ΑΝΤΑΛΛΑΓΗ ΧΡΗΣΙΜΩΝ ΠΛΗΡΟΦΟΡΙΩΝ ΒΑΣΙΣΜΕΝΗ ΣΤΗΝ ΑΞΙΟΠΙΣΤΙΑ ΚΑΙ ΣΤΗΝ ΑΝΤΑΠΟΔΟΤΙΚΟΤΗΤΑ: ΕΦΑΡΜΟΓΗ ΣΕ ΤΙΜΕΣ ΠΡΟΪΟΝΤΩΝ ΑΠΟ ΚΑΤΑΣΤΗΜΑΤΑ

ΚΥΡΙΑΚΗ ΜΑΝΟΥΣΑΡΙΔΟΥ

Α.Μ. 03/10

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Επιβλέπων Καθηγητής: ΙΩΑΝΝΗΣ ΡΕΦΑΝΙΔΗΣ

Π.Μ.Σ. Τμήματος Εφαρμοσμένης Πληροφορικής
ΣΥΣΤΗΜΑΤΑ ΥΠΟΛΟΓΙΣΤΩΝ

Πανεπιστήμιο Μακεδονίας
Θεσσαλονίκη

Οκτώβριος 2012

Copyright © Κυριακή Μανουσαρίδου, 2012
Με επιφύλαξη παντός δικαιώματος. All rights reserved.

Η έγκριση της μεταπτυχιακής εργασίας από το Τμήμα Εφαρμοσμένης Πληροφορικής του Πανεπιστημίου Μακεδονίας δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα εκ μέρους του Τμήματος.

ΕΥΧΑΡΙΣΤΙΕΣ

Στο σημείο αυτό θα ήθελα να απευθύνω τις ευχαριστίες μου σε όλους όσους συνέβαλλαν με τον τρόπο τους στην εκπόνηση της διπλωματικής μου εργασίας.

Πρώτα απ' όλα θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή κ. Ιωάννη Ρεφανίδη για την πολύτιμη καθοδήγηση και υποστήριξή του αλλά και για την εμπιστοσύνη που μου έδειξε. Αξίζει να αναφέρω ότι κάθε φορά που χρειάστηκα τη βοήθεια του ήταν εκεί και πάντα έβρισκε μια άμεση λύση στο εκάστοτε πρόβλημα.

Στη συνέχεια θα ήθελα να ευχαριστήσω το Ίδρυμα Μποδοσάκη για την υποτροφία που μου παρείχε προκειμένου να πραγματοποιήσω και να ολοκληρώσω τελικά τις μεταπτυχιακές μου σπουδές. Χωρίς την οικονομική τους υποστήριξη δεν θα βρισκόμουν σήμερα εδώ.

Επίσης οφείλω ένα ευχαριστώ σε αυτούς τους συναδέλφους μου που αποτέλεσαν την «παρέα διαβάσματος» και το στήριγμα μου τις ατέλειωτες ημέρες και ώρες που περάσαμε διαβάζοντας στα εργαστήρια του Πανεπιστημίου Μακεδονίας. Στην έμπνευση της Ευθαλίας Καρύδη, εκτός των άλλων, οφείλω και το όνομα της διαδικτυακής εφαρμογής που δημιουργήθηκε.

Τέλος θέλω να ευχαριστήσω θερμά τους δικούς μου ανθρώπους για τη συμπαράστασή τους, τη διάθεσή τους να με βοηθήσουν με οποιοδήποτε τρόπο και την αμέριστη κατανόησή τους. Την οικογένειά μου την ευχαριστώ επίσης που μου έμαθε να παλεύω για τα όνειρά μου, ξεπερνώντας κάθε εμπόδιο και βασιζόμενη πάνω απ' όλα στις δικές μου δυνάμεις.

ΠΕΡΙΛΗΨΗ

Την εποχή του Διαδικτύου και της απεριόριστης πληροφόρησης είναι δύσκολος ο εντοπισμός και η επαναχρησιμοποίηση χρήσιμων πληροφοριών. Μια από τις χρήσιμες πληροφορίες που είναι δύσκολο να εντοπιστεί είναι η πληροφορία που αφορά τις τιμές των προϊόντων των σούπερ μάρκετ. Έχουν ήδη αναπτυχθεί εφαρμογές που παρέχουν στο χρήστη τιμές από σούπερ μάρκετ οι οποίες καταγράφηκαν από τους διαχειριστές των εφαρμογών ή από το υπουργείο ανάπτυξης. Η παρούσα διπλωματική εργασία όμως, μελετά τη σχεδίαση και ανάπτυξη ενός συστήματος ανταλλαγής πληροφοριών σχετικών με τιμές σουπερ μάρκετ μεταξύ των χρηστών, λαμβάνοντας υπ' όψιν την αξιοπιστία του κάθε χρήστη και επιβραβεύοντας την ανταποδοτικότητα του. Το σύστημα επιτρέπει στο χρήστη να εισάγει μια λίστα προϊόντων, να επιλέξει τα καταστήματα από τα οποία επιθυμεί να ψωνίσει και να καθορίσει το μέγιστο αριθμό των καταστημάτων που μπορούν να συνδυαστούν σε μια λίστα. Στη συνέχεια, μετά την αγορά των προϊόντων, ζητάει από το χρήστη να αξιολογήσει τις προτάσεις που δέχτηκε. Το σύστημα, μετά από μελέτη των απαιτήσεων και των διαθέσιμων τεχνολογιών, αναπτύχθηκε ως Υπηρεσία Ιστού και Διαδικτυακή Εφαρμογή που καλεί την υπηρεσία. Η Υπηρεσία Ιστού υλοποιήθηκε σε γλώσσα Java και μέσα από το περιβάλλον Eclipse δημοσιεύεται με τη χρήση του Apache Tomcat και του Axis2. Η Διαδικτυακή Εφαρμογή υλοποιήθηκε σε γλώσσα PHP με τη χρήση του CodeIgniter και με τη βοήθεια των γλωσσών HTML, CSS, JavaScript και των χαρτών της Google.

Λέξεις κλειδιά: τιμές σουπερμάρκετ, υπηρεσία ιστού, διαδικτυακή εφαρμογή, αξιοπιστία, ανταποδοτικότητα.

ABSTRACT

In the time of Internet and the unlimited amount of information provided, it is difficult to find and reuse any useful information. One kind of information difficult to find is the information regarding the prices of supermarket products. There have already been developed applications that provide the user with supermarket prices as they were recorded by the administrators of the application or by the Greek Ministry of Development. The current master thesis, however, considers the design and development of a system for exchanging information about supermarket prices between users, taking into account the reliability of each user and rewarding his reciprocity. The system allows the user to insert a list of the products he wants to buy, choose the supermarkets he is willing to go and define the maximum number of the supermarkets that will be combined to form the list. Then, after the purchase of the products, it asks the user to evaluate the suggestions that were made to him. After the study of the requirements and available technologies the system was developed as a Web Service and a Web Application that calls the service. The Web Service was implemented in Java language and is deployed through Eclipse with the use of Apache Tomcat and Axis2. The Web Application was implemented in PHP language using the CodeIgniter framework, the languages HTML, CSS, JavaScript and Google maps.

Keywords: supermarket prices, web service, web application, reliability, reciprocity.

ΠΕΡΙΕΧΟΜΕΝΑ

1 Εισαγωγή 12
1.1 Σκοπός 12
1.2 Δομή της εργασίας 13
2 Σχετικές Εφαρμογές 15
2.1 Shop & Save 15
2.2 Παρατηρητήριο τιμών 16
2.3 Superlista.com 18
2.4 Kill Deal 19
2.5 Live Supermarket 20
2.6 My Supermarket 20
2.7 Σύγκριση Ελληνικών Εφαρμογών 21
3 Τεχνολογίες 24
3.1 Υπηρεσίες Ιστού (Web Services) 24
3.1.1 Ορισμός 24
3.1.2 Χρήση 25
3.1.3 Τεχνολογίες Υπηρεσιών Ιστού 26
3.1.3.1 XML 26
3.1.3.2 Απλό Πρωτόκολλο Πρόσβασης Αντικειμένου (SOAP) 26
3.1.3.3 Γλώσσα Περιγραφής Υπηρεσίας Ιστού (WSDL) 27
3.1.3.4 Καθολική Περιγραφή, Ανακάλυψη και Ένταξη (UDDI) 28
3.1.4 Μια τυπική κλήση Υπηρεσίας Ιστού 28
3.2 Εφαρμογές Διαδικτύου (Web applications) 29
3.2.1 Πρότυπο σχεδίασης Model / View / Controller (MVC) 30
3.2.2 PHP 33
3.2.2.1 CodeIgniter 34
3.2.3 Apache HTTP 34
3.2.4 MySQL 35
3.2.5 XAMPP 36
3.2.6 HTML 36

3.2.7 CSS 37
3.2.8 JavaScript 37
3.2.8.1 JQuery 38
3.2.8.2 AJAX 38
3.3 Σημασιολογικός Ιστός 40
3.3.1 Οντολογίες 40
3.3.2 Resource Description Framework (RDF) 41
3.3.3 RDF Schema 43
3.3.4 Ontology Web Language (OWL) 43
3.3.5 Jena 44
3.3.6 SPARQL 45
3.3.7 TopBraid Composer 45
4 Ανάλυση 46
4.1 Ανάλυση απαιτήσεων 46
4.1.1 Λειτουργίες συστήματος 47
4.1.2 Περιπτώσεις Χρήσης 48
4.2 Ανάλυση υλοποίησης 51
5 Σχεδίαση και Υλοποίηση 54
5.1 Βάση Δεδομένων 54
5.1.1 Σχήμα Βάσης Δεδομένων 54
5.1.2 Συσχετίσεις μεταξύ πινάκων 58
5.1.3 Οντολογία κατηγοριών 60
5.2 Εγκατάσταση όλων των εφαρμογών 62
5.2.1 Διαδικτυακή εφαρμογή 62
5.2.1.1 XAMPP 62
5.2.1.2 MySQL Workbench 63
5.2.1.3 CodeIgniter 63
5.2.2 Υπηρεσίες Ιστού 65
5.2.2.1 Eclipse 65
5.2.2.2 Apache Tomcat στο Eclipse 66
5.2.2.3 Axis2 68
5.2.2.4 Δημιουργία Υπηρεσίας Ιστού στο Eclipse 70

5.2.2.5 Jena 74
5.2.2.6 SoapUI 74
5.3 Δημιουργία εφαρμογής 74
5.3.1 Υπηρεσία Ιστού 75
5.3.1.1 Περιγραφή κλάσεων 76
5.3.2 Διαδικτυακή εφαρμογή 87
5.3.2.1 Περιγραφή των αρχείων 89
6 Παρουσίαση εφαρμογής 95
6.1 Χρήστης 95
6.2 Διαχειριστής 103
7 Συμπεράσματα 105
7.1 Συμπεράσματα 105
7.2 Προτάσεις 106
ΒΙΒΛΙΟΓΡΑΦΙΑ 108

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

Εικόνα 1: Κλήση μιας Υπηρεσίας Ιστού 29
Εικόνα 2: Τα συστατικά του προτύπου MVC [13]..... 31
Εικόνα 3: Παράδειγμα λειτουργίας εφαρμογής που εφαρμόζει το πρότυπο MVC 32
Εικόνα 4: Χρήση των server-side γλωσσών προγραμματισμού στους ιστότοπους 33
Εικόνα 5: Διάγραμμα μεριδίου αγοράς δημοφιλέστερων Εξυπηρετητών Παγκόσμιου Ιστού (Αύγουστος 1995 – Σεπτέμβριος 2012)..... 35
Εικόνα 6: Σύγχρονη διαδικτυακή επικοινωνία 39
Εικόνα 7: Ασύγχρονη διαδικτυακή επικοινωνία 40
Εικόνα 8: Αρχιτεκτονική του συστήματος 53
Εικόνα 9: Διάγραμμα Βάσης Δεδομένων..... 54
Εικόνα 10: Η Οντολογία των Προϊόντων. 61
Εικόνα 11: Η σελίδα "localhost" μετά την εγκατάσταση του XAMPP..... 63
Εικόνα 12: Ο φάκελος του CodeIgniter 64
Εικόνα 13: Οι Ρυθμίσεις για το περιβάλλον εκτέλεσης του εξυπηρετητή στο Eclipse 66
Εικόνα 14: Προσθήκη εξυπηρετητή..... 67
Εικόνα 15: Θέτοντας τη διαδρομή του Apache..... 67
Εικόνα 16: Επιτυχής εγκατάσταση του εξυπηρετητή..... 68
Εικόνα 17:Οι ρυθμίσεις για το Axis2..... 69
Εικόνα 18: Διαδρομή του φακέλου Axis2-1.6.2 69
Εικόνα 19: Επιλογή τύπου νέου project 71
Εικόνα 20:Εισαγωγή πληροφοριών καινούριου project..... 72
Εικόνα 21: Επιλογή τεχνολογιών για το καινούριο project..... 73
Εικόνα 22:Δημιουργία της Υπηρεσίας Ιστού..... 73
Εικόνα 23:Επιλογή του Axis2 74
Εικόνα 24: Κλάσεις του προγράμματος Java - Υπηρεσία Ιστού..... 75
Εικόνα 25: Δημοσιευμένες συναρτήσεις της Υπηρεσίας Ιστού 84
Εικόνα 26: Ο κατάλογος αρχείων της διαδικτυακής εφαρμογής 88
Εικόνα 27: Αρχική σελίδα του Super Basket 95

Εικόνα 28: Εγγραφή χρήστη 96
Εικόνα 29: Αρχική οθόνη εγγεγραμμένων χρηστών 97
Εικόνα 30: Εισαγωγή λίστας αγορών 98
Εικόνα 31: Φτηνότερη λίστα για έναν bronze κι έναν silver χρήστη..... 98
Εικόνα 32: Εισαγωγή Προϊόντος 99
Εικόνα 33: Εισαγωγή σούπερ μάρκετ 100
Εικόνα 34: Εισαγωγή τιμής προϊόντος 101
Εικόνα 35: Εύρεση κοντινότερων σούπερ μάρκετ 101
Εικόνα 36: Σελίδα προφίλ χρήστη 102
Εικόνα 37: Σύνδεση χρήστη 102
Εικόνα 38: Αξιολόγηση προηγούμενης λίστας 103
Εικόνα 39: Εισαγωγή τιμής για μια ολόκληρη επωνυμία 104

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

Πίνακας 1: Σύγκριση των ελληνικών εφαρμογών για τις τιμές των σούπερ μάρκετ 23
Πίνακας 2: Ποσοστά μεριδίου αγοράς δημοφιλέστερων εξυπηρετητών παγκοσμίου ιστού (Αύγουστος 2012 – Σεπτέμβριος 2012) 35
Πίνακας 3 : Οι ιδιωτικές συναρτήσεις της Υπηρεσίας Ιστού 87

ΚΕΦΑΛΑΙΟ 1

1 Εισαγωγή

Καθημερινά γινόμαστε δέκτες τεράστιων ποσοτήτων πληροφορίας, η περισσότερη χωρίς ιδιαίτερη χρησιμότητα. Υπάρχουν όμως πληροφορίες οι οποίες έχουν μεγάλη χρησιμότητα όχι μόνο για εμάς αλλά και για άλλους. Για παράδειγμα, η ύπαρξη μιας προσφοράς ημέρας σε κάποιο κατάστημα ενδεχομένως θα ενδιέφερε πολύ κόσμο. Για το λόγο αυτό θεωρήθηκε πολύ ενδιαφέρουσα η σχεδίαση και ανάπτυξη ενός συστήματος για εύκολη ανταλλαγή τέτοιου είδους πληροφορίας, δηλαδή χρήσιμης, επίκαιρης, παροδικής. Θεωρήθηκε επίσης σημαντικό να μπορούν οι χρήστες του συστήματος να προσθέτουν εύκολα πληροφορίες στο σύστημα και (κυρίως) να τις εντοπίζουν.

1.1 Σκοπός

Ο σκοπός της παρούσας εργασίας είναι να παρουσιάσει τη σχεδίαση και υλοποίηση του συστήματος ανταλλαγής πληροφοριών για τιμές προϊόντων σε σούπερ μάρκετ.

Επιλέχθηκαν τα σούπερ μάρκετ καθώς αποτελούν το πιο συχνό και μεγάλο έξοδο για όλα τα νοικοκυριά. Ειδικά αυτούς τους δύσκολους καιρούς της οικονομικής κρίσης, ένα σύστημα που θα μπορούσε να συγκεντρώσει, να συνδυάσει και να παρουσιάσει τις καλύτερες τιμές για μια λίστα προϊόντων αποτελεί ένα χρήσιμο εργαλείο για κάθε χρήστη ανεξαρτήτως ηλικίας, μορφωτικού επιπέδου, οικογενειακής και οικονομικής κατάστασης. Απευθυνόμενο σε ένα ευρύ κοινό το σύστημα αυτό πρέπει να είναι εύκολο στη χρήση και προσπελάσιμο από όλους όσους διαθέτουν έναν υπολογιστή και σύνδεση στο Διαδίκτυο. Επίσης θα ήταν χρήσιμο να είναι ευέλικτο και επεκτάσιμο σε άλλες συσκευές εκτός των υπολογιστών.

1.2 Δομή της εργασίας

Μετά από μια αναζήτηση στο Διαδίκτυο ανακαλύφθηκαν παρόμοιες εφαρμογές, είτε ελληνικές είτε ξένες, οι οποίες έχουν σκοπό την παρουσίαση των καλύτερων τιμών προϊόντων σε σούπερ μάρκετ. Οι εφαρμογές αυτές παρουσιάζονται και συγκρίνονται στο Κεφάλαιο Σχετικές Εφαρμογές).

Στη συνέχεια στο Κεφάλαιο 3 (Τεχνολογίες) αναλύονται οι Τεχνολογίες που μελετήθηκαν για την ανάπτυξη του συστήματος. Λόγω της επιλογής για ευελιξία και επεκτασιμότητα σε άλλες συσκευές (όπως τα έξυπνα τηλέφωνα) η λειτουργικότητα του συστήματος αναπτύχθηκε ως Υπηρεσία Ιστού η οποία μπορεί να κληθεί από πελάτη οποιασδήποτε πλατφόρμας και λειτουργικού συστήματος. Εκτός από την Υπηρεσία Ιστού δημιουργήθηκε και η Διαδικτυακή Εφαρμογή που θα την καλεί και θα παρέχει στο χρήστη ένα εύκολο στη χρήση γραφικό περιβάλλον. Για τον ευκολότερο εντοπισμό πληροφοριών σχετικά με τα προϊόντα χρησιμοποιήθηκε και η τεχνολογία των οντολογιών του σημασιολογικού ιστού εκτός από τη χρήση της βάσης δεδομένων.

Μετά τη μελέτη των τεχνολογιών ακολουθεί η ανάλυση του συστήματος στο Κεφάλαιο 4 (Ανάλυση). Στο κεφάλαιο αυτό αναλύονται οι απαιτήσεις και οι λειτουργίες του συστήματος. Η λειτουργικότητα του συστήματος περιγράφεται αναλυτικά με τη μορφή κειμένου και με τη μορφή περιπτώσεων χρήσης.

Την ανάλυση του συστήματος ακολούθησε η σχεδίαση της βάσης δεδομένων και της οντολογίας, η εγκατάσταση των εφαρμογών και εργαλείων που χρησιμοποιήθηκαν για την ανάπτυξη τόσο της Υπηρεσίας Ιστού όσο και της Διαδικτυακής Εφαρμογής και τέλος η υλοποίηση του συστήματος. Η Υπηρεσία Ιστού υλοποιήθηκε σε γλώσσα Java, στο περιβάλλον της Eclipse και δημοσιεύεται με τη χρήση του Apache Tomcat και του Axis2, ενώ η Διαδικτυακή Εφαρμογή υλοποιήθηκε σε γλώσσα PHP με τη χρήση του CodeIgniter framework σε συνδυασμό με τις γλώσσες JavaScript (βιβλιοθήκη JQuery), HTML, CSS και την τεχνολογία των χαρτών της Google. Όλα αυτά καθώς και επιπλέον πληροφορίες σχετικά με τη σχεδίαση και υλοποίηση της εφαρμογής παρουσιάζονται στο Κεφάλαιο 5 (Σχεδίαση και Υλοποίηση).

Η τελική εφαρμογή όπως μπορεί να χρησιμοποιηθεί από το χρήστη παρουσιάζεται στο Κεφάλαιο 6 (Παρουσίαση εφαρμογής).

Τέλος στο Κεφάλαιο 7 (Συμπεράσματα) παρουσιάζονται τα συμπεράσματα και οι προτάσεις για μελλοντικές βελτιώσεις ή επεκτάσεις του συστήματος.

ΚΕΦΑΛΑΙΟ 2

2 Σχετικές Εφαρμογές

Στο κεφάλαιο αυτό θα παρουσιαστούν οι ήδη υπάρχουσες εφαρμογές που αφορούν τις τιμές προϊόντων στα σούπερ μάρκετ και τη σύγκριση αυτών. Οι εφαρμογές που παρουσιάζονται είναι διαδικτυακές και βρέθηκαν με τη βοήθεια της μηχανής αναζήτησης της Google. Αρχικά θα παρουσιαστούν οι κυριότερες ελληνικές εφαρμογές και στη συνέχεια ενδεικτικά κάποιες του εξωτερικού.

2.1 Shop & Save

http://www.shopnsave.gr

Το ShopnSave.gr πρόκειται για μια διαδικτυακή ελληνική εφαρμογή η οποία εμφανίζει τις καλύτερες τιμές προϊόντων και τις προσφορές καταστημάτων. Ο χρήστης μπορεί να επιλέξει ένα συγκεκριμένο δήμο της Ελλάδος και να δει για κάθε επωνυμία σούπερ μάρκετ ποια είναι τα φτηνότερα προϊόντα σε σχέση με όλους τους ανταγωνιστές της και τι ποσό και ποσοστό κέρδους θα προκύψει από την αγορά τους. Μπορεί να μειώσει τα αποτελέσματα της αναζήτησης είτε διαλέγοντας μια συγκεκριμένη επωνυμία καταστημάτων είτε ακόμα και διαλέγοντας μια κατηγορία προϊόντων. Έχοντας επιλεγμένο το δήμο που επιθυμεί, ο χρήστης μπορεί επίσης να αναζητήσει ένα συγκεκριμένο προϊόν για να δει σε ποιο κατάστημα πωλείται φτηνότερα, ποια είναι αυτή η καλύτερη τιμή και το κέρδος σε σχέση με τη μέση τιμή της αγοράς. Κάθε προϊόν που επιθυμεί να αγοράσει ο χρήστης μπορεί να προστεθεί στο καλάθι. Στο καλάθι εμφανίζεται η λίστα με τα προϊόντα και την τιμή τους στην κάθε αλυσίδα καταστημάτων, ενώ επιπλέον εμφανίζονται τα συνολικά ποσά κάθε αλυσίδας. Οι αλυσίδες στις οποίες δεν υπάρχει ενημέρωση τιμών για όλα τα προϊόντα της λίστας εμφανίζουν μια ειδική σήμανση στο τελικό τους σύνολο.

Για τη χρήση της εφαρμογής ο χρήστης δεν είναι υποχρεωμένος να εγγραφεί. Παρόλα αυτά αν εγγραφεί θα μπορεί να αποθηκεύει το καλάθι με τα προϊόντα που επιθυμεί. Όπως αναφέρεται στην ιστοσελίδα «Οι τιμοληψίες βασίζονται στα επίσημα δεδομένα του Υπουργείου Ανάπτυξης». Τις προσφορές των καταστημάτων τις εντοπίζουν οι ίδιοι και συγκεκριμένα «οι κατάσκοποί τους». Η ενημέρωση των τιμών γίνεται μια φορά την εβδομάδα ενώ υπάρχει η δυνατότητα μέσω του forum οι εγγεγραμμένοι χρήστες να αποστέλλουν αποδείξεις από τις αγορές τους στους διαχειριστές. Επίσης παρέχεται η δυνατότητα σε καταστήματα να καταχωρήσουν τα προϊόντα τους ή να διαφημιστούν στην ιστοσελίδα.

Αξιολογώντας την εφαρμογή ως προς τις παρεχόμενες υπηρεσίες θα συμπεριελάμβανε κανείς στα θετικά το γεγονός ότι εμφανίζει για κάθε προϊόν το κέρδος που θα προκύψει από την αγορά του, το ότι εμφανίζει για κάθε αλυσίδα εβδομαδιαίες προσφορές που οι ίδιοι κρίνουν αξιόλογες και ότι για κάθε προϊόν εμφανίζεται και η εικόνα του γεγονός που κάνει πιο εύκολη την αναγνώρισή του από το χρήστη.

Επιπλέον δυνατότητες που θα μπορούσαν να προσφέρουν μεγαλύτερη ευκολία στο χρήστη θα ήταν η εμφάνιση των υποκαταστημάτων κάθε αλυσίδας η οποία θα μπορούσε να συνοδεύεται με την απεικόνισή τους σε χάρτες.

2.2 Παρατηρητήριο τιμών

http://www.e-prices.gr

Πρόκειται για το εργαλείο που παρουσίασε επίσημα το Δεκέμβριο του 2010 η Γενική Γραμματεία Εμπορίου (Γ.Γ.Ε) και παρέχει τη δυνατότητα παρακολούθησης και σύγκρισης των τιμών πώλησης για πάνω από 1500 προϊόντα σε 1200 σημεία πώλησης, σε 13 Περιφέρειες, σε 1-4 πόλεις ανά Περιφέρεια. Οι αρμόδιες υπηρεσίες της Γ.Γ.Ε ενημερώνονται σε εβδομαδιαία ή μηνιαία βάση (ανάλογα με το προϊόν) για τις τιμές των προϊόντων που διαθέτουν τα σούπερ μάρκετ και τα αναρτούν άμεσα στην ηλεκτρονική διεύθυνση του Παρατηρητηρίου [1]. Για τη χρήση του εργαλείου δεν απαιτείται εγγραφή αλλά ούτε υποστηρίζεται από το σύστημα αυτή η λειτουργία.

Ο χρήστης καλείται να αναζητήσει και να επιλέξει τα προϊόντα που επιθυμεί να αγοράσει, να τα προσθέσει στο καλάθι και στη συνέχεια να επιλέξει τα σημεία πώλησης που τον ενδιαφέρουν.

Η αναζήτηση ενός προϊόντος γίνεται βάσει της κατηγορίας ή του ονόματος του προϊόντος. Επιλέγοντας μια κατηγορία επιστρέφεται η λίστα με όλα τα προϊόντα της κατηγορίας και τη διακύμανση των τιμών για κάθε προϊόν. Επιλέγοντας το όνομα κάποιου προϊόντος εμφανίζονται οι επωνυμίες καταστημάτων που το πωλούν και οι διακυμάνσεις τιμών για κάθε επωνυμία. Η εισαγωγή των προϊόντων στο καλάθι γίνεται πατώντας στην επιλογή «προσθήκη στο καλάθι». Εκτός από προϊόντα μπορεί να προστεθεί στο καλάθι μια ολόκληρη κατηγορία από την οποία θα επιλεγεί τελικά μόνο το φτηνότερο προϊόν της. Για την αναζήτηση των σημείων πώλησης υπάρχουν δύο επιλογές, βάσει της γεωγραφικής περιοχής ή του ίδιου του σημείου πώλησης. Επιλέγοντας γεωγραφική περιοχή εμφανίζονται όλα τα υποκαταστήματα που ανήκουν στην περιοχή αυτή. Ενώ επιλέγοντας σημείο πώλησης ο χρήστης καλείται να εισάγει το όνομα του καταστήματος και να επιλέξει με βάση τη διεύθυνση του ποιο είναι αυτό που αναζητεί. Αφού επιλεχθούν τόσο τα προϊόντα όσο και τα σημεία πώλησης μπορεί το σύστημα να υπολογίσει τις τιμές πώλησης. Η επιλογή «εμφάνιση τιμών» θα εμφανίσει μια λίστα με όλα τα προϊόντα του καλαθιού και την τιμή του καθενός σε όλα τα σημεία πώλησης. Επίσης εμφανίζει και έναν πίνακα με τα συνολικά ποσά ανά σημείο πώλησης στον οποίο αναφέρεται για κάθε σημείο πώλησης πόσα προϊόντα του καλαθιού βρέθηκαν και ποιο είναι το σύνολο που τους αντιστοιχεί. Τα σημεία πώλησης δεν κατανέμονται με βάση κάποιο κριτήριο στον πίνακα αυτό αλλά με τυχαία σειρά. Μετά την εμφάνιση των τιμών δίνεται η δυνατότητα στο χρήστη να ελέγξει διαγραμματικά την εξέλιξη των τιμών για ένα από τα προϊόντα της λίστας του σε ένα ή περισσότερα από τα σημεία πώλησης που επέλεξε. Επίσης με την επιλογή «φθηνότερη λίστα» ο χρήστης μπορεί να επιλέξει ένα ή περισσότερα από τα σημεία πώλησης και να δει σε ποιο από αυτά είναι φτηνότερο το κάθε προϊόν της λίστας, η τελευταία αυτή λίστα μπορεί να αποθηκευτεί σε αρχείο excel. Τέλος ο χρήστης αν το επιθυμεί μπορεί να αποθηκεύσει το καλάθι του.

Αξιολογώντας την εφαρμογή θα θεωρούνταν θετικό το γεγονός ότι εμφανίζει τα υποκαταστήματα και τις διευθύνσεις για κάθε αλυσίδα σούπερ μάρκετ, καθώς και το ότι μεταξύ των καταστημάτων συμπεριλαμβάνονται λαϊκές αγορές, κρεοπωλεία και ζαχαροπλαστεία. Επίσης είναι ενδιαφέρον χαρακτηριστικό το ότι υπάρχει η δυνατότητα για παρακολούθηση της εξέλιξης των τιμών σε κάθε σημείο πώλησης.

Ένα από τα αρνητικά της εφαρμογής είναι η δυσκολία χρήσης του για τον μέσο χρήστη. Επίσης η εύρεση του φτηνότερου προϊόντος μιας κατηγορίας είναι εφικτή μόνο για κατηγορίες με μικρό αριθμό προϊόντων, καθώς για κατηγορίες με πολλά προϊόντα εμφανίζεται μήνυμα λάθους και δεν υπολογίζονται τιμές για κανένα προϊόν της λίστας.

2.3 Superlista.com

http://www.superlista.com

Το Superlista.com πρόκειται για μια διαδικτυακή εφαρμογή με τη βοήθεια της οποίας ο χρήστης επιλέγει τη λίστα με τα προϊόντα που επιθυμεί να αγοράσει, το σύστημα υπολογίζει πόσο κοστίζει η λίστα του σε κάθε αλυσίδα καταστημάτων και στη συνέχεια αφού ο ίδιος επιλέξει μια αλυσίδα καταστημάτων του εμφανίζονται τα υποκαταστήματα της για να επιλέξει που θα κάνει τις αγορές του. Για να χρησιμοποιήσει κάποιος το σύστημα είναι απαραίτητη η εγγραφή κατά την οποία αποθηκεύεται για κάθε χρήστη εκτός των άλλων και ο νομός στον οποίο επιθυμεί να γίνεται η αναζήτηση των καταστημάτων. Οι τιμές που προτείνει η εφαρμογή προέρχονται από τα φυλλάδια προσφορών των σούπερ μάρκετ και από τη Γραμματεία Γενικού Εμπορίου αλλά όπως αναφέρεται στην ιστοσελίδα στο μέλλον θα προέρχεται από τα ίδια τα σούπερ μάρκετ και οι χρήστες θα επιβεβαιώνουν αν το βρήκαν φτηνότερο ή ακριβότερο.

Για την επιλογή των προϊόντων ο χρήστης μπορεί να εισάγει το όνομα του προϊόντος στη γραμμή αναζήτησης της αρχικής σελίδας ή να επιλέξει μια από τις κατηγορίες που υπάρχουν. Αφού δημιουργηθεί η λίστα το σύστημα αναζητά τις τιμές του κάθε προϊόντος για όλες τις αλυσίδες καταστημάτων. Ο χρήστης επιλέγει την αλυσίδα καταστημάτων που επιθυμεί και στη συνέχεια του εμφανίζονται οι διευθύνσεις των υποκαταστημάτων της αλυσίδας για να

διαλέξει ένα από αυτά. Μπορεί επίσης να εμφανίσει τα υποκαταστήματα στο χάρτη και να επιλέξει μέσω του χάρτη ποιο επιθυμεί. Έχοντας επιλέξει το υποκατάστημα στο οποίο θα πάει ο χρήστης βλέπει τις τιμές του κάθε προϊόντος στις άλλες αλυσίδες, τον χάρτη με το υποκατάστημα που διάλεξε αλλά και προτάσεις για άλλα παρόμοια προϊόντα.

Στα θετικά της εφαρμογής περιλαμβάνονται το βίντεο που εξηγεί σύντομα και συνοπτικά τη διαδικασία που πρέπει να ακολουθήσει ο χρήστης αλλά και η χρήση χαρτών στην οποία φαίνεται η ακριβής θέση του κάθε υποκαταστήματος.

2.4 Kill Deal

http://www.killdeal.gr/

Το Kill Deal σε αντίθεση με τις παραπάνω εφαρμογές πρόκειται για μια μηχανή αναζήτησης των προσφορών της εβδομάδας σε κάθε αλυσίδα σούπερ μάρκετ για όλα τα προϊόντα. Για τη χρήση του δεν απαιτείται εγγραφή από το χρήστη αλλά αν ο ίδιος ο χρήστης το επιθυμεί μπορεί να εγγραφεί στη λίστα για λήψη ενημερωτικών e-mail σχετικών με προσφορές και εκπτώσεις. Η μηχανή αναζήτησης ενημερώνεται καθημερινά.

Ο χρήστης μπορεί να επιλέξει το νομό και την πόλη που επιθυμεί, τα σούπερ μάρκετ, την κατηγορία των προϊόντων ή/και τις μάρκες που επιθυμεί ως κριτήρια στην αναζήτηση προσφορών. Αφού καθορίσει τα φίλτρα της αναζήτησης μπορεί να ξεκινήσει την αναζήτηση. Ως αποτέλεσμα εμφανίζονται τα προϊόντα που είναι σε προσφορά, η προηγούμενη και η νέα τιμή τους, το ποσοστό της έκπτωσης που προκύπτει, η αλυσίδα καταστημάτων στην οποία πωλείται το προϊόν και επίσης πληροφορίες για τους δικαιούχος της προσφοράς. Δικαιούχοι της προσφοράς μπορεί να είναι όλοι οι πελάτες της αλυσίδας ή μόνο όσοι πελάτες διαθέτουν κάρτα μέλους. Εκτός από την αναζήτηση με βάση τα φίλτρα ένας χρήστης μπορεί να επιλέξει από το κάτω μέρος της ιστοσελίδας είτε ένα συγκεκριμένο σούπερ μάρκετ είτε μια συγκεκριμένη κατηγορία είτε μια συγκεκριμένη μάρκα για να προβάλλει όλες τις προσφορές που αφορούν την επιλογή του. Επιλέγοντας μια συγκεκριμένη προσφορά μπορεί να δει ο χρήστης αναλυτικότερα τις πληροφορίες που την αφορούν. Επίσης πατώντας πάνω στο λογότυπο της

αλυσίδας στην οποία πωλείται η συγκεκριμένη προσφορά μπορεί να δει τις διευθύνσεις των υποκαταστημάτων της αλυσίδας.

Η εφαρμογή είναι εύκολη στη χρήση και σχεδόν όλα τα προϊόντα συνοδεύονται από τις φωτογραφίες τους. Επίσης διαθέτει μεγάλο αριθμό προσφορών για τις μεγαλύτερες και γνωστότερες αλυσίδες καταστημάτων. Κάτι που δύσκολα θα ανακαλύψει ο χρήστης είναι ότι αν θέλει να δει τα υποκαταστήματα μιας αλυσίδας σούπερ μάρκετ θα πρέπει αφού επιλέξει την προσφορά να πατήσει στο λογότυπο της αλυσίδας.

2.5 Live Supermarket

http://www.livesupermarket.gr/

Το Live Supermarket αποτελεί άλλη μια διαδικτυακή εφαρμογή που σύμφωνα με την αρχική σελίδα αποσκοπεί στην online σύγκριση τιμών προϊόντων. Ο χρήστης μπορεί να συγκρίνει τις τιμές των προϊόντων σε κάθε κατάστημα, να δημιουργήσει το καλάθι με τα ψώνια του και να δει που κοστίζει φτηνότερα αλλά και να αποθηκεύσει το καλάθι του και να παρακολουθεί αναλυτικά τις μεταβολές του.

Δυστυχώς για τους χρήστες όμως οι υπόλοιπες σελίδες πέρα της αρχικής δεν λειτουργούν. Προσπαθώντας να χρησιμοποιήσει κάποιος την εφαρμογή αντικρίζει συνεχώς μήνυμα σφάλματος το οποίο παροτρύνει τον χρήστη να προσπαθήσει ξανά σε μερικά λεπτά. Ακόμα όμως και μετά από μήνες η κατάσταση της ιστοσελίδας παραμένει ίδια. Προφανώς πρόκειται για μια εφαρμογή που βρίσκεται πλέον εκτός λειτουργίας.

2.6 My Supermarket

http://www.mysupermarket.co.uk/

Το My Supermarket είναι μια διαδικτυακή εφαρμογή η οποία επιτρέπει στο χρήστη να συγκρίνει τις τιμές αλλά και να αγοράζει μέσω διαδικτύου από τα μεγαλύτερα σούπερ μάρκετ της Μεγάλης Βρετανίας. Όσο ο χρήστης προσθέτει προϊόντα στο καλάθι του η εφαρμογή συγκρίνει τις τιμές μεταξύ των καταστημάτων και κάνει προτάσεις αλλαγής καταστήματος ή προϊόντος που μπορεί να οδηγήσουν σε εξοικονόμηση χρημάτων. Η ενημέρωση των τιμών γίνεται καθημερινά σύμφωνα με τα τοπικά καταστήματα.

Για να χρησιμοποιήσει κάποιος την εφαρμογή πρέπει να είναι εγγεγραμμένος χρήστης. Αρχικά επιλέγει το κατάστημα από το οποίο επιθυμεί να αγοράσει, στη συνέχεια επιλέγει τα προϊόντα που επιθυμεί. Η επιλογή ενός προϊόντος μπορεί να γίνει είτε από τις κατηγορίες που αναγράφονται στο πάνω μέρος της σελίδας είτε από το πλαίσιο αναζήτησης προϊόντων. Μετά την επιλογή κάποιου προϊόντος δίνονται οι προτάσεις του συστήματος. Οι προτάσεις αυτές αφορούν είτε την αλλαγή καταστήματος είτε την αλλαγή προϊόντος με άλλο/άλλα παρόμοια για μεγαλύτερη εξοικονόμηση χρημάτων. Καθώς προστίθενται προϊόντα στη λίστα εμφανίζεται συνεχώς και η συνολική τιμή του καλαθιού τόσο στο επιλεγμένο κατάστημα όσο και στα άλλα καταστήματα. Με αυτό τον τρόπο ο χρήστης μπορεί να αλλάξει το κατάστημα με ένα άλλο φτηνότερο. Σε οποιοδήποτε σημείο της διαδικασίας μπορεί ο χρήστης να δει προτάσεις για επιπλέον οικονομία. Αφού ολοκληρώσει το καλάθι αγορών του και πριν το στείλει μπορεί να ελέγξει για κουπόνια που διαθέτει η εφαρμογή. Το καλάθι αγορών μπορεί να σταλεί απευθείας στο σούπερ μάρκετ που τελικά επέλεξε ο χρήστης. Η πληρωμή και η αποστολή των προϊόντων είναι στην αρμοδιότητα του σούπερ μάρκετ. Σε διαφορετική περίπτωση ο χρήστης μπορεί απλά να εκτυπώσει τη λίστα του και να τα αγοράσει ο ίδιος.
Η γενική εντύπωση από την εφαρμογή είναι αρκετά θετική. Είναι εύκολη στη χρήση, παρέχει πολλές προτάσεις για εξοικονόμηση χρημάτων και διευκολύνει αρκετά τους χρήστες.

2.7 Σύγκριση Ελληνικών Εφαρμογών

Στη συνέχεια γίνεται μια σύγκριση των ελληνικών εφαρμογών που αναφέρθηκαν παραπάνω. Στον πίνακα της σύγκρισης οι στήλες αντιπροσωπεύουν την κάθε εφαρμογή και οι γραμμές το κριτήριο με βάση το οποίο συγκρίνονται.

- Εγγραφή χρήστη: εξετάζεται αν για τη χρήση της εφαρμογής είναι απαραίτητη η εγγραφή του χρήστη. Όπως φαίνεται και στον **Πίνακας 1** η εγγραφή είναι απαραίτητη μόνο για τη χρήση της εφαρμογής *Superlista*, είναι προαιρετική για το *Shop n Save* ενώ δεν υπάρχει καν η δυνατότητα εγγραφής για το *Παρατηρητήριο Τιμών* και το *Kill Deal.*

- Φωτογραφίες Προϊόντων: εξετάζεται αν εκτός από το όνομα και τις πληροφορίες για κάθε προϊόν εμφανίζεται και η εικόνα του για ευκολότερη αναγνώριση του από το χρήστη. Οι εικόνες των προϊόντων εμφανίζονται μόνο στις εφαρμογές *Shop n Save* και *Kill Deal*.
- Φτηνότερα κάθε αλυσίδας: το κριτήριο αυτό αφορά την εμφάνιση ή όχι των προϊόντων που μια συγκεκριμένη αλυσίδα καταστημάτων έχει φτηνότερα σε σχέση με όλες τις άλλες. Η δυνατότητα αυτή παρέχεται μόνο από το *Shop n Save* το οποίο επιλέγοντας μια αλυσίδα εμφανίζει τα προϊόντα που αυτή έχει φτηνότερα σε σχέση με τις άλλες αλυσίδες.
- Προσφορές ανά αλυσίδα: το κριτήριο αυτό αφορά την εμφάνιση των προσωρινών προσφορών της κάθε αλυσίδας. Οι προσφορές αυτές είναι συνήθως οι προσφορές που αναγράφονται στα φυλλάδια κάθε αλυσίδας. Η πληροφορία για τις προσφορές παρέχεται μόνο από το *Shop n Save* και το *Kill Deal*.
- Εμφάνιση υποκαταστημάτων: οι εφαρμογές που εμφανίζουν τις διευθύνσεις ή/και τα τηλέφωνα των υποκαταστημάτων κάθε αλυσίδας είναι το *Παρατηρητήριο Τιμών*, η *Superlista* και το *Kill Deal*.
- Φθηνότερο κατηγορίας: σε αυτή την περίπτωση ελέγχεται αν η εφαρμογή μπορεί να δεχτεί μια κατηγορία προϊόντων αντί για συγκεκριμένο προϊόν και να εμφανίσει το φτηνότερο όλης της κατηγορίας. Οι εφαρμογές που παρέχουν αυτή τη λειτουργία είναι το *Παρατηρητήριο Τιμών* και η *Superlista*.
- Εξέλιξη τιμών: από τις εφαρμογές που εξετάζονται η μόνη που δίνει τη δυνατότητα να ελέγξει ο χρήστης την εξέλιξη της τιμής ενός προϊόντος κατά τη διάρκεια των τελευταίων μηνών είναι το *Παρατηρητήριο Τιμών*.
- Τιμοληψία: αφορά τον τρόπο με τον οποίο ενημερώνουν τις τιμές τους οι εφαρμογές. Το *Παρατηρητήριο Τιμών* είναι η εφαρμογή της Γενικής Γραμματείας Εμπορίου (Γ.Γ.Ε.) η οποία συλλέγει τις τιμές τόσο για την ενημέρωση των καταναλωτών μέσω της εφαρμογής όσο και για τη χρήση της βάσης δεδομένων από άλλες εφαρμογές. Το *Shop n Save* αντλεί τις τιμές του από τη Γ.Γ.Ε. αλλά χρησιμοποιεί και δικούς του

τιμολήπτες για να ανακαλύπτουν τυχόν ενδιαφέρουσες προσφορές, το *Superlista* ενημερώνει τις τιμές του από τη Γ.Γ.Ε. και από τα φυλλάδια των σούπερ μάρκετ ενώ το *Kill Deal* ενημερώνεται μόνο από τα φυλλάδια των σούπερ μάρκετ καθώς και το ίδιο προβάλει μόνο τις προσφορές κάθε αλυσίδας, κάτι που δεν υποστηρίζεται από το *Παρατηρητήριο Τιμών*.

	Shop n Save	**Παρατηρητήριο τιμών**	**Superlista**	**Kill Deal**
Εγγραφή χρήστη	Προαιρετικά	Όχι	Υποχρεωτικά	Όχι
Φωτογραφίες προϊόντων	Ναι	Όχι	Όχι	Ναι
Φθηνότερα κάθε αλυσίδας	Ναι	Όχι	Όχι	Όχι
Προσφορές ανά αλυσίδα	Ναι	Όχι	Όχι	Ναι
Εμφάνιση υποκατ/των	Όχι	Ναι	Ναι	Ναι
Φθηνότερο κατηγορίας	Όχι	Ναι	Ναι	Όχι
Εξέλιξη τιμών	Όχι	Ναι	Όχι	Όχι
Τιμοληψία	Γ.Γ.Ε., τιμολήπτες	Γ.Γ.Ε.	Γ.Γ.Ε., Φυλλάδια ΣΜ	Φυλλάδια ΣΜ

Πίνακας 1: Σύγκριση των ελληνικών εφαρμογών για τις τιμές των σούπερ μάρκετ

ΚΕΦΑΛΑΙΟ 3

3 Τεχνολογίες

Στο κεφάλαιο αυτό αναπτύσσονται συνοπτικά οι τεχνολογίες που μελετήθηκαν και χρησιμοποιήθηκαν για την υλοποίηση της εργασίας.

3.1 Υπηρεσίες Ιστού (Web Services)

3.1.1 Ορισμός

Μια *υπηρεσία λογισμικού* (*software service*) είναι κάτι που δέχεται ψηφιακά αιτήματα και επιστρέφει ψηφιακές αποκρίσεις. Με βάση αυτό τον ορισμό μια συνάρτηση της γλώσσας C, ένα αντικείμενο της γλώσσας Java και ένα ερώτημα σε γλώσσα SQL αποτελούν παραδείγματα υπηρεσιών λογισμικού· μια εφαρμογή υπολογιστή μπορεί να θεωρηθεί ως ένα σύνολο από υπηρεσίες λογισμικού. Παλιότερα μια συγκεκριμένη υπηρεσία λογισμικού μπορούσε να χρησιμοποιηθεί μόνο μέσω μιας συγκεκριμένης γλώσσας ή πλατφόρμας και συχνά δεν ήταν προσπελάσιμη μέσω ενός δικτύου. Οι *Υπηρεσίες Ιστού* (*Web Services)* αποτελούν ένα νέο είδος συνιστώσας λογισμικού που είναι ανεξάρτητη γλώσσας, πλατφόρμας και τοποθεσίας. Ο όρος *Υπηρεσίες Ιστού* είναι λίγο παραπλανητικός καθώς πολλοί άνθρωποι πιστεύουν ότι σημαίνει μια υπηρεσία ειδικευμένη για χρήση μέσω του Παγκόσμιου Ιστού. Στην πραγματικότητα ο όρος αποτελεί συντόμευση του *Ιστός από Υπηρεσίες* (*Web of Services*) εννοώντας ότι κατανεμημένες εφαρμογές μπορούν να συντίθεται από έναν ιστό υπηρεσιών λογισμικού με τον ίδιο τρόπο που οι ιστότοποι συντίθεται από έναν ιστό σελίδων HTML. [2]
Ο ορισμός που δίνεται από το W3C Consortium [3] είναι ο εξής «Μια *Υπηρεσία Ιστού* είναι ένα σύστημα λογισμικού σχεδιασμένο να υποστηρίζει την διαλειτουργική αλληλεπίδραση μηχανής προς μηχανή μέσω ενός δικτύου. Η διασύνδεσή της περιγράφεται σε μορφή μηχανικά επεξεργάσιμη (συγκεκριμένα σε WSDL). Άλλα συστήματα αλληλεπιδρούν με την *Υπηρεσία*

Ιστού, με τον τρόπο που υποδεικνύεται στην περιγραφή της, χρησιμοποιώντας μηνύματα SOAP – τυπικά μεταφερόμενα χρησιμοποιώντας HTTP και XML αναπαράσταση σε συνδυασμό με άλλα πρότυπα σχετικά με το διαδίκτυο».

3.1.2 Χρήση

Οι *Υπηρεσίες Ιστού* έχουν δύο τύπους χρήσης:

- Επαναχρησιμοποίηση μέρους εφαρμογών
  Υπάρχουν πράγματα που οι εφαρμογές χρειάζονται συνέχεια, επομένως δεν υπάρχει λόγος να τα φτιάξει κάποιος ξανά. Οι *Υπηρεσίες Ιστού* μπορούν να προσφέρουν μέρη εφαρμογών όπως : μετατροπείς νομισμάτων χωρών, ενημέρωση καιρού ή ακόμα και μετάφραση γλώσσας ως υπηρεσίες.
- Σύνδεση λογισμικών που ήδη υπάρχουν
  Οι *Υπηρεσίες Ιστού* μπορούν να βοηθήσουν στο πρόβλημα διαλειτουργικότητας δίνοντας στις διάφορες εφαρμογές τρόπο να συνδέσουν τα δεδομένα τους. Με τις *Υπηρεσίες Ιστού* μπορεί κάποιος να ανταλλάξει δεδομένα μεταξύ διαφορετικών εφαρμογών και διαφορετικών πλατφορμών. [4]

Σύμφωνα με τον δεύτερο τύπο χρήσης, οι *Υπηρεσίες Ιστού* είναι ένας τρόπος να διαθέσουμε τη λειτουργικότητα ενός πληροφοριακού συστήματος μέσω πρότυπων τεχνολογιών διαδικτύου. Η χρήση πρότυπων τεχνολογιών μειώνει την ετερογένεια και διευκολύνει την ενοποίηση των εφαρμογών [5]. Με αυτό τον τρόπο οι *Υπηρεσίες Ιστού* επιτρέπουν στους οργανισμούς να κοινοποιούν τα δεδομένα τους χωρίς να γνωρίζουν ο ένας τα πληροφοριακά συστήματα του άλλου. Επίσης αποτελούν έναν τρόπο για να επικοινωνούν οι επιχειρήσεις μεταξύ τους αλλά και με τους πελάτες τους. Αντίθετα με το παραδοσιακό μοντέλο πελάτη/εξυπηρετητή (client/server), όπως είναι ένας *εξυπηρετητής του παγκόσμιου ιστού* (*Web server*) ή ένα δίκτυο σελίδων συστήματος, οι *Υπηρεσίες Ιστού* δεν παρέχουν στον χρήστη μια Γραφική Διασύνδεση Χρήστη (Graphical User Interface). Οι εφαρμογές

αλληλεπιδρούν μεταξύ τους και όχι με τους χρήστες. Οι προγραμματιστές μπορούν να προσθέσουν την *Υπηρεσία Ιστού* σε μια Γραφική Διασύνδεση Χρήστη (όπως μια ιστοσελίδα ή ένα εκτελέσιμο πρόγραμμα) για να προσφέρουν συγκεκριμένη λειτουργικότητα στους χρήστες.[6]

3.1.3 Τεχνολογίες Υπηρεσιών Ιστού

Οι τεχνολογίες που χρησιμοποιεί μια Υπηρεσία Ιστού περιγράφονται παρακάτω. Η *XML* χρησιμοποιείται για να επισημάνει τα δεδομένα, το *SOAP* χρησιμοποιείται για να μεταφέρει τα δεδομένα, η *WSDL* για να περιγράψει τις διαθέσιμες υπηρεσίες και η *UDDI* για την καταγραφή των διαθέσιμων υπηρεσιών. [6]

3.1.3.1 XML

Η *XML* (*Extensible Markup Language*) αποτελεί μια γενικότερου – σκοπού έκδοση της HTML που επιτρέπει την αναπαράσταση κάθε είδους δεδομένου με έναν απλό και μεταφέρσιμο τρόπο. Χρησιμοποιώντας *ετικέτες* (*tags*) που δηλώνουν τη σημασία των δεδομένων τα XML έγγραφα είναι αυτό-περιγραφόμενα και εύκολα στο χειρισμό και την μετατροπή. [2] Η *XML* επιτρέπει τη δημιουργία προσαρμοσμένων ετικετών επιτρέποντας έτσι τον ορισμό, τη μετάδοση, την επικύρωση και την ερμηνεία δεδομένων μεταξύ εφαρμογών και μεταξύ οργανισμών. [6]

Ως μεταφορά δεδομένων, εξαλείφει κάθε συσχετισμό δικτύου, λειτουργικού συστήματος ή πλατφόρμας που έχει ένα πρωτόκολλο [7], μπορεί να χρησιμοποιηθεί μεταξύ διαφορετικών γλωσσών προγραμματισμού και να εκφράσει ακόμα και περίπλοκα μηνύματα και συναρτήσεις. Για το λόγο αυτό μαζί το πρωτόκολλο HTTP, που είναι το πρωτόκολλο που χρησιμοποιείται περισσότερο στο διαδίκτυο, αποτελούν τη βασική πλατφόρμα των *Υπηρεσιών Ιστού*. [4]

3.1.3.2 Απλό Πρωτόκολλο Πρόσβασης Αντικειμένου (SOAP)

Οι *Υπηρεσίες Ιστού* χρησιμοποιούν την *XML* για να κωδικοποιούν και να αποκωδικοποιούν δεδομένα και το *Απλό Πρωτόκολλο Πρόσβασης Αντικειμένου* (*Simple Object Access Protocol – SOAP*) για να τα μεταφέρουν

[4]. Πρόκειται για ένα ελαφρύ, βασισμένο στην *XML* πρωτόκολλο ανταλλαγής μηνυμάτων που χρησιμοποιείται για να κωδικοποιεί τις πληροφορίες στα μηνύματα ερωτήσεων και απαιτήσεων μιας *Υπηρεσίας Ιστού* πριν τα στείλει μέσω του δικτύου. Τα *μηνύματα SOAP* είναι ανεξάρτητα από οποιοδήποτε λειτουργικό σύστημα ή πρωτόκολλο και μπορούν να μεταφερθούν χρησιμοποιώντας μια ποικιλία πρωτοκόλλων Διαδικτύου όπως HTTP, SMTP και FTP. [6]

Με τη χρήση του *SOAP*, οι ετερογενείς πελάτες και εξυπηρετητές μπορούν να γίνουν διαλειτουργικοί καθώς έχουν έναν καθορισμένο μηχανισμό μεταφοράς. [7]

3.1.3.3 Γλώσσα Περιγραφής Υπηρεσίας Ιστού (WSDL)

Η *Γλώσσα Περιγραφής Υπηρεσίας Ιστού* (*Web Service Description Language – WSDL*) αποτελεί μια τεχνολογία XML που περιγράφει τη διασύνδεση μιας *Υπηρεσίας Ιστού* με έναν πρότυπο τρόπο και αναπτύχθηκε από κοινού από τη Microsoft και την IBM. [7]

Η *WSDL* τυποποιεί τον τρόπο που μια *Υπηρεσία Ιστού* αναπαριστά τις παραμέτρους εισόδου και εξόδου μιας εξωτερικής κλήσης, τη δομή των συναρτήσεων, τη φύση της κλήσης (μόνο εισόδου, εισόδου/εξόδου κτλ.) και τον συσχετισμό με το πρωτόκολλο της υπηρεσίας. Με αυτό τον τρόπο επιτρέπει σε ανόμοιους πελάτες να καταλαβαίνουν αυτόματα πώς να αλληλεπιδράσουν με την υπηρεσία ιστού [7].

Οι δυνατότητες μιας *Υπηρεσίας Ιστού* περιγράφονται, μέσω αυτής της XML-μορφής γλώσσας, ως συλλογές από τελικά σημεία επικοινωνίας ικανά να ανταλλάσσουν μηνύματα [6]. Η *WSDL* περιγράφει την *Υπηρεσία Ιστού* ξεκινώντας με αυτά τα μηνύματα που ανταλλάσσονται μεταξύ των αιτούντων και των προμηθευτών πρακτόρων. Οι περιγραφές μιας *Υπηρεσίας Ιστού* μπορούν να αντιστοιχιστούν σε οποιαδήποτε γλώσσα υλοποίησης, πλατφόρμα, μοντέλο αντικειμένου ή σύστημα ανταλλαγής μηνυμάτων. Απλές επεκτάσεις της υπάρχουσας υποδομής Διαδικτύου μπορούν να υλοποιήσουν *Υπηρεσίες Ιστού* για αλληλεπίδραση μέσω προγράμματος περιήγησης (browser) ή απευθείας μέσω μιας εφαρμογής. Εφόσον ο αποστολέας και ο

παραλήπτης συμφωνήσουν στην περιγραφή της υπηρεσίας (π.χ. το αρχείο WSDL), οι υλοποιήσεις πίσω από τις *Υπηρεσίες Ιστού* μπορεί να είναι οτιδήποτε. [3]

Η *WSDL* αποτελεί ένα αναπόσπαστο μέρος της *UDDI*, που αποτελεί ένα βασισμένο στην XML παγκόσμιο μητρώο επιχειρήσεων. Συγκεκριμένα είναι η γλώσσα που χρησιμοποιεί η *UDDI*.

3.1.3.4 Καθολική Περιγραφή, Ανακάλυψη και Ένταξη (UDDI)

Η *Καθολική Περιγραφή, Ανακάλυψη και Ένταξη* (*Universal Description, Discovery and Integration – UDDI*) παρέχει ένα παγκόσμιο μητρώο από *Υπηρεσίες Ιστού* με σκοπό τη διαφήμιση, ανακάλυψη και ένταξη. Οι αναλυτές επιχειρήσεων και οι τεχνολόγοι χρησιμοποιούν τη *UDDI* για να ανακαλύψουν διαθέσιμες υπηρεσίες ιστού αναζητώντας για ονόματα, αναγνωριστικά, κατηγορίες ή τις προδιαγραφές που έχουν υλοποιηθεί από την υπηρεσία ιστού. Η *UDDI* παρέχει μια δομή για την εκπροσώπηση επιχειρήσεων, επιχειρηματικών σχέσεων, υπηρεσιών ιστού, προδιαγραφών μεταδεδομένων και σημείων πρόσβασης υπηρεσιών ιστού. [7]

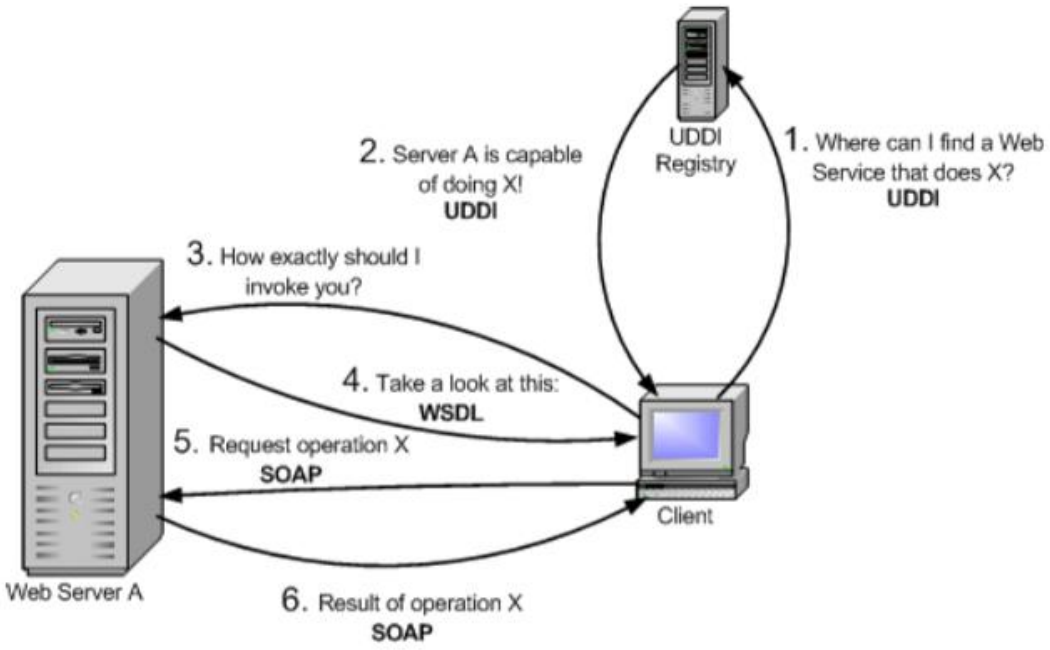
Αποτελεί έναν κατάλογο βασισμένο στον ιστό που επιτρέπει στις επιχειρήσεις να καταγραφούν στο Διαδίκτυο και να ανακαλύπτουν η μία την άλλη, παρόμοια με τους παραδοσιακούς τηλεφωνικούς καταλόγους. [6]

3.1.4 Μια τυπική κλήση Υπηρεσίας Ιστού

Στη συνέχεια θα περιγραφεί μια τυπική κλήση μιας Υπηρεσίας Ιστού όπως παρουσιάζεται στην **Εικόνα 1**. [8]

1. Ο πελάτης μπορεί να μην έχει καμία γνώση για το τι *Υπηρεσία Ιστού* θα καλέσει. Έτσι το πρώτο βήμα θα είναι να βρει μια *Υπηρεσία Ιστού* που ικανοποιεί τις απαιτήσεις του. Για παράδειγμα μπορεί να ενδιαφέρεται στον εντοπισμό μιας *Υπηρεσίας Ιστού* που μπορεί να δώσει την θερμοκρασία σε μια πόλη. Για να γίνει αυτό αρχικά επικοινωνεί με το μητρώο *UDDI*.

2. Το μητρώο *UDDI* θα απαντήσει ποιοι εξυπηρετητές μπορούν να παρέχουν την υπηρεσία που ζητήθηκε (π.χ. η θερμοκρασία στις ελληνικές πόλεις)
3. Ο πελάτης πλέον γνωρίζει την τοποθεσία της *Υπηρεσίας Ιστού* αλλά δεν ξέρει πώς ακριβώς πρέπει να την καλέσει. Το όνομα της συνάρτησης μπορεί να είναι οποιοδήποτε. Για αυτό το λόγο πρέπει να ζητήσει απ την *Υπηρεσία Ιστού* την περιγραφή της.
4. Η *Υπηρεσία Ιστού* απαντά με τη περιγραφή της σε *WSDL*.
5. Πλέον ο πελάτης γνωρίζει που βρίσκεται η υπηρεσία και πώς να την καλέσει. Η κλήση γίνεται μέσω ενός *αιτήματος SOAP* που ζητάει τη θερμοκρασία μια συγκεκριμένης πόλης.
6. Η *Υπηρεσία Ιστού* απαντά με μια *απόκριση SOAP* που περιλαμβάνει τη θερμοκρασία που ζητήθηκε ή ίσως ένα μήνυμα λάθους αν το *αίτημα SOAP* δεν ήταν σωστό.



Εικόνα 1: Κλήση μιας Υπηρεσίας Ιστού

3.2 Εφαρμογές Διαδικτύου (Web applications)

Μια εφαρμογή Διαδικτύου είναι οποιαδήποτε εφαρμογή χρησιμοποιεί ένα πρόγραμμα περιήγησης ιστού ως *πελάτη*. Ο όρος «*πελάτης*» αναφέρεται στο

πρόγραμμα που χρησιμοποιεί το άτομο που εκτελεί την εφαρμογή. Στα περιβάλλοντα *πελάτη/ εξυπηρετητή* ο *πελάτης* είναι η εφαρμογή που χρησιμοποιείται για την εισαγωγή πληροφορίας και ο *εξυπηρετητής* είναι η εφαρμογή που χρησιμοποιείται για να αποθηκεύει την πληροφορία. Μιας και ο *πελάτης* εκτελείται σε ένα πρόγραμμα περιήγησης ιστού ο προγραμματιστής απαλλάσσεται από την ευθύνη να αναπτύξει ένα *πρόγραμμα πελάτη* για κάθε πλατφόρμα ή λειτουργικό σύστημα.

Οι εφαρμογές Διαδικτύου συνήθως χρησιμοποιούν ένα συνδυασμό *κώδικα από πλευράς εξυπηρετητή* (ASP, PHP κτλ) και *κώδικα από πλευράς πελάτη* (HTML, JavaScript, etc) για την ανάπτυξη της εφαρμογής. Ο *κώδικας από πλευράς πελάτη* αναλαμβάνει την παρουσίαση της πληροφορίας ενώ ο *κώδικας από την πλευρά του χρήστη* αναλαμβάνει την αποθήκευση και ανάκτηση της πληροφορίας.[9]

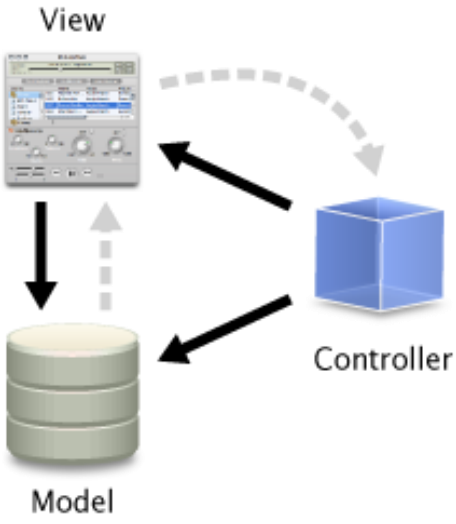
3.2.1 Πρότυπο σχεδίασης Model / View / Controller (MVC)

Είναι λογικό να λεχθεί ότι κάθε εφαρμογή είναι πιθανό να αλλάξει τη διασύνδεσή της στο πέρασμα του χρόνου ή ότι θα έχει πολλές διασυνδέσεις σε οποιαδήποτε χρονική στιγμή. Ωστόσο η υποκείμενη εφαρμογή μπορεί να είναι αρκετά σταθερή. Μια εφαρμογή που εκτελούνταν μέσω γραμμής εντολών μπορεί να είναι ακριβώς ίδια αν σήμερα εκτελείται μέσω Γραφικής Διασύνδεσης Χρήστη (GUI). Η ανάπτυξη μια συγκεκριμένης διασύνδεσης μέσα στην εφαρμογή θα ήταν επιζήμια τόσο για την εφαρμογή , η οποία θα γινόταν λιγότερο εύκαμπτη και δυσκολότερο να μεταφερθεί, όσο και για τη διασύνδεση , η οποία θα μπορούσε να χρησιμοποιηθεί δυσκολότερα από άλλες εφαρμογές. Επομένως είναι λογικό να κρατηθεί η ουσία της εφαρμογής χωριστή από οποιαδήποτε και όλες τις διασυνδέσεις της. Με αυτό τον τρόπο πιθανές αλλαγές της σχεδίασης της γραφικής διασύνδεσης χρήστη δεν θα επηρεάσουν τη σχεδίαση της εφαρμογής. [10]

Το *πρότυπο MVC* χρησιμοποιείται για τη σχεδίαση εφαρμογών με Γραφική Διασύνδεση Χρήστη (GUI) σε αντικειμενοστραφής γλώσσες. Δημιουργήθηκε αρχικά με το Smalltalk-80 αλλά έχει χρησιμοποιηθεί για να περιγράψει προσεγγίσεις σχεδίασης στην ανάπτυξη προγραμμάτων γραφικής

διασύνδεσης χρήστη σε διάφορες γλώσσες προγραμματισμού. Η βασική ιδέα είναι ο διαχωρισμός των διασυνδέσεων χρήστη από τα δεδομένα που παρουσιάζει η ίδια η διασύνδεση χρήστη. [11], [12]

Συχνά το πρότυπο παρουσιάζεται να απαιτεί από τον προγραμματιστή να χρησιμοποιήσει τρεις τύπους αντικειμένων για να αναπτύξει το σύστημα του: *Μοντέλα* (Models), *Όψεις* (Views) και *Ελεγκτές* (Controllers). Τα *Μοντέλα* περιέχουν τα δεδομένα του συστήματος, όπως επίσης και τις μεθόδους για το χειρισμό των δεδομένων. Οι *Όψεις* παρουσιάζουν τα δεδομένα στον χρήστη. Οι *Ελεγκτές* επεξεργάζονται τις εισόδους του χρήστη, ενημερώνουν το *Μοντέλο* και την *Όψη* κατάλληλα και αναφέρονται κυρίως στον κώδικα που χειρίζεται συνολικά τις ενέργειες του συστήματος. Η *Όψη* μαζί με τον *Ελεγκτή* αποτελεί τη διασύνδεση χρήστη της εφαρμογής. Το *Μοντέλο* περιέχει τόσο τις πληροφορίες που παρουσιάζονται από την *Όψη* όσο και τη λογική που αλλάζει αυτές τις πληροφορίες ως απόκριση στην αλληλεπίδραση του χρήστη. Αυτή η προσέγγιση ξεχωρίζει τρεις βασικές έγνοιες του συστήματος και επομένως βοηθάει στην επαναχρησιμοποίηση και συντήρηση του κώδικα. [11], [12]

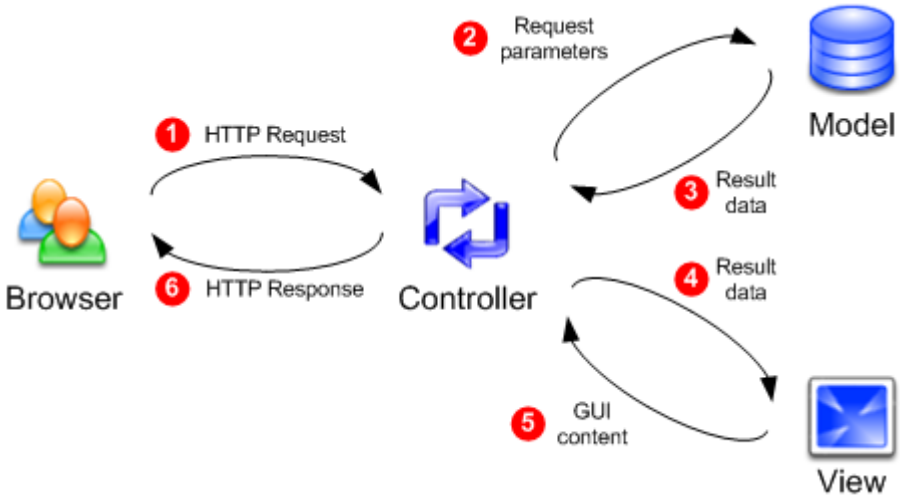


Εικόνα 2: Τα συστατικά του προτύπου MVC [13]

Το πρόβλημα όταν χρησιμοποιείται το *πρότυπο σχεδίασης MVC* για την ανάπτυξη μιας εφαρμογής διαδικτύου προκύπτει από το γεγονός ότι οι εφαρμογές διαδικτύου είναι εγγενώς χωρισμένες μεταξύ του *πελάτη* και του *εξυπηρετητή*. Η *Όψη*, βεβαίως, εμφανίζεται μόνο στον *πελάτη* αλλά το *Μοντέλο* και ο *Ελεγκτής* μπορούν (θεωρητικά) να χωριστούν με οποιοδήποτε

τρόπο μεταξύ του *πελάτη* και του *εξυπηρετητή*. Ο διαχωρισμός αυτός εξαρτάται και από τεχνολογικές αποφάσεις που δεν έχουν να κάνουν με την εφαρμογή. Για παράδειγμα η επιλογή «thin-client» έναντι στην «fat-client». Ο όρος «thin-client» αναφέρεται στον *πελάτη* που βασίζει όλη ή την περισσότερη λειτουργικότητα του στον *εξυπηρετητή*, σε αντίθεση με τον «fat-client» ο οποίος εκτελεί το μεγαλύτερο μέρος της λειτουργικότητας του μόνος του χωρίς τη σύνδεση με τον *εξυπηρετητή*. Όταν αναπτύσσεται ως thin-client μόνο η *Όψη* μιας εφαρμογής διαδικτύου μεταφέρεται στον *πελάτη*· οι *Ελεγκτές* και το *Μοντέλο* παραμένουν στον *εξυπηρετητή*. Όταν αναπτύσσεται ως «fat-client» το *Μοντέλο* και οι *Ελεγκτές* της εφαρμογής διαδικτύου μεταφέρονται κι αυτά στον *πελάτη*. [12]

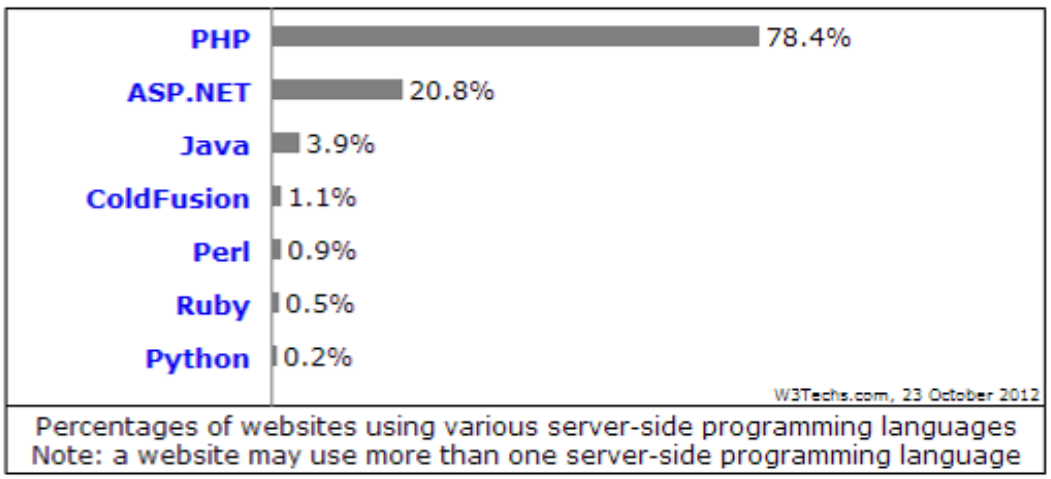
Στην **Εικόνα 3** [14] παρουσιάζεται το παράδειγμα λειτουργίας μιας εφαρμογής διαδικτύου που εφαρμόζει το *πρότυπο σχεδίασης MVC*. Αρχικά ο *Ελεγκτής* λαμβάνει το HTTP αίτημα από τον *πελάτη* (πρόγραμμα περιήγησης), επεξεργάζεται το αίτημα και στέλνει ο ίδιος ένα αίτημα στο *Μοντέλο*. Το *Μοντέλο* αναζητεί και επιστρέφει στον *Ελεγκτή* τα δεδομένα που ζητήθηκαν, τα οποία ο *Ελεγκτής* αποστέλλει στην *Όψη*. Η *Όψη* κατασκευάζει τη Γραφική Διασύνδεση Χρήστη και την αποστέλλει στον *Ελεγκτή* ο οποίος με τη σειρά του την επιστρέφει στον *πελάτη* (πρόγραμμα περιήγησης).



Εικόνα 3: Παράδειγμα λειτουργίας εφαρμογής που εφαρμόζει το πρότυπο MVC

3.2.2 PHP

Η *PHP* είναι μια γενικού σκοπού ευρέως χρησιμοποιημένη γλώσσα που είναι κατάλληλη για προγραμματισμό Διαδικτύου και μπορεί να ενσωματωθεί στην *HTML*. Η σύνταξή της βασίζεται στις γλώσσες C, Java και Perl και είναι εύκολη στην εκμάθηση. Ο κύριος στόχος της είναι να επιτρέψει στους προγραμματιστές διαδικτύου να γράφουν δυναμικά παραγόμενες ιστοσελίδες γρήγορα αλλά μπορεί κάποιος να κάνει πολλά περισσότερα με την *PHP*.[15]
Για την τοπική ανάπτυξη εφαρμογών διαδικτύου είναι απαραίτητη η εγκατάσταση ενός *εξυπηρετητή του παγκόσμιου ιστού* (web server) όπως ο *Apache*, η εγκατάσταση του διερμηνευτή της *PHP* και το πιθανότερο η εγκατάσταση μιας βάσης δεδομένων όπως η *MySQL*.Τα παραπάνω μπορούν να εγκατασταθούν ατομικά ή πιο απλά να χρησιμοποιηθεί ένα προκαθορισμένο πακέτο, ανάλογα με το λειτουργικό σύστημα, που τα εγκαθιστά αυτόματα όλα μόνο με μερικά κλικ. [15]
Η *PHP* χρησιμοποιείται στο μεγαλύτερο ποσοστό των ιστότοπων όπως φαίνεται και στην **Εικόνα 4** [16]. Συγκεκριμένα η *PHP* χρησιμοποιείται στο 78,4% των ιστότοπων, η ASP.NET στο 20,8%, η Java στο 3,9%, η ColdFusion στο 1,1%, η Perl στο 0,9%, η Ruby στο 0,5% και η Python στο 0,2%.
Η έκδοση που χρησιμοποιήθηκε στην ανάπτυξη της εφαρμογής είναι η 5.3.8.



Εικόνα 4: Χρήση των server-side γλωσσών προγραμματισμού στους ιστότοπους

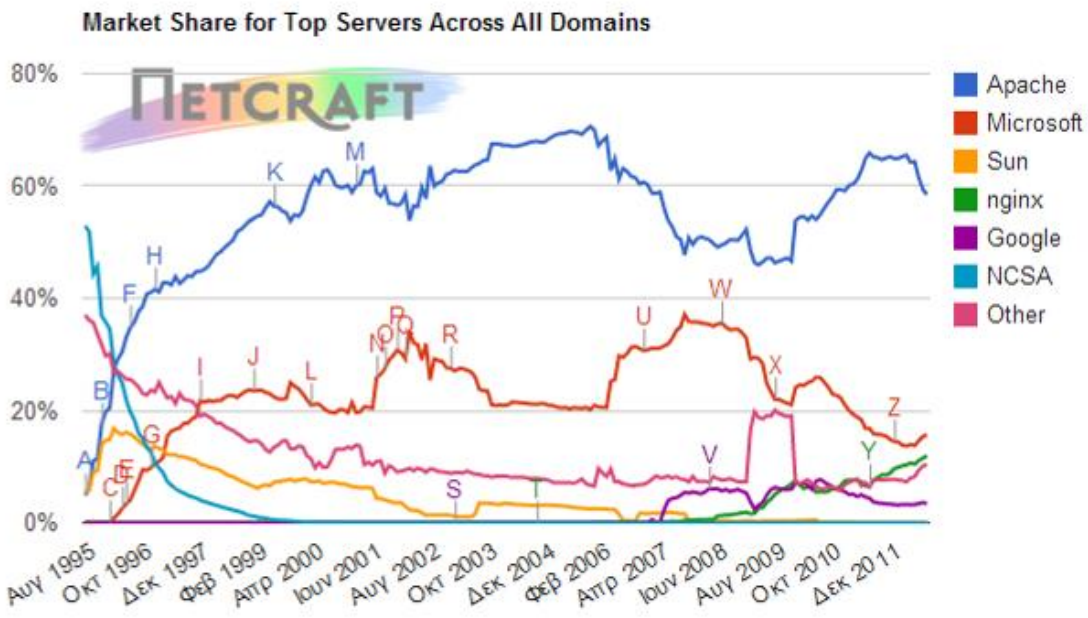
3.2.2.1 CodeIgniter

http://codeigniter.com/

Για την εφαρμογή του *προτύπου σχεδίασης MVC* σε γλώσσα PHP υπάρχουν διάφορα frameworks που μπορούν να βοηθήσουν στην ευκολότερη και καλύτερα οργανωμένη ανάπτυξη μιας εφαρμογής. Χρησιμοποιώντας ένα framework για την συγγραφή κώδικα ο προγραμματιστής γλυτώνει από τον κόπο και χρόνο να γράψει κομμάτια κώδικα που έχουν ήδη γραφεί και χρησιμοποιούνται συχνά. Επίσης με τη χρήση ενός framework μπορεί να γίνει πιο εύκολη η οργάνωση της εφαρμογής, ειδικά για όχι και τόσο έμπειρους προγραμματιστές, καθώς παρέχεται στον προγραμματιστή μια δομή αρχείων που μπορεί να τον βοηθήσει να «χτίσει» την εφαρμογή του.
Το *CodeIgniter* είναι μεταξύ άλλων (Yii, CakePHP, Zend Framework, Symfony κτλ) ένα από τα 10 δημοφιλέστερα frameworks.[17] Παρέχει εκτενή τεκμηρίωση, είναι εύκολο στην εκμάθηση, τη χρήση, εγκατάσταση και διαμόρφωση και έχει μεγάλο αριθμό χρηστών.

3.2.3 Apache HTTP

Ο Apache HTTP, γνωστός και ως *Apache*, αποτελεί έναν *εξυπηρετητή του παγκόσμιου ιστού* (*web server*). Η κύρια λειτουργία ενός *εξυπηρετητή του παγκόσμιου ιστού* είναι να αποστέλλει ιστοσελίδες στους πελάτες ύστερα από αίτημα τους χρησιμοποιώντας το πρωτόκολλο HTTP (Hypertext Transfer Protocol). Ο *Apache* αναπτύσσεται και συντηρείται από μια ομάδα εθελοντών ανά τον κόσμο, διανέμεται δωρεάν και αποτελεί έναν από τους πιο δημοφιλείς εξυπηρετητές παγκόσμιου ιστού όπως φαίνεται στην **Εικόνα 5** και τον **Πίνακας 2** [18]**,**[19] .



Εικόνα 5: Διάγραμμα μεριδίου αγοράς δημοφιλέστερων Εξυπηρετητών Παγκόσμιου Ιστού (Αύγουστος 1995 – Σεπτέμβριος 2012)

Developer	August 2012	Percent	September 2012	Percent	Change
Apache	373,069,751	59.39%	362,714,083	58.49%	-0.90
Microsoft	96,529,586	15.37%	97,368,803	15.70%	0.33
nginx	72,429,976	11.53%	73,976,009	11.93%	0.40
Google	22,561,854	3.59%	21,576,233	3.48%	-0.11

Πίνακας 2: Ποσοστά μεριδίου αγοράς δημοφιλέστερων εξυπηρετητών παγκοσμίου ιστού (Αύγουστος 2012 – Σεπτέμβριος 2012)

3.2.4 MySQL

Η *MySQL* αποτελεί το πιο διαδεδομένο ανοικτού κώδικα *σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων*. Αυτό οφείλεται στην υψηλή της απόδοση, αξιοπιστία και ευκολία στη χρήση. Αποτελεί κεντρικό κομμάτι των πακέτων λογισμικού τύπου *AMP*, τα οποία αναφέρονται στην ενότητα 3.2.5, και χρησιμοποιείται από μεγάλους οργανισμούς και διαδικτυακές εφαρμογές όπως οι Facebook, Google, Adobe, Wikipedia, Twitter, Nokia.com και YouTube. [20]

3.2.5 XAMPP

http://www.apachefriends.org/en/xampp.html

Όπως αναφέρθηκε σε προηγούμενη ενότητα υπάρχουν πακέτα που επιτρέπουν την εγκατάσταση ενός *εξυπηρετητή του παγκόσμιου ιστού* (web server), της *MySQL* και του διερμηνευτή της *PHP* με πολύ εύκολο τρόπο. Τα πακέτα αυτά ονομάζονται *AMP* (Apache – MySQL – PHP). Υπάρχουν τα *LAMPs* (για Linux), τα *WAMPs* (για Windows), τα *MAMPs* (για Macintosh), τα *SAMPs* (για Solaris) και τα *FAMPs* (για FreeBSD).[21]

Το *XAMPP* είναι ένα ελεύθερο και ανοικτού κώδικα πολλαπλής πλατφόρμας πακέτο που μπορεί να χρησιμοποιηθεί για την εγκατάσταση ενός εξυπηρετητή *Apache*, της *MySQL*, και διερμηνευτών της *PHP* και της *Perl*. Είναι διαθέσιμο για τα λειτουργικά συστήματα Linux, Windows, Mac Os X και Solaris και διατίθεται από την ιστοσελίδα του.

3.2.6 HTML

Η *Γλώσσα Χαρακτηρισμού Υπερκειμένου* (*HyperText Markup Language – HTML*) βασίζεται στην SGML (Standard Generalized Markup Language), ένα πολύ μεγαλύτερο σύστημα επεξεργασίας εγγράφων. Είναι μια γλώσσα για την περιγραφή της δομής ενός εγγράφου. Τα *προγράμματα περιήγησης* επιπρόσθετα με τη δυνατότητά τους να ανακτούν ιστοσελίδες από το Διαδίκτυο , λειτουργούν επίσης και σαν μηχανισμοί μορφοποίησης για την HTML. Το *πρόγραμμα περιήγησης* χρησιμοποιεί συγκεκριμένες αντιστοιχίσεις μεταξύ των ονομάτων των στοιχείων της σελίδας και των πραγματικών στυλ που εμφανίζει στην οθόνη. Φροντίζει επίσης για την αναδίπλωση του κειμένου σε επόμενες γραμμές, έτσι ώστε να χωρά πάντα στο τρέχον πλάτος του παραθύρου. Διαφορετικά *προγράμματα περιήγησης* τα οποία τρέχουν σε διαφορετικά υπολογιστικά συστήματα, μπορεί να αντιστοιχίζουν διαφορετικά στυλ σε κάθε στοιχείο μιας σελίδας. Ορισμένα μπορεί να χρησιμοποιούν διαφορετικές γραμματοσειρές από άλλα. Για τον σχεδιαστή μιας σελίδας αυτό σημαίνει πώς οι *HTML* σελίδες που δημιουργεί μπορεί να δείχνουν εντελώς διαφορετικές από σύστημα σε σύστημα και από εφαρμογή σε εφαρμογή. Οι πραγματικές πληροφορίες και οι συνδέσεις που

περιέχουν οι σελίδες θα είναι πάντα εκεί αλλά η εμφάνιση των σελίδων στην οθόνη μπορεί να αλλάζει. [22]

3.2.7 CSS

Τα *φύλλα στυλ με δυνατότητα επικάλυψης* (cascading style sheets – CSS) δίνουν τη δυνατότητα να εφαρμοστούν προχωρημένες ρυθμίσεις μορφοποίησης στις *ετικέτες* (tags) της HTML. [22] Η *CSS* είναι η γλώσσα που χρησιμοποιούν οι σχεδιαστές ιστοσελίδων για τον προσδιορισμό πληροφοριών όπως πλάτη, ύψη, χρώματα, περιθώρια, περιγράμματα, φόντο και τύπους στυλ. Αν η *HTML* παρέχει τη δομή μιας σελίδας, η *CSS* παρέχει την παρουσίασή της. Η *CSS* ελέγχει τα πάντα που αφορούν την εμφάνιση μιας σελίδας (χρώματα, τυπογραφία, διάταξη, μεγέθη). [23] Για τη χρήση της *CSS* είναι απαραίτητο να δομηθεί σωστά ο κώδικας *HTML*, αυτό επιτυγχάνεται τοποθετώντας ετικέτες στα στοιχεία για τα οποία θα εφαρμοστεί κάποιο στυλ. Επιπλέον μπορούν να αποδοθούν οι ιδιότητες «*id*» ή/και «*class*» στα στοιχεία που επιθυμούμε να διαμορφώσουμε. Η ιδιότητα «*id*» χρησιμοποιείται για να προσδιορίσει ένα στοιχείο της σελίδας. Είναι αρκετά σύνηθες να χρησιμοποιείται η ιδιότητα αυτή για να περιγράψει βασικά μέρη μιας σελίδας όπως για παράδειγμα περιοχές περιεχομένου ή κύρια πλοήγηση. Ενώ η ιδιότητα «*id*» χρησιμοποιείται για ένα στοιχείο της σελίδας, το ίδιο όνομα της ιδιότητας «*class*» μπορεί να δοθεί σε οποιοδήποτε αριθμό στοιχείων μιας σελίδας. Η ιδιότητα αυτή χρησιμεύει στον προσδιορισμό τύπων περιεχομένου ή ομοίων αντικειμένων. Χρησιμοποιώντας τις παραπάνω ιδιότητες η γλώσσα *CSS* μπορεί να προσπελάσει τα αναγνωρισμένα στοιχεία και να τους εφαρμόσει τα στυλ που έχει ορίσει ο προγραμματιστής. [24]

3.2.8 JavaScript

Η γλώσσα *JavaScript* αποτελεί μια γλώσσα σεναρίων (scripting language) η οποία επιτρέπει στον προγραμματιστή να παράγει *HTML* δυναμικά αλλά και να κατασκευάσει διαδικτυακές εφαρμογές που δίνουν στον χρήστη τη δυνατότητα να αλληλεπιδρά μαζί τους. Οι εντολές *JavaScript* μπορούν να

αποκρίνονται στα γεγονότα που προκαλεί ο χρήστης όπως κλικ ποντικιού, εισαγωγή στοιχείων σε φόρμες ή περιήγηση στη σελίδα. Ο κώδικας που γράφεται μπορεί είτε να ενσωματωθεί στη σελίδα *HTML* ή να κληθεί ως εξωτερικό αρχείο και μεταγλωττίζεται κατά την εκτέλεση από το διερμηνευτή του προγράμματος περιήγησης. Τα γνωστότερα προγράμματα περιήγησης έχουν ενσωματωμένο τον διερμηνευτή της *JavaScript*. Επειδή ο κώδικας θα εκτελεστεί στο πρόγραμμα περιήγησης του πελάτη είναι πολύ σημαντικό να γράφεται κώδικας ανεξάρτητος προγράμματος περιήγησης ή έστω κώδικας που λαμβάνει υπ' όψιν τόσο τα διαφορετικά προγράμματα περιήγησης όσο και τις διαφορετικές εκδόσεις τους. [25]

3.2.8.1 JQuery

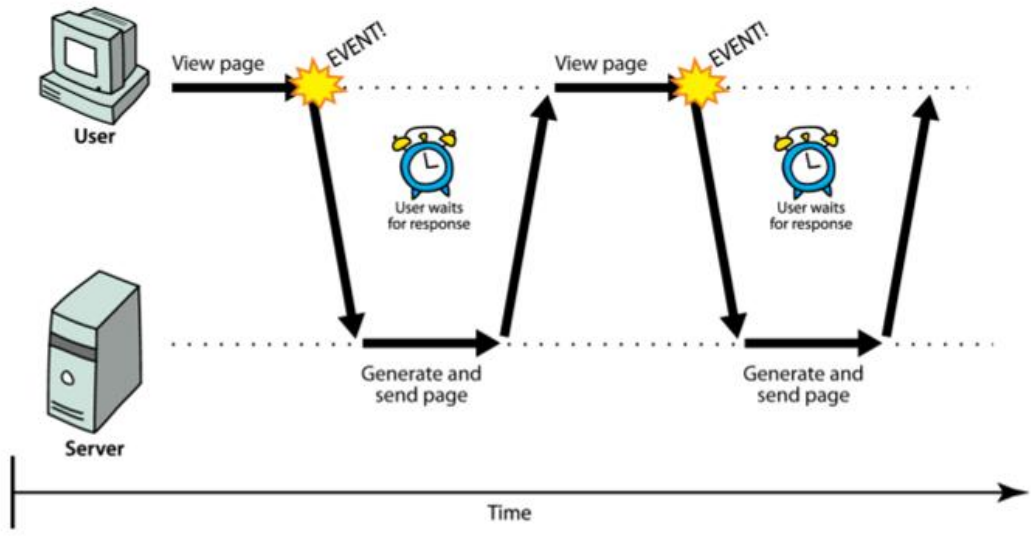
http://jquery.com/

Η *JQuery* είναι μια βιβλιοθήκη της *JavaScript* η οποία σχεδιάστηκε για να απλοποιήσει τη συγγραφή κώδικα από πλευράς πελάτη. Απλοποιεί την επιλογή των στοιχείων της *HTML*, τον χειρισμό συμβάντων, το animation και τις κλήσεις *AJAX* για την ταχεία ανάπτυξη μιας διαδικτυακής εφαρμογής. Παρέχει πλήρη και αναλυτική τεκμηρίωση ενώ έχει μεγάλο αριθμό χρηστών και ενεργή κοινότητα. Στα θετικά της βιβλιοθήκης ανήκει η συμβατότητα της με διαφορετικά προγράμματα περιήγησης (όπως Mozilla Firefox, Internet Explorer, Google Chrome, Opera κτλ), το μικρό μέγεθος της βασικής βιβλιοθήκης ώστε να μην καθυστερεί την εφαρμογή, τα πολλά plugin που διατίθενται, ο μικρός χρόνος που απαιτείται για την εκμάθησή της και η καλή τεκμηρίωση που παρέχει. [26], [27]

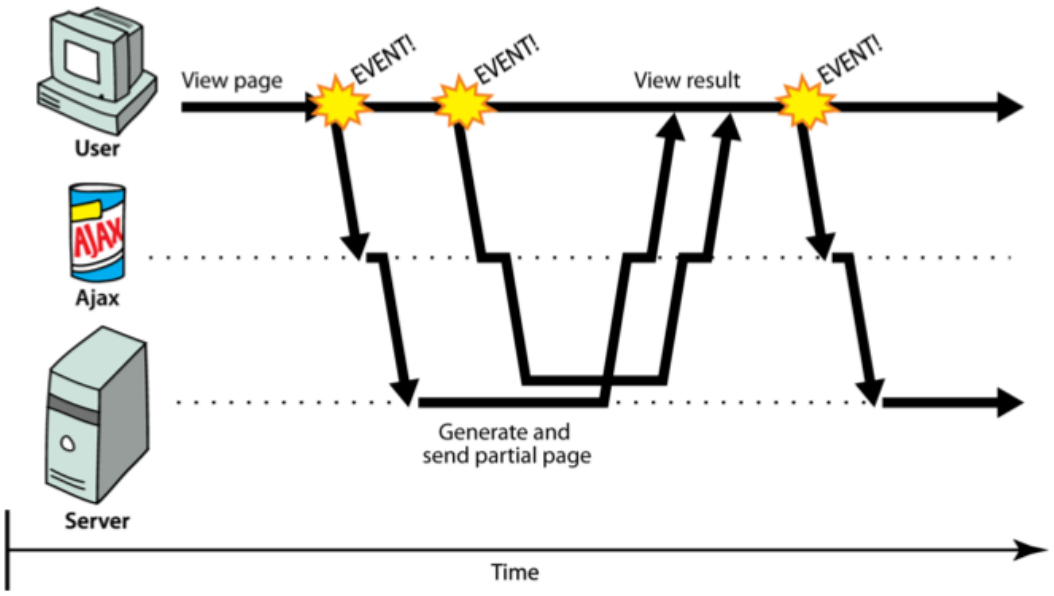
3.2.8.2 AJAX

Η *Asynchronous JavaScript and XML* (AJAX) αποτελεί ένα σύνολο τεχνολογιών που συνδυάζονται για να παρέχουν περισσότερες δυνατότητες στις εφαρμογές διαδικτύου. Συγκεκριμένα μπορούν να δώσουν σε μια εφαρμογή διαδικτύου χαρακτηριστικά που έχει ήδη μια εφαρμογή desktop όπως για παράδειγμα άμεση απόκριση στις κινήσεις του χρήστη. [28]

Μια κλασσική εφαρμογή διαδικτύου λειτουργεί ως εξής: Οι περισσότερες ενέργειες του χρήστη προς τη διασύνδεση της εφαρμογής προκαλούν την αποστολή ενός *HTTP αιτήματος* πίσω στον εξυπηρετητή. Ο εξυπηρετητής εκτελεί κάποιες εργασίες – ανάκτηση δεδομένων, υπολογισμούς, επικοινωνία με άλλα συστήματα – και επιστρέφει μια σελίδα *HTML* στον πελάτη. Όσο ο εξυπηρετητής εκτελεί τις ενέργειές του ο χρήστης απλώς περιμένει. [28]
Μια εφαρμογή που χρησιμοποιεί *AJAX* επιτρέπει την *ασύγχρονη ανταλλαγή δεδομένων* με τον εξυπηρετητή. Αυτό γίνεται εισάγοντας μια μηχανή *AJAX* μεταξύ του πελάτη και του εξυπηρετητή. Αντί για τη φόρτωση της ιστοσελίδας το πρόγραμμα περιήγησης φορτώνει μια μηχανή *AJAX* – γραμμένη σε *JavaScript* και συνήθως κρυμμένη σε ένα κρυφό πλαίσιο. Η μηχανή αυτή είναι υπεύθυνη τόσο για την εμφάνιση της διασύνδεσης που βλέπει ο χρήστης όσο και για να επικοινωνεί με τον εξυπηρετητή εκ μέρους του χρήστη. Με αυτό τον τρόπο επιτρέπει στην αλληλεπίδραση του χρήστη με την εφαρμογή να γίνεται ασύγχρονα – ανεξάρτητα από την επικοινωνία με τον εξυπηρετητή. Έτσι ο χρήστης δεν θα χρειαστεί να περιμένει να ολοκληρώσει ο εξυπηρετητής τις εργασίες του προτού συνεχίσει. [28] Τα παραπάνω απεικονίζονται και γραφικά στις **Εικόνα 6** και **Εικόνα 7**[29]



Εικόνα 6: Σύγχρονη διαδικτυακή επικοινωνία



Εικόνα 7: Ασύγχρονη διαδικτυακή επικοινωνία

3.3 Σημασιολογικός Ιστός

Ο *Σημασιολογικός Ιστός* (Semantic Web) αποτελεί έναν Ιστό από πληροφορίες οι οποίες είναι διαμορφωμένες με τρόπο που μπορούν να διαβαστούν από μηχανές και η έννοια των οποίων είναι σαφώς καθορισμένη από πρότυπα. [30]

3.3.1 Οντολογίες

Μια *οντολογία* ορίζει τους όρους που χρησιμοποιούνται για την περιγραφή και την αναπαράσταση ενός *πεδίου γνώσης*. Αποτελείται από χρησιμοποιήσιμους από τη μηχανή ορισμούς των βασικών εννοιών στο πεδίο και των μεταξύ τους σχέσεων. Συγκεκριμένα, μια *οντολογία* παρέχει περιγραφές για:

- Κλάσεις (γενικές έννοιες) στο πεδίο αναφοράς.
- Ιδιότητες (ή χαρακτηριστικά) που περιγράφουν τις κλάσεις και συχνά τις συσχετίζουν μεταξύ τους.
- Στιγμιότυπα των κλάσεων αυτών.

Μια *οντολογία* παρέχει μετα-πληροφορίες που περιγράφουν τη σήμανση των δεδομένων. Οι *οντολογίες* επιτρέπουν την κοινοποίηση της γνώσης και την επαναχρησιμοποίηση της, όπου οι πηγές των πληροφοριών μπορούν να επικοινωνήσουν μεταξύ των ατόμων ή των πρακτόρων λογισμικού. Οι

σημασιολογικοί δεσμοί στις *οντολογίες* είναι αναγνωρίσιμοι από τη μηχανή και με αυτόν τον τρόπο επιτρέπουν τη δημιουργία δηλώσεων και την εκτέλεση ερωτήσεων για ένα θεματικό τομέα χάρη στη χρήση μιας αντίληψης που περιγράφει τις οντότητες και τους δεσμούς τους. Αυτή η αντίληψη παρέχει στους πράκτορες λογισμικού ένα λεξιλόγιο για να επικοινωνήσουν με τη γνώση. [30]

3.3.2 Resource Description Framework (RDF)

Το *RDF* είναι μια γενική μέθοδος για την αποσύνθεση της γνώσης σε μικρά κομμάτια , με μερικούς κανόνες σχετικούς με τη *σημασιολογία* (semantics), ή έννοια αυτών των κομματιών. Το θέμα είναι να υπάρχει μια μέθοδος τόσο απλή που να εκφράζει οποιοδήποτε γεγονός και όμως τόσο δομημένη ώστε οι εφαρμογές ενός υπολογιστή να μπορούν να χρησιμοποιήσουν τη γνώση που εκφράζεται σε *RDF*. Αναφέρεται ο όρος «μέθοδος» και όχι μορφή γιατί μπορεί κάποιος να καταγράψει αυτά τα κομμάτια με οποιοδήποτε τρόπο και παρόλα αυτά να διατηρήσει την αρχική πληροφορία και δομή, όπως μπορεί κάποιος να εκφράσει το ίδιο νόημα σε διαφορετικές ανθρώπινες γλώσσες ή να υλοποιήσει την ίδια δομή δεδομένων με διαφορετικούς τρόπους.

Κατά κάποιο τρόπο το *RDF* θα μπορούσε να συγκριθεί με την *XML*. Η *XML* έχει επίσης σχεδιαστεί να είναι απλή και εφαρμόσιμη σε κάθε τύπο δεδομένων και είναι κάτι παραπάνω από μια μορφή αρχείων. Αυτό που ξεχωρίζει το *RDF* από την *XML* είναι ότι το *RDF* είναι σχεδιασμένο να αναπαριστά γνώση σε έναν κατανεμημένο κόσμο. Το ότι το *RDF* είναι σχεδιασμένο για γνώση, και όχι για δεδομένα, σημαίνει ότι ασχολείται ιδιαίτερα με το νόημα. Οτιδήποτε θα αναφερθεί σε *RDF* σημαίνει κάτι. Μπορεί να είναι μια αναφορά σε κάτι στον κόσμο, όπως ένα άτομο ή μια ταινία, ή μπορεί να είναι μια αφηρημένη έννοια, όπως η κατάσταση του να είσαι φίλος με κάποιον άλλον. Και τοποθετώντας τρείς τέτοιες οντότητες μαζί, το πρότυπο *RDF* υποδεικνύει πώς να φτάσει κανείς σε ένα γεγονός. Το νόημα του *τριπλότυπου* «John, Bob, η κατάσταση του να είναι φίλοι» μπορεί να είναι ότι ο John και ο Bob είναι φίλοι. Τοποθετώντας πολλά γεγονότα μαζί, κάποιος μπορεί να φτάσει σε κάποια μορφής γνώση. Πρότυπα όπως τα

RDFS και *OWL* προσθέτουν σημασιολογία στο *RDF* για να αντλούνται λογικά συμπεράσματα από δεδομένα.

Συγκριτικά, η *XML* από μόνη της δεν ασχολείται με το νόημα. Οι κόμβοι της *XML* δεν χρειάζεται να είναι συσχετισμένοι με συγκεκριμένες έννοιες και το πρότυπο της *XML* δεν υποδεικνύει πώς να προκύψει ένα γεγονός από ένα έγγραφο. Για παράδειγμα, αν δοθούν σε κάποιον μερικά *XML* έγγραφα των οποίων οι βασικοί κόμβοι είναι σε μια ξένη γλώσσα που δεν μπορεί να καταλάβει, τότε δεν θα μπορεί να κάνει κάτι χρήσιμο με αυτά τα δεδομένα παρά μόνο να τα εμφανίσει. Έγγραφα *RDF* με κόμβους που δεν μπορούμε να καταλάβουμε θα μπορούσαν και πάλι να επεξεργαστούν με κάποιο χρήσιμο τρόπο γιατί το *RDF* προσδιορίζει κάποιο βασικό επίπεδο νοήματος. Αυτό δεν σημαίνει ότι δεν μπορεί κάποιος να αναπτύξει ένα δικό του πρότυπο πάνω σε *XML* το οποίο μπορεί να πει πώς θα προκύψει ένα σύνολο γεγονότων σε ένα *XML* έγγραφο αλλά αυτό θα σημαίνει ότι προσπαθεί να ανακαλύψει ξανά κάτι σαν το *RDF*.

Η δεύτερη βασική πτυχή του *RDF* είναι ότι δουλεύει καλά για κατανεμημένη πληροφορία. Αυτό σημαίνει ότι, οι εφαρμογές *RDF* μπορούν να συναρμολογούν αρχεία *RDF* αναρτημένα από διαφορετικούς ανθρώπους στο Διαδίκτυο και να μαθαίνουν εύκολα από αυτά νέα πράγματα που κανένα έγγραφο μόνο του δεν μπορεί να υποστηρίξει. Αυτό γίνεται με δυο τρόπους, πρώτον με την ένωση εγγράφων χρησιμοποιώντας το κοινό λεξιλόγιο που χρησιμοποιούν και δεύτερον επιτρέποντας οποιοδήποτε έγγραφο να χρησιμοποιεί οποιοδήποτε λεξιλόγιο. Αυτό επιτρέπει τεράστια ευελιξία στην έκφραση γεγονότων σχετικών με ένα ευρύ φάσμα πραγμάτων, αντλώντας πληροφορίες από ένα ευρύ φάσμα πηγών. [31]

Το RDF σχεδιάστηκε για την αναπαράσταση μεταδεδομένων που περιγράφουν πόρους του Ιστού χρησιμοποιώντας τρεις τύπους αντικειμένων:

- *Πόρους* (resources), δηλαδή οντότητες που περιγράφονται στις εκφράσεις του *RDF* και πάντοτε αναγνωρίζονται με κάποιο *URI* ή και ένα επιπλέον id,

- *Ιδιότητες* (properties) ή κατηγορήματα, που ορίζουν συγκεκριμένες παραμέτρους, χαρακτηριστικά, ιδιότητες (attributes) ή σχέσεις τα οποία χρησιμοποιούνται για να περιγράψουν έναν πόρο,
- *Δηλώσεις* (statements) που αντιστοιχούν μια τιμή για μια ιδιότητα σε έναν πόρο (η τιμή αυτή μπορεί να είναι με τη σειρά της μια άλλη *RDF* δήλωση). Έχουν παρόμοια δομή και σημασιολογία με μία πρόταση της μορφής : «υποκείμενο, ρήμα, αντικείμενο», όπου το ρήμα είναι μια ιδιότητα.

Η υποκείμενη δομή οποιασδήποτε έκφρασης του *RDF* είναι μια συλλογή *τριπλότυπων*, κάθε ένα εκ των οποίων περιέχει ένα *υποκείμενο*, ένα *κατηγόρημα* και ένα *αντικείμενο*, Ένα σύνολο από *τριπλότυπα* καλείται *γράφημα RDF*. [30]

3.3.3 RDF Schema

Το *RDF* δεν παρέχει μηχανισμούς για την περιγραφή των ιδιοτήτων ούτε και για την περιγραφή της σχέσης τους με άλλες πηγές. Αυτός είναι ρόλος για το *λεξιλόγιο της γλώσσας περιγραφής*, το *RDF Schema*. Το *RDF Schema* καθορίζει τις κλάσεις και τις ιδιότητες που μπορούν να χρησιμοποιηθούν για να περιγράψουν κλάσεις, ιδιότητες και άλλες πηγές. Αποτελεί μια *σημασιολογική επέκταση* του *RDF*. Παρέχει μηχανισμούς για την περιγραφή συνόλων από σχετικές πηγές και τους δεσμούς μεταξύ τους. Οι λεξιλογικές περιγραφές του *RDF Schema* είναι γραμμένες σε *RDF*.

Οι τρεις σημαντικότερες έννοιες που μας δίνουν το *RDF* και το *RDF Schema* είναι η «*πηγή*» (rdfs:Resource), η «*κατηγορία*» (rdfs:Class) και η «*ιδιότητα*» (rdfs:Property). Η αναπαράσταση των εννοιών γίνεται μέσω των *κλάσεων* (classes). Οι *κλάσεις* μπορούν να ιεραρχηθούν, ενώ η πολλαπλή κληρονομικότητα υποστηρίζεται. [30]

3.3.4 Ontology Web Language (OWL)

Η *OWL* είναι η πιο πρόσφατη εξέλιξη στις *γλώσσες οντολογιών* για το *Σημασιολογικό Ιστό*. Ακολουθεί το συντακτικό του *RDF / RDF Schema* εμπλουτίζοντας όμως το λεξιλόγιό της για την περιγραφή των *κλάσεων* και

των ιδιοτήτων, υποστηρίζοντας, μεταξύ άλλων, σχέσεις μεταξύ των *κλάσεων* (π.χ. ότι δύο κλάσεις είναι ξένες), πληθυσμότητα (περιορισμοί αριθμού), ισότητα, περισσότερα χαρακτηριστικά των ιδιοτήτων (π.χ. συμμετρία) και κλάσεις απαρίθμησης (χρήση ονοματικών).Η μεγαλύτερη εκφραστικότητα που προσδίδεται στη γλώσσα με τον τρόπο αυτό, είναι αναγκαία για το *Σημασιολογικό Ιστό* καθώς επιτρέπει τη χρήση συστημάτων συλλογισμού για τη διεξαγωγή αποτελεσματικών συμπερασμών πάνω στις *οντολογίες* που αναπαρίστανται στον Ιστό με τη γλώσσα αυτή.

Σχετικά με την αναπαράσταση, οι κλάσεις καθορίζονται χρησιμοποιώντας ένα στοιχείο *owl:Class*, ενώ υπάρχουν δύο προκαθορισμένες κλάσεις, το *owl:Thing* και το *owl:Nothing*. Το πρώτο είναι η γενικότερη κλάση που περιέχει τα πάντα (όλα είναι ένα πράγμα) και το τελευταίο είναι μια κενή κλάση. Κατά συνέπεια κάθε κλάση είναι μια υποκλάση του *owl:Thing* και υπερκλάση του *owl:Nothing*. [30]

Η *OWL* έχει σχεδιαστεί για τις εφαρμογές που πρέπει να επεξεργαστούν το περιεχόμενο των πληροφοριών αντί απλά να παρουσιάσουν τις πληροφορίες στους ανθρώπους. Η τυποποίηση των *οντολογιών* σε γλώσσα *OWL* κάνει τα δεδομένα στο Web περισσότερο επεξεργάσιμα από μηχανές και επαναχρησιμοποιήσιμα στις εφαρμογές. [32]

3.3.5 Jena

Το *Apache Jena* αποτελεί ένα Java πλαίσιο (framework) για τη δημιουργία *εφαρμογών σημασιολογικού ιστού* (semantic web applications). Παρέχει μια συλλογή από *εργαλεία* και *βιβλιοθήκες* Java που διευκολύνουν την ανάπτυξη εφαρμογών και εργαλείων του σημασιολογικού ιστού. Επιτρέπει στον προγραμματιστή να χρησιμοποιεί τις μεθόδους του *Jena* για να χειρίζεται *σημασιολογικά δεδομένα*. Συνήθως τα *μοντέλα* του *Jena* αποθηκεύονται σε μια βάση η οποία ονομάζεται «*persistent store*». Το *Jena* παρέχει διεπαφές για την αναπαράσταση *μοντέλων*, πόρων, ιδιοτήτων, δηλώσεων και όλων των υπόλοιπων βασικών εννοιών του *RDF*.

Οι βασικότερες λειτουργίες που μπορούν να εκτελεστούν με το *Jena* είναι:

- Πρόσβαση σε ένα *μοντέλο*: ανάκτηση πληροφοριών που είναι αποθηκευμένες σε ένα *μοντέλο*.

- Αναζήτηση σε ένα *μοντέλο*: αναζήτηση πληροφοριών μέσα σε ένα *μοντέλο* μέσω αποστολής ερωτημάτων
- Χειρισμός *μοντέλων*: παρέχονται τρεις λειτουργίες χειρισμού *μοντέλων*, η ένωση, η τομή και η διαφορά *μοντέλων*. Η ένωση δύο *μοντέλων* είναι η ένωση των συνόλων των *δηλώσεων* (statements) που αναπαριστούν κάθε *μοντέλο*. Με αυτή τη λειτουργία καθίσταται δυνατή η συνένωση δεδομένων από διαφορετικές πηγές. Η τομή και η διαφορά δύο *μοντέλων* μπορεί να υπολογιστεί με παρόμοιο τρόπο. [33]

3.3.6 SPARQL

Η *SPARQL* είναι μια *καθορισμένη γλώσσα για την εκτέλεση ερωτημάτων* σε δεδομένα *RDF* μορφής. Η ανάγκη *εκτέλεσης ερωτημάτων* σε *RDF* δεδομένα δημιουργήθηκε σχεδόν αμέσως μετά την πρότασή της από το *World Wide Web Consortium* (W3C) το 1998. Η *SPARQL* δημοσιεύτηκε το 2004 και από τότε έχει υιοθετηθεί ως πρότυπο για την *εκτέλεση ερωτημάτων* σε δεδομένα του *σημασιολογικού ιστού*. Το 2008 προτάθηκε από το *W3C*. [34]

3.3.7 TopBraid Composer

http://www.topquadrant.com/products/TB_Composer.html

Το *TopBraid Composer* αποτελεί ένα περιβάλλον για την ανάπτυξη *οντολογιών σημασιολογικού ιστού* και *σημασιολογικών εφαρμογών*. Χρησιμοποιεί τα *πρότυπα RDFS*, τη γλώσσα *OWL* και τη γλώσσα ερωτημάτων *SPARQL*. Η δωρεάν έκδοση που χρησιμοποιήθηκε επιτρέπει τη δημιουργία αρχείων *RDF/OWL* και την εκτέλεση *SPARQL ερωτημάτων* σε αυτά.

Στα πλαίσια της διπλωματικής χρησιμοποιήθηκε για τη δημιουργία της οντολογίας των κατηγοριών των προϊόντων. Αρχικά δημιουργήθηκαν σε ένα αρχείο *RDF* και στη συνέχεια μέσω της Java και του *Jena* αντιγράφηκαν στη βάση δεδομένων (*persistent store*).

ΚΕΦΑΛΑΙΟ 4

4 Ανάλυση

Στο κεφάλαιο αυτό περιγράφεται το σύστημα που θα αναπτυχθεί. Αρχικά περιγράφονται οι απαιτήσεις του μέσω κειμένου και στη συνέχεια μέσω περιπτώσεων χρήσης. Τέλος περιγράφεται η αρχιτεκτονική υλοποίησης του συστήματος

4.1 Ανάλυση απαιτήσεων

Το υπό ανάπτυξη σύστημα έχει ως σκοπό να χρησιμοποιηθεί για την εξοικονόμηση χρημάτων και χρόνου στην αγορά προϊόντων για καθημερινή χρήση. Βασίζεται στην ανταλλαγή πληροφοριών μεταξύ των χρηστών σχετικά με τις τιμές των προϊόντων στα σούπερ μάρκετ. Συγκεκριμένα οι εγγεγραμμένοι χρήστες αφού συνδεθούν, θα μπορούν να εισάγουν στο σύστημα τη λίστα των προϊόντων που επιθυμούν να αγοράσουν, να επιλέξουν τα καταστήματα που θα μπορούσαν να επισκεφτούν, ανάλογα με την απόστασή τους από το σπίτι τους, και να επιλέξουν, ανάλογα με την ιδιότητα που έχει ο καθένας, τον μέγιστο αριθμό καταστημάτων από τα οποία επιθυμούν να ψωνίσουν. Στη συνέχεια το σύστημα θα επιστρέφει την οικονομικότερη πρόταση η οποία θα αφορά το συνδυασμό καταστημάτων – προϊόντων που ικανοποιεί τα κριτήρια του χρήστη και επιστρέφει τη μικρότερη συνολική τιμή για την αγορά της λίστας. Η ιδιότητα του χρήστη μπορεί να είναι μια από τις εξής τέσσερεις ιδιότητες: «*bronze*», «*silver*», «*gold*» ή «*platinum*». Αρχικά όλοι οι χρήστες είναι «*bronze*» και μπορούν να επιλέξουν ως μέγιστο αριθμό καταστημάτων το «*1*». Στη συνέχεια ανάλογα με την ανταποδοτικότητα τους εξελίσσονται βαθμιαία σε «*silver*» με μέγιστο αριθμό καταστημάτων «*2*», «*gold*» με μέγιστο αριθμό καταστημάτων «*3*» και «*platinum*» με μέγιστο αριθμό καταστημάτων «*4*». Οι τιμές των προϊόντων θα εισάγονται από τους χρήστες, οι οποίοι θα αξιολογούνται για την ακρίβεια και την ποσότητα των πληροφοριών που παρέχουν. Εκτός από τιμές ένας

χρήστης θα μπορεί να εισάγει στο σύστημα ένα προϊόν ή ένα κατάστημα που πιθανόν δεν υπάρχει στη βάση δεδομένων. Αν ένας χρήστης κριθεί αναξιόπιστος ως προς τις πληροφορίες που παρέχει στο σύστημα τότε δεν λαμβάνονται υπ' όψιν οι επόμενες πληροφορίες που θα παρέχει μέχρις ότου να κριθεί και πάλι αξιόπιστος. Η αξιοπιστία των πληροφοριών επιβεβαιώνεται από τους υπόλοιπους χρήστες, κάθε φορά που χρησιμοποιούν το σύστημα μετά από μια πρόταση αγοράς. Συγκεκριμένα για κάθε πρόταση που του δόθηκε ο χρήστης καλείται να ενημερώσει το σύστημα αν η τιμή που του προτάθηκε ήταν ίδια με την τιμή αγοράς, αν άλλαξε ή αν τελικά ο ίδιος δεν αγόρασε το προϊόν. Σε περίπτωση που ο χρήστης υποδείξει ότι η τιμή ήταν ίδια τότε αυξάνει την αξιοπιστία του χρήστη που «*παρατήρησε*» και εισήγαγε την τιμή στο σύστημα, αλλά και τη δική του αξιοπιστία. Αν ο χρήστης υποδείξει ότι η τιμή άλλαξε τότε καλείται να εισάγει ο ίδιος τη νέα τιμή και επομένως να γίνει αυτός ο «*παρατηρητής*» της τιμής. Ταυτόχρονα αν η προηγούμενη τιμή δεν έχει λάβει πολλές επιβεβαιώσεις εγκυρότητας και αν δεν έχει περάσει μεγάλο χρονικό διάστημα από τη μέρα που εισήχθη στο σύστημα τότε ο προηγούμενος «*παρατηρητής*» της τιμής θα κριθεί αναξιόπιστος για τη συγκεκριμένη πληροφορία. Σε περίπτωση που ένας αναξιόπιστος ως προς το σύστημα χρήστης υποδείξει ότι έχει αλλάξει μια τιμή τότε η υπόδειξή του δεν λαμβάνεται ως έγκυρη και η αποθηκευμένη τιμή δεν μεταβάλλεται.

Χρήστης του συστήματος μπορεί να είναι οποιοσδήποτε έχει πρόσβαση σε έναν υπολογιστή και στο διαδίκτυο. Η χρήση του συστήματος δεν απαιτεί συγκεκριμένες γνώσεις.

4.1.1 Λειτουργίες συστήματος

Σύμφωνα με την παραπάνω περιγραφή οι κυριότερες λειτουργίες του συστήματος είναι οι παρακάτω:

- Εγγραφή Χρήστη (signup)
- Σύνδεση Χρήστη (login)
- Εισαγωγής λίστας προϊόντων από το Χρήστη
- Εύρεση οικονομικότερου συνδυασμού καταστημάτων – προϊόντων

- Εισαγωγή τιμών για προϊόντα – πακέτα προσφορών
- Εισαγωγή καταστήματος
- Εισαγωγή προϊόντος
- Ανατροφοδότηση από το Χρήστη για τις καταχωρημένες τιμές
- Αξιολόγηση Χρήστη

4.1.2 Περιπτώσεις Χρήσης

Στην ενότητα αυτή αποτυπώνονται οι βασικές λειτουργίες της εφαρμογής σύμφωνα με ένα από τα πρότυπα τεκμηρίωσης περιπτώσεων χρήσης της μεθοδολογίας ανάπτυξης λογισμικού *ICONIX* [35]

Εγγραφή Χρήστη

Βασική Ροή

1. Το σύστημα εμφανίζει την αρχική σελίδα.
2. Ο Χρήστης επιλέγει την Εγγραφή Χρήστη
3. Το σύστημα εμφανίζει τη σελίδα Εγγραφής Χρήστη
4. Ο Χρήστης εισάγει τα στοιχεία του και επιλέγει αποθήκευση
5. Το σύστημα αποθηκεύει τα στοιχεία στη βάση
6. Το σύστημα εμφανίζει την αρχική σελίδα για τους συνδεδεμένους χρήστες

Εναλλακτική Ροή

5.α.1 Το όνομα χρήστη που επιλέχθηκε υπάρχει ήδη στη βάση

5.α.2 Το σύστημα εμφανίζει τη φόρμα εισαγωγής στοιχείων με μήνυμα λάθους

5.α.3 Επιστροφή στο βήμα 4 της βασικής ροής

Σύνδεση Χρήστη

Βασική Ροή

1. Ο Χρήστης επιλέγει τη Σύνδεση Χρήστη
2. Το σύστημα εμφανίζει τη σελίδα Σύνδεσης Χρήστη
3. Ο Χρήστης εισάγει το όνομα χρήστη και τον κωδικό και επιλέγει Σύνδεση
4. Το σύστημα αναγνωρίζει τον χρήστη

5. Το σύστημα εμφανίζει την αρχική σελίδα για τους συνδεδεμένους χρήστες.

Εναλλακτική Ροή

4.α.1 Το σύστημα δεν αναγνωρίζει τον χρήστη

4.α.2 Επιστροφή στο βήμα 2 της βασικής ροής

Εισαγωγή λίστας προϊόντων από το Χρήστη

Βασική Ροή

1. Ο Χρήστης επιλέγει την Εισαγωγή Λίστας
2. Το σύστημα εμφανίζει την Οθόνη για την Εισαγωγή Λίστας Προϊόντων
3. Ο Χρήστης εισάγει τα προϊόντα που επιθυμεί να αγοράσει, τα καταστήματα στα οποία θα ενδιαφερόταν να πάει, τον μέγιστο αριθμό καταστημάτων που θα επισκεφτεί και τα υποβάλλει.
4. Το σύστημα αναζητά και βρίσκει την οικονομικότερη λίστα, οικονομικότερο συνδυασμό λίστας – κριτηρίων, αποθηκεύει και εμφανίζει τη λίστα στο χρήστη.

Εναλλακτική Ροή

3.α.1 Το προϊόν που αναζητά ο Χρήστης δεν υπάρχει στο σύστημα.

3.α.2 Ο Χρήστης επιλέγει Εισαγωγή Προϊόντος.

3.α.3 Εκτέλεση βημάτων 2-4 της Λειτουργίας Εισαγωγή Προϊόντος.

3.α.4 Επιστροφή στο βήμα 3 της Βασικής Ροής.

3.β.1 Το κατάστημα που αναζητά ο Χρήστης δεν υπάρχει στο σύστημα.

3.β.2 Ο Χρήστης επιλέγει Εισαγωγή Καταστήματος.

3.β.3 Εκτέλεση βημάτων 2-4 της Λειτουργίας Εισαγωγή Καταστήματος.

3.β.4 Επιστροφή στο βήμα 3 της Βασικής Ροής.

Εισαγωγή τιμών για προϊόντα – πακέτα προσφορών

Βασική Ροή

1. Ο Χρήστης επιλέγει Εισαγωγή τιμών για προϊόντα
2. Το σύστημα εμφανίζει την οθόνη Εισαγωγής Τιμών
3. Ο Χρήστης επιλέγει προϊόν
4. Το προϊόν υπάρχει στη βάση δεδομένων

5. Το σύστημα εμφανίζει τα στοιχεία του προϊόντος
6. Ο Χρήστης επιλέγει κατάστημα από το οποίο πήρε το προϊόν
7. Ο Χρήστης εισάγει τιμή για το προϊόν
8. Το σύστημα αποθηκεύει την τιμή και καταχωρεί τον Χρήστη ως Παρατηρητή της τιμής

Εναλλακτική Ροή

4.α.1 Το προϊόν δεν υπάρχει στη βάση
4.α.2 Ο Χρήστης επιλέγει Εισαγωγή Προϊόντος
4.α.3 Εκτέλεση βημάτων 2-4 της Λειτουργίας «Εισαγωγή Προϊόντος»
4.α.4 Επιστροφή στο βήμα 3 της Βασικής Ροής
6.α.1 Το κατάστημα δεν υπάρχει στο σύστημα
6.α.2 Ο Χρήστης επιλέγει Εισαγωγή Καταστήματος
6.α.3 Εκτέλεση βημάτων 2-4 της Λειτουργίας «Εισαγωγή Καταστήματος»
6.α.4 Επιστροφή στο βήμα 6 της Βασικής Ροής

Εισαγωγή Προϊόντος

Βασική Ροή

1. Ο Χρήστης επιλέγει Εισαγωγή Προϊόντος
2. Το σύστημα εμφανίζει την οθόνη Εισαγωγής Προϊόντος
3. Ο Χρήστης εισάγει τα στοιχεία του προϊόντος και επιλέγει Αποθήκευση
4. Το σύστημα αποθηκεύει το προϊόν

Εισαγωγή Καταστήματος

Βασική Ροή

1. Ο Χρήστης επιλέγει Εισαγωγή Καταστήματος
2. Το σύστημα εμφανίζει την οθόνη Εισαγωγής Καταστήματος
3. Ο Χρήστης εισάγει τα στοιχεία του καταστήματος και επιλέγει Αποθήκευση
4. Το σύστημα αποθηκεύει το κατάστημα

Ανατροφοδότηση από το Χρήστη για τις τιμές που του προτάθηκαν

Βασική Ροή

1. Ο Χρήστης επιλέγει Αξιολόγηση Προηγούμενης Λίστας
2. Το σύστημα εμφανίζει την οθόνη Αξιολόγησης των τιμών που του προτάθηκαν την προηγούμενη φορά
3. Ο Χρήστης επιλέγει ότι η τιμή για μια πρόταση παρέμεινε ίδια
4. Το σύστημα αξιολογεί ως αξιόπιστο τον Παρατηρητή της τιμής και σε πολύ μικρότερο ποσοστό και τον ίδιο τον Χρήστη
5. Το σύστημα αξιολογεί ως ανταποδοτικό τον Χρήστη

Εναλλακτική Ροή

3.α.1 Ο Χρήστης επιλέγει ότι η τιμή για μια πρόταση άλλαξε

3.α.2 Οι προηγούμενες επιβεβαιώσεις της τιμής είναι ελάχιστες και δεν πέρασαν πολλές μέρες από την καταγραφή της υπό-αλλαγή τιμής.

3.α.3 Το σύστημα αξιολογεί ως αναξιόπιστο τον Παρατηρητή της τιμής

3.α.4 Το σύστημα ζητά από τον Χρήστη την καινούρια τιμή και τον ορίζει ως Παρατηρητή της

3.α.5 Επιστροφή στο βήμα 5 της Βασικής Ροής

3.β.1 Ο Χρήστης επιλέγει ότι η τιμή για μια πρόταση άλλαξε

3.β.2 Οι προηγούμενες επιβεβαιώσεις της τιμής είναι πολλές ή πέρασαν πολλές μέρες από την καταγραφή της υπό-αλλαγή τιμής.

3.β.3 Το σύστημα ζητά από τον Χρήστη την καινούρια τιμή και τον ορίζει ως Παρατηρητή της

3.β.4 Επιστροφή στο βήμα 5 της Βασικής Ροής

3.γ.1 Ο Χρήστης επιλέγει ότι δεν αγόρασε τελικά το προϊόν της πρότασης

3.γ.2 Επιστροφή στο βήμα 5 της Βασικής Ροής

3.δ.1 Ο Χρήστης δεν παρέχει πληροφορία για τις προτάσεις

3.δ.2 Το σύστημα αξιολογεί ως μη ανταποδοτικό τον Χρήστη

4.2 Ανάλυση υλοποίησης

Λόγω της χρησιμότητάς της η υπό ανάπτυξη εφαρμογή μπορεί να αποτελέσει ένα εργαλείο απαραίτητο για κάθε χρήστη. Μπορεί να χρησιμοποιηθεί καθημερινά για την εξοικονόμηση χρημάτων στα πιο συχνά

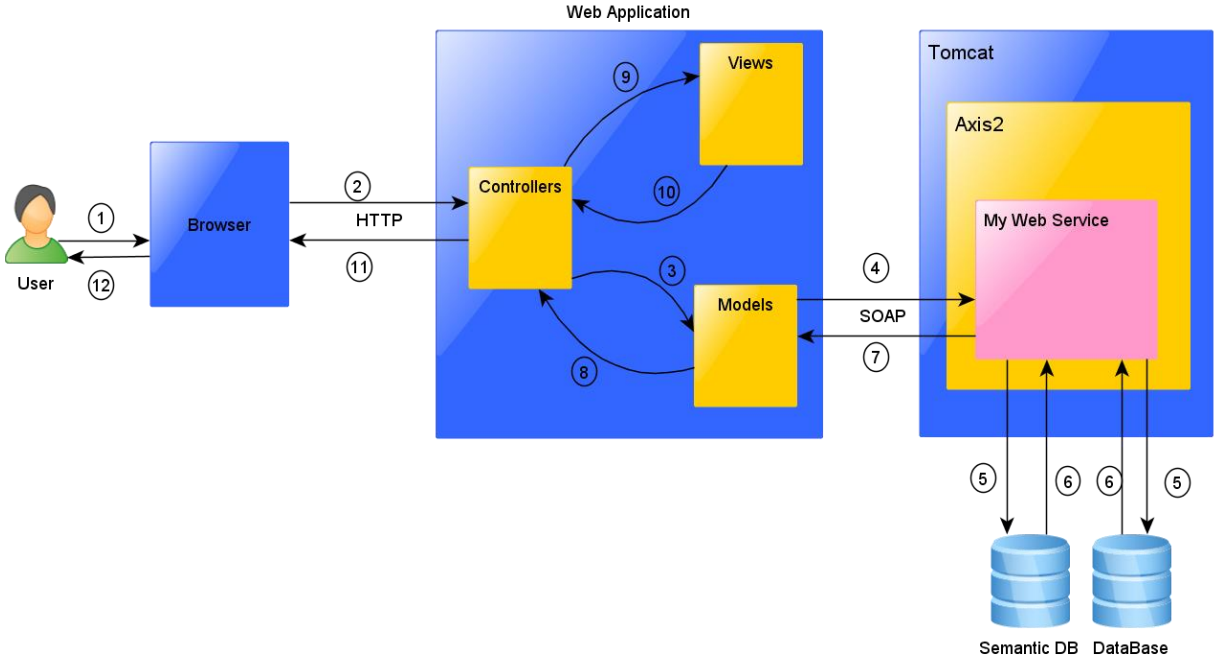
έξοδα ενός νοικοκυριού, τις καθημερινές αγορές προϊόντων από τα σουπερμάρκετ. Για το λόγο αυτό θα μπορούσε να αναπτυχθεί και μια υλοποίησή της για «έξυπνα» κινητά τηλέφωνα.

Με αυτή την προοπτική είναι προτιμότερη η ανάπτυξη όλης της λειτουργικότητας σε μια Υπηρεσία Ιστού, η οποία θα μπορεί να κληθεί από οποιοδήποτε λειτουργικό σύστημα ή πλατφόρμα. Με αυτό τον τρόπο επιτρέπεται στην εφαρμογή να μοιραστεί τη βάση δεδομένων της και με άλλες πιθανές – μελλοντικές εφαρμογές που θα αναπτυχθούν. Επομένως επιλέγεται η ανάπτυξη μιας Υπηρεσίας Ιστού με λειτουργικότητα σύμφωνη με την ανάλυση απαιτήσεων που προηγήθηκε.

Για να κληθεί και να χρησιμοποιηθεί μια Υπηρεσία Ιστού απαιτείται ένα πρόγραμμα – πελάτης. Αυτό το πρόγραμμα – πελάτης θα είναι μια διαδικτυακή εφαρμογή που θα μπορεί να χρησιμοποιηθεί από οποιοδήποτε χρήστη έχει πρόσβαση στο Διαδίκτυο μέσω ενός προγράμματος περιήγησης. Η διαδικτυακή αυτή εφαρμογή θα καλεί τις λειτουργίες της υπηρεσίας και θα προσθέτει, όπου είναι δυνατόν επιπλέον, λειτουργικότητα μέσω ελέγχων εγκυρότητας των εισαγόμενων στοιχείων ή διευκόλυνση του χρήστη στην επιλογή ή αναπαράσταση της πληροφορίας. Η ανάπτυξη της διαδικτυακής εφαρμογής θα ακολουθήσει το *πρότυπο MVC*.

Για την αποθήκευση των δεδομένων είναι απαραίτητη μια βάση δεδομένων ενώ η αποθήκευση όλων των κατηγοριών προϊόντων θα γίνει με τη βοήθεια μια οντολογίας που θα αποθηκευτεί σε χωριστή βάση δεδομένων.

Η αρχιτεκτονική του συστήματος φαίνεται στην **Εικόνα 8** . Συνοπτικά, ο χρήστης μέσω του προγράμματος περιήγησης καλεί τη διαδικτυακή εφαρμογή η οποία μέσω μηνυμάτων SOAP επικοινωνεί με την Υπηρεσία Ιστού. Η Υπηρεσία Ιστού είναι υπεύθυνη για την ανάκτηση δεδομένων από τις δύο βάσεις της εφαρμογής και την αποστολή στη διαδικτυακή εφαρμογή των κατάλληλων SOAP αποκρίσεων. Η διαδικτυακή εφαρμογή, η οποία ακολουθεί το πρότυπο MVC, δημιουργεί σύμφωνα με τα δεδομένα που έλαβε τη σελίδα που θα εμφανιστεί στον χρήστη και του την αποστέλλει μέσω του πρωτοκόλλου HTTP.



Εικόνα 8: Αρχιτεκτονική του συστήματος

ΚΕΦΑΛΑΙΟ 5

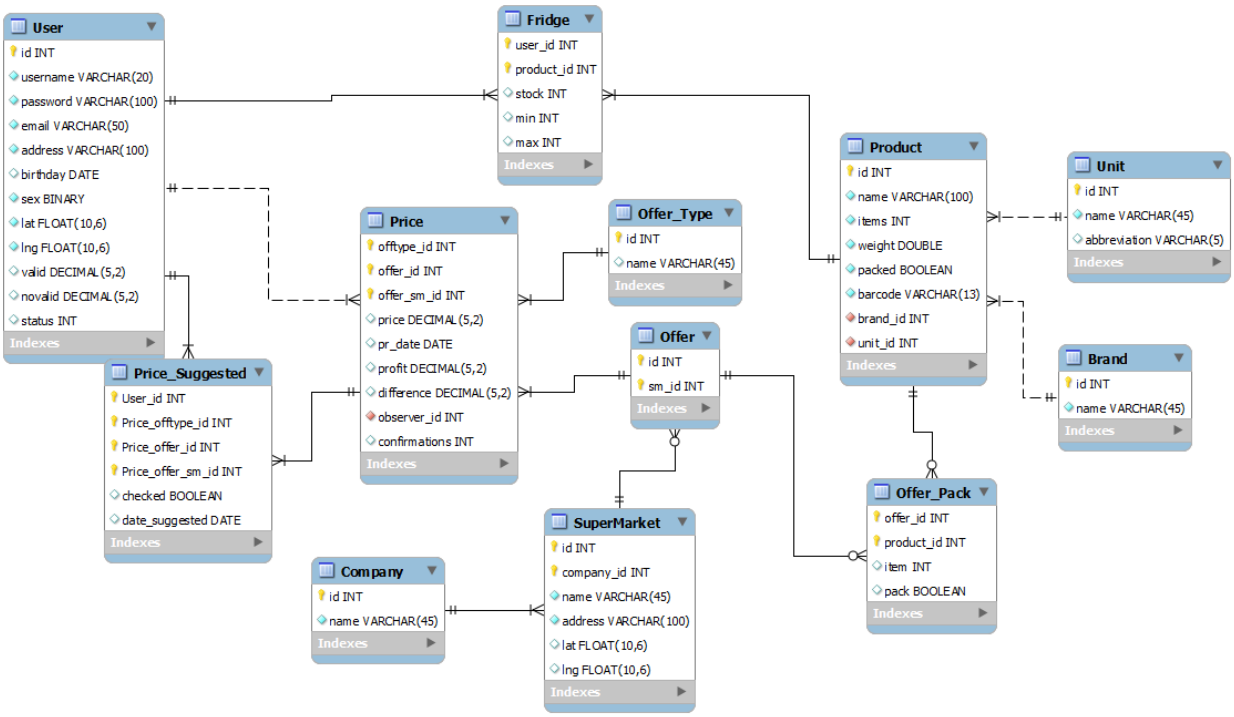
5 Σχεδίαση και Υλοποίηση

Στο κεφάλαιο αυτό παρουσιάζεται ο τρόπος με τον οποίο σχεδιάστηκε η βάση δεδομένων και η οντολογία της εφαρμογής. Στη συνέχεια αναφέρονται οι εφαρμογές που εγκαταστάθηκαν για την υλοποίηση της εφαρμογής καθώς και ο τρόπος εγκατάστασής τους. Τέλος περιγράφεται η υλοποίηση της Υπηρεσίας Ιστού και της Διαδικτυακής Εφαρμογής.

5.1 Βάση Δεδομένων

5.1.1 Σχήμα Βάσης Δεδομένων

Στην ενότητα αυτή παρουσιάζεται διαγραμματικά το σχήμα της βάσης δεδομένων της εφαρμογής που αναπτύχθηκε και περιγράφονται οι πίνακες που αποτελούν τη βάση.



Εικόνα 9: Διάγραμμα Βάσης Δεδομένων

User(id, username, password, email, address, birthday, sex, lat, lng, valid, novalid, status)

Ο κάθε χρήστης χαρακτηρίζεται από έναν μοναδικό *κωδικό* (*id*) και ένα μοναδικό *όνομα χρήστη* (*username*), κατέχει έναν *κωδικό* (*password*) με τον οποίο μπορεί να εισέλθει στο σύστημα και έχει καταχωρήσει την *ηλεκτρονική του διεύθυνση* (*email*), την *ημερομηνία γέννησης* (*birthday*) και το *φύλο* του (*sex*). Επίσης έχει εισάγει στο σύστημα τη *διεύθυνση κατοικίας* του τόσο ως αλφαριθμητικό (*address*) όσο και σε *συντεταγμένες* (*lat*, *lng*). Τέλος για κάθε χρήστη υπολογίζονται δύο αριθμοί (*valid*, *novalid*) οι οποίοι χρησιμοποιούνται για την *αξιολόγησή* του σχετικά με τις πληροφορίες που παρέχει στο σύστημα, ενώ υπάρχει και ένας αριθμός (*status*) που δηλώνει *την ανταποδοτικότητά* του προς το σύστημα (1=bronze, 2=silver, 3=gold, 4=platinum).

Product(id, name, items, weight, packed, barcode, *brand_id, unit_id*)

Ένα προϊόν έχει *κωδικό* (*id*), *όνομα* (*name*), *αριθμό τεμαχίων* που περιέχονται στη συσκευασία (*items*), *βάρος* (weight), μια τιμή *packed* που είναι true αν το αντικείμενο πωλείται *συσκευασμένο* ή false αν πωλείται *χύμα* και το *barcode*. Επίσης έχει κωδικό για τη *μάρκα* του προϊόντος (*brand_id* ξένο κλειδί από τον πίνακα Brand) και κωδικό για τη *μονάδα μέτρησης* του *βάρους* του (*unit_id* ξένο κλειδί από τον πίνακα Unit).

Brand (id, name)

Μία μάρκα κατασκευαστή χαρακτηρίζεται από έναν *κωδικό* (*id*) και το *όνομα* (*name*) που αντιστοιχεί στον κωδικό αυτό.

Unit(id, name, abbreviation)

Μία μονάδα μέτρησης αποτελείται από ένα *κωδικό* (*id*) , ένα *όνομα* που αντιστοιχεί σε αυτόν τον κωδικό (*name*) και μια *συντομογραφία* της (*abbreviation*).

Supermarket(id, *company_id*, name, address, lat, lng)

Ένα σούπερ μάρκετ διαθέτει έναν *κωδικό* (*id*), έναν κωδικό που χαρακτηρίζει σε ποια *αλυσίδα* ανήκει (*company_id*, ξένο κλειδί από τον πίνακα Company), ένα *όνομα* (*name)* και τη *διεύθυνση* του. Η διεύθυνση του αποθηκεύεται τόσο με ένα αλφαριθμητικό (*address*) όσο και με τις *συντεταγμένες* του (*lat, lng*)

Company(id, name)

Μια αλυσίδα σούπερ μάρκετ χαρακτηρίζεται από έναν *κωδικό* (*id*) και ένα *όνομα* που αντιστοιχεί σε αυτόν τον κωδικό (*name*)

Offer_Pack(*offer_id*, *product_id*, item, pack)

Ένα πακέτο προσφοράς αποτελείται από έναν *κωδικό* που χαρακτηρίζει την προσφορά (*offer_id*, ξένο κλειδί από τον πίνακα Offer) και έναν *κωδικό* (*product_id*, ξένο κλειδί από τον πίνακα Product)που χαρακτηρίζει το προϊόν που ανήκει στην προσφορά. Για κάθε προϊόν αποθηκεύεται επιπλέον ο αριθμός των *τεμαχίων* του (*items*) που προσφέρονται, ενώ για κάθε προσφορά υπάρχει η ιδιότητα *pack* η οποία είναι true αν η προσφορά αφορά ένα *μεμονωμένο προϊόν* ή false αν η προσφορά αφορά *περισσότερα του ενός προϊόντα* ή *περισσότερα του ενός τεμάχια* ενός προϊόντος. Για την αναπαράσταση μιας προσφοράς με περισσότερα του ενός προϊόντα θα υπάρχουν στον πίνακα Offer_Pack τόσες εγγραφές όσα και τα διαφορετικά προϊόντα. Οι εγγραφές αυτές θα έχουν τον ίδιο κωδικό *offer_id* αλλά διαφορετικό κωδικό προϊόντος και θα έχουν όλες τιμή true στην ιδιότητα *pack* καθώς αποτελούν πακέτο προσφοράς.

Offer(id, *sm_id*)

Κάθε προσφορά χαρακτηρίζεται από έναν *κωδικό* αριθμό (*id*) και από τον κωδικό του *καταστήματος* στο οποίο πωλείται (*sm_id*, ξένο κλειδί του πίνακα Supermarket).

Price(*offtype_id*, *offer_id*, *offer_sm_id*, price, pr_Date, profit, difference, *observer_id*, confirmations)

Κάθε τιμή που εισάγει ένας *χρήστης* (*observer_id*, ξένο κλειδί του πίνακα User) στο σύστημα αποτελείται από έναν κωδικό (*offtype_id*, ξένο κλειδί του πίνακα Offer_Type) που χαρακτηρίζει τον *τύπο της προσφοράς*, από τον *κωδικό της προσφοράς* (*offer_id*, ξένο κλειδί του πίνακα Offer), τον κωδικό του *καταστήματος* στο οποίο παρατηρείται η συγκεκριμένη τιμή (*offer_sm_id*, ξένο κλειδί του πίνακα Offer), την *τιμή* (*price*) που καταγράφεται και την *ημερομηνία καταγραφής* (*pr_Date*). Επίσης κρατείται ο αριθμός των *επιβεβαιώσεων* (*confirmations*) που έχει λάβει η συγκεκριμένη τιμή σχετικά με την εγκυρότητά της από άλλους χρήστες. Τέλος για κάθε εγγραφή υπολογίζεται η *διαφορά* της (*difference*) από την μικρότερη τιμή της αντίστοιχης προσφοράς στο σύνολο των καταστημάτων και για την περίπτωση των πακέτων προσφορών υπολογίζεται το *κέρδος* (*profit*) που θα αποφέρει η αγορά του πακέτου προσφοράς σε σχέση με την αγορά των μεμονωμένων προϊόντων της προσφοράς από το ίδιο σούπερ μάρκετ.

Offer_Type(id, name)

Ο τύπος μιας προσφοράς αντιπροσωπεύεται από έναν *κωδικό* αριθμό (*id*) και από το *όνομα* του (*name*). Πιθανοί τύποι προσφορών είναι «για όλους» ή «για μέλη».

Price_Suggested(*User_id*, *Price_offtype_id*, *Price_offer_id*, *Price_offer_sm_id*, checked, date_suggested)

Οι προτεινόμενες τιμές είναι οι τιμές που προτείνει το σύστημα στο χρήστη ως καλύτερες σύμφωνα με τη λίστα προϊόντων που έχει εισάγει και τα καταστήματα που έχει επιλέξει. Κάθε εγγραφή του πίνακα αποτελείται από τον *κωδικό* του *χρήστη* (*User_id*, ξένο κλειδί του πίνακα User), τον *τύπο* της *προσφοράς* (*Price_offtype_id*, ξένο κλειδί του πίνακα Price), τον *κωδικό* αριθμό της *προσφοράς* (*Price_offer_id*, ξένο κλειδί του πίνακα Price) , τον κωδικό του *καταστήματος* από το οποίο θα το αγοράσει (*Price_offer_sm_id*, ξένο κλειδί του πίνακα Price), την ιδιότητα *checked* η οποία έχει αρχική τιμή

false και γίνεται true αφού *ελεγχθεί η εγκυρότητά* της από τον χρήστη και η *ημερομηνία* που έγινε η πρόταση(*date_suggested*).

Fridge(user_id, product_id, stock, min, max)

Ο πίνακας αυτός αντιπροσωπεύει το ηλεκτρονικό ψυγείο που μπορεί να συνδεθεί με την εφαρμογή. Κάθε εγγραφή χαρακτηρίζεται από τον *κωδικό* του *χρήστη* (*user_id*, ξένο κλειδί του πίνακα User) , έναν κωδικό *προϊόντος* (*product_id*, ξένο κλειδί του πίνακα Product), τον αριθμό των *τεμαχίων* που υπάρχουν *στο ψυγείο* (*stock*) και το *μέγιστο* (*max*) και *ελάχιστο* (*min*) αριθμό των προϊόντων που *επιθυμούμε* να υπάρχουν στο ψυγείο.

5.1.2 Συσχετίσεις μεταξύ πινάκων

Στη συνέχεια αναφέρονται οι τύποι των συσχετίσεων μεταξύ των πινάκων της βάσης στις οποίες οφείλεται και η δομή των πινάκων όπως παρουσιάστηκαν. Αρχικά αναφέρονται τα ονόματα των πινάκων και στη συνέχεια ο τύπος της συσχέτισης που μπορεί να είναι *ένα προς ένα* (1 : 1), *ένα προς πολλά* (1 : n) ή *πολλά προς πολλά* (n : n).

Product – Unit (n : 1) : Πολλά προς ένα, 1 Product έχει μόνο 1 Unit και 1 Unit μπορεί να συσχετίζεται με 0,1 ή πολλά Product. Λόγω αυτής της συσχέτισης προστέθηκε στον πίνακα Product το πεδίο unit_id ως ξένο κλειδί.

Product – Brand (n : 1) : Πολλά προς ένα, 1 Product έχει μόνο 1 Brand και 1 Brand μπορεί να συσχετίζεται με 0,1 ή πολλά Product. Για αυτό το λόγο προστέθηκε στον πίνακα Product το πεδίο brand_id ως ξένο κλειδί.

Product – Offer (n : n) : Πολλά προς πολλά, 1 Product μπορεί να ανήκει σε 0,1 ή πολλές Offer και 1 Offer μπορεί να περιέχει 1 ή πολλά Product. Αυτή η συσχέτιση οδήγησε στη δημιουργία του πίνακα Offer_Pack με κλειδί τον συνδυασμό των πρωτευόντων κλειδιών των δύο πινάκων.

Product – User (n : n) : Πολλά προς πολλά, 1 Product μπορεί να το έχουν στην κατοχή τους 0,1 ή πολλοί Users και 1 User μπορεί να έχει στην κατοχή του 0,1 ή πολλά Products. Βάσει αυτής της συσχέτισης προέκυψε ο πίνακας Fridge με κύριο κλειδί τον συνδυασμό των πρωτευόντων κλειδιών των δύο πινάκων.

Offer – SuperMarket (n : n) : Πολλά προς πολλά, 1 Offer μπορεί να υπάρχει σε πολλά SuperMarket και 1 SuperMarket μπορεί να έχει 0,1 ή πολλά Offer. Αυτή η συσχέτιση υποδεικνύει πως είναι αναγκαία η ύπαρξη του συνδυασμού κλειδιών των πινάκων Offer και SuperMarket, για αυτό το λόγο προστέθηκε στον πίνακα Offer το κλειδί του πίνακα Supermarket και αποτελούν μαζί το κύριο κλειδί του πίνακα Offer.

SuperMarket – Company (1 : n) : Ένα προς πολλά, 1 Supermarket ανήκει μόνο σε 1 Company αλλά 1 Company μπορεί να έχει 0,1 ή πολλά Supermarket. Λόγω αυτής της συσχέτισης το πρωτεύον κλειδί του πίνακα Company προστίθεται στον πίνακα Supermarket.

Offer – Offer_Type (n : n) : Πολλά προς πολλά, 1 Offer πωλείται σε 1 ή πολλά Offer_Type (μπορεί να πωλείται με άλλη τιμή για τα μέλη ενός SuperMarket και με άλλη τιμή για τους υπόλοιπους πελάτες) και 1 Offer_Type μπορεί να διατίθεται για 0, 1 ή πολλά Offer. Από τη συσχέτιση αυτή προκύπτει ο πίνακας Price με κύριο κλειδί το συνδυασμό των κλειδιών των πινάκων Offer και Offer_Type.

User – Price (n : n) : Πολλά προς πολλά, 1 User έχει λάβει πρόταση από το σύστημα για 1 ή πολλές Prices και 1 Price έχει προταθεί σε 0,1 ή πολλούς Users. Έτσι προκύπτει ο πίνακας Price_Suggested ο οποίος έχει ως κλειδί το συνδυασμό των πρωτευόντων κλειδιών των πινάκων User και Price.

User – Price (n : 1) : Πολλά προς ένα, 1 User μπορεί να εισάγει στο σύστημα (να παρατηρήσει) 0, 1 ή πολλές Prices αλλά 1 Price μπορεί να παρατηρηθεί

μόνο από 1 User. Για το λόγο αυτό στον πίνακα Price υπάρχει και ως ξένο κλειδί το κλειδί του πίνακα User.

5.1.3 Οντολογία κατηγοριών

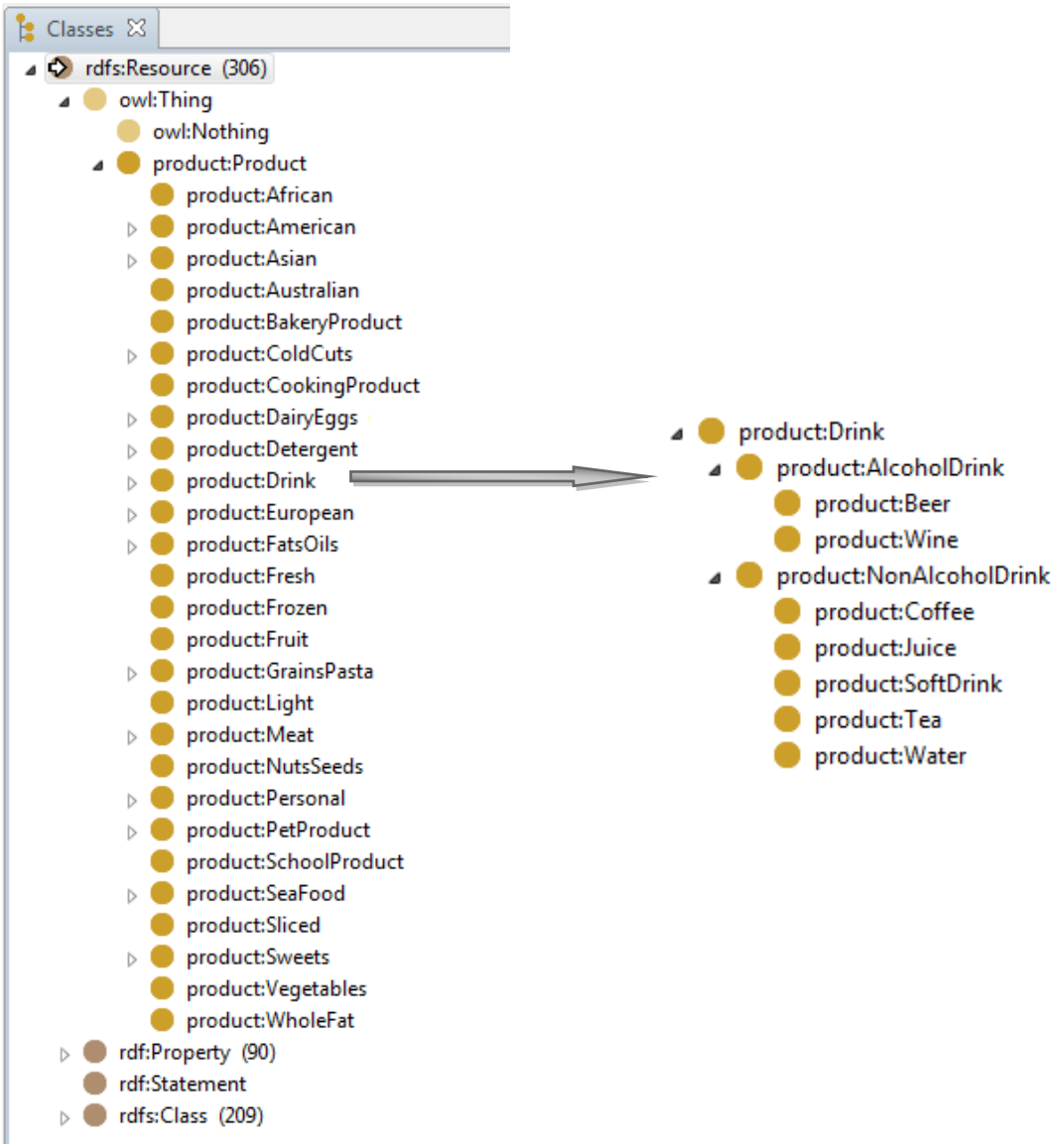
Για την απεικόνιση των κατηγοριών στις οποίες μπορεί να ανήκει ένα προϊόν αντί για πίνακα στην προηγούμενη βάση δεδομένων χρησιμοποιήθηκε μια οντολογία. Η επιλογή της οντολογίας και της αναπαράστασης των κατηγοριών σε OWL και RDF καθιστά πιο εύκολη την αναπαράσταση των σχέσεων μεταξύ των κατηγοριών και την εκτέλεση ερωτημάτων που τις αφορούν. Για παράδειγμα είναι πιο εύκολη η αναπαράσταση της σχέσης «υποκλάση» που υφίσταται μεταξύ 2 κλάσεων, όπως επίσης μπορεί εύκολα να βρεθεί ποια είναι τα προϊόντα που ανήκουν σε μια συγκεκριμένη κατηγορία συμπεριλαμβανομένων και όλων των προϊόντων των υποκλάσεων της.

Η δημιουργία της οντολογίας έγινε με τη βοήθεια του περιβάλλοντος TopBraid Composer. Αρχικά δημιουργήθηκε η κλάση *Προϊόν* και στη συνέχεια δημιουργήθηκαν οι κατηγορίες των προϊόντων. Υπάρχουν τόσο κατηγορίες που αφορούν τη *γεωγραφική προέλευση* των προϊόντων (π.χ. Ευρωπαϊκό, Ασιατικό κτλ) όσο και κατηγορίες που αφορούν το *είδος* των προϊόντων (π.χ. Φρούτο, Ποτό, Αυγά/Γαλακτοκομικά, Ζυμαρικά κτλ). Επίσης υπάρχουν κατηγορίες που χαρακτηρίζουν την *κατάσταση* των προϊόντων όπως Κατεψυγμένο, Φρέσκο ή Σε φέτες αλλά και τη *διατροφική αξία* τους όπως Ελαφρύ ή Πλήρες.

Η κλάση *Προϊόν* (*product:Product*) είναι τύπου (*rdf:type*) *owl:Class* και υποκλάση (*rdfs*:*subClassOf*) της κλάσης *owl:Thing*. Το namespace product αντιπροσωπεύει το URI http://example.org/product# . Ενδεικτικά η κατηγορία προϊόντος *Ποτό* (*product*:*Drink*) είναι τύπου (*rdf*:*type*) *owl*:*Class* και υποκλάση (*rdfs*:*subClassOf*) της *product*:*Product*. Η κατηγορία *Ποτό* διαθέτει δύο υποκλάσεις (*AlcoholDrink* και *NonAlcoholDrink*) καθεμιά από τις οποίες διαθέτει τις δικές της υποκλάσης. Η αναπαράσταση κάθε κλάσης έγινε όπως και προηγουμένως, δηλαδή η κλάση Αλκοολούχο Ποτό (*AlcoholDrink*) είναι

τύπου (*rdf*:*type*) *owl*:*Class* και υποκλάση (*rdfs*:*subClassOf*) της *product*:*Drink*, η κλάση Μη Αλκοολούχο Ποτό (*NonAlcoholDrink*) το ίδιο κ.ο.κ.

Στην **Εικόνα 10** φαίνονται οι κλάσεις της οντολογίας και συγκεκριμένα οι υποκλάσεις της κλάσης *Προϊόν* και ενδεικτικά η κλάση *Ποτό* με όλες τις υποκλάσεις της.



Εικόνα 10: Η Οντολογία των Προϊόντων.

Κατά την εισαγωγή ενός προϊόντος στη βάση δεδομένων αυτό προστίθεται και στην οντολογία. Αυτό γίνεται με την προσθήκη μιας δήλωσης του τύπου (*προϊόν, είναι τύπου, κατηγορία*). Πιο συγκεκριμένα χρησιμοποιείται ο κωδικός αριθμός του προϊόντος (*product id*) για να χαρακτηρίσει το προϊόν και το όνομα της κατηγορίας. Για παράδειγμα η δήλωση ότι το προϊόν 11

είναι της κατηγορίας «Νερό» θα είναι η εξής: <http://example.org/product#11> rdf:type product:Water.

Με βάση την παραπάνω οντολογία το ερώτημα SPARQL που θα εμφανίσει όλα τα προϊόντα που είναι Ποτά είναι το εξής:

```
SELECT ?item
WHERE { ?category rdfs:subClassOf* product:Drink.
        ?item rdf:type ?category }
```

5.2 Εγκατάσταση όλων των εφαρμογών

5.2.1 Διαδικτυακή εφαρμογή

5.2.1.1 XAMPP

Η εγκατάσταση του *Apache server*, της *MySQL* και του διερμηνευτή της *PHP* έγινε με την εγκατάσταση του *XAMPP*. Η εγκατάσταση του *XAMPP* είναι αρκετά εύκολη. Μετά το κατέβασμα του αρχείου από την επίσημη σελίδα αρκεί να ακολουθηθούν οι οδηγίες της εγκατάστασης. Μετά την ολοκλήρωση της πληκτρολογώντας «localhost» στο πρόγραμμα περιήγησης εμφανίζεται η **Εικόνα 11**. Στην εικόνα φαίνεται το μήνυμα επιτυχής εγκατάστασης του πακέτου και διάφορες πληροφορίες σχετικά με την *PHP*, την *Perl* και τον *Apache Tomcat* που εγκαταστάθηκε. Επίσης διατίθενται παραδείγματα αλλά και εργαλεία όπως το *phpMyAdmin* που επιτρέπει στον προγραμματιστή να διαχειρίζεται τη *MySQL* χρησιμοποιώντας ένα πρόγραμμα περιήγησης. Οι ενέργειες που μπορούν να εκτελεστούν είναι δημιουργία, αλλαγή ή διαγραφή βάσεων, πινάκων, πεδίων ή σειρών, εκτέλεση ερωτημάτων *SQL* ή διαχείριση χρηστών και δικαιωμάτων.

Εικόνα 11: Η σελίδα "localhost" μετά την εγκατάσταση του XAMPP

5.2.1.2 MySQL Workbench

https://www.mysql.com/products/workbench/

Για τη δημιουργία της βάσης δεδομένων και του σχεδιαγράμματος της χρησιμοποιήθηκε το πρόγραμμα *MySQL Workbench*. Το πρόγραμμα αυτό επιτρέπει την οπτική σχεδίαση του μοντέλου μιας βάσης, την παραγωγή της βάσης και τη διαχείρισή της. Μπορεί να δημιουργήσει διαγράμματα ER και από αυτά να παράγει τον κώδικα SQL για τη δημιουργία της βάσης ή από μια υπάρχουσα βάση να δημιουργήσει το διάγραμμα ER που της αντιστοιχεί. Επίσης επιτρέπει τη δημιουργία, εκτέλεση και οπτικοποίηση ερωτημάτων SQL. Αποτελεί ένα χρήσιμο εργαλείο για τη διαχείριση μιας βάσης δεδομένων.

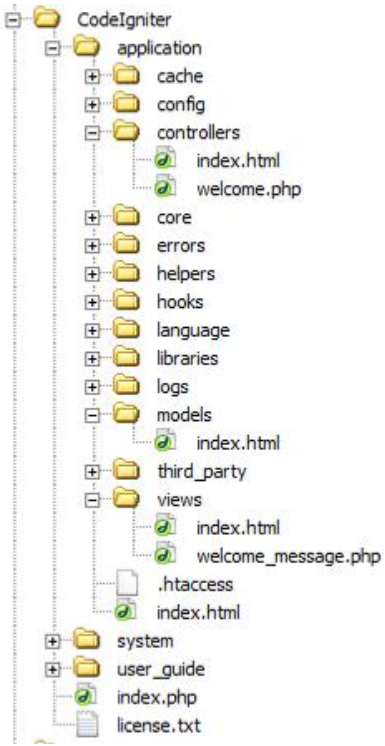
Μέσω του MySQL Workbench δημιουργήθηκε η βάση με το όνομα «market» και οι πίνακες που παρουσιάστηκαν στην προηγούμενη ενότητα.

5.2.1.3 CodeIgniter

Για την εγκατάσταση του απαιτείται ο *διερμηνευτής της PHP*, μια *βάση δεδομένων* (π.χ. MySQL) και ένας *εξυπηρετητής του παγκόσμιου ιστού* (web server). Επίσης ένας *συντάκτης* (editor) PHP (π.χ. Notepad++ ή Dreamweaver) βοηθάει στην αποφυγή και τον εντοπισμό απλών

συντακτικών λαθών κατά τη συγγραφή του κώδικα. Από τη στιγμή που εγκατασταθούν τα παραπάνω η εγκατάσταση του *CodeIgniter* είναι απλή. Κατεβάζει ο χρήστης το φάκελο των αρχείων του *CodeIgniter* από τη σελίδα του framework και τον τοποθετεί στο root φάκελο της εφαρμογής. Στον *XAMPP* ο φάκελος αυτός είναι συνήθως ο φάκελος htdocs [36]

Ο φάκελος του *CodeIgniter* φαίνεται στην **Εικόνα 12**. Αποτελείται από τους φακέλους «application», «system» και «user_guide». Στο φάκελο «application» υπάρχουν οι φάκελοι «models», «views» και «controllers» που αποτελούν τα συστατικά του *προτύπου MVC*. Με αυτό τον τρόπο είναι ευκολότερη η δόμηση της εφαρμογής.



Εικόνα 12: Ο φάκελος του CodeIgniter

Η έκδοση του *CodeIgniter* που χρησιμοποιήθηκε είναι η 2.1.0

Πριν τη συγγραφή κώδικα πρέπει να γίνουν κάποιες ρυθμίσεις του *CodeIgniter*.

- *Ρυθμίσεις βάσης δεδομένων*: Στο φάκελο application -> config στο αρχείο «database.php» εισάγονται τα στοιχεία της βάσης (hostname, username, password, database, dbdriver, char_ser, dbcollat). Στη δική μας περίπτωση όμως επειδή η διαδικτυακή εφαρμογή δεν

επικοινωνεί απευθείας με τη βάση αλλά μέσω της Υπηρεσίας Ιστού η παραπάνω ρύθμιση είναι περιττή.

- *Ρύθμιση κεντρικού Ελεγκτή*: Στο φάκελο application -> config στο αρχείο «routes.php» ορίζεται ο Ελεγκτής που θα καλείται όταν κληθεί αρχικά η εφαρμογή. Στη συγκεκριμένη περίπτωση ο Ελεγκτής που θα καλείται αρχικά είναι ο «home» επομένως η ρύθμιση θα είναι $route['default_controller'] = “home” .
- *Ρυθμίσεις βιβλιοθηκών και helper του framework*: Στο φάκελο application -> config στο αρχείο «autoload.php» υπάρχει η δυνατότητα να εισαχθούν οι βιβλιοθήκες ή/και τα βοηθητικά αρχεία (helper files) που επιθυμεί ο προγραμματιστής να φορτώνονται αυτόματα για όλους τους Ελεγκτές. Οι βιβλιοθήκες και τα βοηθητικά αρχεία αποτελούν σύνολα έτοιμων συναρτήσεων που βοηθούν στη συγγραφή κώδικα. Ενδεικτικά στην εφαρμογή οι βιβλιοθήκες και τα βοηθητικά αρχεία που φορτώνονται αυτόματα είναι:
  $autoload['libraries'] = array('session','javascript','form_validation');
  $autoload['helper'] = array('security','url','form','html');
- *Ρύθμιση κλειδιού κρυπτογράφησης*: Σε περίπτωση που χρησιμοποιηθεί η κλάση Encryption ή η κλάση Session πρέπει να δοθεί και ένα κλειδί κρυπτογράφησης. Το κλειδί κρυπτογράφησης δίνεται στο φάκελο application -> config στο αρχείο «config.php». Στη συγκεκριμένη περίπτωση μιας και χρησιμοποιείται η κλάση Session δόθηκε μια μεγάλη τιμή στη μεταβλητή $config['encryption_key'] .

5.2.2 Υπηρεσίες Ιστού

5.2.2.1 Eclipse

Για την ανάπτυξη της υπηρεσίας Ιστού χρησιμοποιήθηκε η γλώσσα προγραμματισμού Java και το περιβάλλον της Eclipse.

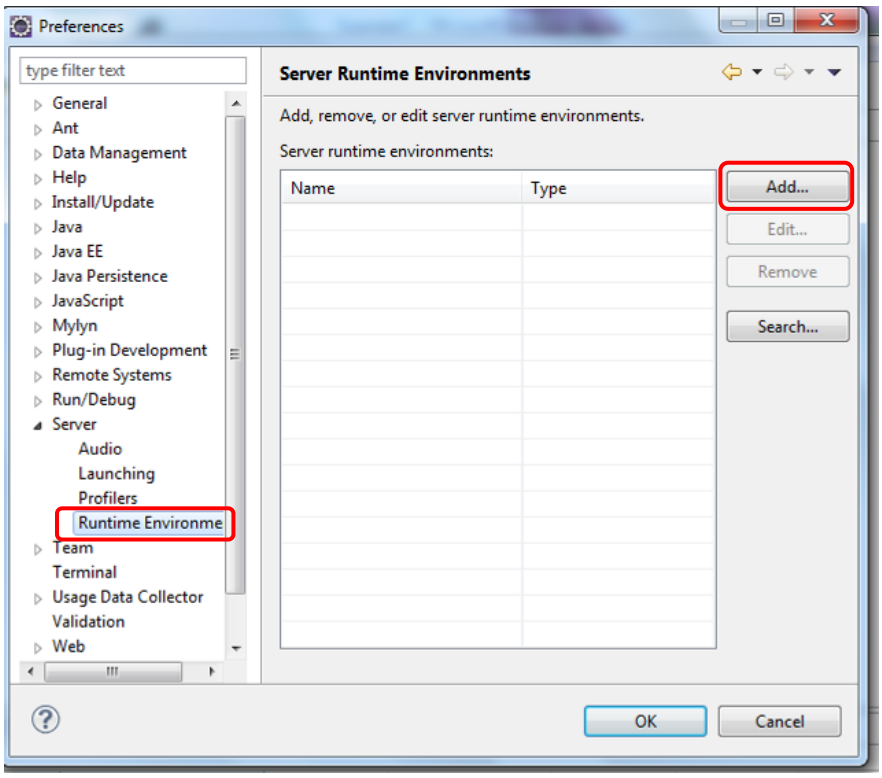
Μετά την εγκατάσταση του Eclipse πρέπει να γίνουν οι κατάλληλες ρυθμίσεις για την ανάπτυξη υπηρεσίας Ιστού. Αρχικά πρέπει να δηλωθεί ο εξυπηρετητής στον οποίο θα τρέχει η Υπηρεσία Ιστού. Συγκεκριμένα θα

δηλωθεί ο Apache Tomcat που εγκαταστάθηκε με το XAMPP. Στη συνέχεια θα δηλωθεί το Axis2.

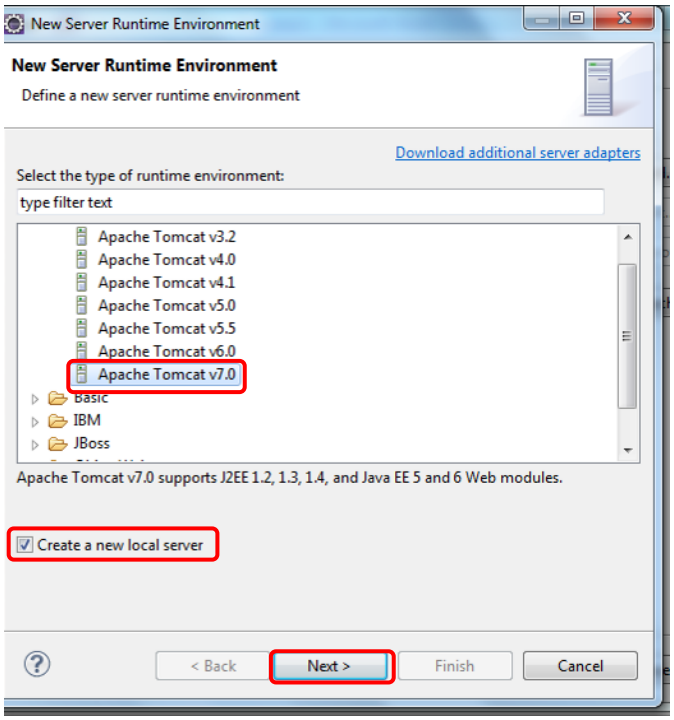
5.2.2.2 Apache Tomcat στο Eclipse

Για να δηλωθεί ο Apache Tomcat ως ο εξυπηρετητής ιστού στον οποίο θα εκτελείται η Υπηρεσία Ιστού γίνονται οι ακόλουθες κινήσεις.

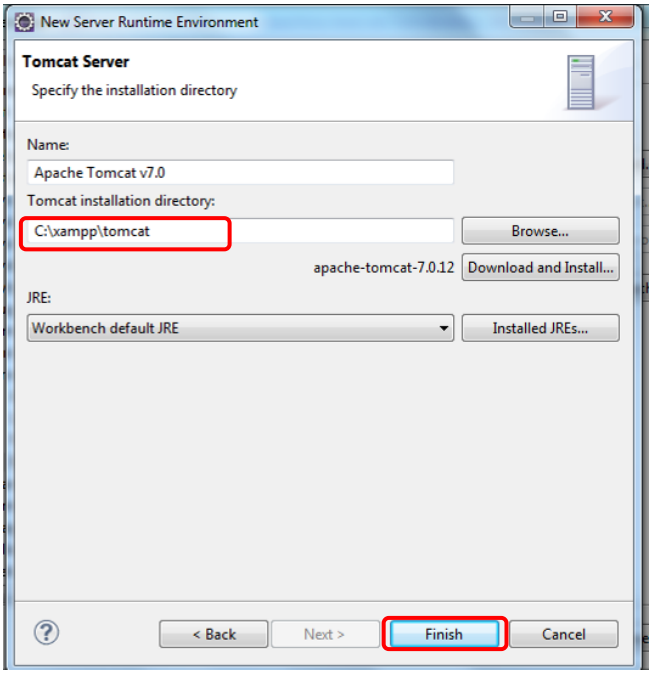
1. Άνοιγμα του Windows -> Preferences -> Server -> Runtime Environment , επιλογή «Add..» (**Εικόνα 13)**
2. Επιλογή του Apache Tomcat v7.0, επιλογή του «Create a new local server» και «Next» (**Εικόνα 14)**
3. Επιλογή της διαδρομής αρχείου του tomcat και «Finish» (**Εικόνα 15**)
4. Επιλογή του «OK» στο παράθυρο «Server Runtime Environments» όπου εμφανίζεται ότι έχει προστεθεί ο Apache Tomcat v7.0 (**Εικόνα 16**)



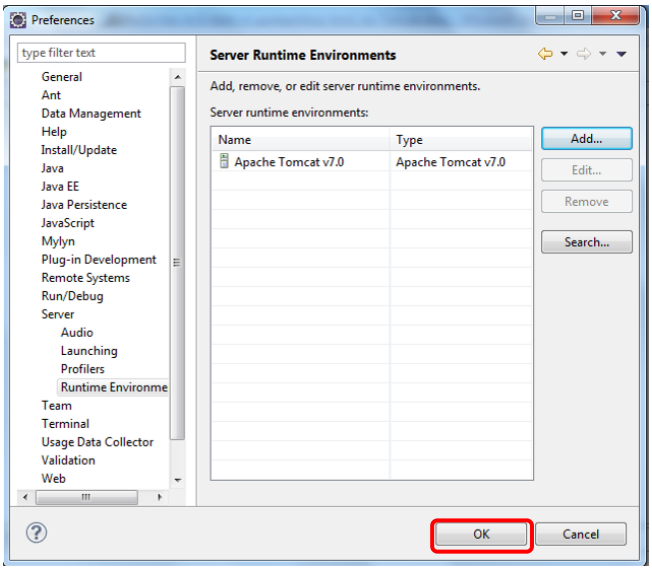
Εικόνα 13: Οι Ρυθμίσεις για το περιβάλλον εκτέλεσης του εξυπηρετητή στο Eclipse



Εικόνα 14: Προσθήκη εξυπηρετητή



Εικόνα 15: Θέτοντας τη διαδρομή του Apache



Εικόνα 16: Επιτυχής εγκατάσταση του εξυπηρετητή

5.2.2.3 Axis2

Το Apache Axis2 αποτελεί μια μηχανή Υπηρεσιών Ιστού / SOAP / WSDL, απόγονο του Apache Axis. Αποτελεί μια βασισμένη σε Java υλοποίηση τόσο του πελάτη όσο και του εξυπηρετητή μιας υπηρεσίας Ιστού. Μερικές από τις κυριότερες εργασίες που επιτρέπει να γίνουν με ευκολία είναι:

- αποστολή μηνυμάτων SOAP,
- παραλαβή και επεξεργασία μηνυμάτων SOAP,
- δημιουργία μιας Υπηρεσίας Ιστού από μια απλή κλάση της Java ,
- δημιουργία κλάσεων υλοποίησης για τον εξυπηρετητή και τον πελάτη χρησιμοποιώντας το αρχείο WSDL
- εύκολη ανάκτηση του αρχείου WSDL μιας υπηρεσίας [37]

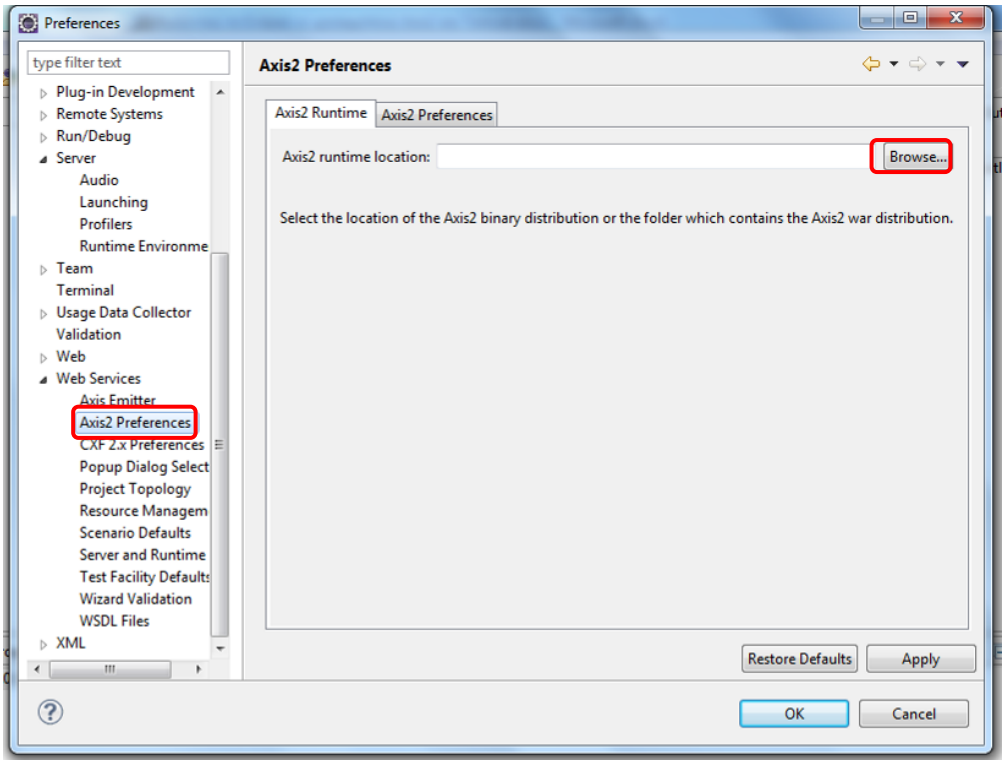
Για την εγκατάσταση του Axis2 ακολουθείται η εξής διαδικασία:

1. Κατέβασμα της «Standard Binary Distribution» από την επίσημη ιστοσελίδα του Axis2
2. Τοποθέτηση του φακέλου axis2-1.6.2 απευθείας στον C:\

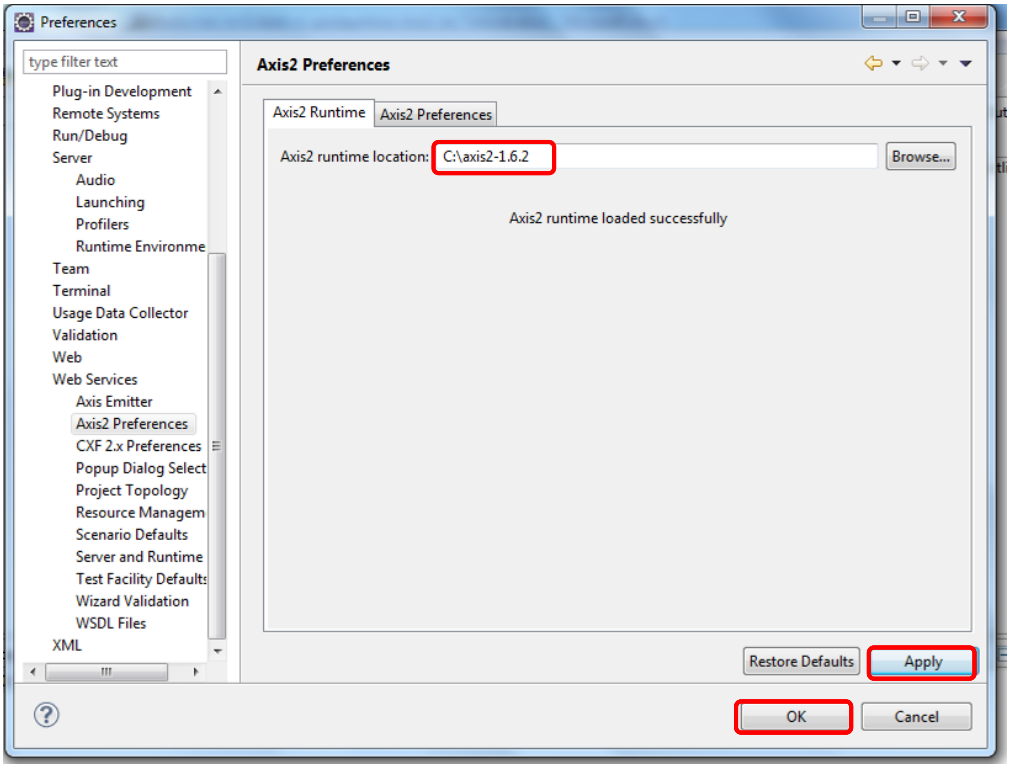
Εγκατάσταση του Axis2 στο Eclipse:

1. Επιλογή του Window -> Preferences -> Web Services -> Axis2 Preferences, επιλογή «Browse» (**Εικόνα 17**)

2. Δήλωση της διαδρομής του φακέλου του Axis2, επιλογή «Apply» και «OK» (**Εικόνα 18**)



Εικόνα 17:Οι ρυθμίσεις για το Axis2



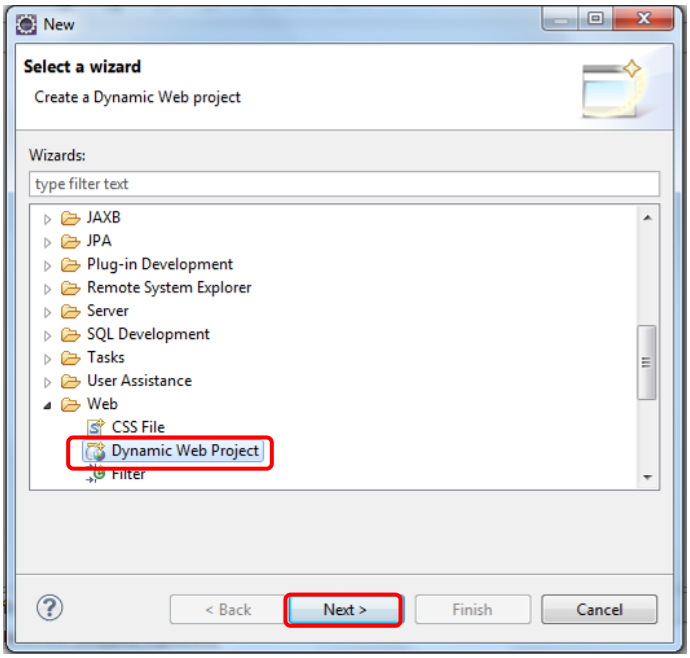
Εικόνα 18: Διαδρομή του φακέλου Axis2-1.6.2

5.2.2.4 Δημιουργία Υπηρεσίας Ιστού στο Eclipse

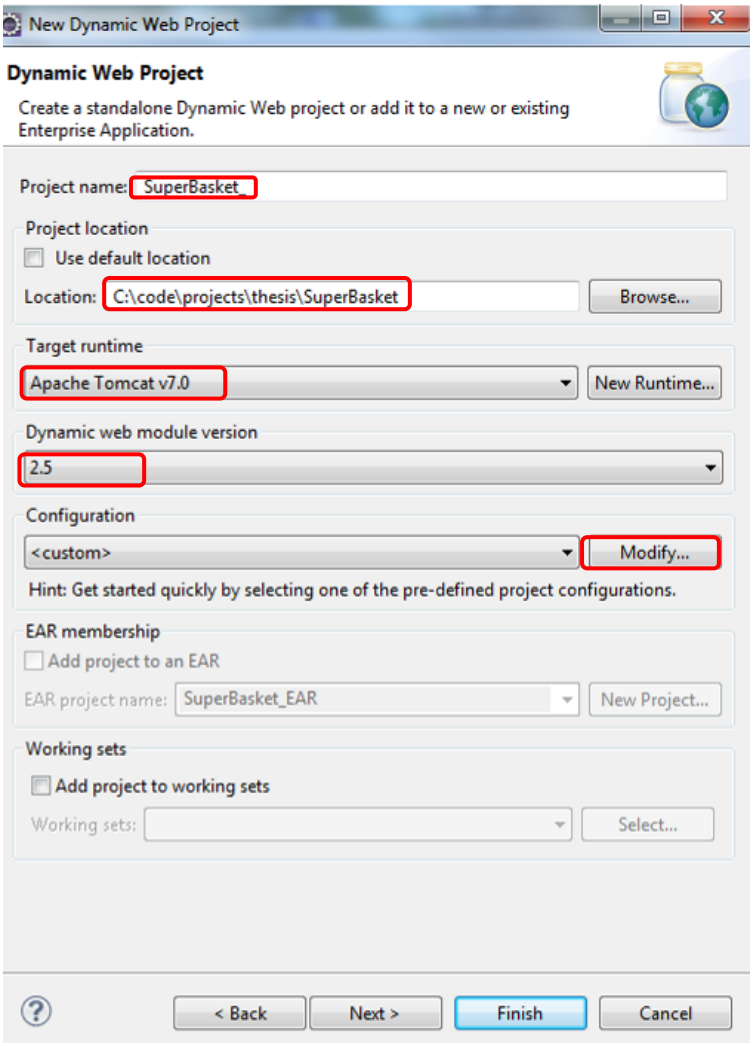
Για τη δημιουργία του project στο Eclipse ακολουθήθηκαν τα παρακάτω βήματα:

1. Επιλογή File -> New -> Other -> Web -> Dynamic Web Project, επιλογή «Next» (**Εικόνα 19**)
2. Πληκτρολόγηση ονόματος για το project, επιλογή φακέλου αποθήκευσης, επιλογή εξυπηρετητή στον οποίο θα τρέξει (Apache Tomcat v7.0), επιλογή της έκδοσης 2.5 για το «Dynamic web module» και επιλογή του «Modify» (**Εικόνα 20**)
3. Επιλογή των Axis2 Web Service, Dynamic Web Module, Java και JavaScript και επιλογή του «OK» (**Εικόνα 21**)
4. Επιλογή «Finish»
5. Δημιουργία της κλάσης «MyService» μέσα στο φάκελο src και ορισμός μια απλής συνάρτησης
6. Στον Project Explorer δεξί κλικ στην κλάση MyService -> Web Services -> Create Web Service
7. Επιλογή του Web Service runtime Apache Axis (**Εικόνα 22**)
8. Επιλογή του Apache Axis2 και του «OK» (**Εικόνα 23**)
9. Επιλογή «Finish»

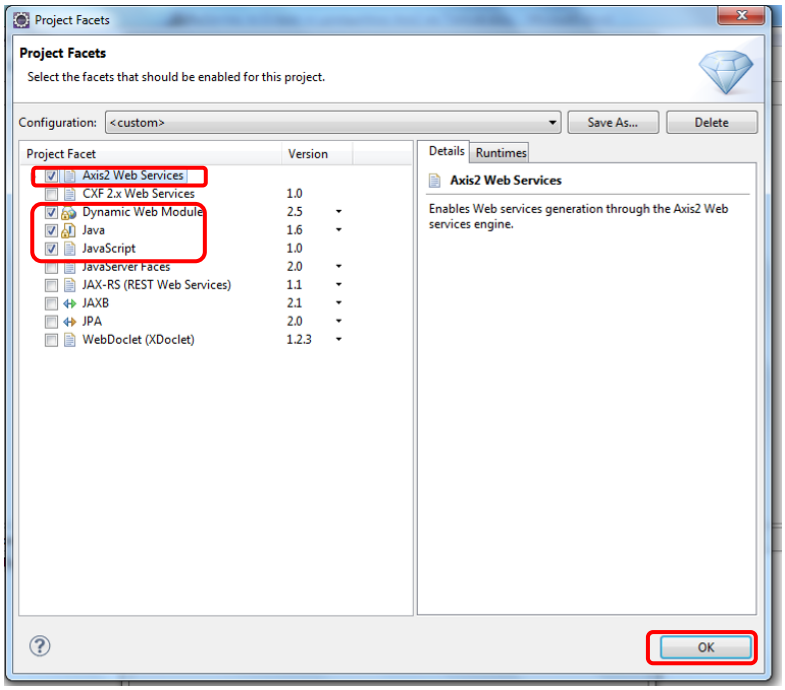
Η υπηρεσία μπορεί να προσπελαστεί στη διεύθυνση http://localhost:8080/SuperBasket/



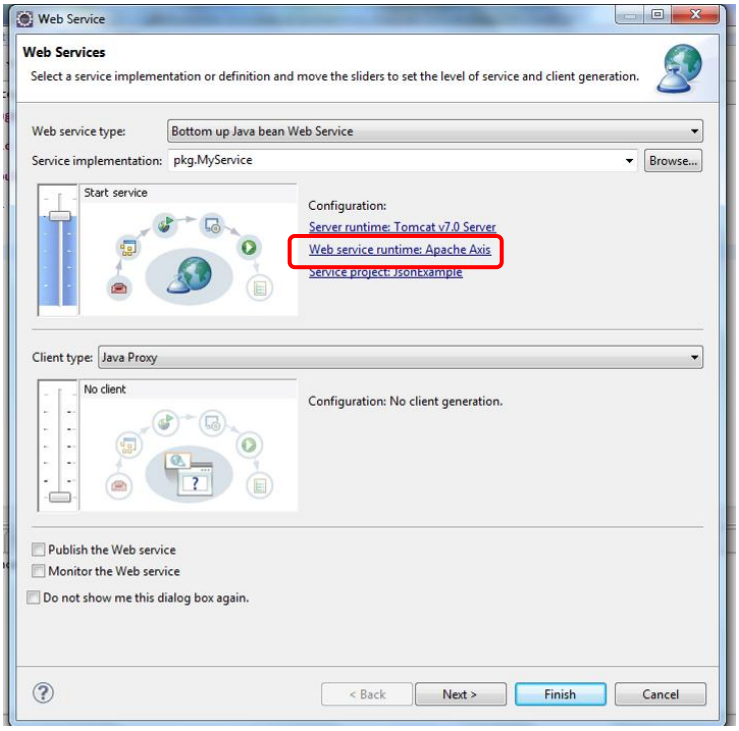
Εικόνα 19: Επιλογή τύπου νέου project



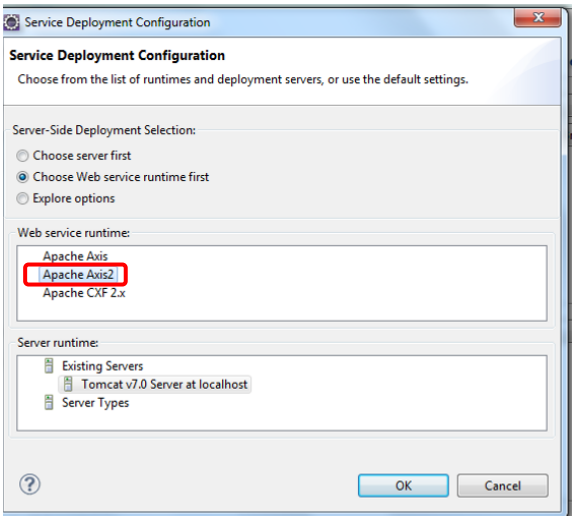
Εικόνα 20:Εισαγωγή πληροφοριών καινούριου project



Εικόνα 21: Επιλογή τεχνολογιών για το καινούριο project



Εικόνα 22:Δημιουργία της Υπηρεσίας Ιστού



Εικόνα 23:Επιλογή του Axis2

5.2.2.5 Jena

Για τη χρήση του Jena μέσω του Eclipse είναι απαραίτητες οι βιβλιοθήκες του, μια κενή βάση δεδομένων για το «*persistent store*» και ο ορισμός του MySQL JDBC driver στο classpath. Για την εγκατάσταση μιας βιβλιοθήκης χρήστη στο Eclipse αρκεί να προστεθούν τα απαραίτητα αρχεία με το όνομα της βιβλιοθήκης (π.χ. JenaLib) μέσα από το Window -> Preferences -> Java ->Build Path -> User Libraries. Τα απαραίτητα αρχεία για τη βιβλιοθήκη βρίσκονται στην ιστοσελίδα του Apache Jena.

5.2.2.6 SoapUI

Το SoapUI είναι ένα δωρεάν και ανοικτού κώδικα λογισμικό το οποίο επιτρέπει τη διενέργεια ελέγχων. Χρησιμοποιήθηκε για τον έλεγχο των λειτουργιών της υπηρεσίας ιστού. Παρέχοντας στην εφαρμογή την περιγραφή WSDL της υπηρεσίας ιστού είναι δυνατόν να στείλει SOAP ερωτήματα σε μια συγκεκριμένη λειτουργία και να λάβει την αντίστοιχη SOAP απόκριση.

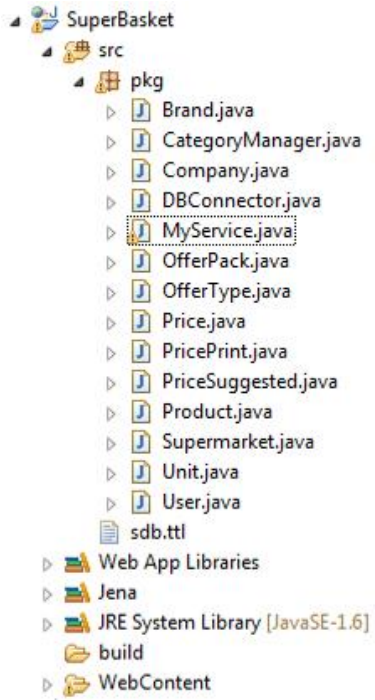
5.3 Δημιουργία εφαρμογής

Μετά την εγκατάσταση των απαραίτητων εφαρμογών και εργαλείων άρχισε η ανάπτυξη της Υπηρεσίας Ιστού και της διαδικτυακής εφαρμογής. Η ανάπτυξή τους έγινε παράλληλα καθώς μετά την ανάπτυξη κάθε λειτουργίας -

συνάρτησης της Υπηρεσίας Ιστού αναπτυσσόταν και το αντίστοιχο μέρος της διαδικτυακής εφαρμογής που θα καλούσε τη λειτουργία για να είναι ευκολότερος ο έλεγχος. Για λόγους ελέγχους χρησιμοποιήθηκε και η εφαρμογή SoapUI για απευθείας κλήσεις προς την Υπηρεσία Ιστού.

5.3.1 Υπηρεσία Ιστού

Η Υπηρεσία Ιστού υλοποιήθηκε ως μια κλάση Java η οποία δημοσιεύεται με τη βοήθεια των Eclipse, Axis2 και Apache Tomcat. Η Υπηρεσία Ιστού είναι υπεύθυνη για την επικοινωνία με τη βάση δεδομένων και για την εκτέλεση των λειτουργιών της εφαρμογής. Λόγω της επικοινωνίας με τη βάση είναι απαραίτητη η δημιουργία επιπλέον κλάσεων, πέρα της κλάσης της υπηρεσίας, που θα αντιπροσωπεύουν και θα αποθηκεύουν δεδομένα που προέρχονται από τον κάθε πίνακα της βάσης. Επιπλέον είναι χρήσιμη και μια κλάση η οποία θα χειρίζεται τη σύνδεση με τη βάση δεδομένων αλλά και μια κλάση για το χειρισμό της οντολογίας. Έτσι οι κλάσεις που δημιουργήθηκαν είναι αυτές που φαίνονται στην **Εικόνα 24**.



Εικόνα 24: Κλάσεις του προγράμματος Java - Υπηρεσία Ιστού

5.3.1.1 Περιγραφή κλάσεων

Η περιγραφή ξεκινά με τις κλάσεις όπως φαίνονται στην **Εικόνα 24**.

Brand : Η κλάση Brand χρησιμοποιείται για να αποθηκεύει αντικείμενα τύπου Brand, δηλαδή αντικείμενα που αποτελούν μια σειρά του πίνακα Brand της βάσης δεδομένων.

- Τα πεδία της κλάσης αντιστοιχούν στις στήλες του πίνακα Brand (int id, String name).
- Οι μέθοδοι της κλάσης είναι οι κατασκευαστές και οι κλασσικές μέθοδοι get/set για κάθε πεδίο.

CategoryManager : Η κλάση CategoryManager είναι υπεύθυνη για το χειρισμό της οντολογίας Product.

- Τα πεδία της είναι το *Store* που αντιπροσωπεύει τη βάση στην οποία αποθηκεύεται το μοντέλο της οντολογίας και τρεις σταθερές τύπου αλφαριθμητικά, το *όνομα του αρχείου* (product.rdf) όπου είναι αποθηκευμένη η οντολογία όπως δημιουργήθηκε στο πρόγραμμα TopBraid Composer, το *URI* (http://example.org/product#) που αντιστοιχεί στην οντολογία και το *namespace* της (product).
- Οι μέθοδοι της κλάσης χρησιμοποιούν τη βιβλιοθήκη του Jena.
  - *initDBandStore()* : συνδέεται με τη βάση δεδομένων, το Store, στην οποία θα αποθηκευτεί η οντολογία και τη διαμορφώνει αν δεν έχει ήδη διαμορφωθεί. Οι πληροφορίες για το όνομα της βάσης και τις διάφορες ρυθμίσεις λαμβάνονται από ένα αρχείο ttl (sdb.ttl).
  - *insertStoreData()* : κλήθηκε μόνο την πρώτη φορά, αποθηκεύει στο Store την οντολογία που υπάρχει στο αρχείο product.rdf.
  - *readModel()* : διαβάζει το μοντέλο από τη βάση.
  - *writeModel()* : γράφει το μοντέλο στη βάση.
  - *storeProduct(int, String)* : με τη βοήθεια της *saveProduct(Model, String, String)* αποθηκεύει ένα νέο προϊόν στην οντολογία. Όπως αναφέρθηκε σε προηγούμενη ενότητα αυτό γίνεται αποθηκεύοντας τη δήλωση:

    (κωδικός_προϊόντος, rdf:type, product:Ονομα_κατηγορίας)

- *deleteProduct(int, String)* : με τη βοήθεια της *removeProduct(Model, String, String)* διαγράφει ένα προϊόν από την οντολογία.
- *retrieveProduct(String)* : με τη βοήθεια της *findProducts(Model, String)* ανακτά από την οντολογία όλους τους κωδικούς προϊόντων που ανήκουν σε μια κατηγορία και τις υποκατηγορίες της. Αυτό επιτυγχάνεται στέλνοντας ένα ερώτημα SPARQL όπως το παρακάτω:

```
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX product:<http://example.org/product#>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
SELECT ?item
WHERE { ?subject  rdfs:subClassOf*  product:category.
            ?item  rdf:type  ?subject }
```

- *retrieveCategoryOfProduct(int)* : με τη βοήθεια της *findCategoryOfProduct(Model, int)* ανακτά από την οντολογία τις κατηγορίες στις οποίες ανήκει ένα προϊόν με βάση τον κωδικό του προϊόντος. Αυτό γίνεται με την αποστολή ενός SPARQL ερωτήματος όπως το :

```
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX product:<http://example.org/product#>
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
SELECT ?object
WHERE { product:productId  rdf:type  ?object }
```

- *printCategoryMenu()* : επιστρέφει ένα αλφαριθμητικό στο οποίο είναι αποθηκευμένες όλες οι κατηγορίες της οντολογίας, με τις υποκατηγορίες καθεμιάς, με σήμανση λίστας σύμφωνα με τις ετικέτες της HTML. Η συνάρτηση το επιτυγχάνει αυτό με τη βοήθεια των συναρτήσεων:
  - *insertInTable(ArrayList<ArrayList<String>>, ArrayList<String>)*
  - *hasSubCategories(String)*
  - *retrieveSubCategories(String)*

Company : Η κλάση Company χρησιμοποιείται για να αποθηκεύει αντικείμενα τύπου Company, δηλαδή αντικείμενα που αποτελούν μια σειρά του πίνακα Company της βάσης δεδομένων.

- Τα πεδία της κλάσης αντιστοιχούν στις στήλες του πίνακα Company (int id, String name).
- Οι μέθοδοι της κλάσης είναι οι κατασκευαστές και οι κλασσικές μέθοδοι get/set για κάθε πεδίο.

DBConnector : Η κλάση DBConnector είναι υπεύθυνη για τη σύνδεση με τη βάση δεδομένων.

- Τα πεδία της κλάσης είναι η *σύνδεση* (Connection conn), η *διεύθυνση* της *βάσης* (String url), το *όνομα* της βάσης (String dbName), το όνομα του *driver* που χρησιμοποιείται για την επικοινωνία με τη βάση (String driver) και το *όνομα* και ο *κωδικός* για τη σύνδεση (String username, String password).
- Οι μέθοδοι της κλάσης αφορούν τη δημιουργία της σύνδεσης (*openConnection()*) και τον έλεγχο της σύνδεσης (*checkConnection()*).

OfferPack : Η κλάση OfferPack χρησιμοποιείται για να αποθηκεύει αντικείμενα τύπου OfferPack, δηλαδή αντικείμενα που αποτελούν μια σειρά του πίνακα OfferPack της βάσης δεδομένων.

- Τα πεδία της κλάσης είναι ο *κωδικός* αριθμός του OfferPack (int id), το *σύνολο* των *κωδικών* των *προϊόντων* που αποτελούν το πακέτο προσφοράς (ArrayList<Integer> productIds), το *σύνολο* των *ονομάτων* των *προϊόντων* που αποτελούν το OfferPack (ArrayList<String> products), το *σύνολο* των *ποσοτήτων* κάθε *προϊόντος* (ArrayList<Integer> items) και ένα *αλφαριθμητικό* που θα εμφανίζει όλα τα προϊόντα της προσφοράς και την ποσότητα του καθενός (String details).
- Οι μέθοδοι της κλάσης είναι ο κατασκευαστής, ο οποίος από το σύνολο των ονομάτων προϊόντων και το σύνολο των ποσοτήτων δημιουργεί και τη μεταβλητή details, και οι κλασσικές μέθοδοι get/set για κάθε πεδίο.

OfferType : Η κλάση OfferType χρησιμοποιείται για να αποθηκεύει αντικείμενα τύπου OfferType, δηλαδή αντικείμενα που αποτελούν μια σειρά του πίνακα OfferType της βάσης δεδομένων.

- Τα πεδία της κλάσης αντιστοιχούν στις στήλες του πίνακα OfferType (int id, String name).
- Οι μέθοδοι της κλάσης είναι ο κατασκευαστής και οι κλασσικές μέθοδοι get/set για κάθε πεδίο.

Price : Η κλάση Price χρησιμοποιείται για να αποθηκεύει αντικείμενα τύπου Price, δηλαδή αντικείμενα που αποτελούν μια σειρά του πίνακα Price της βάσης δεδομένων.

- Τα πεδία της κλάσης αντιστοιχούν στις στήλες του πίνακα Price (int offTypeId, int offerId, int smId, double price, Date prDate, double profit, double difference, int observer_id, int confirmations).
- Οι μέθοδοι της κλάσης είναι οι κατασκευαστές και οι κλασσικές μέθοδοι get/set για κάθε πεδίο.

PricePrint : Η κλάση PricePrint χρησιμοποιείται για την αναπαράσταση μιας Price αλλά χρησιμοποιώντας τα ονόματα κάθε πεδίου αντί για τους κωδικούς αυτών. Δηλαδή αναπαριστά μια Price σε μορφή που μπορεί να γίνει κατανοητή από τον χρήστη και αποτελείται από τα πεδία που ενδιαφέρουν περισσότερο τον χρήστη.

- Τα πεδία της κλάσης είναι το *όνομα* του *τύπου* της *προσφοράς* (String offType), οι *λεπτομέρειες* που αφορούν τα προϊόντα της προσφοράς (String offer), το όνομα του *καταστήματος* (String sm), η *τιμή* (double price) και η *ημερομηνία* της τιμής (Date prDate).
- Οι μέθοδοι της κλάσης είναι ο κατασκευαστής και οι κλασσικές μέθοδοι get/set για κάθε πεδίο.

PriceSuggested : Η κλάση PriceSuggested χρησιμοποιείται για να αποθηκεύει αντικείμενα τύπου PriceSuggested, δηλαδή αντικείμενα που αποτελούν μια σειρά του πίνακα PriceSuggested της βάσης δεδομένων.

- Τα πεδία της κλάσης αντιστοιχούν στις στήλες του πίνακα PriceSuggested (int userId, int offTypeId, int offerId, int smId, boolean checked, Date date_suggested).
- Οι μέθοδοι της κλάσης είναι ο κατασκευαστής και οι κλασσικές μέθοδοι get/set για κάθε πεδίο.

Product : Η κλάση Product χρησιμοποιείται για να αποθηκεύει αντικείμενα τύπου Product, δηλαδή αντικείμενα που αποτελούν μια σειρά του πίνακα Product της βάσης δεδομένων.

- Τα πεδία της κλάσης αντιστοιχούν στις στήλες του πίνακα Product (String name, int items, double weight, int brandId, int unitId, byte packed, String barcode).
- Οι μέθοδοι της κλάσης είναι οι κατασκευαστές και οι κλασσικές μέθοδοι get/set για κάθε πεδίο.

Supermarket : Η κλάση Supermarket χρησιμοποιείται για να αποθηκεύει αντικείμενα τύπου Supermarket, δηλαδή αντικείμενα που αποτελούν μια σειρά του πίνακα Supermarket της βάσης δεδομένων.

- Τα πεδία της κλάσης αντιστοιχούν στις στήλες του πίνακα Supermarket (int id, int companyId, String name, String address, float lat, float lng).
- Οι μέθοδοι της κλάσης είναι οι κατασκευαστές και οι κλασσικές μέθοδοι get/set για κάθε πεδίο.

Unit : Η κλάση Unit χρησιμοποιείται για να αποθηκεύει αντικείμενα τύπου Unit, δηλαδή αντικείμενα που αποτελούν μια σειρά του πίνακα Unit της βάσης δεδομένων.

- Τα πεδία της κλάσης αντιστοιχούν στις στήλες του πίνακα Unit (int id, String name, String abbreviation).
- Οι μέθοδοι της κλάσης είναι οι κατασκευαστές και οι κλασσικές μέθοδοι get/set για κάθε πεδίο

User : Η κλάση User χρησιμοποιείται για να αποθηκεύει αντικείμενα τύπου User, δηλαδή αντικείμενα που αποτελούν μια σειρά του πίνακα User της βάσης δεδομένων.

- Τα πεδία της κλάσης αντιστοιχούν στις στήλες του πίνακα User (int id, String username, String password, String email, String address, Date birthday, byte sex, float lat, float lng, double valid, double novalid).
- Οι μέθοδοι της κλάσης είναι οι κατασκευαστές και οι κλασσικές μέθοδοι get/set για κάθε πεδίο.

MyService : Η κλάση MyService είναι αυτή που παρέχει την κύρια λειτουργικότητα και αυτή που δημοσιεύεται ως Υπηρεσία Ιστού. Όπως ήταν αναμενόμενο είναι η μεγαλύτερη απ' όλες τις κλάσεις (75 συναρτήσεις, περίπου 2300 γραμμές κώδικα). Περιλαμβάνει ιδιωτικές (private) και δημόσιες (public) συναρτήσεις. Οι συναρτήσεις που δηλώνονται ως δημόσιες είναι αυτές που μπορούν να χρησιμοποιηθούν από οποιοδήποτε πρόγραμμα πελάτη θέλει να καλέσει την Υπηρεσία Ιστού. Οι ιδιωτικές συναρτήσεις είναι είτε συναρτήσεις βοηθητικές προς τις δημόσιες είτε συναρτήσεις που θα μπορούσαν να είναι δημόσιες αλλά δεν χρησιμοποιήθηκαν από την διαδικτυακή εφαρμογή που αναπτύχθηκε και για αυτό το λόγο κρατήθηκαν για την ώρα ιδιωτικές. Επίσης κάποιες από τις ιδιωτικές συναρτήσεις είναι αυτές που χρησιμοποιήθηκαν για έλεγχο μέσω του προγράμματος SoapUI.
Πληκτρολογώντας τη διεύθυνση http://localhost:8080/SuperBasket/ και στη συνέχεια απ την επιλογή Services εμφανίζονται τα ονόματα των συναρτήσεων που μπορεί να καλέσει κάποιος (**Εικόνα 25**).
Οι Δημόσιες μέθοδοι της κλάσης MyService:

- *getAllProduct() , getAllUnits(), getAllBrands(), getAllCompanies(), getAllSupermarkets(), getAllOfferTypes(), getAllCategories()* : οι μέθοδοι αυτές όπως υποδεικνύουν και τα ονόματά τους συλλέγουν από τη βάση όλες τις εγγραφές των πινάκων Product, Unit, Brand, Company, Supermarket και OfferType ενώ η *getAllCategories()* ανακτά όλες τις κατηγορίες της οντολογίας, διατηρώντας τις σχέσεις μεταξύ τους, σε μορφή HTML λίστας.
- *getProduct(int), getProductByBarcode(String), getUser(String), getUserById(int), getUnit(String), getProductsOfOffer(int), getBrand(String), getCompany(String), getSupermarket(int), getOfferType(int), getPrice(int, int, int), getPriceSuggested(int, int, int, int)* : οι παραπάνω μέθοδοι όπως μπορεί να γίνει αντιληπτό και από τα ονόματά τους επιστρέφουν ένα αντικείμενο τύπου Product, User, Unit, OfferPack, Brand, Company, Supermarket, OfferType, Price, PriceSuggested με τις τιμές της εγγραφής του πίνακα για την οποία

το/τα id ή το όνομα ισούται με την παράμετρο που δίνεται στη συνάρτηση.

- *getProductOfCategory(String)* : Ανακτά τη λίστα με τα Products που σύμφωνα με την οντολογία ανήκουν σε μια συγκεκριμένη κατηγορία
- *getOffersOfProduct(String)* : Ανακτά τη λίστα με τα OfferPack στα οποία περιέχεται το προϊόν με το συγκεκριμένο barcode
- *getSupermarketsOfCompany(int)* : Ανακτά τη λίστα με τα Supermarket της εταιρείας με το συγκεκριμένο κωδικό αριθμό
- *getListForValidation(String)* : Ανακτά τη λίστα με τις Price από τον πίνακα PriceSuggested για τον χρήστη με το συγκεκριμένο όνομα χρήστη
- *getListForValidationNames(String)* : Ανακτά τη λίστα με τις PricePrint από τον πίνακα PriceSuggested για τον χρήστη με το συγκεκριμένο όνομα
- *storeProductDetails(String, int, double, String, String, String, int, String)* : Αποθηκεύει στη βάση ένα προϊόν με συγκεκριμένο όνομα, αριθμό τεμαχίων, βάρος, μονάδα βάρους, μάρκα, barcode, ιδιότητα που δείχνει αν είναι χύμα ή όχι και στην οντολογία τον κωδικό του στην αντίστοιχη κατηγορία.
- *storeUser(String, String, String, String, Date, int, float, float), storeUnit(String, String), storeBrand(String), storeCompany(String), storeOfferType(String), storeSupermarket(String, int, String, float, float)* : Οι παραπάνω μέθοδοι αποθηκεύουν στον πίνακα User, Unit, Brand, Company, OfferType και Supermarket αντίστοιχα μια εγγραφή όπου τα πεδία της αντιστοιχούν στις παραμέτρους της συνάρτησης. Το πεδίο id των πινάκων ενημερώνεται αυτόματα ενώ πεδία που αφορούν υπολογισμούς και για τα οποία δεν υπάρχει τιμή αρχικά παίρνουν την τιμή NULL.
- *storeOfferPrice(int, String, int, int, int, double, Date, String)* : Αποθηκεύει την τιμή και την προσφορά που την αφορά. Ενημερώνει κατάλληλα τους πίνακες offer και offer_pack για την ύπαρξη της προσφοράς. Στη συνέχεια αν η τιμή υπάρχει την ενημερώνει ενώ αν

δεν υπάρχει την αποθηκεύει. Τέλος αν είναι απαραίτητο ενημερώνει τις διαφορές τιμής για τις άλλες ήδη καταγεγραμμένες τιμές του προϊόντος, τα κέρδη των πακέτων προσφοράς που αφορούν το προϊόν και τον παρατηρητή της τιμής.

- *storeOfferPriceforCompany(int, String, int, int, int, double, Date)* : Καλεί την *storeOfferPrice* για όλα τα καταστήματα που ανήκουν σε μια εταιρεία.
- *storePriceForOfferPack(int, int, int, double, Date)* : Αποθηκεύει τη τιμή για ένα υπάρχον πακέτο προσφοράς. Ενημερώνει κατάλληλα τον πίνακα offer αν είναι απαραίτητο και τις στήλες difference και profit του πίνακα Price.
- *storePriceForOfferPackCompany(int, int, int, double, Date)* : Καλεί την *storePriceForOfferPack* για όλα τα καταστήματα που ανήκουν σε μια εταιρεία.
- *updateUser(String, String, String, Date)* : Ενημερώνει την εγγραφή του χρήστη με ένα συγκεκριμένο όνομα ως προς τα email, διεύθυνση και γενέθλιά του.
- *findProductsMinPrice(String[])* : Επιστρέφει τη χαμηλότερη τιμή κάθε προϊόντος του πίνακα για κάθε διαφορετικό offer_type
- *findPricesOfList(String[], int, String[]) :* Επιστρέφει όλες τις καταγεγραμμένες τιμές όλων των προϊόντων της λίστας για όλα τα σούπερ μάρκετ που δίνονται.
- *findAndStoreBestList(String[], String[], String, String[], String)* : Βρίσκει τον καλύτερο συνδυασμό τιμών για την αγορά μιας λίστας προϊόντων από μια λίστα καταστημάτων, για έναν συγκεκριμένο χρήστη, με μια λίστα ποσοτήτων των προϊόντων και μέγιστο αριθμό συνδυασμών καταστημάτων και την αποθηκεύει για αυτόν τον χρήστη.
- *validateList(String, char[], String[])* : για έναν συγκεκριμένο χρήστη δέχεται τις αξιολογήσεις του σχετικά με τις τιμές των προϊόντων που του προτάθηκαν και μια λίστα καινούριων τιμών για όσες δήλωσε ο χρήστης ότι άλλαξαν. Ενημερώνει τις τιμές και τον παρατηρητή, αλλάζει την αξιοπιστία του χρήστη και του προηγούμενου παρατηρητή

και αλλάζει τις προτεινόμενες τιμές του πίνακα PriceSuggested σε ελεγμένες.

- *hasPriceForValidation(String)* : Ελέγχει αν ο χρήστης με το συγκεκριμένο όνομα έχει προτεινόμενες τιμές που δεν τις έχει αξιολογήσει ακόμα.
- *userIsReliable(String)* : Ελέγχει αν ο χρήστης με το συγκεκριμένο όνομα είναι αξιόπιστος.

Οι υπόλοιπες ιδιωτικές μέθοδοι της Υπηρεσίας Ιστού φαίνονται στον **Πίνακας 3**

Εικόνα 25: Δημοσιευμένες συναρτήσεις της Υπηρεσίας Ιστού

getOfferIdOfProduct(int, boolean)	Επιστρέφει τους κωδικούς των προσφορών στις οποίες εμφανίζεται το προϊόν όταν πωλείται μόνο του (false) ή σε πακέτο προσφοράς (true)
getMax(String, String)	Επιστρέφει το μεγαλύτερο ακέραιο από τη στήλη ενός πίνακα
getPricesOfSoloProducts(ArrayList<Integer>, ArrayList<Integer>)	Επιστρέφει όλες τις τιμές όλων των προϊόντων της λίστας σε όλα τα σούπερ μάρκετ που δίνονται
storeProduct(String, int, double, String, String, String, int)	Αποθηκεύει ένα νέο προϊόν στη βάση
storeCategory(int, String)	Αποθηκεύει ένα προϊόν στην οντολογία στην αντίστοιχη κατηγορία
storeOffer(int, int)	Αποθηκεύει μια νέα εγγραφή στον πίνακα offer της βάσης
storeOfferPack(int, int, int, boolean)	Αποθηκεύει μια νέα εγγραφή στον πίνακα OfferPack της βάσης
storeOfferForProduct(int, int, int, int, boolean)	Εισάγει μια προσφορά στους πίνακες offer και offerpack
storePrice(int, int, int, double, Date)	Εισάγει μια τιμή στη βάση
storeBestList(int, ArrayList<Price>)	Αποθηκεύει μια λίστα από τιμές που προτάθηκαν στον χρήστη στον πίνακα PriceSuggested
storePriceSuggested(int, int, int, int, boolean, Date)	Αποθηκεύει μια νέα εγγραφή στον πίνακα PriceSuggested της βάσης
updateValidation(int, double)	Ενημερώνει την αξιοπιστία του χρήστη
updatePrice(int, int, int, double, Date)	Ενημερώνει την τιμή και την ημερομηνία μιας εγγραφής του πίνακα Price
updateObserver(int, int, int, int)	Ενημερώνει τον παρατηρητή μιας τιμής
updateProfit(int, int, int, double)	Ενημερώνει το κέρδος της τιμής ενός

	πακέτου προσφοράς
updateDifference(int, int, int, double)	Ενημερώνει τη διαφορά μιας τιμής από τη μικρότερη καταγεγραμμένη τιμή της
updatePriceSuggested(int, int, int, int, boolean, Date)	Ενημερώνει μια εγγραφή PriceSuggested ως προς τα πεδία checked, date_suggested
offerExistsInSupermarket(int, int)	Ελέγχει αν μια συγκεκριμένη προσφορά υπάρχει σε συγκεκριμένο σούπερ μάρκετ
priceExists(int, int, int)	Ελέγχει αν υπάρχει τιμή για το συγκεκριμένο συνδυασμό offer Type, offer, supermarket
priceToPrint(ArrayList<Price>)	Μετατρέπει μια λίστα από Prices σε λίστα από PricePrint
findMinOfferPriceOfType(int, int)	Επιστρέφει τη μικρότερη τιμή μιας προσφοράς για ένα τύπο προσφοράς
findPricesOfOffer(int, ArrayList<Integer>)	Επιστρέφει τις τιμές μιας προσφοράς στη λίστα των σουπερ μάρκετ που δίνονται
findCommonSmForOffers(ArrayList<Integer>)	Βρίσκει τα σούπερ μάρκετ τα οποία διαθέτουν όλες τις προσφορές της λίστας
findBestList(String[], String[], String[], String)	Βρίσκει τον καλύτερο συνδυασμό τιμών και σουπερ μάρκετ δεχόμενο μια λίστα προϊόντων, μια λίστα σουπερμάρκετ, μια λίστα ποσοτήτων και το μέγιστο αριθμό συνδυασμών
arrayStringToInt(String[])	Μετατρέπει έναν πίνακα από αλφαριθμητικά σε λίστα ακεραίων
listContainsOfferPack(ArrayList<Integer>, int)	Ελέγχει αν όλα τα προϊόντα ενός πακέτου προσφοράς περιέχονται στη

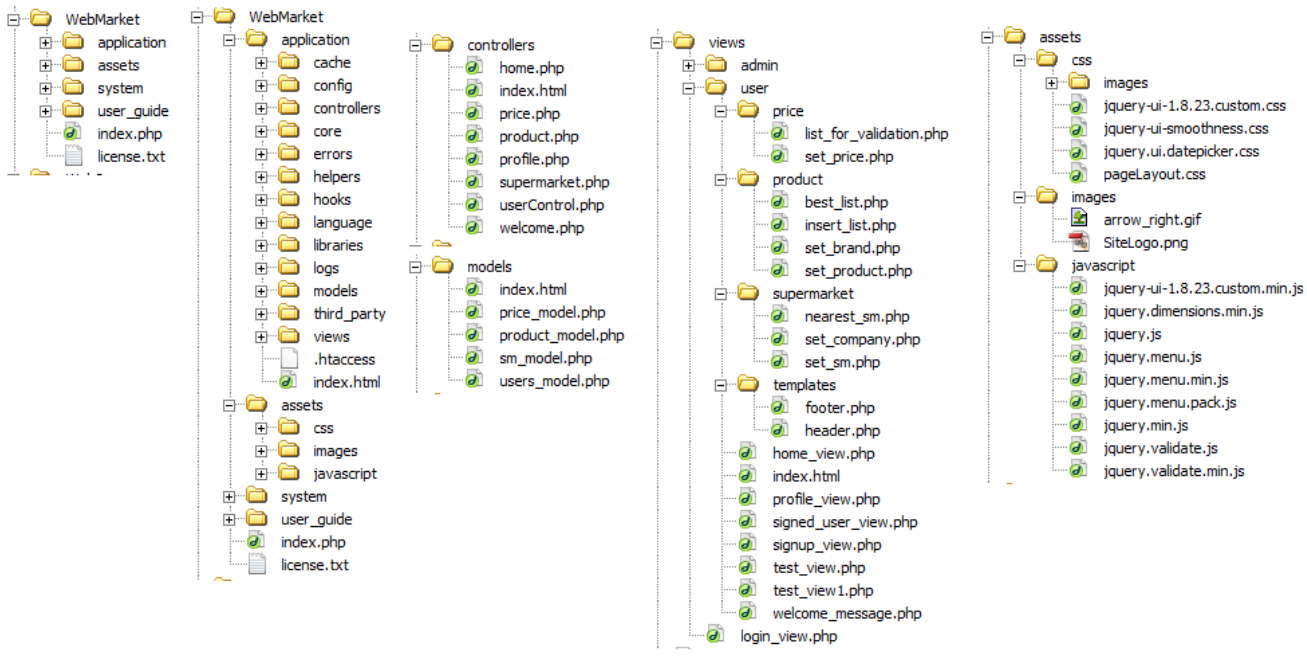
	λίστα των προϊόντων
arraysOfProductOffers(ArrayList<Integer>)	Δέχεται μια λίστα από προϊόντα και επιστρέφει μια λίστα με τις προσφορές των οποίων τα προϊόντα περιέχονται όλα στη λίστα και μια λίστα με προσφορές που έχουν τουλάχιστον ένα προϊόν στη λίστα
soloOffersOfPack(int)	Επιστρέφει τη λίστα με τις προσφορές των μεμονωμένων προϊόντων που περιέχονται σε ένα πακέτο προσφοράς
profitOfPack(int, int, int)	Υπολογίζει το κέρδος που προσφέρει η αγορά ενός πακέτου προσφοράς έναντι της αγοράς των μεμονωμένων προϊόντων της για συγκεκριμένα offer type, offer, sm
bestOfferPackOfAll()	Επιστρέφει το πακέτο προσφοράς με το μεγαλύτερο κέρδος
minPricesOfSm(ArrayList<Integer>, ArrayList<Integer>)	Επιστρέφει για κάθε προϊόν της λίστας τη μικρότερη τιμή του στα σουπερμάρκετ που δίνονται
checkFeedback(String)	Βρίσκει πόσες φορές ο χρήστης δέχτηκε προτάσεις από το σύστημα αλλά δεν της αξιολόγησε μετά

Πίνακας 3 : Οι ιδιωτικές συναρτήσεις της Υπηρεσίας Ιστού

5.3.2 Διαδικτυακή εφαρμογή

Η διαδικτυακή εφαρμογή υλοποιήθηκε σε γλώσσα PHP, με τη χρήση του framework CodeIgniter. Επιπλέον χρησιμοποιήθηκε η γλώσσα Javascript, η βιβλιοθήκη JQuery και η τεχνολογία AJAX για την εκτέλεση σεναρίων εντολών από την πλευρά του πελάτη – προγράμματος περιήγησης και οι γλώσσες HTML και CSS για τη δημιουργία και εμφάνιση των ιστοσελίδων. Η

διαδικτυακή εφαρμογή αποτελεί τον πελάτη της Υπηρεσίας Ιστού, δηλαδή καλεί τις λειτουργίες της Υπηρεσίας Ιστού και τις προσφέρει στον δικό της πελάτη, το πρόγραμμα περιήγησης του χρήστη, μέσα από μια Γραφική Διασύνδεση Χρήστη. Το CodeIgniter προσφέρει μια δομή στην οποία χτίζεται η εφαρμογή. Ο κατάλογος αρχείων της εφαρμογής φαίνεται στην **Εικόνα 26**. Στον φάκελο «application» υπάρχουν οι φάκελοι «models», «views» και «controllers» οι οποίοι περιέχουν τα κύρια αρχεία της εφαρμογής, ενώ στο φάκελο «assets» υπάρχουν τα αρχεία css, οι εικόνες και αρχεία της βιβλιοθήκης JQuery. Οι «controllers» είναι τα αρχεία Ελεγκτές που αναλαμβάνουν τη λειτουργία της διαδικτυακής εφαρμογής και είναι αυτά που καλεί ο χρήστης. Τα «models» είναι τα αρχεία που επικοινωνούν με την Υπηρεσία Ιστού και παρέχουν στους Ελεγκτές τα δεδομένα τα οποία αυτοί «περνάνε» με τη σειρά τους στις «views» για να δημιουργήσουν τις σελίδες που θα εμφανίσουν στο χρήστη. Επομένως για κάθε σελίδα που εμφανίζεται στο χρήστη χρησιμοποιείται μια τριάδα Model – View – Controller.



Εικόνα 26: Ο κατάλογος αρχείων της διαδικτυακής εφαρμογής

5.3.2.1 Περιγραφή των αρχείων

Οι Ελεγκτές είναι κλάσεις που κληρονομούν την κλάση CI_Controller του CodeIgniter. Αποτελούνται από έναν κατασκευαστή, στον οποίο καλούνται τα Μοντέλα ή/και βοηθητικές βιβλιοθήκες του framework που θα χρησιμοποιηθούν, και από δημόσιες συναρτήσεις που μπορούν να κληθούν από το πρόγραμμα περιήγησης του πελάτη με τη μορφή «όνομαΕλεγκτή/ όνομαΣυνάρτησης». Όταν καλείται το όνομα του Ελεγκτή χωρίς κάποια συνάρτηση τότε καλείται η συνάρτηση με το όνομα «index» που αποτελεί την default συνάρτηση των Ελεγκτών. Τα Μοντέλα είναι κι αυτά κλάσεις που κληρονομούν την κλάση CI_Model του CodeIgniter. Όπως οι ελεγκτές αποτελούνται κι αυτά από έναν κατασκευαστή που συνήθως είναι κενός και από δημόσιες συναρτήσεις που καλούνται από τους Ελεγκτές. Τέλος οι Όψεις είναι αρχεία που περιέχουν HTML, ενσωματωμένη JavaScript αλλά και εντολές PHP.

Η περιγραφή των αρχείων θα γίνει βάσει των Ελεγκτών που καλούνται.

Home : είναι ο ελεγκτής που καλείται αρχικά,

- εμφανίζει την «home_view» όταν δεν έχει δημιουργηθεί κάποιο session, δηλαδή όταν καλείται για πρώτη φορά. Η «home_view» είναι η σελίδα που περιγράφει την εφαρμογή και προτρέπει το χρήστη να συνδεθεί στο σύστημα ή να εγγραφεί.
- εμφανίζει την «signed_user_view» όταν έχει δημιουργηθεί ένα session, δηλαδή όταν έχει συνδεθεί ο χρήστης. Η «signed_user_view» είναι η κεντρική σελίδα ενός συνδεδεμένου χρήστη από την οποία μπορεί να ενημερωθεί για τη λειτουργικότητα των υπόλοιπων σελίδων.

UserControl : είναι ο ελεγκτής που χειρίζεται την εγγραφή, τη σύνδεση και την αποσύνδεση ενός χρήστη από το σύστημα,

- ο κατασκευαστής του «φορτώνει» το Μοντέλο «users_model» και τη βιβλιοθήκη «googlemaps» η οποία περιέχει συναρτήσεις για την αρχικοποίηση και την εμφάνιση χαρτών της Google,

- εμφανίζει τη σελίδα εγγραφής χρήστη, «signup_view», στην οποία στέλνει και έναν χάρτη με τη βοήθεια του οποίο θα προσθέσει ο χρήστης τη διεύθυνσή του,
- ελέγχει τα στοιχεία που συμπλήρωσε ο χρήστης στη φόρμα εγγραφής. Με τη βοήθεια της συνάρτησης users_model->get_user ελέγχει αν υπάρχει ήδη αποθηκευμένος κάποιος χρήστης με το ίδιο όνομα και αν δεν υπάρχει καλεί τη συνάρτηση users_model->set_user η οποία αποθηκεύει όλα τα στοιχεία της φόρμας στη βάση δεδομένων,
- εμφανίζει τη σελίδα σύνδεσης χρήστη «login_view»,
- ελέγχει τα στοιχεία που συμπλήρωσε ο χρήστης στη φόρμα σύνδεσης. Με τη βοήθεια της συνάρτησης users_model->get_user ελέγχει αν το όνομα χρήστη και ο κωδικός συμφωνούν με αυτά που είναι αποθηκευμένα στη βάση. Αν το όνομα χρήστη και ο κωδικός είναι σωστά τότε δημιουργεί το session στο οποίο αποθηκεύει και κάποια στοιχεία του χρήστη και στη συνέχεια καλεί τον Ελεγκτή «home»,
- καταστρέφει το session όταν ο χρήστης επιλέξει να αποσυνδεθεί και καλεί τον Ελεγκτή «home».

price : είναι ο Ελεγκτής που χειρίζεται την εισαγωγή τιμής προϊόντος, την επικύρωση των τιμών που προτάθηκαν στον χρήστη την προηγούμενη φορά και άλλες λειτουργίες που χρειάζονται οι Όψεις που καλεί,

- ο κατασκευαστής του φορτώνει τα μοντέλα «price_model», «sm_model», «product_model» και «users_model»,
- εμφανίζει τη σελίδα εισαγωγής τιμής προϊόντος «set_price» στην οποία παρέχει με τη βοήθεια της sm_model->getAllCompanies όλες τις εταιρείες που είναι αποθηκευμένες στη βάση και με τη βοήθεια της price_model->getAllOfferTypes όλους τους τύπους προσφορών που υπάρχουν,
- ελέγχει τα στοιχεία της φόρμας εισαγωγής τιμής προϊόντος, «set_price» και αν όλα είναι εντάξει αποθηκεύει την τιμή στη βάση με τη βοήθεια της συνάρτησης price_model->store_price,

- επιστρέφει στην Όψη «set_price», μετά από μια κλήση AJAX, τα σουπερμάρκετ της συγκεκριμένης εταιρείας που επέλεξε ο χρήστης καλώντας τη συνάρτηση sm_model->getAllSupermarketsOfCompany,
- επιστρέφει στη Όψη «set_price», μετά από μια κλήση AJAX, το όνομα του προϊόντος που αντιστοιχεί στο barcode που πληκτρολόγησε ο χρήστης καλώντας τη συνάρτηση product_model->get_product,
- εμφανίζει την Όψη εισαγωγής νέου τύπου προσφοράς «set_offerType» και μετά τον έλεγχο των στοιχείων που εισήχθησαν την αποθηκεύει στη βάση χρησιμοποιώντας τη συνάρτηση price_model->set_offerType, (αυτή η λειτουργία επιτρέπεται μόνο στο διαχειριστή)
- εμφανίζει την Όψη μαζικής εισαγωγής τιμής για όλα τα υποκαταστήματα μιας εταιρείας «set_price_company» και μετά τον έλεγχο των στοιχείων που εισήχθησαν καλεί την price_model-> store_price_for_company η οποία εισάγει την τιμή για όλα τα υποκαταστήματα στη βάση, (αυτή η λειτουργία επιτρέπεται μόνο στο διαχειριστή)
- εμφανίζει την Όψη επικύρωσης της προηγούμενης λίστας «list_for_validation» στην οποία παρέχει με τη βοήθεια της συνάρτησης users_model->getListForValidation τις προηγούμενες τιμές που προτάθηκαν στον χρήστη αλλά δεν τις επικύρωσε ακόμα,
- ελέγχει τα στοιχεία που εισήγαγε ο χρήστης στη φόρμα επικύρωσης της προηγούμενης λίστας και αν δεν υπάρχει κάποιο λάθος αποθηκεύει τα στοιχεία στη βάση με τη βοήθεια της συνάρτησης price_model->storeValidation.

Product : είναι ο Ελεγκτής που χειρίζεται την εισαγωγή προϊόντος, την εισαγωγή λίστας προϊόντων για την εύρεση καλύτερου συνδυασμού, την εμφάνιση του καλύτερου συνδυασμού και άλλες λειτουργίες που χρησιμοποιούν οι όψεις που καλεί,

- ο κατασκευαστής του φορτώνει τα μοντέλα «product_model», «sm_model», και «users_model» και τις βιβλιοθήκες «table» και «googlemaps»,

- εμφανίζει την Όψη εισαγωγής προϊόντος «set_product», στην οποία παρέχει όλες τις μονάδες μέτρησης βάρους από τη συνάρτηση product_model->getAllUnits, όλες τις μάρκες προϊόντος που υπάρχουν στη βάση από τη συνάρτηση product_model->getAllBrands και όλες τις κατηγορίες προϊόντων που υπάρχουν στη οντολογία από τη συνάρτηση product_model->getAllCategories,
- ελέγχει τα στοιχεία που εισήγαγε ο χρήστης και αποθηκεύει το προϊόν στη βάση χρησιμοποιώντας τη συνάρτηση product_model-> set_product,
- εμφανίζει την Όψη εισαγωγής νέας μονάδας μέτρησης βάρους «set_unit» και μετά τον έλεγχο των στοιχείων που εισήχθησαν την αποθηκεύει στη βάση χρησιμοποιώντας τη συνάρτηση product_model->set_unit, (αυτή η λειτουργία επιτρέπεται μόνο στο διαχειριστή)
- εμφανίζει την Όψη εισαγωγής νέας επωνυμίας προϊόντος «set_brand» και μετά τον έλεγχο των στοιχείων που εισήχθησαν την αποθηκεύει στη βάση χρησιμοποιώντας τη συνάρτηση product_model-> set_brand,
- εμφανίζει την Όψη εισαγωγής λίστας προϊόντων για αγορά «insert_list», στην οποία παρέχει τις κατηγορίες των προϊόντων της οντολογίας από τη συνάρτηση product_model->getAllCategories και το χάρτη που δημιουργεί με τη βοήθεια της βιβλιοθήκης googlemaps και της συνάρτησης sm_model->getSupermarketMarkers η οποία δίνει τις συντεταγμένες των καταστημάτων που θα απεικονιστούν στο χάρτη,
- επιστρέφει στην Όψη «insert_list», μετά από μια κλήση AJAX, τα προϊόντα που ανήκουν σε μια συγκεκριμένη κατηγορία χρησιμοποιώντας τη συνάρτηση product_model-> getProductsOfCategory,
- λαμβάνει τα στοιχεία που εισήγαγε ο χρήστης στην Όψη εισαγωγής λίστας προϊόντων προς αγορά και με τη βοήθεια της συνάρτησης product_model->findAndStoreBestList βρίσκει και αποθηκεύει στη

βάση την καλύτερη πρόταση τιμών αγοράς. Στη συνέχεια καλεί τη δική του συνάρτηση createTablesOfOffers η οποία θα διαμορφώσει τη λίστα προτάσεων σε πίνακες (ένας για κάθε κατάστημα) και θα επιστρέψει τους πίνακες. Τέλος καλεί την Όψη καλύτερης λίστας «best_list» στην οποία παρέχει τους πίνακες και το συνολικό τελικό ποσό της λίστας.

Supermarket : είναι ο Ελεγκτής που χειρίζεται την εισαγωγή σούπερ μάρκετ και εταιρειών και την εύρεση των κοντινότερων σούπερ μάρκετ,

- ο κατασκευαστής του φορτώνει τα Μοντέλα «sm_model», και «users_model» και τη βιβλιοθήκη «googlemaps»,
- εμφανίζει την Όψη εισαγωγής σούπερ μάρκετ στην οποία παρέχει όλες τις επωνυμίες σούπερ μάρκετ με τη χρήση της συνάρτησης sm_model->getAllCompanies και έναν χάρτη στον οποίο έχει τοποθετήσει όλα τα σούπερ μάρκετ με τη βοήθεια της συνάρτησης sm_model->getSupermarketMarkers και τον έχει δημιουργήσει με τη βοήθεια της βιβλιοθήκης googlemaps,
- ελέγχει τα στοιχεία που εισήγαγε ο χρήστης και αν είναι εντάξει αποθηκεύει το σούπερ μάρκετ μέσω της συνάρτησης sm_model-> set_sm,
- εμφανίζει την Όψη εισαγωγής νέας επωνυμίας σούπερ μάρκετ «set_company» και μετά τον έλεγχο των στοιχείων που εισήχθησαν την αποθηκεύει στη βάση χρησιμοποιώντας τη συνάρτηση product_model-> set_company,
- εμφανίζει την Όψη εύρεσης κοντινότερων σουπερ μάρκετ «nearest_sm» και της παρέχει έναν χάρτη στον οποίο έχει τοποθετήσει όλα τα σούπερ μάρκετ με τη βοήθεια της συνάρτησης sm_model->getSupermarketMarkers και τον έχει δημιουργήσει με τη βοήθεια της βιβλιοθήκης googlemaps.

Profile : είναι ο Ελεγκτής που εμφανίζει και ενημερώνει το προφίλ του χρήστη,

- ο κατασκευαστής του φορτώνει το Μοντέλο «users_model»,

- εμφανίζει την Όψη εμφάνισης προφίλ του χρήστη «profile_view» στην οποία παρέχει τα στοιχεία του χρήστη που ανακτά από τη συνάρτηση users_model->get_user,
- ελέγχει τα στοιχεία που έχει εισάγει ο χρήστης και ενημερώνει τον πίνακα της βάσης με τη βοήθεια της συνάρτησης users_model-> update_user.

Τα Μοντέλα της εφαρμογής καλούν τη Υπηρεσία Ιστού για να λάβουν τα δεδομένα που χρειάζεται η εφαρμογή. Κάθε συνάρτηση μοντέλου που αναφέρθηκε παραπάνω καλεί την αντίστοιχη μέθοδο της Υπηρεσίας Ιστού. Ο τρόπος που γίνεται η κλήση μιας μεθόδου είναι ο εξής :

```
$client= new
SoapClient("http://localhost:8080/SuperBasket/services/MyService?wsdl");
$result = $client->όνομα_μεθόδου($πίνακας_παραμέτρων)->return;
```

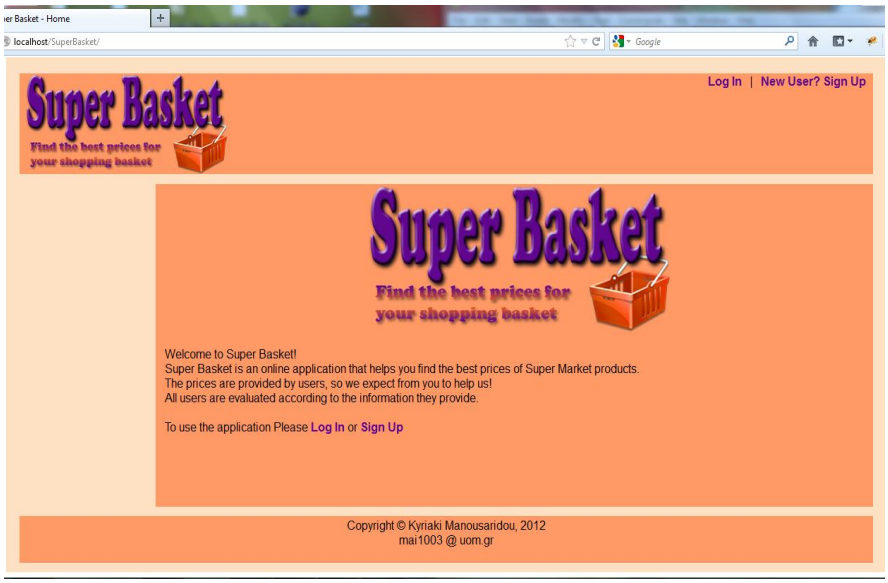
ΚΕΦΑΛΑΙΟ 6

6 Παρουσίαση εφαρμογής

Στην ενότητα αυτή θα παρουσιαστεί η λειτουργία της Διαδικτυακής Εφαρμογής όπως θα εκτελούνταν από έναν καινούριο χρήστη. Επίσης θα περιγραφούν και οι επιπλέον λειτουργίες που μπορεί να εκτελέσει ο διαχειριστής του συστήματος. Προϋπόθεση της λειτουργίας της Διαδικτυακής Εφαρμογής είναι να εκτελείται η Υπηρεσία Ιστού, να είναι δημοσιευμένη και προσπελάσιμη.

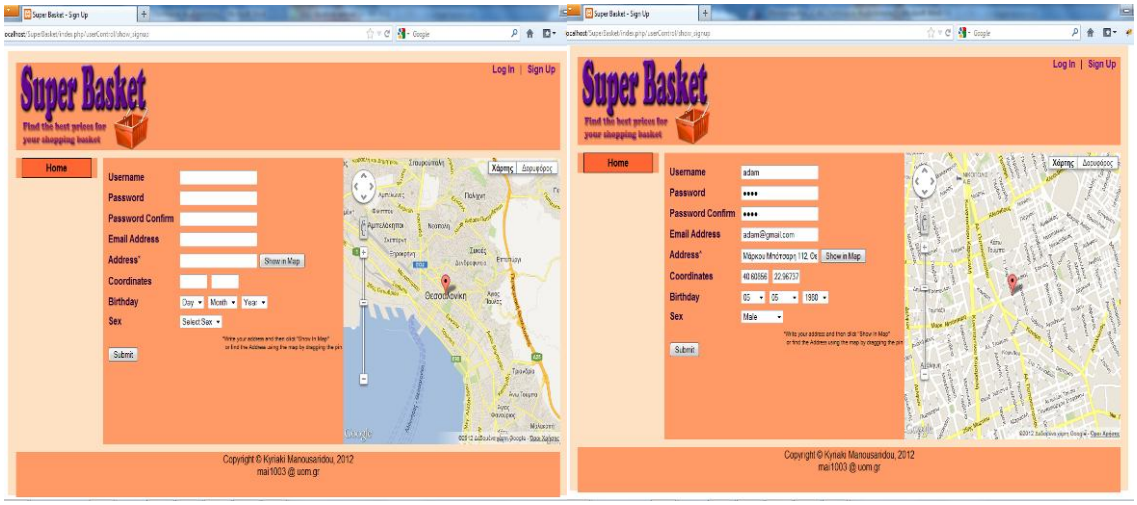
6.1 Χρήστης

Η πρώτη σελίδα που συναντά ένας χρήστης μόλις πληκτρολογήσει τη διεύθυνση της εφαρμογής (http://localhost/SuperBasket/) είναι η **Εικόνα 27.** Στη σελίδα αυτή υπάρχει το μήνυμα καλωσορίσματος, μια περιγραφή της εφαρμογής και η προτροπή προς το χρήστη να συνδεθεί στην εφαρμογή ή να εγγραφεί αν δεν είναι ήδη χρήστης της εφαρμογής. Στην κεφαλίδα αριστερά υπάρχει το λογότυπο της εφαρμογής που αποτελεί σύνδεσμο προς τη κεντρική σελίδα και δεξιά σύνδεσμοι προς τη σύνδεση ή εγγραφή νέου χρήστη.



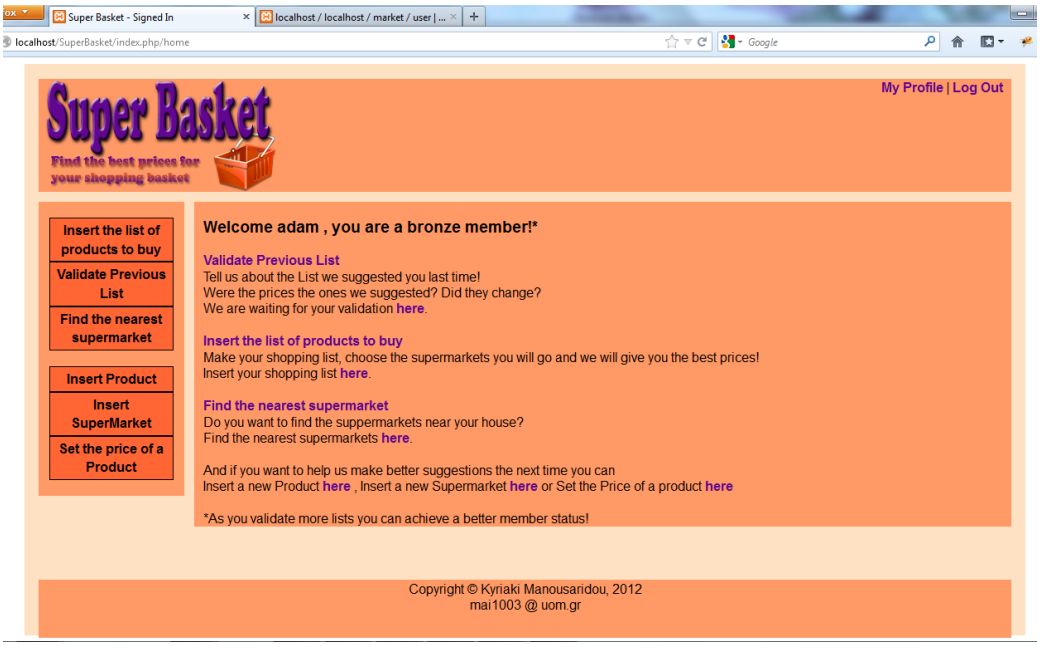
Εικόνα 27: Αρχική σελίδα του Super Basket

Επιλέγοντας «Sign Up» ο χρήστης θα οδηγηθεί στην **Εικόνα 28** στην οποία υπάρχει μια φόρμα για να εισάγει τα στοιχεία του (όνομα χρήστη, κωδικό, ηλεκτρονική διεύθυνση, διεύθυνση κατοικίας, ημερομηνία γέννησης και φύλο). Για την εισαγωγή της διεύθυνσης κατοικίας του υπάρχει ένας χάρτης στον οποίο μπορεί να σύρει την πινέζα στη διεύθυνση κατοικίας του και έτσι να συμπληρωθούν αυτόματα τα πεδία της διεύθυνσης και των συντεταγμένων της. Μόλις ολοκληρώσει την εισαγωγή των στοιχείων του μπορεί να επιλέξει το κουμπί «Submit» για την αποθήκευσή τους. Ο έλεγχος εγκυρότητας και πληρότητας των στοιχείων της φόρμας γίνεται με τη χρήση της JQuery η οποία εμφανίζει μηνύματα λάθους και δεν επιτρέπει την υποβολή της φόρμας αν δεν διορθωθούν τυχόν λάθη ή παραλείψεις.



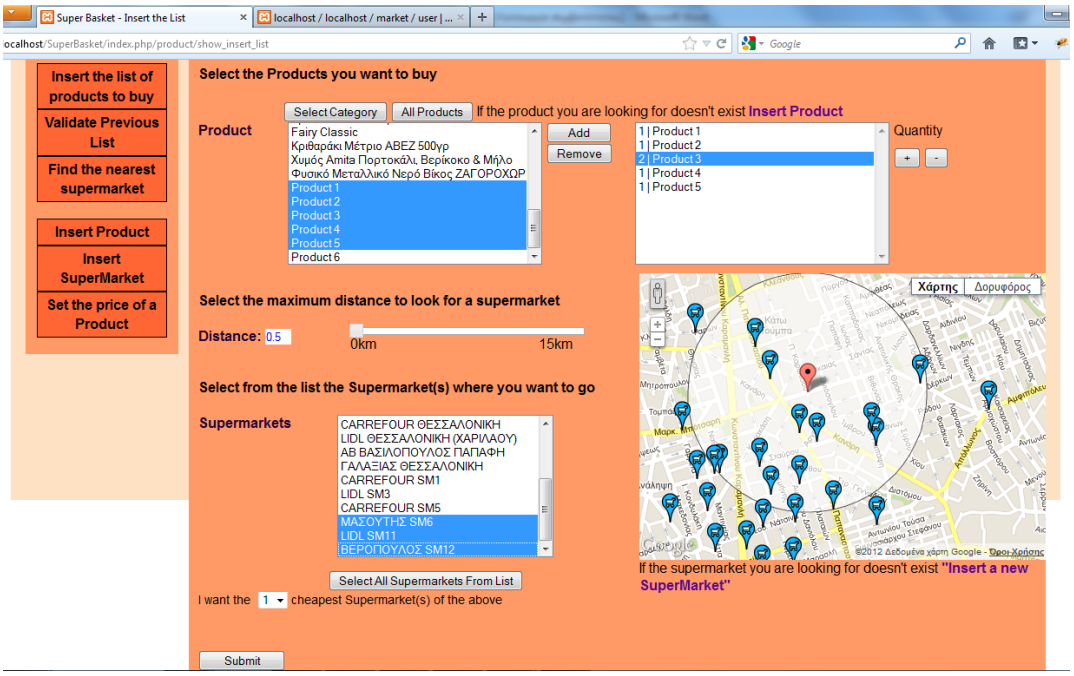
Εικόνα 28: Εγγραφή χρήστη

Μετά την επιτυχή υποβολή των στοιχείων του ο χρήστης οδηγείται στη σελίδα της **Εικόνα 29** που αποτελεί την αρχική σελίδα των εγγεγραμμένων χρηστών. Στη σελίδα αυτή μπορεί να ενημερωθεί για το «status» του ως μέλος, για τις λειτουργίες που μπορεί να εκτελέσει και να μεταφερθεί σε αυτές είτε μέσω των συνδέσμων στο κεντρικό περιεχόμενο της σελίδας είτε μέσω του μενού πλοήγησης στα αριστερά της σελίδας.



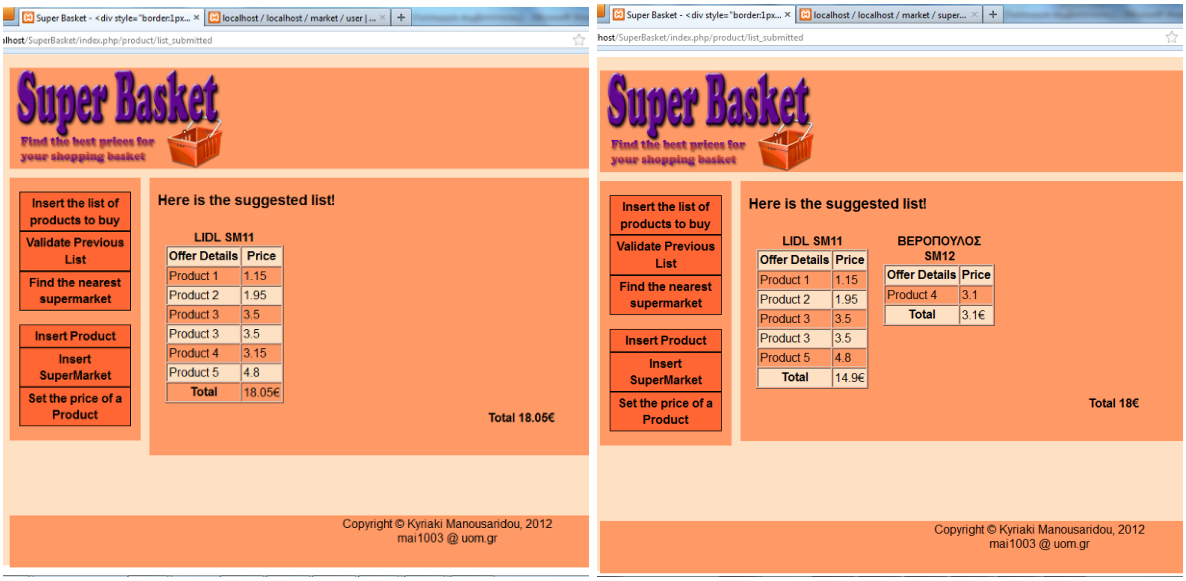
Εικόνα 29: Αρχική οθόνη εγγεγραμμένων χρηστών

Όταν ο χρήστης επιλέξει να εισάγει μια λίστα αγορών για να βρει που τον συμφέρει καλύτερα να ψωνίσει οδηγείται στην **Εικόνα 30**. Στη σελίδα αυτή μπορεί να επιλέξει μια κατηγορία προϊόντος για να εμφανιστούν τα προϊόντα της ή το κουμπί «All Products» το οποίο θα εμφανίσει όλα τα προϊόντα που είναι αποθηκευμένα στο σύστημα. Στη συνέχεια επιλέγει τα προϊόντα που επιθυμεί να αγοράσει και τα προσθέτει στη λίστα με τη βοήθεια των κουμπιών «Add», «Remove». Επιλέγοντας κάποιο προϊόν μπορεί να αυξήσει την ποσότητα του από τα κουμπιά «+» και «-» στα δεξιά της λίστας. Για την επιλογή καταστημάτων μπορεί να επιλέξει την ακτίνα μέσα στην οποία είναι διατεθειμένος να πάει και έτσι εμφανίζονται τα καταστήματα που περιέχονται στην ακτίνα αυτή. Από τα καταστήματα αυτά μπορεί να επιλέξει τα υποψήφια καταστήματα που επιθυμεί και στη συνέχεια να διαλέξει τον αριθμό συνδυασμών που θέλει να κάνει το σύστημα για να βρει τη φτηνότερη λίστα. Ο μέγιστος αριθμός συνδυασμών εξαρτάται από το «status» του χρήστη. Για έναν «bronze»χρήστη είναι 1, για έναν «silver» είναι 2, για έναν «gold» 3 και για έναν «platinum» 4 σούπερ μάρκετ.



Εικόνα 30: Εισαγωγή λίστας αγορών

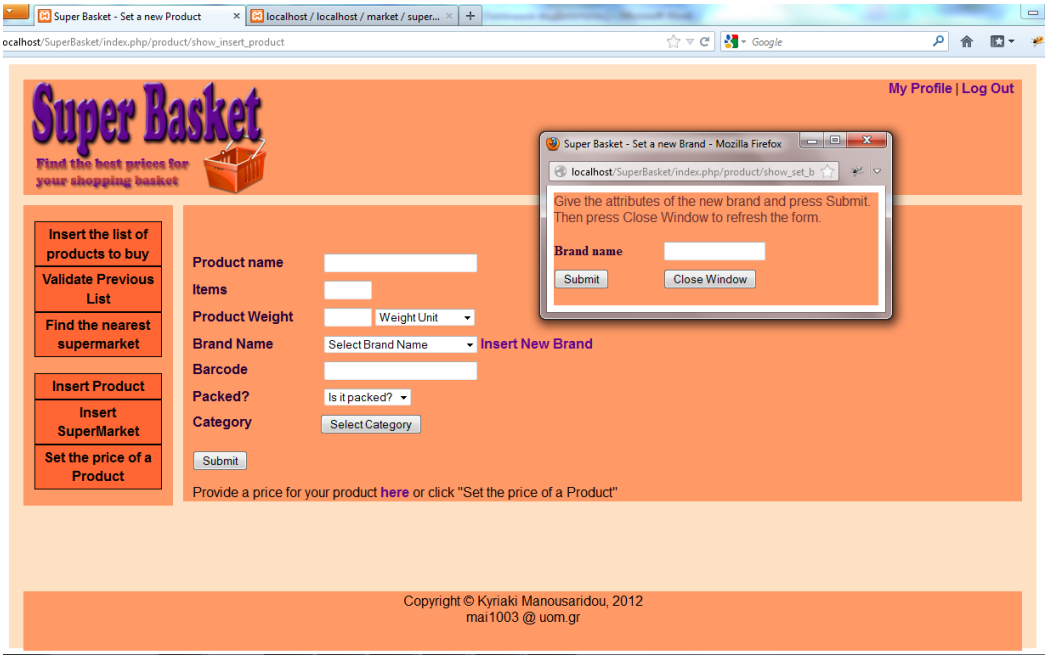
Μετά την υποβολή των επιλογών του χρήστη εμφανίζεται η **Εικόνα 31** στην οποία υπάρχει η φτηνότερη λίστα σύμφωνα με τις τιμές που υπάρχουν καταγεγραμμένες στο σύστημα.



Εικόνα 31: Φτηνότερη λίστα για έναν bronze κι έναν silver χρήστη

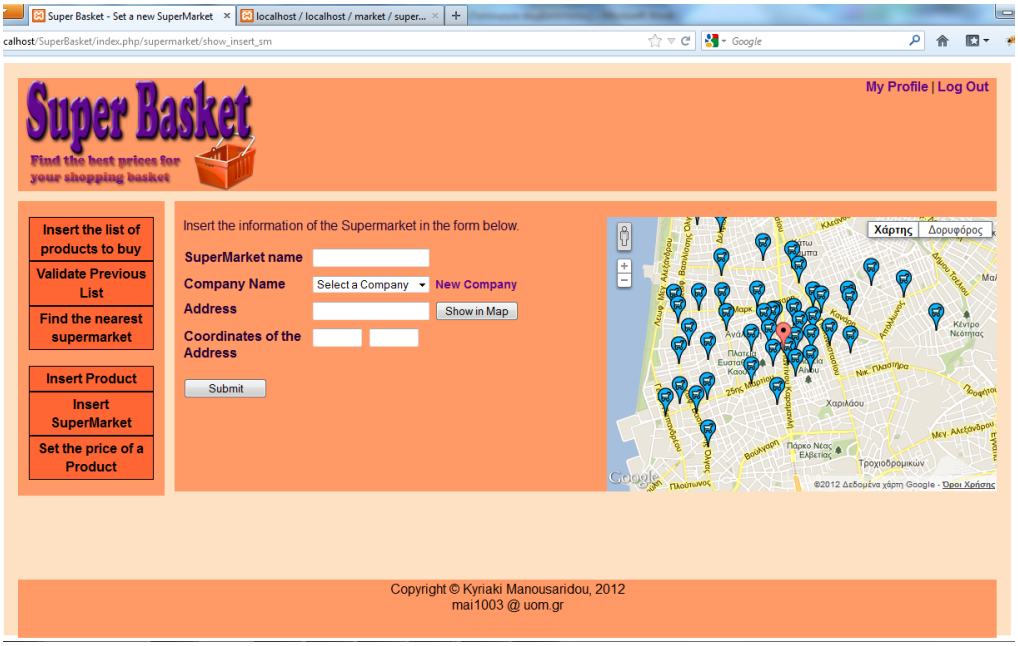
Αν ο χρήστης επιλέξει να εισάγει στη βάση δεδομένων ένα καινούριο προϊόν θα εμφανιστεί η σελίδα της **Εικόνα 32** με τη φόρμα εισαγωγής προϊόντος. Τα στοιχεία που καλείται να προσθέσει είναι το όνομα του προϊόντος, τα τεμάχια που το αποτελούν, το βάρος του και η μονάδα μέτρησης βάρους, η επωνυμία του προϊόντος, το barcode, αν πωλείται χύμα ή συσκευασμένο και

η κατηγορία στην οποία ανήκει. Αν επιλέξει να εισαγάγει μια καινούρια επωνυμία προϊόντος εμφανίζεται το μικρό παράθυρο της εικόνας για την εισαγωγή επωνυμίας στη βάση δεδομένων.



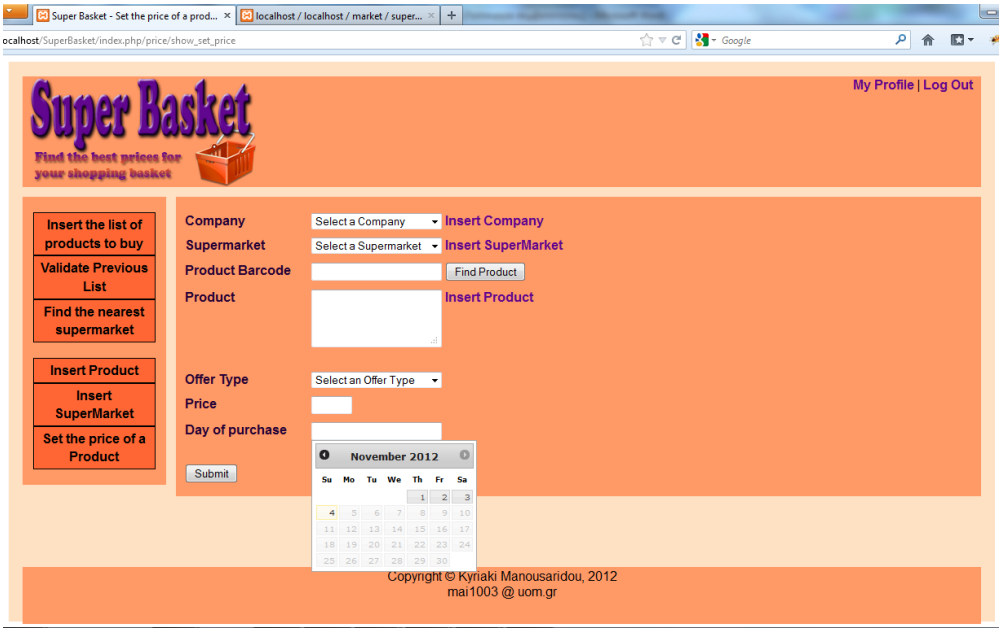
Εικόνα 32: Εισαγωγή Προϊόντος

Αν επιλέξει ο χρήστης να εισάγει ένα νέο κατάστημα στο σύστημα τότε θα εμφανιστεί η σελίδα της **Εικόνα 33**. Στην εικόνα αυτή μπορεί να εισάγει το όνομα του σούπερ μάρκετ, να επιλέξει την επωνυμία στην οποία ανήκει ή να προσθέσει μια άλλη επωνυμία με τη βοήθεια του συνδέσμου «New Company» και να επιλέξει από το χάρτη τη διεύθυνση του καταστήματος. Μπορεί επίσης να δει τα καταστήματα που είναι ήδη καταχωρημένα στη βάση για να μη καταχωρήσει ξανά κάποιο που ήδη υπάρχει.



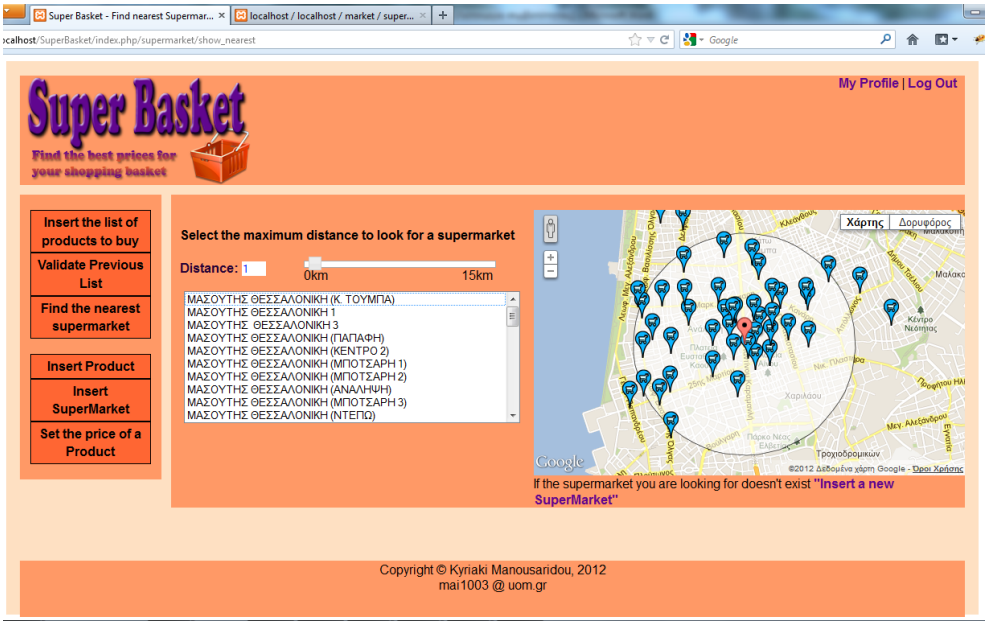
Εικόνα 33: Εισαγωγή σούπερ μάρκετ

Αν επιλέξει ο χρήστης την εισαγωγή τιμής προϊόντος εμφανίζεται η σελίδα της **Εικόνα 34** στην οποία θα επιλέξει την επωνυμία του καταστήματος στο οποίο κατέγραψε την τιμή, το κατάστημα, το barcode του προϊόντος (από το οποίο το σύστημα θα βρει το όνομα του προϊόντος), τον τύπο της προσφοράς στην οποία απευθύνεται η τιμή αυτή, την ίδια την τιμή και την ημερομηνία αγοράς. Αν δεν υπάρχει καταχωρημένη η επωνυμία του σούπερ μάρκετ, το σούπερ μάρκετ ή το προϊόν μπορεί μέσω του αντίστοιχου συνδέσμου να τα εισάγει.



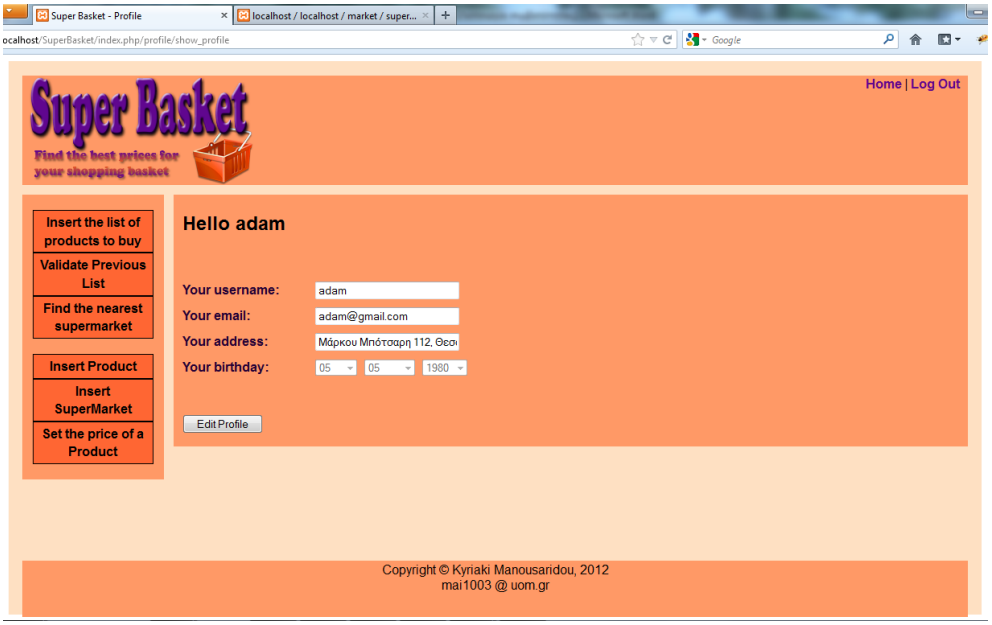
Εικόνα 34: Εισαγωγή τιμής προϊόντος

Αν ο χρήστης θέλει μόνο να δει ποια είναι τα σούπερ μάρκετ που βρίσκονται κοντά στο σπίτι του τότε θα εμφανιστεί η **Εικόνα 35** στην οποία μπορεί να επιλέξει την ακτίνα στην οποία θέλει να ανήκουν τα καταστήματα που αναζητά. Επιλέγοντας ακτίνα μπορεί να διαβάσει στη λίστα ποια είναι τα καταστήματα εντός της ακτίνας.



Εικόνα 35: Εύρεση κοντινότερων σούπερ μάρκετ

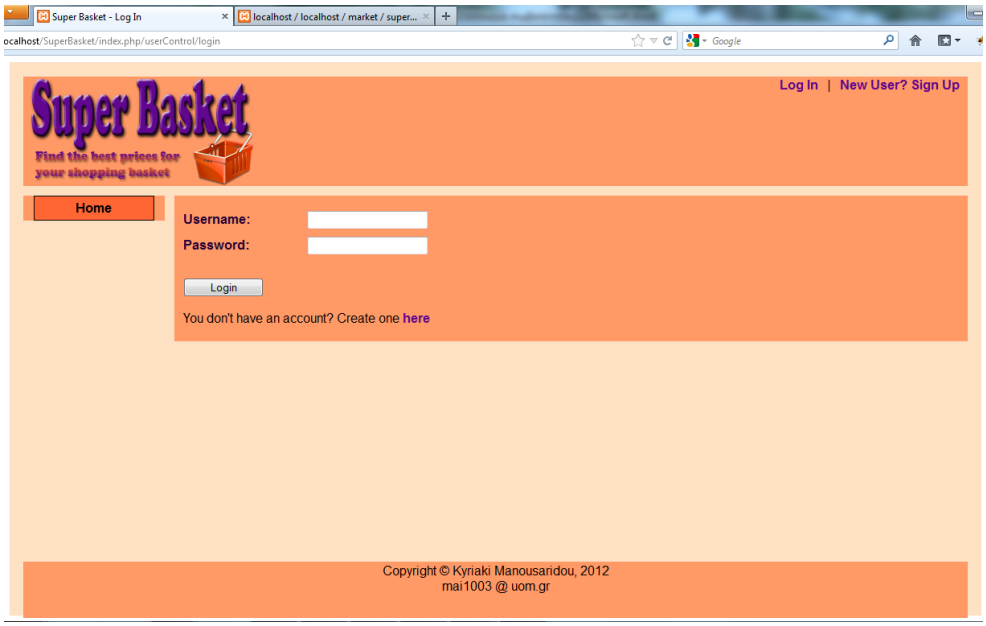
Ανά πάσα στιγμή ο χρήστης μπορεί να δει το προφίλ του (**Εικόνα 36**) από το σύνδεσμο «My Profile» στην κεφαλίδα της ιστοσελίδας. Μπορεί επίσης να επεξεργαστεί κάποια από τα στοιχεία που έχει δηλώσει.



Εικόνα 36: Σελίδα προφίλ χρήστη

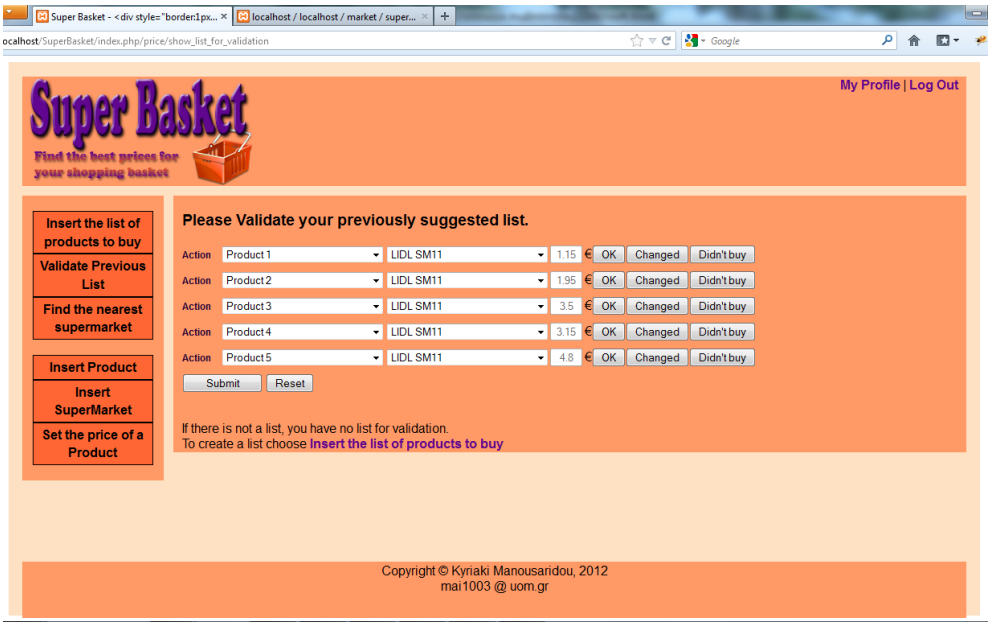
Μόλις τελειώσει με τις ενέργειες που επιθυμεί να εκτελέσει μπορεί να αποσυνδεθεί από το σύνδεσμο «Log Out».

Την επόμενη φορά που θα θελήσει να χρησιμοποιήσει το σύστημα αρχικά θα συνδεθεί με το όνομα χρήστη και τον κωδικό του. (**Εικόνα 37**).



Εικόνα 37: Σύνδεση χρήστη

Στη συνέχεια καλείται να αξιολογήσει τις τιμές της λίστας που του προτάθηκε την προηγούμενη φορά. Στη σελίδα αυτή, όπως φαίνεται στην **Εικόνα 38**, υπάρχουν οι τιμές που του προτάθηκαν και οι επιλογές «OK», «Changed» και «Didn't buy». Αν επιλέξει ότι η τιμή άλλαξε τότε πρέπει να εισάγει στο αντίστοιχο πεδίο τη νέα τιμή. Επιλέγει για κάθε τιμή αν έμεινε ίδια, άλλαξε ή δεν την αγόρασε τελικά και υποβάλλει τη λίστα. Με αυτό τον τρόπο αξιολογεί τους χρήστες που παρατήρησαν τις τιμές και αυξάνει την ανταποδοτικότητά του προς το σύστημα.



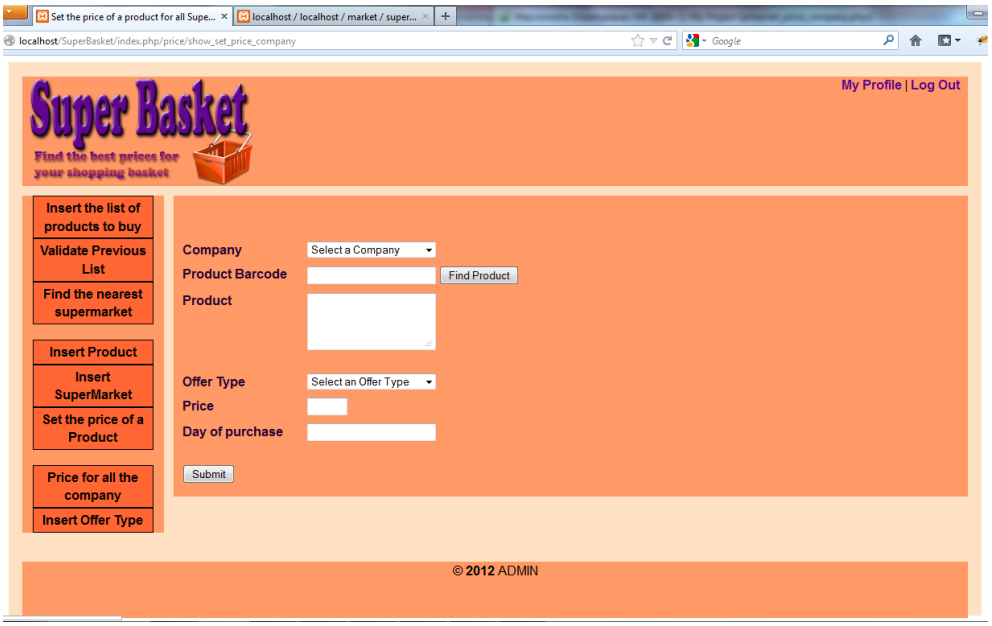
Εικόνα 38: Αξιολόγηση προηγούμενης λίστας

6.2 Διαχειριστής

Ο διαχειριστής της σελίδας μπορεί να εκτελέσει όλες τις παραπάνω λειτουργίες και δύο ακόμα επιπλέον.

Ο διαχειριστής μπορεί να εισάγει τιμή για ένα προϊόν συνολικά για όλα τα καταστήματα μιας επωνυμίας σούπερ μάρκετ. Όπως φαίνεται στην **Εικόνα 39** πρέπει να εισάγει την επωνυμία καταστήματος, το barcode του προϊόντος, τον τύπο της προσφοράς, την τιμή και την ημερομηνία αγοράς.

Η δεύτερη λειτουργία που μπορεί να εκτελέσει είναι να εισάγει νέο τύπο προσφοράς από το σύνδεσμο «Insert Offer Type».



Εικόνα 39: Εισαγωγή τιμής για μια ολόκληρη επωνυμία

ΚΕΦΑΛΑΙΟ 7

7 Συμπεράσματα

Στο κεφάλαιο αυτό καταγράφονται τα συμπεράσματα μετά την υλοποίηση της εργασίας και προτάσεις για μελλοντική εργασία.

7.1 Συμπεράσματα

Τους σύγχρονους καιρούς της πληροφόρησης και της ενημέρωσης μέσω του Διαδικτύου γινόμαστε δέκτες ενός τεράστιου όγκου δεδομένων που πολλές φορές δεν μπορούμε να διαχειριστούμε κατάλληλα ώστε να πάρουμε τις πληροφορίες που επιθυμούμε. Επίσης δεν είναι εύκολο να εντοπίσουμε άλλους ανθρώπους που μπορεί να διαθέτουν ήδη την πληροφορία αυτή ή να τους είναι χρήσιμη μια πληροφορία που διαθέτουμε εμείς. Ειδικά όταν πρόκειται για πληροφορία η οποία είναι παροδική, εμπλέκεται μια ακόμα δυσκολία, ο περιορισμός του χρόνου. Τέτοιο είδος πληροφορίας, χρήσιμη και παροδική, αποτελούν οι τιμές των προϊόντων των σούπερ μάρκετ.

Έχουν γίνει προσπάθειες ανάπτυξης συστημάτων για τη διαχείριση πληροφοριών σχετικών με τις τιμές των σούπερ μάρκετ. Τα συστήματα αυτά είναι διαδικτυακά και παρέχουν στον χρήστη τις τιμές που καταγράφουν τα ίδια, προσφορές που λαμβάνουν από τα ίδια τα σουπερ μάρκετ και τιμές που παρέχει το Υπουργείο Ανάπτυξης μέσω της δικής του εφαρμογής, το Παρατηρητήριο Τιμών. Επίσης δίνουν τη δυνατότητα στο χρήστη να επιλέξει τα προϊόντα που επιθυμεί και να δει τις τιμές τους σε όλα τα καταστήματα. Η τελική λίστα δημιουργείται από τον ίδιο το χρήστη, ο οποίος σύμφωνα με τις τιμές θα επιλέξει μόνος του ποια σουπερ μάρκετ θα επισκεφτεί.

Το σύστημα που αναπτύχθηκε στο πλαίσιο της παρούσας διπλωματικής εργασίας στηρίζεται στην ανταλλαγή πληροφοριών μεταξύ των χρηστών οι οποίοι είναι περισσότεροι από τους τιμολήπτες που μπορεί να παρέχει οποιαδήποτε από τις ήδη υπάρχουσες εφαρμογές. Επίσης το σύστημα δημιουργεί την οικονομικότερη λίστα για τον χρήστη με βάση τις επιλογές του

σχετικά με τα καταστήματα και λαμβάνοντας υπ' όψιν την απόστασή τους από την κατοικία του χρήστη. Μπορεί επίσης να παρέχει και συνδυασμούς καταστημάτων προκειμένου να βρεθεί η φτηνότερη λίστα αγορών, χωρίς να απαιτεί από το χρήστη να κάνει τους απαραίτητους υπολογισμούς.

Για να είναι ευέλικτο το σύστημα υλοποιήθηκε ως Υπηρεσία Ιστού που καλείται από μια Διαδικτυακή εφαρμογή. Για την ανάπτυξή του χρησιμοποιήθηκαν πολλές τεχνολογίες, εργαλεία και σύγχρονες εφαρμογές.

7.2 Προτάσεις

Όπως και σε κάθε άλλο σύστημα που αναπτύσσεται αρχικά υπάρχουν προτάσεις για μελλοντική βελτίωση.

Η πρώτη πρόταση βελτίωσης αφορά την αξιολόγηση και την παροχή κινήτρων στους χρήστες. Στην παρούσα υλοποίηση του συστήματος το κίνητρο παροχής πληροφορίας στο χρήστη είναι η αύξηση του αριθμού των καταστημάτων που επιθυμεί να συνδυαστούν. Θα μπορούσε να προστεθεί κάποιο κίνητρο ακόμα. Επίσης σχετικά με την αξιολόγηση των χρηστών θα μπορούσε να υπάρχει τιμωρία του αναξιόπιστου χρήστη εκτός από την προστασία του συστήματος από κάποιον αναξιόπιστο όπως έχει υλοποιηθεί τώρα. Τέλος καλό θα ήταν να ελέγχεται πότε ένας αναξιόπιστος χρήστης παρέχει όντως αξιόπιστη πληροφορία και να αυξάνεται η αξιοπιστία του όταν παρέχει μια αξιόπιστη πληροφορία μετά από κάποιες αναξιόπιστες.

Μπορούν να υπάρξουν επιπλέον βελτιώσεις στην ταχύτητα του συστήματος. Η βελτίωση της ταχύτητας μπορεί να αφορά την περίπτωση της αρχικοποίησης των χαρτών στους οποίους σημειώνονται όλα τα σούπερ μάρκετ που είναι καταχωρημένα στη βάση αλλά και τον αλγόριθμο εύρεσης καλύτερου συνδυασμού καταστημάτων όταν πρόκειται για πολλά καταστήματα.

Όσον αφορά την επέκτασή του συστήματος, θα ήταν πολύ χρήσιμη η ανάπτυξη και υλοποίηση μιας εφαρμογής – πελάτη της Υπηρεσίας Ιστού για έξυπνα κινητά τηλέφωνα. Μια τέτοια εφαρμογή θα μπορούσε να καταστήσει ακόμα πιο εύκολο στη χρήση το σύστημα και θα μπορούσε να παρέχει επιπλέον διευκολύνσεις στο χρήστη με τη χρήση λειτουργιών που

υποστηρίζουν τα έξυπνα τηλέφωνα (εντοπισμός θέσης του χρήστη, ανάγνωση κωδικών barcode κτλ).

Το σύστημα θα μπορούσε να συνδεθεί επίσης με ένα πρόγραμμα ηλεκτρονικού ψυγείου καθώς ήδη έχει αναπτυχθεί ένας πίνακας στη βάση δεδομένων. Με αυτό τον τρόπο θα μπορούσε να «γνωρίζει» από μόνο του τα προϊόντα που χρειάζεται ο χρήστης και να του παρέχει άμεσα τη λίστα αγοράς.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] "Δελτίο Τύπου: ΠΑΡΑΤΗΡΗΤΗΡΙΑ ΤΙΜΩΝ – ΔΕΙΚΤΗΣ ΦΤΗΝΙΑΣ – ΝΕΑ ΕΡΓΑΛΕΙΑ ΓΙΑ ΤΗΝ ΚΑΛΗ ΛΕΙΤΟΥΡΓΙΑ ΤΟΥ ΑΝΤΑΓΩΝΙΣΜΟΥ – ΑΥΣΤΗΡΟΙ ΕΛΕΓΧΟΙ ΣΤΗΝ ΑΓΟΡΑ ΕΝΟΨΕΙ ΕΟΡΤΩΝ," *Γενική Γραμματεία Εμπορίου, Υπουργείο Ανάπτυξης Ανταγωνιστικότητας & Ναυτιλίας*, 2010. [Online]. Available: http://www.gge.gr/7/sub.asp?4564. [Accessed: 10-Sep-2012].

[2] G. Glass, "The Evolution of Web Services," 2002. [Online]. Available: http://www.informit.com/articles/article.aspx?p=27295. [Accessed: 16-Sep-2012].

[3] W3C Consortium, "Web Services Architecture," 2004.

[4] "Web Services Tutorial," *W3schools*. [Online]. Available: http://www.w3schools.com/webservices/default.asp. [Accessed: 16-Sep-2012].

[5] G. Alonso, F. Casati, H. Kuno, and V. Machiraju, "Web Services," in *Web Services - Concepts, Architectures and Applications*, no. Chapter 1, 2004.

[6] V. Beal, "Understanding Web Services - Webopedia.com," 2005. [Online]. Available: http://www.webopedia.com/DidYouKnow/Computer_Science/2005/web_services.asp. [Accessed: 16-Sep-2012].

[7] D. Chappell and T. Jewell, *Java Web Services*. O' Reilly, 2002.

[8] B. Sotomayor, *The Globus Toolkit 3 Programmer ' s Tutorial*. 2003.

[9] D. Nations, "What is a Web Application?" [Online]. Available: http://webtrends.about.com/od/webapplications/a/web_application.htm. [Accessed: 23-Sep-2012].

[10] J. Deacon, "Model-View-Controller (MVC) Architecture," 2009.

[11] S. Hansen and T. V. Fossum, "Refactoring model-view-controller," *Consortium for Computing Sciences in Colleges*, 2005.

[12] A. Leff and J. T. Rayfield, "Web-Application Development Using the ModelNiewlController Design Pattern," *Control*, pp. 118–127, 2001.

[13] “Model View Controller,” *CocoaLab*. [Online]. Available: http://www.cocoalab.com/?q=node/24. [Accessed: 24-Sep-2012].

[14] “Model View Controller (MVC) with JSP and JSTL,” *Jansipke.nl*. [Online]. Available: http://www.jansipke.nl/model-view-controller-mvc-with-jsp-and-jstl/. [Accessed: 24-Sep-2012].

[15] “PHP: Hypertext Preprocessor,” *The PHP Group*. [Online]. Available: http://gr.php.net/. [Accessed: 23-Sep-2012].

[16] “Usage of Server-side Programming Languages for Websites,” *W3Techs*. [Online]. Available: http://w3techs.com/technologies/overview/programming_language/all. [Accessed: 23-Sep-2012].

[17] “Top 10 PHP Frameworks,” *PHP Zone*. [Online]. Available: http://php.dzone.com/news/top-10-php-frameworks. [Accessed: 24-Sep-2012].

[18] “About the Apache HTTP Server Project,” *Apache Software Foundation*. [Online]. Available: http://httpd.apache.org/ABOUT_APACHE.html. [Accessed: 25-Sep-2012].

[19] “September 2012 Web Server Survey,” *Netcraft*. [Online]. Available: http://news.netcraft.com/archives/2012/09/10/september-2012-web-server-survey.html. [Accessed: 24-Sep-2012].

[20] “Why MySQL?,” *Oracle*. [Online]. Available: http://www.mysql.com/why-mysql/. [Accessed: 25-Sep-2012].

[21] “List of Apache–MySQL–PHP packages,” *Wikipedia*. [Online]. Available: http://en.wikipedia.org/wiki/List_of_AMP_packages. [Accessed: 24-Sep-2012].

[22] L. Lemay, *Πλήρες Εγχειρίδιο της HTML 4*. Μ. Γκιούρδας, 2000.

[23] J. C. Teague, *Speaking in styles : fundamentals of CSS for Web designers*. 2009.

[24] A. Budd, C. Moll, and S. Collison, *CSS mastery : advanced web standards solutions*. 2006.

[25] M. Webb, M. Plungjan, and K. Drakard, *Instant JavaScript*. McGraw - Hill, 2001.

[26] “jQuery: The Write Less, Do More, JavaScript Library,” *JQuery.com*, 2012. [Online]. Available: http://jquery.com/. [Accessed: 25-Sep-2012].

[27] D. Chen, “Top 8 Reasons Why jQuery is so Popular,” 2010. [Online]. Available: http://blogs.msdn.com/b/dorischen/archive/2010/08/27/top-8-reasons-why-jquery-is-so-popular.aspx. [Accessed: 25-Sep-2012].

[28] J. Garrett, “Ajax : A New Approach to Web Applications How Ajax is Different,” pp. 1–5, 2005.

[29] M. Stepp, J. Miller, and V. Kirst, *Web Programming Step by Step*, Second Edi. Step by Step Publishing, 2012.

[30] Νίκος Κολλάρας, “Ο Σημασιολογικός Ιστός,” Πανεπιστήμιο Πατρών, Σχολή Θετικών Επιστημών, 2007.

[31] “What is RDF and what is it good for?,” *rdf about*, 2008. [Online]. Available: http://www.rdfabout.com/intro/. [Accessed: 25-Sep-2012].

[32] Παρασκευή Χαμοπούλου, “Σύστημα Διαχείρισης Δεξιοτήτων , Βασισμένο σε Οντολογίες και στο Σημασιολογικό Ιστό ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ,” 2006.

[33] “An Introduction to RDF and the Jena RDF API,” *Apache Jena*, 2011. [Online]. Available: http://jena.apache.org/tutorials/rdf_api.html. [Accessed: 25-Sep-2012].

[34] J. Pérez, M. Arenas, and C. Gutierrez, “Semantics and complexity of SPARQL,” *ACM Transactions on Database Systems*, vol. 34, no. 3, pp. 1–45, Aug. 2009.

[35] Αλεξανδρος Ν. Χατζηγεωργίου, *Ανάπτυξη συστήματος λογισμικού βάσει της μεθοδολογίας ICONIX*. Ελληνικό Ανοικτό Πανεπιστήμιο, 2008.

[36] J. Blanco and D. Upton, “CodeIgniter 1.7,” 2009.

[37] “Apache Axis2 User’s Guide- Introducing Axis2,” *Apache Axis2*. [Online]. Available: http://axis.apache.org/axis2/java/core/docs/userguide.html. [Accessed: 16-Sep-2012].

ΠΑΡΑΡΤΗΜΑ

Εικόνα Α: Λίστα συναρτήσεων της Υπηρεσίας Ιστού (πράσινο:δημόσιες, κόκκινο: ιδιωτικές)

Οδηγίες Εγκατάστασης Εφαρμογής από το CD.

1. Κατέβασμα και εγκατάσταση του πακέτου XAMPP όπως περιγράφεται στην **Ενότητα 5.2.1.1**
2. Τοποθέτηση του περιεχομένου του φακέλου «Web Application», δηλαδή του φακέλου «SuperBarketApp» στο φάκελο xampp\htdocs.
3. Δημιουργία των βάσεων market και semanticdb στη MySQL με τη βοήθεια του phpMyAdmin και των αρχείων «market.sql» και «semanticdb.sql»
4. Εγκατάσταση του Eclipse
5. Εγκατάσταση του Tomcat στο Eclipse όπως περιγράφηκε στην **ενότητα 5.2.2.2**
6. Εγκατάσταση του Axis2 στο Eclipse όπως περιγράφηκε στην **ενότητα 5.2.2.3**
7. Εγκατάσταση της βιβλιοθήκης χρήστη JenaLib σύμφωνα με τις οδηγίες της **ενότητας 5.2.2.5**
8. Δημιουργία ενός καινούριου project σύμφωνα με τα βήματα 1-4 της **ενότητας 5.2.2.4**
9. Δημιουργία νέο πακέτου pkg στο φάκελο src του project (Δεξί κλικ -> new package -> όνομα pkg)
10. Εισαγωγή των αρχείων του Web Service από το φάκελο του CD WebService\SuperBasket\src\pkg στο φάκελο *ονομαΝεουproject*\src\pkg (Δεξί κλίκ -> import -> General -> FIleSystem -> From directory “WebService\SuperBasket\src\pkg” into “*ονομαΝεουproject*\src\pkg”)
11. Εκτέλεση του project (run on server)
12. Κλήση από τον browser της SuperBasketApp (localhost\SuperBasketApp)