ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ Τ.Ε.

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΘΕΜΑ

Αλυσίδα Supermarket

Ανεθρεπτάκης Ευτύχης Α.Μ. 3741

Κονιδάκης Αντώνης Α.Μ. 3280

Μπόνης Δωρόθεος Α.Μ. 3926

Επιβλέπων καθηγητής: Παπαδάκης Νικόλαος

ΗΡΑΚΛΕΙΟ 2017

Ευχαριστίες

Ολοκληρώνοντας την παρούσα πτυχιακή εργασία θα θέλαμε να ευχαριστήσουμε αρχικά τον επιβλέποντα καθηγητή μας, κύριο Παπαδάκη Νικόλαο για τη συνεχή καθοδήγηση και βοήθεια χωρίς τις οποίες δε θα μπορούσε να ολοκληρωθεί η παρούσα εργασία. Επιπλέον, θα θέλαμε να ευχαριστήσουμε τις οικογένειές μας και όλους εκείνους που ήταν δίπλα μας σε όλα τα χρόνια της φοίτησής μας, παρέχοντάς μας ηθική και συναισθηματική υποστήριξη.

Contents

Σύνοψη

Σκοπός της παρούσας πτυχιακής εργασίας είναι η θεωρητική αλλά και πρακτική δημιουργία μιας αλυσίδας supermarket χρησιμοποιώντας την SQL. Σε θεωρητικό κομμάτι υλοποιήσαμε ένα πλήρες διάγραμμα οντοτήτων – συσχετίσεων, παρουσιάσαμε τα γνωρίσματα όλων των οντοτήτων και των σχέσεων, τα πρωτεύοντα κλειδιά, τη μετάφραση του μοντέλου μας σε σχεσιακό μοντέλο και τη μετατροπή του σε τρίτη κανονική μορφή 3NF.

Στο πρακτικό κομμάτι υλοποιήσαμε τις εντολές σε βάση δεδομένων SQL και δημιουργήσαμε ένα web interface για την αλυσίδα supermarket, χρησιμοποιώντας τεχνολογίες όπως HTML, PHP και Javascript.

Abstract

The purpose of this thesis is to study theoretically and practically a supermarket chain. Theoritically, we create an Entity - Relational Model. We created the attributes of entities, primary keys, the Relational Model and converted the model in 3NF.

Practically, we designed a web interface which represents a supermarket chain.The main technologies used are SQL, HTML, PHP and Javascript.

Κεφάλαιο 1ο

1.1 Εισαγωγή

Η σημερινή εποχή, χαρακτηρίζεται από ραγδαία αύξηση της χρήσης των ηλεκτρονικών υπολογιστών. Συγκεκριμένα όλες σχεδόν οι ενέργειές μας πραγματοποιούνται με τη χρήση ηλεκτρονικών εφαρμογών, ώστε να ελαχιστοποιείται ο χρόνος εκτέλεσης των ενεργειών μας. Ιδιαίτερο χαρακτηριστικό της σημερινής εποχής είναι η ηλεκτρονική παραγγελία αγαθών. Η παρούσα πτυχιακή ασχολείται με την υλοποίηση ενός συστήματος ηλεκτρονικού supermarket, που στόχο έχει, την διευκόλυνση των πελατών για αγορά αγαθών.

1.2 Περίληψη

Στη πτυχιακή αυτή γίνεται περιγραφή των λειτουργιών που υπάρχουν σε ένα σύστημα ηλεκτρονικής αλυσίδας supermarket. Σε ένα τέτοιο σύστημα ένας διαχειριστής του supermarket θα μπορεί να τοποθετήσει προϊόντα σε αυτό, να διαγράψει προϊόντα, να εισάγει αποθήκες και υπαλλήλους, να εισάγει ένα προϊόν σε μια αποθήκη και να αντιστοιχεί προϊόντα σε προμηθευτές.

1.3 Κίνητρο για Διεξαγωγή της εργασίας

Βασικό κίνητρο για τη διεξαγωγή της εργασίας αυτής, αποτέλεσε η ανάγκη όλο και περισσότερων χρηστών να πραγματοποιούν τις ενέργειές τους ηλεκτρονικά. Επιπλέον, κίνητρο παρέχει η ενασχόληση με τη γλώσσα υλοποίησης διαδικτυακών εφαρμογών PHP, HTML και Javascript όπως επίσης και της MySQL που χρησιμοποιούνται για την δημιουργία ιστοσελίδων και web εφαρμογών καθώς επίσης η τελευταία χρησιμοποιείται για κατασκευή βάσεων δεδομένων.

1.4 Σκοπός και στόχοι εργασίας

Σκοπός της εργασίας αυτής, είναι η ενασχόληση με τη γλώσσα υλοποίησης διαδικτυακών εφαρμογών PHP καθώς και οι γλώσσες που χρησιμοποιούνται σε web εφαρμογές όπως η HTML και η Javascript . Στόχος της πραγματοποίησης αυτής της πτυχιακής εργασίας, είναι η απόκτηση εμπειρίας σε θέματα υλοποίησης web εφαρμογών. Πιο συγκεκριμένα ο στόχος που επιδιώκεται είναι μετά το πέρας της εργασίας αυτής, να μπορούν να υλοποιηθούν πλήρως τέτοια συστήματα.

1.5 Δομή εργασίας

Η δομή της εργασίας αυτής, έχει ως εξής:
Στο 1ο κεφάλαιο, παρατίθενται εισαγωγικά στοιχεία, σχετικά με τους λόγους υλοποίησης της συγκεκριμένης πτυχιακής.

Στο 2ο Κεφάλαιο, γίνεται η παρουσίαση των τεχνολογιών που χρησιμοποιήθηκαν για την ανάπτυξη ιστοσελίδων όπως PHP,HTML, Javascript.

Στο 3ο Κεφάλαιο, γίνεται αναφορά στις βάσεις δεδομένων όπως και στην χρήση της MySQL σε συνεργασία με την PHP.

Στο 4ο Κεφάλαιο, διαδραματίζεται το κύριο μέρος της εργασίας μας όπου αναλύουμε τις κινήσεις που έγιναν για να δημιουργηθεί η αλυσίδα supermarket καθώς και τα screenshots από το Interface του προγράμματός μας.

Τέλος στο 5ο Κεφάλαιο εξάγονται τα συμπεράσματα και οι μελλοντικές επεκτάσεις που μπορούν να γίνουν στο πρόγραμμά μας.

Κεφάλαιο 2ο

2.1 PHP

Η γλώσσα PHP (Hypertext PreProcessor) είναι μια γλώσσα προγραμματισμού που χρησιμοποιείται για τη δημιουργία εφαρμογών Web. Μια εφαρμογή Web μπορεί να είναι οτιδήποτε από μία απλή εφαρμογή "login" μέχρι ένα ηλεκτρονικό κατάστημα. Η γλώσσα Php είναι γνωστή ως μια γλώσσα προγραμματισμού server-side. Αυτό σημαίνει ότι λειτουργεί με το Web server. Οι περισσότερες γλώσσες προγραμματισμού Web είναι server-side, αλλά μερικές, όπως ή Javascript είναι client-side, που σημαίνει ότι χρησιμοποιούνται από την πλευρά του browser.
Μία server-side γλώσσα είναι περισσότερο ευέλικτη, δεδομένου ότι μας δίνει τη δυνατότητα να πραγματοποιήσουμε πολλά πράγματα που είναι δύσκολο και έως αδύνατο να κάνουμε με JavaScript. Αν παραδείγματος χάριν, δουλεύουμε σε συνεργασία με αρχεία και βάσεις δεδομένων, ή επεξεργασία εικόνων είναι απαραίτητη η χρήση της PHP.
Η γλώσσα προγραμματισμού PHP έχει το ιδιαίτερο χαρακτηριστικό ότι ο κώδικάς της πρώτα μεταγλωττίζεται στον server σαν ένα κανονικό html έγγραφο, χωρίς ο χρήστης να είναι σε θέση να δει τον αρχικό κώδικα.
Ο Server-side κώδικας είναι πιθανών πιο ασφαλής από ό, τι ο κώδικας JavaScript. Ο JavaScript κώδικας αποστέλλεται στο πρόγραμμα περιήγησης μέσω Web και έτσι είναι εύκολο για έναν επισκέπτη να δει και να επεξεργαστεί τον κώδικα. Ωστόσο, ο Server-side κώδικας, παραμένει στον Web server και δεν είναι προσβάσιμος στους επισκέπτες του site.

2.2 Σύνταξη και εντολές της PHP

Έχουμε τον παρακάτω κώδικα της Php

```
<?php
echo "<html><body>";
echo "Hello World!";
echo
"</body></html>";
?>
```

Με το συγκεκριμένο κομμάτι κώδικα θα δούμε το μήνυμα ***Hello World!*** στην οθόνη μας. Οι ετικέτες (tags) **<*?php*** και ***?>*** χρησιμοποιούνται για να δηλώσουμε ένα μπλοκ κώδικα PHP.

Για να μπορεί ο web server να επεξεργαστεί τον κώδικα της PHP, πρέπει να «γνωρίζει» πού ακριβώς αρχίζει και πού ακριβώς τελειώνει ένα μπλοκ κώδικα PHP.

Η εντολή ***echo*** χρησιμοποιείται για να στείλουμε ένα κείμενο (string) στον φυλλομετρητή (browser). Όλες οι εντολές της Php πρέπει να τελειώνουν με τον χαρακτήρα ***;*** **.**

Όταν ένας φυλλομετρητής «ζητήσει» μια σελίδα PHP, ο server θα την επεξεργαστεί, θα μετατρέψει τον κώδικα PHP σε καθαρή HTML με αυτόν τον τρόπο, ο χρήστης δεν θα μπορέσει να δει τον αρχικό κώδικα PHP.

2.3 Οι μεταβλητές στην PHP

Στην PHP μια μεταβλητή ξεκινάει με το σύμβολο ***$*** . Μια μεταβλητή μπορεί να έχει ένα μικρό όνομα όπως x και y ή ένα πιο περιγραφικό όπως age, total_volume.

Κανόνες για τις PHP μεταβλητές:

- Μια μεταβλητή ξεκινά με το $ ακολουθούμενη από το όνομα της μεταβλητής.
- Μια μεταβλητή πρέπει να ξεκινάει με ένα γράμμα ή με κάτω παύλα.
- Ένα όνομα μεταβλητής δεν μπορεί να ξεκινά με αριθμό.
- Ένα όνομα μεταβλητής μπορεί μόνο να περιλαμβάνει alpha-numeric χαρακτήρες και κάτω παύλες.
- Οι μεταβλητές στην PHP είναι case – sensitive. Δηλαδή η $age και $AGE είναι δύο διαφορετικές μεταβλητές.

2.4 PHP loops

Πολλές φορές στο πρόγραμμά μας, θέλουμε το ίδιο κομμάτι κώδικα να τρέξει πολλές φορές. Επομένως, αντί για την προσθήκη ίδιων γραμμών κώδικα, μπορούμε να χρησιμοποιούμε βρόχους για να εκτελέσουμε μια εργασία.
Στην PHP, έχουμε τις ακόλουθες δηλώσεις γι' αυτό το σκοπό:

- **while** - βρόχος με ένα κομμάτι του κώδικα, ενώ μια συγκεκριμένη συνθήκη είναι αληθής.
- **do ... while** - βρόχος με ένα κομμάτι κώδικα για μία φορά, και στη συνέχεια να επαναλαμβάνεται ο βρόχος εφ 'όσον ένας προκαθορισμένος όρος είναι αληθής.
- **for** - βρόχος με ένα κομμάτι κώδικα ένα συγκεκριμένο αριθμό επαναλήψεων.
- **foreach** - βρόχος με ένα κομμάτι κώδικα για κάθε στοιχείο σε μια συστοιχία.

2.5 Η μεταβλητή $_GET στην PHP

Η $ _GET μεταβλητή χρησιμοποιείται για τη συλλογή τιμών σε μια φόρμα με τιμή: method ="get".
Οι πληροφορίες που αποστέλλονται από μια φόρμα με τη μέθοδο GET είναι ορατές σε όλους. Υπάρχει ένας περιορισμός 2000 χαρακτήρων που αποτελούν την ποσότητα που μπορεί να σταλεί.

Αν έχουμε το ακόλουθο παράδειγμα φόρμας σε PHP:

<form action="hello.php" method="get"> Name: <input type="text" name="fname" /> Age: <input type="text" name="age" /> <input type="submit" /> </form>

Όταν ο χρήστης χρησιμοποιήσει το κουμπί "Submit", η διεύθυνση URL που αποστέλλεται στο διακομιστή θα μπορούσε να η εξής:
http://welcome.php?fname=Giannis&age=25

Το πιο σημαντικό εδώ είναι ότι η μέθοδος αυτή δεν πρέπει να χρησιμοποιείται κατά την αποστολή κωδικών πρόσβασης ή άλλων προσωπικών στοιχείων αφού το περιεχόμενό της είναι ορατό σε όλους!

2.6 Η μεταβλητή $_POST στην PHP

Η $ _POST μεταβλητή χρησιμοποιείται για να συλλέξουμε τις τιμές από μια φόρμα. Στέλνεται με τιμή: method = "post".

Οι πληροφορίες που αποστέλλονται από τη φόρμα με τη μέθοδο POST δεν είναι ορατές για τους άλλους και δεν υπάρχουν όρια για την ποσότητα των πληροφοριών όπως παρατηρήσαμε στην $_GET.

Αν έχουμε το παράδειγμα:

<form action="welcome.php" method="post"> Όνομα: <input type="text" name="fname" /> Ηλικία: <input type="text" name="age" /> <input type="submit" /> </ form>

Όταν ο χρήστης χρησιμοποιήσει το κουμπί "Submit", η διεύθυνση URL που αποστέλλεται στο διακομιστή θα είναι περίπου η ακόλουθη:

http:// welcome.php.

2.7 HTML

Η HTML ή HyperText Markup Language είναι η γλώσσα που καθορίζει τη λογική οργάνωση ενός εγγράφου. Το ίδιο έγγραφο HTML μπορεί να προβληθεί από πολλούς διαφορετικούς “browsers”, με διαφορετικές ιδιότητες. Για παράδειγμα, ένα πρόγραμμα περιήγησης μπορεί κόψει την αρχή μιας παραγράφου, ενώ κάποιο άλλο μπορεί να αφήσει μόνο μια κενή γραμμή.

Η HTML γράφεται υπό μορφή στοιχείων HTML τα οποία αποτελούνται από *ετικέτες*, οι οποίες περικλείονται μέσα σε σύμβολα σαν μονά εισαγωγικά < και >. Οι ετικέτες HTML λειτουργούν ανά ζεύγη (για παράδειγμα <h2> και </h2>).

Τα στοιχεία της HTML χρησιμοποιούνται για να κτίσουν όλους του ιστότοπους. Η HTML επιτρέπει την ενσωμάτωση εικόνων και άλλων αντικειμένων μέσα στη σελίδα

Ο σκοπός ενός web browser είναι να διαβάζει τα έγγραφα HTML και τα συνθέτει σε σελίδες που μπορεί κανείς να διαβάσει ή να ακούσει. Ο browser δεν εμφανίζει τις ετικέτες HTML, αλλά τις χρησιμοποιεί για να ερμηνεύσει το περιεχόμενο και το νόημα της σελίδας. Ο κώδικας σε HTML γράφεται μέσα στις ετικέτες <body> και </body>. Οι πληροφορίες για τη γλώσσα που θα χρησιμοποιηθεί ή για το συγγραφέα της σελίδας γράφονται μέσα στο ζεύγος ετικετών <head>, </head>.

2.8 HTML Elements

Αν έχουμε το επόμενο παράδειγμα σε html:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hello HTML</title>
  </head>
  <body>
    <p>Hello world</p>
  </body>
</html>
```

Το κείμενο ανάμεσα στο <html> και το </html> περιγράφει την ιστοσελίδα, και το κείμενο μεταξύ του <body> και του </body> είναι το ορατό μέρος της. Το κείμενο <title>Hello HTML</title> καθορίζει τον τίτλο που θα εμφανίζεται στην μπάρα τίτλου του browser.To Document Type Declaration (<!DOCTYPE html>) είναι για την HTML5.

Η αλλαγή γραμμής συμβολίζεται με
. Το
 αλλάζει γραμμή χωρίς να αλλάζει τη δομή της σελίδας. Είναι δηλαδή ένα άδειο στοιχείο χωρίς περιεχόμενο και δεν χρειάζεται ετικέτα κλεισίματος.

Στον κώδικα html μπορούν να βοηθήσουν επίσης τα σχόλια τα οποία δίνουν περισσότερες πληροφορίες για τον κώδικα και βοηθούν έναν τρίτο αναγνώστη να κατανοήσει καλύτερα την οργάνωση των τμημάτων κώδικα ή ακόμα και τον ίδιο το συγγραφέα του προγράμματος που μετά από πολύ καιρό όταν ξαναδιαβάσει τον κώδικά που έχει ο ίδιος γράψει θα θυμηθεί αμέσως τι ακριβώς έχει κάνει. Τα σχόλια στην html περικλείονται ανάμεσα στα <!-- και -->.

Επιπλέον το κείμενο μπορεί να πάρει διάφορες μορφές όπως να επισημανθεί και να γίνει πιο σκούρο. ΤΟ έντονο κείμενο εμφανίζεται ανάμεσα στις ετικέτες και . Ωστόσο δεν υποδηλώνουν τι θα γίνει στις συσκευές ανάγνωσης φωνητικών μυνημάτων. Οι ετικές αυτές ισχύουν μόνο για γραπτά κείμενα.

Στον κώδικα html μπορούμε επιπλέον να ενσωματώσουμε εικόνες ρυθμίζοντας το ύψος και το πλάτος τους με συγκεκριμένες ετικέτες όπως width και height, όπως παρουσιάζεται στο παρακάτω παράδειγμα.

`<img src="image.gif" width="50" height="50" border="0"></a>`.

Η ιδιότητα class στην html, δίνει τη δυνατότητα στον προσγραμματιστή να ταξινομεί παρόμοια αντικείμενα στην ίδια κλάση και να τα ομαδοποιεί.

Η ιδιότητα style εφαρμόζεται ώστε να δώσει συγκεκριμένο στυλ εμφάνισης σε στοιχεία.

Ένα επιπλέον σημαντικό στοιχείο είναι αυτό της κωδικοποίησης των χαρακτήρων στα ελληνικά. Εφαρμογές που χρησιμοποιούν ελληνικούς χαρακτήρες θα πρέπει να έχουν το συγκεκριμένο τμήμα κώδικα μέσα στο head ώστε να εμφανίζονται οι ελληνικοί χαρακτήρες σωστά σε utf-8.

```
<html lang="en">

  <head>

    <meta charset="utf-8">

    </head>
```

Το lang = “en” δηλώνει ότι θα χρησιμοποιηθούν ελληνικά.

2. 9 Javascript

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

Η JavaScript είναι μια γλώσσα σεναρίων που βασίζεται στα πρωτότυπα (prototype-based), είναι δυναμική, με ασθενείς τύπους και έχει συναρτήσεις ως αντικείμενα πρώτης τάξης.
Η σύνταξή της είναι επηρεασμένη από τη C.

Η JavaScript αντιγράφει πολλά ονόματα και συμβάσεις ονοματοδοσίας από τη Java, αλλά γενικά οι δύο αυτές γλώσσες δε σχετίζονται και έχουν πολύ διαφορετική σημασιολογία. Οι βασικές αρχές σχεδιασμού της JavaScript προέρχονται από τις γλώσσες προγραμματισμού Self και Scheme. Είναι γλώσσα βασισμένη σε διαφορετικά προγραμματιστικά παραδείγματα (multi-paradigm), υποστηρίζοντας αντικειμενοστρεφές,προστακτικό και συναρτησιακό στυλ προγραμματισμού.

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

Ο κώδικας Javascript μιας σελίδας περικλείεται από τις ετικέτες της HTML <script type="text/javascript"> και </script>. Ο ακόλουθος κώδικας javascript εμφανίζει ένα πλαίσιο διαλόγου με το κείμενο Hello World.

```
<script type="text/javascript">
alert('Hello World');
</script>
```

Οι εντολές της Javascript χωρίζονται μεταξύ τους με ελληνικό ερωτηματικό **;** .

Κεφάλαιο 3ο

Προϊόν

Στοιχεία προϊόντος:

1. Όνομα
2. Είδος
3. Κατασκευαστής/Παραγωγός
4. Τιμή

Αποθήκη

Στοιχεία αποθήκης:

1. Όνομα
2. Διεύθυνση

Υποκατάστημα

Στοιχεία υποκαταστήματος:

1. Όνομα
2. Διεύθυνση

Υπάλληλος

Στοιχεία υπαλλήλου:

1. Ονοματεπώνυμο
2. Διεύθυνση
3. Τηλέφωνο
4. Μισθός
5. Αριθμός Ταυτότητας

Προμηθευτής

Στοιχεία προμηθευτή:

1. Ονοματεπώνυμο
2. Διεύθυνση
3. Τηλέφωνο

Παραγγελία

Στοιχεία παραγγελίας:

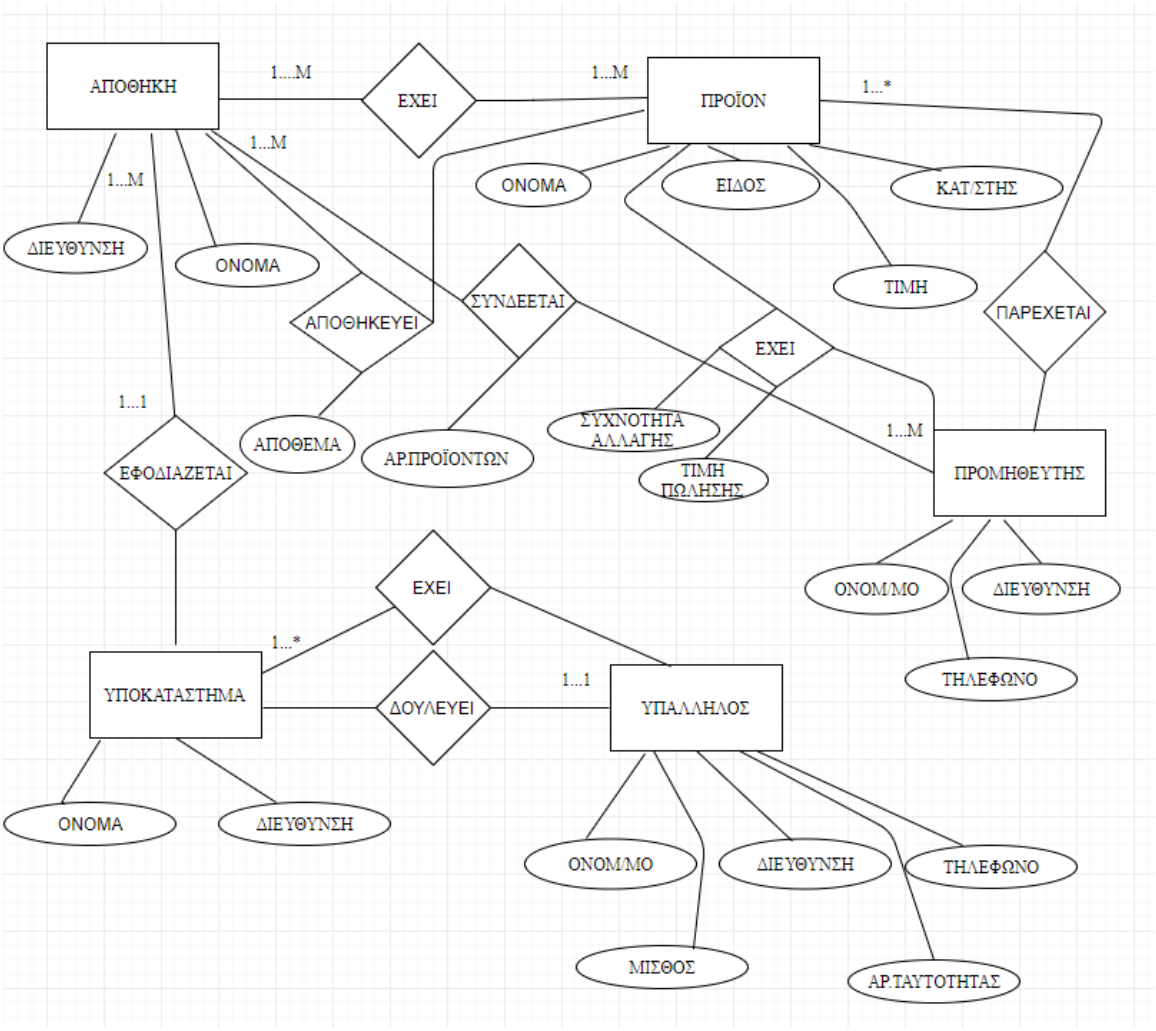
1. Κωδικός
2. Ποσότητα

3.1 Μοντέλο Οντοτήτων-Συσχετίσεων

(Entity Relationship Model-ER)

Οντότητες: Διακριτά αντικείμενα του πραγματικού κόσμου. Αντικείμενα με φυσική ύπαρξη. Στη συγκεκριμένη εργασία τα: Προϊόν, Αποθήκη, Υποκατάστημα, Υπάλληλος, Προμηθευτής και Παραγγελία.

Συμβολίζονται με 



3.2 Γνωρίσματα Οντοτήτων

Τα γνωρίσματα αποτελούν ίδιες ιδιότητες των οντοτήτων. Τα γνωρίσματα διαθέτουν όνομα και τύπο. Επομένως η οντότητα **Προϊόν** έχει τα γνωρίσματα : Όνομα που είναι απλό ή αλλιώς ατομικό γνώρισμα, Είδος που αποτελεί απλό ή ατομικό γνώρισμα, κατασκευαστής/παραγωγός που είναι απλό γνώρισμα και τιμή που είναι σύνθετο καθώς αποτελείται από το ποσό και την μονάδα μέτρησης(πχ: €). Για την οντότητα **Αποθήκη** τα γνωρίσματα είναι: Όνομα που είναι απλό γνώρισμα και Διεύθυνση που είναι σύνθετο γνώρισμα καθώς αποτελείται από Οδό, Αριθμό, Τ.Κ και Πόλη. Για την οντότητα **Υποκατάστημα** τα γνωρίσματα είναι τα: Όνομα που αποτελεί απλό γνώρισμα και Διεύθυνση(σύνθετο γνώρισμα ίδιες αναφέρθηκε παραπάνω). Για την οντότητα **Υπάλληλος** τα γνωρίσματα είναι τα εξής: Ονοματεπώνυμο που αποτελεί σύνθετο γνώρισμα αφού αποτελείται από Όνομα και Επώνυμο, Διεύθυνση(σύνθετο γνώρισμα), Τηλέφωνο που αποτελεί πλειότιμο γνώρισμα, διότι είναι ένα σύνολο από τιμές, Μισθός που αποτελεί σύνθετο γνώρισμα καθώς σχηματίζεται από το ποσό και τη μονάδα μέτρησης (€), και Αριθμός Ταυτότητας που αποτελεί μονότιμο γνώρισμα καθώς μπορεί να λάβει μόνο μια τιμή. Για την οντότητα **Προμηθευτής** τα γνωρίσματα είναι τα: Ονοματεπώνυμο(σύνθετο γνώρισμα), Διεύθυνση(σύνθετο γνώρισμα) και Τηλέφωνο (πλειότιμο γνώρισμα). Τέλος, για την οντότητα **Παραγγελία** τα γνωρίσματα είναι τα: Κωδικός που είναι απλό γνώρισμα και Ποσότητα που είναι απλό γνώρισμα.

Τα γνωρίσματα συμβολίζονται με 

Ο τύπος συσχέτισης R ορίζει μια σύνδεση (σχέση) μεταξύ διάφορων τύπων οντοτήτων και συμβολίζεται με 

3.3 Γνωρίσματα Σχέσεων

Οι σχέσεις που δημιουργούνται έχουν και αυτές κάποια γνωρίσματα(ίδιες φαίνεται και στο παρακάτω μοντέλο).

Η σχέση ΣΥΝΔΕΕΤΑΙ μεταξύ των οντοτήτων ΑΠΟΘΗΚΗ και ΠΡΟΪΟΝ έχει γνώρισμα το Αριθμός Προϊόντων που είναι απλό γνώρισμα. Η σχέση ΑΠΟΘΗΚΕΥΕΙ έχει γνώρισμα το Απόθεμα που είναι απλό γνώρισμα. Η σχέση ΕΧΕΙ μεταξύ των οντοτήτων ΠΡΟΪΟΝ και ΠΡΟΜΗΘΕΥΤΗΣ, διαθέτει τα γνωρίσματα Τιμή Πώλησης που είναι σύνθετο γνώρισμα καθώς αποτελείται από το ποσό και τη μονάδα μέτρησης και Συχνότητα Αλλαγής που είναι απλό γνώρισμα.

3.4 Πρωτεύοντα Κλειδιά

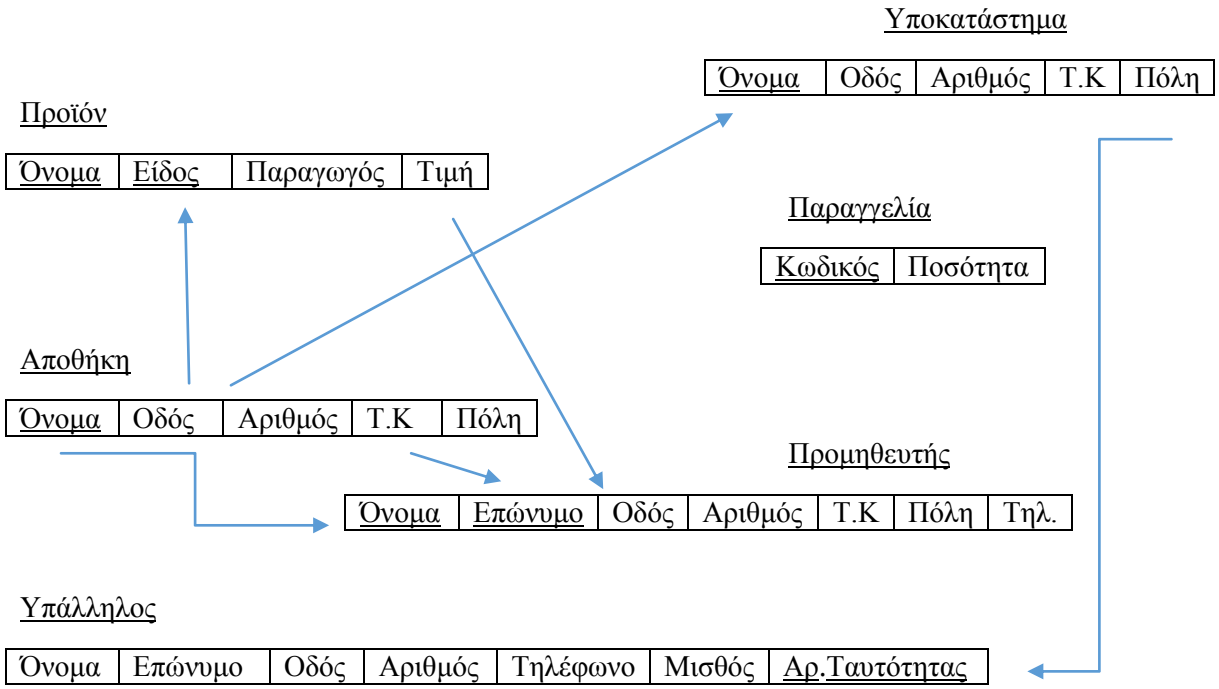
Οι τιμές κάποιου γνωρίσματος προσδιορίζουν μία οντότητα μοναδικά, δηλαδή δεν μπορεί να υπάρχουν δύο οντότητες με ίδιες ίδιες τιμές στα γνωρίσματα – κλειδιά. Το κλειδί είναι ένα σύνολο γνωρισμάτων. Πρωτεύον κλειδί ονομάζεται το υποψήφιο κλειδί που επιλέγουμε (primary key). Υποψήφιο κλειδί είναι αυτό με το μικρότερο αριθμό γνωρισμάτων, δηλαδή αν αφαιρέσουμε ένα γνώρισμα παύει να είναι κλειδί. Επομένως τα πρωτεύοντα κλειδιά είναι τα εξής: Για την οντότητα Προϊόν το Όνομα, Είδος. Για την οντότητα Αποθήκη το Όνομα. Για την οντότητα Υποκατάστημα το Όνομα. Για την οντότητα Υπάλληλος ο Αριθμός Ταυτότητας. Για την οντότητα Προμηθευτής το Ονοματεπώνυμο και για την οντότητα Παραγγελία ο Κωδικός.

Τα γνωρίσματα των οντοτήτων γίνονται ορατά από την περιγραφή ίδιες. Ωστόσο υπάρχουν και κάποια επιπλέον γνωρίσματα που δεν είναι τόσο προφανή. Στη σχέση ΕΧΕΙ μεταξύ ΠΡΟΜΗΘΕΥΤΗ και ΠΡΟΪΟΝ, υπάρχει η τιμή πώλησης που ίδιες αναφέρεται και στην εκφώνηση κάθε προμηθευτής έχει μια τιμή πώλησης που αλλάζει συχνά. Επιπλέον το απόθεμα είναι ένα μη προφανές γνώρισμα στη σχέση ΑΠΟΘΗΚΕΥΕΙ. Ίδιες προμηθευτής αναφέρεται ότι συνδέεται με ίδιες αποθήκες και αντίστροφα οπότε προκύπτουν τα γνωρίσματα Αριθμός Προϊόντων και Αριθμός Προμηθευτών.

3.5 Σχεσιακό Μοντέλο

(Relational Model)

Το σχεσιακό μοντέλο παρουσιάζει μια βάση ως συλλογή από σχέσεις. Μια σχέση είναι ίδιες πίνακας με διακριτό όνομα. Κάθε στήλη στον πίνακα αντιπροσωπεύει ένα γνώρισμα. Κάθε γραμμή στον πίνακα είναι μια πλειάδα. Μια πλειάδα αντιπροσωπεύει μια σχέση μεταξύ τιμών των γνωρισμάτων.



Η οντότητα Διεύθυνση χωρίστηκε στο σχεσιακό μοντέλο σε απλούστερες καθώς αποτελείται από Οδό, Αριθμό, Τ.Κ, Πόλη.

3.6 Περιορισμοί Ακεραιότητας

Ο περιορισμός ακεραιότητας οντοτήτων ικανοποιείται πάντα καθώς η τιμή κανενός πρωτεύοντος κλειδιού είναι null.

3.7 Συναρτησιακές Εξαρτήσεις

Είναι εξαρτήσεις ανάμεσα σε σύνολα από γνωρίσματα. Συμβολίζεται ως S1 →S2, όπου S1 και S2 σύνολα γνωρισμάτων. Αυτό σημαίνει ότι αν ίδιες τιμές στα γνωρίσματα του S1 τότε θα έχουμε ίδιες τιμές στα γνωρίσματα του S2.

Κεφάλαιο 4ο

4.1 Εγχειρίδιο χρήσης

4.1.1 Εγκατάσταση

Για την εργασία χρησιμοποιήθηκε βάση δεδομένων MySQL. Συγκεκριμένα, το περιβάλλον στο οποίο υλοποιήσαμε και ελέγξαμε την εργασία, είχε Debian για λειτουργικό σύστημα και έτρεχε τις latest εκδόσεις των Apache και MySQL.

Στις παρακάτω οδηγίες εγκατάστασης, θεωρείται δεδομένη η ύπαρξη και λειτουργία μιας MySQL βάσης και ενός Apache webserver με τα κατάλληλα extensions ώστε να συνδεθεί με την MySQL.

Το πρώτο πράγμα που πρέπει να γίνει είναι να δημιουργηθεί η βάση δεδομένων. Για να γίνει αυτό, έχει υλοποιηθεί ένα script το οποίο αυτόματα δημιουργεί τη βάση, τους πίνακες, τα απαραίτητα trigger και constraints και εισάγει κάποια ενδεικτικά δεδομένα.

Στη συνέχεια, πρέπει να μετακινήσουμε τα αρχεία του project (“js/main.js”, “*.php”, “index.html”) σε κάποιο directory στο οποίο να έχει πρόσβαση ο Apache (έστω /var/www/html/project).

Τέλος, πρέπει να ανοίξουμε το αρχείο admin.php και να εισάγουμε τα κατάλληλα στοιχεία ώστε να συνδεθούμε στην MySQL. Παρακάτω ακολουθούν εξηγήσεις για το κάθε ένα.

- $servername – Η ip του server που φιλοξενεί τη βάση δεδομένων. Βάζουμε “localhost” αν η βάση φιλοξενείται στο ίδιο μηχάνημα με τον web server.
- $username – Το όνομα χρήστη της MySQL. Σημειώνεται πως ο συγκεκριμένος χρήστης πρέπει να έχει full δικαιώματα ώστε να μπορεί να δημιουργήσει και να επεξεργαστεί μια βάση δεδομένων.
- $password – Ο κωδικός του χρήστη της MySQL.
- $dbname – Το όνομα της βάσης. Παραμένει ”s”, εκτός αν αλλάξουμε το όνομα της βάσης στις πρώτες γραμμές του “init.sql”.

4.1.2 MySQL Triggers

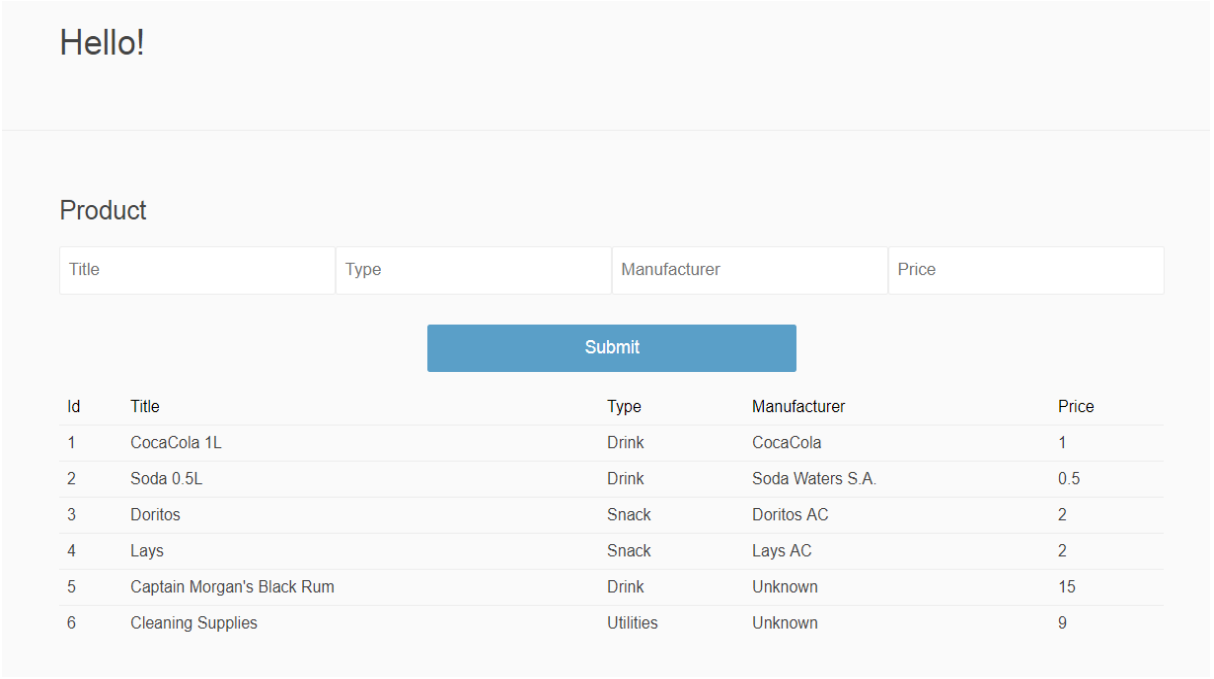
Για τα τελευταία ερωτήματα της εκφώνησης, χρησιμοποιήθηκαν triggers, ώστε να γίνονται αυτόματα παραγγελίες και ανεφοδιασμοί, ανάλογα με τα αποθέματα. Όπως ορίζει η εκφώνηση, όταν κάποιο προϊόν ενός καταστήματος μειωθεί κάτω από το ελάχιστο όριο, γίνεται αυτόματα ανεφοδιασμός από την αποθήκη. Αν και τα προϊόντα της αποθήκης αρχίσουν να μειώνονται, καταχωρείται αυτόματα μια παραγγελία προς τον φθηνότερο προμηθευτή.

4.1.3 PHP

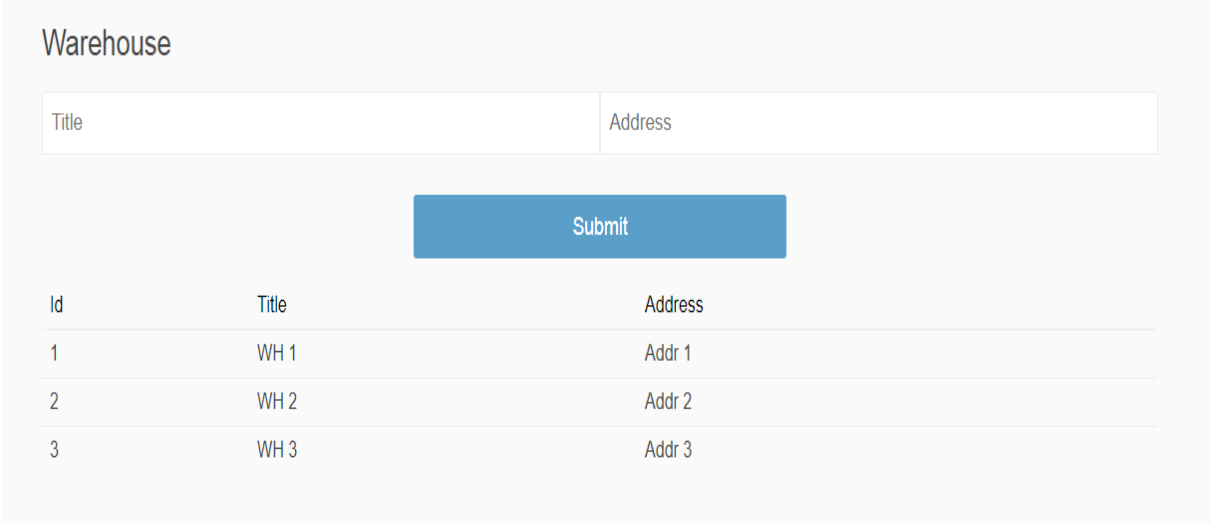
Για την επικοινωνία του Web Interface, χρησιμοποιήθηκε η γλώσσα προγραμματισμού PHP. Για τις ανάγκες του interface, υλοποιήθηκαν τα αρχεία ”get.php ” και “add.php” τα οποία αντίστοιχα ανακτούν και προσθέτουν πληροφορίες στη βάση δεδομένων.

5ο Κεφάλαιο

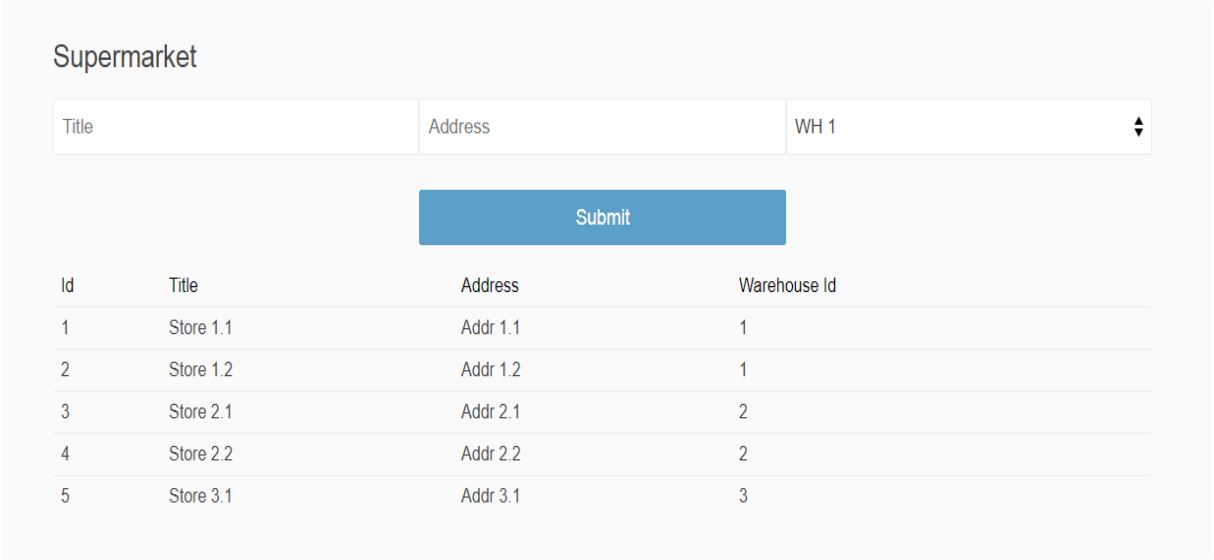
Σε αυτό το κεφάλαιο παρατίθονται screenshots από το πρόγραμμα που υλοποιήσαμε καθώς και τα τμήματα κώδικα με εξήγηση για το καθένα.



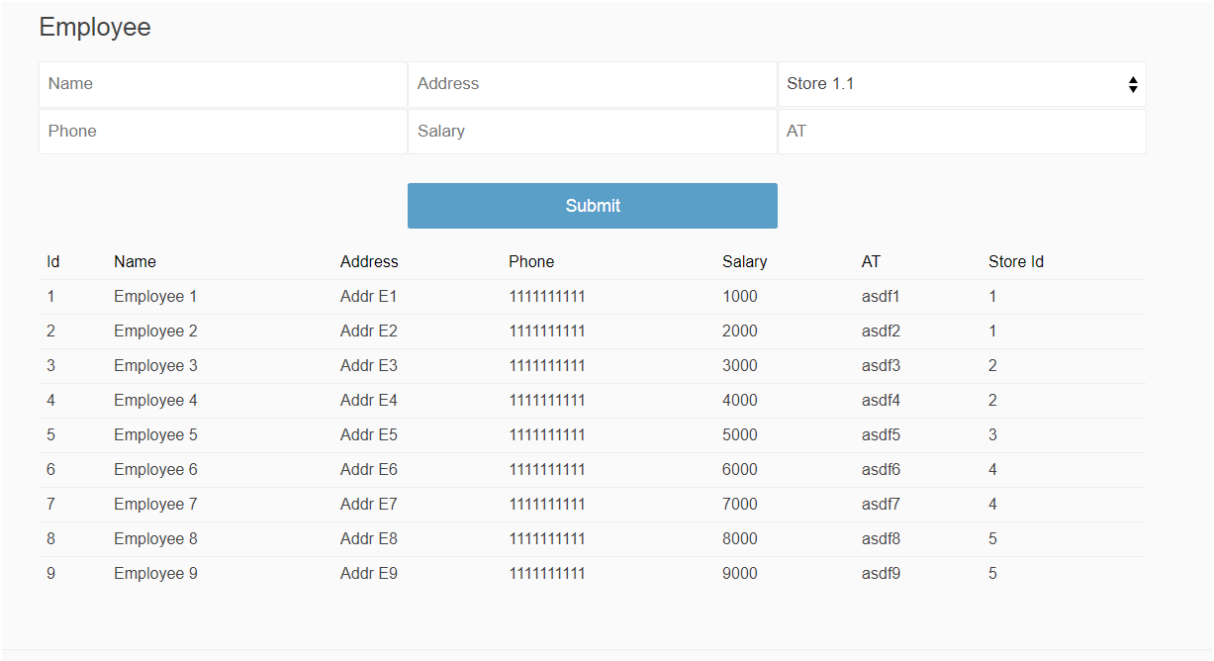
Εικόνα με την οποία αρχίζει η εφαρμογή μας. Με το συγκεκριμένο κουμπί Submit μπορούμε να εισάγουμε ένα προϊόν. Συγκεκριμένα μπορούμε να εισάγουμε τον τίτλο, τον τύπο του προϊόντος, τον κατασκευαστή και την τιμή του.



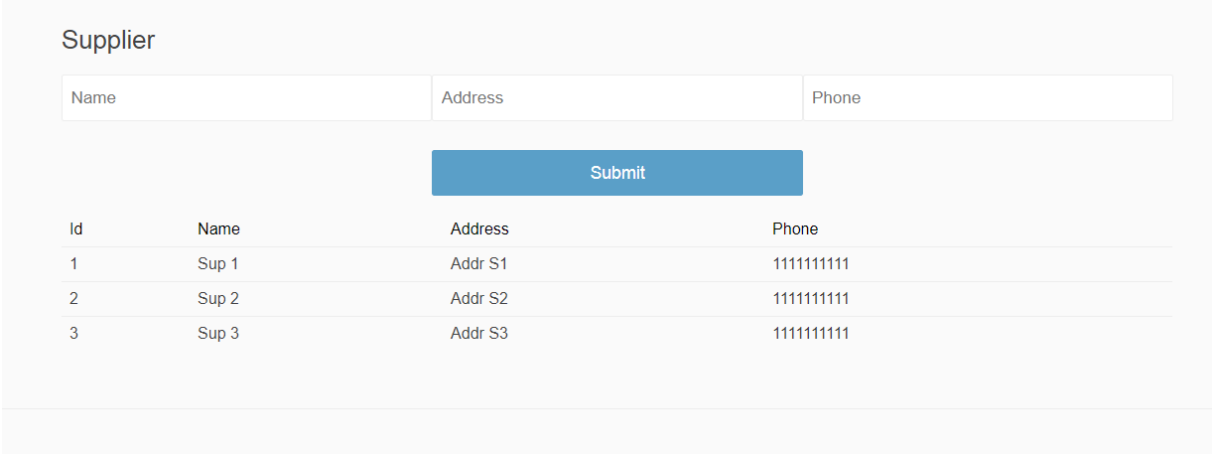
Στη συγκεκριμένη εικόνα εισάγουμε με το κουμπί Submit μια αποθήκη, εισάγοντας τον τίτλο και τη Διεύθυνσή της.



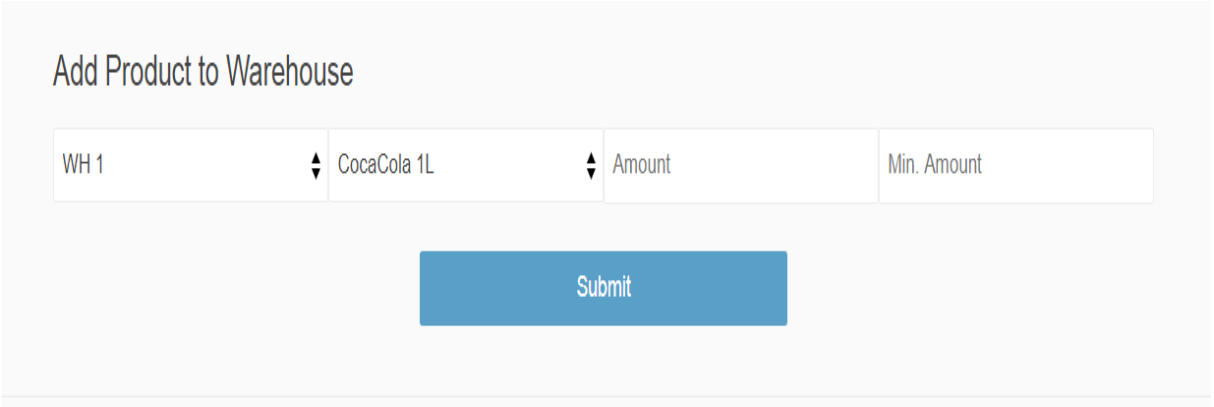
Με το συγκεκριμένο κουμπί Submit εισάγει ο χρήστης τα καταστήματα supermarket. Εισάγει συγκεκριμένα τον τίτλο, τη Διεύθυνση και επιλέγει το Id της αποθήκης με την οποία συνδέεται.



Στο συγκεκριμένο τμήμα του προγράμματος ο χρήστης μπορεί να εισάγει έναν εργαζόμενο και ειδικότερα το όνομά του, τη Διεύθυνσή του, το κατάστημα στο οποίο εργάζεται, τον αριθμό τηλεφώνου του, τον μισθό του και τον αριθμό της ταυτότητάς του.

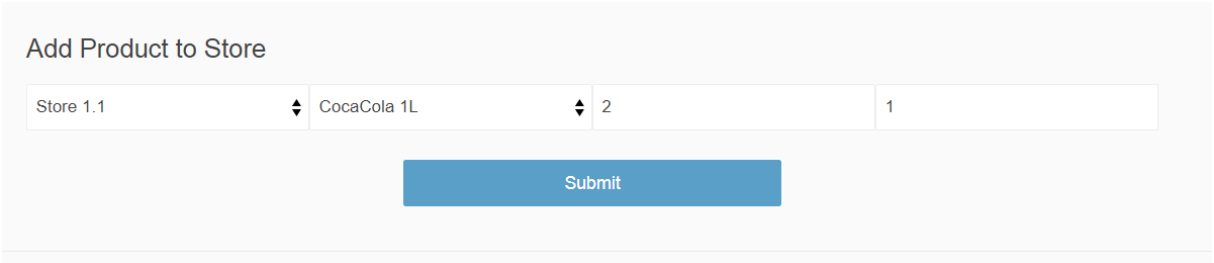


Με το συγκεκριμένο κουμπί ο χρήστης καθίσταται ικανός να εισάγει προμηθευτή. Εισάγει το όνομά του, τη Διεύθυνσή του και τον αριθμό τηλεφώνου του.

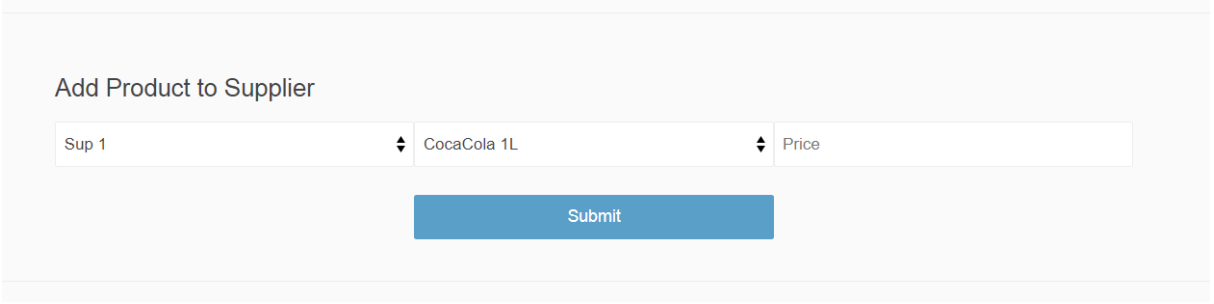


Στο συγκεκριμένο σημείο ο χρήστης εισάγει ένα προϊόν σε μια αποθήκη. Μπορεί να εισάγει τον αναγνωριστικό κωδικό της αποθήκης, το προϊόν που θέλει να αποθηκεύσει, την ποσότητα του προϊόντος και την ελάχιστη ποσότητα προϊόντος που μπορεί να υπάρχει το συγκεκριμένο προϊόν στην αποθήκη.

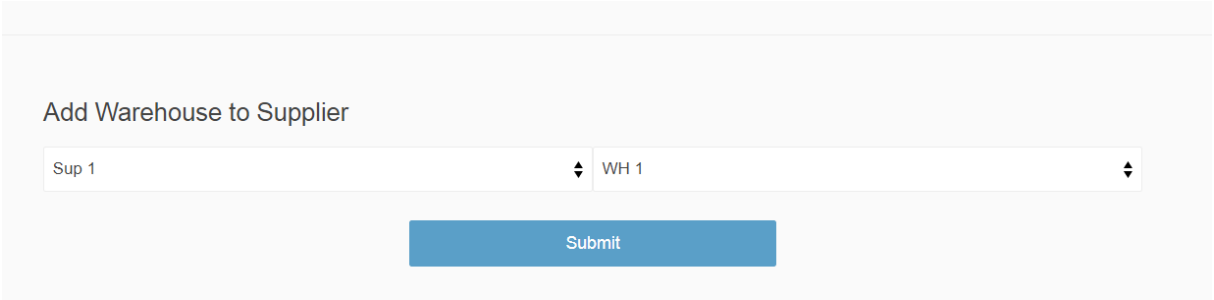
Στην εκφώνηση αναφέρεται ότι αν ένα προϊόν σε μια αποθήκη πέσει κάτω από το min amount, η αποθήκη ξαναζητά από τον προμηθευτή αυτόματα.



Σε αυτό το σημείο ο χρήστης προσθέτει ένα προϊόν σε ένα κατάστημα. Προσθέτει το όνομα του καταστήματος, το προϊόν, την ποσότητα του προϊόντος και την ελάχιστη ποσότητα του προϊόντος σε αυτό το κατάστημα (min_Amount).



Με τη συγκεκριμένη επιλογή προσθέτουμε ένα προϊόν σε έναν προμηθευτή. Προσθέτουμε το όνομα του προμηθευτή στον οποίο θέλουμε να το αντιστοιχίσουμε, το όνομα του προϊόντος και την τιμή του.



Με αυτή την τελευταία επιλογή προσθέτουμε μια αποθήκη σε ένα προμηθευτή. Συγκεκριμένα προσθέτουμε το όνομα του προμηθευτή και το όνομα της αποθήκης.

```php
<?php
require "admin.php";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$fun = $_GET["function"];
```

Το συγκεκριμένο κομμάτι κώδικα παρουσιάζει τη σύνδεση με τη βάση δίνοντας το όνομα του server, το username, το password και το όνομα της βάσης. Αν δεν πραγματοποιηθεί για κάποιο λόγο η σύνδεση θα εμφανιστεί το μήνυμα Connection failed. Στη συνέχεια καλείται η μέθοδος GET.

```php
$fun = $_GET["function"];
switch ($fun) {
case "product":
    add_product($conn);
    break;
case "warehouse":
    add_wh($conn);
    break;
case "supermarket":
    add_supermarket($conn);
    break;
case "employee":
    add_employee($conn);
    break;
case "supplier":
    add_supplier($conn);
    break;
case "whentry":
    add_wh_entry($conn);
    break;
case "storeentry":
    add_store_entry($conn);
    break;
case "supplierentry":
    add_supplier_entry($conn);
    break;
case "whtosupplier":
    add_wh_to_supplier($conn);
    break;
default:
    echo "err";
}
```

Στο παραπάνω τμήμα κώδικα ανάλογα με το αν βλέπει προϊόν, employee, αποθήκη και γενικά κάποια κατηγορία από το supermarket θα καλεί την ανίστοιχη μέθοδο για να προσθέτει την αντίστοιχη κατηγορία στο supermarket.

```php
function add_product($conn) {
    $title = $_GET["title"];
    $type  = $_GET["type"];
    $manuf = $_GET["manuf"];
    $price = $_GET["price"];
    $sql = 'insert into product (title, type, manufacturer, price) values (' .
        '"' . $title . '",' .
        '"' . $type  . '",' .
        '"' . $manuf . '",' .
        $price . ')';

    $r = $conn->query($sql);
    echo $r;
}

function add_wh($conn) {
    $title = $_GET["title"];
    $addr  = $_GET["addr"];
    $sql = 'insert into warehouse (title, address) values (' .
        '"' . $title . '",' .
        '"' . $addr  . '")';

    $r = $conn->query($sql);
    echo $r;
}
```

Όπως φαίνεται παραπάνω η add_product προσθέτει τα χαρακτηριστικά του προϊόντος όπως τίτλος, τύπος και τα εισάγει στη βάση δεδομένων.

Η ad_wh προσθέτει την αποθήκη στη βάση δεδομένων.

```
function add_supermarket($conn) {
    $title = $_GET["title"];
    $addr  = $_GET["addr"];
    $wh    = $_GET["wh"];
    $sql = 'insert into store (warehouse_id, title, address) values (' .
        $wh .          ',' .
        '"' . $title . '",' .
        '"' . $addr  . '")';

    $r = $conn->query($sql);
    echo $r;
}

function add_employee($conn) {
    $name   = $_GET["name"];
    $addr   = $_GET["addr"];
    $store  = $_GET["store"];
    $phone  = $_GET["phone"];
    $salary = $_GET["salary"];
    $at     = $_GET["at"];
    $sql = 'insert into employee (store_id, name, address, phone, salary, `at`) values (' .
        $store  .          ',' .
        '"'     . $name    . '",' .
        '"'     . $addr    . '",' .
        '"'     . $phone   . '",' .
        $salary .          ',' .
        '"'     . $at      . '")';

    $r = $conn->query($sql);
    echo $r;
}
```

Η πρώτη μέθοδος της εικόνας αποθηεύει το κατάστημα στη βάση, η δεύτερη αποθηκεύει τον εργαζόμενο.

```
function add_supplier($conn) {
    $name  = $_GET["name"];
    $addr  = $_GET["addr"];
    $phone = $_GET["phone"];
    $sql = 'insert into supplier (name, address, phone) values (' .
        '"' . $name  . '",' .
        '"' . $addr  . '",' .
        '"' . $phone . '")';

    $r = $conn->query($sql);
    echo $r;
}

function add_wh_entry($conn) {
    $wh       = $_GET["wh"];
    $product  = $_GET["product"];
    $amount   = $_GET["amount"];
    $min      = $_GET["min"];
    $sql = 'insert into warehouse_entry (warehouse_id, product_id, amount, min_amount) values (' .
        $wh      . ',' .
        $product . ',' .
        $amount  . ',' .
        $min     . ')';

    $r = $conn->query($sql);
    echo $r;
}
```

Η μέθοδος add_supplier προσθέτει τον προμηθευτή στη βάση και η μέθοδος add_wh_entry προσθέτει την ποσότητα της αποθήκης στη βάση.

```php
function add_store_entry($conn) {
    $store    = $_GET["store"];
    $product  = $_GET["product"];
    $amount   = $_GET["amount"];
    $min      = $_GET["min"];
    $sql = 'insert into store_stock (store_id, product_id, amount, min_amount) values (' .
        $store   . ',' .
        $product . ',' .
        $amount  . ',' .
        $min     . ')';

    $r = $conn->query($sql);
    echo $r;
}

function add_supplier_entry($conn) {
    $supplier = $_GET["supplier"];
    $product  = $_GET["product"];
    $price    = $_GET["price"];
    $sql = 'insert into supplier_product (supplier_id, product_id, price) values (' .
        $supplier . ',' .
        $product  . ',' .
        $price    . ')';

    $r = $conn->query($sql);
    echo $r;
}

function add_wh_to_supplier($conn) {
    $supplier = $_GET["supplier"];
    $wh = $_GET["wh"];
    $sql = 'insert into supplier_warehouse (supplier_id, warehouse_id) values (' .
        $supplier . ',' .
        $wh       . ')';

    $r = $conn->query($sql);
```

Τέλος η add_store_entry προσθέτει την ποσότητα στο κατάστημα, η add_supplier_entry προσθέτει την ποσότητα στον προμηθευτή και η τελευταία μέθοδος προσθέτει την ποσότητα του προϊόντος στην αποθήκη του προμηθευτή.

```php
<?php
$servername = "localhost";
$username = "root";
$password = "rootpassword";
$dbname = "s";
?>
```

Το αρχείο admin.php περιλαμβάνει το όνομα του server, το username, το password και το όνομα της βάσης δεδομένων.

```php
<?php
require "admin.php";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Perform SELECT queries and add data to a JSON
// Select all warehouses
$sql = "SELECT * FROM warehouse";
$result = $conn->query($sql);
$rows = array();
while ($r = $result->fetch_assoc()) {
    $rows[] = $r;
}
$json = '{"warehouses":' . json_encode($rows);

// Select all stores
$sql = "SELECT * FROM store";
$result = $conn->query($sql);
$rows = array();
while ($r = $result->fetch_assoc()) {
    $rows[] = $r;
}
$json .= ',"stores":' . json_encode($rows);
```

Στο αρχείο get.php, όπως φαίνεται παραπάνω δημιουργείται και πάλι η σύνδεση με τη βάση της mysql και πραγματοποιούνται τα διάφορα select από τη βάση όπως τα select από την αποθήκη και τα καταστήματα.

```php
// Select all products
$sql = "SELECT * FROM product";
$result = $conn->query($sql);
$rows = array();
while ($r = $result->fetch_assoc()) {
    $rows[] = $r;
}
$json .= ',"products":' . json_encode($rows);

// Select all suppliers
$sql = "SELECT * FROM supplier";
$result = $conn->query($sql);
$rows = array();
while ($r = $result->fetch_assoc()) {
    $rows[] = $r;
}
$json .= ',"suppliers":' . json_encode($rows);

// Select all employees
$sql = "SELECT * FROM employee";
$result = $conn->query($sql);
$rows = array();
while ($r = $result->fetch_assoc()) {
    $rows[] = $r;
}
$json .= ',"employees":' . json_encode($rows) . '}';

// Output constructed JSON
echo $json;

// Close connection
$conn->close();
?>
```

Το παρακάτω αρχείο init.sql αποτελεί τον κώδικα για τη δημιουργία της βάσης. Αν υπάρχουν ήδη τα tables τα κάνει drop, δηλαδή τα σβήνει. Στη συνέχεια δημιουργεί ένα table για κάθε κατηγορία οντότητας στην αλυσίδα supermarket. Δηλαδή δημιουργεί table για την αποθήκη, την παραγγελία, τον προμηθευτή κλπ.

```
use s;

/*
 * Drop old tables
 */
drop table if exists store_stock;
drop table if exists supplier_warehouse;
drop table if exists supplier_product;
drop table if exists warehouse_entry;
drop table if exists `order`;
drop table if exists supplier;
drop table if exists employee;
drop table if exists store;
drop table if exists warehouse;
drop table if exists product;

/*
 * Create basic tables
 */
create table product (
    id int primary key auto_increment,
    title varchar(255) not null,
    type varchar(255) not null,
    manufacturer varchar(255) not null,
    price double not null
);
```

```sql
create table warehouse (
    id int primary key auto_increment,
    title varchar(255) not null,
    address varchar(255) not null
);

create table store (
    id int primary key auto_increment,
    warehouse_id int not null,
    title varchar(255) not null,
    address varchar(255) not null,
    foreign key (warehouse_id) references warehouse(id)
);

create table employee (
    id int primary key auto_increment,
    store_id int not null,
    name varchar(255) not null,
    address varchar(255) not null,
    phone varchar(255) not null,
    salary double not null,
    `at` varchar(10) not null,
    foreign key (store_id) references store(id)
);

create table supplier (
    id int primary key auto_increment,
    name varchar(255) not null,
    address varchar(255) not null,
    phone varchar(255) not null
);
```

Στη συνέχεια δημιουργεί τα relations που προκύπτουν μεταξύ των οντοτήτων.

```
create table `order` (
    id int primary key auto_increment,
    product_id int not null,
    supplier_id int not null,
    amount int not null,
    foreign key (product_id) references product(id),
    foreign key (supplier_id) references supplier(id)
);

/*
 * Create relations
 */
create table warehouse_entry (
    warehouse_id int not null,
    product_id int not null,
    amount int not null,
    min_amount int not null,
    primary key (warehouse_id, product_id),
    foreign key (warehouse_id) references warehouse(id),
    foreign key (product_id) references product(id)
);

create table supplier_product (
    supplier_id int not null,
    product_id int not null,
    price double not null,
    primary key (supplier_id, product_id),
    foreign key (supplier_id) references supplier(id),
    foreign key (product_id) references product(id)
);



create table supplier_warehouse (
    supplier_id int not null,
    warehouse_id int not null,
    primary key (supplier_id, warehouse_id),
    foreign key (supplier_id) references supplier(id),
    foreign key (warehouse_id) references warehouse(id)
);

create table store_stock (
    store_id int not null,
    product_id int not null,
    amount int not null,
    min_amount int not null,
    primary key (store_id, product_id),
    foreign key (store_id) references store(id),
    foreign key (product_id) references product(id)
);
```

Στη συνέχεια εισάγουμε στη βάση πειραματικά δεδομένα.

```sql
/*
 * Insert mockup data
 */
insert into product (id, title, type, manufacturer, price) values
(1, "CocaCola 1L", "Drink", "CocaCola", 1.0),
(2, "Soda 0.5L", "Drink", "Soda Waters S.A.", 0.5),
(3, "Doritos", "Snack", "Doritos AC", 2.0),
(4, "Lays", "Snack", "Lays AC", 2.0),
(5, "Captain Morgan's Black Rum", "Drink", "Unknown", 15.0),
(6, "Cleaning Supplies", "Utilities", "Unknown", 9.0);

insert into warehouse (id, title, address) values
(1, "WH 1", "Addr 1"),
(2, "WH 2", "Addr 2"),
(3, "WH 3", "Addr 3");

insert into store (id, warehouse_id, title, address) values
(1, 1, "Store 1.1", "Addr 1.1"),
(2, 1, "Store 1.2", "Addr 1.2"),
(3, 2, "Store 2.1", "Addr 2.1"),
(4, 2, "Store 2.2", "Addr 2.2"),
(5, 3, "Store 3.1", "Addr 3.1");

insert into employee (id, store_id, name, address, phone, salary, `at`) values
(1, 1, "Employee 1", "Addr E1", "1111111111", 1000, "asdf1"),
(2, 1, "Employee 2", "Addr E2", "1111111111", 2000, "asdf2"),
(3, 2, "Employee 3", "Addr E3", "1111111111", 3000, "asdf3"),
(4, 2, "Employee 4", "Addr E4", "1111111111", 4000, "asdf4"),
(5, 3, "Employee 5", "Addr E5", "1111111111", 5000, "asdf5"),
(6, 4, "Employee 6", "Addr E6", "1111111111", 6000, "asdf6"),
(7, 4, "Employee 7", "Addr E7", "1111111111", 7000, "asdf7"),
(8, 5, "Employee 8", "Addr E8", "1111111111", 8000, "asdf8"),
(9, 5, "Employee 9", "Addr E9", "1111111111", 9000, "asdf9");
```

```
insert into supplier (id, name, address, phone) values
(1, "Sup 1", "Addr S1", "1111111111"),
(2, "Sup 2", "Addr S2", "1111111111"),
(3, "Sup 3", "Addr S3", "1111111111");

insert into warehouse_entry (warehouse_id, product_id, amount, min_amount) values
(1, 1, 21, 5), (1, 2, 20, 5), (1, 3, 20, 5), (1, 4, 20, 5), (1, 5, 20, 5), (1, 6, 20, 5),
(2, 1, 22, 5), (2, 2, 20, 5), (2, 4, 20, 5), (2, 5, 20, 5),
(3, 1, 23, 5), (3, 2, 20, 5), (3, 3, 20, 5), (3, 6, 20, 5);

insert into supplier_product (supplier_id, product_id, price) values
(1, 1, 10), (1, 2, 10), (1, 3, 10), (1, 4, 10), (1, 5, 10), (1, 6, 10),
(2, 1, 12), (2, 2, 15), (2, 3, 11), (2, 4, 12), (2, 6, 10),
(3, 1, 8),  (3, 2, 17), (3, 3, 11), (3, 4, 14), (3, 5, 10);

insert into supplier_warehouse (supplier_id, warehouse_id) values
(1, 1), (1, 2), (1, 3),
(2, 1),
(3, 2), (3, 3);

insert into store_stock (store_id, product_id, amount, min_amount) values
(1, 1, 10, 5), (1, 2, 10, 5), (1, 3, 10, 5), (1, 4, 10, 5), (1, 5, 10, 5), (1, 6, 10, 5),
(2, 1, 10, 5), (2, 2, 10, 5), (2, 4, 10, 5), (2, 5, 10, 5),
(3, 1, 10, 5), (3, 2, 10, 5), (3, 3, 10, 5), (3, 6, 10, 5),
(4, 1, 10, 5), (4, 2, 10, 5), (4, 3, 10, 5), (4, 6, 10, 5),
(5, 1, 10, 5), (5, 2, 10, 5), (5, 3, 10, 5), (5, 6, 10, 5);
```

Τέλος υλοποιούμε τα triggers όπως φαίνονται παρακάτω.

```
/*
 * Triggers
 */
drop trigger if exists store_bupd_trigger;
delimiter //
create trigger store_bupd_trigger before update on store_stock for each row
begin
    declare wh_amount int;
    declare wh_id int;

    if NEW.amount < NEW.min_amount then
        set @sid = NEW.store_id;
        set @pid = NEW.product_id;
        set @smin = NEW.min_amount;
        set @sval = NEW.amount;

        select warehouse_id into wh_id from store where id=@sid;
        select amount into wh_amount from warehouse_entry where warehouse_id=wh_id and product_id=@pid;

        set @amount_to_order = (@smin * 2) - @sval;

        update warehouse_entry set amount=amount - @amount_to_order where warehouse_id=wh_id and product_id=@pid;
        set NEW.amount=@sval + @amount_to_order;
    end if;
end//
delimiter ;
```

```
drop trigger if exists warehouse_entry_bupd_trigger;
delimiter //
create trigger warehouse_entry_bupd_trigger before update on warehouse_entry for each row
begin
    declare cheapest_supplier_id int;
    set @val = NEW.amount;
    set @min = NEW.min_amount;
    set @wid = NEW.warehouse_id;
    set @pid = NEW.product_id;

    if @val < @min then
        -- Find cheapest supplier
        select supplier_id into cheapest_supplier_id from supplier_product
            where product_id=@pid
            and supplier_id in (select supplier_id from supplier_warehouse where warehouse_id=@wid)
            order by price asc
            limit 1;

        set @amount_to_order = (@min * 2) - @val;
        set NEW.amount = @val + @amount_to_order;
        insert into `order` (product_id, supplier_id, amount) values (@pid, cheapest_supplier_id, @amount_to_order);
    end if;
end//
delimiter ;
```

Στο index.html αρχείο παρατίθεται ο κώδικας html για την παρουσίαση της web εφαρμογής μας.

```
<html>
  <head>
    <title>Supermarket</title>

    <script src="js/main.js"></script>
    <link rel="stylesheet" type="text/css" href="https://cdn.rawgit.com/doximity/vital/v2.2.1/dist/css/vital.min.css">
  </head>
  <body onload="onLoad()">
    <div class="row">
      <div class="section">
        <h1>Hello!</h1>
      </div>
    </div>
    <hr>

    <!-- Product -->
    <div class="row">
      <div class="section">
        <h2>Product</h2></h2>
        <div class="full-width-forms">
          <div class="col-1-4"><label><input id="ptitle" placeholder="Title" name="title"></label></div>
          <div class="col-1-4"><label><input id="ptype" placeholder="Type" name="type"></label></div>
          <div class="col-1-4"><label><input id="pmanu" placeholder="Manufacturer" name="manufacturer"></label></div>
          <div class="col-1-4"><label><input id="pprice" placeholder="Price" type="number"></label></div>
          <div class="clear"></div>
          <br/>
          <div class="col-1-3 space"></div>
          <div class="col-1-3"><label><input onclick="onAddProductClicked()" class="btn solid blue" type="submit" value="Submit"></label></div>
          <div class="col-1-3 space"></div>
          <div class="clear"></div>
        </div>

        <table id="products-table">
          <thead>
```

Τέλος παραθέτουμε το αρχείο json.

```
/*
 * Takes a json string as input and extracts values from it. That JSON is
 * supposed to be the response JSON from PHP.
 */
function loadValues(jsonText) {
    // Convert json string to json object
    var json = JSON.parse(jsonText);

    // Create variables to store values
    var warehouses = json.warehouses;
    var stores = json.stores;
    var products = json.products;
    var suppliers = json.suppliers;
    var employees = json.employees;

    // Find HTML elements
    var warehouseSelects = document.getElementsByClassName("warehouse-select");
    var storeSelects     = document.getElementsByClassName("store-select");
    var productSelects   = document.getElementsByClassName("product-select");
    var supplierSelects  = document.getElementsByClassName("supplier-select");

    // Fill warehouse selects
    for (i = 0; i < warehouseSelects.length; i++) {
        for (j = 0; j < warehouses.length; j++) {
            var option = document.createElement("option");
            var wh = warehouses[j];
            option.value = wh.id;
            option.text = wh.title;
            warehouseSelects[i].add(option);
        }
    }
```

```
    // Fill product selects
    for (i = 0; i < productSelects.length; i++) {
        for (j = 0; j < products.length; j++) {
            var option2 = document.createElement("option");
            var p = products[j];
            option2.value = p.id;
            option2.text = p.title;
            productSelects[i].add(option2);
        }
    }

    // Fill store selects
    for (i = 0; i < storeSelects.length; i++) {
        for (j = 0; j < stores.length; j++) {
            var option3 = document.createElement("option");
            var s = stores[j];
            option3.value = s.id;
            option3.text = s.title;
            storeSelects[i].add(option3);
        }
    }

    // Fill supplier selects
    for (i = 0; i < supplierSelects.length; i++) {
        for (j = 0; j < suppliers.length; j++) {
            var option4 = document.createElement("option");
            var su = suppliers[j];
            option4.value = su.id;
            option4.text = su.name;
            supplierSelects[i].add(option4);
        }
    }
```

```
// Fill Tables
var productsTable     = document.getElementById("products-table").getElementsByTagName("tbody")[0];
var warehousesTable   = document.getElementById("warehouses-table").getElementsByTagName("tbody")[0];
var supermarketsTable = document.getElementById("supermarkets-table").getElementsByTagName("tbody")[0];
var employeesTable    = document.getElementById("employees-table").getElementsByTagName("tbody")[0];
var suppliersTable    = document.getElementById("suppliers-table").getElementsByTagName("tbody")[0];
var row;
var c0; var c1; var c2; var c3; var c4; var c5; var c6;

// Fill products
for (i = 0; i < products.length; i++) {
    var product = products[i];
    row = productsTable.insertRow(productsTable.rows.length);
    c0 = row.insertCell(0);
    c1 = row.insertCell(1);
    c2 = row.insertCell(2);
    c3 = row.insertCell(3);
    c4 = row.insertCell(4);
    c0.innerHTML = product.id;
    c1.innerHTML = product.title;
    c2.innerHTML = product.type;
    c3.innerHTML = product.manufacturer;
    c4.innerHTML = product.price;
}
```

```
// Fill warehouses
for (i = 0; i < warehouses.length; i++) {
    var warehouse = warehouses[i];
    row = warehousesTable.insertRow(warehousesTable.rows.length);
    c0 = row.insertCell(0);
    c1 = row.insertCell(1);
    c2 = row.insertCell(2);
    c0.innerHTML = warehouse.id;
    c1.innerHTML = warehouse.title;
    c2.innerHTML = warehouse.address;
}

// Fill stores
for (i = 0; i < stores.length; i++) {
    var store = stores[i];
    row = supermarketsTable.insertRow(supermarketsTable.rows.length);
    c0 = row.insertCell(0);
    c1 = row.insertCell(1);
    c2 = row.insertCell(2);
    c3 = row.insertCell(3);
    c0.innerHTML = store.id;
    c1.innerHTML = store.title;
    c2.innerHTML = store.address;
    c3.innerHTML = store.warehouse_id;
}
```

```
// Fill employees
for (i = 0; i < employees.length; i++) {
    var e = employees[i];
    row = employeesTable.insertRow(employeesTable.rows.length);
    c0 = row.insertCell(0);
    c1 = row.insertCell(1);
    c2 = row.insertCell(2);
    c3 = row.insertCell(3);
    c4 = row.insertCell(4);
    c5 = row.insertCell(5);
    c6 = row.insertCell(6);
    c0.innerHTML = e.id;
    c1.innerHTML = e.name;
    c2.innerHTML = e.address;
    c3.innerHTML = e.phone;
    c4.innerHTML = e.salary;
    c5.innerHTML = e.at;
    c6.innerHTML = e.store_id;
}

// Fill suppliers
for (i = 0; i < suppliers.length; i++) {
    var supplier = suppliers[i];
    row = suppliersTable.insertRow(suppliersTable.rows.length);
    c0 = row.insertCell(0);
    c1 = row.insertCell(1);
    c2 = row.insertCell(2);
    c3 = row.insertCell(3);
    c0.innerHTML = supplier.id;
    c1.innerHTML = supplier.name;
    c2.innerHTML = supplier.address;
    c3.innerHTML = supplier.phone;
}
```

```
/*
 * Utility function to retrieve selected value from a <select> element.
 */
function getValueFromSelect(selectId) {
    var select = document.getElementById(selectId);
    var val = select.options[select.selectedIndex].value;
    return val;
}

/*
 * Callback for add product button.
 */
function onAddProductClicked() {
    var title = document.getElementById("ptitle").value;
    var type  = document.getElementById("ptype").value;
    var manuf = document.getElementById("pmanu").value;
    var price = document.getElementById("pprice").value;
    var url = "add.php?function=product&title=" + title + "&type=" + type + "&manuf=" + manuf + "&price=" + price;
    insertToDbAndReload(url);
}

/*
 * Callback for add product button.
 */
function onAddWarehouseClicked() {
    var title = document.getElementById("wtitle").value;
    var addr  = document.getElementById("waddr").value;
    var url = "add.php?function=warehouse&title=" + title + "&addr=" + addr;
    insertToDbAndReload(url);
}
```

```
/*
 * Callback for add supermarket button.
 */
function onAddSupermarketClicked() {
    var title = document.getElementById("stitle").value;
    var addr  = document.getElementById("saddr").value;
    var wh = getValueFromSelect("sselect-wh");
    var url = "add.php?function=supermarket&title=" + title + "&addr=" + addr + "&wh=" + wh;
    insertToDbAndReload(url);
}

/*
 * Callback for add employee button.
 */
function onAddEmployeeClicked() {
    var name = document.getElementById("ename").value;
    var addr = document.getElementById("eaddr").value;
    var store = getValueFromSelect("eselect-store");
    var phone = document.getElementById("ephone").value;
    var salary = document.getElementById("esalary").value;
    var at = document.getElementById("eat").value;
    var url = "add.php?function=employee&name=" + name + "&addr=" + addr + "&store=" + store + "&phone=" + phone + "&salary=" + salary + "&at=" + at;
    insertToDbAndReload(url);
}
```

```
/*
 * Callback for add supplier button.
 */
function onAddSupplierClicked() {
    var name = document.getElementById("ename").value;
    var addr = document.getElementById("eaddr").value;
    var phone = document.getElementById("ephone").value;
    var url = "add.php?function=supplier&name=" + name + "&addr=" + addr + "&phone=" + phone;
    insertToDbAndReload(url);
}

/*
 * Callback for add warehouse entry button.
 */
function onAddWhEntryClicked() {
    var wh = getValueFromSelect("wheselect-wh");
    var product = getValueFromSelect("wheselect-p");
    var amount = document.getElementById("whamount").value;
    var min = document.getElementById("whemin").value;
    var url = "add.php?function=whentry&wh=" + wh + "&product=" + p + "&amount=" + amount + "&min=" + min;
    insertToDbAndReload(url);
}

/*
 * Callback for add store entry button.
 */
function onAddStoreEntryClicked() {
    var store = getValueFromSelect("seselect-store");
    var product = getValueFromSelect("seselect-p");
    var amount = document.getElementById("seamount").value;
    var min = document.getElementById("semin").value;
    var url = "add.php?function=storeentry&store=" + store + "&product=" + p + "&amount=" + amount + "&min=" + min;
    insertToDbAndReload(url);
}
```

```
function onAddProductToSupplierClicked() {
    var supplier = getValueFromSelect("psselect-sup");
    var product = getValueFromSelect("psselect-p");
    var price = document.getElementById("psprice").value;
    var url = "add.php?function=supplierentry&supplier=" + supplier + "&product=" + product + "&price=" + price;
    insertToDbAndReload(url);
}

/*
 * Callback for add-warehouse-to-supplier button.
 */
function onAddWhToSupplierClicked() {
    var supplier = getValueFromSelect("wsselect-sup");
    var wh = getValueFromSelect("wsselect-wh");
    var url = "add.php?function=whtosupplier&supplier=" + supplier + "&wh=" + wh;
    insertToDbAndReload(url);
}

/*
 * Sends a GET request and reloads the page. That GET request is supposed to be on the PHP API.
 * Button callbacks construct URLs to pass on this function.
 */
function insertToDbAndReload(url) {
    var xmlHttp = new XMLHttpRequest();
    xmlHttp.open("GET", url, false); // false for synchronous request
    xmlHttp.send(null);
    location.reload();
}
```

```
/*
 * Callback for initialization. Retrieves values from PHP with an AJAX request.
 */
function onLoad() {
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
            var jsonText = this.responseText;
            loadValues(jsonText);
        }
    };
    xhttp.open("GET", "get.php", true);
    xhttp.send();
}
```

Κεφάλαιο 6ο

Περιορισμοί υλοποίησης και δυνατότητες βελτίωσης

Οι τρέχοντες περιορισμοί του συστήματος, αφορούν κυρίως θέματα ασφάλειας και User Experience. Παρακάτω παρατίθεται μια λίστα με πιθανές μελλοντικές διορθώσεις, σε περίπτωση που παραστεί ανάγκη για επέκταση του συστήματος σε κάτι το οποίο να έχει δυνατότητες εμπορικής αξιοποίησης.

- Αυτή τη στιγμή, τα δεδομένα στέλνονται στην PHP μέσω URL parameters, πράγμα που αποτελεί security risk και καθιστά την εφαρμογή ευάλωτη σε κακόβουλες επιθέσεις. Στο μέλλον θα πρέπει να στέλνονται στην PHP μέσα από την HTTP POST μέθοδο και ίσως να είναι και encrypted.
- Η PHP δεν πραγματοποιεί ελέγχους στα δεδομένα που δέχεται, με αποτέλεσμα να δημιουργείται κενό ασφαλείας ως προς SQL Injection επιθέσεις.
- Η τρέχουσα σχεδίαση της ιστοσελίδας χρήζει άμεσης γραφιστικής παρέμβασης, με στόχο την υλοποίηση καλύτερου User Interface ώστε η όλη εφαρμογή να είναι πιο φιλική προς το χρήστη και εύχρηστη.
- Σε περίπτωση που το πρόγραμμα αποφασιστεί να εξελιχθεί και να γίνει ένα πλήρως εξοπλισμένο πρόγραμμα ERP με διαχείριση αποθήκης, πρέπει να επεκταθεί το σχήμα της βάσης και να προστεθούν παραπάνω περιορισμοί (constraints). Για παράδειγμα, θα μπορούσαν να δημιουργηθεί ένα table με συγκεκριμένους τύπους προϊόντων και να υλοποιηθεί περιορισμός στον πίνακα ” products ”, σύμφωνα με τον οποίο, στο column “type” θα καταχωρούνται μόνο τιμές από τον αντίστοιχο πίνακα.
- Στο μέλλον, για λόγους ασφάλειας αλλά και ταχύτητας/αξιοπιστίας θα χρειαστεί να αντικατασταθεί το σχήμα Apache-PHP με κάποια πιο ολοκληρωμένη λύση που περιλαμβάνει κάποιον Application Server (.Net, Wildfly κλπ..) και επικοινωνεί με το Web Interface μέσω RESTful Web Services.

Βιβλιογραφία

https://el.wikipedia.org

https://www.w3schools.com/html/

https://www.w3schools.com/php/

http://www.dblab.upatras.gr/download/courses/DATABASES%20LABORATORY/2012_13/lect3.pdf

http://www.cs.uoi.gr/~pitoura/courses/db/db09/slides/er09.pdf

Ανάπτυξη Web Εφαρμογών με PHP και MySQL (Welling Thomson).