer;
import gr.eshop7.eshop.admin.util.OrderActionsColumnsGenerator;
import gr.eshop7.eshop.admin.util.Views;
import gr.eshop7.eshop.services.StockService;
import org.eshop7.model.CustomerOrder;
import org.eshop7.model.OrderStatus;
import org.eshop7.model.Product;
import org.vaadin.viritin.layouts.MVerticalLayout;

import javax.inject.Inject;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
@CDIView(Views.ORDER)
```

```
public class OrderView extends AbstractAdminView implements OrderContainer {
	private JPAContainer<CustomerOrder> container;
	private Table table;

	@Inject
	private StockService stockService;

	public OrderView() {
		super("Administration - Orders");
	}

	@Override
	protected Component initContent() {
		container = new JPAContainer<>(CustomerOrder.class);
		container.setEntityProvider(new
CachingMutableEntityProvider<>(CustomerOrder.class));

		table = new Table("Orders", container);
		table.setImmediate(true);
		table.setSizeFull();
		table.setSelectable(true);
		table.setMultiSelect(false);
		table.setSortEnabled(true);
		table.setNullSelectionAllowed(true);
		table.setVisibleColumns("ref", "customer", "price", "product",
"submissionDate", "orderStatus", "deliveryDate", "store");
		table.addGeneratedColumn("actions", new
OrderActionsColumnsGenerator(this));
		table.setColumnHeaders("Code", "Customer", "Price", "Product",
"Submitted", "Status", "Delivered", "Store", "Actions");
		table.setRowHeaderMode(Table.RowHeaderMode.INDEX);
		table.setColumnAlignments(Table.Align.LEFT, Table.Align.LEFT,
Table.Align.CENTER, Table.Align.LEFT, Table.Align.CENTER, Table.Align.LEFT,
Table.Align.LEFT, Table.Align.LEFT, Table.Align.CENTER);
		table.setColumnExpandRatio("customer", 1.0f);

		table.setConverter("submissionDate", new StringToDateConverter() {
			@Override
			protected DateFormat getFormat(Locale locale) {
				return new SimpleDateFormat("dd/MM/YYYY, HH:mm");
			}
		});
		table.setConverter("deliveryDate", new StringToDateConverter() {
			@Override
			protected DateFormat getFormat(Locale locale) {
				return new SimpleDateFormat("dd/MM/YYYY, HH:mm");
			}
		});

		MVerticalLayout layout = new MVerticalLayout()
				.expand(table)
				.withAlign(table, Alignment.MIDDLE_CENTER)
				.withFullHeight()
				.withSpacing(true);

		return layout;
	}

	@Override
	public void setStatusCancelled(Object itemId) {

	container.getItem(itemId).getItemProperty("orderStatus").setValue(OrderStatu
s.CANCELLED);
		Product product = (Product)
container.getItem(itemId).getItemProperty("product");
		stockService.increaseStock(product);
```

```
                Notification.show("Order cancelled",
Notification.Type.TRAY_NOTIFICATION);
        }

        @Override
        public void setStatusSent(Object itemId) {

        container.getItem(itemId).getItemProperty("deliveryDate").setValue(new
Date());

        container.getItem(itemId).getItemProperty("orderStatus").setValue(OrderStatu
s.DELIVERED);
                Notification.show("Order updated",
Notification.Type.TRAY_NOTIFICATION);
        }

        @Override
        public void setStatusInProgress(Object itemId) {

        container.getItem(itemId).getItemProperty("orderStatus").setValue(OrderStatu
s.IN_PROGRESS);
                Notification.show("Order now In Progress",
Notification.Type.TRAY_NOTIFICATION);
        }

        @Override
        public void enter(ViewChangeListener.ViewChangeEvent event) {

        }
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\OrderContainer.java

```
package gr.eshop7.eshop.admin;

/**
 * Created by george on 04/12/2015.
 *
 */
public interface OrderContainer {
        void setStatusCancelled(Object itemId);
        void setStatusSent(Object itemId);
        void setStatusInProgress(Object itemId);
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\product\ProductEditor.java

```
package gr.eshop7.eshop.admin.product;

import com.vaadin.addon.jpacontainer.EntityItem;
import com.vaadin.addon.jpacontainer.JPAContainer;
import com.vaadin.addon.jpacontainer.fieldfactory.SingleSelectConverter;
import com.vaadin.addon.jpacontainer.provider.jndijta.CachingMutableEntityProvider;
import com.vaadin.server.FileResource;
import com.vaadin.ui.*;
import gr.eshop7.eshop.Config;
import gr.eshop7.eshop.admin.AbstractEditor;
import gr.eshop7.eshop.admin.ItemEditorContainer;
import gr.eshop7.eshop.admin.util.UIUtils;
import gr.eshop7.eshop.util.Utils;
import org.eshop7.model.Category;
import org.eshop7.model.Manufacturer;
import org.eshop7.model.Product;
```

```
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
public class ProductEditor extends AbstractEditor {
	private TextField imageField;
	private Image imageView;

	public ProductEditor(EntityItem<Product> item, ItemEditorContainer
container) {
		super("Product editor", item, container);

		setWidth(800, Unit.PIXELS);
	}

	@Override
	protected void initComponents(FormLayout formLayout) {
		TextField nameField = ((TextField) fieldGroup.buildAndBind("name"));
		nameField.setWidth(700, Unit.PIXELS);
		nameField.setNullRepresentation("");
		formLayout.addComponent(nameField);

		CheckBox publishedField = ((CheckBox)
fieldGroup.buildAndBind("published"));
		formLayout.addComponent(publishedField);

		RichTextArea descriptionField = new RichTextArea();
		descriptionField.setNullRepresentation("");
		descriptionField.setWidth(700, Unit.PIXELS);
		fieldGroup.bind(descriptionField, "description");
		formLayout.addComponent(descriptionField);

		JPAContainer<Manufacturer> manufacturerContainer = new
JPAContainer<>(Manufacturer.class);
		manufacturerContainer.setEntityProvider(new
CachingMutableEntityProvider<>(Manufacturer.class));

		ComboBox manufacturerCB = new ComboBox("Manufacturer",
manufacturerContainer);
		manufacturerCB.setNullSelectionAllowed(false);
		manufacturerCB.setItemCaptionPropertyId("name");
		manufacturerCB.setConverter(new
SingleSelectConverter<Manufacturer>(manufacturerCB));
		fieldGroup.bind(manufacturerCB, "manufacturer");
		formLayout.addComponent(manufacturerCB);

		JPAContainer<Category> categoryContainer = new
JPAContainer<>(Category.class);
		categoryContainer.setEntityProvider(new
CachingMutableEntityProvider<>(Category.class));

		ComboBox categoryCB = new ComboBox("Category", categoryContainer);
		categoryCB.setNullSelectionAllowed(false);
		categoryCB.setItemCaptionPropertyId("name");
		categoryCB.setConverter(new
SingleSelectConverter<Category>(categoryCB));
		fieldGroup.bind(categoryCB, "category");
		formLayout.addComponent(categoryCB);


		formLayout.addComponent(fieldGroup.buildAndBind("price"));
```

```
		initImageField(formLayout);
	}

	private void initImageField(FormLayout formLayout) {
		final long itemId = item.getItemId() == null ? 0 : (long)
item.getItemId();

		if (itemId == 0) {
			Label label = new Label("Save the product before adding an
image");
			formLayout.addComponent(label);

			return;
		}

		imageField = new TextField("Image");
		imageField.setNullRepresentation("No image");
		imageField.setReadOnly(true);
		fieldGroup.bind(imageField, "image");
		formLayout.addComponent(imageField);

		Upload upload = new Upload("Image", (filename, mimeType) -> {
			String imageName = UIUtils.generateImageName(itemId,
mimeType);

			try {
				FileOutputStream outputStream = new
FileOutputStream(UIUtils.generateImagePath(itemId, mimeType));
				imageField.setValue(imageName);

				return outputStream;
			} catch (FileNotFoundException e) {
				e.printStackTrace();
			}
			return null;
		});
		upload.addSucceededListener(event -> imageView.setSource(new
FileResource(new File(Config.PRODUCT_IMAGES_BASE + "/" + imageField.getValue()))));
		formLayout.addComponent(upload);

		imageView = new Image();
		imageView.setWidth(150, Unit.PIXELS);
		imageView.setHeight(100, Unit.PIXELS);
		if (!Utils.isEmpty((String)
item.getItemProperty("image").getValue())) {

	imageView.setSource(UIUtils.getImg((Product)item.getEntity()));
		}
		formLayout.addComponent(imageView);
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\product\ProductView.java

```
package gr.eshop7.eshop.admin.product;

import com.vaadin.addon.jpacontainer.EntityItem;
import com.vaadin.addon.jpacontainer.EntityItemProperty;
import com.vaadin.addon.jpacontainer.JPAContainer;
import com.vaadin.addon.jpacontainer.provider.jndijta.CachingMutableEntityProvider;
import com.vaadin.cdi.CDIView;
import com.vaadin.navigator.ViewChangeListener;
import com.vaadin.server.FontAwesome;
```

```
import com.vaadin.ui.Alignment;
import com.vaadin.ui.Button;
import com.vaadin.ui.Component;
import com.vaadin.ui.Table;
import com.vaadin.ui.UI;
import com.vaadin.ui.themes.ValoTheme;
import gr.eshop7.eshop.admin.AbstractAdminView;
import gr.eshop7.eshop.admin.ItemEditorContainer;
import gr.eshop7.eshop.admin.util.ActionsColumnsGenerator;
import gr.eshop7.eshop.admin.util.BooleanColumnGenerator;
import gr.eshop7.eshop.admin.util.PublishedCellStyleGenerator;
import gr.eshop7.eshop.admin.util.Views;
import org.eshop7.model.Product;
import org.vaadin.viritin.layouts.MHorizontalLayout;
import org.vaadin.viritin.layouts.MVerticalLayout;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
@CDIView(Views.PRODUCT)
public class ProductView extends AbstractAdminView implements ItemEditorContainer {
	private JPAContainer<Product> container;
	private Table table;

	public ProductView() {
		super("Administration - Products");
	}

	@Override
	protected Component initContent() {
		container = new JPAContainer<>(Product.class);
		container.setEntityProvider(new
CachingMutableEntityProvider<>(Product.class));
		container.addNestedContainerProperty("category.name");
		container.addNestedContainerProperty("manufacturer.name");

		table = new Table("Products", container);
		table.setImmediate(true);
		table.setSizeFull();
		table.setSelectable(true);
		table.setMultiSelect(false);
		table.setSortEnabled(true);
		table.setNullSelectionAllowed(true);
		table.setCellStyleGenerator(new PublishedCellStyleGenerator());
		table.setVisibleColumns("id", "name", "price", "category.name",
"manufacturer.name", "published");
		table.addGeneratedColumn("actions", new
ActionsColumnsGenerator(this));
		table.addGeneratedColumn("published", new BooleanColumnGenerator());
		table.setColumnHeaders("id", "Name", "Price", "Category",
"Manufacturer", "Published", "Actions");
		table.setRowHeaderMode(Table.RowHeaderMode.INDEX);
		table.setColumnAlignments(Table.Align.CENTER, Table.Align.LEFT,
Table.Align.CENTER, Table.Align.CENTER, Table.Align.CENTER, Table.Align.CENTER,
Table.Align.CENTER);
		table.setColumnExpandRatio("name", 1.0f);

		Button addButton = new Button("New product", FontAwesome.PLUS);
		addButton.addStyleName(ValoTheme.BUTTON_PRIMARY);
		addButton.addClickListener(event -> {
			ProductEditor editor = new
ProductEditor(container.createEntityItem(new Product()), this);
			UI.getCurrent().addWindow(editor);
		});
```

```
		MVerticalLayout layout = new MVerticalLayout()
				.add(new MHorizontalLayout(addButton)
						.withFullWidth())
				.expand(table)
				.withAlign(table, Alignment.MIDDLE_CENTER)
				.withFullHeight()
				.withSpacing(true);

		return layout;
	}

	@Override
	public void onSave(EntityItem item) {
		container.addEntity((Product)item.getEntity());
	}

	@Override
	public void edit(Object itemId) {
		ProductEditor editor = new ProductEditor(container.getItem(itemId),
this);
		UI.getCurrent().addWindow(editor);
	}

	@Override
	public void delete(Object itemId) {
		EntityItemProperty published =
container.getItem(itemId).getItemProperty("published");
		published.setValue(!(Boolean)published.getValue());
		table.refreshRowCache();
	}

	@Override
	public void enter(ViewChangeListener.ViewChangeEvent event) {

	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\statistics\StatisticsView.java

```
package gr.eshop7.eshop.admin.statistics;

import com.vaadin.cdi.CDIView;
import com.vaadin.data.Property;
import com.vaadin.navigator.ViewChangeListener;
import com.vaadin.ui.Alignment;
import com.vaadin.ui.Button;
import com.vaadin.ui.Component;
import com.vaadin.ui.DateField;
import com.vaadin.ui.Notification;
import com.vaadin.ui.Table;
import com.vaadin.ui.VerticalLayout;
import gr.eshop7.eshop.admin.AbstractAdminView;
import gr.eshop7.eshop.admin.util.TableHelpers;
import gr.eshop7.eshop.admin.util.Views;
import gr.eshop7.eshop.services.StatisticsService;
import gr.eshop7.eshop.util.DateUtil;
import gr.eshop7.eshop.util.DoubleValue;
import gr.eshop7.eshop.util.TripleValue;
import org.vaadin.viritin.ListContainer;
import org.vaadin.viritin.button.MButton;
import org.vaadin.viritin.label.Header;
import org.vaadin.viritin.layouts.MFormLayout;
import org.vaadin.viritin.layouts.MHorizontalLayout;
```

```
import javax.inject.Inject;
import java.util.Date;
import java.util.List;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
@CDIView(Views.STATISTICS)
public class StatisticsView extends AbstractAdminView {

	@Inject
	private StatisticsService statisticsService;

	private DateField startDateField, endDateField;
	private Table statsPerStoreTable, statsPerManufacturerTable,
statsPerCategoryTable, statsPerProductTable, meanTimePerStoreTable;
	private Header resultsHeader;
	MFormLayout formLayout;

	public StatisticsView() {
		super("Administration - Statistics");
	}

	@Override
	public void enter(ViewChangeListener.ViewChangeEvent viewChangeEvent) {
	}

	@Override
	protected Component initContent() {
		VerticalLayout layout = new VerticalLayout();
		layout.setWidth(100, Unit.PERCENTAGE);
		layout.setMargin(true);

		Header header = new Header("Reports").setHeaderLevel(2);
		layout.addComponent(header);

		startDateField = new DateField(null,
DateUtil.dayStart(DateUtil.addMonth(-1)));
		startDateField.setDateFormat("dd/MM/yyyy");
		startDateField.addValueChangeListener(p ->
startDateListener(p.getProperty()));

		endDateField = new DateField(null, DateUtil.dayEnd(new Date()));
		endDateField.setDateFormat("dd/MM/yyyy");
		endDateField.addValueChangeListener(p ->
endDateListener(p.getProperty()));

		resultsHeader = new Header("Results").setHeaderLevel(4);

		formLayout = new MFormLayout(
				new MHorizontalLayout(
						endDateField,
						new MButton("Reset", e -> resetDates()))
					.withCaption("End date")
					.alignAll(Alignment.MIDDLE_CENTER),
				new MHorizontalLayout(
						startDateField,
						new MButton("Last month", e ->
setMonthIntervalDates(-1)),
						new MButton("Last 3 months", e ->
setMonthIntervalDates(-3)),
						new MButton("Last 6 months", e ->
setMonthIntervalDates(-6)),
```

```
                                                new MButton("Last year", e ->
setMonthIntervalDates(-12)))
                                        .withCaption("Start date")
                                        .alignAll(Alignment.MIDDLE_CENTER),
                                new Button("Calculate", e -> calculate()),
                                resultsHeader);
                layout.addComponent(formLayout);

                return layout;
        }

        private void addIncomePerStore() {
                if(statsPerStoreTable != null) {
                        formLayout.removeComponent(statsPerStoreTable);
                }

                statsPerStoreTable =
TableHelpers.addIncomePerStore(startDateField.getValue(), endDateField.getValue(),
statisticsService);

                formLayout.addComponent(statsPerStoreTable);
        }

        private void addStatsPerManufacturer() {
                if(statsPerManufacturerTable != null) {
                        formLayout.removeComponent(statsPerManufacturerTable);
                }

                statsPerManufacturerTable =
TableHelpers.addStatsPerManufacturer(startDateField.getValue(),
endDateField.getValue(), statisticsService);

                formLayout.addComponent(statsPerManufacturerTable);
        }

        private void addStatsPerProduct() {
                if(statsPerProductTable != null) {
                        formLayout.removeComponent(statsPerProductTable);
                }

                statsPerProductTable =
TableHelpers.addStatsPerProduct(startDateField.getValue(), endDateField.getValue(),
statisticsService);

                formLayout.addComponent(statsPerProductTable);
        }

        private void addStatsPerCategory() {
                if(statsPerCategoryTable != null) {
                        formLayout.removeComponent(statsPerCategoryTable);
                }

                statsPerCategoryTable =
TableHelpers.addStatsPerCategory(startDateField.getValue(),
endDateField.getValue(), statisticsService);

                formLayout.addComponent(statsPerCategoryTable);
        }

        private void addStatsMeanTimePerStore() {
                if(meanTimePerStoreTable != null) {
                        formLayout.removeComponent(meanTimePerStoreTable);
                }
```

```
		List<DoubleValue> data =
statisticsService.getMeanTimePerStore(startDateField.getValue(),
endDateField.getValue());
		meanTimePerStoreTable = createDoubleTupleTable(data, "Store", "Mean
time");
		meanTimePerStoreTable.setCaption("Mean order completion time");
		meanTimePerStoreTable.setPageLength(data.size() != 0 ? data.size() :
2);

		formLayout.addComponent(meanTimePerStoreTable);
	}

	private Table createDoubleTupleTable(List<DoubleValue> list, String col1,
String col2) {
		Table table = new Table();
		table.setContainerDataSource(new ListContainer<DoubleValue>(list));
		table.setWidth(100, Unit.PERCENTAGE);
		table.setVisibleColumns("label", "value");
		table.setColumnExpandRatio("label", 1.0f);
		table.setColumnHeaders(col1, col2);

		return table;
	}

	private Table createTripleTupleTable(List<TripleValue> list, String col1,
String col2, String col3) {
		Table table = new Table();
		table.setContainerDataSource(new ListContainer<TripleValue>(list));
		table.setWidth(100, Unit.PERCENTAGE);
		table.setVisibleColumns("label", "value1", "value2");
		table.setColumnExpandRatio("label", 1.0f);
		table.setColumnHeaders(col1, col2, col3);

		return table;
	}

	private void resetDates() {
		endDateField.setValue(DateUtil.dayEnd(new Date()));
		startDateField.setValue(DateUtil.dayStart(DateUtil.addMonth(-1)));
	}

	private void setMonthIntervalDates(int monthNo) {
		endDateField.setValue(DateUtil.dayEnd(new Date()));

	startDateField.setValue(DateUtil.dayStart(DateUtil.addMonth(monthNo)));
	}

	private void startDateListener(Property<Date> property) {
		Date value = property.getValue();

		property.setValue(DateUtil.dayStart(value));
	}

	private void endDateListener(Property<Date> property) {
		Date value = property.getValue();

		property.setValue(DateUtil.dayEnd(value));
	}

	private boolean checkDates() {
		Date startDate = startDateField.getValue();
		Date endDate = endDateField.getValue();

		if(startDate.after(endDate)) {
			Notification.show("Please enter valid dates, End date must be
after Start date.", Notification.Type.ERROR_MESSAGE);
```

```
                        return false;
                }

                return true;
        }

        private void calculate() {
                if (checkDates()) {
                        addIncomePerStore();
                        addStatsPerManufacturer();
                        addStatsPerCategory();
                        addStatsPerProduct();
                        addStatsMeanTimePerStore();
                }
        }
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\stock\StockEditor.java

```
package gr.eshop7.eshop.admin.stock;

import com.vaadin.server.FontAwesome;
import com.vaadin.server.Page;
import com.vaadin.shared.ui.window.WindowMode;
import com.vaadin.ui.FormLayout;
import com.vaadin.ui.Notification;
import com.vaadin.ui.TextField;
import com.vaadin.ui.Window;
import com.vaadin.ui.themes.ValoTheme;
import gr.eshop7.eshop.admin.util.dto.StockBean;
import org.vaadin.viritin.button.MButton;
import org.vaadin.viritin.layouts.MHorizontalLayout;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
public class StockEditor extends Window {
        private StockBean stockBean;
        private StockView stockView;
        private TextField stockField;

        public StockEditor(StockView stockView, StockBean stockBean) {
                super("Stock editor");

                this.stockBean = stockBean;
                this.stockView = stockView;

                setWidth(400, Unit.PIXELS);

                initComponents();

                setModal(true);
                setWindowMode(WindowMode.NORMAL);

        }

        protected void initComponents() {
                FormLayout formLayout = new FormLayout();
                formLayout.setMargin(true);

                stockField = new TextField("Stock",
String.valueOf(stockBean.getStock()));
                formLayout.addComponent(stockField);
```

```
		MButton saveButton = new MButton("Save", e -> onSave())
				.withIcon(FontAwesome.SAVE)
				.withStyleName(ValoTheme.BUTTON_PRIMARY);

		MButton cancelButton = new MButton("Cancel", e -> close())
				.withIcon(FontAwesome.TIMES)
				.withStyleName(ValoTheme.BUTTON_FRIENDLY);

		formLayout.addComponent(new MHorizontalLayout(saveButton,
cancelButton));

		setContent(formLayout);
	}

	protected void onSave() {
		try {
			int stock = Integer.parseInt(stockField.getValue());
			if(stock < 0) {
				throw new NumberFormatException();
			}

			stockBean.setStock(stock);

			stockView.onSave(stockBean);
			close();
		} catch(NumberFormatException e) {
			new Notification("Invalid stock",
					"<br/>Please enter a positive number.",
					Notification.Type.ERROR_MESSAGE, true)
					.show(Page.getCurrent());
		}
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\stock\StockView.java

```
package gr.eshop7.eshop.admin.stock;

import com.vaadin.addon.jpacontainer.EntityItem;
import com.vaadin.cdi.CDIView;
import com.vaadin.data.util.BeanContainer;
import com.vaadin.data.util.BeanItem;
import com.vaadin.navigator.ViewChangeListener;
import com.vaadin.ui.Alignment;
import com.vaadin.ui.Component;
import com.vaadin.ui.Table;
import com.vaadin.ui.UI;
import gr.eshop7.eshop.admin.AbstractAdminView;
import gr.eshop7.eshop.admin.ItemEditorContainer;
import gr.eshop7.eshop.admin.util.ActionsColumnsGenerator;
import gr.eshop7.eshop.admin.util.Views;
import gr.eshop7.eshop.admin.util.dto.StockBean;
import gr.eshop7.eshop.services.StockService;
import org.vaadin.viritin.layouts.MVerticalLayout;

import javax.inject.Inject;
import java.util.List;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
@CDIView(Views.STOCK)
```

```
public class StockView extends AbstractAdminView implements ItemEditorContainer {
	@Inject
	private StockService stockService;

	private BeanContainer<String, StockBean> container;
	private Table table;

	public StockView() {
		super("Administration - Products");
	}

	@Override
	protected Component initContent() {
		List<StockBean> allStockBeans = stockService.getAllStockBeans();
		container = new BeanContainer<>(StockBean.class);
		container.setBeanIdProperty("id");
		container.addAll(allStockBeans);

		table = new Table("Stores", container);
		table.setImmediate(true);
		table.setSizeFull();
		table.setSelectable(true);
		table.setMultiSelect(false);
		table.setSortEnabled(true);
		table.setNullSelectionAllowed(true);
		table.addGeneratedColumn("actions", new ActionsColumnsGenerator(this,
true, false));
		table.setVisibleColumns("storeName", "productName", "stock",
"actions");
		table.setColumnHeaders("Store", "Product", "Stock", "Actions");
		table.setRowHeaderMode(Table.RowHeaderMode.INDEX);
		table.setColumnAlignments(Table.Align.LEFT, Table.Align.LEFT,
Table.Align.CENTER, Table.Align.CENTER);
//		table.setColumnExpandRatio("name", 1.0f);
		MVerticalLayout layout = new MVerticalLayout()
				.expand(table)
				.withAlign(table, Alignment.MIDDLE_CENTER)
				.withFullHeight()
				.withSpacing(true);

		return layout;
	}

	public void onSave(StockBean stockBean) {
		stockService.saveStock(stockBean.getStoreId(),
stockBean.getProductId(), stockBean.getStock());

		container.removeAllItems();
		container.addAll(stockService.getAllStockBeans());

		table.refreshRowCache();
	}

	@Override
	public void onSave(EntityItem item) {

	}

	@Override
	public void edit(Object itemId) {
		BeanItem<StockBean> item = container.getItem(itemId);

		StockEditor editor = new StockEditor(this, item.getBean());
		UI.getCurrent().addWindow(editor);
	}
```

```
	@Override
	public void delete(Object itemId) {

	}

	@Override
	public void enter(ViewChangeListener.ViewChangeEvent event) {
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\store\StoreEditor.java

```
package gr.eshop7.eshop.admin.store;

import com.vaadin.addon.jpacontainer.EntityItem;
import com.vaadin.ui.CheckBox;
import com.vaadin.ui.FormLayout;
import com.vaadin.ui.TextField;
import gr.eshop7.eshop.admin.AbstractEditor;
import gr.eshop7.eshop.admin.ItemEditorContainer;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
public class StoreEditor extends AbstractEditor {

	public StoreEditor(EntityItem item, ItemEditorContainer itemEditorContainer)
{
		super("Store editor", item, itemEditorContainer);

		setWidth(400, Unit.PIXELS);
	}

	protected void initComponents(FormLayout formLayout) {
		TextField nameField = ((TextField) fieldGroup.buildAndBind("name"));
		nameField.setNullRepresentation("");
		formLayout.addComponent(nameField);

		CheckBox publishedField = ((CheckBox)
fieldGroup.buildAndBind("published"));
		formLayout.addComponent(publishedField);

		TextField addressField = ((TextField)
fieldGroup.buildAndBind("address"));
		addressField.setNullRepresentation("");
		formLayout.addComponent(addressField);

		TextField phoneField = ((TextField)
fieldGroup.buildAndBind("phone"));
		phoneField.setNullRepresentation("");
		formLayout.addComponent(phoneField);
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\store\StoreView.java

```
package gr.eshop7.eshop.admin.store;

import com.vaadin.addon.jpacontainer.EntityItem;
import com.vaadin.addon.jpacontainer.EntityItemProperty;
import com.vaadin.addon.jpacontainer.JPAContainer;
```

```
import com.vaadin.addon.jpacontainer.provider.jndijta.CachingMutableEntityProvider;
import com.vaadin.cdi.CDIView;
import com.vaadin.navigator.ViewChangeListener;
import com.vaadin.server.FontAwesome;
import com.vaadin.ui.*;
import com.vaadin.ui.themes.ValoTheme;
import gr.eshop7.eshop.admin.AbstractAdminView;
import gr.eshop7.eshop.admin.ItemEditorContainer;
import gr.eshop7.eshop.admin.util.ActionsColumnsGenerator;
import gr.eshop7.eshop.admin.util.BooleanColumnGenerator;
import gr.eshop7.eshop.admin.util.PublishedCellStyleGenerator;
import gr.eshop7.eshop.admin.util.Views;
import gr.eshop7.eshop.services.StoreService;
import org.eshop7.model.Store;
import org.vaadin.viritin.layouts.MHorizontalLayout;
import org.vaadin.viritin.layouts.MVerticalLayout;

import javax.inject.Inject;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
@CDIView(Views.STORE)
public class StoreView extends AbstractAdminView implements ItemEditorContainer {

        @Inject
        private StoreService storeService;

        private JPAContainer<Store> container;
        private Table table;

        public StoreView() {
                super("Administration - Stores");
        }

        @Override
        protected Component initContent() {
                container = new JPAContainer<>(Store.class);
                container.setEntityProvider(new
CachingMutableEntityProvider<>(Store.class));

                table = new Table("Stores", container);
                table.setImmediate(true);
                table.setSizeFull();
                table.setSelectable(true);
                table.setMultiSelect(false);
                table.setSortEnabled(true);
                table.setNullSelectionAllowed(true);
                table.setCellStyleGenerator(new PublishedCellStyleGenerator());
                table.setVisibleColumns("id", "name", "published");
                table.addGeneratedColumn("actions", new
ActionsColumnsGenerator(this));
                table.addGeneratedColumn("published", new BooleanColumnGenerator());
                table.setColumnHeaders("id", "Name", "Published", "Actions");
                table.setRowHeaderMode(Table.RowHeaderMode.INDEX);
                table.setColumnAlignments(Table.Align.CENTER, Table.Align.LEFT,
Table.Align.CENTER, Table.Align.CENTER);
                table.setColumnExpandRatio("name", 1.0f);

                Button addButton = new Button("New store", FontAwesome.PLUS);
                addButton.addStyleName(ValoTheme.BUTTON_PRIMARY);
                addButton.addClickListener(event -> {
                        StoreEditor editor = new
StoreEditor(container.createEntityItem(new Store()), this);
```

```
                        UI.getCurrent().addWindow(editor);
                });

                MVerticalLayout layout = new MVerticalLayout()
                                .add(new MHorizontalLayout(addButton)
                                                .withFullWidth())
                                .expand(table)
                                .withAlign(table, Alignment.MIDDLE_CENTER)
                                .withFullHeight()
                                .withSpacing(true);

                return layout;
        }

        @Override
        public void enter(ViewChangeListener.ViewChangeEvent event) {

        }

        @Override
        public void onSave(EntityItem item) {
                Object itemId = container.addEntity((Store)item.getEntity());
        }

        @Override
        public void edit(Object itemId) {
                StoreEditor editor = new StoreEditor(container.getItem(itemId),
this);
                UI.getCurrent().addWindow(editor);
        }

        @Override
        public void delete(Object itemId) {
                EntityItemProperty published =
container.getItem(itemId).getItemProperty("published");
                published.setValue(!(Boolean) published.getValue());
                table.refreshRowCache();
        }
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\util\ActionsColumnsGenerator.java

```
package gr.eshop7.eshop.admin.util;

import com.vaadin.server.FontAwesome;
import com.vaadin.ui.Table;
import com.vaadin.ui.themes.ValoTheme;
import gr.eshop7.eshop.admin.ItemEditorContainer;
import org.vaadin.viritin.button.MButton;
import org.vaadin.viritin.layouts.MHorizontalLayout;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
public class ActionsColumnsGenerator implements Table.ColumnGenerator {
        private ItemEditorContainer container;
        private boolean showEdit, showDelete;

        public ActionsColumnsGenerator(ItemEditorContainer container, boolean
showEdit, boolean showDelete) {
                this.container = container;
                this.showEdit = showEdit;
                this.showDelete = showDelete;
```

```
	}

	public ActionsColumnsGenerator(ItemEditorContainer container) {
		this(container, true, true);
	}

	@Override
	public Object generateCell(Table source, Object itemId, Object columnId) {
		MHorizontalLayout layout = new MHorizontalLayout();

		if(showEdit) {
			MButton editButton = new MButton(FontAwesome.EDIT)
					.withListener(e -> container.edit(itemId))
					.withStyleName(ValoTheme.BUTTON_SMALL)
					.withStyleName(ValoTheme.BUTTON_FRIENDLY);
			layout.add(editButton);
		}

		if(showDelete) {
			MButton deleteButton = new MButton(FontAwesome.TRASH_O)
					.withListener(e -> container.delete(itemId))
					.withStyleName(ValoTheme.BUTTON_SMALL)
					.withStyleName(ValoTheme.BUTTON_DANGER);
			layout.add(deleteButton);
		}

		return layout;
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\util\BooleanColumnGenerator.java

```
package gr.eshop7.eshop.admin.util;

import com.vaadin.data.Property;
import com.vaadin.server.FontAwesome;
import com.vaadin.shared.ui.label.ContentMode;
import com.vaadin.ui.Label;
import com.vaadin.ui.Table;

/**
 * Created on 9/12/2015.
 *
 * @author george
 */
public class BooleanColumnGenerator implements Table.ColumnGenerator {
	@Override
	public Object generateCell(Table source, Object itemId, Object columnId) {
		Property prop = source.getItem(itemId).getItemProperty(columnId);
		boolean published = (Boolean) prop.getValue();

		String html = published ? FontAwesome.CHECK.getHtml() :
FontAwesome.TIMES.getHtml();
		return new Label(html, ContentMode.HTML);
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\util\dto\StockBean.java

```
package gr.eshop7.eshop.admin.util.dto;

import org.eshop7.model.Product;
import org.eshop7.model.Store;
```

```
import java.io.Serializable;

/**
 * Created on 17/12/2015.
 *
 * @author george
 */
public class StockBean implements Serializable {
	private long productId;
	private String productName;

	private long storeId;
	private String storeName;

	private int stock;

	private String id;

	public StockBean(long productId, String productName, long storeId, String
storeName, int stock) {
		this.productId = productId;
		this.productName = productName;
		this.storeId = storeId;
		this.storeName = storeName;
		this.stock = stock;

		this.id = productId + "_" + storeId;
	}

	public StockBean(Product product, Store store, int stock) {
		this(product.getId(), product.getName(), store.getId(),
store.getName(), stock);
	}

	public long getProductId() {
		return productId;
	}

	public void setProductId(long productId) {
		this.productId = productId;
	}

	public String getProductName() {
		return productName;
	}

	public void setProductName(String productName) {
		this.productName = productName;
	}

	public long getStoreId() {
		return storeId;
	}

	public void setStoreId(long storeId) {
		this.storeId = storeId;
	}

	public String getStoreName() {
		return storeName;
	}

	public void setStoreName(String storeName) {
		this.storeName = storeName;
	}
```

```
	public int getStock() {
		return stock;
	}

	public void setStock(int stock) {
		this.stock = stock;
	}

	public String getId() {
		return id;
	}

	public void setId(String id) {
		this.id = id;
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\util\MenuItem.java

```
package gr.eshop7.eshop.admin.util;

import com.vaadin.server.Resource;
import com.vaadin.ui.UI;
import gr.eshop7.eshop.AdminUI;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
public class MenuItem {
	private String name;
	private Resource icon;
	private String targetView;

	public MenuItem(String name, Resource icon) {
		this.name = name;
		this.icon = icon;
	}

	public MenuItem(String name, Resource icon, String targetView) {
		this.name = name;
		this.icon = icon;
		this.targetView = targetView;
	}

	public String getName() {
		return name;
	}

	public void action() {
		((AdminUI)UI.getCurrent()).navigateTo(targetView);
	}

	public boolean hasAction() {
		return targetView != null;
	}

	public void setName(String name) {
		this.name = name;
	}

	public Resource getIcon() {
		return icon;
	}
```

```
	public void setIcon(Resource icon) {
		this.icon = icon;
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\util\MetroTheme.java

```
package gr.eshop7.eshop.admin.util;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
public interface MetroTheme {
	String LOGO = "img/logo.png";
	String LOGO_SMALL = "img/logo_small.png";


	String LABEL_INLINE = "v-label-inline";
	String MARGIN = "me-margin";
	String PANEL_CAPTION_TEXT_LARGE = "v-panel-caption";
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\util\OrderActionsColumnsGenerator.java

```
package gr.eshop7.eshop.admin.util;

import com.vaadin.addon.jpacontainer.JPAContainer;
import com.vaadin.server.FontAwesome;
import com.vaadin.ui.Button;
import com.vaadin.ui.HorizontalLayout;
import com.vaadin.ui.Table;
import com.vaadin.ui.themes.ValoTheme;
import gr.eshop7.eshop.admin.OrderContainer;
import org.eshop7.model.CustomerOrder;
import org.eshop7.model.OrderStatus;
import org.vaadin.viritin.button.MButton;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
public class OrderActionsColumnsGenerator implements Table.ColumnGenerator {
	private OrderContainer orderContainer;

	public OrderActionsColumnsGenerator(OrderContainer orderContainer) {
		this.orderContainer = orderContainer;
	}

	@Override
	public Object generateCell(Table source, Object itemId, Object columnId) {
		JPAContainer<CustomerOrder> container = (JPAContainer<CustomerOrder>)
source.getContainerDataSource();
		CustomerOrder order = container.getItem(itemId).getEntity();

		HorizontalLayout layout = new HorizontalLayout();
		layout.setSpacing(true);

		if (order != null) {
			if(OrderStatus.SUBMITTED.equals(order.getOrderStatus())) {
```

```
                            layout.addComponent(createInprogressButton(itemId));
                            layout.addComponent(createCancelButton(itemId));
                    } else
if(OrderStatus.IN_PROGRESS.equals(order.getOrderStatus())) {
                            layout.addComponent(createSentButton(itemId));
                            layout.addComponent(createCancelButton(itemId));
                    }
            }

            return layout;
    }

    private Button createInprogressButton(Object itemId) {
            return new MButton(FontAwesome.TRUCK)
                            .withListener(e ->
orderContainer.setStatusInProgress(itemId))
                            .withStyleName(ValoTheme.BUTTON_SMALL)
                            .withStyleName(ValoTheme.BUTTON_FRIENDLY);
    }

    private Button createSentButton(Object itemId) {
            return new MButton(FontAwesome.CHECK)
                            .withListener(e ->
orderContainer.setStatusSent(itemId))
                            .withStyleName(ValoTheme.BUTTON_SMALL)
                            .withStyleName(ValoTheme.BUTTON_FRIENDLY);
    }

    private Button createCancelButton(Object itemId) {
            return new MButton(FontAwesome.TRASH_O)
                            .withListener(e ->
orderContainer.setStatusCancelled(itemId))
                            .withStyleName(ValoTheme.BUTTON_SMALL)
                            .withStyleName(ValoTheme.BUTTON_DANGER);
    }
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\util\PublishedCellStyleGenerator.java

```
package gr.eshop7.eshop.admin.util;

import com.vaadin.data.Property;
import com.vaadin.ui.Table;

/**
 * Created by george on 12/12/2015.
 *
 */
public class PublishedCellStyleGenerator implements Table.CellStyleGenerator {
    @Override
    public String getStyle(Table source, Object itemId, Object propertyId) {
            Property prop = source.getItem(itemId).getItemProperty("published");
            boolean published = (Boolean)prop.getValue();

            return published ? "published" : "unpublished";
    }
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\util\TableHelpers.java

```
package gr.eshop7.eshop.admin.util;

import com.vaadin.server.Sizeable;
```

```
import com.vaadin.ui.Layout;
import com.vaadin.ui.Table;
import gr.eshop7.eshop.services.StatisticsService;
import gr.eshop7.eshop.util.DoubleValue;
import gr.eshop7.eshop.util.TripleValue;
import org.vaadin.viritin.ListContainer;

import java.util.Date;
import java.util.List;

/**
 *
 * Created by george on 27/03/2016.
 */
public class TableHelpers {

	public static Table createDoubleTupleTable(List<DoubleValue> list, String
col1, String col2) {
		Table table = new Table();
		table.setContainerDataSource(new ListContainer<>(DoubleValue.class,
list));
		table.setWidth(100, Sizeable.Unit.PERCENTAGE);
		table.setVisibleColumns("label", "value");
		table.setColumnExpandRatio("label", 1.0f);
		table.setColumnHeaders(col1, col2);

		return table;
	}

	public static Table createTripleTupleTable(List<TripleValue> list, String
col1, String col2, String col3) {
		Table table = new Table();
		table.setContainerDataSource(new ListContainer<>(TripleValue.class,
list));
		table.setWidth(100, Sizeable.Unit.PERCENTAGE);
		table.setVisibleColumns("label", "value1", "value2");
		table.setColumnExpandRatio("label", 1.0f);
		table.setColumnHeaders(col1, col2, col3);

		return table;
	}

	public static Table addIncomePerStore(Date startDate, Date endDate,
StatisticsService statisticsService) {
		List<DoubleValue> data =
statisticsService.getIncomePerStore(startDate, endDate);
		Table table = createDoubleTupleTable(data, "Store", "Total income");
		table.setCaption("Income per store");
		table.setPageLength(data.size());

		return table;
	}

	public static Table addStatsPerManufacturer(Date startDate, Date endDate,
StatisticsService statisticsService) {
		List<TripleValue> data =
statisticsService.getOrderStatsPerManufacturer(startDate, endDate);

		Table table = createTripleTupleTable(data, "Manufacturer", "Order
count", "Total income");
		table.setCaption("Orders per manufacturer");
		table.setPageLength(data.size());

		return table;
	}
```

```
	public static Table addStatsPerProduct(Date startDate, Date endDate,
StatisticsService statisticsService) {
		List<TripleValue> data =
statisticsService.getOrderStatsPerProduct(startDate, endDate);
		Table table = createTripleTupleTable(data, "Product", "Order count",
"Total income");
		table.setCaption("Orders per product");
		table.setPageLength(data.size());

		return table;
	}

	public static Table addStatsPerCategory(Date startDate, Date endDate,
StatisticsService statisticsService) {
		List<TripleValue> data =
statisticsService.getOrderStatsPerCategory(startDate, endDate);
		Table table = createTripleTupleTable(data, "Category", "Order count",
"Total income");
		table.setCaption("Orders per category");
		table.setPageLength(data.size());

		return table;
	}

	public static void addLatestOrders(Layout layout, StatisticsService
statisticsService) {
		List<TripleValue> data = statisticsService.getLatestOrders();
		Table table = createTripleTupleTable(data, "Product", "Shipping
address", "Submission Date");
		table.setCaption("Latest orders");
		table.setPageLength(data.isEmpty() ? 4 : data.size());

		layout.addComponent(table);
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\util\UIUtils.java

```
package gr.eshop7.eshop.admin.util;

import com.vaadin.server.FileResource;
import com.vaadin.server.Resource;
import com.vaadin.server.ThemeResource;
import gr.eshop7.eshop.Config;
import org.eshop7.model.Product;

import java.io.File;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
public class UIUtils {
	public static Resource themeImg(String path) {
		return new ThemeResource(path);
	}

	public static String generateImageName(long itemId, String mimetype) {
		String suffix = "";
		if (mimetype.equals("image/jpeg")) {
			suffix = ".jpeg";

		} else if (mimetype.equals("image/png")) {
			suffix = ".png";
```

```
		}

		return itemId + suffix;
	}

	public static String generateImagePath(long itemId, String mimetype) {
		return Config.PRODUCT_IMAGES_BASE + "/" + generateImageName(itemId,
mimetype);
	}

	public static Resource getImg(Product product) {
		return new FileResource(new File(Config.PRODUCT_IMAGES_BASE + "/" +
product.getImage()));
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\admin\util\Views.java

```
package gr.eshop7.eshop.admin.util;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
public class Views {
	public final static String MAIN = "main";
	public final static String CATEGORY = "category";
	public final static String MANUFACTURER = "manufacturer";
	public final static String PRODUCT = "product";
	public final static String STORE = "store";
	public final static String CUSTOMER = "customer";
	public final static String ORDER = "order";
	public final static String STOCK = "stock";
	public final static String STATISTICS = "statistics";

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\AdminUI.java

```
package gr.eshop7.eshop;

import com.vaadin.annotations.Theme;
import com.vaadin.annotations.Widgetset;
import com.vaadin.cdi.CDIUI;
import com.vaadin.cdi.CDIViewProvider;
import com.vaadin.navigator.Navigator;
import com.vaadin.server.VaadinRequest;
import com.vaadin.ui.UI;
import gr.eshop7.eshop.admin.util.Views;
import gr.eshop7.eshop.services.StockService;

import javax.inject.Inject;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
@CDIUI("")
@Theme("metro")
@Widgetset("gr.eshop7.MyAppWidgetset")
public class AdminUI extends UI {

	@Inject
```

```
	private CDIViewProvider viewProvider;

	@Inject
	private StockService stockService;

	private Navigator navigator;

	@Override
	protected void init(VaadinRequest vaadinRequest) {
		navigator = new Navigator(this, this);
		navigator.addProvider(viewProvider);

		navigateTo(Views.MAIN);
	}

	public void navigateTo(String viewName) {
		navigator.navigateTo(viewName);
	}

	public static StockService getStockService() {
		return ((AdminUI)getCurrent()).stockService;
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\Config.java

```
package gr.eshop7.eshop;


public class Config {
//	public static final String PRODUCT_IMAGES_BASE =
"C:\\development\\_server\\wamp\\www\\eshop7\\products";
	public static final String PRODUCT_IMAGES_BASE =
"C:\\development\\wamp\\www\\eshop7\\products";
	public static final String PRODUCT_IMAGES_URL_BASE =
"http://localhost/eshop7/products";
	public static final String NO_IMAGES = "no-image.png";


}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\services\CategoryService.java

```
package gr.eshop7.eshop.services;

import gr.eshop7.eshop.site.util.CategoryPart;
import org.eshop7.model.Category;

import javax.enterprise.context.RequestScoped;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.servlet.ServletContext;
import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;

/**
 *
 * @author george
 */
@RequestScoped
public class CategoryService {
	@PersistenceContext
	private EntityManager entityManager;
```

```
	public List<Category> getAllList() {
		ArrayList<Category> res = new ArrayList<>();
		List resultList = entityManager.createQuery("SELECT c FROM Category c
ORDER BY c.name DESC").getResultList();

		for (Object obj : resultList) {
			res.add((Category) obj);
		}
		return res;
	}

	public List<Category> getList() {
		ArrayList<Category> res = new ArrayList<>();
		List resultList = entityManager.createQuery("SELECT c FROM Category c
WHERE c.published = true ORDER BY c.name DESC").getResultList();

		for(Object obj: resultList) {
			res.add((Category) obj);
		}
		return res;
	}

	public List<CategoryPart> getPartsList(ServletContext context) {

		return getList().stream()
				.map(c -> new CategoryPart((Category) c, context))
				.collect(Collectors.toList());
	}

	public Category getCategory(long id) {
		Category category = entityManager.find(Category.class, id);

		return category;
	}

	public CategoryPart getCategoryPart(long id, ServletContext context) {
		Category category = entityManager.find(Category.class, id);

		return new CategoryPart(category, context);
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\services\CustomerService.java

```
package gr.eshop7.eshop.services;

import org.eshop7.model.Customer;

import javax.enterprise.context.RequestScoped;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import java.util.List;

/**
 * @author george
 */
@RequestScoped
public class CustomerService {
	@PersistenceContext
	private EntityManager entityManager;

	public Customer authenticate(String email, String password) {
```

```
		List resultList = entityManager.createQuery("SELECT c FROM Customer c
WHERE c.email = :email AND c.password = :password AND c.published = true")
				.setParameter("email", email)
				.setParameter("password", password)
				.getResultList();

		if(resultList == null || resultList.isEmpty()) {
			return null;
		}

		return (Customer) resultList.get(0);
	}

	public Customer findByEmail(String email) {
		List resultList = entityManager.createQuery("SELECT c FROM Customer c
WHERE c.email = :email")
				.setParameter("email", email)
				.getResultList();

		if (resultList == null || resultList.isEmpty()) {
			return null;
		}

		return (Customer) resultList.get(0);
	}

	public Customer createCustomer(String email, String password, String name,
String address, String phone) {
		Customer customer = new Customer(email, password, name, address,
phone);

		entityManager.persist(customer);

		return customer;
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\services\ManufacturerService.java

```
package gr.eshop7.eshop.services;

import gr.eshop7.eshop.site.util.ManufacturerPart;
import org.eshop7.model.Manufacturer;

import javax.enterprise.context.RequestScoped;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.servlet.ServletContext;
import java.util.ArrayList;
import java.util.List;
import java.util.stream.Collectors;

/**
 *
 * @author george
 */
@RequestScoped
public class ManufacturerService {
	@PersistenceContext
	private EntityManager entityManager;

	public List<Manufacturer> getAllList() {
		ArrayList<Manufacturer> res = new ArrayList<>();
```

```
		List resultList = entityManager.createQuery("SELECT m FROM
Manufacturer m ORDER BY m.name DESC").getResultList();

		for (Object obj : resultList) {
			res.add((Manufacturer) obj);
		}
		return res;
	}

	public List<Manufacturer> getList() {
		ArrayList<Manufacturer> res = new ArrayList<>();
		List resultList = entityManager.createQuery("SELECT m FROM
Manufacturer m WHERE m.published = true ORDER BY m.name DESC").getResultList();

		for(Object obj: resultList) {
			res.add((Manufacturer) obj);
		}
		return res;
	}

	public List<ManufacturerPart> getPartsList(ServletContext context) {

		return getList().stream()
				.map(c -> new ManufacturerPart((Manufacturer) c,
context))
				.collect(Collectors.toList());
	}

	public Manufacturer getManufacturer(long id) {
		Manufacturer manufacturer = entityManager.find(Manufacturer.class,
id);

		return manufacturer;
	}

	public ManufacturerPart getManufacturerPart(long id, ServletContext context)
{
		Manufacturer manufacturer = entityManager.find(Manufacturer.class,
id);

		return new ManufacturerPart(manufacturer, context);
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\services\OrderService.java

```
package gr.eshop7.eshop.services;

import gr.eshop7.eshop.util.Utils;
import org.eshop7.model.Customer;
import org.eshop7.model.CustomerOrder;
import org.eshop7.model.OrderPayment;
import org.eshop7.model.OrderShipping;
import org.eshop7.model.OrderStatus;
import org.eshop7.model.Product;
import org.eshop7.model.Store;

import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.transaction.Transactional;
import java.util.ArrayList;
import java.util.Date;
```

```
import java.util.List;

/**
 * @author george
 */
@RequestScoped
@Transactional
public class OrderService {
	@PersistenceContext
	private EntityManager entityManager;

	@Inject
	private StockService stockService;

	public CustomerOrder placeOrder(Product product, Customer customer, String
name, String email, String address, String phone,
								OrderShipping
orderShipping, Store store, OrderPayment orderPayment,
								String ccType,
String ccNumber, String ccName, String ccCcv) {
		float price = product.getPrice();
		String ref = Utils.generateOrderCode();

		String paymentInfo = "Cash";
		if(OrderPayment.CREDIT_CARD.equals(orderPayment)) {
			paymentInfo = Utils.generatePaymentInfo(ccType, ccNumber,
ccName, ccCcv);
		}

		CustomerOrder order = new CustomerOrder(
				OrderStatus.SUBMITTED,
				orderShipping,
				orderPayment,
				customer,
				product,
				store,
				price,
				ref,
				name,
				address,
				phone,
				new Date(),
				null,
				paymentInfo

		);

		entityManager.persist(order);

		updateStock(order);

		return order;
	}

	public void updateStock(CustomerOrder order) {
		if(order.getOrderShipping() == OrderShipping.PICKUP) {
			stockService.decreaseStock(order.getStore(),
order.getProduct());

		} else {
			stockService.decreaseStock(order.getProduct());
		}
	}

	public List<CustomerOrder> getList(Customer customer) {
		ArrayList<CustomerOrder> res = new ArrayList<>();
```

```
		List resultList = entityManager.createQuery("SELECT o FROM
CustomerOrder o WHERE o.customer=:customer")
				.setParameter("customer", customer)
				.getResultList();

		for(Object obj: resultList) {
			res.add((CustomerOrder) obj);
		}

		return res;
	}

	@Transactional
	public void cancel(long orderId) {
		CustomerOrder order = entityManager.find(CustomerOrder.class,
orderId);
		order.setOrderStatus(OrderStatus.CANCELLED);
		stockService.increaseStock(order.getProduct());
		entityManager.merge(order);
	}

	@Transactional
	public void complete(long orderId) {
		CustomerOrder order = entityManager.find(CustomerOrder.class,
orderId);
		order.setOrderStatus(OrderStatus.DELIVERED);
		order.setDeliveryDate(new Date());
		entityManager.merge(order);
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\services\ProductService.java

```
package gr.eshop7.eshop.services;

import gr.eshop7.eshop.site.util.ProductPart;
import org.eshop7.model.Product;

import javax.enterprise.context.RequestScoped;
import javax.inject.Inject;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.servlet.ServletContext;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.stream.Collectors;

/**
 *
 * @author george
 */
@RequestScoped
public class ProductService {
	@PersistenceContext
	private EntityManager entityManager;

	@Inject
	private StockService stockService;

	public List<Product> getAllList() {
		ArrayList<Product> res = new ArrayList<>();
		List resultList = entityManager.createQuery("SELECT p FROM Product p
ORDER BY p.name").getResultList();
```

```
		for(Object obj: resultList) {
			res.add((Product) obj);
		}

		return res;
	}


	public List<Product> getList() {
		ArrayList<Product> res = new ArrayList<>();
		List resultList = entityManager.createQuery("SELECT p FROM Product p
WHERE p.published = true ORDER BY p.name").getResultList();

		for(Object obj: resultList) {
			res.add((Product) obj);
		}

		return res;
	}

	public List<ProductPart> getPartsListForCategory(ServletContext context) {
		return getList().stream()
				.map(c -> new ProductPart((Product) c,
stockService.isInStock((Product) c), context))
				.collect(Collectors.toList());
	}

	public List<Product> getListForCategory(long categoryId) {
		ArrayList<Product> res = new ArrayList<>();
		List resultList = entityManager.createQuery("SELECT p FROM Product p
WHERE p.published = true AND p.category.id = :cid ORDER BY p.name")
				.setParameter("cid", categoryId)
				.getResultList();

		for (Object obj : resultList) {
			res.add((Product) obj);
		}

		return res;
	}

	public List<Product> getListForManufacturer(long manufacturerId) {
		ArrayList<Product> res = new ArrayList<>();
		List resultList = entityManager.createQuery("SELECT p FROM Product p
WHERE p.published = true AND p.manufacturer.id = :cid ORDER BY p.name")
				.setParameter("cid", manufacturerId)
				.getResultList();

		for (Object obj : resultList) {
			res.add((Product) obj);
		}

		return res;
	}

	public List<ProductPart> getPartsListForCategory(long categoryId,
ServletContext context) {
		return getListForCategory(categoryId).stream()
				.map(c -> new ProductPart((Product) c,
stockService.isInStock((Product) c), context))
				.collect(Collectors.toList());
	}

	public List<ProductPart> getPartsListForManufacturer(long manufacturerId,
ServletContext context) {
```

```
		return getListForManufacturer(manufacturerId).stream()
				.map(c -> new ProductPart((Product) c,
stockService.isInStock((Product) c), context))
				.collect(Collectors.toList());
	}

	public List<Product> getRandomList() {
		List<Product> list = getList();
		Collections.shuffle(list);

		return list;
	}

	public Product getProduct(long id) {
		return entityManager.find(Product.class, id);
	}

	public ProductPart getProductPart(long id, ServletContext context) {
		Product product = getProduct(id);

		return new ProductPart(product, stockService.isInStock(product),
context);
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\services\StatisticsService.java

```
package gr.eshop7.eshop.services;

import gr.eshop7.eshop.util.DoubleValue;
import gr.eshop7.eshop.util.TripleValue;
import org.eshop7.model.*;

import javax.ejb.Stateless;
import javax.inject.Inject;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;

/**
 *
 * Created by george
 */
@Stateless
public class StatisticsService {
	private DecimalFormat decimalFormat = new DecimalFormat("#.00");

	@PersistenceContext
	private EntityManager entityManager;

	@Inject
	private StoreService storeService;

	@Inject
	private ManufacturerService manufacturerService;

	@Inject
	private CategoryService categoryService;

	@Inject
	private ProductService productService;
```

```
	public List<DoubleValue> getIncomePerStore(Date startDate, Date endDate) {
		ArrayList<DoubleValue> res = new ArrayList<>();
		List<Store> stores = storeService.getAllList();

		for (Store store : stores) {
			List resultList = entityManager.createQuery("SELECT
sum(o.price) FROM CustomerOrder o WHERE o.store=:store AND o.submissionDate BETWEEN
:startDate AND :endDate GROUP BY o.store")
					.setParameter("store", store)
					.setParameter("startDate", startDate)
					.setParameter("endDate", endDate)
					.getResultList();

			String sum = "0.00";
			if(!resultList.isEmpty()) {
				sum = decimalFormat.format(resultList.get(0));
			}

			res.add(new DoubleValue(store.getName(), sum));
		}

		return res;
	}

	public List<TripleValue> getOrderStatsPerManufacturer(Date startDate, Date
endDate) {
		ArrayList<TripleValue> res = new ArrayList<>();

		List<Manufacturer> manufacturers = manufacturerService.getAllList();

		for (Manufacturer manufacturer : manufacturers) {
			List<Object[]> resultList = entityManager.createQuery("SELECT
count(o), sum(o.price) FROM CustomerOrder o WHERE
o.product.manufacturer=:manufacturer AND o.submissionDate BETWEEN :startDate AND
:endDate GROUP BY o.product.manufacturer")
					.setParameter("manufacturer", manufacturer)
					.setParameter("startDate", startDate)
					.setParameter("endDate", endDate)
					.getResultList();

			String count = "0";
			String sum = "0.00";
			if (!resultList.isEmpty()) {
				count = String.valueOf(resultList.get(0)[0]);
				sum = decimalFormat.format(resultList.get(0)[1]);
			}
			res.add(new TripleValue(manufacturer.getName(), count, sum));
		}

		return res;
	}

	public List<TripleValue> getOrderStatsPerCategory(Date startDate, Date
endDate) {
		ArrayList<TripleValue> res = new ArrayList<>();

		List<Category> categories = categoryService.getAllList();

		for (Category category : categories) {
			List<Object[]> resultList = entityManager.createQuery("SELECT
count(o), sum(o.price) FROM CustomerOrder o WHERE o.product.category=:category AND
o.submissionDate BETWEEN :startDate AND :endDate GROUP BY o.product.category")
					.setParameter("category", category)
					.setParameter("startDate", startDate)
					.setParameter("endDate", endDate)
```

```
                                .getResultList();

                        String count = "0";
                        String sum = "0.00";
                        if (!resultList.isEmpty()) {
                                count = String.valueOf(resultList.get(0)[0]);
                                sum = decimalFormat.format(resultList.get(0)[1]);
                        }
                        res.add(new TripleValue(category.getName(), count, sum));
                }

                return res;
        }

        public List<TripleValue> getOrderStatsPerProduct(Date startDate, Date
endDate) {
                ArrayList<TripleValue> res = new ArrayList<>();

                List<Product> products = productService.getAllList();

                for (Product product : products) {
                        List<Object[]> resultList = entityManager.createQuery("SELECT
count(o), sum(o.price) FROM CustomerOrder o WHERE o.product=:product AND
o.submissionDate BETWEEN :startDate AND :endDate GROUP BY o.product")
                                .setParameter("product", product)
                                .setParameter("startDate", startDate)
                                .setParameter("endDate", endDate)
                                .getResultList();

                        String count = "0";
                        String sum = "0.00";
                        if (!resultList.isEmpty()) {
                                count = String.valueOf(resultList.get(0)[0]);
                                sum = decimalFormat.format(resultList.get(0)[1]);
                        }
                        res.add(new TripleValue(product.getName(), count, sum));
                }

                return res;
        }


        public List<DoubleValue> getMeanTimePerStore(Date startDate, Date endDate) {
                ArrayList<DoubleValue> res = new ArrayList<>();
                List<Store> stores = storeService.getAllList();

                for (Store store : stores) {
                        List<Object[]> resultList = entityManager.createQuery("SELECT
o.submissionDate, o.deliveryDate FROM CustomerOrder o WHERE o.store=:store AND
o.orderStatus=:orderStatus AND o.submissionDate BETWEEN :startDate AND :endDate")
                                .setParameter("store", store)
                                .setParameter("startDate", startDate)
                                .setParameter("endDate", endDate)
                                .setParameter("orderStatus",
OrderStatus.DELIVERED)
                                .getResultList();

                        String meanTime = "-";
                        if (!resultList.isEmpty()) {
                                long sum = 0;
                                int count = resultList.size();

                                for (Object[] dates : resultList) {
                                        long startTime = ((Date) dates[0]).getTime();
                                        long endTime = ((Date) dates[1]).getTime();
                                        sum += endTime/1000 - startTime/1000;
```

```
                            }

                            int avg = (int)((double)sum / count);

                            int days = avg / (24*60*60);
                            avg %= (24*60*60);

                            int hours = avg / (60*60);
                            avg %= (60*60);

                            int minutes = avg / (60);
                            avg %= (60);

                            meanTime = String.format("%d days, %d hours, %d
minutes, %d seconds", days, hours, minutes, avg);
                        }

                        res.add(new DoubleValue(store.getName(), meanTime));
                }

                return res;
        }

        public List<TripleValue> getLatestOrders() {
                ArrayList<TripleValue> res = new ArrayList<>();

                List<CustomerOrder> orders = entityManager.createQuery("SELECT o FROM
CustomerOrder o WHERE o.orderStatus = :orderStatus ORDER BY o.submissionDate DESC")
                            .setParameter("orderStatus", OrderStatus.SUBMITTED)
                            .getResultList();

                orders.forEach(o -> {
                        String shippingAddress = o.getOrderShipping() ==
OrderShipping.COURIER
                                        ? o.getAddress()
                                        : o.getStore().getName();

                        res.add(new TripleValue(o.getProduct().getName(),
shippingAddress, o.getSubmissionDate()));
                });

                return res;
        }
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\services\StockService.java

```
package gr.eshop7.eshop.services;

import gr.eshop7.eshop.admin.util.dto.StockBean;
import org.eshop7.model.Product;
import org.eshop7.model.Stock;
import org.eshop7.model.Store;

import javax.ejb.Stateless;
import javax.inject.Inject;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.transaction.Transactional;
import java.util.ArrayList;
import java.util.List;

/**
 * Created on 17/12/2015.
 *
```

```
 * @author george
 */
@Stateless
@Transactional
public class StockService {
	@PersistenceContext
	private EntityManager entityManager;

	@Inject
	private ProductService productService;

	@Inject
	private StoreService storeService;

	public List<StockBean> getAllStockBeans() {
		ArrayList<StockBean> res = new ArrayList<>();

		List<Product> products = productService.getAllList();
		List<Store> stores = storeService.getAllList();

		for(Store store: stores) {
			for(Product product: products) {
				int stock = getStockValue(store, product);

				res.add(new StockBean(product, store, stock));
			}
		}

		return res;
	}

	public Stock getStock(Store store, Product product) {
		List res = entityManager.createQuery("SELECT s FROM Stock s WHERE
s.product = :product AND s.store = :store")
				.setParameter("product", product)
				.setParameter("store", store)
				.getResultList();

		if(res == null || res.isEmpty()) {
			return null;

		} else {
			return (Stock) res.get(0);
		}
	}

	private List<Stock> getStockForProduct(Product product) {
		return entityManager.createQuery("SELECT s FROM Stock s WHERE
s.product = :product ORDER BY s.stock DESC ")
				.setParameter("product", product)
				.getResultList();

	}

	public int getStockValue(Store store, Product product) {
		Stock stock = getStock(store, product);

		return stock == null ? 0 : stock.getStock();
	}

	public boolean isInStock(Product product) {
		List<Stock> stockForProduct = getStockForProduct(product);

		for (Stock stock : stockForProduct) {
			if(stock.getStock() > 0) {
				return true;
```

```
                }
        }

        return false;
    }

    @Transactional
    public void saveStock(long storeId, long productId, int stock) {
        Store store = entityManager.find(Store.class, storeId);
        Product product = entityManager.find(Product.class, productId);

        saveStock(store, product, stock);
    }

    @Transactional
    private void saveStock(Store store, Product product, int stock) {
        Stock entity = getStock(store, product);

        if(entity == null) {
            entity = new Stock(product, store, stock);
            entityManager.persist(entity);
        } else {
            entity.setStock(stock);
            entityManager.merge(entity);
        }
    }

    @Transactional
    public void decreaseStock(Store store, Product product) {
        Stock stock = getStock(store, product);

        stock.setStock(stock.getStock()-1);
        entityManager.merge(stock);
    }

    @Transactional
    public void increaseStock(Store store, Product product) {
        Stock stock = getStock(store, product);

        stock.setStock(stock.getStock() + 1);
        entityManager.merge(stock);
    }

    public void decreaseStock(Product product) {
        Stock stock = getStockForProduct(product).get(0);

        decreaseStock(stock.getStore(), product);
    }

    public void increaseStock(Product product) {
        Stock stock = getStockForProduct(product).get(0);

        increaseStock(stock.getStore(), product);
    }
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\services\StoreService.java

```
package gr.eshop7.eshop.services;

import org.eshop7.model.Product;
import org.eshop7.model.Stock;
import org.eshop7.model.Store;

import javax.enterprise.context.RequestScoped;
```

```
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import java.util.ArrayList;
import java.util.List;

/**
 *
 * Created by george
 */
@RequestScoped
public class StoreService {

	@PersistenceContext
	private EntityManager entityManager;

	public List<Store> getAllList() {
		ArrayList<Store> res = new ArrayList<>();
		List resultList = entityManager.createQuery("SELECT s FROM Store s
ORDER BY s.name").getResultList();

		for(Object obj: resultList) {
			res.add((Store) obj);
		}

		return res;
	}

	public List<Store> getList() {
		ArrayList<Store> res = new ArrayList<>();
		List resultList = entityManager.createQuery("SELECT s FROM Store s
WHERE s.published = true ORDER BY s.name").getResultList();

		for(Object obj: resultList) {
			res.add((Store) obj);
		}

		return res;
	}

	public List<Store> getList(Product product) {
		ArrayList<Store> res = new ArrayList<>();
		List resultList = entityManager.createQuery("SELECT s FROM Stock s
WHERE s.product=:product AND s.stock > 0")
				.setParameter("product", product)
				.getResultList();

		for (Object obj : resultList) {
			Stock stock = (Stock) obj;
			res.add(stock.getStore());
		}

		return res;
	}


	public Store getStore(long id) {
		return entityManager.find(Store.class, id);
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\actions\CheckoutAction.java

```
package gr.eshop7.eshop.site.actions;
```

```
import gr.eshop7.eshop.services.CustomerService;
import gr.eshop7.eshop.services.OrderService;
import gr.eshop7.eshop.services.ProductService;
import gr.eshop7.eshop.services.StoreService;
import gr.eshop7.eshop.site.views.AbstractEshopServlet;
import org.eshop7.model.*;

import javax.inject.Inject;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.transaction.Transactional;
import java.io.IOException;
import java.util.Map;

/**
 * @author george
 */
@WebServlet(name = "CheckoutActionServlet", urlPatterns = "/checkoutAction")
@Transactional
public class CheckoutAction extends HttpServlet {

	@Inject
	private CustomerService customerService;

	@Inject
	private ProductService productService;

	@Inject
	private OrderService orderService;

	@Inject
	private StoreService storeService;

	protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		processRequest(request, response);
	}

	protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		processRequest(request, response);
	}

	protected void processRequest(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {
		Map<String, String[]> parameterMap = request.getParameterMap();

		String email = request.getParameter("email");
		String name = request.getParameter("name");
		String address = request.getParameter("address");
		String phone = request.getParameter("phone");
		OrderShipping orderShipping =
OrderShipping.valueOf(request.getParameter("orderShipping"));
		Store store = null;

		if(OrderShipping.PICKUP.equals(orderShipping)) {
			store =
storeService.getStore(Long.parseLong(request.getParameter("storeId")));
		}

		String ccType = request.getParameter("ccType");
		String ccNumber = request.getParameter("ccNumber");
		String ccName = request.getParameter("ccName");
```

```
		String ccCcv = request.getParameter("ccCcv");
		OrderPayment orderPayment =
OrderPayment.valueOf(request.getParameter("orderPayment"));

		Customer customer = (Customer)
request.getSession().getAttribute(AbstractEshopServlet.SES_CUSTOMER);
		Product product =
productService.getProduct(Long.parseLong(request.getParameter("id")));

		String paymentInfo = null;

		CustomerOrder order = orderService.placeOrder(product, customer,
name, email, address, phone, orderShipping, store,
			orderPayment, ccType, ccNumber, ccName, ccCcv);


		request.getSession().setAttribute(AbstractEshopServlet.SES_MESSAGE,
"Your order has been placed with code: " + order.getRef() + ".");
		response.sendRedirect(request.getContextPath() + "/main");
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\actions\CompleteOrderAction.java

```
package gr.eshop7.eshop.site.actions;

import gr.eshop7.eshop.services.OrderService;
import gr.eshop7.eshop.site.views.AbstractEshopServlet;

import javax.inject.Inject;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

/**
 * @author george
 */
@WebServlet(name = "CompleteOrderActionServlet", urlPatterns =
"/completeOrderAction")
public class CompleteOrderAction extends HttpServlet {

	@Inject
	private OrderService orderService;

	protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		processRequest(request, response);
	}

	protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		processRequest(request, response);
	}

	protected void processRequest(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {
		String action = request.getParameter("action");
		long orderId = Long.parseLong(request.getParameter("orderId"));

		if("CANCEL".equals(action)) {
			orderService.cancel(orderId);
```

```
		} else { // if(action.equals("COMPLETE")) {
			orderService.complete(orderId);
		}

		request.getSession().setAttribute(AbstractEshopServlet.SES_MESSAGE,
"Order updated.");
		response.sendRedirect(request.getContextPath() + "/profile");
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\actions\LoginAction.java

```
package gr.eshop7.eshop.site.actions;

import gr.eshop7.eshop.services.CustomerService;
import gr.eshop7.eshop.site.views.AbstractEshopServlet;
import org.eshop7.model.Customer;

import javax.inject.Inject;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

/**
 * @author george
 */
@WebServlet(name = "LoginActionServlet", urlPatterns = "/loginAction")
public class LoginAction extends HttpServlet {

	@Inject
	private CustomerService customerService;

	protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		processRequest(request, response);
	}

	protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		processRequest(request, response);
	}

	protected void processRequest(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {
		String email = request.getParameter("email");
		String password = request.getParameter("password");

		Customer c = customerService.authenticate(email, password);

		if (c == null) {

	request.getSession().setAttribute(AbstractEshopServlet.SES_MESSAGE,
"Authentication failed. Please try again.");
			response.sendRedirect(request.getContextPath() + "/login");

		} else {

	request.getSession().setAttribute(AbstractEshopServlet.SES_CUSTOMER, c);

	request.getSession().setAttribute(AbstractEshopServlet.SES_MESSAGE, "Welcome
" + c.getName() + ".");
```

```
                        if(!c.isAdmin()) {
                                response.sendRedirect(request.getContextPath() +
"/profile");
                        } else {
                                response.sendRedirect(request.getContextPath() +
"/admin");
                        }
                }
        }

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\actions\LogoutAction.java

```
package gr.eshop7.eshop.site.actions;

import gr.eshop7.eshop.services.CustomerService;
import gr.eshop7.eshop.site.views.AbstractEshopServlet;

import javax.inject.Inject;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;

/**
 * @author george
 */
@WebServlet(name = "LogoutActionServlet", urlPatterns = "/logoutAction")
public class LogoutAction extends HttpServlet {

        @Inject
        private CustomerService customerService;

        protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
                processRequest(request, response);
        }

        protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
                processRequest(request, response);
        }

        protected void processRequest(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {

        request.getSession().removeAttribute(AbstractEshopServlet.SES_CUSTOMER);

                request.getSession().setAttribute(AbstractEshopServlet.SES_MESSAGE,
"Logout successful.");
                response.sendRedirect(request.getContextPath() + "/main");
        }
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\actions\RegisterAction.java

```
package gr.eshop7.eshop.site.actions;

import gr.eshop7.eshop.services.CustomerService;
import gr.eshop7.eshop.site.views.AbstractEshopServlet;
```

```
import org.eshop7.model.Customer;

import javax.inject.Inject;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.transaction.Transactional;
import java.io.IOException;

/**
 * @author george
 */
@WebServlet(name = "RegisterActionServlet", urlPatterns = "/registerAction")
@Transactional
public class RegisterAction extends HttpServlet {

	@Inject
	private CustomerService customerService;

	protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		processRequest(request, response);
	}

	protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		processRequest(request, response);
	}

	protected void processRequest(HttpServletRequest request,
HttpServletResponse response) throws ServletException, IOException {
		String email = request.getParameter("email");
		String password = request.getParameter("password");
		String name = request.getParameter("name");
		String address = request.getParameter("address");
		String phone = request.getParameter("phone");

		Customer c = customerService.findByEmail(email);

		if(c != null) {

	request.getSession().setAttribute(AbstractEshopServlet.SES_MESSAGE, "This
email address belongs to an existing account.");
			response.sendRedirect(request.getContextPath() + "/login");
		} else {
			c = customerService.createCustomer(email, password, name,
address, phone);


	request.getSession().setAttribute(AbstractEshopServlet.SES_CUSTOMER, c);

	request.getSession().setAttribute(AbstractEshopServlet.SES_MESSAGE, "Welcome
" + c.getName() + ".");
			response.sendRedirect(request.getContextPath() + "/main");
		}
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\util\CategoryPart.java

```
package gr.eshop7.eshop.site.util;
```

```
import org.eshop7.model.Category;

import javax.servlet.ServletContext;

/**
 * Created on 11/11/2015.
 *
 * @author george
 */
public class CategoryPart extends WebPart {

	public CategoryPart(Category category, ServletContext context) {
		super(category.getName(), "", url(category, context));
	}

	public static String url(Category category, ServletContext context) {
		return context.getContextPath() + "/category?id=" + category.getId();
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\util\ManufacturerPart.java

```
package gr.eshop7.eshop.site.util;

import org.eshop7.model.Manufacturer;

import javax.servlet.ServletContext;

/**
 * Created on 11/11/2015.
 *
 * @author george
 */
public class ManufacturerPart extends WebPart {

	public ManufacturerPart(Manufacturer manufacturer, ServletContext context) {
		super(manufacturer.getName(), "", url(manufacturer, context));
	}

	public static String url(Manufacturer category, ServletContext context) {
		return context.getContextPath() + "/category?id=" + category.getId()
+ "&type=manufacturer";
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\util\ProductPart.java

```
package gr.eshop7.eshop.site.util;

import gr.eshop7.eshop.Config;
import gr.eshop7.eshop.util.Utils;
import org.eshop7.model.Product;

import javax.servlet.ServletContext;
import java.text.DecimalFormat;
import java.text.NumberFormat;

/**
 * Created on 11/11/2015.
 *
 * @author george
 */
public class ProductPart extends WebPart {
	public static final NumberFormat FORMATTER = new DecimalFormat("#0.00");
```

```
	private float price;

	private String buyUrl, buyActionUrl;

	private boolean inStock;

	public ProductPart(Product product, boolean inStock, ServletContext context)
{
		super(product.getName(), product.getDescription(),
product.getImage(), url(product, context));
		this.price = product.getPrice();
		this.inStock = inStock;

		this.buyUrl = getBuyUrl(product, context);
		this.buyActionUrl = getBuyActionUrl(product, context);
	}

	public float getPrice() {
		return price;
	}

	@Override
	public String getImage() {
		if(Utils.isEmpty(super.getImage())) {
			return Config.PRODUCT_IMAGES_URL_BASE + "/" +
Config.NO_IMAGES;

		} else {
			return Config.PRODUCT_IMAGES_URL_BASE + "/" +
super.getImage();
		}
	}

	public String getFormattedPrice() {
		return FORMATTER.format(price);
	}

	public static String url(Product product, ServletContext context) {
		return context.getContextPath() + "/product?id=" + product.getId();
	}

	public String getBuyUrl(Product product, ServletContext context) {
		return context.getContextPath() + "/checkout?id=" + product.getId();
	}

	public String getBuyActionUrl(Product product, ServletContext context) {
		return context.getContextPath() + "/checkoutAction?id=" +
product.getId();
	}

	public String getBuyUrl() {
		return buyUrl;
	}

	public String getBuyActionUrl() {
		return buyActionUrl;
	}

	public boolean isInStock() {
		return inStock;
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\util\WebPart.java

```
package gr.eshop7.eshop.site.util;

import java.util.ArrayList;
import java.util.List;

/**
 * Created on 11/11/2015.
 *
 * @author george
 */
public class WebPart {
	private String name;
	private String description;
	private String image;
	private String url;

	private ArrayList<WebPart> contents = new ArrayList<>();
	private ArrayList<WebPart> children = new ArrayList<>();
	private ArrayList<WebPart> secondaryChildren = new ArrayList<>();

	public WebPart(String name, String description, String image, String url) {
		this.name = name;
		this.description = description;
		this.image = image;
		this.url = url;
	}

	public WebPart(String name, String image, String url) {
		this(name, "", image, url);
	}

	public String getName() {
		return name;
	}

	public String getDescription() {
		return description;
	}

	public String getImage() {
		return image;
	}

	public String getUrl() {
		return url;
	}

	public ArrayList<WebPart> getContents() {
		return contents;
	}

	public void addContents(List<? extends WebPart> contents) {
		this.contents.addAll(contents);
	}

	public ArrayList<WebPart> getChildren() {
		return children;
	}

	public void addChildren(List<? extends WebPart> children) {
		this.children.addAll(children);
	}

	public ArrayList<WebPart> getSecondaryChildren() {
		return secondaryChildren;
```

```
	}

	public void addSecondaryChildren(List<? extends WebPart> secondaryChild) {
		this.secondaryChildren.addAll(secondaryChild);
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\views\AbstractEshopServlet.java

```
package gr.eshop7.eshop.site.views;

import org.apache.velocity.Template;
import org.apache.velocity.context.Context;
import org.apache.velocity.tools.view.VelocityViewServlet;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.text.DecimalFormat;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.HashMap;

/**
 *
 * Created by george
 */
public abstract class AbstractEshopServlet extends VelocityViewServlet {
	public final static String SES_MESSAGE = "ses_message";
	public final static String SES_CUSTOMER = "ses_customer";

	private HttpServletRequest request;

	private SimpleDateFormat dateFormat = new SimpleDateFormat("dd/MM/YYYY,
HH:mm");
	private DecimalFormat priceFormat = new DecimalFormat("#.00");

	@Override
	protected Template handleRequest(HttpServletRequest request,
HttpServletResponse response, Context context) {
		HashMap<String, Object> site = new HashMap<>();

		this.request = request;

		site.put("name", "Electronics Shop");
		site.put("homeUrl", getHomeUrl());
		site.put("base", getServletContext().getContextPath());

		boolean loggedIn = request.getSession().getAttribute(SES_CUSTOMER) !=
null;
		site.put("loggedIn", loggedIn);
		if(loggedIn) {
			site.put("customer",
request.getSession().getAttribute(SES_CUSTOMER));
		}

		boolean hasMessage = request.getSession().getAttribute(SES_MESSAGE)
!= null;
		site.put("hasMessage", hasMessage);
		if(hasMessage) {
			site.put("message",
request.getSession().getAttribute(SES_MESSAGE));
		}

		context.put("SITE", site);
		context.put("ACTION", this);
```

```
		process(request, response, context);

		return getTemplate(getTemplateName());
	}

	protected abstract String getTemplateName();

	protected abstract void process(HttpServletRequest request,
HttpServletResponse response, Context context);

	public String getHomeUrl() {
		return getServletContext().getContextPath() + "/main";
	}

	public void cleanup() {
		request.getSession().removeAttribute(SES_MESSAGE);
	}

	public String formatDate(Date date) {
		return dateFormat.format(date);

	}

	public String formatPrice(float price) {
		return priceFormat.format(price);

	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\views\CategoryView.java

```
package gr.eshop7.eshop.site.views;

import gr.eshop7.eshop.services.CategoryService;
import gr.eshop7.eshop.services.ManufacturerService;
import gr.eshop7.eshop.services.ProductService;
import gr.eshop7.eshop.site.util.WebPart;
import org.apache.velocity.context.Context;

import javax.inject.Inject;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Created on 11/11/2015.
 *
 * @author george
 */
@WebServlet(name = "CategoryServlet", urlPatterns = "/category")
public class CategoryView extends AbstractEshopServlet {

	@Inject
	CategoryService categoryService;

	@Inject
	ManufacturerService manufacturerService;

	@Inject
	ProductService productService;

	@Override
	protected String getTemplateName() {
		return "category.vm";
```

```
	}

	@Override
	protected void process(HttpServletRequest request, HttpServletResponse
response, Context context) {
		long id = Long.parseLong(request.getParameter("id"));
		String type = request.getParameter("type");

		WebPart page;

		if(type == null) {
			page = categoryService.getCategoryPart(id,
getServletContext());
			page.addContents(productService.getPartsListForCategory(id,
getServletContext()));
		} else {
			page = manufacturerService.getManufacturerPart(id,
getServletContext());

	page.addContents(productService.getPartsListForManufacturer(id,
getServletContext()));
		}

		page.addChildren(categoryService.getPartsList(getServletContext()));

	page.addSecondaryChildren(manufacturerService.getPartsList(getServletContext
()));

		context.put("PAGE", page);
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\views\CheckoutView.java

```
package gr.eshop7.eshop.site.views;

import gr.eshop7.eshop.services.*;
import gr.eshop7.eshop.site.util.ProductPart;
import org.apache.velocity.context.Context;
import org.eshop7.model.Product;

import javax.inject.Inject;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Created on 11/11/2015.
 *
 * @author george
 */
@WebServlet(name = "CheckoutServlet", urlPatterns = "/checkout")
public class CheckoutView extends AbstractEshopServlet {
	@Inject
	CategoryService categoryService;

	@Inject
	ManufacturerService manufacturerService;

	@Inject
	ProductService productService;

	@Inject
	CustomerService customerService;
```

```
	@Inject
	private StoreService storeService;

	@Override
	protected String getTemplateName() {
		return "checkout.vm";
	}

	@Override
	protected void process(HttpServletRequest request, HttpServletResponse
response, Context context) {
		long id = Long.parseLong(request.getParameter("id"));

		Product product = productService.getProduct(id);
		ProductPart page = productService.getProductPart(id,
getServletContext());
		page.addChildren(categoryService.getPartsList(getServletContext()));

	page.addSecondaryChildren(manufacturerService.getPartsList(getServletContext
()));

		context.put("PAGE", page);
		context.put("STORES", storeService.getList(product));
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\views\ImageServlet.java

```
package gr.eshop7.eshop.site.views;

import gr.eshop7.eshop.Config;

import javax.imageio.ImageIO;
import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.IOException;
import java.io.OutputStream;

/**
 *
 * Created by george on 12/11/2015.
 */
@WebServlet(name = "ImageServlet", urlPatterns = "/image")
public class ImageServlet extends HttpServlet {

	@Override
	protected void doGet(HttpServletRequest req, HttpServletResponse response)
throws ServletException, IOException {
		String img = Config.PRODUCT_IMAGES_BASE + "/" +
req.getParameter("img");
		String ext =img.substring(img.lastIndexOf(".") + 1);

		ServletContext context = getServletContext();
		response.setContentType("image/jpg");

		File f = new File(img);
		BufferedImage bi = ImageIO.read(f);
```

```
                OutputStream out = response.getOutputStream();
                ImageIO.write(bi, "jpg", out);
                out.close();
        }
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\views\LoginView.java

```
package gr.eshop7.eshop.site.views;

import gr.eshop7.eshop.services.CategoryService;
import gr.eshop7.eshop.services.ManufacturerService;
import gr.eshop7.eshop.site.util.WebPart;
import org.apache.velocity.context.Context;

import javax.inject.Inject;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Created on 11/11/2015.
 *
 * @author george
 */
@WebServlet(name = "LoginServlet", urlPatterns = "/login")
public class LoginView extends AbstractEshopServlet {

        @Inject
        CategoryService categoryService;

        @Inject
        ManufacturerService manufacturerService;

        @Override
        protected String getTemplateName() {
                return "login.vm";
        }

        @Override
        protected void process(HttpServletRequest request, HttpServletResponse
response, Context context) {
                WebPart page = new WebPart("Login page", "", "/login");
                page.addChildren(categoryService.getPartsList(getServletContext()));

        page.addSecondaryChildren(manufacturerService.getPartsList(getServletContext
()));

                context.put("PAGE", page);
        }

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\views\MainView.java

```
package gr.eshop7.eshop.site.views;

import gr.eshop7.eshop.services.CategoryService;
import gr.eshop7.eshop.services.ManufacturerService;
import gr.eshop7.eshop.services.ProductService;
import gr.eshop7.eshop.site.util.WebPart;
import org.apache.velocity.context.Context;

import javax.inject.Inject;
```

```
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Created on 11/11/2015.
 *
 * @author george
 */
@WebServlet(name = "MainServlet", urlPatterns = "/main")
public class MainView extends AbstractEshopServlet {

	@Inject
	CategoryService categoryService;

	@Inject
	ManufacturerService manufacturerService;

	@Inject
	ProductService productService;

	@Override
	protected String getTemplateName() {
		return "main.vm";
	}

	@Override
	protected void process(HttpServletRequest request, HttpServletResponse
response, Context context) {
		WebPart page = new WebPart("Home page", "", getHomeUrl());
		page.addChildren(categoryService.getPartsList(getServletContext()));

	page.addSecondaryChildren(manufacturerService.getPartsList(getServletContext
()));


	page.addContents(productService.getPartsListForCategory(getServletContext())
);

		context.put("PAGE", page);
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\views\ProductView.java

```
package gr.eshop7.eshop.site.views;

import gr.eshop7.eshop.services.CategoryService;
import gr.eshop7.eshop.services.ManufacturerService;
import gr.eshop7.eshop.services.ProductService;
import gr.eshop7.eshop.site.util.ProductPart;
import org.apache.velocity.context.Context;

import javax.inject.Inject;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Created on 11/11/2015.
 *
 * @author george
 */
@WebServlet(name = "ProductServlet", urlPatterns = "/product")
```

```
public class ProductView extends AbstractEshopServlet {

	@Inject
	CategoryService categoryService;

	@Inject
	ManufacturerService manufacturerService;

	@Inject
	ProductService productService;

	@Override
	protected String getTemplateName() {
		return "product.vm";
	}

	@Override
	protected void process(HttpServletRequest request, HttpServletResponse
response, Context context) {
		long id = Long.parseLong(request.getParameter("id"));

		ProductPart page = productService.getProductPart(id,
getServletContext());
		page.addChildren(categoryService.getPartsList(getServletContext()));

	page.addSecondaryChildren(manufacturerService.getPartsList(getServletContext
()));

		context.put("PAGE", page);
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\views\ProfileView.java

```
package gr.eshop7.eshop.site.views;

import gr.eshop7.eshop.services.CategoryService;
import gr.eshop7.eshop.services.ManufacturerService;
import gr.eshop7.eshop.services.OrderService;
import gr.eshop7.eshop.site.util.WebPart;
import org.apache.velocity.context.Context;
import org.eshop7.model.Customer;

import javax.inject.Inject;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Created on 11/11/2015.
 *
 * @author george
 */
@WebServlet(name = "ProfileServlet", urlPatterns = "/profile")
public class ProfileView extends AbstractEshopServlet {

	@Inject
	CategoryService categoryService;

	@Inject
	ManufacturerService manufacturerService;

	@Inject
	OrderService orderService;
```

```
	@Override
	protected String getTemplateName() {
		return "profile.vm";
	}

	@Override
	protected void process(HttpServletRequest request, HttpServletResponse
response, Context context) {
		WebPart page = new WebPart("Profile page", "", "/profile");
		page.addChildren(categoryService.getPartsList(getServletContext()));

	page.addSecondaryChildren(manufacturerService.getPartsList(getServletContext
()));

		Customer customer = (Customer)
request.getSession().getAttribute(AbstractEshopServlet.SES_CUSTOMER);

		context.put("PAGE", page);
		context.put("ORDERS", orderService.getList(customer));
	}

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\site\views\RegisterView.java

```
package gr.eshop7.eshop.site.views;

import gr.eshop7.eshop.services.CategoryService;
import gr.eshop7.eshop.services.ManufacturerService;
import gr.eshop7.eshop.site.util.WebPart;
import org.apache.velocity.context.Context;

import javax.inject.Inject;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 * Created on 11/11/2015.
 *
 * @author george
 */
@WebServlet(name = "RegisterServlet", urlPatterns = "/register")
public class RegisterView extends AbstractEshopServlet {

	@Inject
	CategoryService categoryService;

	@Inject
	ManufacturerService manufacturerService;

	@Override
	protected String getTemplateName() {
		return "register.vm";
	}

	@Override
	protected void process(HttpServletRequest request, HttpServletResponse
response, Context context) {
		WebPart page = new WebPart("Registration page", "", "/login");
		page.addChildren(categoryService.getPartsList(getServletContext()));

	page.addSecondaryChildren(manufacturerService.getPartsList(getServletContext
()));
```

```
                context.put("PAGE", page);
        }

}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\util\DateUtil.java

```
package gr.eshop7.eshop.util;

import java.util.Calendar;
import java.util.Date;

/**
 *
 * Created by george on 20/03/2016.
 */
public class DateUtil {

        public static Date addMonth(int monthsNo) {
                return addMonth(monthsNo, new Date());
        }

        public static Date addMonth(int monthsNo, Date date) {
                Calendar calendar = Calendar.getInstance();
                calendar.setTime(date);
                calendar.add(Calendar.MONTH, monthsNo);

                return calendar.getTime();
        }

        public static Date dayStart(Date date) {
                Calendar calendar = Calendar.getInstance();
                calendar.setTime(date);
                calendar.set(Calendar.HOUR_OF_DAY, 0);
                calendar.set(Calendar.MINUTE, 0);
                calendar.set(Calendar.SECOND, 0);

                return calendar.getTime();
        }

        public static Date dayEnd(Date date) {
                Calendar calendar = Calendar.getInstance();
                calendar.setTime(date);
                calendar.set(Calendar.HOUR_OF_DAY, 23);
                calendar.set(Calendar.MINUTE, 59);
                calendar.set(Calendar.SECOND, 59);

                return calendar.getTime();
        }
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\util\DoubleValue.java

```
package gr.eshop7.eshop.util;

/**
 *
 * Created by george
 */
public class DoubleValue {
        private String label;
        private Object value;
```

```
	public DoubleValue(String label, Object value) {
		this.label = label;
		this.value = value;
	}

	public String getLabel() {
		return label;
	}

	public void setLabel(String label) {
		this.label = label;
	}

	public Object getValue() {
		return value;
	}

	public void setValue(Object value) {
		this.value = value;
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\util\TripleValue.java

```
package gr.eshop7.eshop.util;

/**
 *
 * Created by george
 */
public class TripleValue {
	private String label;
	private Object value1, value2;

	public TripleValue(String label, Object value1, Object value2) {
		this.label = label;
		this.value1 = value1;
		this.value2 = value2;
	}

	public String getLabel() {
		return label;
	}

	public void setLabel(String label) {
		this.label = label;
	}

	public Object getValue1() {
		return value1;
	}

	public void setValue1(Object value1) {
		this.value1 = value1;
	}

	public Object getValue2() {
		return value2;
	}

	public void setValue2(Object value2) {
		this.value2 = value2;
	}
}
```

Αρχείο eshop\src\main\java\gr\eshop7\eshop\util\Utils.java

```
package gr.eshop7.eshop.util;


import java.math.BigInteger;
import java.security.SecureRandom;

public class Utils {

	public static boolean isEmpty(String str) {
		return str == null || str.length() == 0;
	}

	public static String generateOrderCode() {
		SecureRandom random = new SecureRandom();
		String ref = new BigInteger(10, random).toString(32)
				+ "-"
				+ new BigInteger(20, random).toString(20);

		return ref.toUpperCase();
	}

	public static String generatePaymentInfo(String ccType, String ccNumber,
String ccName, String ccCcv) {
		return String.format("%s\n****-****-****-%s (%s)\n%s", ccType,
ccNumber.substring(15), ccCcv, ccName);
	}
}
```

Αρχείο eshop\src\main\webapp\index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta http-equiv="refresh" content="0; url=http://localhost:8080/eshop7/main"
/>
</head>
<body/>
</html>
```

Αρχείο eshop\src\main\webapp\META-INF\mysql-ds.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<datasources xmlns="http://www.jboss.org/ironjacamar/schema"
             xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
             xsi:schemaLocation="http://www.jboss.org/ironjacamar/schema
http://docs.jboss.org/ironjacamar/schema/datasources_1_1.xsd">

    <datasource jndi-name="java:jboss/datasources/eshopDS" pool-name="eshopDS"
enabled="true" use-java-context="true">
        <connection-url>${datasource.connectionURL}</connection-url>
        <driver>${datasource.driver}</driver>
        <security>
            <user-name>${datasource.username}</user-name>
            <password>${datasource.password}</password>
        </security>
    </datasource>
</datasources>
```

Αρχείο eshop\src\main\webapp\WEB-INF\templates\category.vm

```
#parse('include/header.vm')
        <div class="col-md-9">
            <div class="row">
```

```
                    #if($PAGE.contents.size() == 0)
            <div class="col-sm-12 col-lg-12 col-md-12">
                            <h2>No products found.</h2>
                            <p>The selected category has no products. Please
select another category.</p>
                    </div>
                    #end
                    #foreach($product in $PAGE.contents)
                <div class="col-sm-4 col-lg-4 col-md-4">
                    <div class="thumbnail">
                        <img src="$product.image" alt="">
                        <div class="caption">
                            <h5 style="white-space: normal; height: 60px; max-
height:60px; overflow: hidden;">
                                                                <a
href="$product.url">$product.name</a>
                            </h5>
                                                        <div class="clearfix">
                                                                <h5 class="pull-left">
                                                                #if(!$product.inStock)
                                                                        <span
style="display: block; padding-top: 9px;">Out of stock</span>
                                                                #elseif($SITE.loggedIn)
                                                                        <a
href="$product.buyUrl" class="btn btn-primary btn-lg active" role="button"
style="padding: 2px 16px;">Buy now</a>
                                                                #else
                                                                        <a
href="$SITE.base/login" style="display: block; padding-top: 9px;">Log in to
buy!</a>
                                                                #end
                                                                </h5>
                                    <h4 class="pull-right" style="padding-top:
6px;"><strong>$product.formattedPrice &euro;</strong></h4>
                            </div>
                        </div>
                    </div>
                </div>
                    #end

                    </div>
                </div>

#parse('include/footer.vm')
```

Αρχείο eshop\src\main\webapp\WEB-INF\templates\checkout.vm

```
#parse('include/header.vm')
                <script>
                        function storeDisable() {
                                var storePickup =
document.getElementById("deliveryPICKUP");

                                if (storePickup.checked == true) {
                                        document.getElementById("storeSelect").disabled
= false;
                                } else {
                                        document.getElementById("storeSelect").disabled
= true;
                                }
                        }

            function paymentDisable() {
                var paymentCash = document.getElementById("paymentCASH");

                if (paymentCash.checked == false) {
```

```
                document.getElementById("ccType").disabled = false;
                document.getElementById("ccNumber").disabled = false;
                document.getElementById("ccName").disabled = false;
                document.getElementById("ccCcv").disabled = false;
            } else {
                document.getElementById("ccType").disabled = true;
                document.getElementById("ccNumber").disabled = true;
                document.getElementById("ccName").disabled = true;
                document.getElementById("ccCcv").disabled = true;
            }
        }
          </script>
          <div class="col-md-9">
        <div class="row">
                        <div class="col-md-12">
                                <h2>Checkout page</h2>

                                <p>You are going to purchase this product:</p>
                        </div>
                  </div>

        <div class="row">
            <div class="col-md-12">
                <h3>Product:</h3>
                                <table class="table">
                                        <thead>
                                                <tr>
                           <th>Name</th>
                           <th>Price</th>
                                                </tr>
                                        </thead>
                                        <tbody>
                        <tr>
                                                        <td>$PAGE.name</td>
                           <td>$PAGE.formattedPrice &euro;</td>
                                                </tr>
                                        </tbody>
                                </table>

            </div>
        </div>

        <div class="row">
            <div class="col-md-12">
                                <p>Please fill in the details and review the
following info.</p>

                      <p>Once you are ready, press the Purchase button to submit
your order.</p>
                <fieldset>
                    <form action="$PAGE.buyActionUrl" method="post">
                        <div class="form-group">
                            <label for="emailField">Email address</label>
                            <input type="email" name="email" class="form-
control" id="emailField" value="$SITE.customer.email"
                                    placeholder="Email" required>
                        </div>
                        <div class="form-group">
                            <label for="nameField">Name</label>
                            <input type="text" name="name" class="form-control"
id="nameField"
                                    value="$SITE.customer.name"
                                    placeholder="Name" required>
                        </div>
                        <div class="form-group">
                            <label for="addressField">Address</label>
```

```
                                <input type="text" name="address" class="form-
control" id="addressField"
                                       value="$SITE.customer.address"
                                       placeholder="Address" required>
                            </div>
                            <div class="form-group">
                                <label for="phoneField">Phone number</label>
                                <input type="text" name="phone" class="form-
control" id="phoneField"
                                       value="$SITE.customer.phone"
                                       placeholder="Phone number" required>
                            </div>
                            <div class="form-group">
                                <label>Delivery method</label>

                                <div class="radio">
                                    <label>
                                                                                <input
type="radio" name="orderShipping" value="COURIER" id="deliveryCOURIER"
onclick="storeDisable()" checked>
                                                                                Send by
courier
                                                                        </label>
                                </div>
                                <div class="radio">
                                    <label>
                                                                                <input
type="radio" name="orderShipping" value="PICKUP" id="deliveryPICKUP"
onclick="storeDisable()">
                                                                                Store
pickup
                                                                        </label>
                                </div>
                            </div>
                            <fieldset class="left-intend">
                                                                <div class="form-group">
                                                                        <label
for="storeSelect">Pickup store</label>
                                                                        <select
id="storeSelect" class="form-control" name="storeId" disabled>
                                                                        #foreach($store in
$STORES)
                                                                                <option
value="$store.id">$store.name ($store.address) #if($foreach.index ==
0)selected#end</option>
                                                                        #end
                                                                        </select>
                                                                </div>
                                                            </fieldset>
                            <div class="form-group">
                                <label>Payment method</label>
                                <div class="radio">
                                    <label>
                                        <input type="radio" name="orderPayment"
value="CASH" id="paymentCASH"
                                               onclick="paymentDisable()" checked>
                                        Cash
                                    </label>
                                </div>
                                <div class="radio">
                                    <label>
                                        <input type="radio" name="orderPayment"
value="CREDIT_CARD" id="paymentCREDIT_CARD"
                                               onclick="paymentDisable()">
                                        Credit card
                                    </label>
```

```
                                </div>
                            </div>
                                                <fieldset class="left-intend">
                                <div class="form-group">
                                    <label for="ccType">Credit card type</label>
                                                                    <select
name="ccType" class="form-control" id="ccType" disabled>

        <option>Visa</option>

        <option>Mastercard</option>
                                                                    </select>
                                </div>
                                <div class="form-group">
                                                                    <label
for="ccNumber">Credit card number</label>
                                                                    <input
pattern=".{4,4}-.{4,4}-.{4,4}-.{4,4}" name="ccNumber" class="form-control"
id="ccNumber" required disabled

placeholder="XXXX-XXXX-XXXX-XXXX" title="Enter you credit card number, eg 1111-
2222-3333-4444.">
                                                                </div>
                                                                <div class="form-group">
                                                                    <label
for="ccName">Holder name</label>
                                                                    <input type="text"
name="ccName" class="form-control" id="ccName" required disabled

placeholder="eg. JOHN SMITH">
                                                                </div>
                                                                <div class="form-group">
                                                                    <label
for="ccCcv">CCV</label>
                                                                    <input
pattern=".{3,3}" name="ccCcv" class="form-control" id="ccCcv" required disabled

placeholder="XXX">
                                                                </div>
                            </fieldset>
                            <button type="submit" class="btn btn-
default">Purchase</button>
                        </form>
                    </fieldset>
                             </div>
            </div>
               </div>
#parse('include/footer.vm')
```

Αρχείο eshop\src\main\webapp\WEB-INF\templates\include\footer.vm

```
</div><!-- /.row -->


</div><!-- /.container -->

<div class="container">

    <hr>

    <!-- Footer -->
    <footer>
        <div class="row">
            <div class="col-lg-12">
```

```
                <p>TEI of Crete, School of Engineering, Department of Informatics
Engineering<br>Graduate Thesis, George Genetzakis, Niki Mallia<br>Advisor: Nikolaos
Papadakis</p>
            </div>
        </div>
    </footer>

</div>
<!-- /.container -->

<!-- jQuery -->
<script src="$SITE.base/js/jquery.js"></script>

<!-- Bootstrap Core JavaScript -->
<script src="$SITE.base/js/bootstrap.min.js"></script>
<script>
       $(function() {
        $().alert();
       });
</script>
</body>

</html>
$ACTION.cleanup()
```

Αρχείο eshop\src\main\webapp\WEB-INF\templates\include\header.vm

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="">
    <meta name="author" content="">

    <title>$SITE.name</title>

    <!-- Bootstrap Core CSS -->
    <link href="$SITE.base/css/bootstrap.css" rel="stylesheet">

    <!-- Custom CSS -->
    <link href="$SITE.base/css/shop-homepage.css" rel="stylesheet">

    <!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements and media queries
-->
    <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
    <!--[if lt IE 9]>
       <script
src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js"></script>
       <script
src="https://oss.maxcdn.com/libs/respond.js/1.4.2/respond.min.js"></script>
    <![endif]-->




</head>

<body>

<!-- Navigation -->
<nav class="navbar navbar-inverse navbar-fixed-top" role="navigation">
    <div class="container">
        <!-- Brand and toggle get grouped for better mobile display -->
```

```
        <div class="navbar-header">
            <button type="button" class="navbar-toggle" data-toggle="collapse"
data-target="#bs-example-navbar-collapse-1">
                <span class="sr-only">Toggle navigation</span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
                <span class="icon-bar"></span>
            </button>
            <a class="navbar-brand" href="$SITE.base">$SITE.name</a>
        </div>
        <!-- Collect the nav links, forms, and other content for toggling -->
        <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
            <ul class="nav navbar-nav">
                      #if($SITE.loggedIn)
                <li>
                    <a href="$SITE.base/profile">My orders</a>
                </li>
                <li>
                    <a href="$SITE.base/logoutAction">Logout</a>
                </li>
                             #if($SITE.customer.admin)
                <li>
                    <a href="$SITE.base/admin">Admin</a>
                </li>
                             #end
                      #else
                <li>
                    <a href="$SITE.base/login">Login</a>
                </li>

                <li>
                    <a href="$SITE.base/register">Register</a>
                </li>
                      #end
            </ul>
        </div>
        <!-- /.navbar-collapse -->
    </div>
    <!-- /.container -->
</nav>

<!-- Page Content -->
<div class="container">
#if($SITE.hasMessage)
    <div class="row">
        <div class="alert alert-info alert-dismissible fade in" role="alert">
            <button type="button" class="close" data-dismiss="alert" aria-
label="Close"><span
                    aria-hidden="true">&times;</span></button>
            $SITE.message
        </div>
       </div>
#end

    <div class="row">
        <div class="col-md-3">
            <p class="lead">$PAGE.name</p>
            <div class="list-group">
                             <h4>Categories</h4>
                             #foreach($child in $PAGE.children)
                    <a href="$child.url" class="list-group-item">$child.name</a>
                             #end
            </div>

            <div class="list-group">
                <h4>Manufacturers</h4>
```

```
                            #foreach($child in $PAGE.secondaryChildren)
                    <a href="$child.url" class="list-group-item">$child.name</a>
                            #end
            </div>

              </div>
```

Αρχείο eshop\src\main\webapp\WEB-INF\templates\login.vm

```
#parse('include/header.vm')
        <div class="col-md-9">

            <div class="row carousel-holder">

                            <div class="col-md-12">
                                    <h2>Login page</h2>

                                    <p>Please enter your username and password to
login.</p>
                                    <p>If you don't have an account you can <a
href="$SITE.base/register">create one here.</a></p>


                                    <fieldset>
                                            <form action="$SITE.base/loginAction"
method="post">
                            <div class="form-group">
                                <label for="emailField">Email address</label>
                                <input type="email" name="email" class="form-
control" id="emailField" placeholder="Email" required>
                            </div>
                            <div class="form-group">
                                <label for="passwordField">Password</label>
                                <input type="password" name="password" class="form-
control" id="passwordField" placeholder="Password" required>
                            </div>
                            <button type="submit" class="btn btn-
default">Login</button>
                        </form>
                                    </fieldset>
                            </div>
            </div>
              </div>
#parse('include/footer.vm')
```

Αρχείο eshop\src\main\webapp\WEB-INF\templates\main.vm

```
#parse('include/header.vm')
        <div class="col-md-9">

            <div class="row carousel-holder">

                <div class="col-md-12">
                    <div id="carousel-example-generic" class="carousel slide" data-
ride="carousel">
                        <ol class="carousel-indicators">
                                                #foreach($product in
$PAGE.contents)
                                <li data-target="#carousel-example-generic" data-
slide-to="$foreach.index" #if($foreach.index == 0)class="active"#end></li>
                                                #end

                        </ol>
                        <div class="carousel-inner">
                                                #foreach($product in
$PAGE.contents)
```

```
                                                    <div class="item
#if($foreach.index == 0)active"#end">
                                                           <img class="slide-image"
src="$product.image" alt="$product.name">
                                                    </div>
                                                    #end
                        </div>
                        <a class="left carousel-control" href="#carousel-example-
generic" data-slide="prev">
                            <span class="glyphicon glyphicon-chevron-left"></span>
                        </a>
                        <a class="right carousel-control" href="#carousel-example-
generic" data-slide="next">
                            <span class="glyphicon glyphicon-chevron-right"></span>
                        </a>
                    </div>
                </div>

                            <div class="col-md-12">
                                    <h2>Welcome to the Electronic's store!</h2>

                                    <p>Our products are spread out in a number of
categories. Please select a category from the left to view its contents.</p>
                            </div>
            </div>
               </div>
#parse('include/footer.vm')
```

Αρχείο eshop\src\main\webapp\WEB-INF\templates\product.vm

```
#parse('include/header.vm')
        <div class="col-md-9">
            <div class="row">
                <div class="col-lg-12">
                                    <h1>$PAGE.name</h1>
                    <div class="clearfix">

                    <h6 class="buy-container text-left pull-left">
                                            #if(!$PAGE.inStock)
                            <span style="font-size: 14px;margin-top: 10px;display:
block;">Out of stock</span>
                                            #elseif($SITE.loggedIn)
                            <a href="$PAGE.buyUrl" class="btn btn-primary btn-lg
active" role="button" style="padding: 2px 16px;">Buy now</a>
                                            #else
                            <a href="$SITE.base/login" style="font-size:
14px;margin-top: 10px;display: block;">Log in to buy!</a>
                                            #end
                    </h6>

                                    <h3 class="pull-left" style="padding-top:3px;
margin-left:20px;">$PAGE.formattedPrice &euro;</h3>

                                    </div>
                    <img class="img-responsive" src="$PAGE.image">

                    <h5>Description:</h5>
                                    <p>$PAGE.description</p>
                            </div>
                      </div>
               </div>
#parse('include/footer.vm')
```

Αρχείο eshop\src\main\webapp\WEB-INF\templates\profile.vm

```
#parse('include/header.vm')
```

```
        <div class="col-md-9">

            <div class="row carousel-holder">

                            <div class="col-md-12">
                                   <h2>Orders</h2>

                                   #if(!$ORDERS.isEmpty())
                                   <table class="table table-hover table-striped
table-responsive">
                                          <thead>
                                                 <tr>
                                                        <th>Order code</th>
                                                        <th>Products</th>
                                                        <th>Total</th>
                                                        <th>Shipping</th>
                                                        <th>Placed on</th>
                                                        <th>Status</th>
                                                 </tr>
                                          </thead>
                                          <tbody>
                                          #foreach($order in $ORDERS)
                                                 <tr>
                                                        <td>$order.ref</td>

       <td>$order.product.name</td>
                                                        <td>$order.price
&euro;</td>
                                                        <td>

       #if($order.orderShipping.name() == 'PICKUP')
                                                               Pickup from
store:<br>

       $order.store.name<br>

       $order.store.address
                                                        #else
                                                               Delivery by
courier:<br>
                                                               $order.name<br>
                                                               $order.address<br>
                                                               $order.phone
                                                        #end
                                                        </td>

       <td>$order.submissionDate</td>
                                                        <td>

       #if($order.orderStatus.name() == 'SUBMITTED')
                                   Pending<br>
                                   <a class="button"
href="$SITE.base/completeOrderAction?action=CANCEL&orderId=$order.id">Cancel</a>

       #elseif($order.orderStatus.name() == 'IN_PROGRESS')
                                                               In progress<br>
                                   <a class="button"
href="$SITE.base/completeOrderAction?action=CANCEL&orderId=$order.id">Cancel</a>

       #elseif($order.orderStatus.name() == 'DELIVERED')
                                   Delivered on
<br>$ACTION.formatDate($order.deliveryDate)

       #elseif($order.orderStatus.name() == 'CANCELLED')
                                   Cancelled
                                                        #end
```

```
                                                                    </td>
                                                            </tr>
                                                    #end
                                                    </tbody>
                                            </table>
                                            #else
                                                    <p>You have not placed any orders
yet.</p>
                                            #end
                                    </div>
                    </div>
                       </div>
#parse('include/footer.vm')
```

Αρχείο eshop\src\main\webapp\WEB-INF\templates\register.vm

```
#parse('include/header.vm')
        <div class="col-md-9">

            <div class="row carousel-holder">

                                    <div class="col-md-12">
                                            <h2>Create a new account</h2>

                                            <p>In this page you can create a new account,
in order to be able to make purchases.</p>
                                            <p>Please enter your information to
proceed.</p>
                                            <p>If you already have an account you should <a
href="$SITE.base/login">instead login here.</a></p>


                                            <fieldset>
                                                    <form action="$SITE.base/registerAction"
method="post">
                                    <div class="form-group">
                                        <label for="emailField">Email address</label>
                                        <input type="email" name="email" class="form-
control" id="emailField" placeholder="Email" required>
                                    </div>
                                    <div class="form-group">
                                        <label for="passwordField">Password</label>
                                        <input type="password" name="password" class="form-
control" id="passwordField" placeholder="Password" required>
                                    </div>
                                    <div class="form-group">
                                        <label for="nameField">Name</label>
                                        <input type="text" name="name" class="form-control"
id="nameField"
                                                   placeholder="Name" required>
                                    </div>
                                    <div class="form-group">
                                        <label for="addressField">Address</label>
                                        <input type="text" name="address" class="form-
control" id="addressField"
                                                   placeholder="Address" required>
                                    </div>
                                    <div class="form-group">
                                        <label for="phoneField">Phone number</label>
                                        <input type="text" name="phone" class="form-
control" id="phoneField"
                                                   placeholder="Phone number" required>
                                    </div>
                                    <button type="submit" class="btn btn-
default">Register</button>
```

```
                        </form>
                                    </fieldset>
                              </div>
            </div>
               </div>
#parse('include/footer.vm')
```

Αρχείο eshop\src\main\webapp\WEB-INF\velocity.properties

```
webapp.resource.loader.path=WEB-INF/templates
webapp.resource.loader.cache = false
input.encoding=UTF-8
output.encoding=UTF-8
```

Αρχείο eshop\src\main\webapp\WEB-INF\web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
         version="3.0">

    <context-param>
        <!-- Avoid "Found multiple containers, please specify which one to use"
warning -->
        <param-name>org.atmosphere.cpr.asyncSupport</param-name>
        <param-value>org.atmosphere.container.JBossWebSocketSupport</param-value>
    </context-param>

    <context-param>
        <param-name>org.atmosphere.cpr.sessionSupport</param-name>
        <param-value>true</param-value>
    </context-param>

    <listener>
        <listener-class>org.atmosphere.cpr.SessionSupport</listener-class>
    </listener>

       <servlet>
              <servlet-name>VaadinCDI</servlet-name>
              <servlet-class>com.vaadin.cdi.server.VaadinCDIServlet</servlet-class>
       </servlet>
       <servlet-mapping>
              <servlet-name>VaadinCDI</servlet-name>
              <url-pattern>/admin/*</url-pattern>
              <url-pattern>/VAADIN/*</url-pattern>
       </servlet-mapping>

    <persistence-context-ref>
        <persistence-context-ref-name>persistence/em</persistence-context-ref-name>
        <persistence-unit-name>eshop</persistence-unit-name>
    </persistence-context-ref>
</web-app>
```

Αρχείο model\pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <parent>
        <artifactId>eshop-parent</artifactId>
        <groupId>gr.eshop7</groupId>
```

```
        <version>1.0-SNAPSHOT</version>
    </parent>
    <modelVersion>4.0.0</modelVersion>

    <artifactId>model</artifactId>

    <dependencies>

        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-entitymanager</artifactId>
        </dependency>
               <dependency>
                       <groupId>javax</groupId>
                       <artifactId>javaee-web-api</artifactId>
                       <version>7.0</version>
               </dependency>

       </dependencies>

</project>
```

Αρχείο model\src\main\java\org\eshop7\model\Category.java

```
package org.eshop7.model;

import org.eshop7.model.misc.PublishedObject;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.OneToMany;
import java.util.ArrayList;
import java.util.List;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
@Entity
public class Category extends PublishedObject {

       private String name;

       @OneToMany(mappedBy="category", fetch = FetchType.EAGER, cascade =
{CascadeType.REMOVE})
       private List<Product> products = new ArrayList<>();

       public Category() {
       }

       public String getName() {
              return name;
       }

       public void setName(String name) {
              this.name = name;
       }
}
```

Αρχείο model\src\main\java\org\eshop7\model\Customer.java

```
package org.eshop7.model;
```

```
import org.eshop7.model.misc.PublishedObject;

import javax.persistence.Column;
import javax.persistence.Entity;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
@Entity
public class Customer extends PublishedObject {
	private String email;
	private String password;

	private String name;
	private String address;
	private String phone;

	@Column(columnDefinition = "boolean NOT NULL DEFAULT 0 ")
	private boolean admin = false;

	public Customer() {
	}

	public Customer(String email, String password, String name, String address,
String phone) {
		this.email = email;
		this.password = password;
		this.name = name;
		this.address = address;
		this.phone = phone;
	}

	public String getEmail() {
		return email;
	}

	public void setEmail(String username) {
		this.email = username;
	}

	public String getPassword() {
		return password;
	}

	public void setPassword(String password) {
		this.password = password;
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public String getAddress() {
		return address;
	}

	public void setAddress(String address) {
		this.address = address;
	}
```

```
	public String getPhone() {
		return phone;
	}

	public void setPhone(String phone) {
		this.phone = phone;
	}

	public boolean isAdmin() {
		return admin;
	}

	public void setAdmin(boolean admin) {
		this.admin = admin;
	}

	@Override
	public String toString() {
		return String.format("%s (%d)", name, getId());
	}
}
```

Αρχείο model\src\main\java\org\eshop7\model\CustomerOrder.java

```
package org.eshop7.model;

import javax.persistence.*;
import java.util.Date;

/**
 *
 * Created by george
 */
@Entity
public class CustomerOrder extends IdObject {

	@Enumerated(EnumType.STRING)
	private OrderStatus orderStatus;

	@Enumerated(EnumType.STRING)
	private OrderShipping orderShipping;

	@Enumerated(EnumType.STRING)
	private OrderPayment orderPayment;

	@ManyToOne
	@JoinColumn(name = "customerId")
	private Customer customer;

	@ManyToOne
	@JoinColumn(name = "productId")
	private Product product;

	@ManyToOne
	@JoinColumn(name = "storeId")
	private Store store;

	private float price;

	private String ref;

	private String name;
	private String address;
	private String phone;
```

```
        private Date submissionDate;
        private Date deliveryDate;

        private String paymentInfo;

        public CustomerOrder() {
        }

        public CustomerOrder(OrderStatus orderStatus, OrderShipping orderShipping,
OrderPayment orderPayment, Customer customer,
                                                Product product, Store store, float
price, String ref, String name, String address, String phone,
                                                Date submissionDate, Date deliveryDate,
String paymentInfo) {
                this.orderStatus = orderStatus;
                this.orderShipping = orderShipping;
                this.orderPayment = orderPayment;
                this.customer = customer;
                this.product = product;
                this.store = store;
                this.price = price;
                this.ref = ref;
                this.name = name;
                this.address = address;
                this.phone = phone;
                this.submissionDate = submissionDate;
                this.deliveryDate = deliveryDate;
                this.paymentInfo = paymentInfo;
        }

        public OrderStatus getOrderStatus() {
                return orderStatus;
        }

        public void setOrderStatus(OrderStatus orderStatus) {
                this.orderStatus = orderStatus;
        }

        public OrderShipping getOrderShipping() {
                return orderShipping;
        }

        public void setOrderShipping(OrderShipping orderShipping) {
                this.orderShipping = orderShipping;
        }

        public OrderPayment getOrderPayment() {
                return orderPayment;
        }

        public void setOrderPayment(OrderPayment orderPayment) {
                this.orderPayment = orderPayment;
        }

        public Customer getCustomer() {
                return customer;
        }

        public void setCustomer(Customer customer) {
                this.customer = customer;
        }

        public Product getProduct() {
                return product;
        }
```

```
public void setProduct(Product product) {
	this.product = product;
}

public Store getStore() {
	return store;
}

public void setStore(Store store) {
	this.store = store;
}

public float getPrice() {
	return price;
}

public void setPrice(float price) {
	this.price = price;
}

public String getRef() {
	return ref;
}

public void setRef(String ref) {
	this.ref = ref;
}

public String getName() {
	return name;
}

public void setName(String name) {
	this.name = name;
}

public String getAddress() {
	return address;
}

public void setAddress(String address) {
	this.address = address;
}

public String getPhone() {
	return phone;
}

public void setPhone(String phone) {
	this.phone = phone;
}

public Date getSubmissionDate() {
	return submissionDate;
}

public void setSubmissionDate(Date submissionDate) {
	this.submissionDate = submissionDate;
}

public Date getDeliveryDate() {
	return deliveryDate;
}

public void setDeliveryDate(Date deliveryDate) {
	this.deliveryDate = deliveryDate;
```

```
        }

        public String getPaymentInfo() {
                return paymentInfo;
        }

        public void setPaymentInfo(String paymentInfo) {
                this.paymentInfo = paymentInfo;
        }
}
```

Αρχείο model\src\main\java\org\eshop7\model\IdObject.java

```
package org.eshop7.model;

import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.MappedSuperclass;
import java.io.Serializable;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
@MappedSuperclass
public class IdObject implements Serializable {
        @Id
        @GeneratedValue(strategy= GenerationType.IDENTITY)
        private long id;

        public IdObject() {
        }

        public long getId() {
                return id;
        }

        public void setId(long id) {
                this.id = id;
        }

        @Override
        public boolean equals(Object o) {
                if (this == o) return true;
                if (!(o instanceof IdObject)) return false;

                IdObject idObject = (IdObject) o;

                return getId() == idObject.getId();

        }

        @Override
        public int hashCode() {
                return (int) (getId() ^ (getId() >>> 32));
        }

        @Override
        public String toString() {
                return getClass().getSimpleName() + "(" + getId() + ")";
        }
}
```

Αρχείο model\src\main\java\org\eshop7\model\Manufacturer.java

```
package org.eshop7.model;

import org.eshop7.model.misc.PublishedObject;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.OneToMany;
import java.util.ArrayList;
import java.util.List;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
@Entity
public class Manufacturer extends PublishedObject {

	private String name;

	@OneToMany(mappedBy="manufacturer", fetch = FetchType.EAGER, cascade =
{CascadeType.REMOVE})
	private List<Product> products = new ArrayList<>();

	public Manufacturer() {
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}
}
```

Αρχείο model\src\main\java\org\eshop7\model\misc\Defaults.java

```
package org.eshop7.model.misc;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
public class Defaults {
	public static final String IMAGES_PATH = "c:/temp/";

}
```

Αρχείο model\src\main\java\org\eshop7\model\misc\PublishedObject.java

```
package org.eshop7.model.misc;

import org.eshop7.model.IdObject;

import javax.persistence.Column;
import javax.persistence.MappedSuperclass;

/**
```

```
 * Created by george on 13/12/2015.
 *
 */
@MappedSuperclass
public class PublishedObject extends IdObject {

	@Column(columnDefinition = "boolean NOT NULL DEFAULT 0 ")
	private boolean published = true;

	public boolean isPublished() {
		return published;
	}

	public void setPublished(boolean published) {
		this.published = published;
	}

}
```

Αρχείο model\src\main\java\org\eshop7\model\OrderPayment.java

```
package org.eshop7.model;

/**
 *
 * Created by george
 */
public enum OrderPayment {
	CASH,
	CREDIT_CARD
}
```

Αρχείο model\src\main\java\org\eshop7\model\OrderShipping.java

```
package org.eshop7.model;

/**
 *
 * Created by george
 */
public enum OrderShipping {
	PICKUP,
	COURIER
}
```

Αρχείο model\src\main\java\org\eshop7\model\OrderStatus.java

```
package org.eshop7.model;

/**
 *
 * Created by george
 */
public enum OrderStatus {
	SUBMITTED,
	IN_PROGRESS,
	DELIVERED,
	CANCELLED
}
```

Αρχείο model\src\main\java\org\eshop7\model\Product.java

```
package org.eshop7.model;

import org.eshop7.model.misc.PublishedObject;

import javax.persistence.*;
import java.util.HashSet;
import java.util.Set;

/**
 * Created on 19/4/2015.
 *
 * @author george
 */
@Entity
public class Product extends PublishedObject {
	private String name;

	@Column(columnDefinition="text")
	private String description;

	private String image;

	private float price;

	@ManyToOne(cascade = {CascadeType.MERGE})
	@JoinColumn(name = "categoryId")
	private Category category;

	@ManyToOne(cascade = {CascadeType.MERGE})
	@JoinColumn(name = "manufacturerId")
	private Manufacturer manufacturer;

	@OneToMany(mappedBy = "product", fetch = FetchType.EAGER, orphanRemoval =
true,
			cascade = {CascadeType.PERSIST, CascadeType.MERGE})
	private Set<Stock> stock = new HashSet<>();

	public Product() {
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public String getImage() {
		return image;
	}

	public void setImage(String image) {
		this.image = image;
	}

	public Category getCategory() {
		return category;
	}

	public void setCategory(Category category) {
		this.category = category;
	}

	public float getPrice() {
		return price;
```

```
	}

	public void setPrice(float price) {
		this.price = price;
	}

	public String getDescription() {
		return description;
	}

	public void setDescription(String description) {
		this.description = description;
	}

	public Manufacturer getManufacturer() {
		return manufacturer;
	}

	public void setManufacturer(Manufacturer manufacturer) {
		this.manufacturer = manufacturer;
	}

	public Set<Stock> getStock() {
		return stock;
	}

	public void setStock(Set<Stock> stock) {
		this.stock = stock;
	}

	@Override
	public String toString() {
		return name;
	}
}
```

Αρχείο model\src\main\java\org\eshop7\model\ProductOrder.java

```
package org.eshop7.model;

import javax.persistence.Entity;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
@Entity
public class ProductOrder extends IdObject {
	public final static String SUBMITTED = "SUBMITTED";
	public final static String CANCELLED = "CANCELLED";
	public final static String SENT = "SENT";

	private String code;
	private String product;
	private String customer;
	private String customerAddress;
	private String customerId;
	private String productId;
	private String status;
	private float price;

	public ProductOrder() {
	}
```

```
	public ProductOrder(String code, String product, String customer, String
customerAddress, String customerId, String productId, String status, float price) {
		this.code = code;
		this.product = product;
		this.customer = customer;
		this.customerAddress = customerAddress;
		this.customerId = customerId;
		this.productId = productId;
		this.status = status;
		this.price = price;
	}

	public String getCode() {
		return code;
	}

	public void setCode(String code) {
		this.code = code;
	}

	public String getProduct() {
		return product;
	}

	public void setProduct(String product) {
		this.product = product;
	}

	public String getCustomer() {
		return customer;
	}

	public void setCustomer(String customer) {
		this.customer = customer;
	}

	public String getCustomerAddress() {
		return customerAddress;
	}

	public void setCustomerAddress(String customerAddress) {
		this.customerAddress = customerAddress;
	}

	public String getCustomerId() {
		return customerId;
	}

	public void setCustomerId(String customerId) {
		this.customerId = customerId;
	}

	public String getProductId() {
		return productId;
	}

	public void setProductId(String productId) {
		this.productId = productId;
	}

	public String getStatus() {
		return status;
	}

	public void setStatus(String status) {
		this.status = status;
```

```
        }

        public float getPrice() {
                return price;
        }

        public void setPrice(float price) {
                this.price = price;
        }

}
```

Αρχείο model\src\main\java\org\eshop7\model\Stock.java

```
package org.eshop7.model;

import javax.persistence.*;
import java.io.Serializable;

/**
 *
 * Created by george
 */
@Entity
public class Stock implements Serializable {

        @EmbeddedId
        private StockId id;

        @MapsId("productId")
        @ManyToOne
        @JoinColumn(name = "productId")
        private Product product;

        @MapsId("storeId")
        @ManyToOne
        @JoinColumn(name = "storeId")
        private Store store;

        @Column
        private int stock;

        public Stock() {}

        public Stock(Product product, Store store) {
                this.id = new StockId(product.getId(), store.getId());
        }

        public Stock(Product product, Store store, int stock) {
                this.id = new StockId(product.getId(), store.getId());
                this.store = store;
                this.product = product;
        }

        public StockId getId() {
                return id;
        }

        public void setId(StockId id) {
                this.id = id;
        }

        public Long getProductId() {
                return id.getProductId();
        }
```

```
        public void setStoreId(Long storeId) {
                id.setStoreId(storeId);
        }

        public Long getStoreId() {
                return id.getStoreId();
        }

        public void setProductId(Long productId) {
                id.setProductId(productId);
        }

        public int getStock() {
                return stock;
        }

        public void setStock(int stock) {
                this.stock = stock;
        }

        public Store getStore() {
                return store;
        }

        public void setStore(Store store) {
                this.store = store;
        }

        public Product getProduct() {
                return product;
        }

        public void setProduct(Product product) {
                this.product = product;
        }

        @Override
        public String toString() {
                return "Stock(" +
                                "product=" + product +
                                ", store=" + store +
                                ')';
        }
}
```

Αρχείο model\src\main\java\org\eshop7\model\StockId.java

```
package org.eshop7.model;

import javax.persistence.Embeddable;
import java.io.Serializable;

/**
 * Created on 18/12/2015.
 *
 * @author george
 */
@Embeddable
class StockId implements Serializable {
        private Long productId;

        private Long storeId;
```

```
	public StockId() {}

	public StockId(Long productId, Long storeId) {
		this.productId = productId;
		this.storeId = storeId;
	}

	public Long getProductId() {
		return productId;
	}

	public void setProductId(Long productId) {
		this.productId = productId;
	}

	public Long getStoreId() {
		return storeId;
	}

	public void setStoreId(Long storeId) {
		this.storeId = storeId;
	}

	@Override
	public boolean equals(Object o) {
		if (this == o) return true;
		if (!(o instanceof StockId)) return false;

		StockId stockId = (StockId) o;

		if (getProductId() != null ?
!getProductId().equals(stockId.getProductId()) : stockId.getProductId() != null)
			return false;
		return !(getStoreId() != null ?
!getStoreId().equals(stockId.getStoreId()) : stockId.getStoreId() != null);

	}

	@Override
	public int hashCode() {
		int result = getProductId() != null ? getProductId().hashCode() : 0;
		result = 31 * result + (getStoreId() != null ?
getStoreId().hashCode() : 0);
		return result;
	}
}
```

Αρχείο model\src\main\java\org\eshop7\model\Store.java

```
package org.eshop7.model;

import org.eshop7.model.misc.PublishedObject;

import javax.persistence.*;
import java.util.HashSet;
import java.util.Set;

/**
 * Created on 18/4/2015.
 *
 * @author george
 */
@Entity
public class Store extends PublishedObject implements Comparable<Store>{
```

```
	@Column(columnDefinition="text")
	private String name;

	private String address;
	private String phone;

	@OneToMany(mappedBy = "store", fetch = FetchType.EAGER, orphanRemoval =
true,
			cascade = {CascadeType.PERSIST, CascadeType.MERGE})
	private Set<Stock> stock = new HashSet<>();

	public Store() {
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}

	public String getAddress() {
		return address;
	}

	public void setAddress(String address) {
		this.address = address;
	}

	public String getPhone() {
		return phone;
	}

	public void setPhone(String phone) {
		this.phone = phone;
	}

	public Set<Stock> getStock() {
		return stock;
	}

	public void setStock(Set<Stock> stock) {
		this.stock = stock;
	}

	@Override
	public String toString() {
		return name;
	}

	@Override
	public int compareTo(Store o) {
		return name.compareTo(o.name);
	}
}
```

Αρχείο pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
```

```
    <groupId>gr.eshop7</groupId>
    <artifactId>eshop-parent</artifactId>
    <packaging>pom</packaging>
    <version>1.0-SNAPSHOT</version>
    <modules>
        <module>eshop</module>
        <module>model</module>
    </modules>

    <properties>
        <vaadin.version>7.6.4</vaadin.version>
        <vaadin.plugin.version>${vaadin.version}</vaadin.plugin.version>
        <jetty.plugin.version>9.2.3.v20140905</jetty.plugin.version>
        <project.source.version>1.8</project.source.version>
        <project.target.version>1.8</project.target.version>
        <project.encoding>UTF-8</project.encoding>

        <datasource.driver>mysql</datasource.driver>

<datasource.connectionURL>jdbc:mysql://127.0.0.1:3306/eshop?autoReconnect=true&amp;
amp;useUnicode=true&amp;amp;characterEncoding=utf-8</datasource.connectionURL>
        <datasource.username>root</datasource.username>
        <datasource.password></datasource.password>
    </properties>

    <build>
        <finalName>eshop7</finalName>

        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.2</version>
                <configuration>
                    <encoding>${project.encoding}</encoding>
                    <source>${project.source.version}</source>
                    <target>${project.target.version}</target>
                </configuration>
            </plugin>

            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-resources-plugin</artifactId>
                <version>2.6</version>
                <configuration>
                    <encoding>${project.encoding}</encoding>
                </configuration>
            </plugin>

            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-source-plugin</artifactId>
                <version>2.4</version>
            </plugin>

        </plugins>
    </build>

    <dependencyManagement>
        <dependencies>
            <dependency>
                <groupId>javax</groupId>
                <artifactId>javaee-web-api</artifactId>
                <version>7.0</version>
                <scope>provided</scope>
```

```
            </dependency>
            <dependency>
                <groupId>javax.servlet</groupId>
                <artifactId>javax.servlet-api</artifactId>
                <version>3.1.0</version>
                <scope>provided</scope>
            </dependency>
            <dependency>
                <groupId>org.hibernate</groupId>
                <artifactId>hibernate-entitymanager</artifactId>
                <version>4.3.7.Final</version>
                <scope>provided</scope>
            </dependency>
            <dependency>
                <groupId>javax.inject</groupId>
                <artifactId>javax.inject</artifactId>
                <version>1</version>
                <scope>provided</scope>
            </dependency>
            <dependency>
                <groupId>javax.enterprise</groupId>
                <artifactId>cdi-api</artifactId>
                <version>1.2</version>
                <scope>provided</scope>
            </dependency>

                      <!-- Vaadin -->
            <dependency>
                <groupId>com.vaadin</groupId>
                <artifactId>vaadin-cdi</artifactId>
                <version>1.0.3</version>
            </dependency>
            <dependency>
                <groupId>com.vaadin.addon</groupId>
                <artifactId>jpacontainer</artifactId>
                <version>3.2.0</version>
            </dependency>
            <dependency>
                <groupId>org.vaadin</groupId>
                <artifactId>viritin</artifactId>
                <version>1.35</version>
            </dependency>
                      <!-- Velocity -->
                      <dependency>
                             <groupId>org.apache.velocity</groupId>
                             <artifactId>velocity</artifactId>
                             <version>1.7</version>
                      </dependency>
                      <dependency>
                             <groupId>org.apache.velocity</groupId>
                             <artifactId>velocity-tools</artifactId>
                             <version>2.0</version>
                      </dependency>
        </dependencies>
    </dependencyManagement>

</project>
```