ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΠΟΛΥΜΕΣΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

« ΣΥΣΤΗΜΑ ΓΙΑ ON_LINE ΔΗΜΟΠΡΑΣΙΕΣ »

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ : ΠΑΠΑΔΑΚΗΣ ΝΙΚΟΛΑΟΣ
ΣΠΟΥΔΑΣΤΡΙΕΣ : ΤΥΜΠΛΑΛΕΞΗ ΑΘΑΝΑΣΙΑ (Α.Μ. 1314)
ΦΟΥΝΙΚΑΚΟΥ ΜΑΡΙΑ (Α.Μ. 1313)

Περιεχόμενα

Ευχαριστίες

Αρχικά θέλουμε να ευχαριστήσουμε τις οικογένειες μας που μας στήριξαν σε όλη μας την προσπάθεια για την ολοκλήρωση της διπλωματικής μας εργασίας. Επιπλέον τους: Χαρά, Ιβάν, Μάκη, Σοφία, Μιχάλη, Βαγγέλη, Λίτσα για την ψυχολογική υποστήριξη και κατανόηση όλο αυτό τον καιρό. Τέλος τον επιβλέποντα καθηγητή μας κύριο Νικόλαο Παπαδάκη για την πολύτιμη και ουσιαστική καθοδήγησή του.

Μαρία - Νάνσυ

Τι είναι το Internet

Γενικά

Το Internet είναι ένα πλέγμα από εκατομμύρια διασυνδεδεμένους υπολογιστές που εκτείνεται σχεδόν σε κάθε γωνιά του πλανήτη και παρέχει τις υπηρεσίες του σε εκατομμύρια χρήστες.
Αποτελεί ένα "Παγκόσμιο Ηλεκτρονικό Χωριό", οι "κάτοικοι" του οποίου, ανεξάρτητα από υπηκοότητα, ηλικία, θρήσκευμα και χρώμα, μοιράζονται πληροφορίες και ανταλλάσσουν ελεύθερα απόψεις πέρα από γεωγραφικά και κοινωνικά σύνορα. Σύμφωνα με τις σχετικές εκτιμήσεις, αυτός ο παγκόσμιος ιστός υπολογιστών και χρηστών αριθμεί σήμερα πάνω από δέκα εκατομμύρια υπολογιστές και εκατό εκατομμύρια χρήστες, ενώ επεκτείνεται διαρκώς με εκθετικούς ρυθμούς. Αναμένεται ότι το 2000 το Internet θα εξυπηρετεί περισσότερους από ένα δισεκατομμύριο χρήστες.

Δύο βασικά χαρακτηριστικά του Internet

Ένα βασικό χαρακτηριστικό του Internet είναι ότι μπορεί να συνδέει υπολογιστές διαφορετικού τύπου, δηλ. υπολογιστές που μπορεί να διαφέρουν όσον αφορά την αρχιτεκτονική του υλικού (hardware), το λειτουργικό σύστημα που χρησιμοποιούν και το πρωτόκολλο δικτύωσης που εφαρμόζεται στο τοπικό τους δίκτυο. Ακριβώς εξαιτίας αυτής της ευελιξίας του, εξαπλώθηκε σε ολόκληρο τον πλανήτη κατ'α τη διάρκεια των τελευταίων δεκαετιών.
Ένα άλλο ενδιαφέρον χαρακτηριστικό του Internet είναι ότι είναι αποκεντρωμένο και αυτοδιαχειριζόμενο. Δεν υπάρχει δηλαδή κάποιος κεντρικός οργανισμός που να το διευθύνει και να παίρνει συνολικά αποφάσεις σχετικά με το είδος των πληροφοριών που διακινούνται, τις υπηρεσίες που παρέχονται από τους διάφορους υπολογιστές του ή τη διαχείρισή του. Καθένα από τα μικρότερα δίκτυα που το αποτελούν διατηρεί την αυτονομία του και είναι το ίδιο υπεύθυνο για το είδος των πληροφοριών που διακινεί, τις υπηρεσίες που προσφέρουν οι υπολογιστές του και τη διαχείρισή του.

Τι μας προσφέρει το Internet

Οι άνθρωποι χρησιμοποιούν το Internet βασικά για δύο πράγματα: α) για να αντλήσουν πληροφορίες και β) για να επικοινωνήσουν με άλλους ανθρώπους που είναι κι αυτοί χρήστες του... Μπορούμε να θεωρήσουμε το Internet σαν μια τεράστια αποθήκη πληροφορίας, μια παγκόσμια βιβλιοθήκη. Στους υπολογιστές του, βρίσκονται αποθηκευμένα χιλιάδες Gigabytes πληροφορίας, αρκετά από τα οποία διατίθενται ελεύθερα στους χρήστες του. Έτσι λοιπόν έχουμε τη δυνατότητα να χρησιμοποιούμε απομακρυσμένες βάσεις δεδομένων, να ανακτάμε αρχεία με προγράμματα, εικόνες, κείμενα, κλπ., να έχουμε πρόσβαση σε βιβλιοθήκες, να διαβάζουμε ηλεκτρονικές εφημερίδες και περιοδικά, ακόμη και να παρακολουθούμε ραδιοφωνικά προγράμματα.

Το Internet είναι επίσης ένα μέσο που μας επιτρέπει να ερχόμαστε σε επαφή με άλλους ανθρώπους γρήγορα και εύκολα. Μπορούμε λοιπόν να ανταλλάξουμε ηλεκτρονικά μηνύματα ή να μιλήσουμε "ζωντανά" με έναν φίλο μας που βρίσκεται π.χ. στις ΗΠΑ, στην Κίνα ή σε κάποιο άλλο μέρος του κόσμου, να γνωρίσουμε καινούργιους ανθρώπους, να εγγραφούμε σε λίστες συζητήσεων εάν μας ενδιαφέρουν οι απόψεις των άλλων γύρω από κάποιο θέμα ή ακόμη να παίξουμε μια σειρά από παιχνίδια με πολλούς αντιπάλους ταυτόχρονα που μπορεί να βρίσκονται διασκορπισμένοι σε διάφορα μέρη της γης.
Με το Internet λοιπόν μπορούμε να κάνουμε το γύρο του κόσμου χωρίς να χρειαστεί να μετακινηθούμε από τον υπολογιστή μας.

Η χρήση του Διαδικτύου

Ως αναφορά τη χρήση του διαδικτύου στην Ελλάδα, παρατηρείται σημαντική αύξηση του αριθμού των χρηστών (από 13% το 2001 σε 31% το 2007) ηλικίας 15 έως 65 ετών που κατέχουν προσωπικό Η/Υ. Αντίστοιχα, παρατηρείται αύξηση των ωρών χρήσης του διαδικτύου που φτάνουν κατά μέσω όρο τις 8,6 ανά εβδομάδα.
Η υπηρεσία που χρησιμοποιείται περισσότερο είναι το ηλεκτρονικό ταχυδρομείο (e-mail) αλλά και η ενημέρωση (νέα, καιρός, αθλητικά) αποτελεί από τους κυριότερους λόγους χρήσης του διαδικτύου. Αντίθετα, η αναζήτηση για προϊόντα και υπηρεσίες ακολουθεί πτωτική πορεία από το 2002. Ιδιαίτερα χαμηλή παραμένει η χρήση του διαδικτύου για αγορά προϊόντων και υπηρεσιών. Περίπου 18% των χρηστών προχώρησε σε κάποια αγορά κατά το 2006 ωστόσο το ποσοστό αυτό ανέρχεται μόλις στο 4,5% του γενικού πληθυσμού. Παρόλα αυτά οι αγορές πραγματοποιήθηκαν κυρίως από ελληνικά sites (41%) έναντι των ξένων sites (35%). Οι χρήστες που αγοράζουν μέσω του διαδικτύου συνήθως δεν επισκέπτονται τα αντίστοιχα καταστήματα, ενώ οι κυριότεροι λόγοι αγοράς είναι η προσιτή τιμή και η καλή εξυπηρέτηση. Τέλος αξιοσημείωτο είναι το γεγονός ότι σε ποσοστό πάνω από 60% οι χρήστες θεωρούν ότι ο κίνδυνος διαρροής προσωπικών δεδομένων κατά τη χρήση πιστωτικής κάρτας στις ηλεκτρονικές αγορές είναι μεγάλος ή πολύ μεγάλος.

Ηλεκτρονικό εμπόριο

Εισαγωγή

Το ηλεκτρονικό εμπόριο ή αλλιώς e-commerce είναι η παροχή αγαθών και υπηρεσιών, μέσω internet, συνήθως, έναντι αμοιβής. Συνδεόμαστε με μια ιστοσελίδα, η οποία προσφέρει κάποια συγκεκριμένη υπηρεσία, συμβουλευόμαστε τον κατάλογο, επιλέγουμε το προϊόν, που θέλουμε και συμπληρώνουμε την εντολή αγοράς, διευκρινίζοντας τον τρόπο πληρωμής (π.χ. πιστωτική κάρτα, επιταγή ή εξόφληση τοις μετρητοίς, κατά την παραλαβή του προϊόντος).
Ηλεκτρονικό εμπόριο είναι, επίσης, η παροχή μη υλικών αγαθών, όπως μουσική ή προγράμματα λογισμικού. Είναι, πλέον, γνωστό σε όλους μας ότι αρκεί μια απλή πληκτρολόγηση του αριθμού της πιστωτικής μας κάρτας, ώστε να "κατεβάσουμε" το τραγούδι ή το λογισμικό της επιλογής μας. Μπορούμε να παρακολουθούμε τις μετοχές στη Σοφοκλέους και να διενεργούμε αγοραπωλησίες, αν θέλουμε, μέσω των on-line υπηρεσιών, που πολλές χρηματιστηριακές εταιρίες προσφέρουν. Ακόμη και πλειστηριασμοί μπορούν να γίνουν, μέσω internet, ή αγοραπωλησίες σε χοντρική τιμή, στο λεγόμενο e-marketplace, μια εικονική αγορά, όπου πωλητές και πιθανοί αγοραστές συναλλάσσονται, εκ του μακρόθεν.
Το τελευταίο χρονικό διάστημα, ένας καινούριος όρος, η ψηφιακή τηλεόραση, μπαίνει στην πραγματικότητα του ηλεκτρονικού εμπορίου και ονομάζεται telecommerce. Πόσο είναι πρόσφορο το έδαφος του e-commerce να δεχτεί το τηλεοπτικό εμπόριο θα φανεί σε λίγα χρόνια.

Ιστορικά στοιχεία

Η ιδέα, στην οποία βασίζεται το ηλεκτρονικό εμπόριο, είναι σχετικά πρόσφατη, αν λάβει κανείς υπόψη του ότι το πρώτο "μηχανογραφημένο" πολυκατάστημα στον κόσμο δημιουργήθηκε το 1970. Πρόκειται για το Telemart, στο Σαν Ντιέγκο της Καλιφόρνια. Τότε, δεν υπήρχε Ίντερνετ και οι πελάτες χρησιμοποιούσαν το αναλογικό τηλέφωνο, για να επιλέξουν τα προϊόντα, που επιθυμούσαν να τους αποσταλούν στο σπίτι.
Σήμερα, 30 χρόνια μετά το πρώτο πείραμα, το ηλεκτρονικό εμπόριο είναι πια διαδεδομένο, γνωστό στην πλειοψηφία των χρηστών του Ίντερνετ, έστω κι αν χρησιμοποιείται ελάχιστα.

Κατηγορίες ηλεκτρονικού εμπορίου

Το ηλεκτρονικό εμπόριο κατηγοριοποιείτε βάσει των συναλλασσόμενων μερών του στις εξής κατηγορίες:

Επιχείρηση – Επιχείρηση (Business to Business, B2B)

Οι εταιρίες χρησιμοποιούν το σύστημα B2B για γρηγορότερες συναλλαγές χωρίς σφάλματα, για καλύτερο έλεγχο των αποθεμάτων, αποτελεσματική διαχείριση των αποθεμάτων κ.λπ. Αυτή η κατηγορία έχει κατοχυρωθεί εδώ και αρκετά χρόνια, ειδικά με την χρησιμοποίηση του EDI σε κλειστά ή διεθνή δίκτυα. Το μεγαλύτερο ποσοστό ηλεκτρονικού εμπορίου που διεξάγεται παγκοσμίως είναι τύπου B2B. Αυτό

συμβαίνει διότι οι εφαρμογές B2B περιλαμβάνουν εκατομμύρια συναλλαγών, τεράστιες επενδύσεις, ενώ η ταχύτητα και η ακρίβεια μπορεί να αποτελέσουν σοβαρό ανταγωνιστικό πλεονέκτημα. Η δυνατότητα ηλεκτρονικής σύνδεσης με προμηθευτές και διανομείς καθώς και η πραγματοποίηση ηλεκτρονικών πληρωμών βελτιώνουν ακόμη περισσότερο την αποτελεσματικότητα, περιορίζοντας το ανθρώπινο σφάλμα, αυξάνοντας την ταχύτητα και μειώνοντας το κόστος των συναλλαγών. Το ηλεκτρονικό εμπόριο προσφέρει τη δυνατότητα αυξημένης πληροφόρησης σχετικά με τα προσφερόμενα προϊόντα.

Επιχείρηση – Καταναλωτής (Business to Consumer, B2C)

Πρόκειται για την πιο διαδεδομένη μορφή ηλεκτρονικού εμπορίου. Εξομοιώνεται με την ηλεκτρονική λιανική πώληση. Αυτός ο τύπος εφαρμογών ηλεκτρονικού εμπορίου έχει αναπτυχθεί τα τελευταία χρόνια, κυρίως μετά την ευρεία χρήση του διαδικτύου και τη βελτίωση των παρεχόμενων υπηρεσιών μέσω αυτού. Ο καταναλωτής έχει πρόσβαση σε μια τεράστια ποικιλία προϊόντων σε δικτυακούς κόμβους. Στα ηλεκτρονικά αυτά καταστήματα, βλέπει, επιλέγει, αν επιθυμεί να αγοράσει είδη ένδυσης μπορεί ενίοτε και να τα δοκιμάζει (μέσω ειδικών προγραμμάτων), ανακαλύπτει προϊόντα τα οποία δεν θα μπορούσε να βρει εύκολα στη χώρα του, συγκρίνει τιμές και τέλος αγοράζει. Κι όλα αυτά χωρίς να βγει από το σπίτι του, κερδίζοντας πολύτιμο χρόνο και κόπο.

Επιχείρηση – Δημόσια διοίκηση (Business to Government, B2G)

Καλύπτει όλες τις συναλλαγές μεταξύ επιχειρήσεων και δημόσιων οργανισμών. Αυτές μπορεί να είναι φορολογία, δημόσιες προμήθειες, εισαγωγές - εξαγωγές μέσω τελωνείων, κ.ά.

Καταναλωτής – Δημόσια διοίκηση (Consumer to Government, C2G)

Καλύπτει όλες τις συναλλαγές μεταξύ καταναλωτών και δημόσιων οργανισμών. Αυτές μπορεί να είναι η ολοκλήρωση φορολογικών υποχρεώσεων, προμήθεια των κατάλληλων πιστοποιητικών και βεβαιώσεων κ.ά.). Συγκεκριμένα στην χώρα μας, ο δικτυακός τόπος http://www.e-gov.gr είναι αυτός που φιλοδοξεί να μαζέψει όλες τις κυβερνητικές υπηρεσίες.

Καταναλωτής – Καταναλωτής (Consumer to Consumer, C2C)

Σε αυτή την κατηγορία οι αγοροπωλησίες γίνονται μεταξύ καταναλωτών, όπως οι δημοπρασίες, μικρές αγγελίες κ.ά.

Δημόσια διοίκηση – Δημόσια διοίκηση (Government to Government, G2G)

Αυτή η κατηγορία καλύπτει τις απαιτήσεις των ενδοκυβερνητικών συναλλαγών και ανταλλαγής πληροφοριών.

Ενδοεπιχειρησιακό ηλεκτρονικό εμπόριο

Είναι η περίπτωση συναλλαγών και ανταλλαγής πληροφόρησης μεταξύ των μερών μιας εταιρίας η οποία εκτείνεται σε διαφορετικές πόλεις, χώρες ή και ηπείρους.

Πιστοποίηση και ασφάλεια

Για την ασφάλεια των ηλεκτρονικών συναλλαγών χρησιμοποιούνται ευρέως τα firewalls. Το firewall αποτελεί λογισμικό ή υλικό, που επιτρέπει μόνο στους εξωτερικούς χρήστες που έχουν τα κατάλληλα δικαιώματα, να προσπελάσουν το προστατευμένο δίκτυο. Ένα firewall επιτρέπει στους εσωτερικούς χρήστες να έχουν πλήρη πρόσβαση στις παρεχόμενες υπηρεσίες, ενώ οι εξωτερικοί χρήστες πρέπει να πιστοποιηθούν. Υπάρχουν πολλοί ένας από τους οποίους παρέχει διαφορετικά επίπεδα προστασίας. Ο συνηθέστερος τρόπος χρησιμοποίησης ενός firewall είναι η τοποθέτηση ενός υπολογιστή ή δρομολογητή μεταξύ συγκεκριμένου δικτύου και του διαδικτύου, και η παρακολούθηση όλης της κυκλοφορίας μεταξύ του εξωτερικού και του τοπικού δικτύου.
Η εμπιστευτική πληροφορία που διακινείται στο διαδίκτυο μπορεί να προστατευθεί με κρυπτογράφηση και χρήση μυστικών κωδικών. Η ασφάλεια του ηλεκτρονικού εμπορίου βασίζεται κατεξοχήν στην κρυπτογράφηση, δηλαδή στην κωδικοποίηση του μεταδιδόμενου κειμένου κατά τέτοιο τρόπο ώστε να μπορεί να αποκρυπτογραφηθεί μόνο με τη χρήση του ειδικού κλειδιού αποκρυπτογράφησης. Η κρυπτογράφηση συνοδεύεται πολλές φορές και από την ψηφιακή υπογραφή του αποστολέα, έτσι ώστε ο παραλήπτης να μπορεί να βεβαιωθεί για την ταυτότητα του πρώτου.

Απαιτήσεις για ανάπτυξη ηλεκτρονικού εμπορίου.

Για την ανάπτυξη του ηλεκτρονικού εμπορίου , μια επιχείρηση πρέπει να εκπληρώσει τις παρακάτω φάσεις:

- Φάση 1: Ανάπτυξη ιστοσελίδας και προώθηση προϊόντος :
    - Δημιουργία ιστοσελίδας, ανάπτυξη και φιλοξενία (hosting).
    - Διαφήμιση και πρώτη εικόνα προϊόντων ή υπηρεσιών.
    - Ζήτηση και διακίνηση πληροφοριών μέσω του διαδικτύου.
- Φάση 2 : Κατασκευή λογισμικού και διαχείριση βάσεων δεδομένων :
    - Παραγγελία προϊόντων ή υπηρεσιών μέσω του διαδικτύου.
    - •Ύπαρξη και διαχείριση βάσεων δεδομένων που απαιτούν οι σύγχρονες πολύπλοκες υψηλές τεχνολογίες.
- Φάση 3 : Πληρωμή και επεξεργασία συναλλαγών :
    - •Αναγνώριση πιστότητας πιστωτικής κάρτας και παραγγελία μέσω διαδικτύου.

- Ηλεκτρονική μεταφορά χρημάτων.

- Φάση 4 : Εκπλήρωση και EDI διαμονή αποθεμάτων :
  - Αποστολή προϊόντος και αποθήκευση.
  - Καταχώρηση παραγγελίας και καταστάσεων.
  - Ηλεκτρονική παραγγελία διαμέσου EDI και εξειδικευμένη παρουσία πελατών στο διαδίκτυο.

- Φάση 5 : Υπηρεσίες τηλεφωνικού κέντρου :
  - Υποστήριξη προϊόντων και ειδικά εκπαιδευμένοι αντιπρόσωποι για την εκπλήρωση ειδικών αναγκών των πελατών.
  - Εξερχόμενο και εισερχόμενο direct marketing.

Πλεονεκτήματα ηλεκτρονικού εμπορίου.

Το ηλεκτρονικό εμπόριο προσφέρει στρατηγικά πλεονεκτήματα σε μια επιχείρηση. Στο διαδίκτυο, το μέγεθος της επιχείρησης δεν παίζει σημαντικό ρόλο. Μεγάλες και μικρές επιχειρήσεις έχουν την ίδια πρόσβαση στους πελάτες και μπορούν να δημιουργήσουν παρόμοια παρουσία στο διαδίκτυο. Ακόμη, η έδρα της επιχείρησης δεν παίζει κανένα ρόλο. Όπου και να βρίσκεται η επιχείρηση, οι πελάτες μπορούν να έχουν πρόσβαση στον δικτυακό της τόπο. Ένα ηλεκτρονικό κατάστημα επιτρέπει όχι μόνο τη διεύρυνση της πελατείας, αλλά και την υπέρβαση των περιορισμών στα ωράρια λειτουργίας, γιατί μπορούν να πουληθούν αγαθά όλο το 24ώρο.

Ηλεκτρονικό εμπόριο και οφέλη για τις Επιχειρήσεις

- Ευρεία γεωγραφική κάλυψη : Η επιχείρηση έχει τη δυνατότητα να απευθυνθεί σε πελάτες που βρίσκονται παντού , χωρίς την σύσταση τοπικού υποκαταστήματος.
- Ελαχιστοποίηση της προμηθευτικής αλυσίδας : Ο προμηθευτής μπορεί να απευθυνθεί απευθείας στον πελάτη, χωρίς την ανάμειξη «ενδιάμεσων».
- Μείωση λειτουργικού κόστους : Η μείωση του λειτουργικού κόστους οφείλεται στο γεγονός ότι η επιχείρηση μπορεί να εξυπηρετήσει τους πελάτες με ελάχιστο κόστος. Επίσης, όσο αυξάνεται ο αριθμός των πελατών του ηλεκτρονικού καταστήματος τόσο μειώνεται το συνολικό κόστος εξυπηρέτησης αυτών.

- Συνεχής λειτουργία : Το διαδίκτυο είναι ίσως τα μοναδικό κανάλι εξυπηρέτησης πελατών που επιτρέπει την πραγματοποίηση αγορών οποιαδήποτε στιγμή το 24ώρο.
- Εργαλείο marketing : Η επιχείρηση μπορεί να εκμεταλλευτεί τις δυνατότητες του διαδικτύου για προσφορές , διαχείριση και ενημέρωση πελατών , στατιστικά στοιχεία πρόσβασης και πωλήσεων.

Ένα ηλεκτρονικό κατάστημα φέρνει πιο κοντά τους πελάτες σε αυτό , αφού μπορούν να έχουν κατευθείαν πρόσβαση στις πληροφορίες που παρέχει το κατάστημα. Επιπλέον , προσφέρει τη δυνατότητα ανάλυσης της αγοράς αξιοποιώντας την αγοραστική συμπεριφορά των καταναλωτών.

Ηλεκτρονικό εμπόριο και οφέλη για τους Καταναλωτές

- Μπορούμε να αγοράσουμε προϊόντα ή υπηρεσίες, που επιθυμούμε, όποτε θελήσουμε, 24 ώρες το 24ωρο, 7 ημέρες την εβδομάδα, όπου κι αν βρισκόμαστε.
- Μπορούμε να επισκεφτούμε, εύκολα και γρήγορα, πολλά ηλεκτρονικά καταστήματα και να συγκρίνουμε τις τιμές και τα χαρακτηριστικά των ειδών που μας ενδιαφέρουν, πριν αγοράσουμε αυτό που επιθυμούμε.
- Μπορούμε να ενημερωθούμε, πληρέστερα, για το προϊόν ή την υπηρεσία που αναζητάμε.
- Γλιτώνουμε χρόνο και ταλαιπωρία, μειώνοντας τις μετακινήσεις μας.
- Πολλά ηλεκτρονικά καταστήματα κάνουν ειδικές εκπτώσεις, στους πελάτες τους.
- Η νομοθεσία για την προστασία του Καταναλωτή, όταν πρόκειται για αγορές από απόσταση, όπως είναι οι ηλεκτρονικές αγορές διασφαλίζει τα δικαιώματά τους (π.χ. το δικαίωμα υπαναχώρησης κ.λπ.)

Οι « νόμοι » του ηλεκτρονικού εμπορίου.

Ο καινούριος εμπορικός « χώρος » διέπεται από νόμους οι οποίοι δεν ισχύουν στην « παλαιά οικονομία » . Αυτοί είναι :

- Τα υλικά δεν έχουν μεγάλη σημασία : Η αξία στην «νέα οικονομία» βρίσκεται στην πληροφορία και την πληροφόρηση, τις υπηρεσίες, τη γνώση, τους ανθρώπους και τις στρατηγικές συμμαχίες.
- Ο χώρος συρρικνώνεται : Οι πελάτες βρίσκονται σε όλο τον κόσμο, όπως και οι ανταγωνιστές.
- Ο χρόνος συρρικνώνεται : Οι εταιρίες μειώνουν τον χρόνο αναζήτησης, σύγκρισης, αγοράς, εκτέλεση της παραγγελίας για τους πελάτες τους. Αυτό έχει σαν αποτέλεσμα, την όξυνση του ανταγωνισμού, διότι οι πελάτες ζητούν άμεση ανταπόκριση στα ερωτήματά τους, υπηρεσίες και άμεση ικανοποίηση γενικότερα.
- Οι άνθρωποι έχουν σημασία : Τα στελέχη και οι πελάτες της εταιρίας έχουν την μεγαλύτερη αξία για αυτήν. Οι πελάτες δίνουν ιδέες για νέα ή βελτιωμένα προϊόντα και μπορούν εύκολα να γίνουν όχι μόνο αγοραστές, αλλά ακόμη και πωλητές προϊόντων. Τα στελέχη συλλέγουν και επεξεργάζονται τα μηνύματα της αγοράς και κατευθύνουν την εταιρία με στόχο το ανταγωνιστικό πλεονέκτημα.
- Οι πελάτες γίνονται και πωλητές : Το διαδίκτυο διευκολύνει τη δυνατότητα «μετάδοσης» των καλών πληροφοριών για ένα αξιόλογο προϊόν. Με κατάλληλες πρακτικές marketing, οι πελάτες μπορούν να γίνουν και πωλητές των προϊόντων μιας εταιρίας.
- Το μερίδιο της αγοράς ανεβάζει την αξία της εταιρίας : Όσο μεγαλύτερο μερίδιο αγοράς έχει μια εταιρία, τόσο αυξάνει την αξιοπιστία της και όσο αυξάνεται η αξιοπιστία της, τόσο περισσότερους πελάτες προσελκύει, αυξάνοντας το μερίδιό της.
- Η πληροφορία ανεβάζει την αξία της εταιρίας : Οι πληροφορίες που μπορούν να συλλέξουν οι εταιρίες για τις προτιμήσεις και τα ενδιαφέροντα των πελατών τους είναι πολύτιμες, γιατί τους επιτρέπουν να βελτιώσουν τα προϊόντα και τις υπηρεσίες τους, να δημιουργήσουν νέα προϊόντα και να τα προωθήσουν την κατάλληλη στιγμή στους κατάλληλους υποψήφιους αγοραστές.
- Συσσωρεύετε η δύναμη των αγοραστών και αυξάνονται οι ευκαιρίες των πωλητών : Οι πωλητές καθορίζουν τις τιμές των προϊόντων σύμφωνα με τις ανάγκες και τις προτιμήσεις των αγοραστών. Το πιο δημοφιλές εργαλείο των πωλητών είναι οι ηλεκτρονικές δημοπρασίες.
- Μαζική εξατομίκευση : Ο αντιφατικός αυτός όρος περιγράφει την προσπάθεια των εταιριών να προσελκύσουν τις μάζες δημιουργώντας προϊόντα και υπηρεσίες για τις ανάγκες του κάθε πελάτη χωριστά.
- Οποιοδήποτε προϊόν διατίθεται οπουδήποτε και οποτεδήποτε : Με κατάλληλες πρακτικές marketing και συνεργασία μεταξύ εταιριών μπορεί οποιοδήποτε προϊόν να διατεθεί οπουδήποτε και οποτεδήποτε.

Ιστοσελίδες

Ορισμός

Η ιστοσελίδα είναι ένα αρχείο που περιέχει πληροφορίες που είναι προορισμένες για δημοσίευση στον Παγκόσμιο Ιστό (www). Μια ιστοσελίδα είναι προσβάσιμη από ένα Φυλλομετρητή (web browser). Οι πληροφορίες της είναι συνήθως γραμμένες σε HTML ή XHTML.
Μια ιστοσελίδα μπορεί να περιέχει ένα σύνολο πληροφοριών όπως κείμενα , γραφικά , φωτογραφίες ,video , ήχους , χρώματα ή ακόμα και διάφορα αρχεία . Οι επισκέπτες της ιστοσελίδας ονομάζονται web clients.

Ορισμός ιστοτόπου

Ο ιστότοπος είναι το σύνολο των ιστοσελίδων που είναι συνδεμένες μεταξύ τους με υπερσυνδέσεις (hyperlinks) και περιγράφουν μία δραστηριότητα.
Ένας ιστότοπος μπορεί να είναι επαγγελματικός ,προσωπικός ή να ανήκει σε έναν οργανισμό ή μια ομάδα. Το σύνολο των ιστοτόπων που υπάρχουν απαρτίζουν στην ουσία τον Παγκόσμιο Ιστό (www).

Web browsers

Οι web browsers είναι τα προγράμματα τα οποία μας επιτρέπουν να βλέπουμε σελίδες στο internet. Κάθε ένας web browser έχει τις δικές του δυνατότητες , επεκτάσεις, εχθρούς και φίλους. Αυτή τη στιγμή υπάρχει μια πλειάδα από web browsers που ικανοποιούν τα γούστα και του πιο απαιτητικού χρήστη.

Οι πιο δημοφιλείς web browsers είναι :

- Internet explorer 7
- Internet explorer 8
- Mozilla Firefox
- Opera
- Safari
- Google chrome
- Netscape navigator (σταμάτησε πλέον η στήριξή του)

Το κύριο χαρακτηριστικό όλων είναι ότι διανέμονται δωρεάν οπότε ο κάθε χρήστης έπειτα από δοκιμή μπορεί να βρει ποιος απ'όλους έχει όλα τα στοιχεία ώστε η πλοήγησή του στο internet να γίνεται πιο ευχάριστη. Κάποιος που ασχολείται με την κατασκευή ιστοσελίδων προτείνεται να τους έχει όλους ώστε να ελέγχει την εμφάνιση των ιστοσελίδων του από διαφορετικά προγράμματα.

Domain name

Τα domain names είναι στην ουσία η ταυτότητα των ιστοτόπων. Όταν θέλουμε να καλέσουμε μια ιστοσελίδα μέσα από ένα web browser θα πρέπει να γράψουμε στη γραμμή διευθύνσεων το όνομά της π.χ. www.epp.teiher.gr.
Γράφοντας αυτό το όνομα στην ουσία ο web browser ψάχνει να βρει σε ποιο web server είναι αποθηκευμένη η συγκεκριμένη σελίδα . Εάν το domain name που πληκτρολογήσαμε είναι υπαρκτό το αποτέλεσμα θα είναι η ιστοσελίδα να εμφανιστεί στον web browser.

Διαχωρισμός ιστοσελίδων

Οι ιστοσελίδες μπορούν να χωριστούν σε δύο κατηγορίες: στατικές και δυναμικές.
Στατικές σελίδες είναι σταθερά ηλεκτρονικά έγγραφα που περιέχουν κείμενα,φωτογραφίες,συνδέσμους κ.α.Είναι κατάλληλες για την δημιουργία "στατικών παρουσιάσεων" όπου δεν χρειάζεται να τροποποιείται το περιεχόμενο τους. Το περιεχόμενο τροποποιείται με απευθείας επεμβάσεις στην ιστοσελίδα, με χρήση authoring tools.
Δυναμικές είναι οι ιστοσελίδες των οποίων το περιεχόμενο παράγεται δυναμικά, δηλαδή δεν προϋπάρχει, αλλά δημιουργείται τη στιγμή που κάποιος επισκέπτεται την ιστοσελίδα.
Για να δημιουργηθεί μια δυναμική ιστοσελίδα, ο server επεξεργάζεται κάποια στοιχεία και παράγει το περιεχόμενο της σελίδας. Για να γίνει αυτό χρησιμοποιείται μια γλώσσα προγραμματισμού, όπως PHP, ASP κλπ. Συνήθως χρησιμοποιούνται πληροφορίες από μια βάση δεδομένων.

Διαφορά στατικών - δυναμικών ιστοσελίδων

Η μεγάλη διαφορά μεταξύ των στατικών και δυναμικών ιστοσελίδων από την πλευρά του πελάτη είναι οτι οι δεύτερες έχουν μεγαλύτερο κόστος ανάπτυξης στην αρχή, αλλά καθιστούν πολύ πιο εύκολη, γρήγορη και φθηνή την προσθήκη υλικού και την ανανέωση μιας ιστοσελίδας από τη στιγμή που αυτή θα δημιουργηθεί. Αυτό έχει ως αποτέλεσμα οι δυναμικές ιστοσελίδες να ενδείκνυνται για την κατασκευή μέτριων εως μεγάλων σε όγκο ιστοσελίδων, ενώ για πολύ σύνθετες ιστοσελίδες αποτελεί τη μόνη πραγματοποιήσιμη λύση.

Εφαρμογή δυναμικών ιστοσελίδων

Γενικά είναι λίγες οι ιστοσελίδες που συμφέρει να κατασκευαστούν στατικά. Η προσθήκη νέου υλικού, κάποιες αλλαγές που μπορεί να χρειαστεί να γίνουν, μελλοντικές διαφοροποιήσεις στον τρόπο διάταξης του περιεχομένου, βελτιστοποίηση για τις μηχανές αναζήτησης, όλα αυτά γίνονται πολύ πιο εύκολα και φθηνά, αν η ιστοσελίδα σας είναι δυναμική. Έτσι, κάτι που είναι φαινομενικά πιο ακριβό, μπορεί τελικά να αποτελεί την πλέον συμφέρουσα λύση για τις δικές σας ανάγκες. Η εταιρεία με την οποία θα συνεργαστείτε για την κατασκευή της ιστοσελίδας σας θα πρέπει να σας εξηγήσει με ποιόν από τους δύο τρόπους θα κατασκευάσει την ιστοσελίδα σας και γιατί.

Δυναμικές ιστοσελίδες - DHTML

Μια άλλη περίπτωση, όπου καλούμε μια ιστοσελίδα δυναμική, είναι όταν ο κώδικας που βρίσκεται στη σελίδα (HTML) και αλλάζει το περιεχόμενο της ιστοσελίδας, χωρίς να χρειαστεί να ξαναφορτώσει καινούρια σελίδα. Για να γίνει αυτό είναι απαραίτητη η χρήση JavaScript. Στην περίπτωση αυτή λέμε ότι γίνεται χρήση DHTML (Dynamic HTML).

Δυναμικές ιστοσελίδες με βάσεις δεδομένων

Οι δυναμικές ιστοσελίδες, μπορεί στην εμφάνιση, σε πολλές περιπτώσεις, να μην έχουν μεγάλη διαφορά με τις στατικές, όμως οι δυνατότητές είναι πολύ περισσότερες, από πολλές πλευρές, καθώς στην περίπτωση αυτή ουσιαστικά πρόκειται για μία εφαρμογή (πρόγραμμα), και όχι ένα απλό ηλεκτρονικό έγγραφο.

Συνήθως, οι δυναμικές ιστοσελίδες, χρησιμοποιούν κάποια βάση δεδομένων (database), όπου αποθηκεύουν πληροφορίες και απ' όπου αντλούν το περιεχόμενό τους, ανάλογα με το τι ζητάει ο χρήστης/επισκέπτης σε κάθε του "κλικ". Η χρήση των βάσεων δεδομένων, είναι αυτή που επιτρέπει την εύκολη προσθαφαίρεση περιεχομένου στις δυναμικές ιστοσελίδες, καθώς δεν απαιτείται να επεξεργάζεται κανείς κάθε φορά την ίδια την ιστοσελίδα, αλλά απλά να διαχειρίζεται έμμεσα το περιεχόμενο στην βάση δεδομένων και οι υπόλοιπες διαδικασίες γίνονται αυτοματοποιημένα από τον "μηχανισμό" της ιστοσελίδας.

HTML

Τα αρχικά HTML προέρχονται από τις λέξεις HyperText Markup Language. Η html δεν είναι μια γλώσσα προγραμματισμού. Είναι μια περιγραφική γλώσσα (markup language), δηλαδή ένας ειδικός τρόπος γραφής κειμένου. Ο καθένας μπορεί να δημιουργήσει ένα αρχείο HTML χρησιμοποιώντας απλώς εναν επεξεργαστή κειμένου. Αποτελεί υποσύνολο της γλώσσας SGML (Standard Generalized Markup Language) που επινοήθηκε από την IBM προκειμένου να λυθεί το πρόβλημα της μη τυποποιημένης εμφάνισης κειμένων στα διάφορα υπολογιστικά συστήματα. Ο browser αναγνωρίζει αυτόν τον τρόπο γραφής και εκτελεί τις εντολές που περιέχονται σε αυτόν. Αξίζει να σημειωθεί ότι η html είναι η πρώτη και πιο διαδεδομένη γλώσσα περιγραφής της δομής μιας ιστοσελίδας. Η html χρησιμοποιεί τις ειδικές ετικέτες (τα tags) να δώσει τις απαραίτητες οδηγίες στον browser. Τα tags είναι εντολές που συνήθως ορίζουν την αρχή ή το το τέλος μιας λειτουργίας. Τα tags βρίσκονται πάντα μεταξύ των συμβόλων < και >. Π.χ. <BODY> Οι οδηγίες είναι case insensitive, δεν επηρεάζονται από το αν έχουν γραφτεί με πεζά (μικρά) ή κεφαλαία. Ένα αρχείο HTML πρέπει να έχει κατάληξη htm ή html.

HTML Standards

Για να μπορούν οι browser να ερμηνεύουν σχεδόν απόλυτα σωστά την html έχουν θεσπιστεί κάποιοι κανόνες. Αυτοί οι κανόνες είναι γνωστοί ως προδιαγραφές. Επομένως σχεδόν κάθε είδος υπολογιστή μπορεί να δείξει το ίδιο καλά μια ιστοσελίδα. Οι πρώτες προδιαγραφές ήταν η html 2.0. Πρόβλημα προέκυψε όταν η Microsoft και η Netscape πρόσθεσαν στην html τέτοιες δυνατότητες που στην αρχή τουλάχιστον ήταν συμβατές μόνο με συγκεκριμένους browser. Ακόμη και σήμερα υπάρχουν διαφορές στην απεικόνιση κάποιας σελίδας από διαφορετικούς browsers. Ιδιαίτερο είναι το πρόβλημα όταν η ιστοσελίδα, εκτός απο "καθαρή" HTML περιλαμβάνει και εφαρμογές Javascript

Η HTML σήμερα

Σήμερα πολλοί είναι εκείνοι που δημιουργούν μια ιστοσελίδα σε κάποιο πρόγραμμα που επιτρέπει την δημιουργία χωρίς την συγγραφή κώδικα. Η κοινή άποψη πάνω στο θέμα όμως είναι ότι κάτι τέτοιο είναι αρνητικό επειδή ο δημιουργός δεν έχει τον απόλυτο έλεγχο του κώδικα με αποτέλεσμα πολλές φόρες να υπάρχει οπτικό χάος στην προσπάθεια των browser να εμφανίσουν την ιστοσελίδα. Για το σκοπό αυτό έχει δημιουργηθεί ειδικό λογισμικό, που επιτρέπει το "στήσιμο" της σελίδας οπτικά, χωρίς τη συγγραφή κώδικα, δίνει όμως τη δυνατότητα παρέμβασης ΚΑΙ στον κώδικα. Χαρακτηριστικό παράδειγμα το λογισμικό Dreamweaver της Adobe και το FrontPage της Microsoft.

SQL

Η γλώσσα SQL είναι τυποποιημένη γλώσσα προγραμματισμού για την πρόσβαση και διαχείριση δεδομένων σε βάσεις δεδομένων.
SQL είναι τα αρχικά της αγγλικής ονομασίας Structured Query Language η οποία στην ελληνική ορολογία μεταφράζεται ώς δομημένη γλώσσα αναζήτησης. Συνοπτικά με την SQL μπορούμε να έχουμε πρόσβαση σε μια βάση δεδομένων, να εκτελούμε αναζητήσεις με κριτήρια, να εισάγουμε νεα δεδομένα, να διαγράψουμε υπάρχοντα δεδομένα να κάνουμε αλλαγές σε υπάρχοντα δεδομένα. Αξίζει να σημειωθει ότι η SQL είναι τυποποιημένη από τον Αμερικανικό Ινστιτούτο Τυποποιήσεων γνωστή και ως ANSI (American National Standards Institute) για την πρόσβαση και διαχείριση βάσεων δεδομένων. Εντόλες SQL χρησιμοποιούνται για την εξαγωγή και ανανέωση δεδομένων απο μια βάση δεδομένων.

MySQL

Η MySQL είναι ένα σχεσιακό σύστημα διαχείρισης βάσεων δεδομένων (RDBMS), η οποία διαθέτει περισσότερες από 6 εκατομμύρια εγκαταστάσεις. Το πρόγραμμα λειτουργεί ως διακομιστής που παρέχουν πολλαπλούς χρήστες πρόσβαση σε μια σειρά από βάσεις δεδομένων.

- Με τη χρήση της MySQL είναι εύκολη η πρόσβαση σ’ αυτές τις πληροφορίες χρησιμοποιώντας μια γλώσσα συγγραφής σεναρίων στην πλευρά του διακομιστή (server-side scripting languages), όπως είναι η PHP.
  Είναι ένα πολύ γρήγορο και δυνατό σύστημα διαχείρισης βάσεων δεδομένων

- Ο MySQL διακομιστής ελέγχει την πρόσβαση στα δεδομένα, για να μπορούν να δουλεύουν πολλοί χρήστες ταυτόχρονα
- Μπορούν να υπάρχουν ταυτόχρονα περισσότερες από μια συνδέσεις με τη βάση χωρίς να υπάρχουν πολλαπλά αντίγραφά της, όπως συμβαίνει με άλλα συστήματα βάσεων δεδομένων
- Η απόδοσή της είναι καλύτερη σε μεγαλύτερο όγκο βάσεων δεδομένων
- Είναι πιο κατάλληλη για χρήση στο Internet
- Παρέχει ευκολίες στο backup
- Είναι ιδιαίτερα βελτιωμένη σε ταχύτητα για την ανάκτηση δεδομένων
- Είναι συμβατή και μεταφέρσιμη σε διάφορες πλατφόρμες και για διάφορα εργαλεία ανάπτυξης
- Είναι οικονομική
- Η MySQL είναι λογισμικό ανοιχτού κώδικα.

Η MySQL υποστηρίζει ένα υποσύνολο του Ansi SQL και περιλαμβάνει πολλές επεκτάσεις.

Μερικά χαρακτηριστικά γνωρίσματα :

- Πολυνηματώδης.
- Όλη η κυκλοφορία κωδικού πρόσβασης κρυπτογραφείται.
- Όλες οι στήλες περιλαμβάνουν προκαθορισμένες τιμές.
- Έλεγχος και τροποποίηση πινάκων.
- Ψευδώνυμα πινάκων και στηλών σύμφωνα με τα πρότυπα SQL92 .
- Μη διαρροή μνήμης.
- Όλες οι συνενώσεις (joins) γίνονται σε ένα πέρασμα.
- Εγγραφές σταθερού και μεταβλητού μήκους.

Διεπαφές : SQL, ODBC, C, Perl, JAVA, C++, Python, command line.

Μέθοδοι πρόσβασης : B-tree στο δίσκο, hash tables στη μνήμη.

Πολυχρηστικό : Ναι.

Δοσοληψίες : Ναι, υποστηρίζει και foreign key constraints.

Κατανεμημένο : Όχι, υπάρχει η δυνατότητα για mirroring.

Γλώσσα Ερωτημάτων : SQL.

Όρια : Πάνω από 32 indexes / table. Κάθε index αποτελείται από 1 έως 16 στήλες. Το μέγιστο πλάτος του index είναι 500 bytes.

Ανθεκτικότητα : Ο κώδικας του B-tree είναι εξαιρετικά σταθερός, εφικτή η 24-ωρη λειτουργία.

Υποστηριζόμενες Πλατφόρμες : BSDOS, SunOS, Solaris, Linux, IRIX, AIX, OSF1, BSD/OS, FreeBSD.

Τεχνολογίες που χρησιμοποιήσαμε

Γενικά

Στην πτυχιακή μας ασχοληθήκαμε με την ανάπτυξη ενός συστήματος on line δημοπρασιών. Δημιουργήσαμε ένα site,στο οποίο μπορεί ένας χρήστης να δει τα προϊόντα που δημοπρατούνται και να αγοράσει κάποιο από αυτά αν το επιθυμεί, καθώς και να ενημερωθεί για τις δημοπρασίες που θα ανοίξουν προσεχώς. Διαχειριστής αυτής της σελίδας είναι ο Admin που έχει τη δυνατότητα να προσθέτει καινούργια προϊόντα ανάλογα με την εταιρία και την κατηγορία στην οποία ανήκουν.Επίσης μπορεί να βλέπει στατιστικά στοιχεία που αφορούν τις δημοπρασίες. Για τη δημιουργία του site χρησιμοποιήσαμε τις τεχνολογίες JSP/Servlets. Η αποθήκευση των δεδομένων γίνεται σε μία σειριακή βάση δεδομένων MySQL ενώ οι πληροφορίες παρουσιάζονται στον χρήστη σε σελίδες HTML. Ο τρόπος εμφάνισης των σελίδων ρυθμίζεται χρησιμοποιώντας αρχεία κανόνων CSS.

JSPs (JavaServer Pages)

Ορισμός

Οι JSP (JavaServer Pages) είναι μια server-side τεχνολογία που παρουσιάστηκε από την Sun Microsystems Corp., και παρέχει ένα γρήγορο και απλό τρόπο για την παραγωγή δυναμικού περιεχομένου μέσα από HTML σελίδες. Χρησιμοποιεί XML tags μαζί με Java scriptlets για να ενσωματώσει και να διαχωρίσει έτσι τη λογική από τον αισθητικό σχεδιασμό και εμφάνιση. Όταν καλείται μια JSP σελίδα, μετατρέπεται δυναμικά σε ένα Servlet το οποίο επεξεργάζεται από τον server για να παραχθεί η τελική HTML/XML σελίδα που θα δεί ο client. Όταν η JSP τεχνολογία χρησιμοποιείται σε συνδυασμό με την τεχνολογία JavaBeans, είναι δυνατό να παραχθούν ποικιλόμορφες και επιδεχόμενες μεγάλης κλιμάκωσης εφαρμογές.

Οι JSPs είναι μια εφαρμογή στην πλευρά του server (server-side application), που σημαίνει ότι δέχονται μια αίτηση (request) και παράγουν μια απόκριση ή απάντηση (response). Σε γενικές γραμμές, οι αιτήσεις γίνονται από έναν Web client και η απόκριση είναι ένα παραγόμενο HTML έγγραφο (ιστοσελίδα) το οποίο στέλνεται πίσω στον Web client. Επειδή οι JSPs είναι μια εφαρμογή στην πλευρά του server, έχουν πρόσβαση σε πηγές (resources) στον server, όπως είναι τα Servlets, JavaBeans, EJBs, αλλά και σε βάσεις δεδομένων.
Υπάρχουν πολλά πλεονεκτήματα από τη χρήση των JavaServer Pages. Επειδή οι JSPs χρησιμοποιούν τη γλώσσα προγραμματισμού Java, ακολουθούν την πολιτική write-once, run-anywhere. Αυτό σημαίνει ότι μια JSP μπορεί να εκτελεσθεί σ' έναν οποιονδήποτε application server ο οποίος υποστηρίζει τις JSPs χωρίς να χρειασθεί κάποια τροποποίηση στον κώδικα.
Οι JSPs μπορούν να γραφούν σ' έναν text editor με την επέκταση (extension) .jsp. Ένα πρόγραμμα το οποίο υποστηρίζει το γράψιμο (δημιουργία) των JSPs είναι το DreamWeaver. Ένα άλλο πλεονέκτημα των JSPs είναι η χρήση των tag libraries. Οι JSPs χρησιμοποιούν τα tags, τα οποία είναι παρόμοια μ' αυτά της HTML και της XML, για να εισάγουν δυναμικό περιεχόμενο (dynamic content). Τα tag libraries

ορίζουν επιπλέον tags τα οποία μπορούν να χρησιμοποιηθούν για να αντικαταστήσουν τμήματα κώδικα.
Ένα άλλο σημαντικό πλεονέκτημα των JSPs είναι ο διαχωρισμός των ρόλων. Οι προδιαγραφές των JSPs επιτρέπουν να μοιραστεί το φορτίο σε δύο κατηγορίες : στο γραφικό περιεχόμενο της σελίδας και στο δυναμικό περιεχόμενο της σελίδας. Αυτό σημαίνει στην πράξη ότι η ομάδα που δεν γνωρίζει τη γλώσσα προγραμματισμού Java μπορεί να δημιουργήσει το γραφικό περιεχόμενο της σελίδας και ένας προγραμματιστής της Java να δημιουργήσει το δυναμικό περιεχόμενο της σελίδας.
Όταν γράφουμε ένα JSP, είναι ευκολότερο να γράψουμε πρώτα τον HTML κώδικα και μετά να εισάγουμε τον κώδικα της Java για να δημιουργήσουμε το δυναμικό περιεχόμενο. Ακολουθεί ένα απλό παράδειγμα JSP το οποίο δείχνει πώς μπορεί να εισαχθεί κώδικας της Java σ' ένα JSP. Αυτό που κάνει αυτό το JSP είναι ότι εισάγει την τρέχουσα ημερομηνία και ώρα του συστήματος του Web server σε μια HTML σελίδα.
Η Java γίνεται σταδιακά όλο και πιο δημοφιλής για τη δημιουργία ποικίλων και επιδεχόμενων κλιμάκωσης, ανεξαρτήτων από πλατφόρμα λύσεων. Μια από τις πιο έντονα αυξανόμενες ανάγκες της Java βρίσκεται στον τομέα του ASP (Application Service Provider). Η Java προσφέρεται σαν η τέλεια λύση σε τέτοιου είδους αγορές έχοντας τα ακόλουθα προτερήματα:
Ανεξαρτησία πλατφόρμας
Αφοσίωση στα υπάρχοντα πρότυπα
Μεγάλη δυνατότητα κλιμάκωσης
Συνέπεια στην απόδοση
Κατανεμημένες, πολυνηματικές, ασφαλείς κτλ. εφαρμογές
Μια πολύ σημαντική και αναπτυσσόμενη τεχνολογία που αναδύθηκε μέσα από την Java είναι η JSP (JavaServer Pages).

Servlets

Εισαγωγή

Οι ιστοσελίδες που περιέχουν κώδικα JSP (JavaServer Pages), δηλ. ανάμιξη κώδικα HTML με κώδικα Java, μετατρέπονται (μεταφράζονται ή μεταγλωττίζονται) σε Servlets πριν εκτελεσθούν στον server. Η κατανόηση αυτής της μετάφρασης (απόδοσης) σε κώδικα Servlet θα μας βοηθήσει να καταλάβουμε καλύτερα τις δραστηριότητες των JSPs στο παρασκήνιο. Ένα Servlet, στη γενική του μορφή, είναι μια τάξη (class) της Java που υλοποιεί (implements) το interface Servlet και δέχεται αιτήσεις (requests) και παράγει (δημιουργεί) αποκρίσεις (responses). Οι αιτήσεις μπορεί να προέρχονται από τάξεις της Java, από Web clients ή και από άλλα Servlets.
Όταν υλοποιούμε ένα interface λέμε ότι η τάξη μάς παρέχει υλοποιήσεις για τις μεθόδους που είναι δηλωμένες στο interface. Συνεπώς, όταν υλοποιούμε το interface Servlet δηλώνουμε ότι ο κώδικάς μας θα παρέχει υλοποιήσεις για τις μεθόδους που βρίσκονται στο interface Servlet. Θα ασχοληθούμε μ' ένα μόνο συγκεκριμένο είδος Servlet, το HttpServlet, το οποίο δέχεται HTTP requests και παράγει HTTP responses. Όταν γράφουμε το δικό μας HttpServlet, δεν υλοποιούμε το interface Servlet απευθείας, αλλά επεκτείνουμε (extend) την τάξη HttpServlet.

Μετατροπή των JSPs σε Servlets

Αφού τα JSPs (ιστοσελίδες με JSP κώδικα) μετατρέπονται σε Servlets, τότε θα μπορεί να ρωτήσει κάποιος γιατί να πρέπει να μάθουμε και να χρησιμοποιούμε και τα δύο; Ο βασικότερος λόγος για το γιατί πρέπει να χρησιμοποιούμε τα JSPs είναι ότι μπορούμε να γράψουμε ευκολότερα τον κώδικα, ενώ αν ασχοληθούμε με το γράψιμο του κώδικα σε Servlet, θα πρέπει να μάθουμε να δημιουργούμε τάξεις (classes) και κληρονομικότητα (inheritance) σε γλώσσα Java, κάτι όχι ιδιαίτερα επιθυμητό από τους περισσότερους.
Θα πρέπει επίσης να μάθουμε πώς είναι ένας κώδικας γραμμένος σε Servlet, για να γνωρίζουμε τι βρίσκεται πίσω από έναν κώδικα που είναι γραμμένος σε JSP.
Όσον αφορά τώρα τη διαδικασία μετατροπής (μετάφρασης), το JSP μεταφράζεται (translated) στον αντίστοιχο κώδικα Servlet, το οποίο είναι ένα αρχείο με επέκταση .java. Αυτό μετά μεταγλωττίζεται (compiled) στο αντίστοιχο bytecode αρχείο με επέκταση .class. Το αρχείο .class εκτελείται, δημιουργείται η έξοδος σε μορφή HTML εγγράφου και στέλνεται πίσω στον client. Μετά από την πρώτη κλήση ενός JSP, τα επόμενα requests δεν θα ακολουθήσουν όλη τη φάση της μετάφρασης αλλά θα πάνε κατευθείαν στο ήδη μεταγλωττισμένο αρχείο .class.

Η Βασική Δομή ενός HttpServlet

Υπάρχουν πολλές μέθοδοι που υπερκαλύπτουμε (override) όταν γράφουμε ένα HttpServlet, όπως :
void init(ServletConfig sc) throws ServletException;
void Service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException;
void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException;
void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException;
Η μέθοδος init() καλείται για να αποδώσει αρχικές τιμές σε ορισμένες παραμέτρους.
Η μέθοδος Service() χρησιμοποιείται για να ελέγξουμε τις καταχωρήσεις του χρήστη όταν δεν μας ενδιαφέρει αν η αίτησή (request) του προήλθε από μια μέθοδο GET ή μια μέθοδο POST. Αν, όμως, έχει σημασία το αν η HTML φόρμα στάλθηκε με μια μέθοδο GET ή μια μέθοδο POST, τότε θα πρέπει να υπερκαλύψουμε μια από τις μεθόδους doGet() ή doPost().
Μια HTML φόρμα διαθέτει την ιδιότητα (attribute) METHOD η οποία ορίζει τον τρόπο με τον οποίο θα σταλούν τα δεδομένα στον server. Η μέθοδος GET προσαρτά τα δεδομένα στο URL και τα στέλνει στον server μ' αυτόν τον τρόπο, ενώ η μέθοδος POST συσκευάζει τα δεδομένα σ' ένα πακέτο και στέλνει το πακέτο στον server.

Η δομή της βάσης δεδομένων

Σχεδίαση

Το πρώτο βήμα για την ανάπτυξη της εφαρμογής μας είναι η σχεδίαση της βάσης δεδομένων. Στηριζόμενοι στις προδιαγραφές του συστήματος φτιάξαμε τους πίνακες με τέτοιο τρόπο ώστε να αποθηκεύουμε όλη την απαιτούμενη πληροφορία. Παρακάτω εξηγούμε τι είναι ο κάθε πίνακας και τι ρόλο παίζουν τα πεδία που υπάρχουν σε αυτούς.

Πίνακας users

	Field	Type	Collation	Attributes	Null	Default	Extra
☐	id	int(11)			No		auto_increment
☐	username	varchar(50)	utf8_general_ci		No		
☐	pass	varchar(10)	utf8_general_ci		No		
☐	email	varchar(50)	utf8_general_ci		No		

Εικόνα 1 Πίνακας users

Ο πίνακας users είναι ο πίνακας στον οποίο αποθηκεύουμε τις πληροφορίες για τους χρήστες του συστήματος.
Κάθε εγγραφή σε αυτόν τον πίνακα αναπαριστά έναν χρήστη του συστήματος frontend.
Περιέχει τα παρακάτω πεδία:

- id
Πρόκειται για τον μοναδικό ακέραιο που είναι το αναγνωριστικό πεδίο κάθε εγγραφής. Είναι ένας ακέραιος(int) ο οποίος έχει την ιδιότητα auto_increment της MySQL. Πρακτικά αυτό εξασφαλίζει ότι οι τιμές της στήλης θα είναι μοναδικές, καθώς δεν τις εισάγουμε χειροκίνητα, αλλά ανήκουν σε μια ακολουθία συνεχώς αυξανόμενων ακεραίων που ανατίθενται αυτόματα σε κάθε εισαγωγή.

- username
Σε αυτό το πεδίο κρατάμε το όνομα κάθε χρήστη, το οποίο και χρησιμοποιεί κατά την διαδικασία εισαγωγής του στο σύστημα. Είναι ένα πεδίο κειμένου, ορισμένο σαν varchar.

- pass
Σε αυτό το πεδίο κρατάμε τον κωδικό κάθε χρήστη, τον οποίο και χρησιμοποιεί κατά την διαδικασία εισαγωγής του στο σύστημα. Είναι ένα πεδίο κειμένου, ορισμένο σαν varchar.

- email
Σε αυτό το πεδίο κρατάμε τον account κάθε χρήστη, το οποίο και χρησιμοποιεί κατά την διαδικασία εισαγωγής του στο σύστημα. Είναι ένα πεδίο κειμένου, ορισμένο σαν varchar.

Δημιουργία κώδικα

```
CREATE TABLE `users` (
  `id` int(11) NOT NULL auto_increment,
  `username` varchar(50) NOT NULL,
  `pass` varchar(10) NOT NULL,
  `email` varchar(50) NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=utf8 AUTO_INCREMENT=25 ;
```

Πίνακας bid

	Field	Type	Collation	Attributes	Null	Default	Extra
☐	auction_id	int(11)			No		
☐	user_id	int(11)			No		
☐	bid_time	timestamp			No	CURRENT_TIMESTAMP	
☐	amount	double			No		

Εικόνα 2 Πίνακας bid

Στον πίνακα bid αποθηκεύουμε όλες τις προσφορές που κάνουν οι χρήστες για κάθε δημοπορασία. Καθώς δεν μας ενδιαφέρει να κρατάμε μόνο τη μεγαλύτερη προσφορά που γίνεται για ένα προϊόν αλλά χρειαζόμαστε πλήρες ιστορικό προσφορών, είναι απαραίτητη η ύπαρξη αυτού του πίνακα. Κάθε εγγραφή του πίνακα αναπαριστά μία προσφορά που έγινε για ένα προϊόν μαζί με τον χρήστη που την έκανε, την ακριβή ώρα και το προσφερόμενο ποσό.

Έχει τα εξής πεδία:

- auction_id
Σε αυτή την στήλη αποθηκεύεται το αναγνωριστικό της δημοπρασίας για την οποία έγινε η συγκεκριμένη προσφορά. Πρόκειται για ένα ξένο κλειδί, το οποίο συσχετίζει τον πίνακα bid με τον πίνακα auction. Κατά συνέπεια είναι και αυτό ένας ακέραιος (int) ο οποίος όμως δεν αυξάνεται αυτόματα αφού χρειάζεται να τον εισάγουμε αυθαίρετα σε κάθε εγγραφή.

-user_id
Σε αυτή τη στήλη αποθηκεύεται το αναγνωριστικό του χρήστη από τον οποίο δόθηκε η μεγαλύτερη προσφορά. Πρόκειται για ένα ξένο κλειδί, το οποίο συσχετίζει τον πίνακα bid με τον πίνακα users. Κατά συνέπεια είναι και αυτό ένας ακέραιος (int) ο οποίος όμως δεν αυξάνεται αυτόματα αφού χρειάζεται να τον εισάγουμε αυθαίρετα σε κάθε εγγραφή.

-bid_time
Σε αυτή τη στήλη αποθηκεύεται η ημερομηνία και η ώρα που καταχωρήθηκε μία προσφορά για μια συγκεκριμένη δημοπρασία.Είναι ένα πεδίο date, ορισμένο σαν timestamp.

-amount
Σε αυτή στήλη αποθηκεύεται το ποσό ή τα ποσά που προσφέρει ο κάθε χρήστης για μια συγκεκριμένη δημοπρασία.Είναι ένα αριθμητικό πεδίο, ορισμένο σαν double.

Δημιουργία κώδικα

```
CREATE TABLE `bid` (
  `auction_id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  `bid_time` timestamp NOT NULL default CURRENT_TIMESTAMP,
  `amount` double NOT NULL
) ENGINE=MyISAM DEFAULT CHARSET=utf8;
```

Πίνακας category

	Field	Type	Collation	Attributes	Null	Default	Extra	Action
	id	int(11)			No		auto_increment	
	name	varchar(50)	utf8_general_ci		Yes	*NULL*		

Εικόνα 3 Πίνακας category

Ο πίνακας category είναι ο πίνακας στον οποίο αποθηκεύουμε τις πληροφορίες για τις κατηγορίες των προϊόντων του συστήματος. Κάθε εγγραφή σε αυτόν τον πίνακα αναπαριστά μια κατηγορία του συστήματος frontend.
Περιέχει τα παρακάτω πεδία:

- id

Πρόκειται για τον μοναδικό ακέραιο που είναι το αναγνωριστικό πεδίο κάθε εγγραφής. Είναι ένας ακέραιος(int) ο οποίος έχει την ιδιότητα auto_increment της MySQL. Πρακτικά αυτό εξασφαλίζει ότι οι τιμές της στήλης θα είναι μοναδικές, καθώς δεν τις εισάγουμε χειροκίνητα, αλλά ανήκουν σε μια ακολουθία συνεχώς αυξανόμενων ακεραίων που ανατίθενται αυτόματα σε κάθε εισαγωγή.

- name

Σε αυτό το πεδίο κρατάμε το όνομα κάθε κατηγορίας , το οποίο και χρησιμοποιείται για να διαχωρίσει τα προϊόντα των δημοπρασιών ανά είδος. Είναι ένα πεδίο κειμένου, ορισμένο σαν varchar.

Δημιουργία κώδικα

```
CREATE TABLE `category` (
  `id` int(11) NOT NULL auto_increment,
  `name` varchar(50) default NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=utf8 AUTO_INCREMENT=20 ;
```

Πίνακας company

	Field	Type	Collation	Attributes	Null	Default	Extra
	id	int(11)			No		auto_increment
	username	varchar(50)	utf8_general_ci		No		
	pass	varchar(10)	utf8_general_ci		No		
	email	varchar(50)	utf8_general_ci		No		

Εικόνα 4 Πίνακας company

Ο πίνακας company είναι ο πίνακας στον οποίο αποθηκεύουμε τις πληροφορίες για τις εταιρίες που ανήκουν τα προϊόντα του συστήματος. Κάθε εγγραφή σε αυτόν τον πίνακα αναπαριστά μια εταιρία του συστήματος frontend.
Περιέχει τα παρακάτω πεδία:

- id
Πρόκειται για τον μοναδικό ακέραιο που είναι το αναγνωριστικό πεδίο κάθε εγγραφής. Είναι ένας ακέραιος(int) ο οποίος έχει την ιδιότητα auto_increment της MySQL. Πρακτικά αυτό εξασφαλίζει ότι οι τιμές της στήλης θα είναι μοναδικές, καθώς δεν τις εισάγουμε χειροκίνητα, αλλά ανήκουν σε μια ακολουθία συνεχώς αυξανόμενων ακεραίων που ανατίθενται αυτόματα σε κάθε εισαγωγή.

- username
Σε αυτό το πεδίο κρατάμε το όνομα κάθε εταιρίας, το οποίο και χρησιμοποιεί κατά την διαδικασία εισαγωγής της στο σύστημα. Είναι ένα πεδίο κειμένου, ορισμένο σαν varchar.

Δημιουργία κώδικα

```
CREATE TABLE `company` (
  `id` int(11) NOT NULL auto_increment,
  `username` varchar(50) NOT NULL,
  `pass` varchar(10) NOT NULL,
  `email` varchar(50) NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=utf8 AUTO_INCREMENT=12 ;
```

Πίνακας auction

	Field	Type	Collation	Attributes	Null	Default	Extra
☐	id	int(11)			No		auto_increment
☐	title	varchar(50)	utf8_general_ci		No		
☐	description	varchar(500)	utf8_general_ci		No		
☐	last_bid_time	datetime			Yes	*NULL*	
☐	start_price	double			No		
☐	current_price	double			No		
☐	user_id	int(11)			Yes	*NULL*	
☐	category_id	int(11)			No		
☐	start_date	datetime			No		
☐	company_id	int(11)			No		
☐	shipped	tinyint(1)			No	0	
☐	image	varchar(100)	utf8_general_ci		Yes	*NULL*	

Εικόνα 5 Πίνακας auction

Στον πίνακα auction αποθηκεύουμε όλες τις προσφορές δημοπορασίες που έχουν γίνει ή πρόκειται να γίνουν. Κάθε εγγραφή του πίνακα αναπαριστά μία δημοπρασία που έγινε ή θα γίνει για ένα προϊόν μαζί με τον χρήστη που την έκανε,την εταιρία στην οποία ανήκει, την κατηγορία του προϊόντος, την ακριβή ώρα έναρξης, την αρχική και την τρέχουσα τιμή της, το όναμα και την πλήρη περιγραφή της, την κατάστασή της (ανοιχτή/κλειστή), την ώρα που έγινε η τελευταία προσφορά, καθώς και αν το προϊόν αποστάλθηκε ή όχι στον πελάτη.

Αν και όλες οι προσφορές που έχουν γίνει για μια δημοπρασία κρατούνται στον πίνακα bid, σε αυτόν τον πίνακα αποθηκεύονται η μεγαλύτερη προσφορά που έχει γίνει μαζί με την ώρα που έχει γίνει και τον χρήστη που την έκανε. Αν κάποια στιγμή γίνει μία καλύτερη προσφορά τότε οι αντίστοιχες τιμές αντικαθιστούν τις υπάρχουσες στον πίνακα.
Έχει τα εξής πεδία:

- id
Πρόκειται για τον μοναδικό ακέραιο που είναι το αναγνωριστικό πεδίο κάθε εγγραφής. Είναι ένας ακέραιος(int) ο οποίος έχει την ιδιότητα auto_increment της MySQL. Πρακτικά αυτό εξασφαλίζει ότι οι τιμές της στήλης θα είναι μοναδικές, καθώς δεν τις εισάγουμε χειροκίνητα, αλλά ανήκουν σε μια ακολουθία συνεχώς αυξανόμενων ακεραίων που ανατίθενται αυτόματα σε κάθε εισαγωγή.

-user_id
Σε αυτή τη στήλη αποθηκεύεται το αναγνωριστικό του χρήστη ο οποίος έκανε τη μεγαλύτερη προσφορά. Πρόκειται για ένα ξένο κλειδί, το οποίο συσχετίζει τον πίνακα auction με τον πίνακα users. Κατά συνέπεια είναι και αυτό ένας ακέραιος (int) ο οποίος όμως δεν αυξάνεται αυτόματα αφού χρειάζεται να τον εισάγουμε αυθαίρετα σε κάθε εγγραφή.
Αυτό το πεδίο έχει αρχική τιμή 0. Οταν γίνει η πρώτη προσφορά που είναι μεγαλύτερη από την αρχική τιμή τότε εισάγεται το αναγνωριστικό του χρήστη που την έκανε.

-company_id
Σε αυτή τη στήλη αποθηκεύεται το αναγνωριστικό της εταιρίας στην οποία ανήκει το προϊόν που δημοπρατείται. Πρόκειται για ένα ξένο κλειδί, το οποίο συσχετίζει τον πίνακα auction με τον πίνακα company. Κατά συνέπεια είναι και αυτό ένας ακέραιος (int) ο οποίος όμως δεν αυξάνεται αυτόματα αφού χρειάζεται να τον εισάγουμε αυθαίρετα σε κάθε εγγραφή.
Το πεδίο παίρνει την τιμή του κατά την δημιουργία της δημοπρασίας.

-category_id
Σε αυτή τη στήλη αποθηκεύεται το αναγνωριστικό κάθε κατηγορίας στην οποία ανήκει το προϊόν που δημοπρατείται. Πρόκειται για ένα ξένο κλειδί, το οποίο συσχετίζει τον πίνακα auction με τον πίνακα category. Κατά συνέπεια είναι και αυτό ένας ακέραιος (int) ο οποίος όμως δεν αυξάνεται αυτόματα αφού χρειάζεται να τον εισάγουμε αυθαίρετα σε κάθε εγγραφή.
Το πεδίο παίρνει την τιμή του κατά την δημιουργία της δημοπρασίας.

-title
Σε αυτή τη στήλη αποθηκεύεται το όνομα κάθε δημοπρασίας. Είναι ένα πεδίο κειμένου, ορισμένο σαν varchar.

-description
Σε αυτή τη στήλη αποθηκεύεται η αναλυτική περιγραφή κάθε δημοπρασίας. Είναι ένα πεδίο κειμένου, ορισμένο σαν varchar.

-last_bid_time

Σε αυτή τη στήλη αποθηκεύεται η ώρα που έγινε η τελευταία προσφορά για την συγκεκριμένη δημοπρασία. Είναι ένα πεδίο date, ορισμένο σαν datetime.
Το πεδίο έχει αρχική τιμή null. Εάν γίνει κάποια προσφορά μεγαλύτερη από την αρχική τιμή τότε εισάγεται η ώρα που έγινε αυτή η προσφορά.

-start_price
Σε αυτή τη στήλη αποθηκεύεται η αρχική τιμή του προϊόντος. Είναι ένα αριθμητικό πεδίο, ορισμένο σαν double.

-current_price
Σε αυτή τη στήλη αποθηκεύεται η τρέχουσα τιμή του προϊόντος. Είναι ένα αριθμητικό πεδίο, ορισμένο σαν double.
Αρχικά το πεδίο έχει τιμή 0. Όταν γίνει μια προσφορά που είναι μεγαλύτερη από την αρχική τιμή τότε αυτή εισάγεται σε αυτό το πεδίο.

-shipped
Σε αυτή τη στήλη αποθηκεύεται αν το προϊόν έχει αποσταλλεί ή όχι στον πελάτη,.
Πρόκειται για ένα πεδίο tinyint, που είναι ο τύπος που χρησιμοποιεί η MySQL για να αναπαραστήσει τις Boolean τιμές.

-image
Σε αυτή τη στήλη αποθηκεύεται το όνομα του αρχείου που περιέχει την εικόνα του προϊόντος. Είναι ένα πεδίο κειμένου, ορισμένο σαν varchar.

Δημιουργία κώδικα

```
CREATE TABLE `auction` (
  `id` int(11) NOT NULL auto_increment,
  `title` varchar(50) NOT NULL,
  `description` varchar(500) NOT NULL,
  `last_bid_time` datetime default NULL,
  `start_price` double NOT NULL,
  `current_price` double NOT NULL,
  `user_id` int(11) default NULL,
  `category_id` int(11) NOT NULL,
  `start_date` datetime NOT NULL,
  `company_id` int(11) NOT NULL,
  `shipped` tinyint(1) NOT NULL default '0',
  `image` varchar(50) NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=utf8 AUTO_INCREMENT=24 ;
```

Δυνατότητες

Διαχωρισμός σε frontend και backend

Σε κάθε ιστοσελίδα πρέπει να υπάρχει διαχωρισμός μεταξύ frontend και backend.Frontend είναι το μέρος στο οποίο έχουν πρόσβαση μόνο οι απλοί χρήστες και μπορούν να ενημερωθούν για τις δημοπρασίες που είναι σε εξέλιξη,που πρόκειται να αρχίσουν στις επόμενες μέρες ή που έχουν ήδη λήξει.Επίσης μπόρουν να κάνουν κάποια προσφορά για όποια δημοπρασία τους ενδιαφέρει.Backend είναι το μέρος της ιστοσελίδας στο οποίο πρόσβαση έχει μόνο ένας χρήστης ο admin,ο οποίος έχει την δικαιοδοσία να διαχειρίζεται όλες τις δημοπρασίες.Μόνο αυτός μπορεί να προσθέσει κάποια δημοπρασία και να συμπληρώσει τα στοιχεία της καθώς επίσης και να προσθέσει ή να αφαιρέσει κάποια εταιρία ή κάποια κατηγορία.

Frontend

Είσοδος χρήστη

Αυτή η επιλογή δίνει την δυνατότητα στον χρήστη να κάνει την εισαγωγή του(login) στην ιστοσελίδα των δημοπρασιών.Η εισαγωγή γίνεται με τον εξής τρόπο: πηγαίνοντας στο κάτω αριστερά μέρος της σελίδας υπάρχουν δύο πεδία,το username και το password.Ο χρήστης θα πρέπει να συμπληρώσει το username που έχει επιλέξει και τον κωδικό πρόσβασης.Πατώντας το κουμπί Είσοδος γίνεται έλεγχος αν τα πεδία αυτά έχουν συμπληρωθεί σωστά,δηλαδή αν το username υπάρχει στην λίστα με τους χρήστες και αν ο κωδικός είναι αυτός που αντιστοιχεί στο συγκεκριμένο username.Αν όχι τότε θα πρέπει ο χρήστης να συμπληρώσει από την αρχή τα πεδία.Αν τα πεδία είναι σωστά συμπληρωμένα τότε γίνεται η εισαγωγή του χρήστη στην σελίδα.Η εισαγωγή δίνει την δυνατότητα στον χρήστη να κάνει κάποια προσφορά για μια δημοπρασία που τον ενδιαφέρει εκτός από το να δει τις δημοπρασίες που έχουν γίνει ή που πρόκειται να γίνουν.

Εγγραφή χρήστη

Στην ιστοσελίδα αυτή,προσφορά σε μια δημοπρασία μπορούν να κάνουν μόνο οι εγγεγραμμένοι χρήστες.Για να κάνει κάποιος επισκέπτης εγγραφή θα πρέπει να επιλέξει το "εγγραφή νέου χρήστη" στο κάτω αριστερά μέρος της σελίδας.Κάνοντας αυτή την επιλογή εμφανίζεται μια φόρμα την οποία πρέπει να συμπληρώσει.Η φόρμα έχει τα εξής πεδία:όνομα χρήστη,κωδικός και e-mail.Στο όνομα χρήστη και στον κωδικό συμπληρώνει το όνομα και το password με το οποίο θέλει να κάνει την εισαγωγή του.Αφού συμπληρωθεί αυτή η φόρμα πατώντας το κουμπί "Εισαγωγή" ολοκληρώνεται η εγγραφή του χρήστη με επιτυχία.

Προβολή λίστας κατηγοριών

Τα προϊόντα που δημοπρατούνται είναι χωρισμένα σε κατηγορίες ανάλογα με το είδος τους. Αυτό έχει γίνει για να είναι πιο εύκολο στον χρήστη που θέλει να κάνει μια προσφορά,να βρει που είναι η δημοπρασία που τον ενδιαφέρει.Η λίστα με τις κατηγορίες βρίσκεται στην μέση της αριστερής πλευράς της ιστοσελίδας.

Προβολή λίστας δημοπρασιών ανά κατηγορία

Πατώντας πάνω σε μια συγκεκριμένη κατηγορία εμφανίζεται μια λίστα η οποία περιέχει όλες τις δημοπρασίες της κατηγορίας αυτής ονομαστικά.Η λίστα αυτή βρίσκεται στην μέση της σελίδας και εκεί ο χρήστης μπορεί να ενημερωθεί για το ποιες από αυτές τις δημοπρασίες δεν έχουν αρχίσει ακόμη καθώς και το πόσος χρόνος απομένει μέχρι να αρχίσουν,ποιες είναι σε εξέλιξη και σε πόση ώρα λήγουν και ποιες έχουν λήξει.Επιπλέον εμφανίζεται και η τιμή του προϊόντος.(η αρχική τιμή αν η δημοπρασία δεν έχει αρχίσει ακόμα ή δεν έχει γίνει κάποια προσφορά ή η τιμή της μεγαλύτερης προσφοράς αν η δημοπρασία είναι ανοιχτή και κάποιος έχει κάνει προσφορά μεγαλύτερη από την αρχική)

Προβολή στοιχείων δημοπρασίας

Αν κάποιος χρήστης επιλέξει ένα συγκεκριμένο προϊόν τότε εμφανίζεται μία άλλη σελίδα η οποία του δίνει περισσότερες πληροφορίες για την δημοπρασία αυτού του προϊόντος.Αρχικά φαίνεται η κατηγορία στην οποία ανήκει αυτό το προϊόν η εταιρία στην οποία ανήκει καθώς και αναλυτική περιγραφή του προϊόντος.Επιπλέον εμφανίζεται και η κατάσταση της δημοπρασίας και η τιμή της.Αν η δημοπρασία δεν έχει αρχίσει ακόμα τότε στην κατάσταση βλέπουμε πόσος χρόνος απομένει μέχρι να αρχίσει και στην τιμή,ποια είναι η εναρκτήρια τιμή για το προϊόν αυτό.Σε περίπτωση που η δημοπρασία είναι ανοιχτή τότε στην κατάσταση βλέπουμε πόσος χρόνος απομένει μέχρι να λήξει και στην τιμή εμφανίζεται η μεγαλύτερη προφρορά καθώς και από ποιον χρήστη έγινε η προσφορά αυτή.Αν η δημοπρασία είναι ανοιχτή αλλά κάνεις χρήστης δεν έχει προσφέρει ποσό μεγαλύτερο από το αρχικό τοτέ στην τιμή εμφανίζεται πάλι η τιμή έναρξης του προϊόντος.Τέλος αν η δημοπρασία έχει λήξει τότε στην κατάσταση εμφανίζεται το μήνυμα “έχει λήξει” και στην τιμή,η τιμή της μεγαλύτερης προσφοράς αν κάποιος χρήστης πρόσφερε κάποιο ποσό ή η τιμή έναρξης αν καμμία προσφορά δεν ήταν μεγαλύτερη από την αρχική.

Προσφορά

Προσφορά για κάποια δημοπρασία μπορούν να κάνουν όλοι οι εγγεγραμμένοι χρήστες.Αφού επιλέξουν την δημοπρασία για την οποία ενδιαφέρονται πηγαίνουν στο σημείο της ιστοσελίδας που γίνεται η προβολή των στοιχείων για το συγκεκριμένο προϊόν.Στο πεδίο που βρίσκεται η τιμή του προϊόντος μπορούν να γράψουν το ποσό το οποίο θέλουν να προσφέρουν και στην συνέχεια πατώντας το διπλανό κουμπί που γράφει “bid” η τιμή αυτή αποθηκεύεται στο σύστημα.Αν η τιμή είναι μεγαλύτερη από την προϋπάρχουσα τότε η τιμή που πρόσφεραν εμφανίζεται στην σελίδα μαζί με το όνομα αυτού που έκανε την προσφορά αυτή.Αν η τιμή είναι μικρότερη τότε δεν υπάρχει καμμία αλλαγή στην σελίδα μας.Προσφορά σε μια δημοπρασία μπορεί να γίνει ακόμα και πριν την ημερομηνία έναρξης της.Η διαδικασία είναι ίδια με πριν με μοναδική διαφορά ότι το ποσό που πρόσφεραν εμφανίζεται αυτόματα την στιγμή που γίνεται η έναρξη της δημοπρασίας.

Αναζήτηση δημοπρασίας βάση ονόματος

Η ιστοσελίδα μας στο πάνω αριστερά μέρος έχει δύο κουμπιά αναζήτησης τα όποια βρίσκονται εκεί για να μπορεί ο χρήστης να βρει κάποιο προϊόν που τον ενδιαφέρει

αν δεν γνωρίζει ακριβώς σε ποια κατηγορία ανήκει.Η αναζήτηση μπορεί να γίνει βάσει ονόματος του προϊόντος ή βάσει εταιρίας.Πληκτολογώντας στο κουτάκι όλο ή μέρος του προϊόντος ή το όνομα της εταιρίας και πατώντας το "Go" στο κέντρο της σελίδας εμφανίζεται μια λίστα με όλα τα προϊόντα που βρέθηκαν με αυτό το όνομα ή όλα τα προϊόντα που ανήκουν στην εταιρία που ζητήσαμε.Η αναζήτηση υπάρχει για την διευκόλυνση του χρήστη.

Backend

Login admin

Όπως έχουμε ήδη πει,πρόσβαση στο backend έχει μόνο ένας χρήστης,ο admin.Για να εξασφαλίσουμε ότι κανένας άλλος εκτός από αυτόν δεν θα μπορεί να μπει,έχουμε ζητήσει να συμπληρώνεται μια φόρμα πριν από την εισαγωγή του.Η φόρμα αυτή έχει τα πεδία "όνομα χρήστη" και "κωδικός πρόσβασης"Αφού συμπληρωθεί αυτή η φόρμα και επιβεβαιωθεί ότι τα στοιχεία της είναι σωστά μόνο τότε γίνεται η εισαγωγή στο backend από όπου ο admin μπορεί να διαχειρίζεται τις δημοπρασίες.

Προσθήκη κατηγορίας

Μια από τις δυνατότητες του admin στο backend είναι η προσθήκη μιας κατηγορίας.Στο αριστερό μέρος της σελίδας υπάρχει ενας σύνδεσμος "Add a category".Πατώντας εκεί,εμφανίζεται στην μέση της σελίδας η φόρμα η οποία πρέπει να συμπληρωθεί για να γίνει η προσθήκη μιας κατηγορίας.Συμπληρώνοντας ένα όνομα μιας νέας κατηγορίας και πατώντας το κουμπί "Εισαγωγή" βλέπουμε το μήνυμα "Η καταχώρηση έγινε με επιτυχία" και ξέρουμε ότι η διαδικασία προσθήκης της συγκεκριμένης κατηγορίας έχει ολοκληρωθεί.

Προβολή λίστας κατηγοριών

Επίσης στο backend μπορούμε να δούμε και όλες τις κατηγορίες που υπάρχουν στην ιστοσελίδα μας.Πατώντας τον σύνδεσμο "Show category list" μας εμφανίζει στο μέσο της σελίδας την λίστα με όλες τις κατηγορίες που έχουμε προσθέσει.Εκτός από το όνομα κάθε κατηγορίας εμφανίζει και το id της,το οποίο είναι αποθηκευμένο στην βάση μας.

Διαγραφή κατηγορίας

Όταν εμφανίζεται η λίστα με όλες τις κατηγορίες ,δίπλα ακριβώς από το όνομα βρίσκεται ένα "X" το οποίο αν πατήσει ο admin μπορεί να διαγράψει την συγκεκριμένη κατηγορία.

Προσθήκη εταιρίας

Επίσης ο admin έχει και την δυνατότητα προσθήκης μιας νέας εταιρίας.Αυτό γίνεται πατώντας στο αριστερό μέρος της σελίδας τον σύνδεσμο "Add a company".Τότε εμφανίζεται στο μέσο της σελίδας μια φόρμα η οποία πρέπει να συμπληρωθεί.Τα πεδία που έχει είναι τα εξής:όνομα εταιρίας όπου γράφουμε το όνομα της εταιρίας που θέλουμε να προσθέσουμε,κωδικός που είναι ένας κωδικός μοναδικός για κάθε εταιρία και e-mail που είναι το e-mail της συγκεκριμένης εταιρίας.Τέλος αφού έχουν συμπληρωθεί όλα τα πεδία πατώντας το κουμπί "Εισαγωγή"εμφανίζεται το μήνυμα "Η καταχώρηση έγινε με επιτυχία"και μόνο τότε ολοκληρώνεται η διαδικασία προσθήκης της νέας εταιρίας.

Προβολή λίστας εταιρειών

Επίσης στο backend μπορούμε να δούμε και όλες τις εταιρίες που υπάρχουν στην ιστοσελίδα μας.Πατώντας τον σύνδεσμο "Show company list" μας εμφανίζει στο μέσο της σελίδας την λίστα με όλες τις εταιρίες που έχουμε προσθέσει.Εκτός από το όνομα κάθε εταιρίας εμφανίζει και το id της,το οποίο είναι αποθηκευμένο στην βάση μας.

Διαγραφή εταιρίας

Όπως και στην διαγραφή κατηγορίας έτσι και στην διαγραφή εταιρίας,όταν εμφανίζεται η λίστα με όλες τις εταιρίες ,δίπλα ακριβώς από το όνομα βρίσκεται ένα "Χ" το οποίο αν πατήσει ο admin μπορεί να διαγράψει την συγκεκριμένη εταιρία.

Προσθήκη χρήστη

Επίσης ο admin έχει και την δυνατότητα προσθήκης ενός νέου χρήστη, σε περίπτωση που είναι απαραίτητο. Αυτό γίνεται πατώντας στο αριστερό μέρος της σελίδας τον σύνδεσμο "Add a user". Τότε εμφανίζεται στο μέσο της σελίδας μια φόρμα η οποία πρέπει να συμπληρωθεί.Τα πεδία που έχει είναι τα εξής:όνομα χρήστη όπου γράφουμε το όνομα του χρήστη που θέλουμε να προσθέσουμε, κωδικός που είναι ένας κωδικός μοναδικός για κάθε χρήστη και e-mail που είναι το e-mail του συγκεκριμένου χρήστη. Τέλος αφού έχουν συμπληρωθεί όλα τα πεδία πατώντας το κουμπί "Εισαγωγή"εμφανίζεται το μήνυμα "Η καταχώρηση έγινε με επιτυχία"και μόνο τότε ολοκληρώνεται η διαδικασία προσθήκης του νέου χρήστη.

Προβολή λίστας χρηστών

Επίσης στο backend μπορούμε να δούμε όλους τους χρήστες που υπάρχουν στην ιστοσελίδα μας. Πατώντας τον σύνδεσμο "Show users list" μας εμφανίζει στο μέσο της σελίδας την λίστα με όλους τους χρήστες που είναι εγγεγραμένοι στο σύστημα. Εκτός από το όνομα κάθε χρήστη εμφανίζει και το id του,το οποίο είναι αποθηκευμένο στην βάση μας.

Διαγραφή χρήστη

Όπως και στην διαγραφή κατηγορίας έτσι και στην διαγραφή χρήστη, όταν εμφανίζεται η λίστα με όλους τους χρήστες,δίπλα ακριβώς από το όνομα βρίσκεται ένα "Χ" το οποίο αν πατήσει ο admin μπορεί να διαγράψει τον συγκεκριμένο χρήστη.

Προσθήκη δημοπρασίας

Στην προσθήκη δημοπρασίας ο admin δημιουργεί και προσθέτει μια νέα δημοπρασία.Πατώντας στον σύνδεσμο "Add an auction" εμφανίζεται μια φόρμα με όλα τα πεδία που πρέπει να συμπληρωθούν.Αρχικά είναι το όνομα της δημοπρασίας που ο admin γράφει τον τίτλο της.Στην συνέχεια βρίσκεται το πεδίο "Εικόνα" όπου με ένα dropdown μενού εμφανίζονται τα ονόματα όλων των φωτογραφιών.Οι φωτογραφίες αποθηκεύονται στον φάκελο C:\Program Files\Apache Software Foundation\Apache Tomcat 6.0.16\webapps\ROOT\product_images. Έπειτα είναι η περιγραφή της δημοπρασίας που γράφει αναλυτικά κάποια χαρακτηριστικά για το προϊόν που δημοπρατείται.Στην ημερομηνία έναρξης,πατώντας στο κουμπί που βρίσκεται δίπλα εμφανίζεται ένα ημερολόγιο και επιλέγει την ημερομηνία και την ώρα που θέλει να ξεκινήσει η δημοπρασία.Στο πεδίο με το όνομα "Τιμή" συμπληρώνει την εναρκτήρια τιμή για το προϊόν που πρόκειται να δημοπρατηθεί.Στην συνέχεια είναι το πεδίο "Κατηγορία" όπου υπάρχει ένα dropdown

μενού με όλες τις υπάρχουσες κατηγορίες και ο admin επιλέγει την κατηγορία στην οποία ανήκει το συγκεκριμένο προϊόν.Τέλος είναι το πεδίο "Εταιρία" που και εδώ με ένα dropdown μενού επιλέγει την εταιρία στην οποία ανήκει το προϊόν που δημοπρατείται.Η διαδικασία προσθήκης ολοκληρώνεται πατώντας το κουμπί "Εισαγωγή".

Προβολή λίστας δημοπρασιών

Για να δούμε την λίστα με όλες τις δημοπρασίες πατάμε στον σύνδεσμο "Show auction list" που μας εμφανίζει στην μέση της σελίδας όλες τις δημοπρασίες.Εκτος από το όνομα τους,εμφανίζει και την κατάσταση αποστολής δηλαδή αν το προϊόν έχει αποσταλλεί ή όχι στον χρήστη που έκανε την μεγαλύτερη προσφορά.Επίσης εμφανίζει και το id της δημοπρασίας,το οποίο είναι αποθηκευμένο στην βάση μας.

Προβολή άγονων δημοπρασιών

Μια άλλη επιλόγη του backend είναι η προβολή των άγονων δημοπρασίων.Άγονη είναι η δημοπρασία στην οποία δεν έχει γίνει καμία προσφορά μεγαλύτερη από την αρχική τιμή.Πατώντας στον σύνδεσμο "Show unproductive auction list" εμφανίζεται ένας πίνακας με τον τίτλο της δημοπρασίας,την αρχική τιμή του προϊόντος και την μεγαλύτερη προσφορά.Το τελευταίο πεδίο υπάρχει γιατί οι χρήστες μπορούν να κάνουν και προσφορές μικρότερες από την εναρκτήρια τιμή του προϊόντος.Κάτω από τον πίνακα με τις άγονες δημοπρασίες αναφέρεται και το σύνολο των άγονων δημοπρασιών.

Προβολή γόνιμων δημοπρασιών

Επίσης η τελευταία επιλογή στο backend είναι η προβολή των γόνιμων δημοπρασίων.Πατώντας στον σύνδεσμο "Show productive auction list" εμφανίζεται ένας πίνακας με τον τίτλο της δημοπρασίας,την αρχική της τιμή,την τιμή κατωχύπωσης και το αν το προϊόν έχει αποσταλλεί ή όχι στον πελάτη.

Προβολή μέσου χρόνου δημοπρασιών

Στο κάτω αριστερά μέρος του backend υπάρχει και μια δήλωση η οποία μας ενημερώνει για την μέση διάρκεια όλων των δημοπρασίων

Δομή εφαρμογής

Κλάσσεις

Util/DBUtil.java

Καθώς σε κάθε σχεδόν αρχείο της εφαρμογής χρειάζεται να συνδεθούμε στην βάση δεδομένων για να πάρουμε πληροφορίες θεωρήσαμε σκόπιμο να δημιουργήσουμε την κλάσση DBUtil η οποία συγκεντρώνει τον κατάλληλο κώδικα για την σύνδεση στην βάση και την εκτέλεση επερωτήσεων. Μέσα σε αυτό το αντικείμενο υπάρχουν 3 static συναρτήσεις, οι query, update και close.

```
public static ResultSet query(String q) throws
ClassNotFoundException, SQLException, SQLException;
```

Η συνάρτηση query αρχικά συνδέεται στην βάση δεδομένων και έπειτα εκτελεί ένα SQL query το οποίο παίρνει σαν παράμετρο.

```
stm.executeUpdate(q);
```

Επιστρέφει ένα ResultSet με τα αποτελέσματα του query.

```
public static void update(String q) throws
ClassNotFoundException, SQLException, SQLException{
```

Η συνάρτηση update εκτελεί μία εισαγωγή, μία διαγραφή ή μια τροποποίηση κάποιου πίνακα στην βάση. Παίρνει σαν παράμετρο το αντίστοιχο query και αφού συνδεθεί στην βάση το εκτελεί.

```
stm.executeUpdate(q);
```

Μία αλλαγή στην βάση δεν επιστρέφει αποτελέσματα, οπότε ο τύπος επιστροφής της συνάρτησης είναι void.

```
public static void close() throws SQLException;
```

Η συνάρτηση close κλείνει μία σύνδεση που έχουμε ήδη ανοίξει στην βάση.

/util/SiteUtil.java

Το αρχείο SiteUtil περιέχει δίαφορες βοηθητικές συναρτήσεις που επειδή καταλαμβάνουν αρκετό όγκο ή είναι αρκετά πολύπλοκες έχουν συγκεντρωθεί εδώ.

```
public static String getAuctionStatus(Date startDate, Date
lastBidDate;
```

Η συνάρτηση getAuctionStatus παίρνει για παράμετρο την ημερομηνία έναρξης μίας δημοπρασίας καθώς και την ημερομηνία που έχει γίνει η τελευταία προσφορά. Η συνάρτηση αποφασίζει αν η δημοπρασία έχει λήξει ή όχι και επιστρέφει το κατάλληλο String.

```
if(currentDate.before(startDate)){
```

Αν η ημερομηνία έναρξης είναι στο μέλλον τότε η δημοπρασία δεν έχει ξεκινήσει ακόμη. Η συνάρτηση υπολογίζει πόσος χρόνος μεσολαβεί μέχρι την ημερομηνία έναρξης και επιστρέφει το κατάλληλο String.

```
}else if(lastBidDate == null && startDate.before(currentDate)
&& endDate10.after(currentDate)){
```

Σε αντίθετη περίπτωση αν η ημερομηνία έναρξης βρίσκεται στο παρελθόν, δεν έχει γίνει κάποια προσφορά αλλά δεν έχουν παρέλθει 10 λεπτά από την ημερομηνία έναρξης, τότε η συνάρτηση υπολογίζει πόσος χρόνος απομένει μέχρι να γίνει άγονη και επιστρέφει το κατάλληλο String.

}else if(lastBidDate != null && lastBidDate.before(currentDate) && endDate5.after(currentDate)){

Σε αντίθετη περίπτωση αν έχει γίνει κάποια προσφορά και δεν έχουν παρέλθει 5 λεπτά από την ώρα που έγινε η δημοπρασία επιστρέφει πόσος χρόνος απομένει από την λήξη του πεντάλεπτου ώστε να κατωχυρωθεί το προϊόν.

```
}else{
      return "Έχει λήξει";
}
```

Σε αντίθετη περίπτωση έχουμε εξαντλήσει όλες τις δυνατές περιπτώσεις οπότε η δημοπρασία έχει λήξει.

```
public static boolean isAuctionOpen(Date startDate, Date
lastBidDate);
```

Η παραπανω συνάρτηση ελέγχει αν η δημοπρασία είναι ανοικτή και επιστρέφει true ή false, παίρνοντας παρόμοιες περιπτώσεις με την getAuctionStatus. Χρησιμοποιείται για να ελέγχουμε αν σε μια δημοπρασία μπορούμε να κάνουμε νέες προσφορές.

```
public static boolean hasAuctionOpened(Date startDate);
```

Η συνάρτηση hasAuctionOpened ελέγχει αν έχει παρέλθει ο χρόνος έναρξης μιας δημοπρασίας και επιστρέφει true ή false αντίστοιχα.

Frontend

frontend/header.jsp

Το αρχείο header,jsp περιέχει τον κώδικα που εμφανίζει την επικεφαλίδα της σελίδας.

frontend/left_menu.jsp

Το αρχείο left_menu,jsp ξεκινάει με τον κώδικα που αναφέρεται στις αναζητήσεις που μπορεί να κάνει κάποιος χρήστης είτε ανά προϊόν, είτε ανά εταιρία. Στην ανά προίόν αναζήτηση έχουμε έναν πίνακα δύο γραμμών. Στην πρώτη γραμμή

αναφέρεται ο τίτλος της αναζήτησης. Η δεύτερη γραμμή περιέχει μία απλή φόρμα html που θα μπορεί να συμπληρώσει ο χρήστης αν θέλει να αναζητήσει ένα συγκεκριμένο προϊόν και περιέχει τα πεδία searchString και Go. Το action της φόρμας είναι το αρχείο searchProduct.jsp που βρίσκεται στον φάκελο frontend.Στην ανά εταιρία αναζήτηση έχουμε έναν πίνακα δύο γραμμών. Στην πρώτη γραμμή αναφέρεται ο τίτλος της αναζήτησης. Η δεύτερη γραμμή περιέχει μία φόρμα html που θα μπορεί να συμπληρώσει ο χρήστης αν θέλει να αναζητήσει ένα προϊόν συγκεκριμένης εταιρίας και περιέχει τα πεδία Go και company_id, το οποίο είναι ένα dropdown μενού. Για να δημιουργήσουμε το dropdown με τις εταιρίες κάνουμε υποερώτηση στη βάση και επιλέγουμε όλα τα περιεχόμενα του πίνακα company:

```
String query2 = "SELECT * FROM company";
```

Με μια while διατρέχουμε όλα τα αποτελέσματα και για κάθε ένα πέρνουμε τα περιεχόμενα των στηλών id και name ώστε να δημιουργήσουμε την αντίστοιχη επιλογή του select:

```
<option value="<%=id%>" ><%=name%></option>
```

Το action της φόρμας είναι το αρχείο searchByCompany.jsp που βρίσκεται στον φάκελο frontend.

Μετά τις αναζητήσεις στο left_menu υπάρχει μία λίστα από links με όλες τις κατηγορίες. Κάνουμε υποερώτηση στη βάση και επιλέγουμε όλα τα περιεχόμενα του πίνακα category, αλλά επιλεγμένα με βάση το όνομα τους.

```
String query1 = "SELECT * FROM category ORDER BY name";
```

Με μια while διατρέχουμε όλα τα αποτελέσματα και για κάθε ένα πέρνουμε τα περιεχόμενα των στηλών id και name ώστε να εμφανίσουμε αυτό που θέλουμε:

```
<li><a href="showCategory.jsp?id=<%=id %>"><%=name%></a></li>
```

Σε περίπτωση που ο χρήστης έχει ήδη εισαχθεί στο σύστημα τότε υπάρχει στο session η μεταβλητή username. Σε αυτή την περίπτωση εμφανίζουμε ένα κείμενο που τον υποδέχεται και ένας σύνδεσμος μέσω του οποίου μπορεί να κάνει logout.

Εάν ο χρήστης δεν έχει κάνει είσοδο στο σύστημα, στο κάτω μέρος του menu εμφανίζεται η φόρμα εισαγωγής στο σύστημα. Είναι μια απλή html φόρμα που περιέχει τα πεδία username, pass και Είσοδος. Το action της φόρμας είναι το Login που βρίσκεται στο φάκελο source Package.

Τέλος η «εγγραφή νέου χρήστη» είναι ένα link στο αρχείο register.jsp που βρίσκεται στο φάκελο source Package.

searchProduct.jsp

Αυτό το αρχείο είναι υπεύθυνο για τα αποτελέσματα της αναζήτησης του πεδίου searchString του παραπάνω αρχείου.

```
request.getParameter("searchString")
```

Κάνουμε υποερώτηση στη βάση και διελέγουμε όλα τα περιεχόμενα του πίνακα auction που ο τίτλος του ταιριάζει με το προϊόν που ψάχνουμε:

```
String query = "SELECT * FROM auction WHERE title LIKE '%" +
request.getParameter("searchString") + "%'";
```

Κατόπιν εκτελούμε την επιλογή :

```
DBUtil. query(query);
```

Με μια while διατρέχουμε όλα τα αποτελέσματα και για κάθε ένα παίρνουμε τα περιεχόμενα των στηλών id, title, start_date, last_bid_time, start_price και current_price και με μία if ελέγχουμε την τιμή της προσφοράς.

Για κάθε εγγραφή δημιουργούμε μια γραμμή ενός πίνακα που περιέχει την φωτογραφία, το όνομα, την τιμή και την κατάσταση της δημοπρασίας. Η φωτογραφία και το όνομα είναι σύνδεσμοι στο αρχείο showAuction.jsp .

searchByCompany.jsp

Αυτό το αρχείο είναι υπεύθυνο για τα αποτελέσματα της αναζήτησης του πεδίου company_id του παραπάνω αρχείου left_menu.

```
request.getParameter("company_id")
```

Κάνουμε υποερώτηση στη βάση και διελέγουμε όλα τα περιεχόμενα του πίνακα auction που το id του ταιριάζει το id της εταιρίας που ψάχνουμε:

```
String query = "SELECT * FROM auction WHERE company_id='" +
request.getParameter("company_id") + "'";
```

Κατόπιν εκτελούμε την επιλογή :

```
DBUtil. query(query);
```

Με μια while διατρέχουμε όλα τα αποτελέσματα και για κάθε ένα παίρνουμε τα περιεχόμενα των στηλών id, title, start_date,last_bid_time, start_price και current_price και με μία if ελέγχουμε την τιμή της προσφοράς.

Για κάθε εγγραφή δημιουργούμε μια γραμμή ενός πίνακα που περιέχει την φωτογραφία, το όνομα, την τιμή και την κατάσταση της δημοπρασίας. Η φωτογραφία και το όνομα είναι σύνδεσμοι στο αρχείο showAuction.jsp .

showCategory.jsp

Αυτό το αρχείο είναι υπεύθυνο για την εμφάνιση της λίστας με τις δημοπρασίες μιας συγκεκριμένης κατηγορίας. Αρχικά βρίσκουμε το όνομα της κατηγορίας που μας ενδιαφέρει:

```
String query = "SELECT * FROM category WHERE id=" +
request.getParameter("id");
```

Δημιουργούμε μια σειρά με συνδέσμους στην αρχική σελίδα και στην κατηγορία.

```
<div class="breadcrumb">
<a href="/Project/index.jsp">Αρχική σελίδα</a> &gt;
<%=catName %>
</div>
```

Κάνουμε υποερώτηση στη βάση και διελέγουμε όλα τα περιεχόμενα του πίνακα auction που το id του ταιριάζει το id της εταιρίας που ψάχνουμε:

```
String query = "SELECT * FROM auction WHERE category_id=" +
request.getParameter("id");
```

Κατόπιν εκτελούμε την επιλογή :

```
DBUtil. query(query);
```

Με μια while διατρέχουμε όλα τα αποτελέσματα και για κάθε ένα παίρνουμε τα περιεχόμενα των στηλών id, title, start_date,last_bid_time, start_price και current_price και με μία if ελέγχουμε την τιμή της προσφοράς.

Για κάθε εγγραφή δημιουργούμε μια γραμμή ενός πίνακα που περιέχει την φωτογραφία, το όνομα, την τιμή και την κατάσταση της δημοπρασίας. Η φωτογραφία και το όνομα είναι σύνδεσμοι στο αρχείο showAuction.jsp .

showAuction.jsp

Αυτό το αρχείο είναι υπεύθυνο για την προβολή των στοιχείων μίας δημοπρασίας. Αρχικά παίρνουμε το id της δημοπρασίας που μας ενδιαφέρει και δημιουργούμε μία επερώτηση με την οποία παίρνουμε όλα τα στοιχεία της, το όνομα της εταιρείας που δημοσίευσε την δημοπρασία και το όνομα της κατηγορίας στην οποία ανήκει.

```
String query = "SELECT auction.id, description, name, image,
title, last_bid_time, start_price, current_price, category_id,
start_date, company_id, company.username as company_name,
user_id FROM auction, company, category WHERE
auction.company_id=company.id AND
auction.category_id=category.id AND auction.id=" +
request.getParameter("id");
```

Έπειτα εμφανίζουμε τα παραπάνω πεδία σε έναν πίνακα.
Εάν η δημοπρασία δεν έχει ξεκινήσει τυπώνεται η αρχική τιμή, σε αντίθετη περίπτωση τυπώνεται η καλύτερη προσφορά που έχει γίνει και το όνομα του χρήστη που την έκανε.

```
if(SiteUtil.hasAuctionOpened(startDate) && lastBidder !=
null){
      out.print(currentPrice + "&euro; από " + lastBidder);
}else{
      out.print(startPrice + " &euro;");
}
```

Επιπλέον εάν ο χρήσης που βλέπει την δημοπρασία έχει συνδεθεί στο σύστημα του δίνεται η δυνατότητα να κάνει μια νέα προσφορά.

```
if(SiteUtil.isAuctionOpen(startDate, lastBidTime) &&
request.getSession().getAttribute("userId") != null){
```

Σε αυτή την περίπτωση δημιουργούμε μια φόρμα με τα πεδία ammount, auctionId και currentPrice που έχει σαν action το PlaceBid.java
Τέλος υπάρχει και ένα τμήμα κώδικα javascript το οποίο είναι υπεύθυνο για την ανανέωση της σελίδας της δημοπρασίας κάθε ένα λεπτό ώστε ο χρήστης να ενημερώνεται για τυχόν αλλαγές.

```
setTimeout("window.location.reload()",60*1000);
```

frontend/footer.jsp

Το αρχείο footer,jsp περιέχει τον κώδικα που εμφανίζεται στο κάτω μέρος κάθε σελίδας.

index.jsp

Κάθε αρχείο του frontend είναι χωρισμένο σε τέσσερα τμήματα. Τα τρία πρώτα είναι η επικεφαλίδα, το κεντρικό μενού και το υποσέλιδο, τα οποία δημιουργούνται κάνοντας include τα τρία παρακάτω αρχεία:

```
<%@include file="frontend/header.jsp" %>
<%@include file="frontend/left_menu.jsp" %>
<%@include file="frontend/footer.jsp" %>
```

Στο κεντρικό τμήμα της σελίδας σχηματίζουμε το string της επερώτησης, και διελέγουμε όλα τα περιεχόμενα του πίνακα auction με βάση την ημερομηνία έναρξής τους κατά φθίνουσα σειρά των πέντε πρώτων δημοπρασιών:

```
String query = "SELECT * FROM auction ORDER BY start_date DESC
LIMIT 5";
```

Κατόπιν εκτελούμε την επιλογή :

```
DBUtil. query(query);
```

Με μια while διατρέχουμε όλα τα αποτελέσματα και για κάθε ένα παίρνουμε τα περιεχόμενα των στηλών id, title και image για κάθε δημοπρασία.

Για κάθε εγγραφή δημιουργούμε μια γραμμή ενός πίνακα που περιέχει την φωτογραφία, το όνομα, την τιμή και την κατάσταση της δημοπρασίας. Η φωτογραφία και το όνομα είναι σύνδεσμοι στο αρχείο showAuction.jsp .

Login.java

Αυτό το servlet είναι υπεύθυνο για την εισαγωγή του χρήστη. Αρχικά χρησιμοποιούμε τις παραμέτρους που μας έχουν δωθεί από το left_menu.jsp για να πάρουμε το όνομα του χρήστη και τον κωδικό του, στην προκειμένη περίπτωση τις παραμέτρους username και pass:

```
request.getParameter("username"),
request.getParameter("pass")
```

Χρησιμοποιώντας το σχηματίζουμε το string της επερώτησης, δηλαδή το:

```
String query = "SELECT * FROM users WHERE
username='"+username+"' AND pass='"+pass+"'";
```

Κατόπιν εκτελούμε την επιλογή :

```
DBUtil. query(query);
```

Τέλος ελέγχουμε αν υπάρχει καταχωρημένο το όνομα και ο κωδικός του χρήστη και με μία if τον ενημερώνουμε αν η εισαγωγή του ολοκληρώθηκε ή όχι.

Logout.java

Αυτό το servlet είναι υπεύθυνο για την έξοδο του χρήστη. Πατώντας το link "έξοδος" αφαιρούμε από το session τις μεταβλητές username και user_id:

```
request.getSession().removeAttribute("username");
request.getSession().removeAttribute("userId");
```

Έπειτα επίστρέφουμε στην αρχική σελίδα, στο αρχείο /index.jsp

register.jsp

Το αρχείο register.jsp περιέχει την φόρμα που χρησιμοποιεί ο χρήστης για την εισαγωγή του. Πρόκειται για μία φόρμα html στην οποία γίνεται χρήση κώδικα JavaScript, και περιέχει τα πεδία username, pass, pass2 και email. Η χρήση JavaScript γίνεται σε όλα τα πεδία για να κάνει υποχρεωτική την συμπλήρωση όλων των πεδίων.Επίσης επιβεβαιώνει τον κωδικό για μεγαλύτερη ασφάλεια.Αν οι δύο κωδικοί δεν συμπίπτουν μεταξύ τους, τότε εμφανίζεται το κατάλληλο μήνυμα. Το action της φόρμας είναι το αρχείο Register.java που βρίσκεται στον source Package φάκελο.

Register.java

Αυτό το servlet είναι υπεύθυνο για την εγγραφή ενός νέου χρήστη στο σύστημα. Αρχικά χρησιμοποιούμε την παράμετρο που μας έχει δωθεί από το αρχείο register.jsp για να πάρουμε το όνομα του χρήστη:

```
request.getParameter("username")
```

Χρησιμοποιώντας το σχηματίζουμε το string της επερώτησης, δηλαδή το:

```
String query = "SELECT * FROM users WHERE username='" +
request.getParameter("username") + "'";
```

Κατόπιν εκτελούμε την επιλογή :

```
DBUtil.query(query);
```

Αρχικά με μία if ελέγχουμε αν το όνομα του χρήστη υπάρχει και έπειτα χρησιμοποιούμε όλες τις παραμέτρους:

```
request.getParameter("username"),
request.getParameter("pass"),
request.getParameter("email")
```

και σχηματίζουμε το string της επερώτησης:

```
   query = "INSERT INTO users (username, pass, email) VALUES
('" + request.getParameter("username") +"',
'"+request.getParameter("pass")+"',
'"+request.getParameter("email")+"')";
```

Κατόπιν εκτελούμε την εισαγωγή :

```
DBUtil.update(query);
```

PlaceBid.java

Αυτό το servlet είναι υπεύθυνο για την προσφορά που μπορεί να κάνει ο χρήστης για κάποια δημοπρασία. Αρχικά χρησιμοποιούμε τις παραμέτρους auction_id, user_id, bid_time, amount:

```
request.getParameter("auctionId"),
request.getSession().getAttribute("userId"),
df.format(last_bid_time),
request.getParameter("ammount")
```

Χρησιμοποιώντας τις παραμέτρους αυτές σχηματίζουμε το string της επερώτησης, δηλαδή το:

```
String query = "INSERT INTO `bid` ( `auction_id` , `user_id` ,
`bid_time` , `amount` ) VALUES ('" +
request.getParameter("auctionId") + "', '" +
request.getSession().getAttribute("userId") + "', '" +
df.format(last_bid_time) + "' , '" + request.getParameter("ammount")
+"')";
```

Κατόπιν εκτελούμε την εισαγωγή στον πίνακα bid.

Επιπλέον κάνουμε μια επερώτηση στον πινακα auctions για να βρούμε την δημοπρασία για την οποία έγινε η προσφορά.

```
Class.forName("com.mysql.jdbc.Driver");
Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/db2?useUni
code=true&characterEncoding=UTF-8", "root", "");
Statement stm = con.createStatement();

...
query = "SELECT auction.id, description, name, title,
last_bid_time, start_price, current_price, category_id, start_date,
status, company_id, company.username as company_name, user_id FROM
auction, company, category WHERE auction.company_id=company.id AND
auction.category_id=category.id AND auction.id=" +
request.getParameter("auctionId");
```

Αν η τρέχουσα προσφορά είναι μεγαλύτερη από την τιμή της δημοπρασίας τότε ανανεώνουμε τα πεδία last_bid_time, user_id και current_price της τρέχουσας δημοπρασίας.

Backend

Τα jsp και html αρχεία που χρησιμοποιούνται στο backend βρίσκονται στον φάκελο admin.

Admin/index.jsp

Το αρχείο index.jsp περιέχει την φόρμα που χρησιμοποιεί ο administrator για την είσοδο του στο σύστημα. Πρόκειται για μία απλή φόρμα html στην οποία δεν γίνεται χρήση κώδικα java, και περιέχει τα πεδία username και pass. Το action της φόρμας είναι το αρχείο index2.jsp που βρίσκεται στον ίδιο φάκελο.

Admin/index2.jsp

Αυτό το αρχείο περιέχει το βασικό μενού του backend, το οποίο είναι μία σειρά html links που παραπέμπουν στις αντίστοιχες λειτουργίες.
Καταρχήν γίνεται έλεγχος των παραμέτρων username και pass που δόθηκαν από το αρχείο admin/index.jsp παραπάνω:

```
if(request.getParameter("username").equals("admin") &&
request.getParameter("pass").equals("lala"))
```

Σε περίπτωση που τα στοιχεία είναι σωστά εμφανίζεται η λίστα με τους συνδέσμους στις λειτουργίες του administrator. Επιπλέον υπολογίζεται και εμφανίζεται η μέση διάρκεια των δημοπρασιών. Για να γίνει αυτό επιλέγουμε όλες τις δημοπρασίες από την βαση:

```
ResultSet rs = DBUtil.query("SELECT * FROM `auction`");
```

Έπειτα διατρέχουμε τις δημοπρασίες. Για κάθε μία από αυτές βρίσκουμε το πότε ξεκίνησε και πότε έγινε η μεγαλύτερη προσφορά. Αν δεν έχει γίνει καμία προσφορά ξέρουμε ότι η δημοπρασία διήρκεσε 10 λεπτά, σε αντίθετη περίπτωση ξέρουμε ότι κράτησε 5 λεπτά έπειτα από την στιγμή που έγινε το μεγαλύτερο bid(προσφορά). Αθροίζοντας την διάρκεια των δημοπρασιών και διαιρώντας την με το πλήθος τους βρίσκουμε την μέση διάρκεια.

```
long avg = sum/count;
Date avgDate = new Date((long)avg);
```

Admin/category.jsp

Το αρχείο category.jsp περιέχει την φόρμα που χρησιμοποιεί ο administrator για την δημιουργία μιας νέας κατηγορίας. Πρόκειται για μία απλή φόρμα html στην οποία δεν γίνεται χρήση κώδικα java, και περιέχει το πεδίο categoryName. Το action της

φόρμας είναι το αρχείο /Project/AddCategory που βρίσκεται στον default package φάκελο.

AddCategory.java

Αυτό το servlet είναι υπεύθυνο για την εισαγωγή μιας νέας κατηγορίας προϊόντων. Αρχικά χρησιμοποιούμε τις παραμέτρους που μας έχουν δωθεί από το Admin/category.jsp για να πάρουμε το όνομα της κατηγορίας, στην προκειμένη περίπτωση την παράμετρο categoryName:

```
request.getParameter("categoryName")
```

Χρησιμοποιώντας το σχηματίζουμε το string της επερώτησης, δηλαδή το:

```
String query = "INSERT INTO category (name) VALUES ('" +
request.getParameter("categoryName") +"')";
```

Κατόπιν εκτελούμε την εισαγωγή :

```
DBUtil.update(query);
```

Τέλος δημιουργούμε τον κατάλληλο κώδικα HTML για να ενημερώνουμε τον χρήστη ότι η εισαγωγή ολοκληρώθηκε.

Admin/company.jsp

Το αρχείο company.jsp περιέχει την φόρμα που χρησιμοποιεί ο administrator για την εισαγωγή μιας νέας εταιρίας. Πρόκειται για μία απλή φόρμα html στην οποία δεν γίνεται χρήση κώδικα java, και περιέχει τα πεδία username, pass και email. Το action της φόρμας είναι το αρχείο /Project/AddCompany που βρίσκεται στον default package φάκελο.

AddCompany.java

Αυτό το servlet είναι υπεύθυνο για την εισαγωγή μιας νέας εταιρίας. Αρχικά χρησιμοποιούμε τις παραμέτρους που μας έχουν δωθεί από το Admin/company.jsp για να πάρουμε το όνομα της κατηγορίας,τον κωδικό της και το email της, στην προκειμένη περίπτωση τις παραμέτρους username,pass και email:

```
request.getParameter("username") ,
request.getParameter("pass"),
request.getParameter("email")
```

Χρησιμοποιώντας τις παραμέτρους αυτές σχηματίζουμε το string της επερώτησης, δηλαδή το:

```
String query = "INSERT INTO company (username, pass, email)
VALUES ('" + request.getParameter("username") +"',
'"+request.getParameter("pass")+"',
'"+request.getParameter("email")+"')";
```

Κατόπιν εκτελούμε την εισαγωγή :

```
DBUtil.update(query);
```

Τέλος δημιουργούμε τον κατάλληλο κώδικα HTML για να ενημερώνουμε τον χρήστη ότι η εισαγωγή ολοκληρώθηκε.

Admin/users.jsp

Το αρχείο users.jsp περιέχει την φόρμα που χρησιμοποιεί ο administrator για την εισαγωγή ενός νέου χρήστη,στην περίπτωση που αυτό κριθεί απαραίτητο. Πρόκειται για μία απλή φόρμα html στην οποία δεν γίνεται χρήση κώδικα java, που περιέχει τα πεδία username, pass και email. Το action της φόρμας είναι το αρχείο /Project/AddUser που βρίσκεται στον default package φάκελο.

AddUser.java

Αυτό το servlet είναι υπεύθυνο για την εισαγωγή ενός νέου χρήστη. Αρχικά χρησιμοποιούμε τις παραμέτρους που μας έχουν δωθεί από το Admin/users.jsp για να πάρουμε το όνομα του χρήστη,τον κωδικό του και το email του. Στην προκειμένη περίπτωση τις παραμέτρους username,pass και email:

```
request.getParameter("username") ,
request.getParameter("pass"),
request.getParameter("email")
```

Χρησιμοποιώντας τις παραμέτρους αυτές σχηματίζουμε το string της επερώτησης, δηλαδή το:

```
String query = "INSERT INTO users (username, pass, email)
VALUES ('" + request.getParameter("username") +"',
'"+request.getParameter("pass")+"',
'"+request.getParameter("email")+"')";
```

Κατόπιν εκτελούμε την εισαγωγή :

```
DBUtil.update(query);
```

Τέλος δημιουργούμε τον κατάλληλο κώδικα HTML για να ενημερώνουμε τον χρήστη ότι η εισαγωγή ολοκληρώθηκε.

Admin/auction.jsp

Το αρχείο auction.jsp περιέχει την φόρμα που χρησιμοποιεί ο administrator για την εισαγωγή μιας νέας δημοπρασίας.Αρχικά υπάρχουν τα links που χρησιμοποιήθηκαν για την εισαγωγή του ημερολογίου. Πρόκειται για μία φόρμα html στην οποία γίνεται και χρήση κώδικα java, που περιέχει τα πεδία title, description, start_date, sstart_price, category_id, company_id. Η χρήση της java γίνεται στα δύο τελευταία πεδία,τα οποία είναι dropdown μενού. Για να δημιουργήσουμε το dropdown με τις κατηγορίες κάνουμε υποερώτηση στη βάση και επιλέγουμε όλα τα περιεχόμενα του πίνακα category:

```
String query = "SELECT * FROM category";
```

Με μια while διατρέχουμε όλα τα αποτελέσματα και για κάθε ένα πέρνουμε τα περιεχόμενα των στηλών id και name ώστε να δημιουργήσουμε την αντίστοιχη επιλογή του select:

```
<option value="<%=id%>" ><%=name%></option>
```

Ομοίως δημιουργούμε τις επιλογές για την εταιρεία, δηλαδή επιλέγουμε τα περιεχόμενα του πίνακα company:

```
String query = "SELECT * FROM company";
```

Και δημιουργούμε τις αντίστοιχες επιλογές:

```
<option value="<%=id%>" ><%=name%></option>
```

Τέλος χρησιμοποιούμε κώδικα JavaScript για την εγκατάσταση του ημερολογίου. Το action της φόρμας είναι το αρχείο /Project/AddAuction που βρίσκεται στον default package φάκελο.

AddAuction.java

Αυτό το servlet είναι υπεύθυνο για την εισαγωγή μιας νέας δημοπρασίας. Αρχικά χρησιμοποιούμε τις παραμέτρους που μας έχουν δωθεί από το Admin/users.jsp για να πάρουμε το όνομα της δημοπρασίας, την περιγραφή της, την αρχική της ημερομηνία, την αρχική και την τρέχουσα τιμή της,το id της εταιρίας και το id της κατηγορίας που ανήκει το προϊόν . Στην προκειμένη περίπτωση τις παραμέτρους title, description, start_date, start_price, category_id, company_id:

```
request.getParameter("title"),
request.getParameter("description"),
request.getParameter("start_date"),
request.getParameter("start_price"),
request.getParameter("category_id"),
request.getParameter("company_id")
```

Χρησιμοποιώντας τις παραμέτρους αυτές σχηματίζουμε το string της επερώτησης, δηλαδή το:

```
String query = "INSERT INTO auction (title , description ,
start_date , start_price , category_id , company_id) VALUES
('" + request.getParameter("title") +"',
'"+request.getParameter("description")+"',
'"+request.getParameter("start_date")+"',
'"+request.getParameter("start_price")+"',
'"+request.getParameter("category_id")+"',
'"+request.getParameter("company_id")+"')";
```

Κατόπιν εκτελούμε την εισαγωγή :

```
DBUtil.update(query);
```

Τέλος δημιουργούμε τον κατάλληλο κώδικα HTML για να ενημερώνουμε τον χρήστη ότι η εισαγωγή ολοκληρώθηκε.

ListCategories.java

Σε αυτό το servlet δημιουργούμε την λίστα με τις κατηγορίες που υπάρχουν αποθηκευμένες στο σύστημα. Αρχικά δημιουργούμε έναν πίνακα HTML, κάθε γραμμή του οποίου είναι μία κατηγορία. Έπειτα κατασκευάζουμε το string query με το οποίο επλέγουμε τα περιεχόμενα του πίνακα category και εκτελούμε την επερώτηση.

```
String query = "SELECT * FROM category ORDER BY id";
ResultSet rs = DBUtil.query(query);
```

Τα αποτελέσματα αποθηκεύονται στο ResultSet rs, το οποίο και διατρέχουμε. Για κάθε εγγραφή δημιουργούμε μια νέα γραμμή του πίνακα η οποία περιέχει τρία κελιά. Το πρώτο κελί είναι το id της κατηγορίας:

```
out.println("<td>" + rs.getInt("id") + "</td>");
```

Το επόμενο περιέχει το όνομα της κατηγορίας:

```
out.println("<td>" + rs.getString("name") + "</td>");
```

Το τελευταίο κελί περιέχει ένα link στο servlet DeleteCategory με παράμετρο το id της συγκεκριμένης κατηγορίας, σε περίπτωση που επιθυμούμε να την διαγράψουμε:

```
out.println("<td><a
href=\"http://localhost:8084/Project/DeleteCategory?id=" +
rs.getInt("id") + "\"><img
src=\"/Project/images/delete.png\"></a></td>");
```

CompanyList.java

Σε αυτό το servlet δημιουργούμε την λίστα με τις εταιρίες που υπάρχουν αποθηκευμένες στο σύστημα. Αρχικά δημιουργούμε έναν πίνακα HTML, κάθε γραμμή του οποίου είναι μία εταιρία. Έπειτα κατασκευάζουμε το string query με το οποίο επλέγουμε τα περιεχόμενα του πίνακα company και εκτελούμε την επερώτηση.

```
String query = "SELECT * FROM company ORDER BY id";
ResultSet rs = DBUtil.query(query);
```

Τα αποτελέσματα αποθηκεύονται στο ResultSet rs, το οποίο και διατρέχουμε. Για κάθε εγγραφή δημιουργούμε μια νέα γραμμή του πίνακα η οποία περιέχει τρία κελιά. Το πρώτο κελί είναι το id της εταιρίας:

```
out.println("<td>" + rs.getInt("id") + "</td>");
```

Το επόμενο περιέχει το όνομα της εταιρίας:

```
out.println("<td>" + rs.getString("username") + "</td>");
```

Το τελευταίο κελί περιέχει ένα link στο servlet DeleteCompany με παράμετρο το id της συγκεκριμένης εταιρίας, σε περίπτωση που επιθυμούμε να την διαγράψουμε:

```
out.println("<td><a
href=\"http://localhost:8084/Project/DeleteCompany?id=" +
```

```
rs.getInt("id") + "\"><img
src=\"/Project/images/delete.png\"></a></td>");
```

UsersList.java

Σε αυτό το servlet δημιουργούμε την λίστα με τους χρήστες που υπάρχουν αποθηκευμένοι στο σύστημα. Αρχικά δημιουργούμε έναν πίνακα HTML, κάθε γραμμή του οποίου είναι ένας χρήστης. Έπειτα κατασκευάζουμε το string query με το οποίο επλέγουμε τα περιεχόμενα του πίνακα users και εκτελούμε την επερώτηση.

```
String query = "SELECT * FROM users ORDER BY id";
ResultSet rs = DBUtil.query(query);
```

Τα αποτελέσματα αποθηκεύονται στο ResultSet rs, το οποίο και διατρέχουμε. Για κάθε εγγραφή δημιουργούμε μια νέα γραμμή του πίνακα η οποία περιέχει τρία κελιά. Το πρώτο κελί είναι το id του χρήστη:

```
out.println("<td>" + rs.getInt("id") + "</td>");
```

Το επόμενο περιέχει το όνομα του χρήστη:

```
out.println("<td>" + rs.getString("username") + "</td>");
```

Το τελευταίο κελί περιέχει ένα link στο servlet DeleteUsers με παράμετρο το id του συγκεκριμένου χρήστη, σε περίπτωση που επιθυμούμε να τον διαγράψουμε:

```
out.println("<td><a
href=\"http://localhost:8084/Project/DeleteUsers?id=" +
rs.getInt("id") + "\"><img
src=\"/Project/images/delete.png\"></a></td>");
```

AuctionList.java

Σε αυτό το servlet δημιουργούμε την λίστα με όλες τις δημοπρασίες που υπάρχουν αποθηκευμένες στο σύστημα. Αρχικά δημιουργούμε έναν πίνακα HTML, κάθε γραμμή του οποίου είναι μία δημοπρασία. Έπειτα κατασκευάζουμε το string query με το οποίο επλέγουμε τα περιεχόμενα του πίνακα auction και εκτελούμε την επερώτηση.

```
String query = "SELECT * FROM auction ORDER BY id";
ResultSet rs = DBUtil.query(query);
```

Τα αποτελέσματα αποθηκεύονται στο ResultSet rs, το οποίο και διατρέχουμε. Για κάθε εγγραφή δημιουργούμε μια νέα γραμμή του πίνακα η οποία περιέχει τέσσερα κελιά.

Το πρώτο κελί είναι το id της δημοπρασίας:

```
out.println("<td>" + rs.getInt("id") + "</td>");
```

Το επόμενο περιέχει το όνομα της δημοπρασίας:

```
out.println("<td>" + rs.getString("title") + "</td>");
```

Το τελευταίο κελί περιέχει ένα link στο servlet DeleteAuction με παράμετρο το id της συγκεκριμένης δημοπρασίας, σε περίπτωση που επιθυμούμε να την διαγράψουμε:

```
out.println("<td><a
href=\"http://localhost:8084/Project/DeleteAuction?id=" +
rs.getInt("id") + "\"><img
src=\"/Project/images/delete.png\"></a></td>");
```

Εδώ υπάρχει ένα επιπλέον κελί στον πίνακα όπου με μία if δηλώνουμε αν το προϊόν αποστάλθηκε ή όχι στον πελάτη.

ProductiveAuctions.java

Σε αυτό το servlet δημιουργούμε την λίστα με τις γόνιμες δημοπρασίες που υπάρχουν αποθηκευμένες στο σύστημα. Αρχικά δημιουργούμε έναν πίνακα HTML, κάθε γραμμή του οποίου είναι μία κατοχυρωμένη δημοπρασία. Έπειτα κατασκευάζουμε το string query με το οποίο επλέγουμε τα περιεχόμενα του πίνακα auction και εκτελούμε την επερώτηση.

```
ResultSet rs = DBUtil.query("SELECT * FROM `auction` WHERE
`last_bid_time` IS NOT NULL");
```

Τα αποτελέσματα αποθηκεύονται στο ResultSet rs, το οποίο και διατρέχουμε. Για κάθε εγγραφή δημιουργούμε μια νέα γραμμή του πίνακα η οποία περιέχει πέντε κελιά.

Το πρώτο κελί είναι το id της κατοχυρωμένης δημοπρασίας:

```
out.println("<td>" + rs.getInt("id") + "</td>");
```

Το επόμενο περιέχει το όνομα της κατοχυρωμένης δημοπρασίας:

```
out.println("<td>" + rs.getString("title") + "</td>");
```

Το επόμενο περιέχει την αρχική τιμή της κατοχυρωμένης δημοπρασίας:

```
out.println("<td>" + startPrice + "</td>");
```

Το τέταρτο κελί περιέχει την τρέχουσα τιμή της κατοχυρωμένης δημοπρασίας:

```
out.println("<td>" + currentPrice + "</td>");
```

Στο τελευταίο κελί του πίνακα με μία if δηλώνουμε αν το προϊόν αποστάλθηκε ή όχι στον πελάτη.

UnproductiveAuctions.java

Σε αυτό το servlet δημιουργούμε την λίστα με τις άγονες δημοπρασίες που υπάρχουν αποθηκευμένες στο σύστημα. Αρχικά δημιουργούμε έναν πίνακα HTML, κάθε γραμμή του οποίου είναι μία άγονη δημοπρασία. Έπειτα κατασκευάζουμε το string query με το οποίο επλέγουμε τα περιεχόμενα του πίνακα auction και εκτελούμε την επερώτηση.

```
ResultSet rs = DBUtil.query("SELECT * FROM `auction` WHERE
last_bid_time` IS NULL");
```

Τα αποτελέσματα αποθηκεύονται στο ResultSet rs, το οποίο και διατρέχουμε. Για κάθε εγγραφή δημιουργούμε μια νέα γραμμή του πίνακα η οποία περιέχει τέσσερα κελιά.

Το πρώτο κελί είναι το id της άγονης δημοπρασίας:

```
out.println("<td>" + rs.getInt("id") + "</td>");
```

Το επόμενο περιέχει το όνομα της άγονης δημοπρασίας:

```
out.println("<td>" + rs.getString("title") + "</td>");
```

Το επόμενο περιέχει την αρχική τιμή της άγονης δημοπρασίας:

```
out.println("<td>" + startPrice + "</td>");
```

Τέλος κατασκευάζουμε ένα νέο string query με το οποίο επιλέγουμε το πεδίο maxAmount του πίνακα bid και εκτελούμε την επερώτηση.

```
String query = "SELECT max( amount ) as maxAmount FROM `bid`
WHERE auction_id = " + auctionId;
```

Τα αποτελέσματα αποθηκεύονται στο ResultSet rs2, τον οποίο και διατρέχουμε.Έπειτα με μία if ελέγχουμε αν υπήρξε ή όχι μεγαλύτερη προσφορά και το τοποθετούμε στο τέταρτο καλί του πίνακα.

Στο τέλος εμφανίζουμε το σύνολο των άγονων δημοπρασιών.

Πηγαίος Κώδικας

DButil.java

```
package util;


import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class DBUtil {
    private static Connection con = null;

    public static ResultSet query(String q) throws
ClassNotFoundException, SQLException, SQLException{
        if(con == null){
            Class.forName("com.mysql.jdbc.Driver");
            con =
DriverManager.getConnection("jdbc:mysql://localhost/db2?useUni
code=true&characterEncoding=UTF-8", "root", "");
        }
        Statement stm =
con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_READ_ONLY);


        ResultSet rs = stm.executeQuery(q);

        return rs;
    }

    public static void close() throws SQLException{
        if(con != null)
            con.close();
            con = null;
    }

    public static void update(String q) throws
ClassNotFoundException, SQLException, SQLException{
        if(con != null && !con.isClosed()){
            Class.forName("com.mysql.jdbc.Driver");
            con =
DriverManager.getConnection("jdbc:mysql://localhost/db2?useUni
code=true&characterEncoding=UTF-8", "root", "");
        }
        Statement stm =
con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_READ_ONLY);

        stm.executeUpdate(q);
        con.close();
        con = null;
    }
}
```

Siteutil.java

```
package util;

import java.text.SimpleDateFormat;
import java.util.Date;

/**
 *
```

```
 * @author nancy
 */
public class SiteUtil {

    public static String getAuctionStatus(Date startDate, Date
lastBidDate){
        Date endDate10 = new Date(startDate.getTime() +
10*60*1000); // 10 λεπτά από τώρα
        Date endDate5 = null;
        if(lastBidDate != null){
            endDate5 = new Date(lastBidDate.getTime() +
5*60*1000); // 5 λεπτά από το last bid time
        }

        Date currentDate = new Date();

        if(currentDate.before(startDate)){
            long startMillis = startDate.getTime();
            long currentMillis = currentDate.getTime();
            long interval = startMillis - currentMillis;
            SimpleDateFormat df = null;
            if(interval > 24*60*60*1000){
                df = new SimpleDateFormat("dd μέρες HH ώρες
και mm λεπτά");
            }else if(interval > 60*60*1000){
                df = new SimpleDateFormat("HH:mm");
            }else{
                df = new SimpleDateFormat("mm");
            }

            return "Έναρξη σε " + df.format(interval);
        // Αν δεν έχει γίνει bid λήγει σε 10 λεπτά από την ώρα
έναρξης.
        }else if(lastBidDate == null &&
startDate.before(currentDate) &&
endDate10.after(currentDate)){
            long endMillis = endDate10.getTime();
            long currentMillis = currentDate.getTime();
            long interval = endMillis - currentMillis;
            SimpleDateFormat df = new SimpleDateFormat("m");

            return "Γίνεται άγονη σε " + df.format(interval) +
" λεπτά";
        // Αν έχει γίνει bid από κάποιον λήγει 5 λεπτά μετά
από την ώρα που έγινε το bid // lb < now < 5+lb
        }else if(lastBidDate != null &&
lastBidDate.before(currentDate) &&
endDate5.after(currentDate)){
            long endMillis = endDate5.getTime();
            long currentMillis = currentDate.getTime();
            long interval = endMillis - currentMillis;
            SimpleDateFormat df = new SimpleDateFormat("m");

            return "Κατωχυρώνεται σε " + df.format(interval) +
" λεπτά";
        }else{ // endDate.before(currentDate)
            return "Έχει λήξει";
        }
    }

    public static boolean isAuctionOpen(Date startDate, Date
lastBidDate){
        Date endDate10 = new Date(startDate.getTime() +
10*60*1000); // 10 λεπτά από τώρα
        Date endDate5 = null;
        if(lastBidDate != null){
            endDate5 = new Date(lastBidDate.getTime() +
5*60*1000); // 5 λεπτά από το last bid time
        }
```

```
        Date currentDate = new Date();

        if(currentDate.before(startDate)){
            return true;
        // Αν δεν έχει γίνει bid λήγει σε 10 λεπτά από την ώρα
έναρξης.
        }else if(lastBidDate == null &&
startDate.before(currentDate) &&
endDate10.after(currentDate)){
            return true;
        // Αν έχει γίνει bid από κάποιον λήγει 5 λεπτά μετά
από την ώρα που έγινε το bid // lb < now < 5+lb
        }else if(lastBidDate != null &&
lastBidDate.before(currentDate) &&
endDate5.after(currentDate)){
            return true;
        }else{ // endDate.before(currentDate)
            return false;
        }
    }

    public static boolean hasAuctionOpened(Date startDate){
        Date currentDate = new Date();

        if(currentDate.before(startDate)){
            return false;
        }else{ // endDate.before(currentDate)
            return true;
        }
    }

}
```

Backend

Admin/Index.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Admin</title>
    </head>
    <body>
        <form method ="post" action="index2.jsp">
        <table>
            <tr>
                <td align="center" colspan="2"><h2>Εισάγετε τα
στοιχεία σας</h2></td>
            </tr>
            <tr>
                <td class="text">Όνομα χρήστη:</td>
                <td><input type="text" name="username" ></td>
            </tr>
            <tr>
                <td class="text">Κωδικός:</td>
                <td><input type="password" name="pass" ></td>
            </tr>
            <tr>
                <td align="center" colspan="2"><input
type="submit" value="Είσοδος" ></td>
```

```
            </tr>
        </table>
        </form>
    </body>
</html>
```

Admin/Index2.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.text.SimpleDateFormat"%>
<%@page import="java.util.Date"%>
<%@page import="util.DBUtil"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Admin</title>
    </head>
    <body>
        <table>

        <%
        if(request.getParameter("username").equals("admin") &&
request.getParameter("pass").equals("lala")){
        %>
            <tr>
                <td valign="top">
                    <ul>
                        <li><a href="category.jsp"
target="PAGES">Add a category</a></li>
                        <li><a href="/Project/ListCategories"
target="PAGES">Show category list</a></li>
                        <li><a href="users.jsp"
target="PAGES">Add a user</a></li>
                        <li><a href="/Project/UsersList"
target="PAGES">Show users list</a></li>
                        <li><a href="company.jsp"
target="PAGES">Add a company</a></li>
                        <li><a href="/Project/CompanyList"
target="PAGES">Show company list</a></li>
                        <li><a href="auction.jsp"
target="PAGES">Add an auction</a></li>
                        <li><a href="/Project/AuctionList"
target="PAGES">Show auction list</a></li>
                        <li><a
href="/Project/UnproductiveAuctions" target="PAGES">Show
unproductive auction list</a></li>
                        <li><a
href="/Project/ProductiveAuctions" target="PAGES">Show
productive auction list</a></li>
                    </ul>
        <%
            ResultSet rs = DBUtil.query("SELECT * FROM
`auction`");

            int count = 0;
            long sum = 0;
            while(rs.next()){
                int id = rs.getInt("id");
                Date startDate =
rs.getTimestamp("start_date");
```

```
                Date lastBidDate =
rs.getTimestamp("last_bid_time");

                long current = 0;

                count++;
                // An to lastBidDate einai null h dhmoprasia
htan agonh, opote
                // krathse 10 lepta
                if(lastBidDate == null){
                    current = 10*60*1000;
                }else{
                    Date endDate = new Date(
lastBidDate.getTime() + 5*60*1000 );
                    current = endDate.getTime() -
startDate.getTime();
                }
                sum += current;
            }

            if(count > 0 ){
                long avg = sum/count;
                Date avgDate = new Date((long)avg);
                SimpleDateFormat df = new
SimpleDateFormat("m:ss");


                out.println("<p>Μέση διάρκεια <b>όλων</b> των
δημοπρασιών: " + df.format(avgDate) + "</p>");
            }

        %>
                </td>
                <td>
                    <iframe width="500" height="700"
name="PAGES" frameborder="0"></iframe>
                </td>
            </tr>
        <table>

       <%
       }else{
            out.println("Εσφαλμένα στοιχεία.");
       }
       %>
    </body>
</html>
```

Admin/Category.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Δημιουργία νέας κατηγορίας</title>
    </head>
    <body>
        <h1>Δημιουργία κατηγορίας</h1>
        <form action="/Project/AddCategory" method="get">
            <table>
                <tr>
                    <td>Ονομα κατηγορίας</td>
```

```
                    <td><input type="text"
name="categoryName"/></td>
                </tr>

                <tr>

                    <td align="center" colspan="2"><input
type="submit" value="Εισαγωγή"/></td>
                </tr>
            </table>
        </form>

    </body>
</html>
```

AddCategory.java

```
import util.DBUtil;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class AddCategory extends HttpServlet {

    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        try {
            request.setCharacterEncoding("utf-8");
            String query = "INSERT INTO category (name) VALUES
('" + request.getParameter("categoryName") +"')";
            DBUtil.update(query);

            response.setContentType("text/html;charset=UTF-
8");
            PrintWriter out = response.getWriter();
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Καταχώρηση
κατηγορίας</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Η καταχώρηση έγινε με
επιτυχία.</h1>");
            out.println("</body>");
            out.println("</html>");

        }catch(Exception e){
            e.printStackTrace();

        } finally {
//            out.close();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
```

```
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

Admin/Company.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Εισαγωγή νέας εταιρίας</title>
    </head>
    <body>
        <h1>Εισαγωγή εταιρίας</h1>
         <form action="/Project/AddCompany" method="post">
            <table>
                <tr>
                    <td>Ονομα εταιρείας</td>
                    <td><input type="text"
name="username"/></td>
                </tr>
                <tr>
                    <td>Κωδικός</td>
                    <td><input type="password"
name="pass"/></td>
                </tr>
                <tr>
                    <td>Email</td>
                    <td><input type="text" name="email"
                      /></td>
                </tr>
                <tr>
                    <td align="center" colspan="2"><input
type="submit" value="Εισαγωγή"/></td>
                </tr>
```

```
            </table>
        </form>

    </body>
</html>
```

AddCompany.java

```
import util.DBUtil;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
 *
 * @author nancy
 */
public class AddCompany extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {

        try {
            request.setCharacterEncoding("utf-8");
            String query = "INSERT INTO company (username,
pass, email) VALUES ('" + request.getParameter("username")
+"', '"+request.getParameter("pass")+"',
'"+request.getParameter("email")+"')";
            DBUtil.update(query);

            response.setContentType("text/html;charset=UTF-
8");
        PrintWriter out = response.getWriter();

            out.println("<html>");
            out.println("<head>");
            out.println("<title>Καταχώρηση εταιρίας</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Η καταχώρηση έγινε με
επιτυχία.</h1>");
            out.println("</body>");
            out.println("</html>");

             }catch(Exception e){
            e.printStackTrace();

        } finally {
           // out.close();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
```

```
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

Admin/users.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Εισαγωγή νέου χρήστη</title>
    </head>
    <body>
        <h1>Εισαγωγή χρήστη</h1>
        <form action="/Project/AddUser" method="post">
            <table>
                <tr>
                    <td>Ονομα χρήστη</td>
                    <td><input type="text" name="username"
/></td>
                </tr>
                <tr>
                    <td>Κωδικός</td>
                    <td><input type="password" name="pass"
/></td>
                </tr>
                <tr>
                    <td>Email</td>
```

```
                    <td><input type="text" name="email"
/></td>
                </tr>
                <tr>
                    <td align="center" colspan="2"><input
type="submit" value="Εισαγωγή"/></td>
                </tr>
            </table>
        </form>

    </body>
</html>
```

AddUser.java

```
import util.DBUtil;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 *
 * @author nancy
 */
public class AddUser extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        try {
            request.setCharacterEncoding("utf-8");
            String query = "INSERT INTO users (username, pass,
email) VALUES ('" + request.getParameter("username") +"',
'"+request.getParameter("pass")+"',
'"+request.getParameter("email")+"')";
            DBUtil.update(query);

            response.setContentType("text/html;charset=UTF-
8");
            PrintWriter out = response.getWriter();
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Καταχώριση χρήστη</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Η καταχώρηση έγινε με
επιτυχία.</h1>");
            out.println("</body>");
            out.println("</html>");


        }catch(Exception e){
            e.printStackTrace();
```

```
            } finally {
//                out.close();
            }
        }

        // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
        /**
         * Handles the HTTP <code>GET</code> method.
         * @param request servlet request
         * @param response servlet response
         * @throws ServletException if a servlet-specific error
occurs
         * @throws IOException if an I/O error occurs
         */
        @Override
        protected void doGet(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
            processRequest(request, response);
        }

        /**
         * Handles the HTTP <code>POST</code> method.
         * @param request servlet request
         * @param response servlet response
         * @throws ServletException if a servlet-specific error
occurs
         * @throws IOException if an I/O error occurs
         */
        @Override
        protected void doPost(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
            processRequest(request, response);
        }

        /**
         * Returns a short description of the servlet.
         * @return a String containing servlet description
         */
        @Override
        public String getServletInfo() {
            return "Short description";
        }// </editor-fold>

}
```

Admin/auction.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="util.DBUtil"%>
<%@page import="java.io.File"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Δημιουργία δημοπρασίας</title>
        <style type="text/css">@import
url(/Project/jscalendar/calendar-win2k-1.css);</style>
        <script type="text/javascript"
src="/Project/jscalendar/calendar.js"></script>
```

```
        <script type="text/javascript"
src="/Project/jscalendar/lang/calendar-en.js"></script>
        <script type="text/javascript"
src="/Project/jscalendar/calendar-setup.js"></script>

    </head>
    <body>
        <h1>Δημιουργία δημοπρασίας</h1>
         <form action="/Project/AddAuction" method="post">
            <table>
                <tr>
                    <td>Ονομα δημοπράσιας</td>
                    <td><input type="text" name="title"/></td>
                </tr>
                <tr>
                    <td>Εικόνα</td>
                    <td>
                        <select name="image">
<%
    File imagesFolder = new
File("../webapps/ROOT/product_images");
    String images[] = imagesFolder.list();

    for(String image : images){
        %>
                <option value="<%=image%>" ><%=image%></option>
<%  }   %>

                        </select>
                    </td>
                </tr>
                <tr>
                    <td>Περιγραφή</td>
                    <td><textarea rows="4" cols="20"
name="description" ></textarea>
                </tr>
                <tr>
                    <td>Ημερομηνία έναρξης</td>
                    <td><input type="text" id="start_date"
name="start_date"/><button type="button"
id="calendar">...</button>
</td>
                </tr>
                <tr>
                    <td>Τιμή</td>
                    <td><input type="text"
name="start_price"/></td>
                </tr>
                <tr>
                    <td>Κατηγορία</td>
                    <td>
                        <select name="category_id">
<%
    String query = "SELECT * FROM category";
    ResultSet rs = DBUtil.query(query);

    while(rs.next()){
        int id = rs.getInt("id");
        String name = rs.getString("name");%>
                <option value="<%=id%>" ><%=name%></option>
<%    }     %>

                        </select>
                    </td>
                </tr>
                <tr>
                    <td>Εταιρεία</td>
                    <td>
                        <select name="company_id">
```

```
<%
    query = "SELECT * FROM company";
    rs = DBUtil.query(query);

    while(rs.next()){
        int id = rs.getInt("id");
        String name = rs.getString("username");%>
                <option value="<%=id%>" ><%=name%></option>
<%    }     %>

                        </select>


                    </td>
                </tr>
                <tr>
                    <td align="center" colspan="2"><input
type="submit" value="Εισαγωγή"/></td>
                </tr>
            </table>
        </form>
        <script type="text/javascript">
          Calendar.setup(
            {
              inputField  : "start_date",         // ID of the
input field
              ifFormat    : "%Y-%m-%d %H:%M:%S",    // the
date format
              button      : "calendar",       // ID of the
button
              showsTime   : true
            }
          );
        </script>

    </body>
</html>
```

AddAuction.java

```
import util.DBUtil;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 *
 * @author nancy
 */
public class AddAuction extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        try {
```

```
            request.setCharacterEncoding("utf-8");
            String query = "INSERT INTO auction (title ,
description , start_date , start_price , current_price,
category_id , company_id, image) VALUES ('" +
request.getParameter("title") +"',
'"+request.getParameter("description")+"',
'"+request.getParameter("start_date")+"',
'"+request.getParameter("start_price")+"',
'"+request.getParameter("start_price")+"',
'"+request.getParameter("category_id")+"',
'"+request.getParameter("company_id")+"',
'"+request.getParameter("image")+"')";
            DBUtil.update(query);

            response.setContentType("text/html;charset=UTF-
8");
            PrintWriter out = response.getWriter();

            out.println("<html>");
            out.println("<head>");
            out.println("<title>Καταχώρηση
δημοπρασίας</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Η καταχώρηση έγινε με
επιτυχία.</h1>");
            out.println("</body>");
            out.println("</html>");

        }catch(Exception e){
            e.printStackTrace();

        } finally {
//            out.close();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }
```

```
    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

ListCategories.java

```
import util.DBUtil;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.ResultSet;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 *
 * @author nancy
 */
public class ListCategories extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            response.setContentType("text/html;charset=UTF-
8");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Λίστα κατηγοριών</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<table border=\"1\">");

            String query = "SELECT * FROM category ORDER BY
id";
            ResultSet rs = DBUtil.query(query);

            while(rs.next()){
                out.println("<tr>"); // Αρχή γραμμής

                out.println("<td>" + rs.getInt("id") +
"</td>");
                out.println("<td>" + rs.getString("name") +
"</td>");
                out.println("<td><a
href=\"http://localhost:8084/Project/DeleteCategory?id=" +
rs.getInt("id") + "\"><img
src=\"/Project/images/delete.png\"></a></td>");
```

```
                out.println("</tr>");
            }



            out.println("</table>");
            out.println("</body>");
            out.println("</html>");

        } catch(Exception e){
            e.printStackTrace();

         } finally {
            out.close();
         }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

CompanyList.java

```
import util.DBUtil;
import java.io.IOException;
import java.io.PrintWriter;
```

```
import java.sql.ResultSet;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
 *
 * @author nancy
 */
public class CompanyList extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            String query = "SELECT * FROM company ORDER BY
id";
            ResultSet rs = DBUtil.query(query);

            response.setContentType("text/html;charset=UTF-
8");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Λίστα εταιριών</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<table border=\"1\">");


            while(rs.next()){
                out.println("<tr>");

                out.println("<td>" + rs.getInt("id") +
"</td>");
                out.println("<td>" + rs.getString("username")
+ "</td>");
                out.println("<td><a
href=\"http://localhost:8084/Project/DeleteCompany?id=" +
rs.getInt("id") + "\"><img
src=\"/Project/images/delete.png\"></a></td>");


                out.println("</tr>");
            }



            out.println("</table>");
            out.println("</body>");
            out.println("</html>");

       } catch(Exception e){
            e.printStackTrace();

        } finally {
            out.close();
        }
    }
```

```
    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

UsersList.java

```
import util.DBUtil;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
 *
 * @author nancy
 */
public class UsersList extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
```

```
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            String query = "SELECT * FROM users";
            ResultSet rs = DBUtil.query(query);

            response.setContentType("text/html;charset=UTF-
8");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Λίστα χρηστών</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<table border=\"1\">");

            while(rs.next()){
                out.println("<tr>");

                out.println("<td>" + rs.getInt("id") +
"</td>");
                out.println("<td>" + rs.getString("username")
+ "</td>");
                out.println("<td><a
href=\"http://localhost:8084/Project/DeleteUsers?id=" +
rs.getInt("id") + "\"><img
src=\"/Project/images/delete.png\"></a></td>");


                out.println("</tr>");
            }
            DBUtil.close();

            out.println("</table>");
            out.println("</body>");
            out.println("</html>");


            } catch(Exception e){
            e.printStackTrace();

        } finally {
            out.close();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }
```

```
    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

AuctionList.java

```
import util.DBUtil;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.ResultSet;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
 *
 * @author nancy
 */
public class AuctionList extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            String query = "SELECT * FROM auction";
            ResultSet rs = DBUtil.query(query);

            response.setContentType("text/html;charset=UTF-
8");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Λίστα δημοπρασιών</title>");
            out.println("</head>");
            out.println("<body>");
```

```
            out.println("<table border=\"1\">");

            out.println("<tr>");
            out.println("<td>id</td>");
            out.println("<td>Όνομα</td>");
            out.println("<td>Κατάσταση αποστολής</td>");
            out.println("<td> </td>");
            out.println("</tr>");

            while(rs.next()){
                out.println("<tr>");

                out.println("<td>" + rs.getInt("id") +
"</td>");
                out.println("<td>" + rs.getString("title") +
"</td>");

                if(!rs.getBoolean("shipped")){
                    out.println("<td
align=\"center\">Απεστάλει");
                }else{
                    out.println("<td
align=\"center\">Εκκρεμεί");
                }

                out.println("<td><a
href=\"http://localhost:8084/Project/DeleteAuction?id=" +
rs.getInt("id") + "\"><img
src=\"/Project/images/delete.png\"></a></td>");

                out.println("</tr>");
            }



            out.println("</table>");
            out.println("</body>");
            out.println("</html>");

        } catch(Exception e){
            e.printStackTrace();

         } finally {
            out.close();
         }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
```

```
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

Deletecategory.java

```
import util.DBUtil;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 *
 * @author nancy
 */
public class DeleteCategory extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            String query = "DELETE FROM category WHERE id=" +
request.getParameter("id");
            DBUtil.update(query);

            response.setContentType("text/html;charset=UTF-
8");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Διαγραφή κατηγορίας</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Η διαγραφή έγινε με
επιτυχία.</h1>");
            out.println("</body>");
```

```
            out.println("</html>");

        }catch(Exception e){
            e.printStackTrace();

        } finally {
            out.close();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

DeleteCompany.java

```
import util.DBUtil;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
```

```
 *
 * @author nancy
 */
public class DeleteCompany extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            String query = "DELETE FROM company WHERE id=" +
request.getParameter("id");
            DBUtil.update(query);

            response.setContentType("text/html;charset=UTF-
8");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Διαγραφή εταιρίας</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Η διαγραφή έγινε με
επιτυχία.</h1>");
            out.println("</body>");
            out.println("</html>");

        }catch(Exception e){
            e.printStackTrace();

        } finally {
            out.close();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
```

```
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

DeleteUsers.java

```
import util.DBUtil;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
 *
 * @author nancy
 */
public class DeleteUsers extends HttpServlet {

  protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            String query = "DELETE FROM users WHERE id=" +
request.getParameter("id");
            DBUtil.update(query);

            response.setContentType("text/html;charset=UTF-
8");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Διαγραφή χρήστη</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Η διαγραφή έγινε με
επιτυχία.</h1>");
            out.println("</body>");
            out.println("</html>");

        }catch(Exception e){
            e.printStackTrace();

        } finally {
            out.close();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
```

```
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

DeleteAuction.java

```
import util.DBUtil;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
 *
 * @author nancy
 */
public class DeleteAuction extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
```

```
        PrintWriter out = response.getWriter();
        try {
            String query = "DELETE FROM auction WHERE id=" +
request.getParameter("id");
            DBUtil.update(query);

            response.setContentType("text/html;charset=UTF-
8");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Διαγραφή
δημοπρασίας</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Η διαγραφή έγινε με
επιτυχία.</h1>");
            out.println("</body>");
            out.println("</html>");

        }catch(Exception e){
            e.printStackTrace();

        } finally {
            out.close();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

ProductiveAuctions.java

```
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import util.DBUtil;

/**
 *
 * @author nancy
 */
public class ProductiveAuctions extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Λίστα κατοχυρωμένων
δημοπρασιών</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Λίστα κατοχυρωμένων
δημοπρασιών</h1>");
            out.println("<table border=\"1\">");

            out.println("<tr>");
            out.println("<td>A/A</td>");
            out.println("<td>Τίτλος</td>");
            out.println("<td>Αρχική τιμή</td>");
            out.println("<td>Τιμή κατοχύρωσης</td>");
            out.println("<td>Έχει αποσταλεί</td>");
            out.println("</tr>");

            ResultSet rs = DBUtil.query("SELECT * FROM
`auction` WHERE `last_bid_time` IS NOT NULL");

            int i=1;
            while(rs.next()){
                int auctionId = rs.getInt("id");
                String title = rs.getString("title");
                double startPrice =
rs.getDouble("start_price");
                double currentPrice =
rs.getDouble("current_price");
                boolean shipped = rs.getBoolean("shipped");
```

```
                out.println("<tr>");
                out.println("<td>" + i +"</td>");
                out.println("<td>" + title + "</td>");
                out.println("<td>" + startPrice + "</td>");
                out.println("<td>" + currentPrice + "</td>");
                if(shipped){
                    out.println("<td
align=\"center\">Ναι</td>");
                }else{
                    out.println("<td
align=\"center\">Οχι</td>");
                }
                out.println("</tr>");

                i++;
            }
            out.println("</table>");
            out.println("</body>");
            out.println("</html>");

        } catch(ClassNotFoundException e) {
            e.printStackTrace();
        }catch(SQLException e){
            e.printStackTrace();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

UnproductiveAuctions.java

```
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import util.DBUtil;

/**
 *
 * @author nancy
 */
public class UnproductiveAuctions extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Λίστα άγονων
δημοπρασιών</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Λίστα άγονων δημοπρασιών</h1>");
            out.println("<table border=\"1\">");

            out.println("<tr>");
            out.println("<td>A/A</td>");
            out.println("<td>Τίτλος</td>");
            out.println("<td>Αρχική τιμή</td>");
            out.println("<td>Μεγαλύτερη προσφορά</td>");
            out.println("</tr>");

            ResultSet rs = DBUtil.query("SELECT * FROM
`auction` WHERE `last_bid_time` IS NULL");

            int i=1;
            while(rs.next()){
                int auctionId = rs.getInt("id");
                String title = rs.getString("title");
                double startPrice =
rs.getDouble("start_price");

                String query = "SELECT max( amount ) as
maxAmount FROM `bid` WHERE auction_id = " + auctionId;
                ResultSet rs2 = DBUtil.query(query);

                rs2.next();
                double maxAmount = rs2.getDouble("maxAmount");

                out.println("<tr>");
                out.println("<td>" + i +"</td>");
```

```
                out.println("<td>" + title + "</td>");
                out.println("<td>" + startPrice + "</td>");
                if( maxAmount == 0){
                    out.println("<td>Δεν υπήρξε
προσφορά</td>");
                }else{
                    out.println("<td>" + maxAmount + "</td>");
                }
                out.println("</tr>");

                i++;
            }

            out.println("</table>");
            out.println("<p>Σύνολο άγονων δημοπρασιών: " + (i-
1) + "</p>");
            out.println("</body>");
            out.println("</html>");

        } catch(ClassNotFoundException e) {
            e.printStackTrace();
        }catch(SQLException e){
            e.printStackTrace();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

Frontend

Frontend/header.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<img src="images/top-left.png" class="logoImage">
<img src="images/logo-title.png" width="80%">
<div class="line"></div>
```

Frontend/left_menu.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="util.DBUtil"%>

<table>
    <tr>
        <td class="text">Αναζήτηση προϊόντος:</td>
    </tr>
    <tr>
        <td>
            <form method ="post" action="searchProduct.jsp"
onsubmit="return validate1()">
                <input type="text" name="searchString"
id="searchString"><input type="submit" value="Go" >
            </form>
                    <script type="text/javascript">
                        function validate1(){

if(document.getElementById("searchString").value == null
                            ||
document.getElementById("searchString").value == ""){
                                alert("Παρακαλώ συμπληρώστε το
όνομα του προϊόντος που θέλετε να αναζητήσετε.");
                                return false;
                            }
                            return true;
                        }
                    </script>

        </td>
    </tr>
    <tr>
    <tr>
        <td class="text">Αναζήτηση ανά εταιρεία:</td>
    </tr>
        <td>
            <form method ="post" action="searchByCompany.jsp">
                <select name="company_id">
                        <option value="" > </option>
                <%
                    String query2 = "SELECT * FROM company";
                    ResultSet rs2 = DBUtil.query(query2);

                    while(rs2.next()){
                        int id = rs2.getInt("id");
                        String name =
rs2.getString("username");%>
                                <option value="<%=id%>"
><%=name%></option>
                <%    }     %>
                </select><input type="submit" value="Go" >
            </form>

        </td>
```

```
    </tr>
</table>


<ul class="left-menu">

<%
    String query1 = "SELECT * FROM category ORDER BY name";
    ResultSet rs1 = DBUtil.query(query1);

    while(rs1.next()){
        int id = rs1.getInt("id");
        String name = rs1.getString("name");%>
        <li><a href="showCategory.jsp?id=<%=id
%>"><%=name%></a></li>
<%    }     %>
</ul>
<% if(request.getSession().getAttribute("username") != null){
%>
<div style="text-align:center">
 <p class="text">Καλωσόρισες
<%=request.getSession().getAttribute("username") %></p>
 <a href="Logout">Έξοδος</a>
 </div>
 <br>
<% }else{ %>
<form method ="post" action="Login"  onsubmit="return
validate()">
<table>
    <tr>
        <td class="text">Όνομα χρήστη:</td>
    </tr>
    <tr>
        <td><input type="text" name="username"
id="username"></td>
    </tr>
    <tr>
        <td class="text">Κωδικός:</td>
    </tr>
    <tr>
        <td><input type="password" name="pass" id="pass"></td>
    </tr>
    <tr>
        <td align="center"><input type="submit"
value="Είσοδος" ></td>
    </tr>
    <tr>
        <td align="center"><a class="text"
href="register.jsp">Εγγραφή νέου χρήστη</a></td>
    </tr>

</table>
</form>
<script type="text/javascript">
    function validate(){
        if(document.getElementById("username").value == null
         || document.getElementById("username").value == ""
         || document.getElementById("pass").value == null
         || document.getElementById("pass").value == ""){
            alert("Παρακαλώ συμπληρώστε τα στοιχεία σας.");
            return false;
         }
         return true;
    }
</script>

<% }%>
```

searchProduct.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.util.Date"%>
<%@page import="java.util.GregorianCalendar"%>
<%@page import="util.DBUtil"%>
<%@page import="util.SiteUtil"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <link href="frontend.css" type="text/css"
rel="stylesheet"/>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
        <title>Στο σφυρί</title>
    </head>
    <body>
        <center>
        <table border="0">
            <tr>
                <td colspan="2" width="760">
                <%@include file="frontend/header.jsp" %>
                </td>
            </tr>
            <tr>
                <td width="200">
                    <%@include file="frontend/left_menu.jsp"
%>
                </td>
                <td valign="top">

                    <table width="100%">

                    <tr>
                        <td colspan="3"><h2>Αποτελέσματα
αναζήτησης για: <%=request.getParameter("searchString")
%></h2></td>
                    </tr>
                    <%
                    String query = "SELECT * FROM auction
WHERE title LIKE '%" + request.getParameter("searchString") +
"%'";
                    ResultSet rs = DBUtil.query(query);

                    while(rs.next()){
                        int id = rs.getInt("id");
                        String title = rs.getString("title");
                        Date startDate =
rs.getTimestamp("start_date");
                        Date lastBidTime =
rs.getTimestamp("last_bid_time");
                        double startPrice =
rs.getDouble("start_price");
                        double currentPrice =
rs.getDouble("current_price");
                        String image = rs.getString("image");

                        double price = 0;
                        if(currentPrice == 0)
                            price = startPrice;
                        else
                            price = currentPrice;
                    %>
                    <tr class="auction">
```

```
                        <td><a href="showAuction.jsp?id=<%=id
%>"><img src="/product_images/<%=image%>" width="50"
height="50"></a></td>
                        <td><a href="showAuction.jsp?id=<%=id
%>"><%=title %></a></td>
                        <td><%=price %> &euro;</td>
                        <td class="auction-
status"><%=SiteUtil.getAuctionStatus(startDate, lastBidTime)
%></td>
                    </tr>
                    <% } %>

                    </table>
                </td>
            </tr>
            <tr>
                <td colspan="2" width="760">
                <%@include file="frontend/footer.jsp" %>
                </td>
            </tr>
        </table>
        </center>
    </body>
</html>
```

searchByCompany.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.util.Date"%>
<%@page import="java.util.GregorianCalendar"%>
<%@page import="util.DBUtil"%>
<%@page import="util.SiteUtil"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <link href="frontend.css" type="text/css"
rel="stylesheet"/>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
        <title>Στο σφυρί</title>
    </head>
    <body>
        <center>
        <table border="0">
            <tr>
                <td colspan="2" width="760">
                <%@include file="frontend/header.jsp" %>
                </td>
            </tr>
            <tr>
                <td width="200">
                    <%@include file="frontend/left_menu.jsp"
%>
                </td>
                <td valign="top">

                    <table width="100%">

                    <tr>
                        <td colspan="3"><h2>Αποτελέσματα
αναζήτησης εταιρείας</h2></td>
                    </tr>
                    <%
```

```
                    String query = "SELECT * FROM auction
WHERE company_id='" + request.getParameter("company_id") +
"'";
                    ResultSet rs = DBUtil.query(query);

                    while(rs.next()){
                        int id = rs.getInt("id");
                        String title = rs.getString("title");
                        String image = rs.getString("image");
                        Date startDate =
rs.getTimestamp("start_date");
                        Date lastBidTime =
rs.getTimestamp("last_bid_time");
                        double startPrice =
rs.getDouble("start_price");
                        double currentPrice =
rs.getDouble("current_price");

                        double price = 0;
                        if(currentPrice == 0)
                            price = startPrice;
                        else
                            price = currentPrice;
                    %>
                    <tr class="auction">
                        <td><a href="showAuction.jsp?id=<%=id
%>"><img src="/product_images/<%=image%>" width="50"
height="50"></a></td>
                        <td><a href="showAuction.jsp?id=<%=id
%>"><%=title %></a></td>
                        <td><%=price %> &euro;</td>
                        <td class="auction-
status"><%=SiteUtil.getAuctionStatus(startDate, lastBidTime)
%></td>
                    </tr>
                    <% } %>
                    </table>
                </td>
            </tr>
            <tr>
                <td colspan="2" width="760">
                <%@include file="frontend/footer.jsp" %>
                </td>
            </tr>

        </table>
        </center>
    </body>
</html>
```

showCategory.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.util.Date"%>
<%@page import="java.util.GregorianCalendar"%>
<%@page import="util.DBUtil"%>
<%@page import="util.SiteUtil"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <link href="frontend.css" type="text/css"
rel="stylesheet"/>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
```

```
        <title>Στο σφυρί</title>
    </head>
    <body>
        <center>
        <table border="0">
            <tr>
                <td colspan="2" width="760">
                <%@include file="frontend/header.jsp" %>
                </td>
            </tr>
            <tr>
                <td width="200">
                    <%@include file="frontend/left_menu.jsp"
%>
                </td>
                <td valign="top">
                    <%
                    String query = "SELECT * FROM category
WHERE id=" + request.getParameter("id");
                    ResultSet rs = DBUtil.query(query);
                    rs.next();
                    String catName = rs.getString("name");
                    %>
                    <div class="breadcrumb">
                            <a
href="/Project/index.jsp">Αρχική σελίδα</a> &gt;
                            <%=catName %>
                    </div>

                    <table width="100%">

                    <tr>
                        <td colspan="3"><h2><%=catName
%></h2></td>
                    </tr>
                    <%
                    query = "SELECT * FROM auction WHERE
category_id=" + request.getParameter("id");
                    rs = DBUtil.query(query);

                    while(rs.next()){
                        int id = rs.getInt("id");
                        String title = rs.getString("title");
                        String image = rs.getString("image");
                        Date startDate =
rs.getTimestamp("start_date");
                        Date lastBidTime =
rs.getTimestamp("last_bid_time");
                        double startPrice =
rs.getDouble("start_price");
                        double currentPrice =
rs.getDouble("current_price");

                        double price = 0;
                        if(currentPrice == 0)
                            price = startPrice;
                        else
                            price = currentPrice;
                    %>
                    <tr class="auction">
                        <td><a href="showAuction.jsp?id=<%=id
%>"><img src="/product_images/<%=image%>" width="50"
height="50"></a></td>
                        <td><a href="showAuction.jsp?id=<%=id
%>"><%=title %></a></td>
                        <td><%=price %> &euro;</td>
                        <td class="auction-
status"><%=SiteUtil.getAuctionStatus(startDate, lastBidTime)
%></td>
```

```
                    </tr>
                    <% } %>

                    </table>
                </td>
            </tr>
            <tr>
                <td colspan="2" width="760">
                <%@include file="frontend/footer.jsp" %>
                </td>
            </tr>

        </table>
        </center>
    </body>
</html>
```

showAuction.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.util.Date"%>
<%@page import="java.text.SimpleDateFormat"%>
<%@page import="util.DBUtil"%>
<%@page import="util.SiteUtil"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <link href="frontend.css" type="text/css"
rel="stylesheet"/>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
        <title>Στο σφυρί</title>
    </head>
    <body>
        <center>
        <table border="0">
            <tr>
                <td colspan="2" width="760">
                <%@include file="frontend/header.jsp" %>
                </td>
            </tr>
            <tr>
                <td width="200" valign="top">
                    <br><br>
                    <%@include file="frontend/left_menu.jsp"
%>
                </td>
                <td valign="top" width="600">
                    <%
                    String query = "SELECT auction.id,
description, name, image, title, last_bid_time, start_price,
current_price, category_id, start_date, company_id,
company.username as company_name, user_id FROM auction,
company, category WHERE auction.company_id=company.id AND
auction.category_id=category.id AND auction.id=" +
request.getParameter("id");
                    ResultSet rs = DBUtil.query(query);
                    rs.next();
                    int auctionId = rs.getInt("id");
                    String title = rs.getString("title");
                    String image = rs.getString("image");
                    String description =
rs.getString("description");
```

```
                    Date lastBidTime =
rs.getTimestamp("last_bid_time");
                    double startPrice =
rs.getDouble("start_price");
                    double currentPrice =
rs.getDouble("current_price");
                    int userId = rs.getInt("user_id");
                    String companyName =
rs.getString("company_name");
                    int categoryId = rs.getInt("category_id");
                    String categoryName =
rs.getString("name");
                    Date startDate =
rs.getTimestamp("start_date");
                    String lastBidder = null;

                    if(userId != 0){
                        query = "SELECT id, username FROM
users WHERE id=" + userId;
                        rs = DBUtil.query(query);
                        rs.next();
                        lastBidder = rs.getString("username");
                    }
                    %>
                    <div class="breadcrumb">
                        <a
href="/Project/index.jsp">Αρχική σελίδα</a> &gt;
                        <a
href="showCategory.jsp?id=<%=categoryId %>"><%=categoryName
%></a> &gt;
                        <%=title %>
                    </div>

                    <table width="80%">
                    <tr>
                        <td colspan="2"><h2><%=title
%></h2></td>
                    </tr>
                    <tr>
                        <td colspan="2"><img class="product-
image" src="/product_images/<%=image%>" ></td>
                    </tr>
                    <tr>
                        <td>Κατηγορία:</td>
                        <td><a
href="showCategory.jsp?id=<%=categoryId %>"><%=categoryName
%></a></td>
                    </tr>
                    <tr>
                        <td>Κατάσταση:</td>

<td><%=SiteUtil.getAuctionStatus(startDate, lastBidTime)
%></td>
                    </tr>
                    <tr>
                        <td>Τρέχουσα τιμή:</td>
                        <td>
                        <%

if(SiteUtil.hasAuctionOpened(startDate) && lastBidder !=
null){
                            out.print(currentPrice + "&euro;
από " + lastBidder);
                        }else{
                            out.print(startPrice + " &euro;");
                        }
                        %>
                        </td>
                    </tr>
```

```
                    <% if(SiteUtil.isAuctionOpen(startDate,
lastBidTime) && request.getSession().getAttribute("userId") !=
null){ %>
                    <tr>
                        <td>Μεγαλύτερη προσφορά:</td>
                        <td>
                            <form action="PlaceBid"
method="post">
                                <input type="text"
name="ammount" size="5">
                                <input type="hidden"
name="auctionId" value="<%=auctionId %>">
                                <input type="hidden"
name="currentPrice" value="<%=currentPrice %>">
                                <input type="submit"
value="Bid">
                            </form>
                        </td>
                    </tr>
                    <% }%>
                    <tr>
                        <td>Εταιρεία:</td>
                        <td><%=companyName %></td>
                    </tr>
                    <tr>
                        <td valign="top">Περιγραφή:</td>
                        <td><%=description %></td>
                    </tr>

                    </table>
                </td>
            </tr>
            <tr>
                <td colspan="2" width="760">
                <%@include file="frontend/footer.jsp" %>
                </td>
            </tr>

        </table>
        <script
type="text/javascript">setTimeout("window.location.reload()",6
0*1000); </script>
        </center>
    </body>
</html>
```

Frontend/footer.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>

<div class="footer" width="100%">&copy;2009 Διπλωματική
εργασία Μαρίας Φουνικάκου και Νάνσυ Τυμπαλαίξη</div>
```

Index.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">
<%@page import="java.sql.ResultSet"%>
<%@page import="java.util.Date"%>
<%@page import="java.util.GregorianCalendar"%>
<%@page import="util.DBUtil"%>
<%@page import="util.SiteUtil"%>

<html>
    <head>
```

```
        <link href="frontend.css" type="text/css"
rel="stylesheet"/>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
        <title>Στο σφυρί</title>
    </head>
    <body>
        <div class="bodyContainer">
            <center>
            <table>
                <tr>
                    <td colspan="2" width="760">
                    <%@include file="frontend/header.jsp" %>
                    </td>
                </tr>
                <tr>
                    <td width="200">
                        <%@include
file="frontend/left_menu.jsp" %>
                    </td>
                    <td valign="top">
                    <p style="color:#FFFFFF; ">Καλωσορίσατε
στην σελίδα δημοπρασιών "Στο σφυρί".</p>
                    <table width="100%" style="text-align:
center; ">

                    <%
                    String query = "SELECT * FROM auction
ORDER BY start_date DESC LIMIT 5";
                    ResultSet rs = DBUtil.query(query);

                    while(rs.next()){
                        int id = rs.getInt("id");
                        String title = rs.getString("title");
                        String image = rs.getString("image");

                    %>
                    <tr class="auction">
                        <td><a href="showAuction.jsp?id=<%=id
%>"><img src="/product_images/<%=image%>" width="50"
height="50"></a></td>
                        <td><a href="showAuction.jsp?id=<%=id
%>"><%=title %></a></td>
                    </tr>
                    <% } %>

                    </table>

                    </td>
                </tr>
                <tr>
                    <td colspan="2" width="760">
                    <%@include file="frontend/footer.jsp" %>
                    </td>
                </tr>

            </table>
            </center>
        </div>
    </body>
</html>
```

Login.java

```
package frontend;

import java.io.IOException;
import java.sql.ResultSet;
```

```
import java.sql.SQLException;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import util.DBUtil;

/**
 *
 * @author nancy
 */
public class Login extends HttpServlet {


    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        try {
            String username =
request.getParameter("username");
            String pass = request.getParameter("pass");
            String query = "SELECT * FROM users WHERE
username='"+username+"' AND pass='"+pass+"'";
            ResultSet rs = DBUtil.query(query);

            RequestDispatcher dispatcher =
getServletContext().getRequestDispatcher("/index.jsp");


            if(rs.next()){
                System.out.println("user found!");
                request.getSession().setAttribute("username",
rs.getString("username"));
                request.getSession().setAttribute("userId",
rs.getInt("id"));

            }else{
                System.out.println("user not found!");
            }
            dispatcher.forward(request,response);

        }catch(ClassNotFoundException e){
        }catch(SQLException e){
        } finally {

        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
```

```
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

Logout.java

```
package frontend;

import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 *
 * @author nancy
 */
public class Logout extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {

        try {
            RequestDispatcher dispatcher =
getServletContext().getRequestDispatcher("/index.jsp");
            request.getSession().removeAttribute("username");
            request.getSession().removeAttribute("userId");
            dispatcher.forward(request,response);
        } finally {

        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
```

```
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

register.jsp

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">

<html>
    <head>
        <link href="frontend.css" type="text/css"
rel="stylesheet"/>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8" />
        <title>Στο σφυρί</title>
    </head>
    <body>
        <center>
        <table border="0">
            <tr>
                <td colspan="2" width="760">
                <%@include file="frontend/header.jsp" %>
                </td>
            </tr>
            <tr>
                <td width="200">
                    <%@include file="frontend/left_menu.jsp"
%>
                </td>
                <td valign="top">
```

```
        <h2>Εγγραφή χρήστη</h2>
            <form action="/Project/Register" method="post"
onsubmit="return validate()">
            <table>
                <tr>
                    <td>Ονομα χρήστη</td>
                    <td><input type="text" name="username"
id="username"/></td>
                </tr>
                <tr>
                    <td>Κωδικός</td>
                    <td><input type="password" name="pass"
id="pass"/></td>
                </tr>
                <tr>
                    <td>Επιβεβαίωση κωδικού</td>
                    <td><input type="password" name="pass2"
id="pass2"/></td>
                </tr>
                <tr>
                    <td>Email</td>
                    <td><input type="text" name="email"
id="email"/></td>
                </tr>

                <tr>
                    <td align="center" colspan="2"><input
type="submit" value="Εισαγωγή"/></td>
                </tr>
            </table>
        </form>
        <script type="text/javascript">
            function validate(){
                if(document.getElementById("username").value
== null
                 || document.getElementById("username").value
== ""
                 || document.getElementById("email").value ==
null
                 || document.getElementById("email").value ==
""
                 || document.getElementById("pass").value ==
null
                 || document.getElementById("pass").value ==
""
                 || document.getElementById("pass2").value ==
null
                 || document.getElementById("pass2").value ==
""){
                    alert("Παρακαλώ συμπληρώστε όλα τα
πεδία.");
                    return false;
                 }else
if(document.getElementById("pass").value !=
document.getElementById("pass2").value){
                    alert("Παρακαλώ εισάγετε τον ίδιο κωδικό
και στα δύο πεδία.");
                    return false;
                 }
                 return true;
            }
        </script>
                </td>
            </tr>
            <tr>
                <td colspan="2" width="760">
                <%@include file="frontend/footer.jsp" %>
                </td>
```

```
            </tr>

        </table>
        </center>
    </body>
</html>
```

Register.java

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */

package frontend;

import java.io.IOException;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import util.DBUtil;

/**
 *
 * @author nancy
 */
public class Register extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        try{
            String query = "SELECT * FROM users WHERE
username='" + request.getParameter("username") + "'";
            ResultSet rs = DBUtil.query(query);
            if(rs.next()){
                request.setAttribute("userExists", "true");
                RequestDispatcher dispatcher =
getServletContext().getRequestDispatcher("/register.jsp");
                dispatcher.forward(request,response);
                return;
            }

            query = "INSERT INTO users (username, pass, email)
VALUES ('" + request.getParameter("username") +"',
'"+request.getParameter("pass")+"',
'"+request.getParameter("email")+"')";
            DBUtil.update(query);

            RequestDispatcher dispatcher =
getServletContext().getRequestDispatcher("/index.jsp");
            request.setAttribute("username",
request.getParameter("username"));
            dispatcher.forward(request,response);

        }catch(ClassNotFoundException e){
```

```
            }catch(SQLException e){
            } finally {

            }
        }

        // <editor-fold defaultstate="collapsed" desc="HttpServlet
    methods. Click on the + sign on the left to edit the code.">
        /**
         * Handles the HTTP <code>GET</code> method.
         * @param request servlet request
         * @param response servlet response
         * @throws ServletException if a servlet-specific error
    occurs
         * @throws IOException if an I/O error occurs
         */
        @Override
        protected void doGet(HttpServletRequest request,
    HttpServletResponse response)
        throws ServletException, IOException {
            processRequest(request, response);
        }

        /**
         * Handles the HTTP <code>POST</code> method.
         * @param request servlet request
         * @param response servlet response
         * @throws ServletException if a servlet-specific error
    occurs
         * @throws IOException if an I/O error occurs
         */
        @Override
        protected void doPost(HttpServletRequest request,
    HttpServletResponse response)
        throws ServletException, IOException {
            processRequest(request, response);
        }

        /**
         * Returns a short description of the servlet.
         * @return a String containing servlet description
         */
        @Override
        public String getServletInfo() {
            return "Short description";
        }// </editor-fold>

    }
```

PlaceBid.java

```
    package util;

    import java.io.IOException;
    import java.io.PrintWriter;
    import java.sql.Connection;
    import java.sql.DriverManager;
    import java.sql.ResultSet;
    import java.sql.SQLException;
    import java.sql.Statement;
    import java.text.SimpleDateFormat;
    import java.util.Date;
    import javax.servlet.RequestDispatcher;
    import javax.servlet.ServletException;
    import javax.servlet.http.HttpServlet;
    import javax.servlet.http.HttpServletRequest;
    import javax.servlet.http.HttpServletResponse;
```

```
/**
 *
 * @author nancy
 */
public class PlaceBid extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            Date last_bid_time = new Date();

            double currentPrice =
Double.parseDouble(request.getParameter("currentPrice"));
            double ammount =
Double.parseDouble(request.getParameter("ammount"));

            Class.forName("com.mysql.jdbc.Driver");
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/db2?useUni
code=true&characterEncoding=UTF-8", "root", "");
            Statement stm = con.createStatement();

            SimpleDateFormat df = new SimpleDateFormat("yyyy-
MM-dd HH:mm:ss");
            String query = "INSERT INTO `bid` ( `auction_id` ,
`user_id` , `bid_time` , `amount` ) VALUES ('" +
request.getParameter("auctionId") + "', '" +
request.getSession().getAttribute("userId") + "', '" +
df.format(last_bid_time) + "' , '" +
request.getParameter("ammount") +"')";

            System.out.println(query);
            stm.executeUpdate(query);

            if(currentPrice < ammount){
                query = "SELECT auction.id, description, name,
title, last_bid_time, start_price, current_price, category_id,
start_date, status, company_id, company.username as
company_name, user_id FROM auction, company, category WHERE
auction.company_id=company.id AND
auction.category_id=category.id AND auction.id=" +
request.getParameter("auctionId");
                ResultSet rs = stm.executeQuery(query);
                rs.next();
                Date startDate =
rs.getTimestamp("start_date");
                Date now = new Date();

                if(now.before(startDate)){
                    last_bid_time = startDate;
                }

                df = new SimpleDateFormat("yyyy-MM-dd
HH:mm:ss");
                query = "UPDATE auction SET user_id=" +
request.getSession().getAttribute("userId") +
```

```
                        ", last_bid_time='" +
df.format(last_bid_time) + "', current_price=" +
request.getParameter("ammount") + " WHERE id=" +
request.getParameter("auctionId");
                System.out.println(query);
                stm.executeUpdate(query);
            }
            con.close();
            RequestDispatcher dispatcher =
getServletContext().getRequestDispatcher("/showAuction.jsp?id=
" + request.getParameter("auctionId"));
            dispatcher.forward(request,response);

        }catch(ClassNotFoundException e){
            e.printStackTrace();
        }catch(SQLException e){
            e.printStackTrace();
        } finally {
            out.close();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Returns a short description of the servlet.
     * @return a String containing servlet description
     */
    @Override
    public String getServletInfo() {
        return "Short description";
    }// </editor-fold>

}
```

frontend.css

```
package util;
```

```
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

/**
 *
 * @author nancy
 */
public class PlaceBid extends HttpServlet {

    /**
     * Processes requests for both HTTP <code>GET</code> and
<code>POST</code> methods.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    protected void processRequest(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        try {
            Date last_bid_time = new Date();

            double currentPrice =
Double.parseDouble(request.getParameter("currentPrice"));
            double ammount =
Double.parseDouble(request.getParameter("ammount"));

            Class.forName("com.mysql.jdbc.Driver");
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost/db2?useUni
code=true&characterEncoding=UTF-8", "root", "");
            Statement stm = con.createStatement();

            SimpleDateFormat df = new SimpleDateFormat("yyyy-
MM-dd HH:mm:ss");
            String query = "INSERT INTO `bid` ( `auction_id` ,
`user_id` , `bid_time` , `amount` ) VALUES ('" +
request.getParameter("auctionId") + "', '" +
request.getSession().getAttribute("userId") + "', '" +
df.format(last_bid_time) + "' , '" +
request.getParameter("ammount") +"')";

            System.out.println(query);
            stm.executeUpdate(query);

            if(currentPrice < ammount){
                query = "SELECT auction.id, description, name,
title, last_bid_time, start_price, current_price, category_id,
start_date, status, company_id, company.username as
company_name, user_id FROM auction, company, category WHERE
auction.company_id=company.id AND
```

```
auction.category_id=category.id AND auction.id=" +
request.getParameter("auctionId");
                ResultSet rs = stm.executeQuery(query);
                rs.next();
                Date startDate =
rs.getTimestamp("start_date");
                Date now = new Date();

                if(now.before(startDate)){
                    last_bid_time = startDate;
                }

                df = new SimpleDateFormat("yyyy-MM-dd
HH:mm:ss");
                query = "UPDATE auction SET user_id=" +
request.getSession().getAttribute("userId") +
                        ", last_bid_time='" +
df.format(last_bid_time) + "', current_price=" +
request.getParameter("ammount") + " WHERE id=" +
request.getParameter("auctionId");
                System.out.println(query);
                stm.executeUpdate(query);
            }
            con.close();
            RequestDispatcher dispatcher =
getServletContext().getRequestDispatcher("/showAuction.jsp?id=
" + request.getParameter("auctionId"));
            dispatcher.forward(request,response);

        }catch(ClassNotFoundException e){
            e.printStackTrace();
        }catch(SQLException e){
            e.printStackTrace();
        } finally {
            out.close();
        }
    }

    // <editor-fold defaultstate="collapsed" desc="HttpServlet
methods. Click on the + sign on the left to edit the code.">
    /**
     * Handles the HTTP <code>GET</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doGet(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }

    /**
     * Handles the HTTP <code>POST</code> method.
     * @param request servlet request
     * @param response servlet response
     * @throws ServletException if a servlet-specific error
occurs
     * @throws IOException if an I/O error occurs
     */
    @Override
    protected void doPost(HttpServletRequest request,
HttpServletResponse response)
    throws ServletException, IOException {
        processRequest(request, response);
    }
```

```
        /**
         * Returns a short description of the servlet.
         * @return a String containing servlet description
         */
        @Override
        public String getServletInfo() {
            return "Short description";
        }// </editor-fold>

    }
```

Βιβλιογραφία

1. «Servlets και σελίδες διακομιστή Java», Marty Hall – Larry Brown, εκδόσεις Κλειδάριθμος
2. Java Servlet Technology, http://java.sun.com/j2ee/tutorial/1_3-fcs/doc/Servlets.html
3. JavaServer Pages Technology, http://java.sun.com/j2ee/tutorial/1_3-fcs/doc/JSPIntro.html
4. SQL INSERT INTO Statement, http://www.w3schools.com/SQL/sql_insert.asp
5. SQL UPDATE Statement, http://www.w3schools.com/SQl/sql_update.asp
6. Select (SQL), http://en.wikipedia.org/wiki/Select_(SQL)
7. Java™ Platform, Standard Edition 6, API Specification, http://java.sun.com/javase/6/docs/api/index.html
8. CSS Tutorial, http://www.w3schools.com/css/
9. JavaScript Form Validation, http://www.w3schools.com/jS/js_form_validation.asp