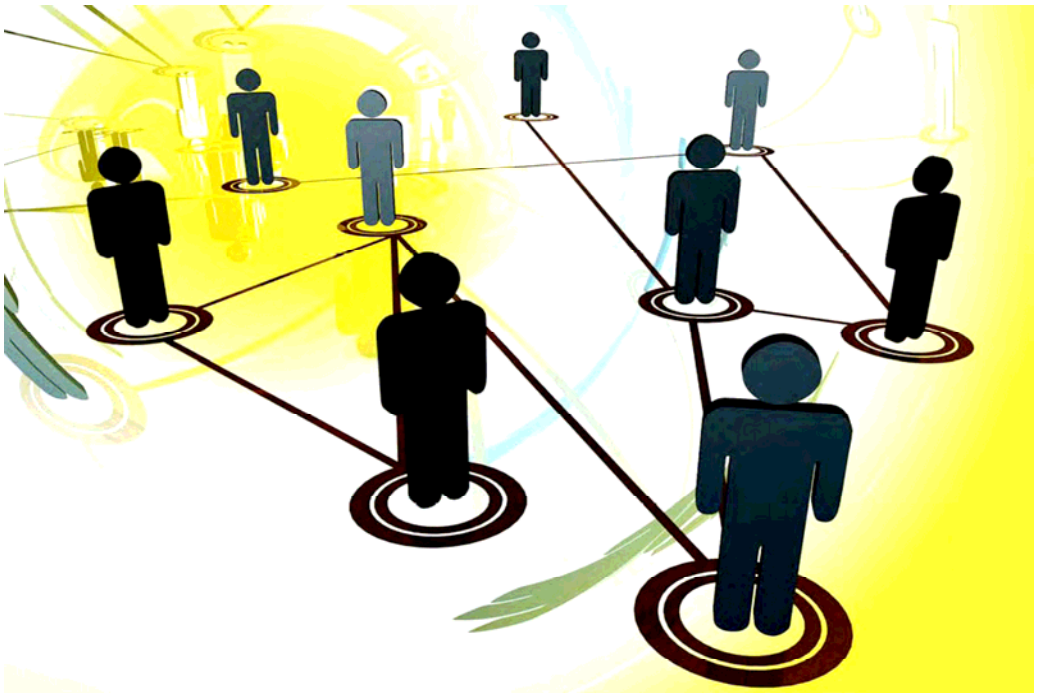
ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή εργασία

Δημιουργία ενός δυναμικού κοινωνικού ιστοχώρου για τους υποψηφίους των εξετάσεων του ΑΣΕΠ (online social network site) με χρήση PHP, MySQL και JavaScript.



Του φοιτητή:
Κωνσταντίνου Μητσαράκη
Αρ. Μητρώου: 1717

Επιβλέπων καθηγητής:
Αθανάσιος Μάργαρης

Θεσσαλονίκη 2009

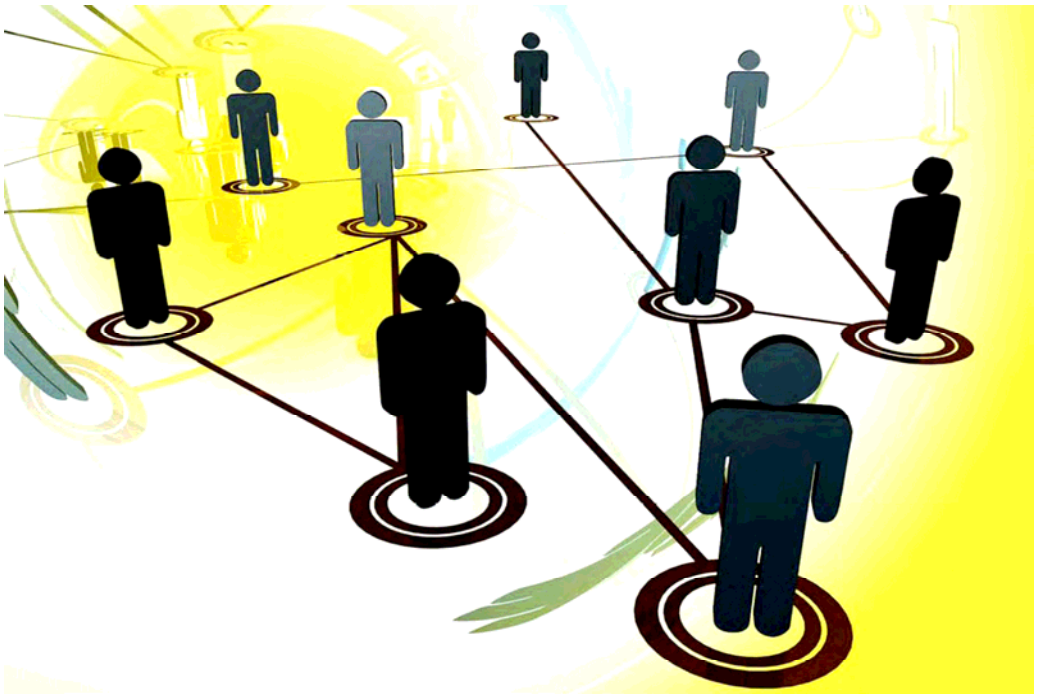
ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πτυχιακή εργασία

Δημιουργία ενός δυναμικού κοινωνικού ιστοχώρου για τους υποψηφίους των εξετάσεων του ΑΣΕΠ (online social network site) με χρήση PHP, MySQL και JavaScript.



Του φοιτητή:
Κωνσταντίνου Μητσαράκη
Αρ. Μητρώου: 1717

Επιβλέπων καθηγητής:
Αθανάσιος Μάργαρης

Θεσσαλονίκη 2009

Στην οικογένειά μου

ΠΡΟΛΟΓΟΣ

Η εφαρμογή αυτή δημιουργήθηκε το ακαδημαϊκό έτος 2008 – 2009 από τον Κωνσταντίνο Μητσαράκη, φοιτητή του τμήματος Πληροφορικής του ΑΤΕΙ, υπό την εποπτεία του καθηγητή και Επιστημονικού Συνεργάτη της σχολής κ. Αθανάσιο Μάργαρη, στα πλαίσιο εκπόνησης πτυχιακής εργασίας με τίτλο: Δημιουργία ενός δυναμικού κοινωνικού ιστοχώρου για τους υποψηφίους των εξετάσεων του ΑΣΕΠ (online social network site) με χρήση PHP, MySQL και JavaScript.

Η ιδέα του θέματος αυτού αποφασίστηκε όταν τον Οκτώβριο του 2008, πάνω από 70.000 άτομα αιτήθηκαν για υποψηφιότητα στις γραπτές εξετάσεις των εκπαιδευτικών που προκηρύχθηκαν από το Ανώτατο Συμβούλιο Επιλογής Προσωπικού (ΑΣΕΠ). Σημειώνεται ότι το ΑΣΕΠ είναι υπεύθυνο για την πρόσληψη χιλιάδων εργαζομένων κάθε χρόνο σε δεκάδες οργανισμούς και οι προσλήψεις αυτές γίνονται είτε με γραπτούς διαγωνισμούς είτε με μοριοδότηση.

Περίληψη

Στα πλαίσια της πτυχιακής αυτής, θα αναπτυχθεί μία διαδικτυακή εφαρμογή ενός κοινωνικού ιστοχώρου για τους υποψηφίους των Διαγωνισμών του ΑΣΕΠ, με χρήση των τεχνολογιών HTML, CSS, PHP, MySQL και JavaScript. Η εφαρμογή θα παρέχει σε κάθε χρήστη τις δυνατότητες που προσφέρει κάθε κοινωνικός ιστοχώρος, όπως είναι η δημιουργία προφίλ χρήστη, η αποστολή μηνυμάτων και η δημιουργία λίστας φίλων. Επιπλέον, θα παρέχει τη δυνατότητα σε κάθε μέλος να αναρτά με δυναμικό τρόπο αρχεία καθώς επίσης και την επιλογή και κατέβασμα αυτών που υπάρχουν ήδη αποθηκευμένα στην εφαρμογή από άλλους χρήστες. Σημαντικό σημείο επίσης είναι και η δυνατότητα αναζήτησης προκηρύξεων του ΑΣΕΠ. Πέρα από την υλοποίηση της εφαρμογής με την χρήση των προαναφερομένων τεχνολογιών, θα αναπτυχθεί η τεκμηρίωση, η οποία θα περιλαμβάνει το αρχικό σχέδιο έργου, καθώς και η ανάλυση του συστήματος. Η υλοποίηση θα περιλαμβάνει αρχικά την σχεδίαση και δημιουργία μίας βάσης δεδομένων η οποία θα εξυπηρετήσει στην αποθήκευση της σχετικής πληροφορίας, κι έπειτα την ανάπτυξη της διεπαφής χρήστη για την αλληλεπίδραση των μελών τόσο με την εφαρμογή όσο και μεταξύ τους. Τέλος, η εφαρμογή θα αναρτηθεί σε μία πραγματική ιστοσελίδα για να είναι διαθέσιμη για όλους.

Λέξεις Κλειδιά: PHP, MySQL, JavaScript, DOM, HTML, CSS, SNS.

Abstract

In this thesis will be developed a web application of a profile based community website focused on candidates of the Examines of Highest Council of Personel Selecltion (EHCPS), using HTML, PHP, CSS, MySQL and JavaScript. The application will provide at all members of it the functions that such an application offers, like user account creation, sent messages and friend lists. Moreover, will be provided the capabilities of dynamic file uploading, file selection an file downloading, from thoses files that already are stored in the application uploaded by other users. Another important element is the EHCPS news search. Apart from the implementation of the application using the required technologies, the documentation will be developed also, whici will include the initial project plan as well as the analysis of the system. The implementation firstly will include the design and creation of the database which will served at the store of the relevant information, and secondly the development the layout required for user interaction with the application as well as with each other. Finally, the application will be published on a real website to be available for all.

Keywords: PHP, MySQL, JavaScript, DOM, HTML, CSS, SNS.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΡΟΛΟΓΟΣ 1

Περίληψη 5

Abstract 6

ΠΕΡΙΕΧΟΜΕΝΑ 7

Ευρετήριο Σχημάτων και Πινάκων 11

ΕΙΣΑΓΩΓΗ 12

ΠΡΩΤΟ ΜΕΡΟΣ 13

ΚΕΦΑΛΑΙΟ 1ο 13

ΔΙΑΔΙΚΤΥΑΚΟΙ ΚΟΙΝΩΝΙΚΟΙ ΙΣΤΟΧΩΡΟΙ 13

1.1 Εισαγωγή 13

1.2 Χρησιμότητα ενός Διαδικτυακού Κοινωνικού Ιστοχώρου 13

1.3 Τι είναι εικονική κοινότητα 14

1.4 Τι είναι Διαδικτυακός Κοινωνικός Ιστοχώρος 15

1.5 Χαρακτηριστικά των Διαδικτυακών Κοινωνικών Ιστοχώρων 16

1.6 Κυριότεροι διαδικτυακοί κοινωνικοί ιστοχώροι 18

1.7 Επίλογος 19

ΚΕΦΑΛΑΙΟ 2ο 20

WEB 2.0 20

2.1 Εισαγωγή 20

2.2 Χρησιμότητα του Web 2.0 20

2.3 Κύριες αρχές του Web 2.0 21

2.4 Τεχνολογία του συμμετοχικού διαδικτύου 21

2.5 Ανάπτυξη Web 2.0 ιστοσελίδων 23

2.6 Web 2.0 και Online Communities 24

2.7 Επίλογος 26

ΚΕΦΑΛΑΙΟ 3ο 27

HTML 27

3.1 Εισαγωγή 27

3.2 World Wide Web Consortium (W3C) 27

3.3 Στοιχεία της HTML 28

3.4 Χρήση HTML στην εφαρμογή 28

3.5 Επίλογος 36
ΚΕΦΑΛΑΙΟ 4ο *37*
CSS *37*
4.1 Εισαγωγή 37
4.2 Διαχωρισμός HTML και εμφάνισης 37
4.3 Δομή – Παρουσίαση – Συμπεριφορά 37
4.4 Βασικές Αρχές του CSS 38
4.5 CSS Selectors 40
4.6 Διακριτότητα 50
4.7 Επισύναψη Style Sheet στα έγγραφα 51
4.8 Διακριτότητα διαφορετικών πηγών στυλ 54
4.9 Επίλογος 54
ΚΕΦΑΛΑΙΟ 5ο *55*
JAVASCRIPT ΚΑΙ DOM *55*
5.1 Εισαγωγή 55
5.2 JavaScript 55
5.3 JavaScript και Document Object Model (DOM) 57
5.4 Επίλογος 76
ΚΕΦΑΛΑΙΟ 6ο *77*
PHP *77*
6.1 Εισαγωγή 77
6.2 Βασικά χαρακτηριστικά 77
6.3 Αρχιτεκτονική Βάσης Δεδομένων με PHP – MySQL 78
6.4 Apache Web Server 81
6.5 Hypertext Transfer Protocol (HTTP) 81
6.6 Εγκατάσταση εργαλείων 81
6.7 Επίλογος 82

ΔΕΥΤΕΡΟ ΜΕΡΟΣ *83*

ΚΕΦΑΛΑΙΟ 7ο *83*
ΑΝΑΛΥΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ *83*
7.1 Εισαγωγή 83
7.2 Ανάλυση σχεδίου εργασίας έργου και φάσεων ανάπτυξης 83
7.3 Τεκμηρίωση με εργαλεία Επιχειρησιακής Έρευνας 87
7.4 Διερεύνηση υπαρχόντων συστημάτων 88

7.5 Μελέτη έργου 88
7.6 Επιλογή μοντέλου ανάπτυξης της εφαρμογής 89
7.7 Επίλογος 100
ΚΕΦΑΛΑΙΟ 8ο *101*
ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ *101*
8.1 Εισαγωγή 101
8.2 Τι είναι MySQL 101
8.3 Κανονικοποίηση 101
8.4 Σχεσιακό Μοντέλο 103
8.5 Οντότητες και Συσχετίσεις 106
8.6 Διάγραμμα Οντοτήτων – Συσχετίσεων (ER) 107
8.7 Τεκμηρίωση πινάκων 116
ΚΕΦΑΛΑΙΟ 9ο *130*
ΥΛΟΠΟΙΗΣΗ ΣΕ PHP *130*
9.1 Εισαγωγή 130
9.2 Υπερκαθολικές μεταβλητές $_GET και $_POST 130
9.3 Η υπερκαθολική μεταβλητή $_SESSION 132
9.4 SQL injection και προστασία βάσης δεδομένων 136
9.5 Σύνδεση με την βάση δεδομένων 138
9.6 Διαφυγή χαρακτήρων 139
9.7 Έλεγχος εγκυρότητας τιμών 140
9.8 Κρυπτογράφηση κωδικών 141
9.9 Επικύρωση εγγραφής χρήστη 142
9.10 Πρότυπο layout 144
9.11 Επιλογή κλάδου 145
9.12 Διαμόρφωση ημερομηνίας 145
9.13 Ενημέρωση για ανάρτηση νέων αρχείων 146
9.14 Εύρεση πλήθους φίλων 146
9.15 Εμφάνιση ή όχι του profile χρήστη 147
9.16 Εμφάνιση τυχαίων φίλων 148
9.17 Αποδοχή μηνυμάτων 148
9.18 Διαγραφή φίλων 149
9.19 Έλεγχος αναφοράς σελίδας 151
9.20 Έλεγχος δικαιώματος πρόσβασης σελίδας 152
9.21 Ανάρτηση αρχείου 153

9.22 Download αρχείου 155
9.23 BlackList 155
9.24 Διαγραφή μέλους 156
9.25 Αποσύνδεση χρήστη 157
9.26 Σελιδοποίηση αποτελεσμάτων 157
9.27 Εμφάνιση αποτελεσμάτων 160
9.28 Έλεγχος δεδομένων με JavaScript 161
Οδηγός Χρήσης Λογισμικού *163*
Βιβλιογραφία *164*

Ευρετήριο Σχημάτων και Πινάκων

Πίνακες

Πίνακας I: Σύγκριση Web 1.0 και Web 2.0 21
Πίνακας II: Τύποι κοινωνικών λογισμικών 22
Πίνακας III: Υπολογισμός διακριτότητας 49
Πίνακας VI: Σύμβολα Διαγράμματος ER 104

Σχήματα

Σχήμα I: Τα επίπεδα ενός Web εγγράφου 36
Σχήμα II: Τα στοιχεία ενός CSS rule 37
Σχήμα III: Αναπαράσταση ενός δέντρου DOM 55
Σχήμα IV: Elements και attributes 57
Σχήμα V: Η σχέση browser / server 77
Σχήμα VI: Βασική Web αρχιτεκτονική βάσεων δεδομένων 78
Σχήμα VII: Διάγραμμα Οντοτήτων – Συσχετίσεω 107
Σχήμα VIII: Πίνακας Admin 108
Σχήμα IX: Πίνακας BlackList 108
Σχήμα X: Πίνακας Changes 109
Σχήμα XI: Πίνακας Contacts 109
Σχήμα XII: Πίνακας deletedMembers 110
Σχήμα XIII: Πίνακας Files 110
Σχήμα XIV: Πίνακας Members 111
Σχήμα XV: Πίνακας Messages 111
Σχήμα XVI: Πίνακας News 112
Σχήμα XVII: Πίνακας Online 112
Σχήμα XVIII: Πίνακας Settings 113
Σχήμα XIX: Πίνακας unconfirmMembers 113
Σχήμα XX: Οι πίνακες του συστήματος 114

ΕΙΣΑΓΩΓΗ

Οι δικτυακοί κοινωνικοί ιστοχώροι όπως το Friendster, το Myspace και το Facebook έχουν πολλαπλασιάσει αλματωδώς τα μέλη τους κατά τη διάρκεια των λίγων χρόνων που έκαναν την εμφάνισή τους και έχουν προσελκύσει την προσοχή τόσο της ακαδημαϊκής όσο και της εμπορικής έρευνας, κάθε μίας για τους δικούς της λόγους. Αυτά τα δίκτυα προσφέρουν εύκολα κι ελκυστικά μέσα για αλληλεπίδραση και επικοινωνία μεταξύ των μελών τους εκμεταλλευόμενα τις δυνατότητες που παρέχονται από την χρήση του Web 2.0. Από την άλλη μεριά όμως, εγείρουν θέματα περί προστασίας της ιδιωτικής ζωής και δικτυακής ασφάλειας λόγω της έκθεσης προσωπικών δεδομένων στο τόσο μεγάλο διαδικτυακό κοινό. Σε αυτήν την πτυχιακή εργασία αναλύουμε, σχεδιάζουμε και υλοποιούμε μία εφαρμογή δικτυακού κοινωνικού ιστοχώρου με δυνατότητα υποστήριξης δυναμικής ανάρτησης (upload) αρχείων, εξετάζοντας τις απαραίτητες εκείνες λειτουργίες που χαρακτηρίζουν μία τέτοια εφαρμογή καθώς και τις αναγκαίες ενέργειες που πρέπει να γίνουν για την ελαχιστοποίηση των προβλημάτων που μπορεί να προκύψουν.

Το θέμα του ιστοχώρου προσεγγίζει τους διαγωνισμούς που διοργανώνονται από τον ΑΣΕΠ κάθε χρόνο και λαμβάνουν μέρος σε αυτούς χιλιάδες υποψήφιοι. Παραδείγματα τέτοιων είναι ο διαγωνισμός για την πρόσληψη εκπαιδευτικών, εφοριακών, το Τεστ Δεξιοτήτων καθώς και διαγωνισμοί μοριοδότησης για πρόσληψη σε διάφορα Υπουργεία του κράτους.

Στόχος την πτυχιακής είναι η δημιουργία ενός χώρου όπου οι υποψήφιοι θα μπορούν να ενημερώνονται σχετικά με τους διαγωνισμούς αλλά και να έχουν την δυνατότητα γνωριμίας μεταξύ τους ώστε να ανταλλάσσουν πληροφορίες και αρχεία για τις διάφορες προκηρύξεις.

Τα κεφάλαια που ακολουθούν χωρίζονται σε δύο μέρη:

Στο πρώτο μέρος αναλύεται ο ορισμός των κοινωνικών ιστοχώρων και των χαρακτηριστικών του κι έπειτα αναπτύσσονται οι διάφορες τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση της εφαρμογής.

Στο δεύτερο μέρος αναπτύσσεται η ανάλυση της εφαρμογής καθώς και τα κυριότερα κομμάτια του κώδικα αλλά και το εγχειρίδιο χρήσης της.

ΠΡΩΤΟ ΜΕΡΟΣ

ΚΕΦΑΛΑΙΟ 1ο

ΔΙΑΔΙΚΤΥΑΚΟΙ ΚΟΙΝΩΝΙΚΟΙ ΙΣΤΟΧΩΡΟΙ

1.1 Εισαγωγή

Από την πρώτη στιγμή της λειτουργίας τους, οι Δικτυακοί Κοινωνικοί Ιστοχώροι – ΔΚΙ – (social network sites –SNSs-) όπως το MySpace, το Facebook και το Bebo έχουν προσελκύσει εκατομμύρια χρήστες, πολλοί από τους οποίους χρησιμοποιούν καθημερινά τις δυνατότητες που τους προσφέρουν. Αυτήν την στιγμή υπάρχουν εκατοντάδες SNSs που καλύπτουν με παρόμοιους ή όχι τρόπους μία ευρεία κλίμακα ενδιαφερόντων. Τα περισσότερα δίκτυα αυτού του είδους υποστηρίζουν την διατήρηση προηγουμένων δομών κοινωνικών δικτύων βασισμένα στην φυσική διάσταση της ζωής, όμως άλλα επιτρέπουν σε άγνωστα μεταξύ τους άτομα να έρθουν σε επαφή βασισμένοι σε κοινά ενδιαφέροντα, πολιτικές ιδέες, στόχους ή δραστηριότητες. Μερικές ιστοσελίδες έχουν ως στόχο να έχουν μεγάλη ποικιλία κοινού, ενώ άλλες προτιμούν να προσελκύουν ανθρώπους που έχουν την ίδια γλώσσα, φύλο, σεξουαλικό προσανατολισμό, θρησκεία ή εθνικότητα. Επίσης, υπάρχουν διαφορές σε κάθε εφαρμογή, όσον αφορά στην αλληλεπίδραση μεταξύ των μελών τους μέσω των εργαλείων επικοινωνίας που προσφέρουν, όπως μηνύματα, forum, blog, φωτογραφίες κ.ά.
Στην συνέχεια γίνεται αναφορά στην αλλαγή τάσης της χρήσης του διαδικτύου με την έλευση του Web 2.0, στους στόχους και τη χρήση των δικτυακών κοινωνικών ιστοχώρων και ακολουθεί εννοιολογική περιγραφή και ιστορική αναδρομή τους.

1.2 Χρησιμότητα ενός Διαδικτυακού Κοινωνικού Ιστοχώρου

Εκτός από τους προφανείς κοινωνικούς λόγους (γνωριμίες, φιλίες, κοινωνικότητα) που μπορεί να έχει κανείς για να δημιουργήσει τον δικό του χώρο σε ένα online social network site ή ακόμα και να φτιάξει το δικό του SNS, προσφέρεται και η δυνατότητα προβολής ταλέντων, δυνατοτήτων ή ακόμα και προσωπικών έργων που μπορούν να προσελκύσουν αγοραστές ή κόσμο που να μπορεί να προωθήσει σε σωστούς κύκλους κάποια ταλέντα. Για παράδειγμα κάποιος άσημος τραγουδιστής ή τραγουδίστρια, από τη μια μέρα στην άλλη μπορεί να γίνει

διάσημος ανεβάζοντας ένα βίντεο και επιδεικνύοντας τη φωνή του, προσελκύοντας έτσι μάνατζερ και κόσμο που θα προωθήσουν το άτομο αυτό. Μάλιστα υπάρχει μεγάλη άνοδος του αριθμού των λεγόμενων web celebrities, δηλαδή διασημοτήτων που έγιναν γνωστοί μέσω του διαδικτύου και δεν είναι λίγοι από αυτούς που έγιναν γνωστοί μέσω τέτοιων SNS.

Αλλά ισχύει και το ανάποδο, δηλαδή φτασμένοι καλλιτέχνες ή διάσημα πρόσωπα, να φτιάχνουν είτε τα δικά τους social networks, είτε σελίδες μελών σε γνωστά SNSs. Χαρακτηριστικό παράδειγμα είναι το My.BarackObama.com, ένα social network site προώθησης της προεδρικής καμπάνιας του υποψηφίου για το χρίσμα των Δημοκρατικών στις Ηνωμένες Πολιτείες. Επίσης, οι εταιρείες παρακολουθώντας τα τεκταινόμενα στο διαδίκτυο και την τεράστια προσέλκυση κοινού – άρα και πιθανών πελατών – σε τέτοιου είδους community sites έχουν κάνει και αυτές τα δικά τους βήματα σε αυτόν τον τομέα.

Ένα βήμα πιο πέρα είναι οι προτάσεις που πραγματικά δείχνουν τι να περιμένουμε από την ηλεκτρονική παιχνιδοβιομηχανία στο μέλλον, είναι τα 3D community network applications που με τέτοιου είδους υπηρεσίες δίνουν τη δυνατότητα σε παίχτες να επικοινωνούν, να μοιράζονται και να ανταλλάσσουν δεδομένα τα οποία οι ίδιοι δημιουργούν γνωρίζουν τεράστια επιτυχία. Τα περισσότερα άτομα 14 – 30 χρονών γνωρίζουν τα διαδικτυακά παιχνίδια ρόλων, όπως είναι τα Lineage και World of Warcraft (WoW), τα οποία έχουν ήδη μεγάλη επιτυχία αφού φέρνουν σε αλληλεπίδραση εκατομμύρια ανθρώπους από ολόκληρο τον κόσμο και διαθέτουν το δικό τους δίκτυο παικτών.

Αυτό δείχνει τη δυνατότητα να υπάρχουν μεγάλα οικονομικά οφέλη μέσα σε τόσο τεράστιες κοινότητες, πολλές φορές μάλιστα κατηγοριοποιημένων σε συγκεκριμένα target groups. Για παράδειγμα υπάρχουν social networks για τουρισμό, για σχέσεις, για μαθητές, για μουσική, για δημοσιογραφία κ.ά.

1.3 Τι είναι εικονική κοινότητα

Μία εικονική κοινότητα είναι ένα πληροφοριακό σύστημα δικτυακών κοινωνικών ιστοχώρων όπου οι συμμετέχοντες έχουν τα ίδια ενδιαφέροντα, ιδέες, σκοπούς, προσδοκίες κ.ά. και επικοινωνούν σε μία εικονική κοινωνία πέρα από τα όρια που θέτει ο χρόνος και η απόσταση επιτρέποντας την ανάπτυξη προσωπικών σχέσεων ανάμεσα στα μέλη της. Αποτελείται από ανθρώπους που επικοινωνούν και ανταλλάζουν δεδομένα και πληροφορίες με τη χρήση γραμμάτων, τηλεφώνου,

ηλεκτρονικού ταχυδρομείου παρά πρόσωπο με πρόσωπο και η επικοινωνία αυτή γίνεται για κοινωνικούς, επαγγελματικούς, εκπαιδευτικούς ή άλλους σκοπούς. Αν ο μηχανισμός επικοινωνίας, όπως η περίπτωση που εξετάζουμε, είναι ένα δίκτυο υπολογιστών, τότε αυτή η κοινότητα λέγεται δικτυακή κοινότητα. Οι εικονικές και οι δικτυακές κοινότητες έχουν γίνει συμπληρωματική μορφή επικοινωνίας ανάμεσα στους ανθρώπους που έχουν γνωριστεί εκ των προτέρων στην καθημερινή τους ζωή αν και δε σημαίνει ότι πρέπει να υπάρχει απαραιτήτως ένας ισχυρός δεσμός μεταξύ των μελών. Συνήθως όμως οι δεσμοί των ατόμων που συσχετίζονται μέσα από τέτοιες κοινότητες είναι πολιτισμικοί. Οι εικονικές κοινότητες εξαρτώνται από την επικοινωνία μεταξύ των χρηστών με τις δυνατότητες και τα εργαλεία που προσφέρονται από τα μέσα αλληλεπίδρασης. Τα εργαλεία που χρησιμοποιούνται σε τέτοιες εφαρμογές παρέχουν δυνατότητες επικοινωνίας και αλληλεπίδρασης και χειρίζονται την παρουσίαση και την αποθήκευση της επικοινωνίας η οποία συνήθως είναι γραπτή αλλά μπορεί να περιλαμβάνει εικόνα και/ή ήχο. Επίσης, υπάρχουν εργαλεία που αποσκοπούν στην καθιέρωση και διατήρηση σύνδεσης ανάμεσα σε χρήστες χρησιμοποιώντας μηχανισμούς συζήτησης (chat) σύγχρονης ή και ασύγχρνης (mail, blog, forum). Τα εργαλεία επικοινωνίας είναι συνήθως ασύγχρονα ενώ τα εργαλεία αλληλεπίδρασης είναι σύγχρονα επιτρέποντας στους χρήστες να επικοινωνούν σε πραγματικό χρόνο.

1.4 Τι είναι Διαδικτυακός Κοινωνικός Ιστοχώρος

Μετά την κοινωνικοποίηση στην πραγματική ζωή έρχεται η ώρα που πλέον η κοινωνικοποίηση στην εικονική ζωή του διαδικτύου έρχεται να επιβεβαιώσει την προσωπικότητα κάποιου ατόμου και να το καταξιώσει μέσα στο περιβάλλον του. Παλιότερα αυτό γινόταν μέσα από τη συμμετοχή σε λέσχες, συλλόγους οργανώσεις και άλλων παρομοίων κοινωνικών οντοτήτων μέσα στον τρισδιάστατο κόσμο που νοούνταν ως κοινωνικός περίγυρος του καθενός. Σήμερα η επιθυμία αυτή σε ένα βαθμό γίνεται πραγματικότητα με τη βοήθεια της τεχνολογίας και αποτέλεσμα αυτού είναι η συμμετοχή ενός ολοένα και αυξανόμενου αριθμού χρηστών του Ίντερνετ σε εφαρμογές, ή μάλλον καλύτερα, σε εικονικές κοινωνίες στο χώρο του διαδικτύου που στην γλώσσα της κοινωνιολογίας αλλά και της πληροφορικής είναι πλέον γνωστά ως κοινωνικοί ιστοχώροι (social networks). Χαρακτηριστικά παραδείγματα τέτοιων είναι το MySpace, το Facebook και το Hi5 τα οποία έχουν αλλάξει ριζικά τον τρόπο με τον οποίο γίνεται η επικοινωνία και η

ανταλλαγή πληροφορίας στην σημερινή κοινωνία. Μέσω αυτών, τα άτομα έχουν την δυνατότητα να εκφράζονται και να διατηρούν επαφή με φίλους αλλά και να αναζητήσουν γνωριμία με καινούρια άτομα που θα έχουν τις ίδιες προσδοκίες ζωής με αυτά, κοινές αξίες και ενδιαφέροντα και οι πιθανότητες να βρουν ενδιαφέρουσες γνωριμίες είναι πολύ μεγάλες αφού τέτοιες εφαρμογές έχουν χιλιάδες καινούριους χρήστες κάθε μέρα και χρησιμοποιούνται από εκατομμύρια ανθρώπους σε καθημερινή βάση έχοντας πλέον καθιερωθεί σαν μέρος της καθημερινής ζωής.

Τί είναι όμως δικτυακός κοινωνικός ιστοχώρος; Ορίζουμε ως δικτυακό κοινωνικό ιστοχώρο μία σειρά υπηρεσιών βασισμένες στο διαδίκτυο οι οποίες επιτρέπουν σε ξεχωριστά άτομα:

1. Να δημιουργούν ένα δημόσιο λογαριασμό προφίλ μέσα σε ένα ορισμένο σύστημα.
2. Να δημιουργούν και να χειρίζονται λίστες άλλων χρηστών με τους οποίους έχουν συνάψει μία διαδικτυακή σύνδεση - σχέση.
3. Να μπορούν να δουν και να προσπελάσουν άλλες λίστες επαφών που έχουν δημιουργήσει άλλοι χρήστες μέσα στο σύστημα. Η φύση και ο ορισμός αυτών των συνδέσεων μπορούν να διαφέρουν από εφαρμογή σε εφαρμογή.

Αυτό που κάνει τους δικτυακούς κοινωνικούς ιστοχώρους μοναδικούς δεν είναι το ότι επιτρέπουν σε άτομα να γνωρίζονται με αγνώστους αλλά το ότι δίνεται η δυνατότητα στους χρήστες να χειρίζονται και να κάνουν ορατά τα κοινωνικά τους δίκτυα. Αυτό έχει ως αποτέλεσμα στο να δημιουργηθούν σχέσεις ανάμεσα σε άτομα που υπό κανονικές συνθήκες δεν θα μπορούσαν να δημιουργηθούν αφού δεν θα μπορούσαν ποτέ να υπάρξουν οι κατάλληλες συνθήκες, αν και σε πολλές περιπτώσεις τα μέλη που χρησιμοποιούν τέτοιου είδους εφαρμογές δεν αποσκοπούν στην γνωριμία με άλλα άτομα αλλά στην διατήρηση των επαφών με αυτά που ήδη γνωρίζουν.

1.5 Χαρακτηριστικά των Διαδικτυακών Κοινωνικών Ιστοχώρων

Ενώ τα SNSs παρέχουν μία μεγάλη ποικιλία τεχνικών χαρακτηριστικών που ενσωματώθηκαν όλα σε ένα πακέτο, τρία είναι αυτά που τα έχουν κάνει να ξεχωρίζουν και τους δώσανε τη λειτουργικότητα και την ευελιξία ώστε να γνωρίσουν την επιτυχία που έχουν σήμερα. Αυτά είναι τα:

1. Προφίλ των χρηστών – μελών.
2. Οι λίστες φίλων.
3. Τα μηνύματα μεταξύ των χρηστών – μελών.

Μία συνοπτική παρουσίασή αυτών γίνεται στη συνέχεια.

Μετά τη σύνδεσή τους με ένα τέτοιο σύστημα, θα ζητηθεί από τα άτομα αυτά να δημιουργήσουν ένα λογαριασμό που να αντιπροσωπεύει ψηφιακά τον εαυτό τους και το πως θέλουν αυτός να παρουσιάζεται στους υπόλοιπους χρήστες. Χρησιμοποιώντας κείμενο, εικόνες, ήχο, βίντεο και συνδέσμους, ο χρήστης καλείται να συμπληρώσει κάποιες φόρμες που περιέχουν μία σειρά από ερωτήσεις καθώς και άλλου τύπου διαδραστικές εφαρμογές και βάση των απαντήσεων που θα δοθούν δημιουργείται ένα προφίλ (profile) το οποίο είναι μία μοναδική σελίδα που δηλώνει πώς οι ίδιοι αντιλαμβάνονται τον εαυτό τους. Έτσι, μόνο με τη δημιουργία ενός profile δίνεται η δυνατότητα γνωριμίας και επικοινωνίας με χιλιάδες άλλα άτομα. Όμως για την αποφυγή ανεπιθύμητων ενεργειών από άλλα μέλη κάποιος χρήστης μπορεί να επιλέξει αν θέλει το προφίλ του να είναι ορατό από όλους τους χρήστες της εφαρμογής ή μόνο από αυτούς που ο ίδιος θα επιλέξει.

Αν όμως η δημιουργία προφίλ ήταν η μόνη λειτουργία που παρείχε η εφαρμογή τότε αυτή δε θα είχε κάποια ιδιαίτερη διαφορά από τις στατικές προσωπικές ιστοσελίδες που υπάρχουν στο διαδίκτυο. Αντιθέτως, αυτά τα προφίλ των μελών δημιουργούν έναν κορμό από μία αρθρωτή Λίστα φίλων οι οποίοι είναι επίσης χρήστες του συστήματος. Αυτό σημαίνει ότι κάθε χρήστης μπορεί να σημειώσει κάποιον άλλον ως φίλο και αν το άλλο άτομο συμφωνήσει τότε δημιουργείται μία διαδικτυακή σχέση μεταξύ τους ώστε κάθε ένας να μπορεί να αλληλεπιδράσει με τον άλλον και επίσης να έχει πρόσβαση στα ιδιωτικά στοιχεία του άλλου. Αυτό σημαίνει ότι κάποιο μέλος – φίλος μπορεί να δει τα μέλη – φίλους κάποιου άλλου χρήστη αφού προηγουμένως υπάρχει αμοιβαία αποδοχή του αιτήματος φιλίας. Έτσι, όλα τα προφίλ μπορούν να γίνουν προσπελάσιμα από τους άλλους μέσα από ένα τεράστιο δίκτυο που λέγεται Λίστες φίλων (Friend lists).

Η δυνατότητα περιήγησης από φίλο σε φίλο και η επικοινωνία με κάθε έναν που έχει ορατό προφίλ είναι το κύριο πρότυπο για την επιλογή νέων φίλων. Θα μπορούσε κάποιος να το παραλληλίσει με το σύστημα της πυραμίδας. Φυσικά η αναζήτηση με βάση κάποια κριτήρια δίνει την δυνατότητα σε κάποιον να βρει χιλιάδες άτομα τα οποία είναι και πιθανοί φίλοι.

Εδώ αξίζει να παρατηρήσουμε ότι η Λίστα Φίλων δεν απαρτίζεται πάντα από κάποιους γνωστούς ή πραγματικούς φίλους του χρήστη που τον συνδέουν με τους υπόλοιπους στενοί δεσμοί. Αντίθετα, αυτό το χαρακτηριστικό επιτρέπει κάθε μέλος αυτής της εικονικής κοινωνίας να δημιουργήσει ένα επιθυμητό περιβάλλον από γνωστούς ώστε να δείχνει πιο κοινωνικός. Όμως παρόλο που υπάρχουν εκατομμύρια χρήστες σε τέτοιους ιστοχώρους και η λίστα φίλων του καθενός μπορεί να απαρτίζεται από χιλιάδες άλλα μέλη, στην πραγματικότητα ο κάθε χρήστης ενδιαφέρεται να έχει σχέσεις – ή τουλάχιστον – στενές σχέσεις, με ελάχιστους από αυτούς.

Τα προφίλ και οι λίστες φίλων είναι δύο από τα τρία κύρια χαρακτηριστικά των κοινωνικών ιστοχώρων. Το τρίτο είναι τα σχόλια και τα μηνύματα μεταξύ των χρηστών. Αυτό το χαρακτηριστικό επιτρέπει την αποστολή μηνυμάτων ανάμεσα σε χρήστες. Πρέπει όμως να σημειωθεί ότι η ανάρτηση προσωπικών πληροφοριών στο προφίλ του κάθε χρήστη είναι και η πιο αμφιλεγόμενη λειτουργία από πλευράς ηθικής αφού πλέον ολόκληρη η προσωπική ζωή κάποιου μπορεί να γίνει θέα του διαδικτυακού κοινωνικού του περίγυρου. Αυτό όμως είναι προσωπική επιλογή του κάθε ατόμου και πρέπει να γίνεται έχοντας λάβει υπόψη τους κινδύνους από μία τέτοια διαρροή προσωπικών δεδομένων.

Αυτά είναι τα τρία χαρακτηριστικά που αποτελούν την κύρια δομή των κοινωνικών ιστοχώρων αν και υπάρχουν πολλά άλλα χαρακτηριστικά που μπορούν να βρεθούν κατά την περιήγηση σε τέτοια δίκτυα.

1.6 Κυριότεροι διαδικτυακοί κοινωνικοί ιστοχώροι

Παρακάτω γίνεται συνοπτική παρουσίαση μερικών από τους κυριότερους κοινωνικούς ιστοχώρους από την αρχή της εμφάνισής τους μέχρι σήμερα.

1.6.1 Classmates

Το Classmates είναι ένας κοινωνικός ιστοτόπος για την σύνδεση και επαφή μεταξύ φίλων και γνωστών από το νηπιαγωγίο μέχρι το πανεπιστήμιο. Με δωρεάν εγγραφή και δημιουργία λογαριασμού οποιοσδήποτε μπορεί να αναζητήσει παλιούς συμμαθητές και με ένα μικρό χρηματικό ποσό μπορεί να επικοινωνήσει με παλιούς φίλους, να γίνει μέλος σε κοινότητες και να κάνει chat σε online φόρουμ.

1.6.2 Friendster

Το Friendster έχει πολλά παρόμοια χαρακτηριστικά και εστιάζει στην προσέγγιση οποιουδήποτε και όχι απαραιτήτως παλιών συμμαθητών. Τα άτομα που χρησιμοποιούν αυτήν την ιστοσελίδα κατά κύριο λόγο στέλνουν μηνύματα ο ένας στον άλλον, προσκαλούν άλλους και κάνουν χρήση μπλοκ για να κρατήσουν επαφή.

1.6.3 Facebook

Το Facebook είναι παρόμοιο με το Friendster και ξεκίνησε τον Φεβρουάριο του 2004 στοχεύοντας στην πανεπιστημιακή κοινότητα και συγκεκριμένα άρχισε σαν τρόπος ενδοπανεπιστημιακής επικοινωνίας μεταξύ συμφοιτητών του Χάρβαρντ. Γρήγορα όμως εξελίχτηκε με αποτέλεσμα να είναι σήμερα φορέας επικοινωνίας για πάνω από 70 εκατομμύρια άτομα τα οποία είναι μέλη ενός ή και περισσότερων δικτύων του. Αξίζει να σημειωθεί ένα από τα κυριότερα χαρακτηριστικά του Facebook είναι οι διαδραστικές εφαρμογές που μπορεί να δημιουργήσει κάθε χρήστης οι οποίες είναι στη διάθεση και άλλων χρηστών.

1.6.4 MySpace

Το MySpace.com είναι ένας κοινωνικός ιστοτόπος ο οποίος ξεκίνησε το 2003 με σκοπό την προώθηση της ανεξάρτητης μουσικής και την κοινωνικοποίηση της μουσικής σκηνής παγκοσμίως. Εκτός από το ότι υπάρχει η δυνατότητα να ακούσει κάποιος καινούρια μουσική και να μάθει πληροφορίες σχετικά με events που έγιναν ή θα γίνουν, το MySpace είναι και ένας καλλιτεχνικός χώρος όπου οι χρήστες μπορούν να δημιουργήσουν λογαριασμό και να μοιραστούν βίντεο με φίλους.

1.7 Επίλογος

Οι κοινωνικοί ιστοχώροι έχουν γίνει πλέον αναπόσπαστο μέρος της καθημερινότητας εκατομμυρίων ανθρώπων και ήδη δείχνουν μία σημαντική αλλαγή του τρόπου κοινωνικοποίησης των ατόμων. Πριν όμως συμβεί αυτή η αλλαγή, είχε συντελεστεί η αλλαγή χρήσης του Web κυρίως από τους ανθρώπους που προγραμμάτιζαν εφαρμογές σε αυτό. Μάλιστα, είχε αλλάξει το ίδιο το Web. Από Web 1.0 εξελίχθηκε σε Web 2.0.

ΚΕΦΑΛΑΙΟ 2ο

WEB 2.0

2.1 Εισαγωγή

Το Web 2.0 ή συμμετοχικό διαδίκτυο όπως μεταφράζεται στα ελληνικά, είναι ένας όρος ο οποίος περιγράφει την αλλαγή χρήσης της τεχνολογίας WWW καθώς και της κατασκευής ιστοσελίδων και έχει ως στόχο να ενσωματώσει την δημιουργικότητα, την διασφάλιση κατανομής πληροφοριών, την συνεργασία μεταξύ των χρηστών καθώς και την λειτουργικότητα του διαδικτύου. Η σύλληψη του Web 2.0 έχει οδηγήσει στην ανάπτυξη και την επανάσταση των διαδικτυακών κοινοτήτων και των υπηρεσιών που αυτές παρέχουν όπως είναι οι διαδικτυακοί κοινωνικοί ιστοχώροι, οι ιστοσελίδες διαμοίρασης βίντεο, οι online εγκυκλοπαίδειες και τα blog. Αν και ο όρος προτείνει μία καινούρια έκδοση του WWW, δεν αναφέρεται σε καμία αναβάθμιση της υπάρχουσας τεχνολογίας αλλά στην αλλαγή των τρόπων που οι κατασκευαστές λογισμικού και οι χρήστες χρησιμοποιούν το διαδίκτυο. Αναφέρεται στις υπηρεσίες δεύτερης γενιάς οι οποίες είναι χτισμένες πάνω σε διαδικτυακές κοινότητες και πιο συγκεκριμένα στις πληροφορίες και το υλικό που παρέχουν οι χρήστες που κάνουν χρήση τέτοιων κοινοτήτων.

2.2 Χρησιμότητα του Web 2.0

Το Web 2.0 ενσωματώνει την ιδέα της αλληλεπίδρασης μεταξύ του χρηστών μεταξύ τους αλλά και με το περιεχόμενο των εφαρμογών. Χαρακτηριστικό είναι ότι οι συμμετέχοντες στο συμμετοχικό διαδίκτυο δεν είναι στατικοί παρατηρητές αλλά συνεισφέρουν στο περιεχόμενο της ιστοσελίδας με δυναμικό τρόπο κάνοντας χρήση της γνώσης τους αλλά και του κοινωνικού κεφαλαίου που εκφράζει την αξία των ανθρωπίνων σχέσεων στα διάφορα δίκτυα. Οι ιστοσελίδες που κάνουν χρήση του Web 2.0 επιτρέπουν στους χρήστες να κάνουν πολλά περισσότερα από την απλή ανάκτηση πληροφοριών. Μπορούν να χτίσουν πάνω στην αλληλεπίδραση που τους προσφέρει το Web 1.0 ώστε να δίνουν τη δυνατότητα στους χρήστες να εκτελούν εφαρμογές λογισμικού εξολοκλήρου μέσω του φυλλομετρητή (browser). Οι χρήστες επίσης μπορούν να έχουν τα δικά τους δεδομένα σε μία ιστοσελίδα βασισμένη στο Web 2.0 και να έχουν τον έλεγχο πάνω σε αυτά. Οι ιστοσελίδες αυτές προτρέπουν τους χρήστες να προσθέτουν

αξία στην εφαρμογή καθώς την χρησιμοποιούν. Αυτό έρχεται σε αντίθεση με τις παλιές παραδοσιακές ιστοσελίδες οι οποίες περιόριζαν τους επισκέπτες στην απλή προβολή και ανάγνωση του περιεχομένου τους, το οποίο μόνο ο ιδιοκτήτης της ιστοσελίδας μπορούσε να τροποποιήσει. Αντιθέτως, οι χρήστες αντιμετωπίζονται ως συνδημιουργοί της εκάστοτε ιστοσελίδας.

2.3 Κύριες αρχές του Web 2.0

Το διαδίκτυο λειτουργεί ως πλατφόρμα, με τα δυναμικά δεδομένα να είναι η κινητήρια δύναμη των ιστοσελίδων και υπάρχει η απαίτηση προς τους χρήστες να υιοθετήσουν μία διαφορετική συμπεριφορά και χρήση του διαδικτύου από την μέχρι πρότινος. Τα αποτελέσματα του διαδικτύου παράγονται από την συμμετοχή των χρηστών ενώ η καινοτομία στην δημιουργία συστημάτων και ιστοσελίδων επαφίεται στην συγκέντρωση χαρακτηριστικών από απομακρυσμένους ανεξάρτητους δημιουργούς (ένα είδος ανάπτυξης όπως αυτό του ανοικτού κώδικα).

Το τελευταίο ενισχύει το στοιχείο της δημοκρατίας ανάμεσα στους χρήστες αφού κάθε ένας έχει το δικαίωμα και τη δυνατότητα να συμβάλλει στην ανάπτυξη της ιστοσελίδας.

Τα κύρια χαρακτηριστικά του Web 2.0 είναι:

- Πλούσια εμπειρία χρήστη
- Συμμετοχή του χρήστη
- Δυναμικό περιεχόμενο
- Μεταδεδομένα
- Web standards
- Εξελισιμότητα
- Συλλογική νοημοσύνη

2.4 Τεχνολογία του συμμετοχικού διαδικτύου

Από τεχνολογικής άποψης οι εφαρμογές Web 2.0 λειτουργούν με έναν απλό τρόπο και σε ένα φιλικό προς τον χρήστη περιβάλλον χωρίς να υπάρχει η ανάγκη για κάποιες εξεζητημένες τεχνικές προγραμματισμού, ώστε ο κάθε χρήστης να μπορεί να δημιουργήσει κάποιο περιεχόμενο με αυτές τις εφαρμογές και να το δημοσιεύσει στο WWW. Η σημασία αυτού του γεγονότος είναι ότι το περιεχόμενο

των ιστοσελίδων αυτών είναι πιο σημαντικό από την τεχνολογία που χρησιμοποιείται.

Πριν την εμφάνιση του Web 2.0, ένα από τα χαρακτηριστικά των υπηρεσιών που παρέχονταν ήταν ότι το υλικό που οι χρήστες μπορούσαν να διαβάσουν, ακούσουν ή να δουν, ήταν απόρροια δημιουργίας λίγων μόνο δημιουργών. Σε αντίθεση με αυτό, η ουσία του Web 2.0 είναι ότι η πληροφορία δημιουργείται και μοιράζεται από τους ίδιους τους χρήστες. Ένα καλό παράδειγμα είναι η Wikipedia, η οποία είναι μία ανοιχτή ηλεκτρονική εγκυκλοπαίδεια στην οποία ο καθένας έχει τη δυνατότητα να προσθέσει κάποιο υλικό.

Οι ιστοσελίδες βασισμένες στο Web 2.0 περιέχουν συνήθως τα ακόλουθα χαρακτηριστικά / τεχνικές:

- Αναζήτηση (Search): Η διευκόλυνση εύρεσης πληροφορίας μέσω αναζήτησης με λέξεις κλειδιά.
- Σύνδεσμοι (Links): Οδηγοί που δείχνουν σε σημαντικές πληροφορίες. Οι καλύτερες σελίδες είναι αυτές που δείχνουν οι περισσότεροι σύνδεσμοι σε αυτές.
- Δημιουργία (Authoring): Η δυνατότητα δημιουργίας ενός συνεχώς ενημερωμένου περιεχομένου.
- Ετικέτες (Tags): Η κατηγοριοποίηση του περιεχομένου με τη δημιουργία ετικετών οι οποίες είναι απλές μονολεκτικές περιγραφές που διευκολύνουν την αναζήτηση και αποτρέπουν την στατικότητα των προκατασκευασμένων κατηγοριών.
- Επεκτάσεις (Extensions): Ο αυτοματισμός εργασιών και προτύπων με τη χρήση αλγορίθμων.
- Σήματα (Signals): Η χρήση τεχνολογίας RSS (Really Simple Syndication) για την ενημέρωση των χρηστών σχετικά με τις αλλαγές του περιεχομένου της ιστοσελίδας.

Ακολουθεί ο πίνακας 2.1 όπου παρουσιάζονται μερικές Web 2.0 εφαρμογές σε αντιπαράθεση με τις αντίστοιχες Web 1.0.

Πίνακας Ι: Σύγκριση Web 1.0 και Web 2.0

Web 1.0	**Web 2.0**
DoubleClick	Google AdSense
Ofoto	Flickr
Akamai	BitTorrent
mp3.com	Napster
Britanica Online	Wikipedia
Personal websites	Blogging
Evite	upcoming.org, EVDB
Domain-name speculation	Search-optimization
Page views	Cost per click
Screen scraping	Web services
Publishing	Participation
Content management systems	Wikis
Directories (taxonomy)	Tagginng (“folksonomy”)
Stickiness	Syndication

2.5 Ανάπτυξη Web 2.0 ιστοσελίδων

Η μορφή Web 2.0 ιστοσελίδων βασίζεται στην κύρια ιδέα του να παρέχεις περισσότερα με λιγότερα (deliver more with less). Αυτό βοηθά τους σχεδιαστές να στοχεύουν ακριβώς στα απαιτούμενα χαρακτηριστικά της ιστοσελίδας ώστε οι επισκέπτες να την βρίσκουν λειτουργική και επαρκής μέσα από σωστά επιλεγόμενα οπτικά στοιχεία. Αυτό σημαίνει λιγότερες λέξεις αλλά πιο περιεκτικές έννοιες καθώς και την απόρριψη της ιδέας ότι δε μπορούμε να μαντέψουμε τι θέλουν οι χρήστες από την κάθε ιστοσελίδα.

Τα κύρια χαρακτηριστικά της μορφής των Web 2.0 ιστοσελίδων είναι:

1. Απλότητα: Η σχεδίαση ιστοσελίδων είναι πιο απλή από ποτέ με την χρήση όσο γίνεται λιγότερο χαρακτηριστικών για την επίτευξη των τεθιμένων απαιτήσεων με την αφαίρεση κάθε άχρηστου στοιχείου.
2. Κεντρικό layout: Η χρήση ενός κύριου κεντρικού layout δημιουργεί την αίσθηση της αμεσότητας και της γρήγορης παράθεσης πληροφορίας.
3. Χρήση λιγότερων στηλών: Πριν λίγα χρόνια η χρήση τριών στηλών για την κατασκευή ιστοσελίδων ήταν ο κανόνας ενώ δεν ήταν λίγες οι φορές που

χρησιμοποιούνταν και τέσσερις στήλες. Σήμερα, η χρήση δύο στηλών είναι ο κανόνας και οι τρεις στήλες είναι το κάτι παραπάνω από αρκετό. Αυτό έχει ως αποτέλεσμα την απλότητα και την αμεσότητα.

4. Διαφοροποίηση των πάνω τμημάτων: Η παραδοχή αυτή ενισχύει την ιδέα της διαφοροποίησης του πάνω τμήματος της ιστοσελίδας με το υπόλοιπο περιεχόμενο.
5. Ξεχωριστές ενότητες: Η διαφοροποίηση της κάθε ενότητας από όλες τις υπόλοιπες μέσα στην ίδια σελίδα είναι πολύ σημαντική και πρέπει να γίνεται με τον άμεσο οπτικό διαχωρισμό. Αυτό επιτυγχάνεται κυρίως με τη χρήση ξεχωριστών χρωμάτων για κάθε ενότητα.
6. Απλή πλοήγηση: Είναι πολύ σημαντικό να υπάρχει μία μόνιμη μέθοδος πλοήγησης στην ιστοσελίδα που να είναι εμφανής και εύκολα επιλέξιμη σε κάθε σελίδα. Το Web 2.0 design κάνει την ολική πλοήγηση (global navigation) εμφανή καθώς και τους συνδέσμους μέσα στο κείμενο να διαφέρουν από το υπόλοιπο σώμα κειμένου.
7. Έντονα logo: Η χρήση μεγάλων έντονων logo κάνουν πιο αποτελεσματική την απόδοση αυτής της λειτουργίας.
8. Μεγαλύτερη γραμματοσειρά κειμένου: Αυτό έχει άμεση συνάφεια με την απλότητα και την απόρριψη οποιασδήποτε περιττής λειτουργίας αφού όσο περισσότερος χώρος υπάρχει τόσο περισσότερο εμφανή μπορούν να γίνουν τα στοιχεία που χρειάζονται να τονιστούν.
9. Εμπλουτισμός διεπιφάνειας: Η χρήση έντονων αλλά σωστά επιλεγμένων χρωμάτων, τα όμορφα γραφικά, η χρήση Flash κ.ά. είναι πολύ σημαντικά στοιχεία για την προβολή της ιστοσελίδας τα οποία όταν χρησιμοποιηθούν κατάλληλα την κάνουν πιο ελκυστική στους επισκέπτες.

2.6 Web 2.0 και Online Communities

Με την διάδοση της τεχνολογίας του Internet, εμφανίστηκαν πολλές νέες ευκαιρίες για επικοινωνία και απόκτηση πληροφορίας. Καινούρια είδη λογισμικού βρίσκονται στην διάθεση των χρηστών, με τα οποία μπορούν να εκφράσουν τη γνώμη τους, να συνεργαστούν με άλλους, να συζητήσουν ένα πρόβλημα ή και να ασκήσουν κριτική για κάτι ή σε κάποιον. Μέσα από αυτούς τους μηχανισμούς διαλόγου, τα λογισμικά αυτά προωθούν την δημιουργία και την λειτουργία συνεργαζόμενων κοινωνικών δικτύων. Σήμερα, τα αρχικά του World Wide Web

“WWW” θα μπορούσαν επίσης να σημαίνουν World Wide World, όσον αφορά το email και τους άλλους μηχανισμούς για την αποστολή άμεσων μηνυμάτων. Οι εφαρμογές του Web 2.0 μπορούν εύκολα να προωθήσουν τη συνεργασία μεταξύ ανθρώπων καθώς και την δημιουργία online κοινοτήτων πολύ πιο εύκολα από ποτέ.

Η έκφραση “κοινωνικό λογισμικό” πρωτοεμφανίστηκε στις αρχές του 1990, αλλά μόνο τώρα έχει αποκτήσει το πραγματικό της νόημα. Ακόμα κι αν δεν έχει έναν γενικώς αποδεκτό ορισμό, με τον όρο κοινωνικό λογισμικό εννοείτα το λογισμικό το οποίο ενισχύει ή εμπεριέχει την συνεργασία, την οργάνωση και την διαχείριση κοινοτήτων, την έκφραση του ατόμου, την κοινωνικοποίηση καθώς και την αλληλεπίδραση και το feedback των συμμετεχόντων. Διασφαλίζει αυτές τις δυνατότητες σε μία οριζόντια δομή όπου δεν υπάρχει κανένα πλαίσιο οργανισμού, δεν υπάρχουν σχέσεις βασισμένες στην ανωτερότητα και την κατωτερότητα ή στην επιβολή ελέγχου. Άλλο σημαντικό στοιχείο των κοινωνικών λογισμικών είναι η δυνατότητα δομημένης ανταλλαγής απόψεων μεταξύ των ατόμων, με έναν κεντροποιημένο ή αυτοοργανούμενο τρόπο.

Τα διαφορετικά κοινωνικά δίκτυα (Myspace, Facebook, Orkut), οι υπηρεσίες blog (LifeJournal, Xanga, Blogger) και οι ιστοσελίδες προβολής media (YouTube, Flickr) μπορούν όλα να θεωρηθούν ως κοινωνικό λογισμικό. Αυτές οι υπηρεσίες έχουν πολλά κοινά στοιχεία, όπως δημιουργία πληροφορίας από χρήστες και μεταβίβασή της σε άλλα άτομα, γραφική απεικόνιση των σχέσεων μεταξύ των χρηστών, παροχή δημόσιας επικοινωνίας μέσα από forum και συνεχής παρακολούθηση της συμπεριφοράς των χρηστών για λόγους διασφάλισης κοινωνικής ακεραιότητας των χρηστών.

Ακολουθεί ο πίνακας διαφόρων υπηρεσιών που χρησιμοποιούν μερικοί από τους διαδικτυακούς κοινωνικούς ιστοχώρους.

Πίνακας ΙΙ: Τύποι κοινωνικών λογισμικών

Software	Example
E-mail	Outlook, Sendmail, Pine, Hotmail
Weblog, Wiki	Movable Type, Blogger, Wikipedia
Messenger systems	ICQ, MSN, Trilliam
Document editing systems	Groove, Hydra, Lotus Notes
Group diaries	Livejournal
Introductory systems	MeetUp, Udate, Ryze
Systems for organising group discussions and exchange of views	SmartGroup, BBS, Usenet

2.7 Επίλογος

Πέρα από την αλλαγή χρήσης του παγκόσμιου ιστού ώστε αυτός να εξελιχθεί σε αυτό που ονομάζεται Web 2.0, οι τεχνολογίες που χρησιμοποιεί είναι οι ίδιες και βασίζονται στα ίδια πρότυπα και οδηγίες με το Web 1.0. Οι τεχνολογίες αυτές παρουσιάζονται στα παρακάτω κεφάλαια.

ΚΕΦΑΛΑΙΟ 3ο

HTML

3.1 Εισαγωγή

Ο βασικός πυρήνας των ιστοσελίδων που υπάρχουν στο Ίντερνετ είναι γραμμένος στην γλώσσα προγραμματισμού HTML (Hypertext Markup Language). Η HTML η οποία, όπως υποδηλώνει το όνομα της, αποτελεί μια markup γλώσσα για τηη περιγραφή hypertext κειμένων η οποία πρωτοεμφανίστηκε στις αρχές του 1990. Η γλώσσα αυτή χρησιμοποιείται στο WWW (World Wide Web) και αποτελεί υποσύνολο της γλώσσας SGML (Standard Generalized Markup Language) που επινοήθηκε από την IBM προκειμένου να λυθεί το πρόβλημα της μη τυποποιημένης εμφάνισης κειμένων σε διάφορα υπολογιστικά συστήματα.
Η γλώσσα αυτή έχει τυποποιηθεί από τον παγκόσμιο οργανισμό τυποποίησης τεχνολογιών του Web, το World Wide Web Consortium (W3C).

3.2 World Wide Web Consortium (W3C)

Το World Wide Web Consortium (W3C) είναι μια διεθνής κοινοπραξία όπου οι Οργανισμοί Μέλη, το προσωπικό πλήρους απασχόλησης και το κοινό δουλεύουν μαζί για να αναπτύξουν πρότυπα του Παγκοσμίου Ιστού. Η αποστολή του W3C είναι να οδηγήσει τον Παγκόσμιο Ιστό στο μέγιστο των δυνατοτήτων του, αναπτύσσοντας πρωτόκολλα και οδηγίες που εξασφαλίζουν μακροπρόθεσμη ανάπτυξη του Παγκοσμίου Ιστού.
Το W3C απαρτίζεται από οργανισμούς σε όλο τον κόσμο που ανήκουν σε ποικίλα πεδία και στόχος τους είναι να συμμετάσχουν σε έναν ουδέτερο τόπο συζήτησης για τη δημιουργία προτύπων του Παγκοσμίου Ιστού. Το W3C επιδιώκει την αποστολή του πρωταρχικά μέσα από τη δημιουργία προτύπων του Παγκοσμίου Ιστού και οδηγιών. Για να φτάσει ο Παγκόσμιος Ιστός στο μέγιστο των δυνατοτήτων του, οι πιο βασικές τεχνολογίες του Παγκοσμίου Ιστού θα πρέπει να είναι συμβατές μεταξύ τους και να επιτρέπουν στον εξοπλισμό (hardware) και στο λογισμικό που χρησιμοποιείται να έχουν πρόσβαση στον Παγκόσμιο Ιστό και να συνεργάζονται. Το W3C αναφέρεται σε αυτό το στόχο ως ”δια-λειτουργικότητα στον Παγκόσμιο Ιστό” (“Web interoperability”). Με την έκδοση ανοιχτών προτύπων για τις γλώσσες και τα πρωτόκολλα του Παγκοσμίου Ιστού, το W3C

επιδιώκει να αποφύγει κατακερματισμό της αγοράς και άρα τον κατακερματισμό του Παγκοσμίου Ιστού.

3.3 Στοιχεία της HTML

Όταν το 1989 ο Tim Berners Lee ανακάλυψε την Hypertext Markup Language, κανένας δε μπορούσε να αναλογιστεί την αλματώδη ανάπτυξη στην οποία θα οδηγούσε. Τα πρώτα χρόνια, η HTML χρησιμοποιούνταν μόνο για την δημιουργία στατικών ιστοσελίδων όπως επίσης και για την διάταξη των εγγράφων. Η HTML συνεχίζει να παραμένει, 20 χρόνια περίπου μετά την δημιουργία της, ιεραρχικά δομημένη και αποτελούμενη από ετικέτες.

Οι ετικέτες (tags) αυτές περικλείουν και διαφοροποιούν τα bit κειμένου, υποδεικνύοντας την λειτουργία και τον σκοπό του κειμένου που βρίσκεται ανάμεσα στις ετικέτες. Οι ετικέτες είναι γραμμένες απευθείας σε μορφή απλού κειμένου στο html έγγραφο όπου μπορούν να διερμηνευτούν από το λογισμικό του υπολογιστή. Οι ετικέτες αυτές καθ' εαυτές δεν παρουσιάζονται στον browser και είναι ξεχωριστές από το περιεχόμενο που περικλείουν.

Σημειώνεται ότι ανοίγουν με τη μορφή <tag> και κλείνουν με τη μορφή </tag>. Για παράδειγμα το   ανοίγει και κλείνει μία παράγραφο.

3.4 Χρήση HTML στην εφαρμογή

Παρακάτω παρουσιάζεται κάθε HTML ετικέτα που χρησιμοποιήθηκε στην εφαρμογή καθώς και η περίληψη των διαθέσιμων ιδιοτήτων της όπου αυτό απαιτείται.

3.4.1 Ετικέτα <html>

Το στοιχείο αυτό είναι το πρώτο το οποίο χρησιμοποιείται στο έγγραφο και περικλείει όλα τα υπόλοιπα μέσα σε αυτό. Επίσης, είναι γνωστό ως στοιχείο ρίζα (root element) και δεν έχει ιδιότητες και αυτό που κάνει είναι να καθορίζει που θα αρχίζει και που θα τελειώνει το έγγραφο.

3.4.2 Ετικέτα <head>

Η ετικέτα αυτή περιέχει πληροφορίες σχετικά με το HTML έγγραφο, συμπεριλαμβάνει τις λέξεις κλειδιά που περιγράφουν την ιστοσελίδα καθώς και συνδέσμους σε άλλα αρχεία τα οποία χρησιμοποιεί το έγγραφο όπως τα αρχεία

CSS και JavaScript. Τίποτα από ό,τι υπάρχει στο τμήμα του <head> δεν εμφανίζεται στον browser του χρήστη εκτός από τα περιεχόμενα της ετικέτας <title> όπου αυτά τυπώνονται στην αντίστοιχη μπάρα του browser.

3.4.3 Ετικέτα <base>

Η ετικέτα αυτή βοηθάει στη δημιουργία μίας URL διεύθυνσης η οποία θα είναι η ρίζα όλων των συνδέσμων στο έγγραφο.

3.4.4 Ετικέτα <link>

Η ετικέτα αυτή ορίζει τις σχέσεις μεταξύ δύο συνδεδεμένων εγγράφων. Η πιο κοινή του χρήση είναι στο να συνδέει εξωτερικά φύλλα στυλ (external style sheets) στο έγγραφο.

3.4.5 Ετικέτα <meta>

Η ετικέτα αυτή παρέχει πληροφορίες σχετικά με το έγγραφο. Οι μηχανές αναζήτησης συχνά χρησιμοποιούν αυτές τις πληροφορίες κατά τις αναζητήσεις στο Ίντερνετ. Με την χρήση αυτής της ετικέτας παρέχονται λέξεις κλειδιά και περιγραφές ώστε να ταξινομηθεί η τρέχουσα ιστοσελίδα κατά την αναζήτηση.

3.4.5 Ετικέτα <script>

Η ετικέτα <script> παίζει σημαντικό ρόλο στη δυναμικότητα της ιστοσελίδας. Επιτρέπει την προσθήκη γλωσσών script στην ιστοσελίδα όπως είναι η JavaScript.

3.4.6 Ετικέτα <style>

Η ετικέτα αυτή δημιουργεί ένα εσωτερικό στυλ (internal style) στο έγγραφο.

3.4.7 Ετικέτα <body>

Η ετικέτα <body> έρχεται αμέσως μετά την ετικέτα <head> και πρέπει να κλείσει ακριβώς πριν από την </html>. Το στοιχείο <body> είναι επιπέδου μπλοκ και μπορεί να περιέχει μόνο μπλοκ επιπέδου στοιχεία.

Τα στοιχεία επιπέδου μπλοκ καταλαμβάνουν όλο το διαθέσιμο πλάτος της ιστοσελίδας κάτι που σημαίνει ότι δύο στοιχεία επιπέδου μπλοκ θα εμφανιστούν

το ένα κάτω από το άλλο. Αντίθετα, τα ένθετα στοιχεία εμφανίζονται το ένα δίπλα στο άλλο.

Σημειώνεται ότι οποιοδήποτε κείμενο ή ένθετο στοιχείο πρέπει να βρίσκεται σε κάποιο μπλοκ επιπέδου στοιχείο και όχι απευθείας στο <body>.

3.4.8 Ετικέτα Κεφαλίδας

Η HTML έχει έξι επίπεδα κεφαλίδων από το 1 μέχρι το 6 (<h1>, <h2>, <h3>, <h4>, <h5>, <h6>) με το πρώτο επίπεδο να είναι το ανώτερο. Οι κεφαλίδες τυπικά εμφανίζονται με μεγαλύτερες ή και πιο έντονες γραμματοσειρές από ότι το κανονικό κείμενο. Η πρώτη κεφαλίδα σε κάθε κείμενο θα πρέπει να έχει το σημάδι <h1>.

3.4.9 Ετικέτα <p>

Η ετικέτα αυτή χρησιμοποιείται για τον διαχωρισμό των προτάσεων κειμένου κατά την εμφάνισή τους από τον browser. Οι παράγραφοι είναι στοιχεία επιπέδου μπλοκ και πρέπει να περιέχουν μόνο κείμενο και ένθετα στοιχεία.

3.4.10 Ετικέτες μορφής λίστας

Οι λίστες είναι μία συλλογή από δύο ή περισσότερα αντικείμενα του ίδιου τύπου. Υπάρχουν οι unordered lists (<ul>), οι oredered lists (<od>) και οι definition lists (<dl>).

3.4.10.1 Ετικέτα

Μία <ul> λίστα χρησιμοποιείται σε περιπτώσεις όπου η σειρά των στοιχείων μέσα σε αυτήν δεν έχει κάποια σημασία. Με την σειρά τους, κάθε αντικείμενο της λίστας ορίζεται από μία ετικέτα <li> που η αρχή και το τέλος της πρέπει να περικλείεται από την <ul>. Το στοιχείο <ul> είναι επιπέδου μπλοκ και μόνο στοιχεία <li> επιτρέπεται να υπάρχουν μέσα σε αυτήν. Από προεπιλογή οι unordered list εμφανίζονται από τους browser σε εσοχή και με μία βούλα να βρίσκεται στα αριστερά κάθε <li>.

3.4.10.2 Ετικέτα

Μία <ol> λίστα χρησιμοποιείται σε περιπτώσεις όπου η σειρά των στοιχείων μέσα σε αυτήν έχει σημασία. Με την σειρά τους, κάθε αντικείμενο της λίστας ορίζεται

από μία ετικέτα <li> που η αρχή και το τέλος της πρέπει να περικλείεται από την <ul>. Το στοιχείο <ol> είναι επιπέδου μπλοκ και μόνο στοιχεία <li> επιτρέπεται να υπάρχουν μέσα σε αυτήν. Από προεπιλογή οι ordered list εμφανίζονται από τους browser σε εσοχή και με έναν αύξων αριθμό να βρίσκεται στα αριστερά κάθε <li>.

3.4.10.3 Ετικέτα <dl>

Ένα <dl> στοιχείο δημιουργεί μία definition list. Αυτή είναι ένα στοιχείο επιπέδου μπλοκ το οποίο με τη σειρά του πρέπει να περιέχει τουλάχιστον έναν όρο <dt> και τουλάχιστον μία περιγραφή <dd>. Μόνο αυτά τα δύο στοιχεία επιτρέπονται ως παιδιά μέσα σε ένα στοιχεία <dl>. Το στοιχείο <dt>, το οποίο είναι επιπέδου μπλοκ, μπορεί να περιέχει μόνο κείμενο ή ένθετα στοιχεία μέσα σε αυτό και προσδιορίζει τον όρο ο οποίος πρόκειται να περιγραφεί ενώ το στοιχείο <dd> περιέχει την περιγραφή του <dt> στοιχείου, ακολουθεί ακριβώς μετά από το στοιχείο <dt>, είναι επιπέδου μπλοκ και μπορεί να περιέχει μόνο κείμενο ή ένθετα στοιχεία μέσα σε αυτό.

3.4.11 Ετικέτα

Η ετικέτα αυτή δίνει έμφαση στο κείμενο το οποίο εμπεριέχεται σε αυτήν και αυτό εμφανίζεται σε έντονη γραφή.

3.4.12 Ετικέτα

Η ετικέτα αυτή αναγκάζει τον browser να σταματήσει να τυπώνει στην τρέχουσα γραμμή και να συνεχίσει να τυπώνει στην αμέσως επόμενη. Το στοιχείο
 είναι ένα άδειο στοιχείο και αυτό σημαίνει ότι δεν έχει περιεχόμενο και κλείνει στην ίδια ετικέτα (
).

3.4.13 Ετικέτα

Η ετικέτα δημιουργεί μία λογική ενότητα στο έγγραφο η οποία ομαδοποιεί το συσχετιζόμενο με αυτήν περιεχόμενο. Είναι σημασιολογικά ουδέτερη και στην ουσία σημαίνει: “Ό,τι βρίσκεται μέσα σε αυτήν την περιοχή είναι μία ολότητα και είναι κάτι διαφορετικό από όλα τα υπόλοιπα στοιχεία.”.

Το στοιχείο είναι εξαιρετικά χρήσιμο για την οργάνωση του περιεχομένου σε μεγάλα μπλοκ ώστε να μπορεί έπειτα να μορφοποιηθεί πιο εύκολα με τη χρήση

CSS ή να χειριστεί με JavaScript ιδιαίτερα αν του έχει δοθεί μία ξεχωριστή ιδιότητα <id>.
Ένα είναι ένα στοιχείο επιπέδου μπλοκ, μπορεί να περιέχει κείμενο και οποιοδήποτε άλλο στοιχείο είτε αυτό είναι επιπέδου μπλοκ είτε ένθετο και το περιεχόμενό του ξεκινάει σε μία νέα γραμμή και καταλαμβάνει όλο το διαθέσιμο πλάτος.

3.4.14 Ετικέτα

Η ετικέτα αυτή είναι παρόμοια με την με τη διαφορά ότι αυτή είναι ένθετη και χρησιμοποιείτε για να διαχωρίσει τμήματα κειμένου ώστε αυτά να μορφοποιηθούν διαφορετικά με την χρήση CSS.

3.4.15 Ετικέτα <img>

Το ένθετο στοιχείο <img> χρησιμοποιείτε για να δηλώσει την εισαγωγή ενός αρχείου εικόνας στο HTML έγγραφο και απαιτεί τη χρήση της ιδιότητας src για να καθορίσει την θέση του αρχείου.
Επίσης, απαιτείται η ιδιότητα alt η οποία παρέχει ένα εναλλακτικό κείμενο το οποίο προσδιορίζει την εικόνα. Το κείμενο αυτό εμφανίζεται στην περίπτωση που η εικόνα δεν είναι διαθέσιμη και δεν είναι δυνατό να εμφανιστεί στον browser. Επίσης, χρησιμοποιείτε για να αναγνωστεί από συσκευές ανάγνωσης οθόνης (screen readers) σε άτομα με προβλήματα όρασης.
Σημαντικές είναι και οι ιδιότητες width και height. Η χρήση τους συνίσταται στο να δεσμεύουν το χώρο για την εικόνα που πρόκειται να εμφανιστεί κάνοντας τις διαστάσεις της γνωστές στον browser πριν αυτή φορτωθεί σε αυτόν. Έτσι αποφεύγονται φαινόμενα wrapping.
Αν το πλάτος και το ύψος δεν είναι ίδια με τις φυσικές διαστάσεις της εικόνας τότε ο browser θα εφαρμόσει κλίμακα σε αυτήν ώστε να χωρέσει στις τρέχουσες διαστάσεις. Σε τέτοιες περιπτώσεις όμως αν η εικόνα είναι μεγαλύτερη από τις δοθείσες διαστάσεις δε θα εμφανιστεί σωστά ενώ αν είναι μικρότερη τότε θα εμφανιστεί τραχιά και τα αρχεία που σχετίζονται με αυτήν θα κάνουν περισσότερη ώρα να εμφανιστούν.

3.4.16 Ετικέτα <a>

Η χρήση της ετικέτας <a> προκαλεί τον browser να μεταφερθεί σε μία διαφορετική τοποθεσία από την τρέχουσα όταν ο χρήστης κάνει κλικ πάνω στον σύνδεσμο που εμφανίζεται ανάμεσα στα <a> .

3.4.17 Ετικέτα <table>

Οι πίνακες χρησιμοποιούνται στις περιπτώσεις όπου υπάρχει ανάγκη για παρουσίαση δεδομένων όπου γίνονται πιο κατανοητά όταν εμφανίζονται σε σειρές και στήλες. Εδώ πρέπει να σημειωθεί ότι μέχρι πριν λίγα χρόνια οι πίνακες ήταν το κατεξοχήν εργαλείο στον σχεδιασμό και μορφοποίηση των ιστοσελίδων αλλά με την έλευση του CSS αυτό έχει περιοριστεί σε μεγάλο βαθμό. Ένας πίνακας έχει κεφαλές, όπου γίνεται η επεξήγηση της πληροφορίας των γραμμών και των στηλών καθώς και του κάθε κελιού. Οι ετικέτες <table> και </table> θα πρέπει να περιβάλουν τον ορισμό όλου του πίνακα. Το πρώτο στοιχείο εντός του πίνακα είναι η ετικέτα <caption> η χρήση του οποίου είναι προαιρετική. Έπειτα, με τη βοήθεια των ετικετών <tr> και </tr> μπορούν να οριστούν τόσες σειρές όσες απαιτούνται. Εντός μίας γραμμής μπορεί να οριστεί οποιοσδήποτε αριθμός κελιών που ορίζονται με τις ετικέτες <td> … </td> και <th> … </th>. Κάθε γραμμή ενός πίνακα μπορεί να μορφοποιηθεί ξεχωριστά και ανεξάρτητα από τις προηγούμενες και επόμενες γραμμές.

3.4.18 Ετικέτα <form>

Το στοιχείο <form> χρησιμοποιείται όπου απαιτείται η εισαγωγή δεδομένων από τον χρήστη. Όλη η λειτουργικότητα της φόρμας περικλείεται σε ένα και μόνο στοιχείο <form> το οποίο και στέλνει τα δεδομένα προς επεξεργασία.

Η φόρμα είναι στοιχείο επιπέδου μπλοκ και δεν μπορεί να έχει ένθετα περιεχόμενα. Στην περίπτωση που απαιτείται η εισαγωγή περισσοτέρων του ενός στοιχείου φόρμας σε ένα έγγραφο τότε κάθε ένα από αυτά πρέπει να είναι εντελώς ξεχωριστά – δεν επιτρέπεται να υπάρχει φόρμα μέσα σε μία άλλη φόρμα.

Το στοιχείο φόρμα απαιτεί την ιδιότητα action της οποίας η τιμή είναι μία διεύθυνση URL. Με αυτόν τον τρόπο τα δεδομένα μπορούν να χρησιμοποιηθούν είτε από την πλευρά του πελάτη με κάποια γλώσσα όπως η JavaScript είτε από την πλευρά του εξυπηρετητή με κάποια γλώσσα όπως οι PHP, Ruby, ASP ή ASP.NET.

Η ιδιότητα method είναι προαιρετική και μπορεί να δεχθεί μία από τις δύο τιμές get ή post για να υποδείξει τον τρόπο με τον οποίο θα υποβληθεί η φόρμα. Όταν η τιμή της ιδιότητας method είναι get, τα δεδομένα αποστέλλονται μέσω τις URL με τη μορφή string το οποίο απαρτίζεται από ζεύγη ονόματος / τιμής για κάθε δεδομένο. Όταν η τιμή της ιδιότητας method είναι post, τα δεδομένα αποστέλλονται απευθείας στον χειριστή φόρμας της εφαρμογής για επεξεργασία στον εξυπηρετητή.
Η μέθοδος get χρησιμοποιείται συνήθως όταν απαιτούνται στατικά δεδομένα από τον εξυπηρετητή για προσωρινή χρήση ή όταν αυτά θα χρησιμοποιηθούν ξανά σύντομα. Πρέπει να σημειωθεί ότι η χρήση της μεθόδου get δεν πρέπει να γίνεται για ευαίσθητα δεδομένα καθώς αυτά είναι ορατά και από τρίτους.
Η μέθοδος post χρησιμοποιείται συνήθως όταν τα δεδομένα που αποστέλλονται πρέπει να αποθηκευτούν για μελλοντική χρήση ή όταν για λόγους ασφαλείας τα δεδομένα δεν πρέπει να είναι εμφανή.
Η προεπιλεγμένη μέθοδος είναι η get η οποία και υπονοείται όταν η ιδιότητα method παραλείπεται και δεν δηλώνεται καθόλου.

3.4.19 Ετικέτα <input>

Πολλά στοιχεία τα οποία ανήκουν σε φόρμα μπορούν να δημιουργηθούν με το ένθετο στοιχείο <input> και κάθε τέτοιο ορίζεται με την αντίστοιχη ιδιότητα type. Επειδή το στοιχείο <input> είναι ένθετο, πολλά τέτοια στοιχεία μπορούν να εμφανιστούν στην ίδια σειρά μέχρις ότου να καταληφθεί όλο το διαθέσιμο πλάτος αλλά όλα πρέπει να ανήκουν σε έναν γονέα επιπέδου μπλοκ. Σημειώνεται επίσης ότι το στοιχείο <input> δεν έχει κείμενο δε μπορεί να περιέχει άλλα στοιχεία και πρέπει να κλείνει μόνο του (με />).
Σημαντικές ιδιότητες του <input> είναι οι checked=”checked”, disabled=”disabled”, maxlength και type. Η πρώτη, όταν είναι δηλωμένη θέτει την αρχική τιμή για τα checkbox και τα radio buttons. Η δεύτερη απενεργοποιεί το αντίστοιχο στοιχείο ώστε να μη μπορεί να τροποποιηθεί. Η τιμή ενός απενεργοποιημένου στοιχείου δεν αποστέλλεται για επεξεργασία. Η ιδιότητα maxlength καθορίζει το μέγιστο αριθμό χαρακτήρων που μπορούν να εισαχθούν σε ένα πεδίο κειμένου. Η ιδιότητα αυτή υπάρχει μόνο για στοιχεία κειμένου και κωδικού (input type=”text”, input type=”password”). Τέλος, η ιδιότητα type καθορίζει τον τύπο του αντίστοιχου στοιχείου της φόρμας. Ο προεπιλεγμένος τύπος είναι αυτός του κειμένου (text).

3.4.20 Ετικέτα <select>

Το ένθετο στοιχείο <select> δημιουργεί ένα πτυσσόμενο μενού επιλογών. Το μενού αυτό μπορεί να αναπτύσσεται είτε οριζόντια είτε κάθετα. Όταν αποστέλλεται η φόρμα τότε αποστέλλεται μόνο η τιμή που έχει επιλέξει ο χρήστης. Πρέπει να σημειωθεί ότι η ιδιότητα name είναι απαιτούμενη για το στοιχείο <select> έτσι ώστε να μπορεί να προσδιοριστεί. Τέλος, επειδή δε μπορεί να είναι άδειο, πρέπει να περιέχει τουλάχιστον μία τιμή.

3.4.21 Ετικέτα <option>

Κάθε επιλογή ενός στοιχείου <select> περιέχεται μέσα σε ένα στοιχείο <option>. Το στοιχείο αυτό δε μπορεί να είναι κενό αλλά μπορεί να περιέχει μόνο μία ετικέτα κειμένου (label). Μία επιλογή μπορεί να προεπιλεχτεί συμπεριλαμβάνοντας την ιδιότητα selected=”selected”. Αν δεν έχει οριστεί η ιδιότητα value τότε η τιμή που θα αποσταλεί κατά την αποστολή της φόρμας είναι το κείμενο της επιλογής.

3.4.22 Ετικέτα <textarea>

Το στοιχείο <textarea> δημιουργεί ένα πεδίο πολλών γραμμών σε περίπτωση που το κείμενο προς εισαγωγή είναι πολύ μεγάλο για να χωρέσει σε ένα πεδίο μίας γραμμής κειμένου. Το μέγεθός του καθορίζεται από τις υποχρεωτικές ιδιότητες rows και cols όπου η τιμή που αντιστοιχεί στην ιδιότητα rows είναι η οριζόντιες γραμμές ενώ η τιμή της cols αντιστοιχεί στον αριθμό των χαρακτήρων που μπορεί να έχει κάθε γραμμή. Επίσης υποχρεωτική είναι και η ιδιότητα name ώστε να είναι δυνατή η συσχέτιση η τιμή που θα σταλεί με το συγκεκριμένο στοιχείο κατά την αποστολή της φόρμας.

3.4.23 Ετικέτα <fieldset>

Η ετικέτα <fieldset> ομαδοποιεί σε μία λογική ενότητα όλα τα στοιχεία τα οποία εμπεριέχει. Επειδή μία φόρμα δε μπορεί να περιέχει ένθετα στοιχεία το στοιχείο <fieldset> χρησιμεύει στο να τα συμπεριλάβει αυτό.

3.4.24 Ετικέτα <legend>

Το στοιχείο <legend> παρέχει έναν τίτλο σε ένα πεδίο συνόλου (fieldset). Είναι ένθετο στοιχείο και μπορεί να περιέχει μόνο κείμενο και άλλα ένθετα στοιχεία.

Πρέπει να σημειωθεί ότι πολλοί κανόνες του CSS δε μπορούν να εφαρμοστούν ή εφαρμόζονται με προβλήματα στο στοιχείο <legend>.

3.4.25 Ετικέτα <label>

Το στοιχείο <label> είναι πολύ χρήσιμο κατά τον σχεδιασμό φορμών αφού δημιουργεί μία ετικέτα κειμένου για κάθε πεδίο που απαιτείται. Ένα τέτοιο στοιχείο μπορεί να περιέχει και το πεδίο ελέγχου και την ετικέτα κειμένου και σε μια τέτοια περίπτωση η σύνδεση ανάμεσα σε αυτά τα δύο υπονοείται από το περιεχόμενο. Εναλλακτικά, μπορεί να γίνει χρήση της ιδιότητας for στην οποία αποδίδεται το μοναδικό id του πεδίου και ως εκ τούτου ορίζεται η σχέση μεταξύ τους. Αξίζει να σημειωθεί ότι όταν μία ετικέτα είναι σωστά συσχετισμένη με το αντίστοιχο πεδίο της τότε οι περισσότεροι browser θα εστιάσουν στο συγκεκριμένο πεδίο μόλις γίνει κλικ πάνω στην ετικέτα. Αυτό αυξάνει την προσβασιμότητα και την χρηστικότητα των πεδίων και κυρίως των checkboxes και radio buttons τα οποία είναι αρκετά μικρά σε διαστάσεις.

3.5 Επίλογος

Η HTML έχει αλλάξει ελάχιστα από τότε που δημιουργήθηκε. Αντίθετα, αυτό που έχει αλλάξει είναι η μορφοποίηση των στοιχείων της HTML κάτι που σήμερα γίνεται με την χρήση του CSS. Θα μπορούσε λοιπόν να ειπωθεί ότι τα στοιχεία της γλώσσας HTML χρησιμοποιούνται για την κατασκευή της δομής των ιστοσελίδων όμως αυτό που δίνει μορφή στις ιστοσελίδες αυτές είναι το CSS. Έτσι, η HTML θα μπορούσε πολύ εύστοχα να χαρακτηριστεί ως τα πινέλα με τα οποία δημιουργείται μία ιστοσελίδα ενώ το CSS ως τα χρώματα με τα οποία αυτή χρωματίζεται.

ΚΕΦΑΛΑΙΟ 4ο

CSS

4.1 Εισαγωγή

Τα Φύλλα Διαμόρωσης Στυλ (Cascading Style Sheets – CSS) είναι μία γλώσσα διαμόρφωσης ιστοσελίδων και χρησιμοποιείται για να περιγράψει την παρουσίαση ενός εγγράφου το οποίο έχει γραφτεί σε κάποια από τις γλώσσες σήμανσης (markup languages). Χρησιμοποιείται κυρίως σε εφαρμογές γραμμένες σε HTML και XHTML αλλά μπορεί να χρησιμοποιηθεί επίσης και σε οποιοδήποτε έγγραφο της XML για τη μορφοποίησή του.

4.2 Διαχωρισμός HTML και εμφάνισης

Το CSS έχει σχεδιαστεί ώστε να επιτρέπει τον διαχωρισμό του περιεχομένου του εγγράφου, το οποίο συνήθως γράφεται σε HTML, και της παρουσίασής του, η οποία περιλαμβάνει στοιχεία όπως τα χρώματα, οι γραμματοσειρές και η μορφή του. Ο διαχωρισμός αυτός μπορεί να βελτιώσει σημαντικά την λειτουργικότητα αλλά και την προσβασιμότητα παρέχοντας μεγαλύτερη ευελιξία και έλεγχο στον καθορισμό των χαρακτηριστικών εκείνων που άπτονται της μορφολογίας του εγγράφου, επιτρέποντας παράλληλα σε πολλά έγγραφα να μοιράζονται την ίδια μορφοποίηση με αποτέλεσμα τη μείωση της πολυπλοκότητας και της επανάληψης κώδικα.

Επίσης, η χρήση CSS επιτρέπει την παρουσίαση του ίδιου περιεχομένου με διαφορετικούς τρόπους όπως είναι η εμφάνιση σε οθόνη, η εκτύπωση, η αφήγηση καθώς και σε συσκευές ανάγνωσης βασισμένες στο σύστημα Braille.

Σημειώνεται ότι κατά την συγγραφή αυτής της πτυχιακής το τρέχον CSS είναι το CSS3 και ότι αρκετά από τα χαρακτηριστικά του έτσι όπως αυτά καθορίζονται από το World Wide Consortium, δεν υλοποιούνται πλήρως από όλους τους browsers.

4.3 Δομή – Παρουσίαση – Συμπεριφορά

Τα τρία επίπεδα ενός σύγχρονου Web εγγράφου είναι αυτά της Δομής, της Παρουσίασης και της Συμπεριφοράς (Structure Layer, Presentation Layer, Behavior Layer).

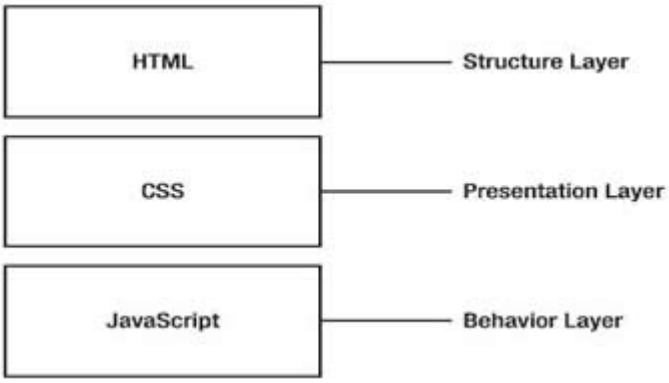
Το πρώτο επίπεδο, αυτό της δομής, περιέχει το περιεχόμενο του εγγράφου μαζί με τις σημασιολογικές πληροφορίες τα οποία υποδεικνύουν το ποιόν κάθε bit

κειμένου (π.χ. κεφαλίδα, παράγραφος, λίστα κτλ). Το επίπεδο της δομής ενός εγγράφου Web είναι συνήθως γραμμένο στη γλώσσα HTML.

Το επίπεδο της παρουσίασης περιγράφει τον τρόπο με τον οποίο το έγγραφο πρέπει να παρουσιαστεί στον επισκέπτη της ιστοσελίδας, συμπεριλαμβάνοντας πληροφορίες όπως τα χρώματα, οι γραμματοσειρές, τη μορφοποίηση των εικόνων κ.ά. Γενικά, το επίπεδο της παρουσίασης ενός εγγράφου Web είναι γραμμένο με τη χρήση CSS.

Τέλος, το επίπεδο συμπεριφοράς χρησιμοποιείται για την ανανέωση, πρόσθεση και αφαίρεση ενός Web εγγράφου σε συνάρτηση πάντα με τη συμπεριφορά του χρήστη. Το επίπεδο αυτό που άπτεται της διαχείρισης του Document Object Model (DOM), συνήθως είναι γραμμένο στην γλώσσα JavaScript.

Το Σχήμα Ι παρουσιάζει τα τρία αυτά επίπεδα.

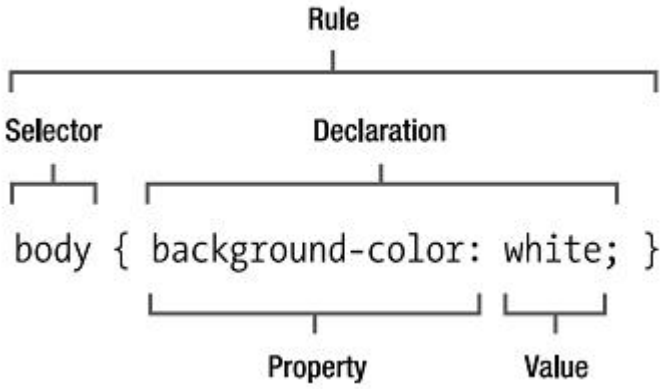


Σχήμα Ι: Τα επίπεδα ενός Web εγγράφου

4.4 Βασικές Αρχές του CSS

Το CSS επιτρέπει την προσθήκη στυλ στις ιστοσελίδες, ενσωματώνοντας και βελτιώνοντας την μορφή στο περιεχόμενο. Κάθε HTML στοιχείο προσδίδει κάποιο περιεχόμενο και οι ιδιότητες οι οποίες δηλώνονται με κάθε ένα από αυτά παρέχουν περισσότερη πληροφορία σε αυτά τα στοιχεία.

Το δομικό στοιχείο του CSS είναι ο κανόνας (rule) ο οποίος είναι ένα σύνολο από εντολές τις οποίες ο browser ακολουθεί ώστε να αλλάξει την προεπιλεγμένη παρουσίαση των HTML στοιχείων ανάλογα με τις τιμές που έχουν οριστεί. Ένας CSS κανόνας αποτελείται από τα στοιχεία που απεικονίζονται στο Σχήμα ΙΙ:



Σχήμα ΙΙ: Τα στοιχεία ενός CSS rule

Ο επιλογέας (selector) είναι το κομμάτι αυτό του κανόνα το οποίο συγκεκριμενοποιεί το στοιχείο το οποίο πρόκειται να μορφοποιηθεί. Η σκοπιά του μπορεί να είναι μεγάλου εύρους, επηρεάζοντας τη μορφή κάθε στοιχείου, ή ακόμα και πολύ μικρή και συγκεκριμένη, επηρεάζοντας για παράδειγμα μόνο ένα στοιχείο.

Κάθε δήλωση (declaration) αποτελείται από δύο επιμέρους στοιχεία τα οποία ακολουθούν συγκεκριμένη σειρά και είναι η ιδιότητα (property) και η τιμή (value).

Η ιδιότητα είναι το τμήμα εκείνο του στοιχείου το οποίο πρόκειται να μορφοποιηθεί, όπως είναι το χρώμα, το πλάτος, το ύψος, η γραμματοσειρά ή ακόμα και η θέση όπου πρόκειται να εμφανιστεί το στοιχείο.

Η τιμή της ιδιότητας (property value) προσδίδει το συγκεκριμένο στυλ το οποίο πρόκειται να εφαρμοστεί στο επιλεγμένο στοιχείο. Οι τιμές οι οποίες είναι αποδεκτές εξαρτώνται από την κάθε ιδιότητα, και μερικές ιδιότητες δέχονται πολλαπλές τιμές οι οποίες χωρίζονται με κόμμα.

Κάθε δήλωση γίνεται ανάμεσα σε ένα ζεύγος άγκιστρων ({ και }) και υπάρχει η δυνατότητα να εφαρμοστούν πολλές δηλώσεις σε ένα συγκεκριμένο selector η κάθε μία από τις οποίες μπορεί να μορφοποιεί ξεχωριστά σημεία κάθε στοιχείου ή ακόμα και όλα εξαρχής. Κάθε ιδιότητα χωρίζεται από την τιμή της με μία άνω κάτω τελεία (:) και η δήλωση τελειώνει με ένα ελληνικό ερωτηματικό (;). Το ερωτηματικό αυτό είναι σημαντικό για το διαχωρισμό κάθε δήλωσης αλλά αν υπάρχει μόνο μία δήλωση στον κανόνα ή στην περίπτωση της τελευταίας δήλωσης τότε το ερωτηματικό είναι προαιρετικό. Καλό είναι πάντως αυτά να τοποθετούνται κάθε φορά.

4.5 CSS Selectors

Ένας selector, όπως φανερώνει και το όνομά του, επιλέγει ένα στοιχείο στο HTML έγγραφο. Οι διαθέσιμοι selector είναι σχετικά λίγοι και κάθε ένας τους έχει διαφορετικό βαθμό διάκρισης των στοιχείων που τον ακολουθούν. Διάκριση είναι ο τρόπος μέτρησης της σκοπιάς ενός selector, δηλαδή πόσα στοιχεία θα επιλεγούν από αυτόν. Το CSS είναι σχεδιασμένο έτσι ώστε ο πιο συγκεκριμένος γίνεται ένας selector να υπερβαίνει τον πιο γενικό. Η δυνατότητα αυτή της διάκρισης είναι ένα από τα πιο ισχυρά στοιχεία του CSS. Παρακάτω παρατίθενται με περισσότερες λεπτομέρειες οι κανόνες αυτής της.

4.5.1 Παγκόσμιος Επιλογέας (Universal Selector)

Ο universal selector εκφράζεται με το σημείο του αστερίσκου (*) και λειτουργεί ως wild card για την επιλογή κάθε στοιχείου στο έγγραφο. Για παράδειγμα, ο κανόνας:

```
* { color: blue; }
```

θα προσδώσει ένα μπλε υπόβαθρο σε όλα τα στοιχεία. Επικεφαλίδες, παράγραφοι, λίστες, κελιά πίνακα ακόμα και σύνδεσμοι θα μορφοποιηθούν με μπλε χρώμα. Αυτός είναι και ο λιγότερο συγκεκριμένος selector αφού στην ουσία επιλέγει όλα τα στοιχεία.

4.5.2 Επιλογέας Στοιχείου (Element Selector)

Ένας element selector επιλέγει όλα τα στιγμιότυπα ενός στοιχείου τα οποία καθορίζονται από το όνομα της ετικέτας τους. Ο selector αυτός χαρακτηρίζεται από μεγαλύτερη διακριτότητα από ότι ο universal selector αλλά δεν είναι και πολύ συγκεκριμένος αφού κάθε επιλέγει κάθε στοιχείο στο έγγραφο που έχει το ίδιο όνομα ετικέτας, όσες φορές κι αν εμφανιστεί αυτό. Για παράδειγμα, ο κανόνας:

```
em { color: red; }
```

προσδίνει σε κάθε στοιχείο με την ετικέτα em στο ίδιο κόκκινο χρώμα υποβάθρου ακόμα κι αν υπάρχουν χιλιάδες τέτοια στοιχεία σε όλο το έγγραφο. Πρέπει να σημειωθεί ότι οι element selectors ονομάζονται και type selectors.

4.5.3 Επιλογέας Κλάσης (Class Selector)

Ένας class selector στοχεύει σε κάθε στοιχείο το οποίο έχει μία συγκεκριμένη κλάση η οποία του έχει δοθεί πιο πριν με την χρήση της ιδιότητας class. Αφού η ιδιότητα αυτή μπορεί να δοθεί σε πολλά στοιχεία ο selector αυτός δεν είναι πολύ συγκεκριμένος αλλά καθορίζει τη διακριτότητα σε μεγαλύτερο βαθμό από ότι ο element selector.

Στο CSS, οι class selectors αναγνωρίζονται από την τελεία (.) που προηγείται πριν από αυτούς. Για παράδειγμα, ο παρακάτω κανόνας θα μορφοποιήσει κάθε στοιχείο που ανήκει στην κλάση “info”:

```
.info { color: blue; }
```

4.5.4 Επιλογέας Ταυτότητας (Id Selector)

Ένας id selector επιλέγει μόνο το στοιχείο που φέρει το αντίστοιχο id. Συγκεκριμένα κάθε στοιχείο μπορεί να έχει κάποιο id αλλά μόνο ένα id του ίδιου ονόματος μπορεί να υπάρχει σε όλο το έγγραφο. Ο id selector στοχεύει σε ένα μόνο στοιχείο σε κάθε σελίδα και αυτό τον κάνει πολύ πιο συγκεκριμένο από τον class selector.

Στο CSS, οι id selectors αναγνωρίζονται από τον χαρακτήρα κάγκελο (#) που προηγείται πριν από αυτούς. Για παράδειγμα, ο παρακάτω κανόνας θα μορφοποιήσει το στοιχείο με id “introduction” και θα του δώσει ως χρώμα υποβάθρου το πράσινο:

```
#introduction { color: green; }
```

4.5.5 Επιλογέας Ψευδοκλάσης (Pseudo Class Selector)

Μία ψευδοκλάση είναι περίπου το ίδιο με τον επιλογέα κλάσης και έχει τον ίδιο βαθμό διακριτότητας με την κλάση αλλά επιλέγει μία συγκεκριμένη κατάσταση ενός στοιχείου. Πριν από μία ψευδοκλάση προηγείται μία άνω κάτω τελεία (:) και μόνο μερικές ψευδοκλάσεις είναι διαθέσιμες:

```
:link { color: blue; }
:visited {color: purple; }
:active { color: red; }
:hover { color: green; }
:focus { color: orange; }
```

Η ψευδοκλάση :link επιλέγει όλα τα στοιχεία τα οποία είναι σύνδεσμοι. Η ψευδοκλάση :visited επιλέγει τους συνδέσμους εκείνους οι οποίοι ο προορισμός που περιγράφουν έχει προηγουμένως επισκεφτεί. Η ψευδοκλάση :active επιλέγει εκείνους τους συνδέσμους οι οποίοι είναι σε ενεργή κατάσταση. Η ψευδοκλάση :hover επιλέγει όλα τα στοιχεία εκείνα στα οποία η θέση του mouse είναι από πάνω τους. Η ψευδοκλάση :focus επιλέγει κάθε στοιχείο σε κατάσταση προεπιλογής.

4.5.6 Ψευδοστοιχεία (Pseudo – Elements)

Σύμφωνα με τον ορισμό του W3C τα ψευδοστοιχεία δημιουργούν γενικεύσεις σχετικά με το έγγραφο πέρα από τον βαθμό που καθορίζονται από την εκάστοτε γλώσσα.

Τα ψευδοστοιχεία είναι πολύ ισχυρά στοιχεία του CSS και περιλαμβάνουν τα ακόλουθα:

```
:first-line
:first-letter
:before
:after
```

Η χρήση τους είναι παρόμοια με αυτήν των ψευδοκλάσεων αλλά τα ψευδοστοιχεία μπορούν να εφαρμοστούν μόνο στον τελευταίο απλό selector ενός συνδυασμένου selector.

Πρέπει να σημειωθεί ότι τα ψευδοστοιχεία :first-line και :first-letter υποστηρίζονται από τον IE 6 αλλά αυτό δεν ισχύει και για τα ψευδοστοιχεία :before και :after.

Παρακάτω αναλύουμε σε μεγαλύτερο βαθμό αυτά τα ισχυρά εργαλεία.

4.5.6.1 Ψευδοστοιχείο :first-line

Το ψευδοστοιχείο αυτό στοχεύει την πρώτη γραμμή ενός στοιχείου στο οποίο έχει οριστεί. Σημειώνεται ότι το στοιχείο στο οποίο έχει οριστεί το :first-line πρέπει να ανήκει σε έναν από τους τύπους block-level, inline-block, table-caption ή table-cell. Για παράδειγμα, ο παρακάτω κανόνας:

```
p:first-line { text-transform: uppercase; }
```

θα μετατρέψει αυτόματα όλα τα γράμματα της πρώτης γραμμής κάθε παραγράφου σε κεφαλαία.

4.5.6.2 Ψευδοστοιχείο :first-letter

Παρόμοια με το ψευδοστοιχείο :first-line, το :first-letter επιλέγει τον πρώτο αλφαριθμητικό χαρακτήρα του εκάστοτε καθορισμένου στοιχείου εφόσον δεν προηγείται περιεχόμενο άλλης μορφής (π.χ. κάποια εικόνα). Σημειώνεται ότι το στοιχείο στο οποίο έχει οριστεί το :first-line πρέπει να ανήκει σε έναν από τους τύπους block-level, inline-block, table-caption ή table-cell. Για παράδειγμα, υποθέτοντας την παρακάτω παράγραφο:

```
<p>We eat food.</p>
```

και εφαρμόζοντας τον κανόνα:

```
p:first-letter {
     font-size: 20px;
     color: blue;
}
```

το αποτέλεσμα θα είναι το πρώτο γράμμα της παραγράφου που στην προκειμένη περίπτωση είναι το W να μορφοποιηθεί στα 20px και με μπλε χρώμα.

4.5.6.3 Ψευδοστοιχεία :before και :after

Τα ψευδοστοιχεία αυτά δίνουν την δυνατότητα εισαγωγής περιεχομένου το οποίο παράγεται αυτόματα κάθε φορά που τα προαπαιτούμενα το επιτρέπουν. Σε αυτήν την περίπτωση η ιδιότητα content είναι απαραίτητη για να επιτευχθεί αυτό.

Για παράδειγμα, οι παρακάτω κανόνες:

```
p.note:before {
     content: "[";
}

p.note:after {
     content: "]";
}
```

θα έχουν ως αποτέλεσμα την εμφάνιση των αγκύλων "[" και "]" πριν και μετά το περιεχόμενο κάθε παραγράφου.

4.5.7 Επιλογέας απογόνου (Descendant Selector)

Ένας από τους πιο χρήσιμους και ισχυρούς selector είναι ο descendant selector ο οποίος συντελείται από δύο ή περισσότερους βασικούς selector (universal, element, class, id, pseudo class) χωρισμένους με κενά. Αυτό που κάνει είναι να επιλέγει τη θέση του συγκεκριμένου στοιχείου στο έγγραφο.
Για παράδειγμα, ο selector:

```
#introduction em { color: yellow; }
```

θα προσδώσει το κίτρινο χρώμα σε οποιοδήποτε στοιχείο με ετικέτα em το οποίο υπάρχει μέσα στο στοιχείο με το id "introduction".

Οι descendant selectors επιτρέπουν εξαιρετικά συγκεκριμένες επιλογές, όπως στο παρακάτω παράδειγμα:

```
#introduction .info p * { color: pink; }
```

που θα επιλέξει όλα τα στοιχεία τα οποία είναι απόγονοι ενός στοιχείου p το οποίο είναι απόγονος ενός στοιχείου που είναι της κλάσης info το οποίο είναι απόγονος ενός στοιχείου που έχει το id introduction.

4.5.8 Ομαδοποίηση Επιλογέων (Grouping Selectors)

Το CSS παρέχει τη δυνατότητα ομαδοποίησης των επιλογέων σε έναν κανόνα έτσι ώστε το ίδιο σύνολο δηλώσεων να μπορεί να εφαρμοστεί σε πλήθος

στοιχείων χωρίς να απαιτείται η επανάληψη συγγραφής του ίδιου κανόνα. Κάθε selector στον κανόνα χωρίζεται με ένα κόμμα. Για παράδειγμα:

```
p, h1, h2 { color: blue; }
```

Ο παραπάνω κανόνας εφαρμόζει το ίδιο χρώμα (μπλε) σε κάθε στιγμιότυπο στοιχείου p, h1 και h2. Ένας πιο σύνθετος κανόνας είναι ο παρακάτω:

```
p#introduction em, a.info:hover, h2.info { color: gold; }
```

ο οποίος θα εφαρμοστεί σε όλα τα στοιχεία em τα οποία είναι απόγονοι της παραγράφου με id introduction, σε όλους τους συνδέσμους οι οποίοι βρίσκονται σε κατάσταση hovered και ανήκουν στην κλάση info και τέλος σε όλα τα στοιχεία h2 τα οποία ανήκουν στην κλάση info. Παραπάνω έχει σημειωθεί ότι διαφορετικοί τύποι στοιχείων μπορούν να ανήκουν στην ίδια κλάση.
Η ομαδοποίηση και ο συνδυασμός selector είναι ένας πολύ καλός τρόπος για την συγγραφή περιεκτικών, σύντομων και συντηρήσιμων CSS εγγράφων.

4.5.9 Επιλογείς Παιδιών (Child Selectors)

Οι child selectors είναι παρόμοιοι με τους descendant selectors με τη διαφορά ότι αυτοί επιλέγουν παιδιά αντί για όλους τους απογόνους. Για να οριστεί ένας child selector γίνεται χρήση του συμβόλου μεγαλύτερο (>). Πρέπει να σημειωθεί ότι ο Microsoft Internet Explorer 6 και πιο κάτω δεν υποστηρίζει την χρήση child selector.
Αν υποτεθεί για παράδειγμα ο παρακάτω HTML κώδικας:

```
<ul><li>Item one</li>
     <li>Item two</li>
     <ol><li>Sub-item one</li>
          <li>Sub-item two</li>
     </ol>
</ul>
```

Και ο αντίστοιχος κώδικας CSS child selector:

```
ul > li { color: blue; }
```

το αποτέλεσμα θα είναι να μορφοποιηθούν με μπλε χρώμα τα στοιχεία με ετικέτες <li> και <ol> αφού μόνο αυτά είναι άμεσα παιδιά του στοιχεία με ετικέτα <ul>.

4.5.10 Επιλογείς Αδέρφια (Adjacent Sibling Selectors)

Οι adjacent sibling selectors επιτρέπουν την στόχευση ενός στοιχείου το οποίο έχει τον ίδιο γονέα και ακολουθεί αμέσως μετά από το στοιχείο το οποίο καθορίζεται στον κανόνα. Πρέπει να σημειωθεί ότι ο Microsoft Internet Explorer 6 και πιο κάτω δεν υποστηρίζει την χρήση sibling selectors.

```
<body>
     <h1>This is a header</h1>
     <p>This is a paragraph.</p>
     <p>This is another paragraph.</p>
</body>
```

Για να γίνει καλύτερα κατανοητή η χρήση τέτοιων selectors υποθέτουμε τον παραπάνω HTML κώδικα.

Οι παράγραφοι έχουν προεπιλογής περιθώριο 1 em πάνω και κάτω. Με τον παρακάτω CSS rule το περιθώριο αυτό γίνεται 0:

```
h1 + p { margin-top: 0;
         margin-botom: 0;
}
```

4.5.11 Επιλογείς Ιδιοτήτων (Attribute Selectors)

Με τον selector αυτόν, ο οποίος δηλώνεται με τη χρήση αγκύλων ([ ]), επιτρέπεται η στόχευση των στοιχείων οι οποίοι βασίζονται στις ιδιότητές τους. Η επιλογή τέτοιων στοιχείων βασίζεται στην παρουσία των παρακάτω:

- Μίας ιδιότητας σε ένα στοιχείο.
- Μίας ακριβώς ορισμένης ιδιότητας μέσα σε ένα στοιχείο.
- Μίας μερικής ιδιότητας μέσα σε ένα στοιχείο (για παράδειγμα, τμήμα της URL).

- Ένα συγκεκριμένο όνομα ιδιότητας σε συνδυασμό της τιμής ή του τμήματος ενός στοιχείου.

Πρέπει να σημειωθεί ότι ο Microsoft Internet Explorer 6 και πιο κάτω δεν υποστηρίζει την χρήση attribute selector.
Στα παρακάτω γίνεται ανάλυση των selector αυτών.

4.5.12 Παρουσία Ιδιότητας σε στοιχείο (Presence of an Attribute)

Ας θεωρηθούν τα anchor tags ως ένα παράδειγμα σχετικά με τον τρόπο που αυτοί οι selectors λειτουργούν. Είναι γνωστό ότι χρησιμοποιούνται και ως σύνδεσμοι αλλά και ως άγκυρες σε μία σελίδα. Υπάρχει η δυνατότητα να μορφοποιηθούν με δύο διαφορετικά στυλ ακολουθώντας τους παρακάτω κανόνες:

```
a[href] { color: red; }
```

κάτι το οποίο θα επιλέξει όλα τα στοιχεία <a> τα οποία έχουν την ιδιότητα href.
Αντίστοιχα, ο κώδικας:

```
a[href][title] { color: red; }
```

θα επιλέξει όλα τα στοιχεία <a> τα οποία έχουν και την ιδιότητα href αλλά και την ιδιότητα title.
Για να επιλεχτούν όλα τα στοιχεία τα οποία έχουν κάποια συγκεκριμένη ιδιότητα τότε γίνεται χρήση του universal selector. Ο κώδικας:

```
*[src]
```

επιλέγει τα στοιχεία εκείνα τα οποία έχουν την ιδιότητα src.

4.5.13 Ακριβής Τιμή Ιδιότητας (Exact Attribute Value)

Υπάρχει επίσης η δυνατότητα επιλογής στοιχείου βασισμένη στην τιμή μίας ιδιότητας. Για παράδειγμα, αν υποτεθεί ότι οι σύνδεσμοι που δείχνουν στην αρχική σελίδα της εφαρμογής πρέπει να εμφανίζονται διαφορετικά από τους υπόλοιπους συνδέσμους, έχοντας τον παρακάτω σύνδεσμο:

```
<a href=http://ourcompany.com/ title=”Our
homepage”>Home</a>
```

μπορούν για τον σκοπό αυτό να εφαρμοστούν οι παρακάτω κανόνες:

```
a[title=”Our homepage”] { color: red; }
a[href=http://ourcompany.com] { color: red; }
```

4.5.14 Μερική Τιμή Ιδιότητας (Partial Attribute Values)

Για ιδιότητες οι οποίες δέχονται ορίσματα λίστας λέξεων χωρισμένα με κενά (όπως είναι η ιδιότητα class), είναι δυνατόν να γίνει επιλογή βασισμένη στην παρουσία μίας συγκεκριμένης λέξης μέσα στην λίστα. Θεωρώντας τον παρακάτω κώδικα:

```
<p class=”warning help”>Hi, I am a paragraph that I help
you</p>
```

για να γίνει η επιλογή αυτής της παραγράφου με το ακριβές ταίριασμα της τιμής, πρέπει να εφαρμοστεί ο κανόνας:

```
p[class=”warning help”]
```

Σημειώνεται ότι οι παρακάτω κανόνες δεν θα έφερναν το επιθυμητό αποτέλεσμα:

```
p[class=”warning”]
p[class=”help”]
```

Κάνοντας χρήση όμως του δείκτη περισπωμένη και ίσον (~=), μπορεί να γίνει επιλογή βάση της παρουσίας μίας λέξης η οποία βρίσκεται σε μία λίστα χωρισμένα με κενά, όπως είναι η παρακάτω:

```
p[class~=”help”] { color: “red” }
```

Σημειώνεται ότι η λειτουργία αυτή είναι ισοδύναμη με τον class selector (p.warning ή p.help) αλλά ο class selector υποστηρίζεται πολύ περισσότερο από τους διάφορους browsers.

4.5.15 Επιλογέας Συγκεκριμένης Ιδιότητας (Particular Attribute Selector)

Ο επιλογέας αυτός ταιριάζει τις τιμές των ιδιοτήτων οι οποίες προηγουμένως έχουν ταιριάξει ακριβώς ή αρχίζουν με ένα συγκεκριμένο κείμενο. Για την δήλωση

αυτού του selector χρησιμοποιούνται τα κάθετος και ίσον (|=). Για παράδειγμα, ο παρακάτω κανόνας:

```
img[src |= "vacation"] { float: left; }
```

θα στοχεύσει κάθε εικόνα της οποίας η τιμή src αρχίζει με το κείμενο vacation. Δηλαδή θα ταιριάξει τα vacation/photo1.jpg και vacation1.jpg αλλά όχι το /vacation/photo1.jpg.

Όπως όμως σημειώθηκε πιο πάνω, ο Microsoft Internet Explorer 6 και πιο κάτω δεν υποστηρίζει την χρήση τέτοιων selector.

4.5.16 Συνδυασμός Επιλογέων (Combining Selectors)

Είναι εφικτός ο συνδυασμός δύο ή περισσότερων selector, όπως είναι ενός στοιχείου και ενός id ή ενός id και μίας κλάσης. Για παράδειγμα, ο παρακάτω κανόνας:

```
p.info { color: blue; }
```

θα επιλέξει μόνο τις παραγράφους (στοιχεία p) οι οποίες ανήκουν στην κλάση info. Όλα τα υπόλοιπα στοιχεία αυτής της κλάσης θα παραλειφθούν και όλες οι υπόλοιπες παράγραφοι οι οποίες δεν ανήκουν στην κλάση info θα μείνουν ανέπαφες.

Χαρακτηριστικό παράδειγμα του πόσο ισχυροί μπορούν να γίνουν οι συνδυασμοί selector είναι το παρακάτω:

```
p#introduction a.info:hover { color: silver; }
```

Ο κανόνας αυτός θα εφαρμοστεί μόνο σε hovered συνδέσμους (στοιχεία a) οι οποίοι ανήκουν στην κλάση info και οι οποίοι είναι απόγονοι της παραγράφου με id introduction.

Τέλος, γίνεται παράθεση ενός παραδείγματος το οποίο ενσωματώνει όλους τους τύπους των επιλογέων σε έναν κανόνα:

```
#primary-content div.story h1 + ul > li
a[href]="http://ourcompany.com" em {
     color: blue;
}
```

Ο κανόνας αυτός θα δώσει μπλε χρώμα σε όλα τα στοιχεία em τα οποία περιέχονται σε anchors των οποίων η ιδιότητα href αρχίζει με http://ourcompany.com και είναι απόγονοι ενός στοιχείου li το οποίο είναι παιδί ενός στοιχείου ul το οποίο είναι αδερφό στοιχείο ενός στοιχείου h1 το οποίο είναι απόγονος ενός div που είναι της κλάσης story το οποίο βρίσκεται μέσα στο στοιχείο με id primary-content.

4.6 Διακριτότητα

Όπως αναφέρθηκε παραπάνω, κάθε τύπος selector ορίζει ένα συγκεκριμένο βαθμό διάκρισης ανάμεσα στα στοιχεία του εγγράφου και καθορίζει τον αριθμό των στοιχείων τα οποία επηρεάζονται από αυτόν τον επιλογέα. Οι σύγχρονοι browser ακολουθούν μία σύνθετη φόρμουλα για τον υπολογισμό της διακριτότητας του κάθε selector. Οι κανόνες που ακολουθούνται είναι οι εξής:

- Ένας universal selector δεν προσδίδει καθόλου διάκριση.
- Ένας selector στοιχείου είναι περισσότερο συγκεκριμένος από τον universla selector.
- Μία κλάση ή ψευδοκλάση προσδίδει μεγαλύτερη διακριτότητα από έναν selector στοιχείου.
- Ένας id selector προσδίδει μεγαλύτερη διακριτότητα από μία κλάση ή μία ψευδοκλάση.
- Οι ιδιότητες ενός ένθετου στυλ (inline style) προσδίδουν τη μεγαλύτερη διακριτότητα από όλα.

Ο υπολογισμός της διακριτότητας γίνεται με συνδυαστικό και αθροιστικό τρόπο και κάθε τύπος επιλογέα φέρει διαφορετικό βάρος βάση των όρων της διακριτότητας. Για παράδειγμα, ένας επιλογέας με δύο κλάσεις είναι περισσότερο συγκεκριμένος από ότι ένας επιλογέας με μία κλάση, ενώ ένας id selector είναι περισσότερο συγκεκριμένος από έναν επιλογέα με δύο κλάσεις. Ο αλγόριθμος υπολογισμού διακριτότητας είναι σχεδιασμένος κατά τέτοιον τρόπο ώστε οποιοσδήποτε αριθμός λιγότερο διακριτών επιλογέων να μην ξεπερνάει ποτέ το βάρος ενός πιο διακριτού επιλογέα. Έτσι, κανένα πλήθος element selector δε μπορούν ποτέ να είναι πιο συγκεκριμένο από μία και μόνο κλάση και κανένα πλήθος κλάσεων δεν μπορεί να είναι ποτέ πιο συγκεκριμένο από έναν και μόνο id selector.

Στον Πίνακα 4.1 παρουσιάζονται μερικά παραδείγματα υπολογισμού της διακριτότητας.

Πίνακας III: Υπολογισμός διακριτότητας

Selector	Inline Style	# of ID Selectors	# of Class Selectors	# of Element Selectors
H1	0,	0,	0,	1
P	0,	0,	0,	1
Div h1	0,	0,	0,	2
Div p	0,	0,	0,	2
div.module h1	0,	0,	1,	2
div.module p	0,	0,	1,	2
Div#content h1	0,	1,	0,	2
Div#content p	0,	1,	0,	2

4.7 Επισύναψη Style Sheet στα έγγραφα

Για την επισύναψη ενός style sheet σε κάποια σελίδα, πρέπει να γίνει η σύνδεση του εγγράφου με το συγκεκριμένο style sheet. Όταν ο browser κάνει download το HTML έγγραφο τότε αυτόματα θα αναζητήσει τους CSS κανόνες που συμπεριλαμβάνουν τις οδηγίες σχετικά με τη μορφή που θα έχει το έγγραφο. Υπάρχουν διάφοροι τρόποι για να εισαχθούν αυτοί οι κανόνες οι οποίοι περιγράφονται παρακάτω.

4.7.1 Ένθετα Στυλ (Inline Style)

Οι δηλώσεις του CSS μπορούν να συμπεριληφθούν με τη χρήση της ιδιότητας style σε κάθε στοιχείο ξεχωριστά. Τα inline styles δεν γράφονται σαν κανόνες και δεν υπάρχει επιλογέας επειδή οι ιδιότητες και οι τιμές επισυνάπτονται απευθείας με το κάθε στοιχείο. Τα inline styles είναι τα πιο συγκεκριμένα από όλα επειδή εφαρμόζονται σε ακριβώς ένα στοιχείο και σε κανένα άλλο.

Παρόλα αυτά πρέπει να αποφεύγεται η χρήση ένθετων στυλ επειδή η μίξη της δομής με την παρουσίαση ακυρώνουν πολλά από τα πλεονεκτήματα που παρέχει η χρήση του CSS.

Παράδειγμα ενός τέτοιου στυλ είναι το παρακάτω:

```
<p style= "color: blue">text</p>
```

Με την χρήση αυτού του style η συγκεκριμένη παράγραφος θα μορφοποιηθεί με μπλε χρώμα. Τέτοιες μορφοποιήσεις μπορούμε να κάνουμε σχεδόν σε κάθε HTML στοιχείο.

4.7.2 Ενσωματωμένα Στυλ (Embedded Style Sheets)

Τα ενσωματωμένα στυλ δηλώνονται στο στοιχείο <head> του εγγράφου και οι κανόνες που εμπεριέχονται σε αυτό ακολουθούνται μόνο από το συγκεκριμένο έγγραφο. Ένα embedded style sheet περιέχεται μέσα στο στοιχείο style και διαχωρίζει σε μεγάλο βαθμό τη δομή από την παρουσίαση του εγγράφου. Παράδειγμα ενός τέτοιου στυλ είναι το παρακάτω:

```
<html>
     <head>
          <title>CSS Example</title>
          <style type="text /css">
          p { color: green; }
          a { color: blue; }
          </style>
     </head>
<html>
```

Το κομμάτι κώδικα αυτό θα κάνει όλες τις παραγράφους της HTML σελίδας πράσινες και όλους τους συνδέσμους της σελίδας μπλε. Όμως το στυλ αυτό έχει το μειονέκτημα ότι μπορεί να εφαρμοστεί μόνο σε μία σελίδα κι έτσι δεν είναι επαναχρησιμοποιήσιμο κάτι που οδηγεί στην επανάληψη του ίδιου κώδικα.

4.7.3 Εξωτερικά Φύλλα Στυλ (External Style Sheet)

Η τελευταία και καλύτερη επιλογή συγγραφής των CSS κανόνων είναι σε ένα ξεχωριστό, εξωτερικό αρχείου το οποίο συνδέεται απευθείας με τα έγγραφα όπου αυτό απαιτείται. Ένα external style sheet είναι ένα απλό αρχείο κειμένου το οποίο αποθηκεύεται με κατάληψη .css. Αυτή η προσέγγιση διαχωρίζει πλήρως την παρουσίαση από το περιεχόμενο αφού αυτά δεν βρίσκονται στο ίδιο αρχείο.

Επίσης, ένα τέτοιο αρχείο μπορεί να συνδεθεί με πολλά HTML έγγραφα επιτρέποντας έτσι σε πλήθος ιστοσελίδων να ελέγχονται από ένα και μόνο κεντρικό αρχείο. Οι αλλαγές σε αυτό το αρχείο θα εφαρμοστούν προοδευτικά σε όλες τις σελίδες οι οποίες είναι συνδεδεμένες με αυτό. Κάτι τέτοιο είναι πολύ ευέλικτο και δείχνει το πόσο ισχυρό και χρήσιμο είναι το CSS.
Ένα HTML έγγραφο συνδέεται με ένα external style sheet μέσω ενός συνδέσμου στο στοιχείο <head> του εγγράφου. Το παρακάτω τμήμα κώδικα δείχνει πως επιτυγχάνεται αυτό:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
     <html xmlns="http://www.w3.org/1999/xhtml" lang="en"
xml:lang="en">
     <head>
          <title>Spaghetti and Cruft : Our Menu</title>
          <link rel="stylesheet" type="text/css"
     href="styles.css" />
     </head>
     <body>
          <h2>This is managable via a external style
     sheet</h2>
          <p>Me too!>p>
     </body>
</html>
```

4.7.4 !important

Εκτός από τους προηγούμενους τρόπους επισύναψης των CSS στυλ, υπάρχει κι ένας ακόμα τρόπος ο οποίος είναι εξαιρετικά σπάνιος. Αυτός είναι μέσω της λέξης κλειδί !important και αναγκάζει τον browser να υπερβεί όλους τους άλλους κανόνες CSS που προηγουμένως είχαν δηλωθεί με τους κανόνες που δηλώνονται με τη χρήση αυτής της λέξης κλειδί. Ένα τέτοιο παράδειγμα είναι ο παρακάτω κανόνας:

```
h1 { color: blue !important; }
```

ο οποίος θα μορφοποιήσει με μπλε χρώμα το συγκεκριμένο <h1> στοιχείο.
Πρέπει να σημειωθεί ότι η χρήση αυτού του τρόπου επισύναψης στυλ πρέπει να αποφεύγεται καθώς ακυρώνει πολλά από τα πλεονεκτήματα που προσφέρει το CSS.

4.8 Διακριτότητα διαφορετικών πηγών στυλ

Κάθε τρόπος επισύναψης στυλ έχει το δικό του ξεχωριστό βάρος σε σχέση με το ποιο θα είναι τελικά το στυλ που θα εφαρμοστεί. Έτσι, ακολουθείται η παρακάτω σειρά:

1. Browser style sheet
2. User style sheet
3. External style sheet
4. Embedded style sheet
5. Inline style
6. !important

4.9 Επίλογος

Όπως γράφτηκε πιο πάνω, τα επίπεδα του περιεχομένου και της παρουσίασης, τα οποία σε προγραμματιστικό επίπεδο μεταφράζονται σε HTML και CSS αντίστοιχα, είναι δύο από τα τρία μέρη από τα οποία αποτελείται μία σύγχρονη ιστοσελίδα. Το τρίτο μέρος είναι το επίπεδο της συμπεριφοράς το οποίο αναλύεται στο επόμενο κεφάλαιο.

ΚΕΦΑΛΑΙΟ 5ο

JAVASCRIPT ΚΑΙ DOM

5.1 Εισαγωγή

Με την πάροδο του χρόνου, όσο περισσότερες ιστοσελίδες δημιουργούσαν οι σχεδιαστές και οι προγραμματιστές, τόσο μεγάλωνε η απαίτηση για δυναμικές ιστοσελίδες. Ένα από τα προβλήματα όμως που είχαν να αντιμετωπίσουν ήταν ο πόλεμος των browser που είχε σαν αποτέλεσμα την ασυμβατότητα των προγραμμάτων ανάμεσα σε browser διαφορετικών εταιρειών. Μέχρι πριν λίγο καιρό δεν ήταν εύκολο να δημιουργηθούν εφαρμογές JavaScript οι οποίες να εκτελούνταν παντού το ίδιο. Αυτό όμως άρχισε να αλλάζει με την χρήση της τεχνολογίας DOM.

5.2 JavaScript

Η γλώσσα προγραμματισμού JavaScript αναπτύχθηκε από την εταιρεία Netscape, σε συνεργασία με την Sun Microsystems και η πρώτη της έκδοση δημοσιεύτηκε το 1995. Ακολούθησε η αντίστοιχη γλώσσα της Microsoft η οποία ονομάστηκε Jscript και η επόμενη έκδοση της JavaScript που είχε το όνομα ECMAScript που αργότερα όμως καθιερώθηκε με το όνομα που είναι γνωστό μέχρι σήμερα.

Η JavaScript είναι μία διερμηνευμένη (interpreted) γλώσσα προγραμματισμού με ιδιότητες αντικειμενοστραφούς γλώσσας προγραμματισμού, χωρίς όμως να μπορεί να χαρακτηριστεί ως πλήρης αντικειμενοστραφής.

Η γλώσσα αυτή, κτίστηκε ουσιαστικά πάνω στο πρότυπο των γλωσσών C, C++ και Java. Από την άλλη όμως έχει μία πολύ σημαντική διαφορά στο ότι διαχειρίζεται τους τύπους δεδομένων πιο χαλαρά (loosely typed) σε σχέση με τη σφικτή διαχείριση τύπων δεδομένων (strongly typed) που γίνεται στις προαναφερόμενες γλώσσες. Στην JavaScript οι μεταβλητές δεν είναι απαραίτητο να έχουν ένα συγκεκριμένο τύπο ή ακόμη είναι δυνατόν να αλλάζουν τύπο κατά τη διάρκεια της ζωής τους. Επίσης, δεν πρέπει να συγχέεται η JavaScript με την Java της Sun Microsystems. Η μία δεν έχει καμία σχέση με την άλλη. Η χρήση του ονόματος JavaScript έγινε για λόγους προώθησης της γλώσσας σε μία εποχή που η εξάπλωση της Java ήταν πολύ μεγάλη.

5.2.1 Χρήσεις της JavaScript

Η γλώσσα JavaScript χρησιμοποιείται κυρίως για την εξυπηρέτηση των παρακάτω σκοπών:

- Λιγότερος φόρτος των server: Ο έλεγχος και η επικύρωση των δεδομένων που εισάγονται από τους χρήστες γίνεται από τη μεριά του browser κι έτσι δεδομένα τα οποία δεν είναι σε κατάλληλη μορφή δεν αποστέλλονται στον server. Αυτό όμως δεν σημαίνει ότι ο έλεγχος δεν πρέπει να γίνεται και στη μεριά των εξυπηρετητών καθώς κάποιος χρήστης μπορεί να μην έχει ενσωματωμένη την JavaScript στον browser του ή υπάρχει πιθανότητα να την έχει απενεργοποιήσει.
- Άμεση αλληλεπίδραση με τους χρήστες: Με την χρήση της JavaScript για τον έλεγχο των δεδομένων μειώνονται οι χρόνοι αναμονής του χρηστών αφού αυτοί δεν χρειάζεται να περιμένουν μεγάλα χρονικά διαστήματα επαναφόρτωσης της σελίδας σε περίπτωση που έχουν ξεχάσει να εισάγουν κάποιο δεδομένο ή έχουν εισάγει κάτι λάθος.
- Αυτόματη διόρθωση λαθών: Ένα παράδειγμα που μπορεί να κάνει περισσότερο κατανοητό το πώς μπορεί να χρησιμοποιηθεί η JavaScript με αυτόν τον τρόπο είναι αυτό της ημερομηνίας. Πολλά συστήματα βάσεων δεδομένων αποθηκεύουν δεδομένα ημερομηνιών σε μορφή dd-mm-yyyy. Αν κάποιος χρήστης εισάγει κάποια ημερομηνία σε μορφή dd/mm/yyyy τότε κάτι τέτοιο θα μπορούσε να ανιχνευτεί αυτόματα από τον browser και να μετατραπεί στην σωστή μορφή πριν τα δεδομένα αποσταλούν στον server.
- Αυξημένη χρηστικότητα: Αυτό επιτυγχάνεται επιτρέποντας στον χρήστη την αλλαγή και αλληλεπίδραση με το γραφικό περιβάλλον χωρίς την επαναφόρτωση της σελίδας. Ένα τέτοιο παράδειγμα είναι τα πτυσσόμενα μενού.
- Αυξημένη δυνατότητα αλληλεπίδρασης: Ένα τέτοιο παράδειγμα όπου κάτι τέτοιο επιτυγχάνεται είναι τα μενού τα οποία αλληλεπιδρούν όταν ο χρήστης περάσει το mouse πάνω από αυτά – η λειτουργία hover – κάτι το οποίο έχει ως αποτέλεσμα να δημιουργηθεί μία σειρά από γεγονότα τα οποία έχουν προγραμματιστεί να λειτουργούν με έναν συγκεκριμένο τρόπο.

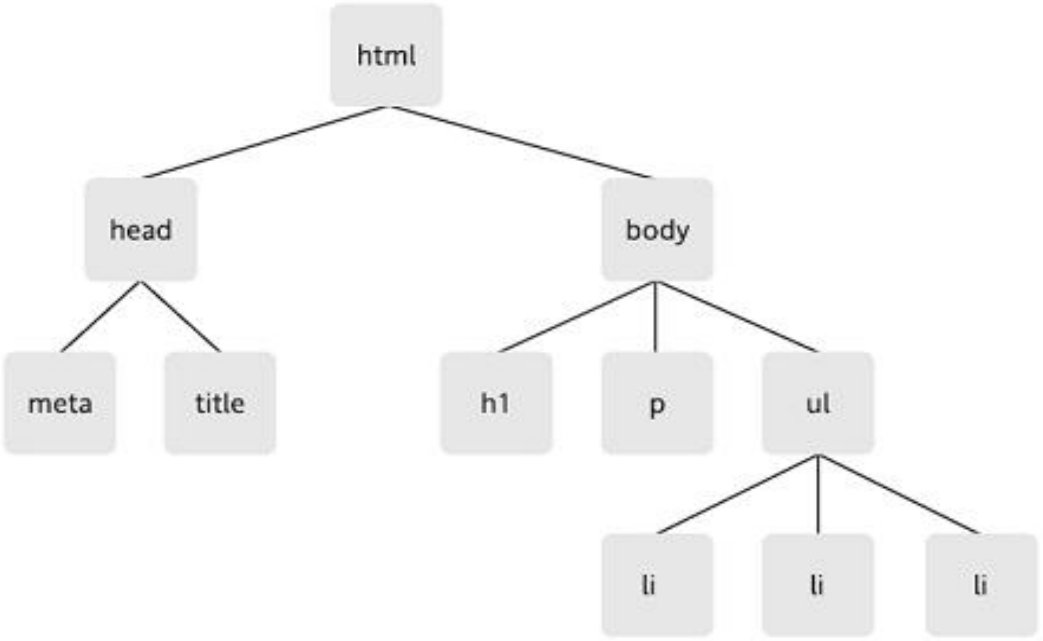
- Καλύτερα γραφικά περιβάλλοντα: Χρησιμοποιώντας την JavaScript μπορούν να συμπεριληφθούν αντικείμενα με λειτουργίες drag-and-drop καθώς και plug-ins, όπως είναι το Flash.
- Ελαφρότερα περιβάλλοντα: Αντί της απαίτησης download ενός μεγάλου αρχείου Java applet ή ενός Flash movie, τα προγράμματα γραμμένα σε JavaScript είναι μικρά σε μέγεθος και αποθηκεύονται στη μνήμη του browser μόλις κατέβουν.

5.3 JavaScript και Document Object Model (DOM)

Σύμφωνα με τον ορισμό του W3C, το DOM είναι ένα ουδέτερο σε λειτουργικό και γλώσσα προγραμματισμού περιβάλλον το οποίο επιτρέπει σε προγράμματα και script να έχουν πρόσβαση και να ανανεώνουν το περιεχόμενο, τη δομή και το στυλ των εγγράφων. Το έγγραφο μπορεί να επεξεργαστεί περαιτέρω και τα αποτελέσματα αυτής της επεξεργασίας μπορούν να ενσωματωθούν στην σελίδα η οποία βρίσκεται υπό παρουσίαση.

Το πιο σημαντικό στοιχείο το οποίο χρησιμοποιείται από το DOM είναι η αναπαράσταση του εγγράφου ως ένα δέντρο. Πιο συγκεκριμένα, ολόκληρο το έγγραφο αναπαριστάται ως ένα οικογενειακό δέντρο. Για παράδειγμα, το Σχήμα ΙΙΙ αντιστοιχεί στο παρακάτω κομμάτι κώδικα:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
        <meta http-equiv="content-type" content="text/html; charset=utf-8" />
        <title>Shopping list</title>
</head>
<body>
        <h1>What to buy</h1>
        <p title="a gentle reminder">Don’t forget to buy this stuff.</p>
        <ul id="purchases">
                <li>A tin of beans</li><li>Cheese</li><li>Milk</li>
        </ul>
</body>
</html>
```



Σχήμα III: Αναπαράσταση ενός δέντρου DOM

5.3.1 Βασικά χαρακτηριστικά ενός DOM εγγράφου

5.3.1.1 Κόμβοι

Ο όρος κόμβος χρησιμοποιείται για να δείξει ένα σημείο σύνδεσης ανάμεσα σε δύο στοιχεία του δέντρου. Παρακάτω περιγράφονται οι τρεις τύποι κόμβων του DOM.

5.3.1.1.1 Element node

Όπως προειπώθηκε, το κύριο συστατικό του DOM είναι οι κόμβοι. Ένα είδος κόμβου είναι το στοιχείο (element). Τέτοια element είναι το <body>, το <p>, το <ul> κτλ. Η ετικέτα (tag) δίνει το όνομα των element. Τα element μπορούν να περιέχουν άλλα element όπως για παράδειγμα ενός <ul> που περιέχει <li>. Μόνο το στοιχείο <html> δεν περιέχεται σε άλλο element και για αυτό και ονομάζεται και ως στοιχείο ρίζα (root).

5.3.1.1.2 Text node

Άλλο είδος node είναι τα text node. Αν ένα έγγραφο αποτελούνταν μόνο από άδεια στοιχεία τότε αυτό θα είχε δομή αλλά καθόλου περιεχόμενο. Έτσι, χρησιμοποιούνται τα text node. Για παράδειγμα ένα element <p> περιέχει τον text node “Hello world”. Στην XHTML τα text nodes περιέχονται πάντα μέσα σε

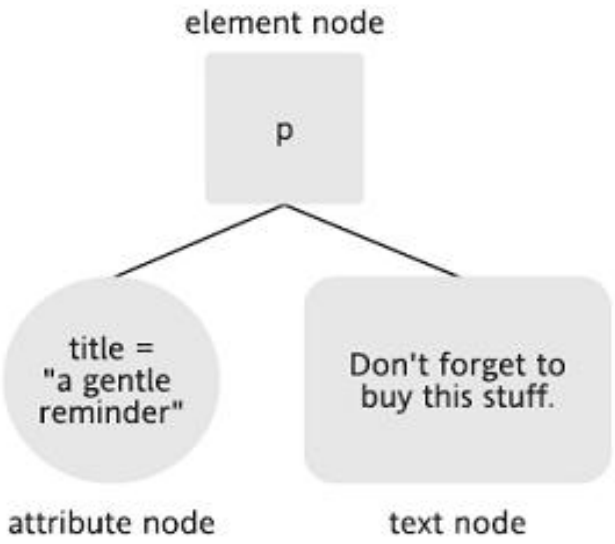
element nodes. Αλλά όλα τα element nodes δεν περιέχουν text nodes. Έτσι, ένα <ol> περιέχει <li> που αυτά περιέχουν text node.

5.3.1.1.3 Attribute node

Άλλο είδος node είναι το attribute node. Τέτοιο για παράδειγμα είναι το <title> που χρησιμοποιείται για να προσδώσει μία πιο συγκεκριμένη πληροφορία για το παρακάτω στοιχείο <p>.

```
<p title=”a gentle reminder”>Don’t forget to buy this stuff.</p>
```

Σημειώνεται ότι δεν περιέχουν όλα τα στοιχεία ιδιότητες, αλλά όλες οι ιδιότητες περιέχονται από στοιχεία. Για να γίνει αυτό πιο κατανοητό παρατίθεται το Σχήμα IV:



Σχήμα IV: Elements και attributes

5.3.2 Μέθοδοι του DOM

Το τμήμα αυτό της εργασίας, περιέχει μία λίστα μερικών από τις σημαντικότερες και πιο χρήσιμες μεθόδους οι οποίες παρέχονται από το DOM.

5.3.2.1 CreateElement

Η μέθοδος createElement δημιουργεί ένα καινούριο στοιχείο κόμβου με ένα συγκεκριμένο όνομα ετικέτας. Η μέθοδος επιστρέφει μία αναφορά στον νέο ορισμένο στοιχείο:

```
reference = document.createElement(element)
```

Η μέθοδος παίρνει μία μόνο παράμετρο και αυτή είναι το όνομα του στοιχείου που πρόκειται να δημιουργηθεί. Για παράδειγμα:

```
reference = document.createElement("h1")
```

Η αναφορά η οποία επιστρέφεται από την createElement είναι ένα αντικείμενο κόμβου. Επειδή αυτό είναι ένα στοιχείο κόμβου η ιδιότυτα nodeType θα έχει την τιμή 1.

```
reference = document.createElement("p")
```

Στο παραπάνω παράδειγμα η εντολή para.nodeType επιστρέφει την τιμή 1. Αντίθετα, η εντολή para.nodeName επιστρέφει την τιμή “p” ή “P”.

Ένα στοιχείο το οποίο δημιουργείται κατά αυτόν τον τρόπο δεν προστίθεται αυτόματα στο έγγραφο. Ο νέος κόμβος δεν έχει κάποιον κόμβο πατέρα οπότε για να το προσθέσουμε στα υπόλοιπα πρέπει να χρησιμοποιήσουμε μία από τις μεθόδους appendChild, insertBefore ή και replaceChild. Για παράδειγμα:

```
var para = document.createElement("p");
document.body.appentChild(para);
```

Τα παραπάνω δημιουργούν ένα στοιχείο παραγράφου και το προσθέτουν ως τελευταίο παιδί στο στοιχείο <body>.

Η εφαρμογή μεθόδων σε ένα στοιχείο το οποίο δημιουργείται κατά αυτόν τον τρόπο μπορεί να γίνει οποιαδήποτε στιγμή έπειτα από την δημιουργία τους.

5.3.2.2 createTextNode

Η μέθοδος αυτή δημιουργεί έναν καινούριο κόμβο κειμένου ο οποίος περιέχει ένα συγκεκριμένο κείμενο. Η μέθοδος επιστρέφει μία αναφορά στο νέο δημιουργημένο κόμβο κειμένου. Για παράδειγμα:

```
reference = document.createTextNode(text)
```

Η μέθοδος παίρνει μία παράμετρο η οποία είναι το κείμενο το οποίο πρόκειται να δημιουργηθεί:

```
reference = document.createTextNode("hello world")
```

Η αναφορά που επιστρέφεται από την createTextNode είναι ένα αντικείμενο κόμβου και η ιδιότητα nodeType θα έχει την τιμή 3. Στο παράδειγμα:

```
var message = document.createTextNode("hello world");
```

Η εντολή message.nodeType θα επιστρέψει την τιμή 1 ενώ η message.nodeName θα επιστρέψει την τιμή "#text".

Ένας κόμβος κειμένου που δημιουργείται με τη μέθοδο αυτή δεν προστίθεται αυτόματα στο έγγραφο αφού δεν έχει κάποιον κόμβο πατέρα οπότε για να προστεθεί στα υπόλοιπα στοιχεία πρέπει να χρησιμοποιηθεί μία από τις μεθόδους appendChild, insertBefore ή και replaceChild, όπως ακριβώς γίνεται και με τη μέθοδο createElement.

5.3.2.3 clodeNode

Το DOM παρέχει μεθόδους για την αντιγραφή κόμβων. Η μέθοδος cloneNode δημιουργεί ένα αντίγραφο ενός συγκεκριμένου κόμβου. Η μέθοδος αυτή επιστρέφει μία αναφορά σε έναν καινούριο κλωνοποιημένο κόμβο και είναι της μορφής:

```
reference = node.cloneNode(deep)
```

Η μέθοδος παίρνει μία παράμετρο η οποία μπορεί να έχει μία τιμή δυαδικής φύσης δηλαδή true ή false. Η παράμετρος αυτή καθορίζει αν ο καινούριος κόμβος θα περιέχει τους ίδιους κόμβους-παιδιά όπως ο αρχικός. Αν η τιμή της παραμέτρου είναι true τότε ο καινούριος κόμβος θα έχει όλα τα ίδια παιδιά με τον

κόμβο από τον οποίο κλωνοποιήθηκε. Αν η τιμή είναι false τότε ο καινούριος κόμβος δεν θα έχει κόμβους παιδιά. Αν ο κόμβος είναι ένας κόμβος στοιχείου, αυτό σημαίνει ότι οποιοδήποτε κείμενο μέσα στο αρχικό στοιχείο δε θα κλωνοποιηθεί αλλά οι ιδιότητες θα το κάνουν:

```
reference = node.clodeNode(true)
reference = node.cloneNode(false)
```

Η reference η οποία επιστράφηκε από τη μέθοδο cloneNode είναι ένα αντικείμενο κόμβου. Αυτός ο κόμβος μπορεί να έχει τις ίδιες ιδιότητες nodeType και nodeName που είχε ο αρχικός κόμβος:

```
var para = document.createElement("p");
var newpara = para.cloneNode(false);
```

Το παραπάνω παράδειγμα δημιουργεί έναν καινούριο κόμβο, τον para κι έπειτα ένας ακόμα κόμβος, ο newpara, δημιουργείται. Η τιμή para.nodeType είναι 1 κι έτσι η τιμή της newpara.nodeType είναι επίσης 1.

Ένας κόμβος κειμένου που δημιουργείται με τη μέθοδο αυτή δεν προστίθεται αυτόματα στο έγγραφο αφού δεν έχει κάποιον κόμβο πατέρα οπότε για να προστεθεί στα υπόλοιπα πρέπει να χρησιμοποιηθεί μία από τις μεθόδους appendChild, insertBefore ή και replaceChild, όπως ακριβώς γίνεται και με τη μέθοδο createElement.

Σημειώνεται ότι αν κλωνοποιηθεί ένα στοιχείο το οποίο έχει οριστεί η ιδιότητα id για αυτό, τότε πρέπει πάντα να τροποποιείται η τιμή της ιδιότητας id για το νέο στοιχείο αφού αυτή πρέπει να είναι μοναδική για όλο το έγγραφο.

5.3.2.4 AppendChild

Η μέθοδος αυτή προσθέτει ένα νέο κόμβο-παιδί σε έναν κόμβο και είναι της μορφής:

```
reference = element.appendChild(newChild)
```

Ο νέος κόμβος γίνεται ο τελευταίος κόμβος του στοιχείου στο οποίο έχει προστεθεί. Η μέθοδος αυτή επιστρέφει μία αναφορά προς τον νέο κόμβο.

Η μέθοδος αυτή συνήθως χρησιμοποιείται σε συνδυασμό με τις μεθόδους createElement και createTextNode οι οποίες χρησιμοποιούνται για να δημιουργήσουν νέους κόμβους.

Στο παρακάτω παράδειγμα, δημιουργείται ένα στοιχείο παραγράφου. Ένας κόμβος κειμένου δημιουργείται με τη μέθοδο createTextNode. Ο κόμβος κειμένου εισάγεται στην παράγραφο με τη χρήση της μεθόδου appendChild:

```
var para = document.createElement("p");
var message = document.createTextNode("hello world");
para.appendChild(message);
```

Το στοιχείο παραγράφου μαζί με τα παιδιά της μπορεί να εισαχθεί στη δομή του εγγράφου χρησιμοποιώντας εκ νέου την appendChild:

```
document.body.appendChild(para);
```

Στην παραπάνω περίπτωση το στοιχείο para προστέθηκε στο στοιχείο body του εγγράφου.

5.3.2.5 InsertBefore

Η μέθοδος αυτή χρησιμοποιείται για να εισάγει έναν καινούριο κόμβο μέσα σε ένα στοιχείο πριν από ένα συγκεκριμένο κόμβο-παιδί του στοιχείου και έχει την παρακάτω μορφή:

```
reference = element.insertBefore(newNode, targetNode)
```

Ο κόμβος newNode εισάγεται στο στοιχείο ακριβώς πριν τον κόμβο targetNode. Ο κόμβος targetNode πρέπει να είναι ένας κόμβος-παιδί του στοιχείου. Αν ο targetNode δεν καθοριστεί, τότε ο newNode θα προστεθεί στο τέλος του στοιχείου element. Σε αυτήν την περίπτωση συμπεριφέρεται ακριβώς σαν τη μέθοδο appendChild.

Η μέθοδος insertBefore χρησιμοποιείται συνήθως με τις μεθόδους createElement και createTextNode για την εισαγωγή κόμβων στο δέντρο του εγγράφου.

Στο παρακάτω παράδειγμα, το έγγραφο έχει ένα στοιχείο με την τιμή της ιδιότητας id να είναι “content”. Το στοιχείο αυτό περιέχει ένα στοιχείο με το id του να είναι “fineprint”. Δημιουργείται μία νέα παράγραφος χρησιμοποιώντας την

createElement και έπειτα αυτή εισάγεται μέσα στο στοιχείο με id “content” ακριβώς πριν από το στοιχείο με id “fineprint”.

```
var container = document.getElementById(“content”);
var message = document.getElementById(“fineprint”);
var para = document.createElement(“p”);
```

Σημειώνεται ότι αν ο κόμβος ο οποίος εισάγεται έχει με τη σειρά του κόμβους-παιδιά τότε αυτά θα εισαχθούν επίσης πριν από τον targetNode.

Όμως, εκτός από την εισαγωγή νέων κόμβων, η μέθοδος insertBefore μπορεί να χρησιμοποιηθεί και για τη μετακίνηση κόμβων οι οποίοι υπάρχουν ήδη στο έγγραφο. Στο παρακάτω παράδειγμα, το έγγραφο έχει ένα στοιχείο με id “content”. Το στοιχείο αυτό περιέχει ένα στοιχείο με id “fineprint”. Σε κάποια άλλη θέση του εγγράφου, υπάρχει ένα στοιχείο με id “headline”. Το στοιχείο αυτό μετακινείται μέσα στο στοιχείο “content” και τοποθετείται πριν το στοιχείο “fineprint”.

```
var container = document.getElementById(“content”);
var message = document.getElementById(“fineprint”);
var announcement = document.getElementById(“headline”);
```

Το στοιχείο με id “headline” πρώτα αφαιρείται από το δέντρο του εγγράφου κι έπειτα εισάγεται ξανά στην καινούρια θέση του πριν από το στοιχείο “fineprint” και μέσα στο στοιχείο “content”.

5.3.2.6 removeChild

Η μέθοδος removeChild αφαιρεί έναν κόμβο-παιδί από ένα συγκεκριμένο στοιχείο-γονέα και έχει την παρακάτω μορφή:

```
reference = element.removeChild(node)
```

Η μέθοδος επιστρέφει μία αναφορά προς τον κόμβο ο οποίος έχει αφαιρεθεί.

Όταν ένας κόμβος αφαιρεθεί με τη μέθοδο αυτή, όλοι οι κόμβοι-παιδιά που περιέχονται σε αυτόν αφαιρούνται επίσης.

Στο παρακάτω παράδειγμα, το στοιχείο με id “content” περιέχει ένα στοιχείο με id “fineprint”. Το στοιχείο “fineprint” αφαιρείται από το στοιχείο “content” χρησιμοποιώντας τη μέθοδο removeChild.

```
var container = document.getElementById("content");
var message = document.getElementById("fineprint");
container.removeChild(message);
```

Σε περίπτωση που απαιτείται η αφαίρεση ενός κόμβου αλλά δεν υπάρχει αναφορά προς τον κόμβο-πατέρα, μπορεί να χρησιμοποιηθεί η ιδιότητα parentNode του κόμβου ο οποίος πρέπει να αφαιρεθεί. Για παράδειγμα:

```
var message = document.getElementById("fineprint");
var container = message.parentNode;
container.removeChild(message);
```

Σε περίπτωση που υπάρχει η απαίτηση μετακίνησης ενός κόμβου από ένα μέρος του εγγράφου σε ένα άλλο δεν χρειάζεται η χρήση της removeChild. Οι μέθοδοι appendChild και insertBefore αυτόματα αφαιρούν κόμβους από ένα μέρος του δέντρου και τους εισάγουν στις νέες τους θέσεις.

5.3.2.7 replaceChild

Το DOM παρέχει μία μέθοδο για την αντικατάσταση κόμβων στο έγγραφο και αυτή είναι η replaceChild η οποία αντικαθιστά έναν κόμβο-παιδί που βρίσκεται σε έναν συγκεκριμένο γονιό με έναν άλλον κόμβο. Η μέθοδος αυτή είναι της μορφής:
reference = element.replaceChild(newChild, oldChild)
Ο κόμβος oldChild πρέπει να είναι κόμβος τύπου στοιχείου (element). Η μέθοδος επιστρέφει μία αναφορά προς τον κόμβο ο οποίος έχει αντικατασταθεί.
Στο παρακάτω παράδειγμα, ένα στοιχείο με id “content” περιέχει ένα στοιχείο με το id “fineprint”. Δημιουργείται ένα νέο στοιχείου παραγράφου χρησιμοποιώντας τη μέθοδο createElement. Με τη χρήση της replaceChild το καινούριο στοιχείο αντικαθιστά το στοιχείο “fineprint”.

```
var container = document.getElementById("content");
var message = document.getElementById("fineprint");
var para = document.createElement("p");
container.replaceChild(para, message);
```

Αν ο νέος κόμβος έχει με την σειρά του κόμβους-παιδιά, τότε θα εισαχθούν και αυτοί στο έγγραφο.

Η μέθοδος replaceChild χρησιμοποιείται επίσης και για κόμβους οι οποίοι είναι ήδη μέρος του εγγράφου. Αν ο κόμβος newChild υπάρχει ήδη τότε πρώτα θα αφαιρεθεί κι έπειτα θα αντικατασταθεί από τον κόμβο oldChild.

5.3.2.8 setAttribute

Η μέθοδος setAttribute προσθέτει μία νέα τιμή σε μία ορισμένη ιδιότητα ή αλλάζει την τιμή μίας υπάρχουσας ιδιότητας ενός συγκεκριμένου κόμβου και είναι της μορφής:

```
element.setAttribute(attributeName, attributeValue)
```

Το όνομα και η τιμή της ιδιότητας ορίζονται στη μέθοδο ως παράμετροι. Αν η ιδιότητα υπάρχει ήδη η τιμή της θα ανανεωθεί. Αν η ιδιότητα δεν υπάρχει τότε θα δημιουργηθεί και θα της δοθεί η τιμή της παραμέτρου. Η μέθοδος setAttribute μπορεί να χρησιμοποιηθεί μόνο με κόμβους τα οποία είναι στοιχεία (element).
Στο παρακάτω παράδειγμα, προστίθεται η ιδιότητα title με την τιμή “this is important” στο στοιχείο με id “fineprint”.

```
var message = document.getElementById(“fineprint”);
message.setAttribute(“title”, “this is important”);
```

Ασχέτως αν το στοιχείο είχε ή όχι την ιδιότητα title, η εφαρμογή του παραπάνω κώδικα θα έχει ως αποτέλεσμα το στοιχείο να έχει την τιμή “this is important” στην ιδιότητα title.

5.3.2.9 getAttribute

Η μέθοδος getAttribute επιστρέφει την τιμή μίας συγκεκριμένης ιδιότητας ενός ορισμένου κόμβου και είναι της μορφής:

```
attributeValue = element.getAttribute(attributeName)
```

Το όνομα της ιδιότητας περνιούνται στη μέθοδο ως παράμετροι. Η τιμή της ιδιότητας επιστρέφεται με τη μορφή συμβολοσειράς (string). Αν η ιδιότητα δεν υπάρχει τότε η getAttribute επιστρέφει μία άδεια συμβολοσειρά.
Το παρακάτω παράδειγμα ανακτά την ιδιότητα title ενός στοιχείου με id “fineprint” και το αποθηκεύει σε μία μεταβλητή με όνομα titletext:

```
var message = document.getElementById("fineprint");
var titletext = message.getAttribute("title");
```

5.3.2.10 getElementById

Η μέθοδος αυτή βρίσκει ένα στοιχείο το οποίο έχει ένα συγκεκριμένο id και είναι της μορφής:

```
element = document.getElementById(id)
```

Η μέθοδος επιστρέφει έναν κόμβο με το συγκεκριμένο id. Αν δεν υπάρχει τέτοιο στοιχείο τότε η getElementById επιστρέφει null. Η μέθοδος αυτή μπορεί να εφαρμοστεί μόνο στο αντικείμενο document.

Ο κόμβος που επιστρέφεται είναι ένα αντικείμενο, το οποίο περιέχει όλες τις ιδιότητες όπως είναι οι nodeName, nodeType, parentNode, childNodes κτλ.

Το παρακάτω παράδειγμα ανακτά το στοιχείο με id “fineprint” και το αποθηκεύει στη μεταβλητή message. Ο κόμβος-γονιός της message είναι επίσης ένα στοιχείο και αποθηκεύεται στην μεταβλητή container:

```
var message = document.getElementById("fineprint");
var container = message.parentNode;
```

Αν ένα στοιχείο έχει ένα id τότε η μέθοδος getElementById είναι ο πιο απλός και γρήγορος τρόπος για να γίνει αναφορά σε αυτό. Έπειτα, μπορεί να γίνει εφαρμογή των μεθόδων setAttribute, cloneNode, appendChild κτλ.

5.3.2.11 getElementsByTagName

Η μέθοδος αυτή βρίσκει όλα τα στοιχεία που έχουν ένα συγκεκριμένο όνομα ετικέτας και είναι της μορφής:

```
elements = document.getElementsByTagName(tagName)
```

Η μέθοδος αυτή επιστρέφει μία λίστα με όλα τα στοιχεία που έχουν τη συγκεκριμένα ετικέτα (tag). Σημειώνεται ότι η λίστα αυτή μπορεί να χειριστεί όπως και οι πίνακες. Η ιδιότητα length της λίστας είναι ίση με τον αριθμό των στοιχείων που επιστράφηκαν από την εκτέλεση της μεθόδου. Κάθε στοιχείο στον πίνακα

είναι ένα αντικείμενο το οποίο έχει όλες τις ιδιότητες όπως είναι οι nodeName, nodeType, parentNode, childNodes κτλ.

Το παρακάτω παράδειγμα ανακτά όλες τις παραγράφους σε ένα έγγραφο. Η ιδιότητα length της επιστρεφόμενης λίστας αποθηκεύεται στη μεταβλητή howmany:

```
var paras = document.getElementsByTagName("p");
var howmany = paras.length;
```

Η μέθοδος αυτή χρησιμοποιείται συχνά μέσα σε έναν βρόγχο for ώστε να προσπελάσει όλα τα στοιχεία της επιστρεφόμενης λίστας. Με αυτόν τον τρόπο, κάθε στοιχείο μπορεί να χειριστεί με τις μεθόδους setAttribute, cloneNode, appendChild κτλ.

Στο παρακάτω παράδειγμα, προσπελαύνονται όλες οι παράγραφοι του εγγράφου και τίθεται η ιδιότητα title κάθε μίας σε με την τιμή μίας άδειας συμβολοσειράς:

```
var paras = document.getElementsByTagName("p");
for (var i=0; i < paras.length; i++) {
     paras[i]. setAttribute("title", "");
}
```

Στο παραπάνω παράδειγμα η μεταβλητή paras είναι μία λίστα κόμβων. Τα αντικείμενα σε αυτήν την λίστα μπορούν να προσπελαστούν όπως σε κάθε άλλο πίνακα, δηλαδή paras[0], paras[1], paras[2] κοκ. Εναλλακτικά, μπορεί να χρησιμοποιηθεί η μέθοδος item, δηλαδή paras.item(0), paras.item(1), paras.item(2) κοκ.

Σημειώνεται ότι η μέθοδος getElementsByTagName μπορεί να χρησιμοποιηθεί σε όλο το document αλλά και σε ξεχωριστά στοιχεία.

5.3.2.12 hasChildNodes

Η μέθοδος αυτή χρησιμοποιείται για να ελεγχθεί αν ένα συγκεκριμένο στοιχείο έχει κόμβους-παιδιά ή όχι και είναι της μορφής:

```
booleanValue = element.hasChildNodes
```

Η μέθοδος επιστρέφει μία τιμή τύπου Boolean που είναι είτε true είτε false. Αν το συγκεκριμένο στοιχείο έχει κόμβους-παιδιά τότε η hasChildNode επιστρέφει true ενώ σε διαφορετική περίπτωση επιστρέφει false.
Οι κόμβοι κειμένου και οι ιδιότητες δεν μπορούν να περιέχουν παιδιά. Έτσι, η μέθοδος hasChildNodes όταν εφαρμόζεται σε τέτοια αντικείμενα θα επιστρέφει πάντα την τιμή false.
Η μέθοδος αυτή χρησιμοποιείται συχνά σε μία πρόταση if. Το παρακάτω παράδειγμα βρίσκει ένα στοιχείο με το id “fineprint” και το αποθηκεύει σε μία μεταβλητή με το όνομα message. Αν το στοιχείο αυτό έχει παιδιά, τότε αυτά αποθηκεύονται σε έναν πίνακα με το όνομα children.

```
var message = document.getElementById(“fineprint”);
if (message.hasChildNodes) {
     var children = message.childNodes;
}
```

Η μέθοδος hasChildNodes δεν επιστρέφει τους κόμβους-παιδιά ενός στοιχείου. Οι κόμβοι αυτοί μπορούν να ανακτηθούν με την ιδιότητα childNodes του στοιχείου. Αν η hasChildNodes επιστρέψει τιμή false τότε η ιδιότητα childNodes είναι ένας άδειος πίνακας.
Παρομοίως, αν η hasChildNodes επιστρέψει τιμή false για κάποιο στοιχείο τότε οι ιδιότητες firstChild και lastChild θα είναι null.

5.3.2.13 className

Για την ανάθεση ή αλλαγή του style κάποιου περιεχομένου χρησιμοποιείται η μέθοδος setAttribute και η κλάση που απαιτείται. Για παράδειγμα, έστω ότι υπάρχει η class:

```
.intro {
     color: blue;
     font-size: 1.2em;
}
```

Για να ανανεωθεί το style ενός element χρησιμοποιείται η παρακάτω μέθοδος:

```
element.setAttribute (“class”, “intro”);
```

Επίσης, η χρήση της className γίνεται για να ανακτηθεί η κλάση ενός element και αυτό γίνεται με τη μορφή:

```
element.className
```

Τέλος, η ανάθεση μίας κλάσης σε ένα στοιχείο γίνεται επίσης με το:

```
element.className = value
```

5.3.2.14 insertAfter function

Κανονικά στο DOM δεν υπάρχει μέθοδος insertAfter αλλά αυτή μπορεί να δημιουργηθεί πολύ εύκολα και είναι η παρακάτω:

```
function insertAfter(newElement, targetElement) {
     var parent = targetElement.parentNode;
     if (parent.lastChild == targetElement) {
          parent.appentChild(newElement);
     }
     else {
          parent.insertBefore(newElement,
targetElement.nextsibling);
     }
```

5.3.3 Ιδιότητες των στοιχείων στο DOM

Το τμήμα αυτό περιέχει μία λίστα μερικών από τις σημαντικότερες και πιο χρήσιμες ιδιότητες οι οποίες περιέχοντα στο DOM.

5.3.3.1 nodeName

Η ιδιότητα nodeName επιστρέφει μία συμβολοσειρά η οποία περιέχει το όνομα του συγκεκριμένου κόμβου και η μορφή της είναι:

```
name = node.nodeName
```

Υπάρχουν τρεις περιπτώσεις σχετικά με την ιδιότητα αυτή οι οποίες είναι οι εξής:

- Αν ο συγκεκριμένος κόμβος είναι ένας element node, τότε η ιδιότητα nodeName επιστρέφει το όνομα του στοιχείου. Αυτό είναι ισοδύναμο με την ιδιότητα tagName.S
- Αν ο συγκεκριμένος κόμβος είναι ένας attribute node, τότε η ιδιότητα nodeName επιστρέφει το όνομα της ιδιότητας.
- Αν ο συγκεκριμένος κόμβος είναι ένας text node, τότε η ιδιότητα nodeName επιστρέφει την συμβολοσειρά “#text”.

Η ιδιότητα nodeName είναι μόνο ανάγνωσης και δε μπορεί να τροποποιηθεί απευθείας.

5.3.3.2 nodeType

Η ιδιότητα αυτή επιστρέφει έναν ακέραιο ο οποίος υποδηλώνει τον τύπο του συγκεκριμένου κόμβου. Η μορφή της ιδιότητας αυτής είναι:

```
integer = node.nodeType
```

Υπάρχουν δώδεκα πιθανές τιμές για την ιδιότητα nodeType. Η αριθμητική τιμή που επιστρέφεται από την ιδιότητα αυτή αντιστοιχεί σε έναν από τους δώδεκα τύπους κόμβου που υπάρχουν και είναι οι παρακάτω:

1. ELEMENT_NODE
2. ATTRIBUTE_NODE
3. TEXT_NODE
4. CDATA_SECTION_NODE
5. ENTITY_REFERENCE_NODE
6. ENTITY_NODE
7. PROCESSING_INSTRUCTION_NODE
8. COMMENT_NODE
9. DOCUMENT_NODE
10. DOCUMENT_TYPE_NODE
11. DOCUMENT_FRAGMENT_NODE
12. NOTATION_NODE

Από τους δώδεκα αυτούς τύπους, οι πρώτοι τρεις είναι οι πιο σημαντικοί αφού το μεγαλύτερο μέρος της συγγραφής DOM κώδικα αφορά τον χειρισμό στοιχείων, ιδιοτήτων και κειμένου.

5.3.3.3 nodeValue

Η ιδιότητα αυτή επιστρέφει την τιμή ενός συγκεκριμένου κόμβου σε μορφή συμβολοσειράς και η χρησιμοποιείται ως εξής:

```
value = node.nodeValue
```

Υπάρχουν τρεις περιπτώσεις σχετικά με την ιδιότητα αυτή οι οποίες είναι οι παρακάτω:

- Αν ο συγκεκριμένος κόμβος είναι ένας attribute node, τότε η nodeValue επιστρέφει την τιμή μίας ιδιότητας.
- Αν ο συγκεκριμένος κόμβος είναι ένας text node, τότε η nodeValue επιστρέφει το περιεχόμενου του κόμβου.
- Αν ο συγκεκριμένος κόμβος είναι ένας element node, τότε η nodeValue επιστρέφει null.

Η ιδιότητα nodeValue είναι και ανάγνωσης και εγγραφής. Παρόλα αυτά, δεν υπάρχει η δυνατότητα ανάθεσης τιμής αν αυτή έχει οριστεί ως null. Δηλαδή δεν γίνεται να οριστεί η nodeValue για έναν element node αλλά αυτό γίνεται για έναν text node.

Το παρακάτω παράδειγμα ΔΕΝ θα λειτουργήσει αφού προσπαθεί να αναθέσει τιμή σε έναν element node:

```
var message = document.getElementById("fineprint");
message.nodeValue = "this will not work";
```

Το παρακάτω παράδειγμα είναι ΠΙΘΑΝΟΝ να λειτουργήσει. Προσπαθεί να θέσει μία τιμή για το πρώτο παιδί ενός element node. Αν ο πρώτος κόμβος-παιδί του στοιχείου είναι text node τότε η τιμή θα ανατεθεί:

```
var message = document.getElementById("fineprint");
message.firstChild.nodeValue = "this might work";
```

Το παρακάτω παράδειγμα θα λειτουργήσει ΣΙΓΟΥΡΑ. Το πρώτο που κάνει είναι να ελέγξει ότι το πρώτο παιδί ενός στοιχείου είναι ένας text node:
Η ιδιότητα nodeValue παρέχει τον πιο απλό μηχανισμό για την ανανέωση της τιμής ενός κόμβου κειμένου. Για την ανανέωση ενός τιμής ενός attribute node είναι συνήθως προτιμότερο να χρησιμοποιηθεί η μέθοδος setAttribute στο στοιχείο γονέα.

```
var message = document.getElementById("fineprint");
if (message.firstChild.nodeType == 3) {
     message.firstChild.nodeValue = "this will work";
}
```

5.3.3.4 childNodes

Η ιδιότητα childNodes επιστρέφει έναν πίνακα με τα παιδιά ενός συγκεκριμένου element node και είναι της μορφής:

```
nodeList = node.childNodes
```

Ο πίνακας που επιστρέφεται από τη μέθοδο αυτή μία λίστα κόμβων. Κάθε κόμβος αυτής της λίστας είναι ένα αντικείμενο. Αυτοί οι κόμβοι αντικείμενα έχουν όλες τις ιδιότητες ενός κόμβου όπως είναι οι nodeType, nodeName, nodeValue κτλ.
Οι text και attribute nodes δε μπορούν να περιέχουν παιδιά. Η εφαρμογή της ιδιότητας childNodes σε αυτούς επιστρέφει πάντα έναν άδειο πίνακα.
Για την εύρεση του πλήθους των παιδιών ενός στοιχείου γίνεται χρήση της ιδιότητας length στον πίνακα childNodes ως εξής:

```
node.childNodes.length
```

Αν το στοιχείο έχει μόνο ένα παιδί, τότε η ιδιότητα childNodes θα επιστρέψει και πάλι έναν πίνακα από κόμβους και όχι έναν μόνο κόμβο απλά το μήκος του πίνακα θα είναι 1.
Η ιδιότητα childNodes είναι μόνο ανάγνωσης. Για την προσθήκη ενός κόμβου-παιδιού σε ένα στοιχείο χρησιμοποιούνται οι μέθοδοι appendChild και insertBefore. Για την αφαίρεση ενός κόμβου-παιδιού από ένα στοιχείο

χρησιμοποιείται η μέθοδος removeChild. Αμέσως μετά την χρήση μίας από τις παραπάνω μεθόδων, η ιδιότητα childNodes ανανεώνεται αυτόματα.

5.3.3.5 firstChild

Η ιδιότητα αυτή επιστρέφει το πρώτο παιδί ενός συγκεκριμένου element:

```
reference = node.firstNode
```

Η ιδιότητα αυτή επιστρέφει μία αναφορά σε ένα αντικείμενο κόμβου. Αυτό το αντικείμενο έχει όλες τις ιδιότητες ενός κόμβου όπως είναι οι nodeType, nodeName, nodeValue κτλ.

Οι text και attribute nodes δε μπορούν να περιέχουν παιδιά. Οι δικές τους ιδιότητες firstChild επιστρέφουν πάντα την τιμή null.

Η ιδιότητα firstChild ενός στοιχείου είναι ισοδύναμη με τον πρώτο κόμβο στην λίστα κόμβων childNodes. Δηλαδή:

```
reference = node.childNodes[0]
```

Για την διαπίστωση σχετικά με το αν ένας κόμβος έχει παιδιά κόμβους τότε χρησιμοποιείται η μέθοδος hasChildNodes. Αν ένας κόμβος δεν περιέχει κόμβους-παιδιά τότε η ιδιότητα firstChild επιστρέφει την τιμή null.

Σημειώνεται ότι η ιδιότητα αυτή είναι μόνο ανάγνωσης.

5.3.3.6 lastChild

Η ιδιότητα lastChild επιστρέφει το τελευταίο παιδί ενός συγκεκριμένου element node και είναι της μορφής:

```
reference = node.lastChild
```

Η ιδιότητα αυτή επιστρέφει μία αναφορά σε ένα αντικείμενο κόμβου. Αυτό το αντικείμενο έχει όλες τις ιδιότητες ενός κόμβου όπως είναι οι nodeType, nodeName, nodeValue κτλ.

Οι text και attribute nodes δε μπορούν να περιέχουν παιδιά. Οι δικές τους ιδιότητες lastChild επιστρέφουν πάντα την τιμή null.

Η ιδιότητα firstChild ενός στοιχείου είναι ισοδύναμη με τον πρώτο κόμβο στην λίστα κόμβων lastChild. Δηλαδή:

```
reference = node.childNodes[elementNode.childNodes.length-1]
```

Για την διαπίστωση σχετικά με το αν ένας κόμβος έχει παιδιά κόμβους τότε χρησιμοποιείται η μέθοδος hasChildNodes. Αν ένας κόμβος δεν περιέχει κόμβους-παιδιά τότε η ιδιότητα lastChild επιστρέφει την τιμή null.
Σημειώνεται ότι η ιδιότητα αυτή είναι μόνο ανάγνωσης.

5.3.3.7 nextSibling

Η ιδιότητα nextSibling επιστρέφει τον επόμενο κόμβο που βρίσκεται αμέσως μετά από έναν συγκεκριμένο κόμβο και είναι της μορφής:

```
reference = node.nextSibling
```

Η ιδιότητα αυτή επιστρέφει μία αναφορά σε ένα αντικείμενο κόμβου. Αυτό το αντικείμενο έχει όλες τις ιδιότητες ενός κόμβου όπως είναι οι nodeType, nodeName, nodeValue κτλ.
Αν δεν υπάρχουν κόμβοι που να ακολουθούν αμέσως μετά τον ορισμένο κόμβο, τότε η ιδιότητα nextSibling επιστρέφει την τιμή null.
Σημειώνεται ότι η ιδιότητα αυτή είναι μόνο ανάγνωσης.

5.3.3.8 parentNode

Η ιδιότητα parentNode επιστρέφει τον γονέα ενός συγκεκριμένου κόμβου και είναι της μορφής:

```
reference = node.parentNode
```

Η ιδιότητα αυτή επιστρέφει μία αναφορά σε ένα αντικείμενο κόμβου. Αυτό το αντικείμενο έχει όλες τις ιδιότητες ενός κόμβου όπως είναι οι nodeType, nodeName, nodeValue κτλ.
Ο κόμβος ο οποίος επιστρέφεται είναι πάντοτε ένας element node αφού μόνο element nodes μπορούν να έχουν παιδιά. Η μόνο εξαίρεση είναι αυτή του

document node όπου δεν έχει παιδιά. Σε αυτήν την περίπτωση η parentNode επιστρέφει την τιμή null.
Σημειώνεται ότι η ιδιότητα αυτή είναι μόνο ανάγνωσης.

5.3.3.9 previousSibling

Η ιδιότητα previousSibling επιστρέφει τον προηγούμενο κόμβο αμέσως πριν από έναν συγκεκριμένο κόμβο και είναι της μορφής:

```
reference = node.previousSibling
```

Η ιδιότητα αυτή επιστρέφει μία αναφορά σε ένα αντικείμενο κόμβου. Αυτό το αντικείμενο έχει όλες τις ιδιότητες ενός κόμβου όπως είναι οι nodeType, nodeName, nodeValue κτλ.
Αν δεν υπάρχουν κόμβοι αμέσως πριν από τον ορισμένο κόμβο τότε η previousSibling επιστρέφει την τιμή null.
Σημειώνεται ότι η ιδιότητα αυτή είναι μόνο ανάγνωσης.

5.4 Επίλογος

Ολοκληρώνοντας την παρουσίαση της JavaScript και του DOM έχει ολοκληρωθεί η παράθεση της τεχνολογίας από την πλευρά του browser. Στο επόμενο κεφάλαιο παρουσιάζεται η γλώσσα προγραμματισμού PHP η οποία εκτελείται στη μεριά του server και είναι ο πυρήνας της εφαρμογής που αναπτύσσεται.

ΚΕΦΑΛΑΙΟ 6ο

PHP

6.1 Εισαγωγή

Η PHP (Hypertext Preprocessor) είναι μία ευρέως χρησιμοποιούμενη, ανοιχτού κώδικα και γενικού σκοπού γλώσσα σεναρίου που είναι ειδικά σχεδιασμένη για την ανάπτυξη εφαρμογών διαδικτύου και μπορεί να ενσωματωθεί μέσα σε κώδικα HTML και να εκτελείται κάθε φορά που ο χρήστης επισκέπτεται την σελίδα. Ο PHP κώδικας μεταφράζεται στον Web διακομιστή και δημιουργεί κώδικα HTML ή άλλη έξοδο που θα δει ο επισκέπτης.

Αυτό που διαχωρίζει την PHP από τα client-side JavaScripts είναι ότι ο κώδικας εκτελείται στον server (εξυπηρετητή). Αν υπήρχε ένα script PHP, ο browser θα έπαιρνε τα αποτελέσματα της εκτέλεσης αυτού του script, χωρίς να μπορεί να καταλάβει με κανένα τρόπο τι κώδικας υπάρχει από κάτω. Μπορούμε ακόμα να ρυθμίσουμε τον Web Server ώστε να χειρίζεται όλα τα HTML αρχεία με την PHP.

Αν και η ανάπτυξη της PHP εστιάζεται σε server-side scripting (scripting στην πλευρά του διακομιστή), μπορούνε να γίνουν πολύ περισσότερα με αυτήν.

Σημειώνεται ότι η τελευταία έκδοση της PHP είναι η 5.2.9 και είναι αυτή η οποία χρησιμοποιείται στην εφαρμογή.

6.2 Βασικά χαρακτηριστικά

Η PHP ενσωματώνει την ισχύ και τη δυναμικότητα σχετικά παλαιότερων γλωσσών όπως η Perl αλλά καταργώντας τις αδυναμίες τους. Αναφέρουμε μερικά από τα βασικά χαρακτηριστικά της:

- Ο συντακτικός αναλυτής της, καθώς και ο πηγαίος κώδικάς της διανέμεται ελεύθερα στο διαδίκτυο δίνοντας την δυνατότητα σε όποιον θέλει να κατασκευάζει και να διανέμει εφαρμογές για εμπορική και μη χρήση.
- Μπορεί να μεταφραστεί και να τρέξει στα περισσότερα λειτουργικά συστήματα που κυκλοφορούν στην αγορά (Microsoft Windows, Linux, BSD, Solaris, Macintosh OS X, και UNIX servers).
- Συνεργάζεται χωρίς προβλήματα με τους πιο δημοφιλείς Web Servers που κυκλοφορούν όπως τον Apache και τον Microsoft IIS.

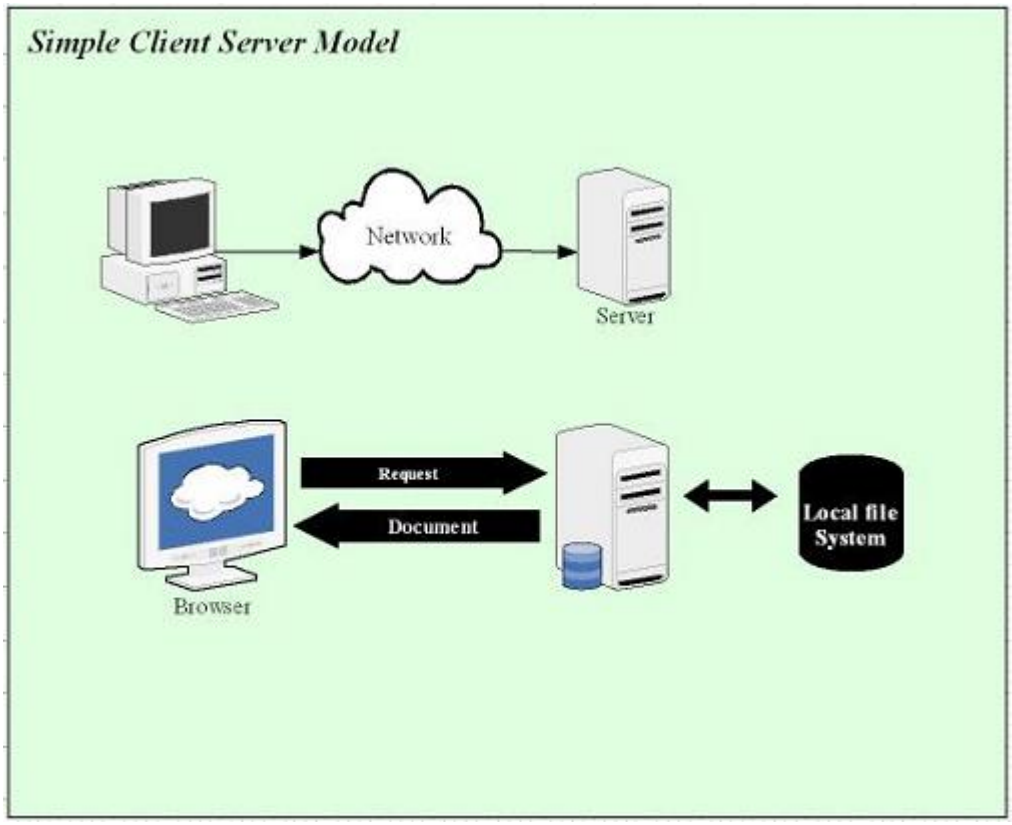
- Διαθέτει ενσωματωμένες εντολές υποστήριξης για ένα μεγάλο αριθμό βάσεων δεδομένων όπως MySQL, Sybase, Oracle, Ingres. Προσφέρει ένα σύνολο από Database API's τις ενοποιημένες ODBC συναρτήσεις (unified ODBC functions), που εξασφαλίζουν την προσπέλαση σε μια υποκείμενη βάση δεδομένων, χρησιμοποιώντας τις εγγενείς μεθόδους της εκάστοτε βάσης για να μεγιστοποιήσουν την απόδοση (IBM DB2).
- Είναι πιο απλό να συντάξει κάποιος κώδικα PHP από ότι σε οποιαδήποτε άλλη γλώσσα σεναρίου.
- Μπορεί να χρησιμοποιηθεί στη δημιουργία εικόνων, ανάγνωση / εγγραφή σε αρχεία και για αποστολή email. Για να προσφέρει αυτές τις υπηρεσίες, η PHP επικοινωνεί με αρκετά πρωτόκολλα όπως: HTTP (Ιστοσελίδες), POP3 (e-mail), SNMP και LDAP.
- Υποστηρίζει τόσο τον διαδικαστικό προγραμματισμό όσο και τον αντικειμενοστραφή.

6.3 Αρχιτεκτονική Βάσης Δεδομένων με PHP – MySQL

Η βασική λειτουργία ενός Web server φαίνεται στο Σχήμα V. Αυτό το σύστημα αποτελείται από δύο αντικείμενα από τα οποία το ένα είναι ο Web browser και το άλλο ο Web server. Απαιτείται μεταξύ τους μία σύνδεση επικοινωνίας. Ένας browser κάνει μία αίτηση στον server κι έπειτα ο server στέλνει πίσω μία απόκριση. Αυτή η αρχιτεκτονική εξυπηρετεί όταν ο διακομιστής παρέχει στατικές σελίδες.

Σημειώνεται ότι οι διακομιστές είναι τα μηχανήματα που προσφέρουν υπηρεσίες ενώ οι πελάτες είναι τα μηχανήματα που ζητούν και δέχονται τις υπηρεσίες αυτές. Ένα μηχάνημα μπορεί να είναι οποιουδήποτε τύπου, ακόμα και των δύο τύπων ταυτόχρονα.

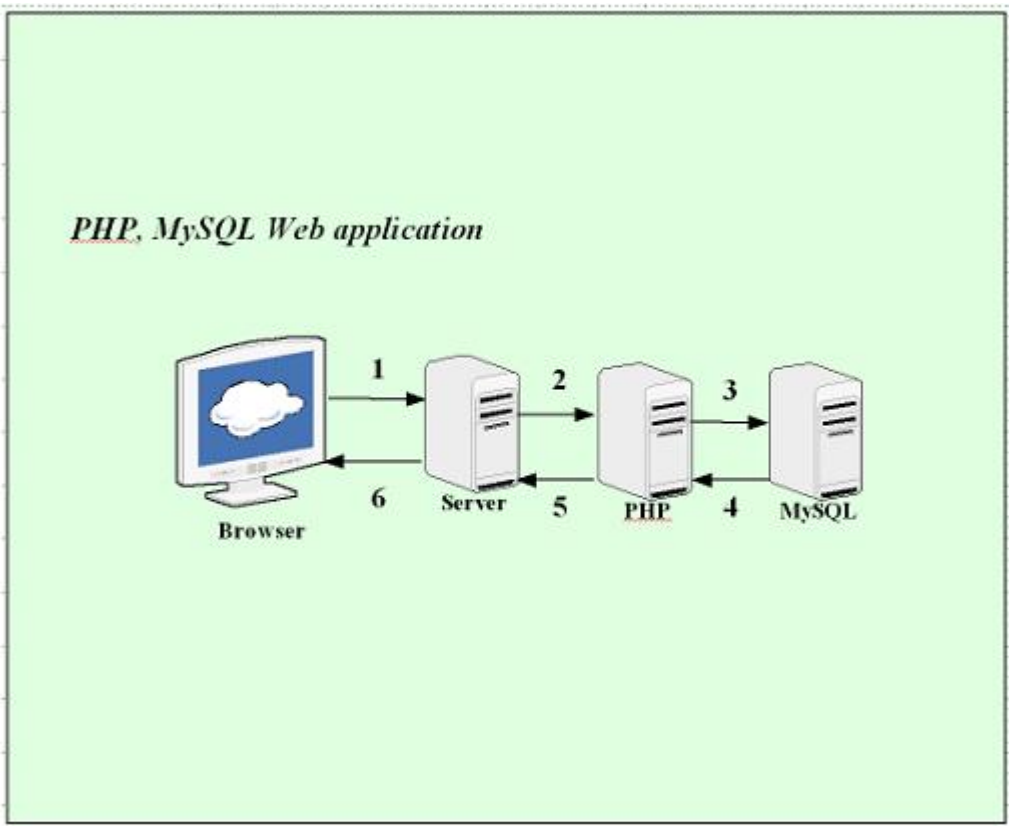
Επίσης, η γλώσσα μορφοποίησης που χρησιμοποιείται για τη δημιουργία ιστοσελίδων, είναι η HTML (Hypertext Markup Language) και το πρωτόκολλο το οποίο χρησιμοποιείται για την μεταφορά των σελίδων από τον διακομιστή στον πελάτη είναι το HTTP (Hypertext Transfer Protocol).



Σχήμα V: Η σχέση browser / server

Η αρχιτεκτονική που υποστηρίζει μία Web τοποθεσία με βάση δεδομένων είναι λίγο πιο περίπλοκη.

Η Web εφαρμογή με βάση δεδομένων που αναπτύσσεται στην παρούσα εργασία ακολουθεί την γενική δομή που φαίνεται στο Σχήμα VI.



Σχήμα VI: Βασική Web αρχιτεκτονική βάσεων δεδομένων

Μία τυπική Web συναλλαγή βάσεων δεδομένων αποτελείται από τις παρακάτω φάσεις:

1. Ο browser ενός χρήστη κάνει μία HTTP αίτηση για μία συγκεκριμένη σελίδα.
2. Ο server λαμβάνει την αίτηση για την συγκεκριμένη σελίδα, ανακαλεί το αρχείο και το περνά στην μηχανή PHP για επεξεργασία.
3. Η PHP μηχανή αρχίζει την ανάλυση του script. Μέσα στον κώδικα, υπάρχει μία εντολή που κάνει την σύνδεση με την βάση δεδομένων και εκτελεί ένα ερώτημα. Η PHP ανοίγει μία σύνδεση με τον MySQL διακομιστή και στέλνει το κατάλληλο ερώτημα.
4. Ο MySQL διακομιστής λαμβάνει το ερώτημα της βάσης δεδομένων, το επεξεργάζεται και στέλνει τα αποτελέσματα ξανά στην PHP μηχανή.
5. Η PHP μηχανή σταματά την εκτέλεση του script, που συνήθως περιλαμβάνει την μορφοποίηση των αποτελεσμάτων του ερωτήματος σε HTML. Μετά, επιστρέφει την τελική HTML σελίδα στον server.

6. Ο server περνά την HTML σελίδα ξανά στον browser, όπου ο χρήστης μπορεί να δει τα αποτελέσματα.

Η διαδικασία είναι βασικά η ίδια, ανεξάρτητα από το ποια μηχανή script ή ποιος server βάσης δεδομένων χρησιμοποιείται. Συνήθως το πρόγραμμα του server, η PHP μηχανή και ο server της βάσης δεδομένων βρίσκονται στον ίδιο υπολογιστή. Ωστόσο, είναι πολύ συνηθισμένο ο server της βάσης δεδομένων να βρίσκεται σε διαφορετικό υπολογιστή. Αυτό μπορεί να γίνει για λόγους ασφάλειας, για μεγαλύτερη χωρητικότητα ή για κατανομή του φόρτου.

6.4 Apache Web Server

Ο Apache Web Server είναι ένας πολύ δημοφιλής διακομιστής διαδικτύου που διανέμεται ελεύθερα στο διαδίκτυο. Αναπτύχθηκε και συντηρείται από μια ομάδα εθελοντών που ήθελαν να υλοποιήσουν έναν εύρωστο κώδικα για διακομιστή δικτύου, που να είναι εμπορικός και να έχει πολλά χαρακτηριστικά. Σήμερα ο Apache θεωρείται από τους πιο σταθερούς διακομιστές δικτύου που κυκλοφορούν και θα πρέπει να τονίσουμε ότι αρκετοί εμπορικοί διακομιστές διαδικτύου, όπως ο HTTP Server της IBM, χρησιμοποιούν τον πυρήνα του Apache.

6.5 Hypertext Transfer Protocol (HTTP)

Το πρωτόκολλο HTTP καθορίζει τον τρόπο επικοινωνίας στο διαδίκτυο μεταξύ των διακομιστών και των πελατών (servers - clients). Είναι ένα γενικό, αντικειμενοστραφές πρωτόκολλο που μεταβιβάζει πληροφορία μεταξύ των διακομιστών και των πελατών. Ξεκίνησε από την έκδοση HTTP/0.9 κατά τη διάρκεια της πρώιμης ανάπτυξης του διαδικτύου και ακολούθησε η έκδοση HTTP/1.0 το 1995. Η πιο πρόσφατη έκδοσή του, HTTP/1.1, παρέχει περισσότερη λειτουργικότητα και υποστηρίζει πολλαπλές συναλλαγές μεταξύ πελάτη και διακομιστή κατά τη διάρκεια της ίδιας αίτησης.

6.6 Εγκατάσταση εργαλείων

Η εγκατάσταση των εργαλείων PHP, MySQL, Apache και PHPMyAdmin γίνεται πλέον πολύ εύκολα κατεβάζοντας ένα και μόνο αρχείο από την ιστοσελίδα http://www.wampserver.com/en/download.php.

6.7 Επίλογος

Με την ολοκλήρωση αυτού του κεφαλαίου, ολοκληρώνεται και το πρώτο μέρος της πτυχιακής. Τώρα που οι τεχνολογίες που θα χρησιμοποιηθούν έχουν γίνει γνωστές, μένει το κομμάτι της υλοποίησης το οποίο και αποτελεί το δεύτερο μέρος του παρόντος συγγράματος. Σημειώνεται ότι η MySQL αναφέρεται στο δεύτερο μέρος επειδή η βάση δεδομένων της εφαρμογής έχει να κάνει και με την ανάλυση.

ΔΕΥΤΕΡΟ ΜΕΡΟΣ

ΚΕΦΑΛΑΙΟ 7ο

ΑΝΑΛΥΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

7.1 Εισαγωγή

Στο στάδιο αυτό αναφέρονται συνοπτικά όλες οι απαραίτητες δραστηριότητες καθώς επίσης και η μεταξύ τους χρονική αλληλουχία ολοκλήρωσης, ώστε το πέρας των δραστηριοτήτων αυτών να σημάνει και την ταυτόχρονη λήξη του έργου. Οι δραστηριότητες είναι πραγματικές και ακολουθήθηκαν με αυτόν τον τρόπο με σκοπό την προσέγγιση και τήρηση ενός οργανωμένου χρονοδιαγράμματος με βάση τις ανάγκες του έργου.

Παράλληλα, γίνεται και η ανάλυση της εφαρμογής διασπώντας τις λειτουργίες που πρέπει να υλοποιηθούν ώστε αυτές να γίνουν πιο κατανοητές αλλά και πιο εύκολα υλοποιήσιμες κατά τον προγραμματισμό.

7.2 Ανάλυση σχεδίου εργασίας έργου και φάσεων ανάπτυξης

Η υλοποίηση του έργου θα γίνει σε έξι στάδια ακολουθώντας όλες τις φάσεις της ανάπτυξης λογισμικού σε μία προκαθορισμένη σειρά η οποία καθορίζεται από το γενικό χρονοδιάγραμμα. Η γενική μεθοδολογία που θα ακολουθηθεί σε κάθε στάδιο βασίζεται στον κύκλο του Deming (Plan – Act – Do – Check). Χαρακτηριστικό είναι ότι τέθηκε σοβαρά ότι το έργο θα υλοποιηθεί από ένα άτομο καθώς επίσης και η μεγάλη ανάγκη παραλληλοποίησης των εργασιών για την καλύτερη εκμετάλλευση του χρόνου.

Η καταγραφή και η συνοπτική ανάλυση των σταδίων αυτών γίνεται ακολούθως:

1. Σχεδίαση Έργου: Αποτελεί το πρώτο στάδιο υλοποίησης του έργου το οποίο αποτελείται από το γενικό πλάνο των ενεργειών που πρέπει να γίνουν καθώς και από την γενική ανάλυση των στοιχείων που θα χρησιμοποιηθούν.
    a. Σχέδιο εργασίας έργου και Επιχειρησιακή Έρευνα: Κατά το πρώτο αυτό στάδιο γίνεται η σχεδίαση όλων των ενεργειών που κρίνεται απαραίτητο να γίνουν για την υλοποίηση του έργου. Επίσης, κατασκευάζεται ένα γενικό πλάνο το οποίο και τεκμηριώνεται με εργαλεία της Επιχειρησιακής Έρευνας ώστε να τεθούν τα χρονικά

περιθώρια αλλά και να γίνει μια πρώτη εκτίμηση για τη στιγμή ολοκλήρωσης του έργου. Με λίγα λόγια, παρέχεται μία καταγραφή σχετικά με το τί πρέπει να γίνει αλλά και πότε.

b. Διερεύνηση υπαρχόντων συστημάτων: Στο σημείο αυτό γίνεται έρευνα για την εύρεση κάποιου παρόμοιου συστήματος ώστε να υπάρξει μία καλύτερη κατανόηση του τί πρέπει να γίνει βάση εμπειριών από ήδη υπάρχουσες εφαρμογές. Επίσης, στο σημείο αυτό βάση των στοιχείων που έχουν συλλεχθεί αποφασίζεται το μοντέλο ανάπτυξης του.
c. Έρευνα απαιτούμενων τεχνολογιών: Περιλαμβάνει την έρευνα σχετικά με τις τεχνολογίες που θα χρησιμοποιηθούν καθώς και την εγκατάσταση των αναλόγων.
d. Βιβλιογραφική ανάλυση των στοιχείων της εφαρμογής: Γίνεται συνοπτική παρουσίαση του έργου προς υλοποίηση, ιστορική αναδρομή, ανάλυση των απαιτούμενων τεχνολογιών και των εφαρμογών που θα χρησιμοποιηθούν αφού αυτές εγκατασταθούν.

2. Ανάλυση: Αυτό είναι το δεύτερο στη σειρά στάδιο στην υλοποίηση του έργου και περιλαμβάνει την συλλογή και ανάλυση των απαιτήσεων της εφαρμογής, δηλαδή του τι είναι επιθυμητό να περιλαμβάνει η εφαρμογή και τι λειτουργίες θα περιέχονται σε αυτήν κι έπειτα βάση αυτών ένα λεπτομερές σχέδιο όλων των οθονών (layout) στο χαρτί ώστε να αποτελέσει τη βάση για την κατασκευή του layout στον υπολογιστή. Στο τέλος έχουμε την ανασκόπηση των λειτουργιών για τη διαπίστωση της πληρότητας των τεθιμένων απαιτήσεων καθώς και για τον έλεγχο του layout.
    a. Συλλογή και ανάλυση απαιτήσεων: Μετά την ολοκλήρωση της διερεύνησης των υπαρχόντων συστημάτων και έχοντας μια ιδέα για τις λειτουργίες τις εφαρμογής αρχίζει η συλλογή και η ανάλυση των απαιτήσεων για την καταγραφή των επιθυμητών λειτουργιών. Η ενέργεια αυτή εκτελείται παράλληλα με την έρευνα των απαιτούμενων τεχνολογιών (1c).
    b. Layout στο χαρτί: Μόλις τελειώσει η συλλογή και ανάλυση των απαιτήσεων και τεθούν οι γενικές απαιτήσεις τότε αρχίζει η σχεδίαση του layout στο χαρτί ώστε να αποτελέσει βάση για την κατασκευή

της ιστοσελίδας με τη βοήθεια του λογισμικού. Η εργασία αυτή είναι παράλληλη με την βιβλιογραφική ανάλυση των στοιχείων της εφαρμογής (1d).

c. Ανασκόπηση λειτουργιών: Στο σημείο αυτό γίνεται μία γενική ανασκόπηση των λειτουργιών που πρόκειται να υλοποιηθούν, τον έλεγχο της τήρησης των απαιτήσεων καθώς και τον έλεγχο της πληρότητας και εφικτής υλοποίησης του layout.

3. Σχεδίαση: Κατά τη φάση της σχεδίασης γίνεται η μετάβαση της ιδεατής μορφής της εφαρμογής στη λογική μορφή της. Δηλαδή αφού έχει καθοριστεί τι πρέπει να γίνει, τώρα καθορίζεται το πώς θα γίνει αυτό στα πλαίσια της εφαρμογής ώστε να καθίσταται λειτουργικό και επιτεύξιμο κατά τη φάση της υλοποίησης. Εδώ περιλαμβάνονται οι κύριες διαδικασίες του λογικού επιπέδου όπως η παράλληλη σχεδίαση στον υπολογιστή όλων των layout μαζί με τη σχεδίαση της βάσης δεδομένων καθώς και της τεκμηρίωσης των λειτουργιών.

   a. Έρευνα: Εφόσον τώρα έχει καθοριστεί τί περίπου λειτουργίες θα υλοποιεί η εφαρμογή είναι αφιερωθεί ένα σημαντικό χρονικό διάστημα για την έρευνα σχετικά με τη χρήση των εργαλείων που θα χρησιμοποιηθούν κατά τη φάση της υλοποίησης. Η έρευνα περιλαμβάνει κυρίως τους τομείς των γλωσσών προγραμματισμού που θα χρησιμοποιηθούν, το τεχνικό μέρος των server και της σύνδεσής τους με βάση δεδομένων και ιστοσελίδων καθώς επίσης και την εξασφάλιση της δυναμικότητας και της συντήρησης της εφαρμογής.

   b. Layout στο Fireworks: Με την ολοκλήρωση της ανασκόπησης των λειτουργιών (2c) δίνεται η δυνατότητα της σχεδίασης του τελικού layout με τη χρήση λογισμικού. Αναφέρουμε ότι το λογισμικό που θα χρησιμοποιηθεί θα είναι το Fireworks. Αυτό το κομμάτι της ανάπτυξης είναι πολύ σημαντικό επειδή από εδώ κρίνεται η αλληλεπίδραση του χρήστη με την εφαρμογή κάτι το οποίο εφόσον σχεδιαστεί είναι εξαιρετικά δύσκολο να αλλάξει στο μέλλον.

   c. Τεκμηρίωση λειτουργιών: Στο στάδιο αυτό γίνεται η τεκμηρίωση των λειτουργιών της εφαρμογής ώστε έπειτα να είναι εύκολη η υλοποίησή τους όσον αφορά το προγραμματιστικό μέρος. Τα

εργαλεία που θα χρησιμοποιηθούν για αυτόν τον σκοπό θα είναι τα Visual Paradigm και SmartDraw. Η εργασία αυτή γίνεται παράλληλα με τη σχεδίαση layout με χρήση λογισμικού (3b) λόγω της συνάφειάς τους.

d. Σχεδίαση βάσης δεδομένων: Το στάδιο αυτό καθορίζει τη λογική δομή της βάσης δεδομένων καθώς και τις σχέσεις που θα διέπουν τα δεδομένα μεταξύ τους. Το κύριο μέλημα είναι η υλοποίηση μίας γρήγορης δυναμικής βάσης δεδομένων ώστε να επιτυγχάνεται η υλοποίηση όλων των λειτουργιών που έχουν οριστεί με όσο το δυνατόν λιγότερο φόρτο κατά την προγραμματιστική ανάπτυξη. Η εργασία αυτή πραγματοποιείται παράλληλα με την σχεδίαση του layout με χρήση λογισμικού (3b) και μόλις ολοκληρωθεί η σχεδίαση της UML (3c).

e. Σχεδίαση SQL queries: Με την ολοκλήρωση της σχεδίασης της βάσης δεδομένων αρχίζει η ανάπτυξη των ερωτημάτων σε SQL με στόχο την ελαχιστοποίηση της πολυπλοκότητας και την πλήρη εκμετάλλευση της βάσης δεδομένων για την επίτευξη της ταχύτητας αφού η ταχύτητα υλοποίησης των ερωτημάτων στη βάση έχει άμεση σχέση με την ταχύτητα απόκρισης της εφαρμογής και την ποιότητα αλληλεπίδρασης του χρήστη με την εφαρμογή.

4. Υλοποίηση: Στο στάδιο αυτό γίνεται η μετάβαση από το λογικό στο φυσικό επίπεδο της ανάπτυξης. Δηλαδή αφού έχει καθοριστεί το τί και το πώς θα επιτευχθεί καθώς και τί μορφή θα έχει η εφαρμογή και τα δεδομένα τώρα γίνεται αυτή καθ’ εαυτή η υλοποίησή τους. Σε αυτή τη φάση ανάπτυξης γίνεται η συγγραφή κώδικα, το στις γλώσσες προγραμματισμού HTML και PHP JavaScript και CSS, καθώς και η υλοποίηση των εναπομείναντων λειτουργιών όπως ο έλεγχος της λειτουργικότητας κάθε λειτουργίας και όλες οι απαραίτητες τροποποιήσεις που θα κριθούν απαραίτητο να γίνουν.

   a. Συγγραφή HTML και PHP κώδικα: Το σημείο αυτό ίσως είναι το πιο δύσκολο σε όλη την εφαρμογή. Η υλοποίησή του προϋποθέτει την δημιουργία του τελικού layout, της βάσης δεδομένων, των SQL queries καθώς και την προηγούμενη κατάλληλη έρευνα για τις συγκεκριμένες γλώσσες. Κάθε λανθασμένη ενέργεια σε κάποιο από τα προηγούμενα θα επιφέρει χρονικό κόστος το οποίο σε αυτό το

σημείο θα απαιτήσει πολύ μεγαλύτερη προσπάθεια αναλογικά με μια πιο έγκαιρη διόρθωσή του. Στην ουσία εδώ υλοποιούνται όλες οι λειτουργίες που θέλουμε να έχει η εφαρμογή και όλος σχεδόν ο κώδικας γράφεται σε αυτό το στάδιο.

   b. Υλοποίηση εναπομείναντων λειτουργιών: Κατά την ολοκλήρωση του προγραμματισμού γίνεται η υλοποίηση τυχόν ενεργειών οι οποίες ήταν αδύνατον να προβλεφθούν ή αφέθηκαν για το τέλος οι οποίες ήταν αδύνατον να γίνουν πιο πριν ώστε η εφαρμογή να είναι πλήρως λειτουργική.

5. Έλεγχος: Στη φάση αυτή ελέγχεται η εφαρμογή και όλες οι λειτουργίες μία προς μία, διορθώνονται τα λάθη και σε διαφορετική περίπτωση επισημαίνονται για προσοχή και μελλοντική διόρθωση.
   a. Έλεγχος εφαρμογής: Ελέγχεται κάθε λειτουργία και τα αποτελέσματα καταγράφονται και συγκρίνονται με αντίστοιχα λογικά ορθά αποτελέσματα. Καταγράφονται όλες οι λειτουργίες για τις οποίες παρατηρείται απόκλιση.
   b. Debugging: Γίνεται η διόρθωση κάθε λειτουργίας που δεν εκτελείται σωστά και σε περίπτωση αδυναμίας διόρθωσης δημιουργείται μία καταγραφή του προβλήματος για μελλοντική αναφορά.

7.3 Τεκμηρίωση με εργαλεία Επιχειρησιακής Έρευνας

Μετά την ανάπτυξη του γενικού σχεδίου έργου και τη σχεδίαση των ενεργειών για την υλοποίησή του πρέπει να γίνει η χρονική και διαδικαστική τεκμηρίωσή του με εργαλεία της Επιχειρησιακής Έρευνας. Τα αποτελέσματα της ανάλυσής αυτής είναι τα ακόλουθα:

1. Σχεδίαση Έργου
   a. Σχέδιο εργασίας έργου και Επιχειρησιακή Έρευνα
   b. Διερεύνηση υπαρχόντων συστημάτων
   c. Έρευνα απαιτούμενων τεχνολογιών
   d. Βιβλιογραφική ανάλυση των στοιχείων της εφαρμογής
2. Ανάλυση
   a. Συλλογή και ανάλυση απαιτήσεων
   b. Layout στο χαρτί

   c. Ανασκόπηση λειτουριών
3. Σχεδίαση
   a. Έρευνα
   b. Layout στο Fireworks
   c. Τεκμηρίωση λειτουργιών
   d. Σχεδίαση βάσης δεδομένων
   e. Σχεδίαση SQL queries
4. Υλοποίηση
   a. Συγγραφή HTML και PHP κώδικα
   b. Υλοποίηση εναπομείναντων λειτουργιών
5. Έλεγχος
   a. Έλεγχος εφαρμογής
   b. Debugging

7.4 Διερεύνηση υπαρχόντων συστημάτων

Ο κύριος στόχος αυτής της δραστηριότητας είναι να μελετηθούν εφαρμογές και συστήματα τα οποία υπάρχουν ήδη στο εμπόριο κι έτσι να γίνει ένας αρχικός καθορισμός των απαιτήσεων και λειτουργιών της εφαρμογής ώστε να αποτελέσει το υπόβαθρο για το στάδιο της σχεδίασης και ανάλυσης. Έτσι, συλλέγονται πολύτιμες πληροφορίες σχετικά με την τεχνική φύση του προβλήματος όπως ποιες είναι οι τεχνολογίες που απαιτούνται, πώς είναι οργανωμένες παρόμοιες εφαρμογές καθώς και τι θέση έχουν στην αγορά. Όλα τα παραπάνω θα βοηθήσουν πάρα πολύ σχετικά με την ανάλυση και σχεδίαση του συστήματος αφού δίνουν μια πρώτη ιδέα για το πώς πρέπει να είναι η εφαρμογή και τι περίπου θα χρειαστεί για την ανάπτυξή της.

7.5 Μελέτη έργου

Στο σημείο αυτό γίνεται μελέτη του κατά πόσο δύσκολη είναι η υλοποίηση ενός τέτοιου έργου και αν η ολοκλήρωσή του είναι εφικτή σε λογικό χρόνο. Έτσι, είναι αναγκαία μία έρευνα αγοράς για τη διαθεσιμότητα, τη μορφή αλλά και το βαθμό δυσκολίας υλοποίησης μίας τέτοιας εφαρμογής όπως το social networking website.

Σημειώνεται ότι πέρα από την αρχική τους λειτουργία οι κοινωνικοί ιστοχώροι έχουν αναγνωριστεί και ως χώροι όπου κάποιος θα μπορούσε να έχει σημαντικά

κέρδη και οφέλη μέσω της ενασχόλησής τους σε αυτούς. Αξίζει να σημειωθεί το παράδειγμα του Facebook όπου ο δημιουργός του είναι σήμερα δισεκατομμυριούχος. Έτσι, για κάποιον ο οποίος θα ασχολούνταν με μία επαγγελματική εφαρμογή είναι απαραίτητη μία αναγνωριστική έρευνα στην αγορά του Ίντερνετ.

Παρατηρήθηκε, όπως ήταν αναμενόμενο άλλωστε, ότι η βασική δομή της εφαρμογής πρέπει να είναι ένα GUI συνδεδεμένο με μία βάση δεδομένων και έναν server. Αυτό το συμπέρασμα προέκυψε κατόπιν αναζήτησης και συνοπτικής ανάλυσης γνωστών κοινωνικών δικτύων όπως το Facebook και το Myspace. Έτσι, οι κύριες γλώσσες προγραμματισμού θα είναι οι HTML, PHP και MySQL, γλώσσες για τις οποίες υπάρχει πλούσιο διαθέσιμο υλικό.

7.6 Επιλογή μοντέλου ανάπτυξης της εφαρμογής

Στο σημείο και αφού πλέον από τη διερεύνηση των υπαρχόντων συστημάτων έχουν καταδειχτεί τα κύρια σημεία του συστήματος, πρέπει να επιλεγεί η μέθοδος σύμφωνα με την οποία θα αναπτυχθεί η εφαρμογή. Αυτό θα γίνει έχοντας υπόψη τον κύκλο του Deming (Plan – Do – Check – Act) σχετικά με την υλοποίηση εφαρμογών και τη διασφάλιση της ποιότητας. Η επιλογή αυτή πρέπει να είναι δεσμευτική μέχρι την περάτωση της εφαρμογής καθώς οποιαδήποτε ασυνέχεια ή ασυμφωνία ενεργειών θα έχει πολύ μεγάλο κόστος σε χρόνο.

Τα μοντέλα αυτά εξετάζονται παρακάτω και η επιλογή γίνεται με γνώμονα το μέγεθος της ομάδας, τον προσδοκώμενο χρόνο ολοκλήρωσης του έργου καθώς και την φύση της εφαρμογής η οποία δεν πρέπει να ξεχνάμε ότι είναι πτυχιακή εργασία. Εδώ πρέπει να σημειώσουμε ότι κανένα μοντέλο δεν είναι το απόλυτο και η χρήση του κάθε ένα εξαρτάται από τη φύση της εφαρμογής και των χρηστών για τους οποίους προορίζεται. Ακολούθως αναλύονται οι δυνατές επιλογές.

7.6.1 Build – and – Fix model

Το πρότυπο αυτό χαρακτηρίζεται από την extreme programming προσέγγισή του δηλαδή την κατασκευή της εφαρμογής χωρίς προδιαγραφές και σχέδιο. Η συγγραφή του κώδικα είναι το πρώτο στάδιο της υλοποίησης χωρίς να προηγούνται τα στάδια της ανάλυσης και σχεδίασης. Η προσέγγιση αυτή μπορεί να λειτουργήσει αποδοτικά για προγράμματα μικρού μεγέθους, 200 – 500

γραμμών κώδικα, αλλά φυσικά είναι απαράδεκτη για μία εφαρμογή όπως αυτήν που εξετάζουμε τώρα.

7.6.2 Rapid Prototyping model

Το μοντέλο αυτό δημιουργεί σε μικρό χρονικό διάστημα μία μικρογραφία της εφαρμογής με τις κυριότερες λειτουργίες της και δίνει στους χρήστες τη δυνατότητα να τις δοκιμάσουν και να πειραματιστούν με αυτές. Όταν οι χρήστες δώσουν την έγκρισή τους για το πρωτότυπο τότε γίνεται η τεκμηρίωση των προδιαγραφών. Το πλεονέκτημα αυτής της μεθοδολογίας στην περίπτωσή μας είναι ότι δίνεται η δυνατότητα για μία γρήγορη δημιουργία μίας μικρής λειτουργικής εφαρμογής κάτι το οποίο εκτός από μία πρώτη εικόνα όλου του project δίνει και ψυχολογικά κίνητρα στους κατασκευαστές αφού δίνει την άμεση εντύπωση της δημιουργίας. Το πρόβλημα είναι όμως ότι το πρωτότυπο που θα αναπτυχθεί κατά την πρώτη φάση μπορεί να διαφέρει εντελώς από την τελική εφαρμογή αν οι αρχικές και οι τελικές απαιτήσεις απέχουν αρκετά. Αυτό θα δυσκολέψει πάρα πολύ τη διαδικασία της επανασχεδίασης αφού πιθανόν θα πρέπει να γίνει τροποποίηση στο layout, στη βάση δεδομένων ή και στα δύο οπότε αυτό έχει μεγάλο κόστος όσον αφορά τον χρόνο υλοποίησης. Για τον λόγο αυτό η προσέγγιση αυτή αυτόν απορρίπτεται.

7.6.3 Incremental model

Με τη μέθοδο αυτή το τελικό προϊόν δημιουργείται από τη συνένωση ξεχωριστών κομματιών κώδικα και λειτουργιών που αλληλεπιδρούν μεταξύ τους. Έτσι, η υλοποίηση χωρίζεται σε στάδια στο τέλος των οποίων έχει αναπτυχθεί ένα λειτουργικό σύνολο το οποίο όμως είναι υποσύνολο όλων των απαιτήσεων.
Η δυσκολία εδώ επαφίεται στο ότι κάθε πρόσθετη λειτουργία και ενότητα πρέπει να ενσωματωθεί στην ήδη υπάρχουσα δομή με τις ελάχιστες δυνατές τροποποιήσεις.

7.6.4 Waterfall model

Σύμφωνα με το μοντέλο αυτό, πρώτα γίνεται ο πλήρης καθορισμός των απαιτήσεων και των λειτουργιών της εφαρμογής, έπειτα καθορίζεται τι ακριβώς πρέπει να κάνει η εφαρμογή και μετά καθορίζεται ένα σχέδιο με την περιγραφή του πως η εφαρμογή πρόκειται να τα κάνει. Όταν καθοριστούν αυτά αρχίζει η

υλοποίηση κατά τη διάρκεια της οποίας υπάρχει η δυναμική δυνατότητα επαναπροσδιορισμού των απαιτήσεων και του κώδικα ώστε οποιεσδήποτε αλλαγές και βελτιώσεις να είναι εφικτές. Το πρόβλημα εδώ είναι η μικρή απόκλιση που πρέπει να υπάρχει στις αρχικές και τελικές απαιτήσεις. Έτσι είναι πολύ δύσκολο να ξέρουμε από την αρχή τί ακριβώς λειτουργίες θα χρειαστούμε.

7.6.5 Extreme prototyping

Η μέθοδος αυτή χρησιμοποιείται κυρίως για την ανάπτυξη web εφαρμογών. Το κύριο στοιχείο της είναι ότι η ανάπτυξη της εφαρμογής διαιρείται σε τρεις φάσεις κάθε μία από τις οποίες βασίζεται στην προηγούμενη. Η πρώτη φάση είναι ένα στατικό πρωτότυπο το οποίο αποτελείται κυρίως από σελίδες HTML ή το γραφικό μέρος του layout της εφαρμογής. Κατά τη δεύτερη φάση, προγραμματίζονται όλες οι διεπαφές χρήστη ώστε να είναι πλήρως λειτουργικές. Στην τρίτη φάση υλοποιούνται οι υπηρεσίες και οι λειτουργίες που πρέπει να ικανοποιούν τις προδιαγραφές. Με το μοντέλο αυτό ο χρόνος καταγραφής απαιτήσεων καθώς και ο χρόνος ανάπτυξης μπορεί να μειωθεί κατά πολύ ενώ οι εργασίες μπορούν να διασπαστούν και να υλοποιηθούν από πολλές ομάδες που εργάζονται παράλληλα. Άλλο πλεονέκτημα είναι ότι δίνεται η δυνατότητα άμεσης οπτικής αναπαράστασης όλης της εφαρμογής καθώς και το ότι η συγγραφή κώδικα μπορεί να αρχίσει αρκετά νωρίς χωρίς αυτό να προκαλέσει προβλήματα.
Παρατηρείται λοιπόν ότι το μοντέλο αυτό δίνει τη δυνατότητα για:

- Γρήγορη καταγραφή των απαιτήσεων.
- Άμεση οπτική αναπαράσταση της εφαρμογής.
- Γρήγορο ξεκίνημα συγγραφής κώδικα.
- Δυνατότητα επανατροφοδότησης (feedback) και διόρθωσης.
- Παράλληλη εργασία.

Για όλους τους παραπάνω λόγους και λαμβάνοντας υπόψη τη φύση της εφαρμογής η οποία είναι web-oriented καθώς και ότι το έργο θα επιτελεστεί από ένα άτομο, έχει επιλεχθεί το μοντέλο αυτό για την υλοποίηση του έργου. Για τον λόγο αυτό, παρακάτω αναλύονται συνοπτικά τα κύρια στάδια του μοντέλου αυτού.

7.6.5.1 Πρώτη Φάση - Static Prototype phase

- Κατασκευή των κύριων σελίδων.

- Δημιουργία των CSS αρχείων.
- Υλοποίηση λογικών μοντέλων δεδομένων για την υποστήριξη των διεπαφών (screens).

7.6.5.2 Δεύτερη Φάση - Dynamic prototype phase

- Δυνατότητα πλήρης περιήγησης στην εφαρμογή.
- Χρήση κώδικα όπου χρειάζεται (JavaScript).
- Διανομή όλων των λειτουργιών στα αντίστοιχα πεδία των σελίδων.
- Δυνατότητα ενός πλήρους λειτουργικού UI αλλά χωρίς λειτουργικότητα.

7.6.5.3 Τρίτη Φάση - Service Implementation phase

- Application Program Interface (API) document.
- Υλοποίηση κάθε λειτουργίας με τη σύνδεσή της σε βάση δεδομένων ή σε ό,τι άλλους πόρους απαιτείται.

7.6.5.3.1 API document

Εδώ πρέπει να ειπωθούν δύο λόγια για το API document το οποίο αποτελείται από:

1. Μία λίστα όλων των διεπαφών (screen) του συστήματος.
2. Κάθε διεπαφή περιέχει:
   2.1. Μία εικόνα για κάθε οθόνη
   2.2. Μία λίστα με τις αρμοδιότητες κάθε οθόνης ή το πώς λειτουργεί κάθε οθόνη
   2.3. Μία λίστα των API για εκείνη την οθόνη
3. Κάθε API περιέχει:
   3.1. Όνομα
   3.2. Την αρμοδιότητα ή την λειτουργία του API
   3.3. Δεδομένα που εισάγονται
   3.4. Δεδομένα που εξάγονται

Το έγγραφο αυτό μπορεί να χρησιμοποιηθεί ώστε οι σχεδιαστές του UI να έχουν μια ιδέα για το πώς θα είναι οι οθόνες και για το τι ακριβώς χρειάζεται η εφαρμογή. Επίσης, το έγγραφο αυτό μπορεί να χρησιμοποιηθεί αντί του εγγράφου περιπτώσεων χρήσης.

Παρακάτω παρατίθεται το έγγραφο αυτό χωρίς όμως τις εικόνες για κάθε διεπαφή αφού αυτές παρουσιάζονται στην αναρτημένη εφαρμογή στην διεύθυνση: http://kostptyx.net76.net.

7.6.5.3.2 Λειτουργίες της εφαρμογής

Παρακάτω παρατίθεται η λίστα με τις κυριότερες διεπαφές της εφαρμογής καθώς και οι κυριότερες λειτουργίες κάθε μίας από αυτές:

1. Αρχική σελίδα index
    a. Σύνδεση
    b. Εγγραφή
    c. Ανάκτηση κωδικού
    d. Footer (όπως και σε κάθε σελίδα της εφαρμογής)
2. Σελίδα μετά τη σύνδεση ή εγγραφή
    a. Home
    b. Profile
    c. Φίλοι
    d. Μηνύματα
    e. Αναζήτηση
    f. Αρχεία
    g. Προκηρύξεις
    h. Ρυθμίσεις
    i. Αποσύνδεση
3. Home
    a. Header (όπως και σε κάθε σελίδα της εφαρμογής όσο ο χρήστης είναι συνδεδεμένος)
    b. Εμφάνιση φωτογραφίας
    c. Συνδεδεμένα μέλη
    d. Συνδεδεμένοι φίλοι
    e. Εμφάνιση τυχαίων φίλων
    f. Αναγγελία νέων μηνυμάτων
    g. Αναγγελία νέων αιτήσεων φιλίας
    h. Αναγγελία ανάρτησης νέων αρχείων
    i. Αναγγελία ανάρτησης αρχείων του ίδιου κλάδου

j. Αναγγελία αλλαγών στοιχείων φίλων (profile, όνομα, email)
k. Αναγγελία δημιουργίας νέας φιλίας ανάμεσα σε φίλους
l. Εξατομικευμένη διαφήμιση (με κριτήριο τον κλάδο του μέλους)

4. Profile
    a. Εμφάνιση
        1. Εμφάνιση φωτογραφίας
        2. Εμφάνιση τύπου μέλους (golden, silver, null)
        3. Εμφάνιση αριθμού αρχείων που ανάρτησε
        4. Συνδεδεμένα μέλη
        5. Συνδεδεμένοι φίλοι
        6. Εμφάνιση τυχαίων φίλων
        7. Εμφάνιση πληροφοριών profile χρήστη
    b. Επεξεργασία
        1. Επιλογή φωτογραφίας
        2. Επιλογή προεπιλεγμένης φωτογραφίας
        3. Εμφάνιση μενού με τις πληροφορίες του χρήστη και δυνατότητα αλλαγής αυτών (εκτός ονόματος και email)
        4. Αποθήκευση αλλαγών
        5. Ακύρωση αλλαγών
5. Φίλοι
    a. Οι φίλοι σου
        1. Συνδεδεμένοι φίλοι
        2. Εμφάνιση πλήθους φίλων
        3. Εμφάνιση όλων των φίλων με σελιδοποίηση
            1. Στοιχεία φίλου
            2. Λίστα φίλων
            3. Αποστολή μηνύματος
        4. Επιλογή και διαγραφή φίλων
    b. Αναζήτηση φίλων
        1. Αναζήτηση με κριτήρια:
            1. Όνομα
            2. Email
            3. Κλάδος
            4. Οι πιο πρόσφατοι

5. Συνδεδεμένοι Τώρα
2. Εμφάνιση αποτελεσμάτων:
    1. Στοιχεία φίλου
    2. Λίστα Φίλων
    3. Αποστολή Μηνύματος

6. Μηνύματα
    a. Εισερχόμενα
        1. Όλα
            1. Εμφάνιση πλήθους μηνυμάτων, πλήθος χρηστών από όπου αυτά προέρχονται και πλήθος μη αναγνωσμένων
        2. Αναγνωσμένα
            1. Εμφάνιση πλήθους μηνυμάτων
        3. Μη αναγνωσμένα
            1. Εμφάνιση πλήθους μηνυμάτων
        4. Εμφάνιση μηνυμάτων με σελιδοποίηση
            1. Στοιχεία αποστολέα
            2. Στοιχεία μηνύματος
            3. Λίστα Φίλων αποστολέα
            4. Αποστολή μηνύματος στον αποστολέα
            5. Επιλογή εμφάνισης μηνύματος
        5. Επιλογή και διαγραφή μηνυμάτων
    b. Δημιουργία Νέου
        1. Επιλογή παραλήπτη
        2. Στοιχεία μηνύματος
        3. Αποστολή
        4. Ακύρωση
    c. Απεσταλμένα
        1. Εμφάνιση πλήθους μηνυμάτων και πλήθος χρηστών όπου αυτά απευθύνονται
        2. Εμφάνιση μηνυμάτων με σελιδοποίηση
            1. Στοιχεία παραλήπτη
            2. Στοιχεία μηνύματος
            3. Λίστα Φίλων παραλήπτη
            4. Αποστολή μηνύματος στον παραλήπτη

5. Επιλογή εμφάνισης μηνύματος
3. Επιλογή και διαγραφή μηνυμάτων

7. Αναζήτηση
    a. Αναζήτηση Ατόμων
        1. Εμφάνιση των δέκα πιο πρόσφατων νέων μελών
        2. Αναζήτηση Ατόμων με κριτήρια:
            1. Όνομα
            2. Email
            3. Κλάδο
            4. Συνδεδεμένοι Τώρα
            5. Με φωτογραφία
        3. Εμφάνιση αποτελεσμάτων
            1. Στοιχεία μέλους
            2. Λίστα Φίλων
            3. Αποστολή μηνύματος
            4. Αίτηση Φιλίας
    b. Αναζήτηση Αρχείων (ίδιο με το Αρχεία)
    c. Αναζήτηση Φίλων (ίδιο με το Φίλοι)
    d. Αναζήτηση Προκήρυξης (ίδιο με το Προκηρύξεις)
8. Αρχεία
    a. Ανάρτηση Αρχείου
        1. Επιλογή αρχείου
        2. Εισαγωγή στοιχείων του αρχείου
        3. Ανάρτηση του αρχείου
        4. Εμφάνιση όλων των αναρτημένων αρχείων ταξινομημένων και με σελιδοποίηση
            1. Στοιχεία μέλους που ανάρτησε το αρχείο
            2. Στοιχεία αρχείου
            3. Download αρχείου
    b. Αναζήτηση Αρχείων
        1. Αναζήτηση αρχείων με κριτήρια:
            1. Όνομα αρχείου
            2. Θέμα αρχείου
            3. Κλάδο

4. Ημερομηνία ανάρτησης
5. Αναρτημένα από φίλους

2. Αναζήτηση
3. Εμφάνιση όλων των αποτελεσμάτων με σελιδοποίηση
    1. Στοιχεία μέλους που ανάρτησε το αρχείο
    2. Στοιχεία αρχείου
    3. Download αρχείου

9. Προκηρύξεις
    a. Προκηρύξεις
        1. Εμφάνιση πλήθους αναρτημένων προκηρύξεων με σελιδοποίηση
        2. Εμφάνιση προκηρύξεων
            1. Στοιχεία προκήρυξης
            2. Συνοδευτικά αρχεία
            3. Επιλογή προκήρυξης
    b. Αναζήτηση προκήρυξης
        1. Αναζήτηση με κριτήρια:
            1. Ημερομηνία
        2. Αναζήτηση
        3. Εμφάνιση όλων των αποτελεσμάτων με σελιδοποίηση
            1. Στοιχεία προκήρυξης
            2. Συνοδευτικά αρχεία
            3. Επιλογή προκήρυξης
10. Ρυθμίσεις
    a. Επεξεργασία Λογαριασμού
        1. Αλλαγή ονόματος
        2. Αλλαγή email
        3. Αλλαγή κωδικού
        4. Διαγραφή λογαριασμού
            1. Επιβεβαίωση διαγραφής λογαριασμού
            2. Ακύρωση διαγραφής λογαριασμού
    b. Προσωπικές ρυθμίσεις
        1. Επιλογή προτίμησης προβολής profile
        2. Επιλογή προτίμησης προβολής φωτογραφίας

3. Επιλογή προτίμησης προβολής στοιχείων επικοινωνίας
4. Επιλογή προτίμησης αποδοχής μηνυμάτων
5. Επιλογή προτίμησης προβολής στην Αναζήτηση Ατόμων
6. Φραγή Μέλους
7. Αφαίρεση Φραγής
8. Αποθήκευση
9. Ακύρωση

11. Αποσύνδεση
    a. Διαγραφή του μέλους από τους συνδεδεμένους
    b. Τερματισμός του session
    c. Επιστροφή στην αρχική σελίδα index

7.6.5.3.3 Συμπληρωματικές Οθόνες

Στην κατηγορία αυτή ανήκουν όλες οι οθόνες εκείνες οι οποίες δεν υπάρχουν στα κύρια μενού αλλά ο χρήστης τις επιλέγει μέσα από ένθετα μενού και επιλογές. Αυτές είναι:

1. Προβολή profile άλλου μέλους
    c. Εμφάνιση φωτογραφίας
    d. Εμφάνιση τύπου μέλους (golden, silver, null)
    e. Αποστολή μηνύματος
    f. Λίστα Φίλων
    g. Αίτηση φιλίας
    h. Εμφάνιση αριθμού αρχείων που ανάρτησε
    i. Κοινοί φίλοι
    j. Εμφάνιση τυχαίων φίλων
    k. Εμφάνιση πληροφοριών profile χρήστη
2. Αποστολή Μηνύματος
    a. Επιλεγμένος παραλήπτης
    b. Φωτογραφία παραλήπτη
    c. Στοιχεία μηνύματος
    d. Αποστολή
    e. Ακύρωση
3. Λίστα Φίλων

a. Εμφάνιση πλήθους φίλων του χρήστη
b. Εμφάνιση πλήθους κοινών φίλων
c. Εμφάνιση όλων των φίλων με σελιδοποίηση
    1. Στοιχεία φίλου
    2. Λίστα Φίλων
    3. Αποστολή Μηνύματος
    4. Αίτηση Φιλίας

4. Επιλογή Μηνύματος
    a. Επόμενο – προηγούμενο μήνυμα
    b. Στοιχεία αποστολέα – παραλήπτη
    c. Στοιχεία μηνύματος
    d. Διαγραφή μηνύματος
    e. Απάντηση μηνύματος
        i. Επιλεγμένος παραλήπτης
        ii. Στοιχεία μηνύματος
        iii. Αποστολή
        iv. Ακύρωση
5. Επιλογή Προκήρυξης
    a. Ημερομηνία ανάρτησης
    b. Στοιχεία προκήρυξης
6. Αίτηση Φιλίας
    a. Εμφάνιση πλήθους αιτήσεων φιλίας
    b. Αιτήσεις Φιλίας με σελιδοποίηση
        i. Στοιχεία αιτούντος χρήστη
        ii. Κοινοί φίλοι
        iii. Λίστα φίλων
        iv. Αποστολή μηνύματος
        v. Αποδοχή
        vi. Απόρριψη
7. Κοινοί Φίλοι
    a. Εμφάνιση πλήθους κοινών φίλων
    b. Εμφάνιση πλήθους φίλων του χρήστη
    c. Εμφάνιση κοινών φίλων με σελιδοποίηση
        i. Στοιχεία χρήστη

ii. Λίστα Φίλων
iii. Αποστολή Μηνύματος

7.6.5.3.4 Δυνατότητες της εφαρμογής

Η παρούσα εφαρμογή με την ολοκλήρωσή της παρέχει τις εξής υπηρεσίες:

- Δυνατότητα εγγραφής νέου μέλους
- Δυνατότητα προβολής μέλους μέσω εισαγωγής στοιχείων του μέλους
- Δυνατότητα επεξεργασίας των στοιχείων του μέλους
- Δυνατότητα γνωριμίας μεταξύ των μελών
- Δυνατότητα δημιουργίας φιλίας
- Δυνατότητα αναζήτησης μελών
- Δυνατότητα αποστολής μηνυμάτων μεταξύ των μελών
- Δυνατότητα ανάρτησης αρχείων
- Δυνατότητα download αρχείων
- Δυνατότητα αναζήτησης αρχείων
- Δυνατότητα αναζήτησης προκηρύξεων
- Δυνατότητα διασφάλισης προσωπικών δεδομένων
- Δυνατότητα ηθελημένης διαγραφής μέλους
- Δυνατότητα διαγραφής αρχείων από τον διαχειριστή
- Δυνατότητα διαγραφής μελών από τον διαχειριστή
- Δυνατότητα ανάρτησης προκηρύξεων από τον διαχειριστή
- Δυνατότητα διαγραφής προκηρύξεων από τον διαχειριστή

7.7 Επίλογος

Η ανάλυση είναι πολύ σημαντικό στοιχείο κατά την διαδικασία της υλοποίησης μίας εφαρμογής αφού σε αυτήν θέτονται οι απαιτήσεις και οι λειτουργίες του προγράμματος που πρόκειται να συγγραφεί. Μάλιστα, σε περίπτωση που υπάρξει κάποιο λάθος κατά τη διάρκεια αυτής της φάσης, τότε η συντήρηση της εφαρμογής γίνεται πολύ δύσκολη.

Έτσι, έχοντας ολοκληρώσει αυτό το κομμάτι, στο επόμενο κεφάλαιο παρουσιάζεται η βάση δεδομένων που θα χρησιμοποιηθεί για τις ανάγκες της εφαρμογής.

ΚΕΦΑΛΑΙΟ 8ο

ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ

8.1 Εισαγωγή

Στο κεφάλαιο αυτό παρουσιάζεται η ανάλυση της βάσης δεδομένων που χρησιμοποιήθηκε στην εφαρμογή καθώς και η τεκμηρίωσή της παραθέτοντας τους σχετικούς πίνακες και διαγράμματα.

8.2 Τι είναι MySQL

Η MySQL είναι ένα πολύ γρήγορο δυνατό, σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων. Ακολουθεί το σχεσιακό μοντέλο (relational) και είναι συμβατή με ANSI-SQL. Ο MySQL διακομιστής ελέγχει την πρόσβαση στα δεδομένα ώστε να διασφαλίσει ότι πολλοί χρήστες θα μπορούν να δουλεύουν ταυτόχρονα, ώστε να επιτυγχάνεται γρηγορότερη πρόσβαση και να διασφαλιστεί η πιστοποίηση των χρηστών. Συνεπώς, η MySQL είναι ένας πολυνηματικός διακομιστής πολλαπών χρηστών. Χρησιμοποιεί την SQL (Structured Query Language), την τυπική γλώσσα ερωτημάτων για βάσεις δεδομένων, παγκόσμια. Η MySQL είναι διαθέσιμη από το 1996 αλλά η ιστορία της ξεκινάει από το 1979. Είναι παγκοσμίως η πιο δημοφιλής βάση δεδομένων ανοικτού κώδικα. Επίσης, η έκδοση MySQL Server είναι κατάλληλη για τη διαχείριση μεγαλύτερων βάσεων δεδομένων και την παρουσίαση δεδομένων σε web site στο internet. Παρέχει υψηλή απόδοση και ασφάλεια.

8.3 Κανονικοποίηση

Η θεωρία της κανονικοποίησης αναπτύχθηκε με στόχο να ανιχνεύεται και να προλαμβάνεται προβληματικές καταστάσεις όταν η εφαρμογή είναι ακόμη στο στάδιο του σχεδιασμού της. Τέτοιες καταστάσεις είναι διπλοεγγραφή δεδομένων, ασυνεπή δεδομένα κ.ά.

8.3.1 Μη Κανονική Μορφή

Στη μη κανονική μορφή δεν υπάρχει ακόμα πίνακα με την σχεσιακή έννοια. Τα πεδία τοποθετούνται το ένα δίπλα στο άλλο με μόνο επιπλέον στοιχείο πληροφορίας την ομαδοποίηση πεδίων που αντιστοιχούν σε χαρακτηριστικά που παίρνουν πολλαπλές τιμές. Η ομαδοποίηση των τελευταίων αναπαριστάται με

ζεύγη εσωτερικών παρενθέσεων στην περιγραφή της δομής του αντίστοιχου πίνακα.

8.3.2 Πρώτη Κανονική Μορφή (1NF)

Για να έρθει ένας πίνακας σε πρώτη κανονική μορφή, θα πρέπει να απομακρυνθούν οι επαναλαμβανόμενες ομάδες πεδίων, έτσι ώστε η τομή μιας γραμμής και μιας στήλης του πίνακα, να αντιστοιχεί πάντα σε μια απλή τιμή. Η πρώτη κανονική μορφή δημιουργήθηκε για να αποτρέψει την εμφάνιση στα πεδία ενός πίνακα σύνθετων τιμών, πολλαπλών τιμών καθώς και συνδυασμούς αυτών των δύο. Ένας πίνακας βρίσκεται σε 1NF, όταν η τιμή του κάθε πεδίου σε κάθε εγγραφή, είναι ατομική δηλαδή δεν μπορεί να διασπαστεί σε μικρότερες μονάδες πληροφορίας. Δηλαδή δεν επιτρέπεται να υπάρχουν πεδία στον πίνακα που να παίρνουν πολλαπλές τιμές.

8.3.3 Δεύτερη Κανονική Μορφή (2NF)

Για να έρθει ένας πίνακας σε δεύτερη κανονική μορφή, θα πρέπει πρώτα να είναι σε πρώτη κανονική μορφή και στη συνέχεια να απομακρυνθούν όλες οι μερικές συναρτησιακές εξαρτήσεις (partial dependencies) που υφίστανται ανάμεσα στα πεδία του. Η δεύτερη κανονική μορφή προκύπτει από την πρώτη κανονική μορφή εάν μετασχηματιστεί η δομή του πίνακα με τέτοιο τρόπο ώστε να απομακρυνθούν όλες οι μερικές συναρτησιακές εξαρτήσεις (partial dependencies) που υφίστανται ανάμεσα στα πεδία του. Με άλλα λόγια ένας πίνακας βρίσκεται σε δεύτερη κανονική μορφή όταν όλα τα πεδία που δεν ανήκουν στο πρωτεύον κλειδί του πίνακα εξαρτώνται συναρτησιακώς μόνο από τα πεδία του πρωτεύοντος κλειδιού και μάλιστα, μέσω πλήρους συναρτησιακής εξάρτησης (full dependency). Επειδή ο ορισμός της πλήρους και της μερικής συναρτησιακής εξάρτησης μεταξύ δύο πεδίων Χ και Υ εφαρμόζεται μόνο όταν το Χ περιλαμβάνει περισσότερα από ένα πεδία, είναι προφανές πως η αναγωγή ενός πίνακα στη δεύτερη κανονική μορφή, έχει νόημα μόνο όταν το πρωτεύον κλειδί του είναι σύνθετο. Αντίθετα, όταν το κλειδί του πίνακα είναι ένα απλό πεδίο, η αναγωγή του πίνακα σε 1NF τον μετασχηματίζει αυτόματα και σε 2NF. Ένας πίνακας είναι σε δεύτερη κανονική μορφή όταν είναι σε πρώτη κανονική μορφή και επιπλέον το κάθε πεδίο του που δεν συμμετέχει στο σχηματισμό του κύριου κλειδιού εξαρτάται από το σύνολο του κυρίως κλειδιού και τίποτα λιγότερο.

8.3.4 Τρίτη Κανονική Μορφή

Για να έρθει ένας πίνακας σε τρίτη κανονική μορφή, θα πρέπει πρώτα να τον είναι σε δεύτερη κανονική μορφή, και στη συνέχεια να απομακρυνθούν όλες οι μεταβατικές συναρτησιακές εξαρτήσεις (transitive dependencies) που υφίστανται ανάμεσα στα πεδία του. Η τρίτη κανονική μορφή προκύπτει από τη δεύτερη κανονική μορφή εάν ο πίνακας μετασχηματίστεί με τέτοιον τρόπο ώστε να απομακρυνθούν όλες οι μεταβατικές συναρτησιακές εξαρτήσεις (transitive dependencies) που υφίστανται ανάμεσα στα πεδία του. Σημειώνεται ότι μια συναρτησιακή εξάρτηση A→B ονομάζεται μεταβατική, όταν υπάρχει ένα πεδίο C που δεν ανήκει στο πρωτεύον κλειδί του πίνακα, τέτοιο ώστε C = C(A) και B = B(C). Ένας πίνακας είναι σε τρίτη κανονική μορφή όταν είναι σε δεύτερη κανονική μορφή και επιπλέον δεν υπάρχει πεδίο που να μην συμμετέχει στο σχηματισμό του κύριου κλειδιού το οποίο να εξαρτάται συναρτησιακά από άλλο πεδίο που επίσης δεν συμμετέχει στο σχηματισμό του κύριου κλειδιού. Εξαίρεση είναι το εναλλακτικό κλειδί όταν αυτό υπάρχει.

Παρακάτω, γίνεται η ανάλυση του κάθε πίνακα της εφαρμογής ξεχωριστά κι έπειτα παρουσιάζεται η ολοκληρωμένη βάση δεδομένων στην τρίτη κανονική μορφή της.

8.4 Σχεσιακό Μοντέλο

Το σχεσιακό μοντέλο αναπαράστασης των δεδομένων μίας εφαρμογής (relational data model) καθιερώθηκε από τον Codd το 1970 και αποτέλεσε ένα από τα πιο απλά και ευέλικτα μοντέλα αυτού του είδους. Είναι ένα γράφημα που απεικονίζει τις συσχετίσεις ανάμεσα στις οντότητες της βάσης. Σε αυτό το μοντέλο, τα δεδομένα μιας εφαρμογής αναπαρίστανται ως ένα σύνολο από σχέσεις (relations) οι οποίες μπορεί να είναι πίνακες ή αρχεία. Στις πιο πολλές περιπτώσεις υιοθετείται η χρήση πινάκων (tables) που περιέχουν ένα πλήθος γραμμών (rows) και στηλών (columns). Η κάθε μια από αυτές τις γραμμές – οι οποίες στην ορολογία του μοντέλου ονομάζονται και πλειάδες (tuples) – περιέχει ένα σύνολο απλών πεδίων (attributes) τα οποία συσχετίζονται μεταξύ τους.

Κάθε πεδίο ή στήλη ενός πίνακα, δέχεται τιμές οι οποίες ανήκουν σε ένα συγκεκριμένο και καθορισμένο εκ των προτέρων σύνολο τιμών (domain). Το είδος των τιμών αυτού του συνόλου καθορίζεται από τον τύπο δεδομένων του πεδίου

του πίνακα, ο οποίος με τη σειρά του ορίζεται κατά το στάδιο της λογικής σχεδίασης της εφαρμογής.
Οι τιμές που καταχωρούνται σε μια πλειάδα, θα πρέπει να είναι ατομικές (atomic) να μην υποδιαιρούνται δηλαδή σε μικρότερες μονάδες πληροφορίας. Σύνθετες τιμές δεδομένων, δεν μπορούν να χρησιμοποιηθούν αλλά θα πρέπει να γραφούν με τρόπο που να μην παραβιάζει την παραπάνω αρχή.
Παρακάτω παρατίθονται το σχεσιακό μοντέλο της εφαρμογής.

Admin **(adminId, name, email, password, privileges, lastSign)**

Στον πίνακα Admin αποθηκεύονται όλοι οι διαχειριστές της εφαρμογής οι οποίοι θα μπορούν να έχουν πρόσβαση σε όλους τους χρήστες, τα αρχεία, τις φωτογραφίες και τις προκηρύξεις.

BlackList (bLid, bLfromMid, bLToMid, bLinsertionDate)

Στον πίνακα BlackList γίνονται οι καταχωρήσεις κάθε μέλους το οποίο δεν επιθυμεί να έχει επικοινωνία από κάποιο άλλο μέλος της εφαρμογής.

Changes (chId, chMid, chProfile, chOldName, chNewName, chOldEmail, chNewEmail, chDate)

Στον πίνακα Changes καταχωρούνται όλες οι αλλαγές που έχουν κάνει οι χρήστες σχετικά με το profile, το όνομα και το email τους. Οι αλλαγές αυτές είναι ορατές από τα μέλη φίλοι τους.

Contacts (cId, cFromMid, cToMid, requestDate, confirmDate, confirmStatus)

Ο πίνακας Contacts περιέχει κάθε αίτηση φιλίας που γίνεται μεταξύ των χρηστών όσο αυτή δεν απορριφθεί ή διαγραφή από κάποιο από τα δύο μέλη.

deletedMembers (deletionId, deletedMid, deletionDate)

Ο πίνακας deletedMembers περιέχει όλα τα διαγραμμένα μέλη τα οποία έχουν κάνει απενεργοποίηση λογαριασμού.

Files (fId, mId, fName, fSubject, fBranch, fUploadDate)

Στον πίνακα αυτόν αποθηκεύονται τα στοιχεία κάθε αρχείου που έχει τη δυνατότητα να φορτώσει κάθε μέλος της εφαρμογής.

Members (mId, name, email, password, branch, regDate, lastSign, region, sex, birthDate, occupation, activities, personality, website, phone, pic)

Ο πίνακας Members είναι το κυριότερο σημείο της βάσης δεδομένων αφού σε αυτόν αποθηκεύονται τα μέλη – χρήστες.

Messages (messageId, fromMid, toMid, subject, content, sentDate, status, inboxDelete, sentDelete)

Στον πίνακα Messages καταχωρούνται τα μηνύματα που ανταλλάζουν οι χρήστες μεταξύ τους μέχρι αυτά να διαγραφούν και από τα δύο μέλη.

News (nId, subject, content, nFiles, nUploadDate)

Στον πίνακα News γίνονται οι καταχωρήσεις των προκηρύξεων του ΑΣΕΠ και τα στοιχεία τους.

Online (onId, onMid, ip)

Ο πίνακας Online περιέχει τα μέλη εκείνα που είναι συνδεδεμένα κάθε στιγμή στην εφαρμογή.

Settings (sId, sMid, viewableProfile, viewableContact, viewablePhoto, acceptMessages, viewableAtSearch)

Ο πίνακας Settings περιέχει τις ρυθμίσεις εκείνες τις οποίες κάνει κάθε χρήστης και είναι απαραίτητες για τη διασφάλιση των προσωπικών στοιχείων του.

unconfirmMembers (uMid, hashedMid, name, email, password, branch)

Ο πίνακας unconfirmMembers περιέχει όλα εκείνα τα μέλη τα οποία δεν έχουν κάνει ενεργοποίηση λογαριασμού μέσα από το email τους.

8.5 Οντότητες και Συσχετίσεις

Όταν μοντελοποιείται η πραγματικότητα, απεικονίζεται σε ένα σύστημα που έχει να κάνει με οντότητες και συσχετίσεις.

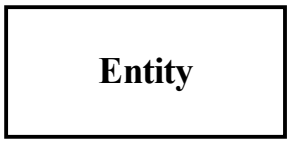
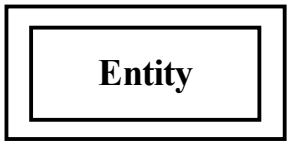
Οντότητα είναι η κάθε περίπτωση μονάδας του πραγματικού συστήματος η οποία παρακολουθείται πληροφοριακά και η οποία έχει αυτόνομη ύπαρξη μέσα στον κόσμο της υπό ανάπτυξη εφαρμογής. Η κάθε οντότητα έχει μια σειρά από χαρακτηριστικά, τα οποία είναι γνωστά και προς επεξεργασία. Οι τιμές των χαρακτηριστικών καθορίζουν (στον κόσμο της εφαρμογής) την κάθε μια στιγμή οντότητας και την διακρίνουν από τις υπόλοιπες στιγμές της ίδιας οντότητας. Ένα από τα χαρακτηρίστηκα ή κάποιος συνδυασμός τους διακρίνεται από την ικανότητα που έχει να προσδιορίζει μονοσήμαντα, με την τιμή του, την κάθε στιγμή της οντότητας. Αυτό είναι το κύριο κλειδί της οντότητας. Υπάρχουν περιπτώσεις όπου περισσότερα του ενός χαρακτηριστικά ή συνδυασμοί χαρακτηριστικών μπορούν να παίξουν τον ρόλο του κυρίου κλειδιού για μια οντότητα. Σε αυτήν την περίπτωση υπάρχουν πολλά υποψήφια κλειδιά. Αν συμβεί αυτό τότε επιλέγεται ένα από αυτά ως κύριο κλειδί και τα υπόλοιπα ονομάζονται εναλλακτικά κλειδιά.

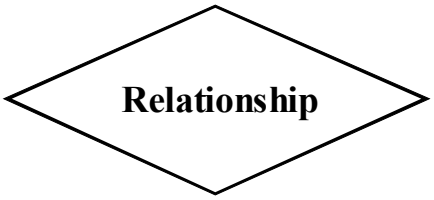
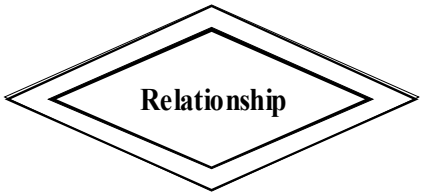
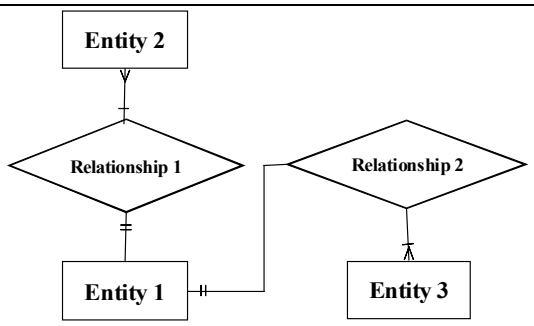
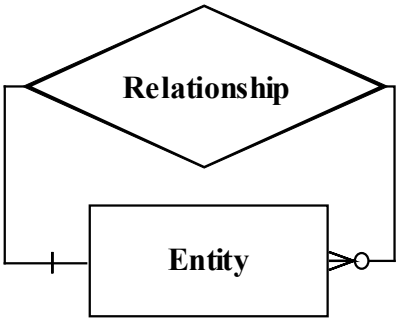
Στο χώρο του μοντέλου της εφαρμογής, οι οντότητες δεν είναι απομονωμένες αλλά αλληλεπιδρούν με το περιβάλλον τους. Οι αλληλεπιδράσεις απεικονίζονται υπό τη μορφή συσχετίσεων που ορίζουν να υπάρχουν ανάμεσα σε δύο η περισσότερες οντότητες.

8.6 Διάγραμμα Οντοτήτων – Συσχετίσεων (ER)

Οντότητες και συσχετίσεις παρουσιάζονται διαγραμματικά, για ένα συγκεκριμένο μοντέλο εφαρμογής, με το λεγόμενο διάγραμμα ER (Entity – Relationship Diagram). Αυτή η διαγραμματική τεχνική, επιτρέπει τη σχεδίαση των οντοτήτων μιας βάσης και των συσχετίσεων που υφίστανται ανάμεσα τους χρησιμοποιώντας ειδικά σύμβολά τα οποία παρουσιάζονται στον Πίνακα ΙΙΙ που ακολουθεί:

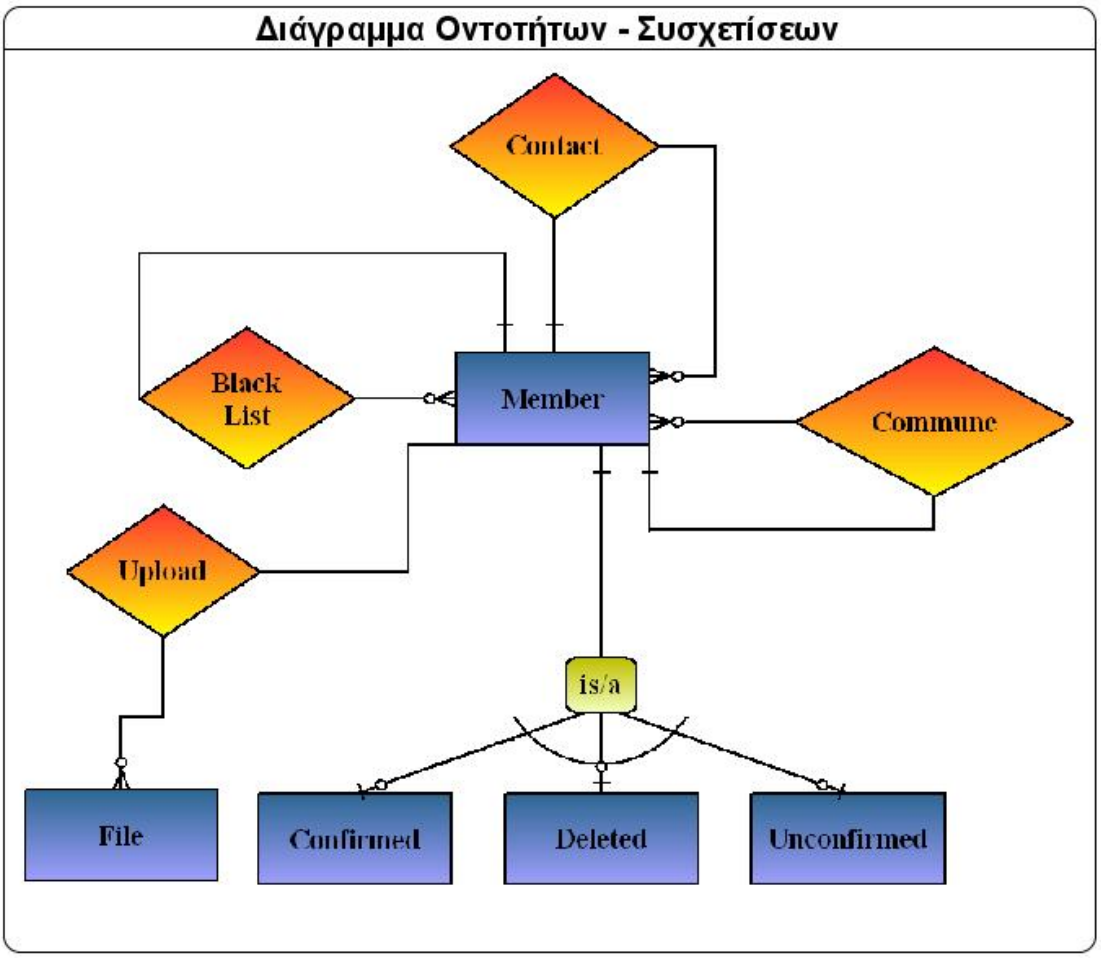
Πίνακας ΙΙΙ: Σύμβολα Διαγράμματος ER

	**Entity**
	**Weak Entity**
	**Key Attribute**
	**Multivalued Attribute**

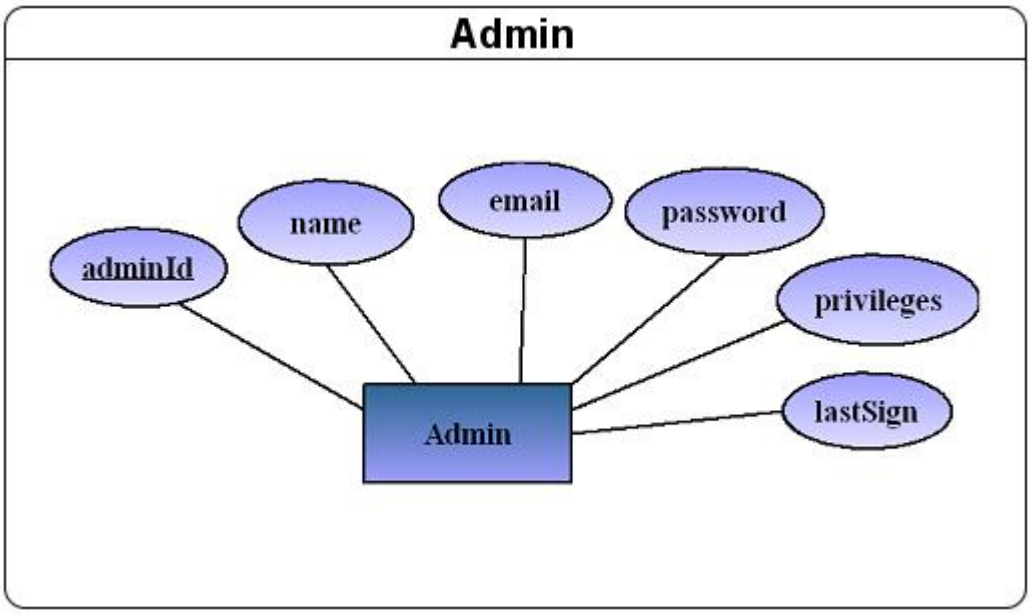
	**Derived Attribute**
	**Relationship**
	**Relationship with Weak Entity**
	**Cardinality**
	**Recursive Relationship**

Στο Σχήμα VII φαίνεται το βασικό διάγραμμα ER του συστήματος της εφαρμογής που αναπτύσσεται. Για την καλύτερη κατανόηση του παρακάτω διαγράμματος δεν

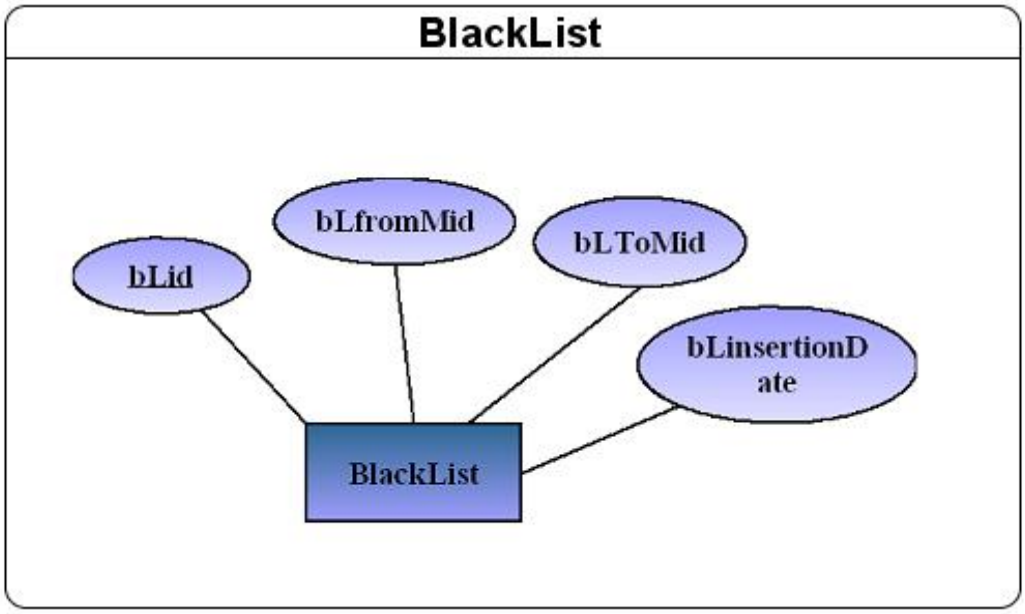
υπάρχουν τα πεδία και τα πεδία κλειδιά των οντοτήτων. Αυτά φαίνονται στα επόμενα σχήματα.



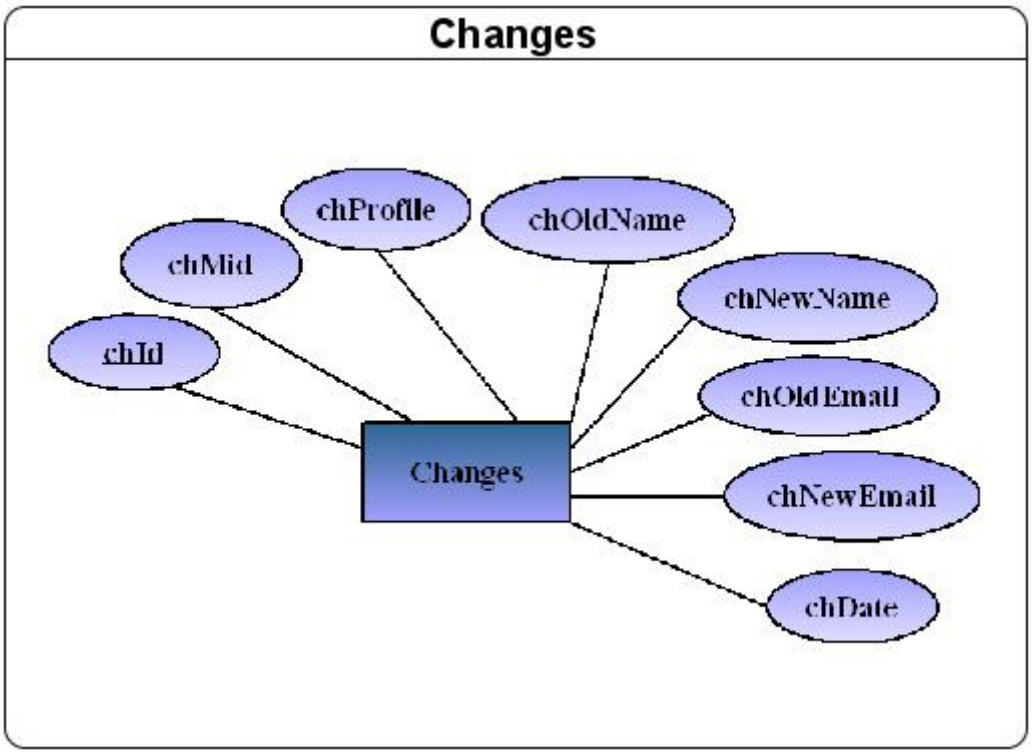
Σχήμα VII: Διάγραμμα Οντοτήτων – Συσχετίσεων



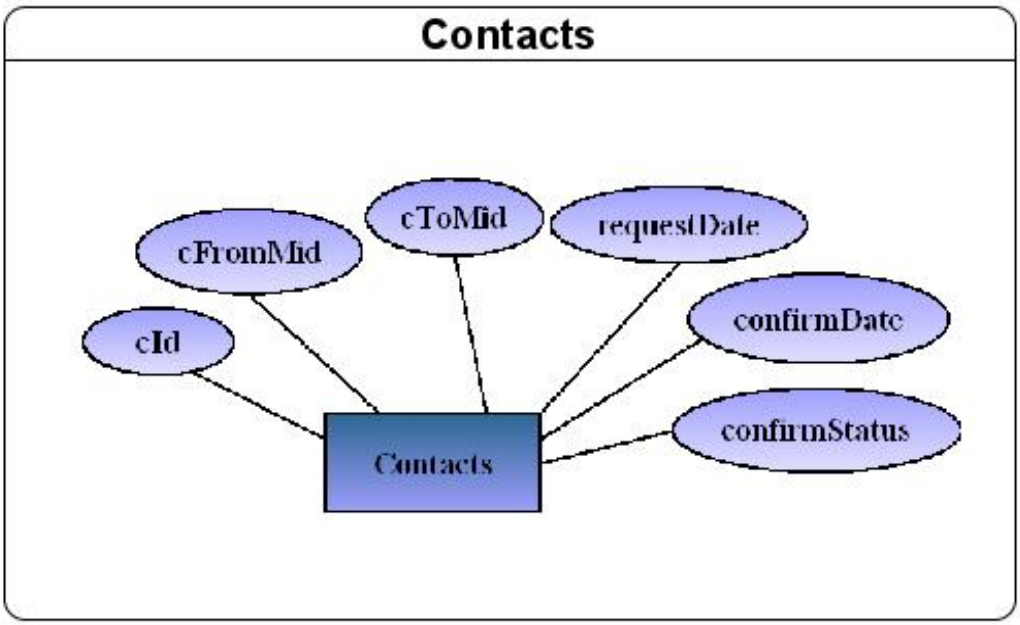
Σχήμα VIII: Πίνακας Admin



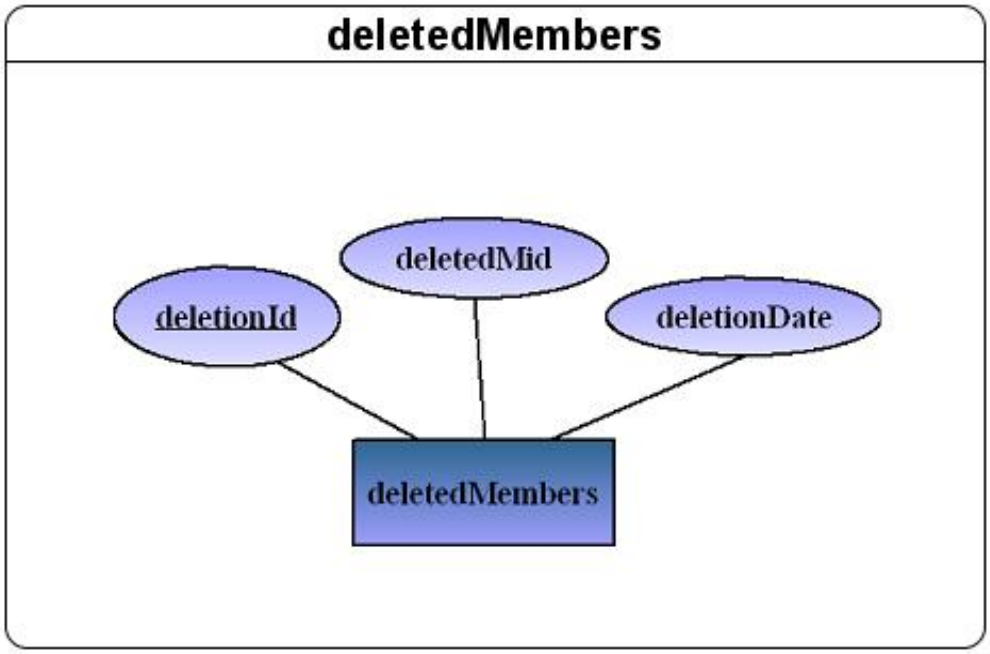
Σχήμα IX: Πίνακας BlackList



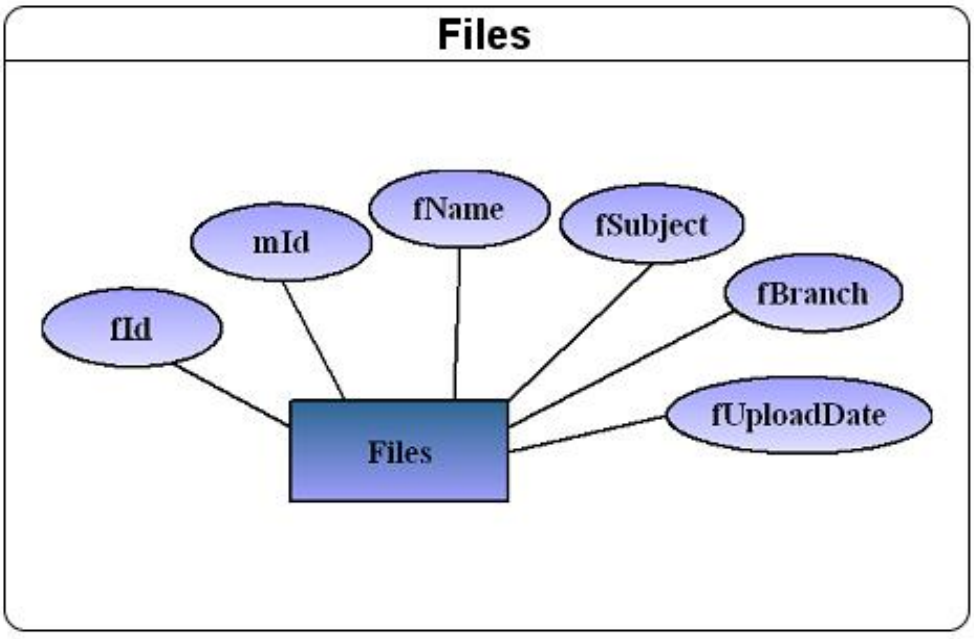
Σχήμα X: Πίνακας Changes



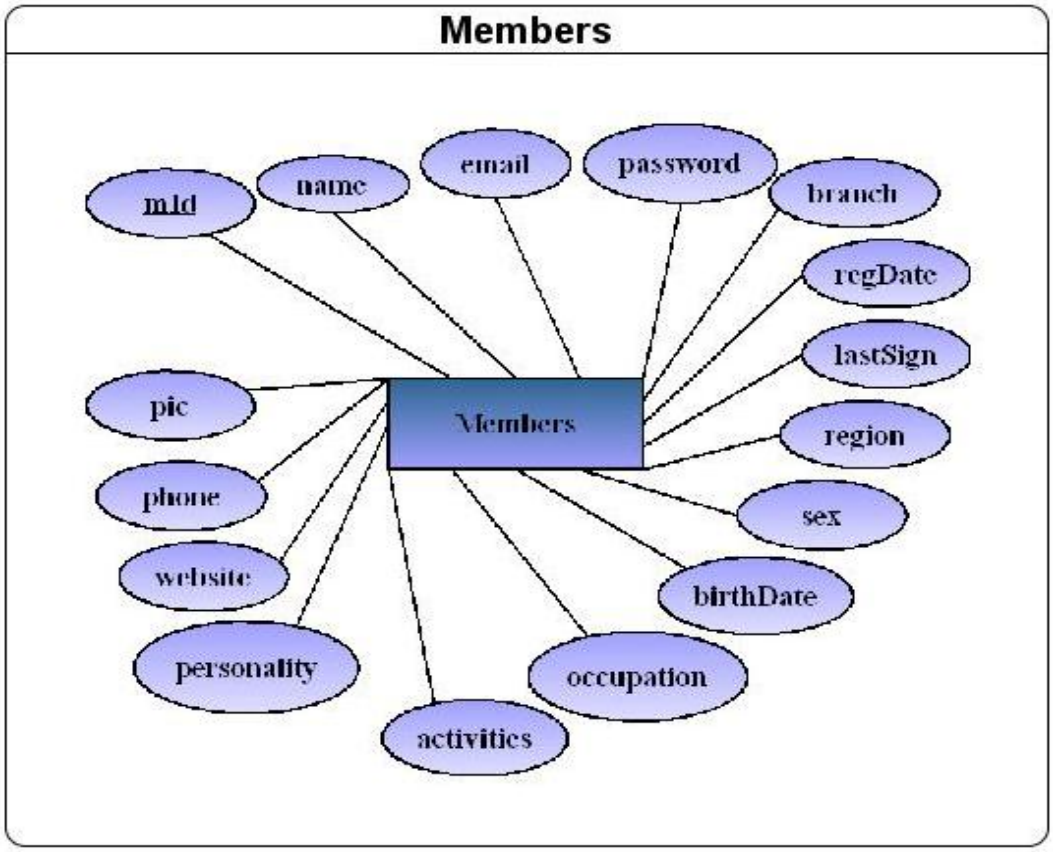
Σχήμα XI: Πίνακας Contacts



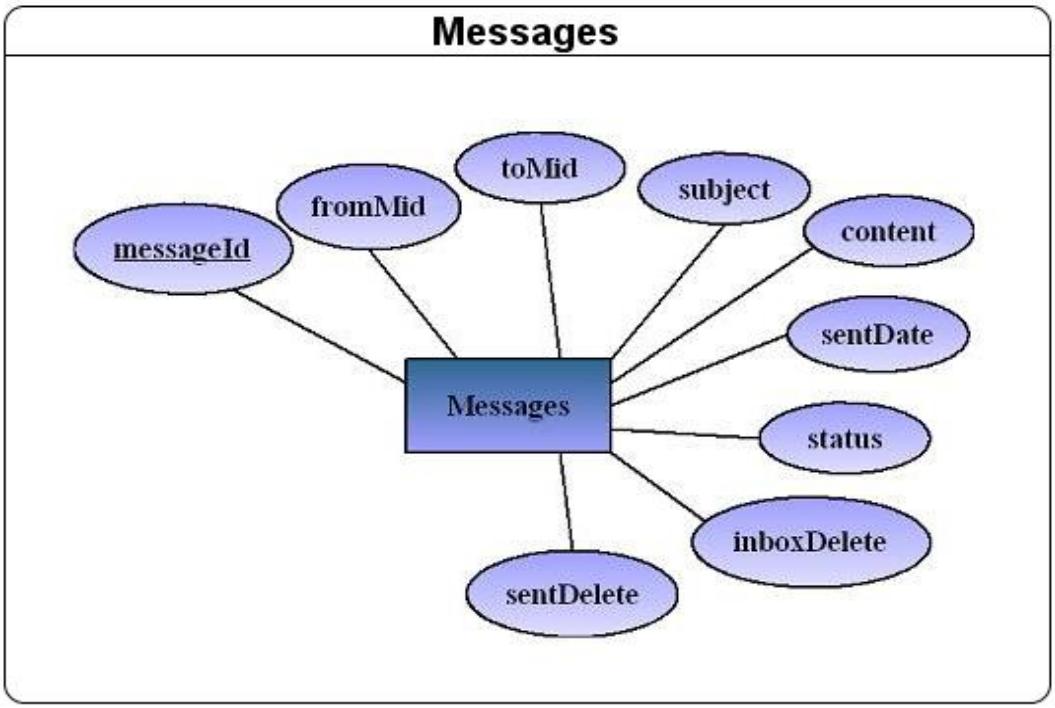
Σχήμα XII: Πίνακας deletedMembers



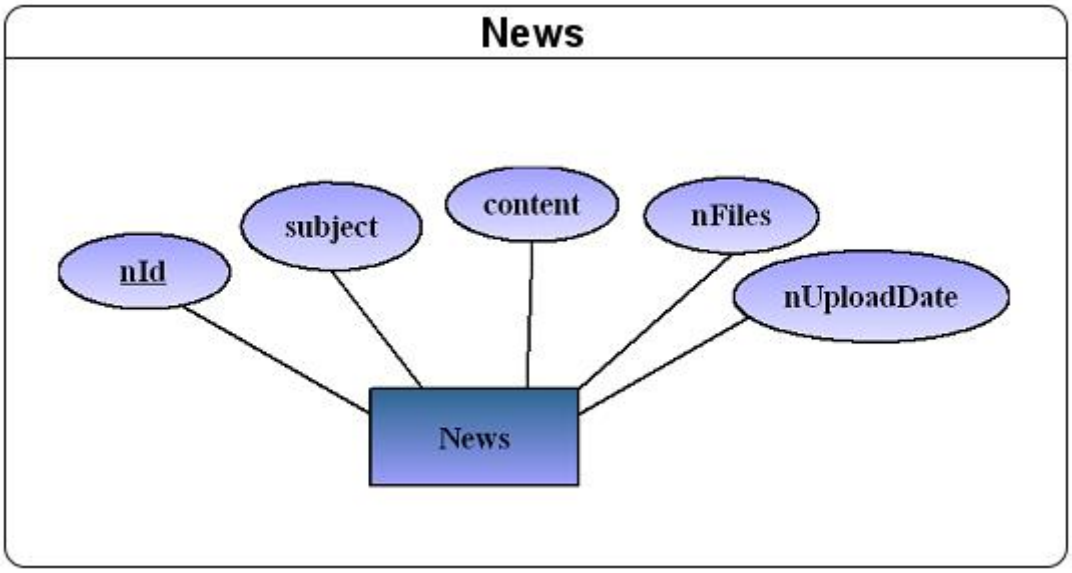
Σχήμα XIII: Πίνακας Files



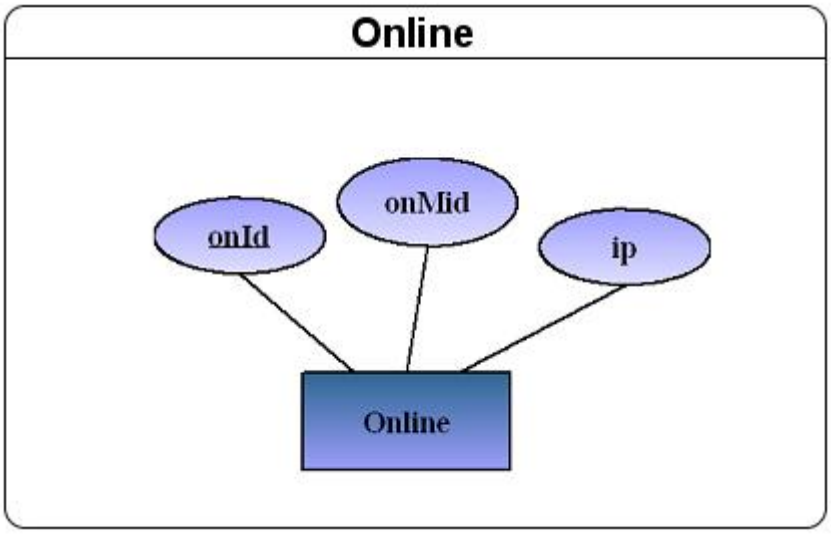
Σχήμα XIV: Πίνακας Members



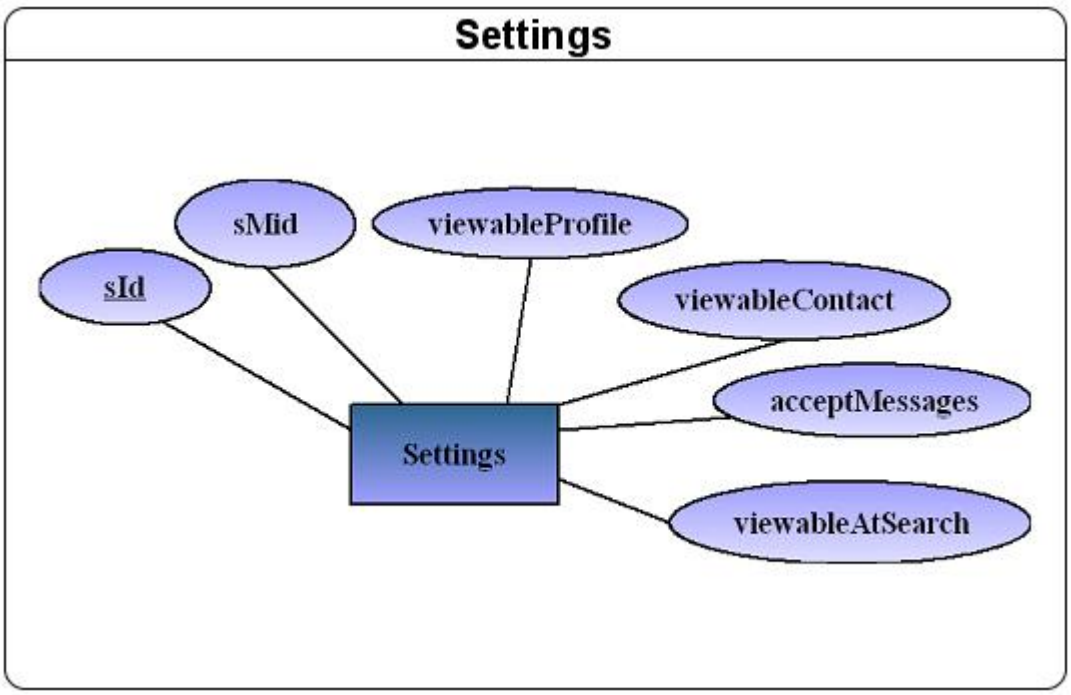
Σχήμα XV: Πίνακας Messages



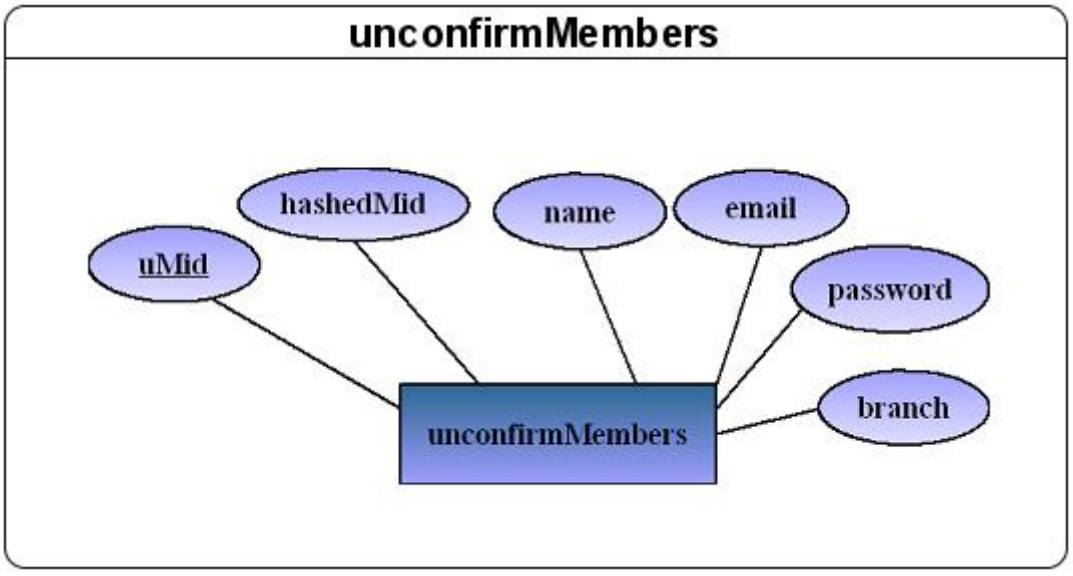
Σχήμα XVI: Πίνακας News



Σχήμα XVII: Πίνακας Online



Σχήμα XVIII: Πίνακας Settings



Σχήμα XIX: Πίνακας unconfirmMembers

8.7 Τεκμηρίωση πινάκων

Στην συγκεκριμένη εφαρμογή χρησιμοποιούνται οι δώδεκα παραπάνω πίνακες οι οποίοι φαίνονται και στο Σχήμα XX η οποία δείχνει την βάση δεδομένων της εφαρμογής μέσα από το εργαλείο phpmyamdin:

Σχήμα XX: Οι πίνακες του συστήματος

8.7.1 Περιγραφή πεδίων

Ακολουθεί η τεκμηρίωση των πινάκων με όλα τα πεδία, την περιγραφή τους.

Πίνακας: Admin	**Περιγραφή**
adminId	Ο κωδικός του κάθε διαχειριστή.
name	Το όνομα του κάθε διαχειριστή
email	Το email του κάθε διαχειριστή.
password	Ο κωδικός του κάθε διαχειριστή.
privileges	Τα προνόμια του κάθε διαχειριστή.
lastSign	Η τελευταία σύνδεση του κάθε διαχειριστή.

Πίνακας: BlackList	Περιγραφή
bLid	Ο κωδικός κάθε εγγραφής.
bLfromMid	Ο κωδικός του μέλους που αιτήθηκε την φραγή μέλους.
bLToMid	Το κωδικός του προς φραγή μέλους.
bLinsertionDate	Η ημερομηνία αίτησης φραγής.

Πίνακας: Changes	**Περιγραφή**
chId	Ο κωδικός της κάθε αλλαγής.
chMid	Ο κωδικός του μέλους που έκανε την αλλαγή.
chProfile	Το πεδίο σημαία σε περίπτωση κάποιας αλλαγής στο profile.
chOldName	Το πεδίο σημαία σε περίπτωση αλλαγής ονόματος.
chNewName	Το πεδίο που περιέχει το καινούριο όνομα του μέλους.
chOldEmail	Το πεδίο σημαία σε περίπτωση αλλαγής email.
chNewEmail	Το πεδίο που περιέχει το καινούριο email του μέλους.
chDate	Η ημερομηνία αλλαγής.

Πίνακας: Contacts	**Περιγραφή**
cId	Ο κωδικός κάθε αίτησης φιλίας.
cFromMid	Ο κωδικός του μέλους που αιτήθηκε την φιλία.
cToMid	Ο κωδικός του μέλους στο οποίο έγινε η αίτηση φιλίας.
requestDate	Η ημερομηνία αίτησης της φιλίας.
confirmDate	Η ημερομηνία αποδοχής της φιλίας.
confirmStatus	Το πεδίο υπόδειξης αποδοχής ή αναμονής της φιλίας.

Πίνακας: deletedMembers	Περιγραφή
deletionId	Ο κωδικός της εγγραφής.
deletedMid	Ο κωδικός του μέλους που διαγράφηκε.
deletionDate	Η ημερομηνία διαγραφής του μέλους.

Πίνακας: Files	**Περιγραφή**
fId	Ο κωδικός του κάθε αρχείου.
mId	Ο κωδικός του μέλους που ανάρτησε το αρχείο.
fName	Το όνομα του αρχείου.
fSubject	Το θέμα του αρχείου.
fBranch	Ο κλάδος στον οποίο ανήκει το αρχείο.
fUploadDate	Η ημερομηνία ανάρτησης του αρχείου.

Πίνακας: Members	Περιγραφή
mId	Ο κωδικός του μέλους.
name	Το όνομα του μέλους.
email	Το email του μέλους.
password	Ο κωδικός του μέλους.
branch	Ο κλάδος του μέλους.
regDate	Η ημερομηνία εγγραφής του μέλους.
lastSign	Η ημερομηνία τελευταίας σύνδεσης του μέλους.
region	Η περιοχή διαμονής του μέλους.
sex	Το φύλο του μέλους.
birthDate	Η ημερομηνία γέννησης του μέλους.
occupation	Το επάγγελμα του μέλους.
activities	Οι δραστηριότητες του μέλους.
personality	Η προσωπικότητα του μέλους.
website	Η ιστοσελίδα του μέλους.
phone	Το τηλέφωνο του μέλους.
pic	Το όνομα του αρχείου της φωτογραφίας του μέλους.

Πίνακας: Messages	Περιγραφή
messageId	Ο κωδικός κάθε μηνύματος.
fromMid	Ο κωδικός του μέλους που έστειλε το μήνυμα.
toMid	Ο κωδικός του μέλους που έλαβε το μήνυμα.
subject	Το θέμα του μηνύματος.
content	Το περιεχόμενο του μηνύματος.
sentDate	Η ημερομηνία αποστολής του μηνύματος.
status	Το πεδίο που υποδηλώνει αν το μήνυμα είναι αναγνωσμένο.
inboxDelete	Το πεδίο που υποδηλώνει αν το παραλήπτης έχει διαγράψει το μήνυμα.
sentDelete	Το πεδίο που υποδηλώνει αν ο αποστολέας έχει διαγράψει το μήνυμα.

Πίνακας: News	**Περιγραφή**
nId	Ο κωδικός κάθε προκήρυξης.
subject	Το θέμα κάθε προκήρυξης.
content	Το περιεχόμενο κάθε προκήρυξης.
nFiles	Το όνομα του συνοδευτικού αρχείου της προκήρυξης.
nUploadDate	Η ημερομηνία ανάρτησης της προκήρυξης.

Πίνακας: Online	**Περιγραφή**
onId	Ο κωδικός κάθε εγγραφής.
onMid	Ο κωδικός του μέλους που είναι συνδεδεμένο στην εφαρμογή.
ip	Η ip διεύθυνση του μέλους που είναι συνδεδεμένο στην εφαρμογή.

Πίνακας: Settings	**Περιγραφή**
sId	Ο κωδικός κάθε εγγραφής.
sMid	Ο κωδικός του μέλους.
viewableProfile	Το πεδίο που υποδηλώνει αν το profile του μέλους θα είναι ορατό από όλους ή μόνο από τους φίλους του.
viewableContact	Το πεδίο που υποδηλώνει αν τα στοιχεία επικοινωνίας του μέλους θα είναι ορατά από όλους ή μόνο από τους φίλους του.
viewablePhoto	Το πεδίο που υποδηλώνει αν η φωτογραφία του μέλους θα είναι ορατή από όλους ή μόνο από τους φίλους του.
acceptMessages	Το πεδίο που υποδηλώνει αν μέλος θα δέχεται μηνύματα από όλους ή μόνο από τους φίλους του.
viewableAtSearch	Το πεδίο που υποδηλώνει αν το μέλος θα εμφανίζεται στις αναζητήσεις άλλων μελών ή όχι.

Πίνακας: unconfirmMembers	**Περιγραφή**
uMid	Ο κωδικός του μέλους το οποίο δεν έχει κάνει επικύρωση εγγραφής.
hashedMid	Ο κωδικοποιημένος κωδικός του μέλους.
name	Το όνομα του μέλους.
email	Το email του μέλους.
password	Ο κωδικός του μέλους.
branch	Ο κλάδος του μέλους.

Πίνακας: Admin

Πεδίο	Τύπος
adminId	mediumint(9)
name	varchar(30)
email	varchar(32)
password	varchar(40)
privileges	set('y','n')
lastSign	timestamp

Πίνακας: BlackList

Πεδίο	Τύπος
bLid	mediumint(9)
bLfromMid	mediumint(9)
bLToMid	mediumint(9)
bLinsertionDate	timestamp

Πίνακας: Changes

Πεδίο	Τύπος
chId	mediumint(9) auto_increment
chMid	mediumint(9)
chProfile	set('y','n')
chOldName	varchar(40)
chNewName	varchar(32)
chOldEmail	varchar(40
chNewEmail	varchar(32)
chDate	timestamp

Πίνακας: Contacts

Πεδίο	Τύπος
cId	mediumint(9)
cFromMid	mediumint(9)
cToMid	mediumint(9)
requestDate	timestamp
confirmDate	timestamp
confirmStatus	set('y','n','u')

Πίνακας: deletedMembers

Πεδίο	Τύπος
deletionId	mediumint(9) auto_increment
deletedMid	mediumint(9)
deletionDate	timestamp

Πίνακας: Files

Πεδίο	Τύπος
fId	mediumint(9) auto_increment
mId	mediumint(9)
FName	varchar(100)
FSubject	varchar(150)
FBranch	smallint(6)
fUploadDate	timestamp

Πίνακας: Members

Πεδίο	Τύπος
mId	`mediumint(9) auto_increment`
name	`varchar(40)`
email	`varchar(32)`
password	`varchar(40)`
branch	`varchar(20)`
regDate	`date`
lastSign	`timestamp`
region	`varchar(30)`
sex	`set('u','m','f')`
birthDate	`date`
occupation	`varchar(300)`
activities	`varchar(700)`
personality	`varchar(700)`
website	`varchar(50)`
phone	`varchar(30)`
pic	`varchar(100)`

Πίνακας: Messages

Πεδίο	Τύπος
messageId	`int(11) auto_increment`
fromMid	`mediumint(9)`
toMid	`mediumint(9)`
subject	`varchar(100)`
content	`text`
sentDate	`timestamp`
status	`set('u','r')`
inboxDelete	`set('n','y')`
sentDelete	`set('y','n')`

Πίνακας: News

Πεδίο	Τύπος
nId	mediumint(9)
subject	varchar(200)
content	mediumtext
nFiles	varchar(100)
nUploadDate	timestamp

Πίνακας: Online

Πεδίο	Τύπος
onId	mediumint(9)
onMid	mediumint(9)
ip	varchar(14)

Πίνακας: Settings

Πεδίο	Τύπος
sId	mediumint(9) auto_increment
sMid	mediumint(9)
viewableProfile	set('a','f')
viewableContact	set('a','f')
viewablePhoto	set('a','f')
acceptMessages	set('a','f')
viewableAtSearch	set('y','n')

Πίνακας: unconfirmMembers

Πεδίο	Τύπος
uMid	int(11) auto_increment
hashedMid	varchar(40)
name	varchar(40)
email	varchar(32)
password	varchar(40)
branch	int(20)

8.7.2 MySQL Κώδικας

Η δημιουργία των πινάκων έγινε με τον SQL κώδικα που δίνεται παρακάτω. Πρέπει να σημειωθεί ότι η βάση για να δέχεται ελληνικούς χαρακτήρες πρέπει να δηλωθεί με τύπο utf8_unicode_ci.

```
CREATE TABLE `admin` (
  `adminId` mediumint(9) NOT NULL,
  `name` varchar(30) character set latin1 NOT NULL,
  `email` varchar(32) character set latin1 NOT NULL,
  `password` varchar(40) character set latin1 NOT NULL,
  `privileges` set('y','n') character set latin1 NOT NULL
default 'n',
  `lastSign` timestamp NOT NULL default CURRENT_TIMESTAMP,
  PRIMARY KEY  (`adminId`),
  UNIQUE KEY `email` (`email`)
)
```

```
CREATE TABLE `blacklist` (
  `bLid` mediumint(9) NOT NULL auto_increment,
  `bLFromMid` mediumint(9) NOT NULL,
  `bLToMid` mediumint(9) NOT NULL,
  `bLInsertionDate` timestamp NOT NULL default
CURRENT_TIMESTAMP,
  PRIMARY KEY  (`bLid`)
)
```

```
CREATE TABLE `changes` (
  `chId` int(11) NOT NULL auto_increment,
  `chMId` mediumint(9) NOT NULL,
  `chProfile` set('y','n') character set latin1 NOT NULL,
  `chOldName` varchar(40) character set latin1 NOT NULL,
  `chOldEmail` varchar(32) character set latin1 NOT NULL
default '',
  `chNewName` varchar(40) character set latin1 NOT NULL,
  `chNewEmail` varchar(32) character set latin1 NOT NULL,
  `chDate` timestamp NOT NULL default CURRENT_TIMESTAMP,
  PRIMARY KEY  (`chId`)
)
```

```
CREATE TABLE `contacts` (
  `cId` mediumint(9) NOT NULL auto_increment,
  `cFromMid` mediumint(9) NOT NULL,
  `cToMid` mediumint(9) NOT NULL,
  `requestDate` timestamp NOT NULL default CURRENT_TIMESTAMP,
  `confirmDate` timestamp NULL default '0000-00-00 00:00:00',
  `confirmStatus` set('y','n','u') character set latin1 NOT
NULL default '',
  PRIMARY KEY  (`cId`)
)
```

```
CREATE TABLE `deletedmembers` (
  `deletionId` mediumint(9) NOT NULL auto_increment,
  `deletedMid` mediumint(9) NOT NULL,
  `deletionDate` timestamp NOT NULL default CURRENT_TIMESTAMP,
  PRIMARY KEY  (`deletionId`)
)
```

```
CREATE TABLE `files` (
  `fId` mediumint(9) NOT NULL auto_increment,
  `mId` mediumint(9) NOT NULL,
  `fName` varchar(100) collate utf8_unicode_ci NOT NULL,
  `fSubject` varchar(150) collate utf8_unicode_ci NOT NULL,
  `fBranch` smallint(6) NOT NULL,
  `fUploadDate` timestamp NOT NULL default CURRENT_TIMESTAMP,
  PRIMARY KEY  (`fId`)
)
```

```
CREATE TABLE `members` (
  `mId` mediumint(9) NOT NULL auto_increment,
  `name` varchar(40) character set latin1 NOT NULL,
  `email` varchar(32) character set latin1 NOT NULL,
  `password` varchar(40) character set latin1 NOT NULL,
  `branch` varchar(20) character set latin1 NOT NULL,
  `regDate` date NOT NULL,
  `lastSign` timestamp NOT NULL default CURRENT_TIMESTAMP on
update CURRENT_TIMESTAMP,
  `region` varchar(30) character set latin1 default NULL,
  `sex` set('u','m','f') character set latin1 NOT NULL default
'u',
  `birthDate` date default NULL,
  `occupation` varchar(300) character set latin1 default NULL,
  `activities` varchar(700) character set latin1 default NULL,
  `personality` varchar(700) character set latin1 default
NULL,
  `website` varchar(50) character set latin1 default NULL,
  `phone` varchar(30) character set latin1 default NULL,
  `pic` varchar(100) character set latin1 default NULL,
  PRIMARY KEY  (`mId`)
```

```
CREATE TABLE `messages` (
  `messageId` int(11) NOT NULL auto_increment,
  `fromMid` mediumint(9) NOT NULL,
  `toMid` mediumint(9) NOT NULL,
  `subject` varchar(100) collate utf8_unicode_ci NOT NULL,
  `content` text collate utf8_unicode_ci NOT NULL,
  `sentDate` timestamp NOT NULL default CURRENT_TIMESTAMP,
  `status` set('u','r') character set latin1 NOT NULL,
  `inboxDelete` set('n','y') character set latin1 NOT NULL,
  `sentDelete` set('y','n') character set latin1 NOT NULL,
  PRIMARY KEY  (`messageId`)
)
```

```
CREATE TABLE `news` (
  `nId` mediumint(9) NOT NULL auto_increment,
  `subject` varchar(200) collate utf8_unicode_ci NOT NULL,
  `content` mediumtext collate utf8_unicode_ci NOT NULL,
  `nFiles` varchar(100) collate utf8_unicode_ci default
NULL,
  `nUploadDate` timestamp NOT NULL default
CURRENT_TIMESTAMP,
```

```
CREATE TABLE `online` (
  `onId` mediumint(9) NOT NULL auto_increment,
  `onMid` mediumint(9) NOT NULL,
  `ip` varchar(14) character set latin1 NOT NULL,
  PRIMARY KEY  (`onId`)
)
```

```
CREATE TABLE `settings` (
  `sId` mediumint(9) NOT NULL auto_increment,
  `sMid` mediumint(9) NOT NULL,
  `viewableProfile` set('a','f') character set latin1 NOT
NULL default 'a',
  `viewableContact` set('a','f') character set latin1 NOT
NULL default 'a',
  `viewablePhoto` set('a','f') character set latin1 NOT NULL
default 'a',
  `acceptMessages` set('a','f') character set latin1 NOT NULL
default 'a',
  `viewableAtSearch` set('y','n') character set latin1 NOT
NULL default 'y',
  PRIMARY KEY  (`sId`)
)
```

```
CREATE TABLE `unconfirmmembers` (
  `uMid` int(11) NOT NULL auto_increment,
  `hashedMid` varchar(40) character set latin1 default NULL,
  `name` varchar(40) character set latin1 NOT NULL,
  `email` varchar(32) character set latin1 NOT NULL,
  `password` varchar(40) character set latin1 NOT NULL,
  `branch` int(20) NOT NULL,
  PRIMARY KEY  (`uMid`)
)
```

8.8 Επίλογος

Έχοντας ολοκληρώσει τον σχεδιασμό και την υλοποίηση της βάσης δεδομένων, το μόνο που απομένει είναι ο προγραμματισμός σε PHP σύμφωνα με της απαιτήσεις και τις λειτουργίες που έχουν αναφερθεί κατά την φάση της ανάλυσης.

ΚΕΦΑΛΑΙΟ 9ο

ΥΛΟΠΟΙΗΣΗ ΣΕ PHP

9.1 Εισαγωγή

Στο κεφάλαιο αυτό παρουσιάζεται η υλοποίηση της εφαρμογής με τον απαραίτητο κώδικα και σχολιασμό του. Η δομή που θα ακολουθηθεί είναι κώδικας – εξήγηση ώστε η εφαρμογή να γίνει περισσότερο κατανοητή.

9.2 Υπερκαθολικές μεταβλητές $_GET και $_POST

Επειδή η HTML είναι stateless δηλαδή χωρίς κατάσταση, πρέπει να βρεθεί κάποιος τρόπος ώστε να ελέγχονται οι διάφορες καταστάσεις και μεταβλητές των εφαρμογών. Αυτό γίνεται με τις υπερκαθολικές μεταβλητές $_GET και $_POST.
Η php παίρνει από τη διεύθυνση URL ό,τι υπάρχει μετά το ? και το τοποθετεί σε έναν πίνακα που είναι μία υπερκαθολική μεταβλητή, η $_GET. Με την εντολή print_r($_GET) μπορούμε δούμε τα περιεχόμενα της $_GET. Είναι πολύ χρήσιμο κατά το πέρασμα URL με PHP να χρησιμοποιούμε μία μεταβλητή για να πάρουμε την τοποθεσία URL όπου βρισκόμαστε. Για παράδειγμα ο κώδικας:

```
$id = $_GET['id'];
```

θα πάρει τη μεταβλητή $id από την URL και θα την αποθηκεύσει σε μία τοπική μεταβλητή $id.
Με αυτόν τον τρόπο μπορούν να αποσταλούν τιμές ανάμεσα στις ιστοσελίδες. Δηλαδή για να χρησιμοποιηθεί ένας σύνδεσμος που να οδηγεί από την τρέχουσα σελίδα στη σελίδα αποθήκευσης των δεδομένων, γράφεται:

```
<a href = "submintpage.php?address=kostas&id=1">Submit
Page</a>
```

Ενώ στον κώδικα της Submit σελίδας γράφουμε:

```
$address = $_GET['address'];
$id = $_GET['id'];
```

Έτσι, για παράδειγμα στην σελίδα Submit μπορούν να τυπωθούν όλες οι τιμές που έστειλε η προηγούμενη σελίδα, με την εντολή print_r($_GET).

Πρέπει να σημειωθεί ότι το σύμβολο με το οποίο συνενώνονται οι τιμές στη γραμμή διευθύνσεων είναι το &. Σε περίπτωση όμως που το & πρέπει να χρησιμοποιηθεί σαν δεδομένο πρέπει να χειριστεί ιδιαίτερα. Για παράδειγμα, στην περίπτωση του παρακάτω (προσοχή στο kostas&&id=1):

```
<a href = "submintpage.php?address=kostas&&id=1">Submit
Page</a>
```

Σε αυτήν την περίπτωση το δεύτερο "&" θα χαθεί χωρίς να υπάρχει δυνατότητα απόκτησής του από την γραμμή διευθύνσεων.

Το σωστό εδώ είναι να γραφτεί στην αρχική σελίδα ο παρακάτω κώδικας ο οποίος κάνει χρήση της μεθόδου urlencode():

```
<a href = "submintpage.php?name=<?php echo
urlencode("kostas&"); ?>&id=1">Submit Page</a>
```

Κατά το πέρασμα στην URL, ο,τιδήποτε δεν είναι αλφαριθμητικό, παύλα (-) ή κάτω παύλα (_), περνάει ως ένα % ακολουθούμενο από δύο ψηφία. Για παράδειγμα, το κενό είναι το %20 ενώ το & το %26. Με τη μέθοδο urlencode() το κενό από %20 γίνεται +. Καλό είναι να εφαρμόζεται η urlencode() κάθε φορά που χρησιμοποιείται η $_GET επειδή δεν είναι πάντα σίγουρο αν αυτό το κάνει ο server αυτόματα με τις URL ή τους συνδέσμους. Επίσης, μπορεί να χρησιμοποιηθεί και η συνάρτηση rawurlencode η οποία έχει ως αποτέλεσμα να γίνονται οι ίδιες μετατροπές με αυτές που θα έκανε ο server αυτόματα (για παράδειγμα μετατρέπει το κενό σε %20).

Η rawurlencode() χρησιμοποιείται σε περίπτωση που δημιουργίας μία διεύθυνση URL δυναμικά και χρησιμοποιείται για ο,τιδήποτε υπάρχει πριν το πρώτο "?". Για ο,τιδήποτε υπάρχει μετά το πρώτο "?" χρησιμοποιείται η απλή urlencode(). Έτσι, για την περίπτωση της Submit Page η rawurlencode() χρησιμοποιείται για το submintpage.php ενώ για όλο το υπόλοιπο εφαρμόζεται η urlencode().

Σε περίπτωση δημιουργίας και αποστολής HTML κώδικα στην έξοδο χρησιμοποιείται η htmlspecialchars(). Επίσης, αυτός είναι τρόπος διαφυγής για τους ειδικούς χαρακτήρες από δεδομένα που παίρνουμε από βάση δεδομένων εκτός από τις URL. Έτσι, το < θα γραφτεί ως lt ενώ το > ως gt εξασφαλίζοντας ότι αυτά τα σύμβολα δε θα έχουν την HTML σημασία τους. Το ίδιο γίνεται και με την

συνάρτηση htmlentities η οποία αφαιρεί οποιαδήποτε HTML σημασία από το περιεχόμενο της παραμέτρου της.
Για παράδειγμα, ο κώδικας:

```
$i = "<p><strong>Hello</strong></p>";
echo $i;
```

Εκτυπώνει στην οθόνη μία παράγραφο με έντονη γραμματοσειρά.
Αντίθετα όμως, η πρόταση:

```
echo htmlentities($i);
```

Εκτυπώνει στην οθόνη την συμβολοσειρά:
" Hello "
Οι παραπάνω περιπτώσεις διαφυγής συνοψίζονται ως εξής:

a. Καθαρίζεται ο σύνδεσμος ώστε να μπορεί να χρησιμοποιηθεί με τις rawurlencode() και urlencode().
b. Διαφεύγονται οι ειδικοί χαρακτήρες με τη χρήση μίας εκ των htmlspecialchars() και htmlentities() ώστε να εμφανίζονται με ασφάλεια οι σύνδεσμοι και εν συνεχεία οι σελίδες.

Η χρήση της $_POST είναι πολύ πιο απλή και ασφαλής από την $_GET. Κάθε φορά που γίνεται ένα Submit σε μία φόρμα η οποία έχει την ιδιότητα action ορισμένω ως post τότε χρησιμοποιείται η υπερκαθολική μεταβλητή $_POST. Για παράδειγμα η πρόταση:

```
$name = $_POST['username'];
```

Θα αποθηκεύσει στη μεταβλητή $name την τιμή που εισήχθηκε το πεδίο της φόρμας με id ορισμένο με "username".
Με την _POST δεν χρειάζεται να γίνει urlencode ή urldecode επειδή αυτά έχουν οριστεί να γίνονται αυτόματα.

9.3 Η υπερκαθολική μεταβλητή $_SESSION

Μία session είναι ένα αρχείο που αποθηκεύεται στον web server. Μπορούμε να βρούμε ποιο αρχείο ανήκει σε κάθε χρήστη κάνοντας χρήση των cookies. Θέτουμε ένα ειδικό session cookie στον browser και μετά κοιτάμε σε αυτό το

cookie για να βρούμε το μέρος όπου βρίσκεται το αρχείο στον server μας κι έπειτα να ψάξουμε στο αρχείο αυτό για τις πληροφορίες που θέλουμε. Οι πληροφορίες αυτές δεν μπορούν να αλλαχτούν αλλά το μόνο που μπορούμε να κάνουμε είναι να αλλάξουμε το id number το οποίο είναι πολύ μεγάλο για λόγους ασφαλείας. Έτσι για να χρησιμοποιήσουμε μία session πρώτα δημιουργούμε ένα τέτοιο αρχείο και να θέσουμε το cookie στον υπολογιστή του χρήστη ή αν αυτό έχει ήδη γίνει να βρούμε αυτό το cookie και να βρούμε το αντίστοιχο αρχείο σε αυτόν τον υπολογιστή. Έτσι πρέπει να αρχίσουμε με μία session_start(). Αυτή πρέπει να γίνει στην αρχή του αρχείου πριν εκτυπώσουμε οποιαδήποτε HTML ή ακόμα και κενό και πριν στείλουμε οποιαδήποτε πληροφορία στον browser του χρήστη. Έτσι η PHP βρίσκει από τον browser το id number του session cookie, βρίσκει το αρχείο και το ανοίγει ώστε να χρησιμοποιηθεί. Αν δεν βρει το id number ή αν δε μπορεί να βρει το session file τότε αρχίζει μία καινούρια session και διαμορφώνει το κατάλληλο cookie. Μετά το session_start() μπορούμε να γράψουμε για παράδειγμα:

```
$_SESSION['email'] = "kostas";
$name = $_SESSION['email'];
```

Τα βασικά βήματα για την χρήση τους τα οποία είναι:

- Εκκίνηση μίας συνόδου λειτουργίας.
- Εγγραφή των μεταβλητών της συνόδου λειτουργίας.
- Χρήση μεταβλητών της συνόδου λειτουργίας.
- Ακύρωση της εγγραφής των μεταβλητών και καταστροφή της συνόδου.

Σημειώνεται ότι αυτά τα βήματα δεν συμβαίνουν όλα στο ίδιο script και μερικά από αυτά μπορούν να συμβούν σε πολλαπλά script.
Παρακάτω περιγράφεται πιο αναλυτικά το κάθε ένα από αυτά τα βήματα.

9.3.1 Εκκίνηση μίας συνόδου λειτουργίας

Πριν χρησιμοποιηθεί λειτουργικά μία σύνοδος, πρέπει πρώτα να ξεκινήσει. Υπάρχουν δύο τρόποι για να γίνει αυτό.
Ο πρώτος και απλούστερος, είναι να ξεκινήσει ένα script με μία κλήση στην συνάρτηση session_start(). Αυτή η συνάρτηση ελέγχει αν υπάρχει ήδη μία τρέχουσα σύνοδος. Αν όχι, θα δημιουργήσει μία, παρέχοντας πρόσβαση στον

υπερκαθολικό πίνακα $_SESSION. Αν υπάρχει ήδη μία σύνοδος, η session_start() φορτώνει τις εγγεγραμμένες μεταβλητές συνόδου ώστε να μπορούν να χρησιμοποιηθούν. Σημειώνεται ότι η session_start() πρέπει να είναι η πρώτη γραμμή που εκτελείται όταν χρειάζεται να χρησιμοποιηθεί μία σύνοδος λειτουργίας.

Ο δεύτερος τρόπος για να ξεκινήσει μία σύνοδος είναι να οριστεί η PHP έτσι ώστε να ξεκινά μία σύνοδο αυτόματα όταν κάποιος έρχεται στην τοποθεσία της εφαρμογής. Αυτό μπορεί να γίνει με την επιλογή session.auto_start στο αρχείο php.ini. Αυτή η μέθοδος όμως έχει το μεγάλο μειονέκτημα ότι με την επιλογή auto_start ενεργοποιημένη, δεν μπορούν να χρησιμοποιηθούν αντικείμενα ως μεταβλητές συνόδου.

9.3.2 Εγγραφή των μεταβλητών της συνόδου λειτουργίας

Οι μεταβλητές συνόδου αποθηκεύονται στην υπερκαθολικό πίνακα $_SESSION αλλά και στον $HTTP_SESSION_VARS. Σημειώνεται όμως ότι ο $HTTP_SESSION_VARS είναι καλό να μην χρησιμοποιείται επειδή αυτή η αντιμετώπιση των συνόδων είναι αρκετά παλιά και ξεπερασμένη.

Για να δημιουργηθεί μία μεταβλητή συνόδου απλώς ορίζεται ένα στοιχείο αυτού του πίνακα, ως εξής:

```
$_SESSION['myvar'] = 5;
```

Επίσης, σε παλαιότερες εκδόσεις της PHP, για να παρακολουθηθεί μία μεταβλητή από ένα script σε ένα άλλο, θα έπρεπε πρώτα να εγγραφεί με μία κλήση στην session_register(). Αυτή η μέθοδος όμως δεν είναι ασφαλής και καλό είναι να μην χρησιμοποιείται τώρα.

Η μεταβλητή συνόδου που μόλις δημιουργήθηκε θα παρακολουθηθεί έως ότου τελειώσει η σύνοδος ή μέχρι να ακυρωθεί.

9.3.3 Χρήση μεταβλητών συνόδου λειτουργίας

Για να χρησιμοποιηθεί μία μεταβλητή συνόδου, πρέπει πρώτα να ξεκινήσει η σύνοδος χρησιμοποιώντας την session_start(). Έπειτα, η πρόσβαση σε αυτήν γίνεται μέσα από τον υπερκαθολικό πίνακα $_SESSION, για παράδειγμα ως $_SESSION[‘myvar’].

9.3.4 Ακύρωση εγγραφής μεταβλητών και καταστροφή συνόδου

Η ακύρωση μίας μεταβλητής συνόδου γίνεται κατευθείαν ακυρώνοντας το κατάλληλο στοιχείο στον πίνακα $_SESSION. Για παράδειγμα:

```
unset($_SESSION['myvar']);
```

Πιο παλιά χρησιμοποιούνται οι συναρτήσεις session_unregister() και session_unset() αλλά τώρα αυτές δεν συστήνονται λόγω της εμφάνισης της $_SESSION.

Τέλος, όταν ακυρωθούν όλες οι μεταβλητές πρέπει να γίνει κλήση στην session_destroy() ώστε να καθαρίσει ο κωδικός της συνόδου.

Στην εφαρμογή χρησιμοποιείται ο παρακάτω κώδικας για τον έλεγχο αν κάποιος χρήστης είναι συνδεδεμένος ή όχι:

```
session_start();
function logged_in() {
     return isset($_SESSION['email']);
}
function confirm_logged_in() {
     if (!logged_in()) {
          redirect_to("index.php");
     }
}
function redirect_to($location = NULL) {
     if ($location != NULL) {
          header("Location: {$location}");
          exit;
     }
}
```

Το πρώτο πράγμα που γίνεται όταν ο χρήστης περιηγείται στις σελίδες είναι να γίνει κλήση της συνάρτησης confirm_logged_in(), η οποία ελέγχει αν το email του χρήστη βρίσκεται στον πίνακα $_SESSION. Το email του κάθε μέλους είναι μοναδικό και αποθηκεύεται στον πίνακα αυτόν κατά την είσοδό του στην εφαρμογή. Έτσι, αν ο χρήστης είναι συνδεδεμένος τότε συνεχίζει την περιήγησή

του στην εφαρμογή, διαφορετικά καλείται η συνάρτηση redirect_to() όπου τον μεταφέρει στην αρχική σελίδα. Εδώ σημειώνεται ότι η εντολή header() στέλνει HTTP header ώστε να μεταφερθεί η εκτέλεση σε άλλη ιστοσελίδα. Πολύ σημαντικό είναι ότι πρέπει να γίνει κλήση της header() πριν σταλεί στην έξοδο οποιοσδήποτε HTML κώδικας (ετικέτες, κείμενο ή ακόμα και κενά) αλλιώς θα προκληθεί λάθος.

9.4 SQL injection και προστασία βάσης δεδομένων

Ο όρος SQL injection περιγράφει την τεχνική εισαγωγής κώδικα η οποία εκμεταλλεύεται πιθανά κενά ασφαλείας στην βάση δεδομένων μίας εφαρμογής. Το κενό στην ασφάλεια έγκειται στο μη αποτελεσματικό φιλτράρισμα δεδομένων από τον χρήστη με συνέπεια να υπάρχουν μέσα σε αυτά ένθετα SQL ερωτήματα ή σε περίπτωση που τα δεδομένα που εισάγει ο χρήστης δεν είναι ισχυρώς τυποποιημένα (strongly typed) υπάρξει απρόσμενη εκτέλεση κώδικα. Παρακάτω εξετάζονται τα δύο αυτά ενδεχόμενα.

9.4.1 Αναποτελεσματικό φιλτράρισμα δεδομένων

Η μορφή αυτής της SQL injection συμβαίνει όταν ο χρήστης εισάγει δεδομένα τα οποία δεν ελέγχονται για χαρακτήρες διαφυγής και έτσι περνιούνται όπως είναι μέσα σε μία SQL έκφραση. Αυτό έχει ως αποτέλεσμα πιθανή εισαγωγή κώδικα από τους χρήστες της εφαρμογής. Ο παρακάτω κώδικας δείχνει μία τέτοια περίπτωση:

```
statement = "SELECT * FROM users WHERE name = '" +
userName + "';"
```

Ο κώδικας αυτός προορίζεται για να επιλέγει συγκεκριμένων ονομάτων χρηστών (userName) από τον πίνακα των χρηστών. Όμως, αν η μεταβλητή userName πάρει κάποιες συγκεκριμένες τιμές όπως είναι η παρακάτω τότε θα υπάρχει απρόσμενη εκτέλεση κώδικα:

```
A' or 't'='t
```

Έτσι τώρα η έκφραση statement γίνεται:

```
SELECT * FROM users WHERE name = 'a' OR 't'='t';
```

Αν ο κώδικας αυτός χρησιμοποιείται σε μία συνάρτηση πιστοποίησης χρήστη τότε θα είχε ως αποτέλεσμα την επιλογή όλων των χρηστών αφού η έκφραση 't'='t' είναι πάντα αληθής.

Αν και οι περισσότεροι SQL server επιτρέπουν την πολλαπλή εκτέλεση SQL εκφράσεων, αυτό δεν γίνεται από το php mysql_query για λόγους ασφαλείας. Αυτό αποτρέπει κακόβουλους χρήστες να εισάγουν εντελώς ξεχωριστά ερωτήματα προς την βάση αλλά παρόλα αυτά δεν αποτρέπει την τροποποίηση των ερωτημάτων. Η ακόλουθη τιμή της μεταβλητής userName θα έχει ως αποτέλεσμα την διαγραφή του πίνακα users καθώς και την επιλογή όλων των δεδομένων από τον πίνακα data.

```
A'; DROP TABLE users; SELECT * FROM data WHERE name LIKE '%
```

Έτσι τώρα η έκφραση statement γίνεται:

```
SELECT * FROM users WHERE name = 'a';DROP TABLE users;
SELECT * FROM DATA WHERE name LIKE '%';
```

9.4.2 Λανθασμένος χειρισμών τύπων δεδομένων

Αυτή η μορφή της SQL injection συμβαίνει όταν ένα πεδίο εισαγωγής δεδομένων δεν ελέγχεται για περιορισμούς τύπων δεδομένων. Αυτό μπορεί να συμβεί όταν ένα αριθμητικό πεδίο πρόκειται να χρησιμοποιηθεί σε μία SQL έκφραση αλλά ο προγραμματιστής δεν ελέγχει τα δεδομένα που εισάγει ο χρήστης ώστε να διαπιστώσει αν αυτά είναι όντως αριθμητικά. Για παράδειγμα:

```
statement = "SELECT * FROM data WHERE id = " + a_variable +
";"
```

Είναι προφανές ότι από την δήλωση αυτή ότι ο προγραμματιστής περιμένει να εισαχθούν αριθμητικά δεδομένα για το πεδίο “id”. Παρόλα αυτά, αν δοθούν δεδομένα τύπου string, τότε ο χρήστης μπορεί να χειριστεί την SQL έκφραση όπως αυτός επιλέξει. Για παράδειγμα, θέτοντας τη μεταβλητή a_variable σε:

```
1;DROP TABLE users
```

θα διαγραφεί ο πίνακας users από την βάση δεδομένων, αφού η SQL θα εκτιμήσει την έκφραση ως:

```
SELECT * FROM DATA WHERE id=1;DROP TABLE users;
```

9.5 Σύνδεση με την βάση δεδομένων

Πριν αρχίσει κάποια εργασία με τη βάση δεδομένων, θα πρέπει πρώτα να υπάρξει σύνδεση με τον διακομιστή. Για το λόγο αυτό η PHP παρέχει τη ρουτίνα mysql_connect(). Η mysql_connect() δεν απαιτεί κανένα όρισμα αλλά δέχεται τρία strings : το hostname, ένα username και ένα password. Εάν δεν δωθεί κανένα από αυτά τα ορίσματα, η ρουτίνα υποθέτει ότι το host είναι localhost και ότι τα username και password δεν έχουν οριστεί στο mysqluser πίνακα, εκτός αν έχουν οριστεί στο αρχείο php.ini. Η mysql_connect() επιστρέφει έναν αναγνωριστή συνδέσμου (link identifier) αν η σύνδεση είναι επιτυχής. Η επιστρεφόμενη τιμή μπορεί να αποθηκευτεί σε μία μεταβλητή ώστε να μπορεί να προσπελαστεί ξανά χωρίς να χρειαστεί να καλεστεί πάλι η μέθοδος mysql_connect().

Όταν γίνει η σύνδεση με τον διακομιστή βάσης δεδομένων θα πρέπει να επιλεγεί η βάση δεδομένων με την οποία θα λειτουργεί η εφαρμογή. Η δυνατότητα αυτή δίνεται από την συνάρτηση mysql_select_db(). Η συγκεκριμένη μέθοδος απαιτεί το όνομα μιας βάσης δεδομένων και προαιρετικά έναν αναγνωριστή συνδέσμου. Η mysql_select_db() επιστρέφει true αν η βάση δεδομένων υπάρχει και μπορεί να γίνει χειρισμός αυτής.

Το επόμενο κομμάτι κώδικα χρησιμοποιεί τη mysql_connect() και mysql_select_db() για να συνδεθεί με τον διακομιστή βάσης δεδομένων MySQL και να χρησιμοποιήσει τη βάση δεδομένων που καθορίζει η μεταβλητή MY_DATABASE:

```
require_once("constants.php");
function db_connection() {
     $connection = mysqli_connect(SERVER, USER,
MY_PASSWORD, MY_DATABASE);
     if (!$connection) {
          die("Database connection failed: " .
mysqli_error());}
     return $connection;}
```

```
function db_selection() {
     $db_select = mysqli_select_db(MY_DATABASE,
db_connection());
     if (!$db_select) {
          die("Database selection failed: " .
mysqli_error());
     }
     return $db_select;
}
```

Σημειώνεται ότι για λόγους ασφαλείας καλό είναι το όνομα της βάσης, ο κωδικός αλλά και το όνομα χρήστη να αποθηκεύονται σε ξεχωριστό αρχείο στον server και να γίνεται εισαγωγή αυτού με την συνάρτηση include ή require.

9.6 Διαφυγή χαρακτήρων

Για την αποφυγή αυτών των προβλημάτων συνίσταται η χρήση μεθόδων διαφυγής χαρακτήρων. Στην PHP χρησιμοποιείται η μέθοδος mysql_real_escape_string(). Παρακάτω παρουσιάζεται η μέθοδος αυτή όπου εφαρμόζεται σε όλη την παρούσα εφαρμογή:

```
function mysql_prep($value, $connection) {
     $magic_quotes_active = get_magic_quotes_gpc();
     $new_enough_php =
function_exists("mysql_real_escape_string");
     if ($new_enough_php) {
          if ($magic_quotes_active) {
               $value = stripslashes($value);}
     $value = mysqli_real_escape_string($connection,
$value);
     } else {
          if (!$magic_quotes_active) {
               $value = addslashes($value);
          }    }
     return $value;}
```

Παρατηρείται η συνάρτηση get_magic_quotes_gpc(). Αυτό που κάνει η εντολή αυτή είναι να ελέγχει αν η οδηγία magic_quotes_gpc είναι ενεργοποιημένη ή όχι. Σε περίπτωση που η οδηγία αυτή είναι ενεργοποιημένη τότε η διαφυγή χαρακτήρων γίνεται αυτόματα προσθέτοντας όπου αυτό απαιτείται τον χαρακτήρα “ ` ” Όμως αυτή η οδηγία πρόκειται να αφαιρεθεί από τις μελλοντικές εκδόσεις της PHP και ήδη η χρήση της δεν συνίσταται. Έτσι, αφού ελεγχθεί αν η οδηγία αυτή είναι ενεργοποιημένη, τότε ελέγχεται αν υπάρχει η συνάρτηση mysql_real_escape_string αφού αυτή προστέθηκε μετά την έκδοση 4. Αυτό γίνεται επειδή μερικές φορές μπορεί ο προγραμματιστής να μην γνωρίζει την έκδοση της PHP που χρησιμοποιεί ο server. Έπειτα γίνεται έλεγχος και γίνεται η διαφυγή χαρακτήρων όπου αυτό είναι απαραίτητο και επιστρέφεται η κατάλληλη τιμή με ασφαλή πλέον δεδομένα.

9.7 Έλεγχος εγκυρότητας τιμών

Για την εγγραφή κάποιου χρήστη στην εφαρμογή πρέπει να εισάγει το κατάλληλο email και κωδικό του. Ο παρακάτω κώδικας ελέγχει την εγκυρότητα του email που εισάγει ο χρήστης. Σε περίπτωση που δεν περάσει όλους τους ελέγχους τότε επιστρέφεται αντίστοιχο μήνυμα λάθους:

```
if (!strstr($email, '@') || !strstr($email, '.') ||
empty($email))
```

Η μέθοδος strstr() ελέγχει αν μία συγκεκριμένη συμβολοσειρά περιέχεται μέσα σε μία άλλη. Εδώ ελέγχεται αν στο email που έχει εισάγει ο χρήστης υπάρχουν οι χαρακτήρες “@” και “.” Οι οποίοι είναι απαραίτητοι χαρακτήρες για κάθε διεύθυνση email. Επίσης, με την βοήθεια της συνάρτησης empty() ελέγχεται αν έχει δοθεί τιμή στο συγκεκριμένο πεδίο της φόρμας. Εδώ σημειώνεται ότι η συνάρτηση empty() επιστρέφει true και σε περίπτωση όπου το πεδίο έχει πάρει την τιμή 0 ή και null.

Στην συνέχεια ελέγχεται αν το μήκος του δοθέντος email είναι μέσα στο όριο ενός συγκεκριμένου πλήθους χαρακτήρων. Σε αυτήν την περίπτωση δεν πρέπει να ξεπερνάει τους 32 αλλά ούτε και να είναι μικρότερο από 6. Αυτό επιτυγχάνεται με την συνάρτηση strlen().

```
if (strlen($email) > 32 || strlen($email) < 6)
```

Κατόπιν, γίνεται έλεγχος ώστε να διαπιστωθεί αν οι χαρακτήρες που εισάγονται είναι αποδεκτοί. Για παράδειγμα, χαρακτήρες όπως οι ":", "/" "|", "*" δεν επιτρέπονται. Αυτός ο έλεγχος γίνεται με την χρήστη της συνάρτησης ereg() όπου δέχεται δύο ορίσματα. Το ένα είναι μία κανονική έκφραση και το άλλο μία συμβολοσειρά μέσα στην οποία πρέπει να υφίστανται οι κανόνες της κανονικής έκφρασης που δόθηκε πριν. Αυτό στην εφαρμογή γίνεται ως εξής:

```
if (ereg('[^A-Za-z_0-9.@]+',$email))
```

Στην συνέχεια ελέγχεται αν το email που δόθηκε χρησιμοποιείται ήδη. Αυτό επιτυγχάνεται ελέγχοντας την βάση δεδομένων εκτελώντας ένα ερώτημα:

```
$query = "SELECT email FROM Members WHERE email LIKE
'$email'";
$result = mysqli_query($connection, $query);
$num_results = mysqli_num_rows($result);
```

Εδώ, πρώτα γράφεται το ερώτημα $query, έπειτα εκτελείται με την συνάρτηση mysqli_query() κι έπειτα καλείται η mysqli_num_rows() για να βρεθεί ο αριθμός των αποτελεσμάτων που βρέθηκαν. Σε περίπτωση που αυτά δεν είναι 0 τότε το email που δόθηκε χρησιμοποιείται ήδη.

Τέλος, για λόγους ασφαλείας απαγορεύεται να γίνεται χρήση του ίδιου κωδικού με το τμήμα του email που υπάρχει πριν από τον χαρακτήρα "@".

```
$token = strtok($email, '@');
```

Η συνάρτηση strtok() διαχωρίζει την συμβολοσειρά της πρώτης παραμέτρου στα σημεία όπου υπάρχει ο χαρακτήρας της δεύτερης παραμέτρου. Έπειτα, ελέγχεται η μεταβλητή $token με τον κωδικό που έχει δοθεί.

9.8 Κρυπτογράφηση κωδικών

Οι κωδικοί πρόσβασης δεν πρέπει να αποθηκεύονται σε μορφή απλού κειμένου. Ένας αλγόριθμος hash μίας κατεύθυνσης, μπορεί να παρέχει λίγο περισσότερη ασφάλεια.

Η PHP παρέχει διάφορες συναρτήσεις hash μίας κατεύθυνσης. Ο καλύτερος και ισχυρότερος από αυτούς παρέχεται με την συνάρτηση sha1() η οποία έχει ενσωματωθεί στην PHP από την έκδοση 5 (παλιότερα χρησιμοποιούνταν η συνάρτηση md5()). Το πρωτότυπο της sha1() είναι:

```
string sha1 ( string str [, bool raw_output] )
```

Αν δοθεί η συμβολοσειρά str, η συνάρτηση θα επιστρέψει μία ψευδοτυχαία συμβολοσειρά 40 χαρακτήρων. Αν οριστεί το raw_output να είναι true, τότε θα επιστραφεί μία συμβολοσειρά 20 χαρακτήρων από δυαδικά δεδομένα. Αυτό είναι πολύ σημαντικό επειδή πρέπει το πεδίο του πίνακα της βάσης δεδομένων στο οποίο θα αποθηκευτεί ο κωδικός πρέπει να οριστεί ανάλογα. Η επιστρεφόμενη συμβολοσειρά δεν μπορεί να αποκρυπτογραφηθεί και να μετατραπεί στην αρχική της μορφή. Η ιδιότητα που κάνει την sha1() χρήσιμη είναι ότι η έξοδός της είναι προσδιορίσιμη. Αν δοθεί η ίδια συμβολοσειρά, η sha1() θα επιστρέφει το ίδιο αποτέλεσμα κάθε φορά που θα τρέχει. Έτσι, αυτό που ελέγχεται είναι αν ο κωδικός πρόσβασης που πληκτρολογήθηκε είναι ο ίδιος με αυτόν που αρχικά έτρεξε η sha1(). Αυτό στην εφαρμογή επιτυγχάνεται ως εξής:

```
if (sha1($password) != $row['password'])
```

όπου $password ο κωδικός που πληκτρολογήθηκε από τον χρήστη ενώ όπου $row['password'] ο κωδικός που είναι αποθηκευμένος στο πεδίο των κωδικών της βάσης δεδομένων.

9.9 Επικύρωση εγγραφής χρήστη

Σε πολλές εφαρμογές, όταν κάποιος χρήστης εγγραφεί, αμέσως μετά του ζητά να επικυρώσει την εγγραφή του με ένα email ενεργοποίησης. Στην τρέχουσα εφαρμογή αυτό επιτυγχάνεται με τον ακόλουθο κώδικα:

```
$query = "INSERT INTO unconfirmMembers (hashedMid, name,
email, password, branch) VALUES ('".$hashedMid."',
'".$name."', '".$email."', '".$hashed_password."',
'".$branch."');";
$result = mysqli_query($connection, $query);
confirm_query($result);
$id = mysqli_insert_id($connection);
$hashedMid = sha1($id);
$query2 = "UPDATE unconfirmMembers SET hashedMid =
'$hashedMid' WHERE uMid = '$id';";
$result2 = mysqli_query($connection, $query2);
confirm_query($result2);
$email = $email;
$link =
"http://kostptyx.net76.net/index.php?action=newmember&mid=
$hashedMid";
$mesg = "Για να ολοκληρώσετε την εγγραφή σας στην
Διαδικτυακή Κοινότητα του ΑΣΕΠ, κάντε κλικ στον παρακάτω
σύνδεσμο: $link.";
$from = "From: asepkoinotita@yahoo.gr \r\n";
notify_password($email, $mesg, $from);
```

Αυτό που θα κάνει ο παραπάνω κώδικας είναι να εισάγει στον πίνακα unconfirmMembers τον χρήστη που μόλις εγγράφηκε, έπειτα να κρυπτογραφήσει για λόγους ασφαλείας το id που μόλις εισήχθηκε με τη χρήση της sha1(), να ανανεώσει τον πίνακα των επικυρωμένων μελών και τέλος να στείλε ένα email επικύρωσης στο email του χρήστη με τη μέθοδο notify_password().

```
function notify_password($email, $mesg, $from) {
     if (mail($email, "Information", $mesg, $from))
}
```

Η συνάρτηση mail() στέλνει ένα email και οι παράμετροι αντιπροσωπεύουν την διεύθυνση στην οποία θα σταλεί το μήνυμα, το θέμα του μηνύματος και τα περιεχόμενα του μηνύματος αντίστοιχα.
Ο λόγος που κρυπτογραφήθηκε το id του χρήστη είναι ώστε όταν ο χρήστης επικυρώσει την εγγραφή του να μην είναι σε θέση να γνωρίζει πως γίνεται αυτό αφού ο σύνδεσμος πάνω στον οποίο θα πρέπει να κάνει κλικ θα είναι κωδικοποιημένος. Επίσης, χρησιμοποιείται η συνάρτηση mysqli_insert_id() η οποία επιστρέφει το τελευταίο πεδίο id που δημιουργήθηκε αυτόματα από το προηγούμενο κάθε φορά ερώτημα.
Έτσι, όταν ο χρήστης κάνει κλικ στον σύνδεσμο επικύρωσης θα επιλεγεί το συγκεκριμένο id, ο χρήστης θα εγγραφεί στον πίνακα Members των κανονικών μελών, έπειτα θα γραφτεί στον πίνακα των συνδεδεμένων μελών, θα διαγραφεί από τον πίνακα των μη επικυρωμένων μελών και τέλος θα εγγραφούν οι προεπιλεγμένες προτιμήσεις του στον πίνακα Settings.
Σημειώνεται ότι ένα από τα πεδία του πίνακα Online είναι και η διεύθυνση ip του χρήστη. Αυτή βρίσκεται ως εξής:

```
$ip = $_SERVER['REMOTE_ADDR'];
```

9.10 Πρότυπο layout

Για την επίτευξη κοινού layout σε όλες τις σελίδες της εφαρμογής χρησιμοποιείται στην αρχή κάθε σελίδας η μέθοδος όπου έχει ως όρισμα το τίτλο της σελίδας, το αρχείο με τους CSS κανόνες καθώς και τα αρχεία JavaScript. Αυτή είναι η:

```
function header_login($title, $stylesheet, $javascript) {
…..
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
<title>'."$title".'</title>
<link href='."$stylesheet".' rel="stylesheet"
type="text/css" />
```

Με τη μέθοδο αυτή επιτυγχάνεται η απευθείας ανάθεση του CSS καθώς και ενός προκαθορισμένου στυλ σε όλες τις σελίδες πριν ακόμα αρχίσει να εκτελείται οποιοδήποτε κομμάτι κώδικα.

9.11 Επιλογή κλάδου

Η επιλογή και προβολή του κλάδου κάθε μέλους γίνεται με τη χρήση της HTML ετικέτας <select>. Οι πίνακες $allBranches και $allBranchesDetails περιέχουν τις τιμές και της αντίστοιχα ονόματα του κάθε κλάδου. Ελέγχεται κάθε ένα από τα στοιχεία $ allBranches και όποιο από αυτά ισούται με την παράμετρο $br που είναι ο κλάδος του χρήστη αποθηκεύεται η αντίστοιχη τιμή του στη μεταβλητή $i. Έπειτα επιστρέφεται η αντίστοιχη περιγραφή του κλάδου από τον πίνακα $allBranchesDetails που βρίσκεται στο στοιχείο i.

```
function show_branches($br) {
     require_once("all_branches.php");
     $allBranches = array(0, 1, …);
     $allBranchesDetails = array("Κλάδος...", "ΠΕ01
Θεολογίας", …);
     $i = 0;
     foreach ($allBranches as $current) {
          if ($br != $current) {
               $i++;
          } elseif ($br == $current) {
               break;
          }
     }
     return $allBranchesDetails[$i];
}
```

9.12 Διαμόρφωση ημερομηνίας

Τα πεδία τύπου Timestamp της βάσης δεδομένων έχουν τη μορφή YYYY-MM-DD hh:mm:ss. Για να την επιλογή μόνο του μέρους της ημερομηνίας και την ταυτόχρονη μορφοποίησή του σε DD-MM-YYYY, χρησιμοποιείται η παρακάτω μέθοδος:

```
function proper_date($d) {
     list($ymd, $hms) = split('[ ]', $d);
     list($year, $month, $day) = split('[-]', $ymd);
     $pd = $day."-".$month."-".$year;
     return $pd;
}
```

Αυτό που κάνει είναι να διαχωρίζει την συμβολοσειρά που δέχτηκε με delimiter τον χαρακτήρα του κενού, χρησιμοποιώντας τη μέθοδο split() κι έπειτα να την αποθηκεύει στις μεταβλητές $ymd και $hms. Τέλος, διαχωρίζεται εκ νέου η μεταβλητή $ymd αλλά αυτή τη φορά με διαχωριστικό την παύλα (-) και οι τιμές που επιστρέφονται αποθηκεύονται στις αντίστοιχες $year, $month, $day ώστε να μπουν συνενωμένες με την σωστή σειρά στην μεταβλητή $pd.

9.13 Ενημέρωση για ανάρτηση νέων αρχείων

Για την ενημέρωση του χρήστη σχετικά με την ανάρτηση νέων αρχείων από την τελευταία του σύνδεση γίνεται με το παρακάτω ερώτημα το οποίο μετράει όλα τα αρχεία των οποίων η τιμή του πεδίου fUploadDate είναι μεγαλύτερη από την τελευταία σύνδεση του χρήστη:

```
$query = "SELECT COUNT(*) AS num FROM Files WHERE
fUploadDate > '$lastSign';";
```

9.14 Εύρεση πλήθους φίλων

Η εύρεση του πλήθους των φίλων ενός χρήστη γίνεται ελέγχοντας τα πεδία cToMid και cFromMid εκ των οποίων ένα από τα δύο κάθε φορά πρέπει να είναι ίσο με το mId του χρήστη. Εκτός από αυτό το κριτήριο πρέπει το πεδίο confirmStatus να έχει την τιμή ‘y’, δηλαδή να έχει γίνει αποδοχή φιλίας.

```
$query = "SELECT cFromMid From Contacts WHERE cToMid =
'$mId' AND confirmStatus LIKE 'y'
     UNION
 SELECT cToMid FROM Contacts WHERE cFromMid = '$mId' AND
confirmStatus LIKE 'y';";
```

Υπενθυμίζεται ότι ο τελεστής UNION σημαίνει ένωση και αυτό που κάνει είναι να ενώνει τα αποτελέσματα που παράχθηκαν από τα δύο ή και περισσότερα SELECT.

9.15 Εμφάνιση ή όχι του profile χρήστη

Ένα από τα σημαντικότερα στοιχεία της εφαρμογής είναι η δυνατότητα απόκρυψης του profile από χρήστες οι οποίοι δεν είναι εξουσιοδοτημένοι (τα δύο μέλη δεν είναι φίλοι). Κάτι τέτοιο σημαίνει ότι το όνομα του χρήστη θα φαίνεται ως κείμενο και όχι ως σύνδεσμος. Αυτό επιτυγχάνεται με μια σειρά από queries τα οποία ελέγχουν διαδοχικά τους πίνακες.

Η πρώτη περίπτωση είναι ο χρήστης του οποίου το όνομα εμφανίζεται, να είναι διεγραμμένος:

```
$query20 = "SELECT * FROM deletedMembers WHERE deletedMid =
'$mId';";
```

Έπειτα ελέγχεται αν ο χρήστης είναι ο ίδιος του ο εαυτός οπότε και ο σύνδεσμος θα δείχνει κατευθείαν στο profile του:

```
$_SESSION['id'] == $mId;
```

Αν κάτι τέτοιο δεν ισχύει τότε θα ελεγχθεί αν οι δύο χρήστες βρίσκονται ο ένας στη μαύρη λίστα (BlackList) του άλλου:

```
$query5 = "SELECT bLFromMid FROM BlackList WHERE bLToMid =
'$id' AND bLFromMid = '$mId';";
```

Αν κάτι τέτοιο δεν ισχύει τότε θα ελεγχθεί αν οι δύο χρήστες είναι φίλοι:

```
$query = "SELECT cFromMid From Contacts WHERE (cToMid =
'$id' AND cFromMid = '$mId' AND confirmStatus LIKE 'y')
     UNION
SELECT cToMid FROM Contacts WHERE (cFromMid = '$id' AND
cToMid = '$mId' AND confirmStatus LIKE 'y');";
```

Σε περίπτωση που κάτι τέτοιο ισχύει τότε το profile είναι ορατό ενώ σε διαφορετική περίπτωση θα ελεγχθεί αν ο χρήστης αυτός έχει ορίσει στις επιλογές των ρυθμίσεών του (Settings) το profile του να είναι ορατό από όλους:

```
$query2 = "SELECT * FROM Settings WHERE sMid = '$mId';";
$result2 = mysqli_query($connection, $query2);
confirm_query($result2);
$row2 = mysqli_fetch_array($result2);
if ($row2['viewableProfile'] == 'a')
```

Αν κάτι τέτοιο ισχύει τότε το όνομά του εμφανίζεται με την μορφή συνδέσμου διαφορετικά με τη μορφή κειμένου.

9.16 Εμφάνιση τυχαίων φίλων

Στην αρχική αλλά και στην σελίδα του profile, εμφανίζεται ένας πίνακας με το πολύ 6 τυχαίους φίλους του χρήστη. Για να επιτευχθεί αυτό, πρώτα αποθηκεύονται όλοι οι φίλοι του χρήστη σε έναν πίνακα, κι έπειτα εφαρμόζεται η συνάρτηση shuffle πάνω σε αυτόν. Η συνάρτηση αυτή αλλάζει την σειρά των στοιχείων του πίνακα ώστε να υπάρχει κάθε φορά διαφορετικό αποτέλεσμα.

```
while ($row = mysqli_fetch_array($result)) {
     $fr[] = $row['id'];
}
shuffle($fr);
```

Σημειώνεται ότι η μεταβλητή $result περιέχει τα αποτελέσματα που επιστράφηκαν από την εκτέλεση του query για την εύρεση όλων των φίλων του χρήστη.

9.17 Αποδοχή μηνυμάτων

Κάθε χρήστης της εφαρμογής μπορεί να επιλέξει αν θα δέχεται μηνύματα από όλους ή μόνο από τους φίλους του. Αυτό επιτυγχάνεται εκτυπώνοντας ή όχι το αντίστοιχο κουμπί αποστολής μηνύματος ανάλογα με το αν ο άλλος χρήστης έχει το δικαίωμα να στείλει μήνυμα σε αυτόν τον χρήστη ή όχι. Θεωρούμε Μέλος1 τον χρήστη ο οποίος θέλει να στείλει μήνυμα και Μέλος2 τον χρήστη ο οποίος ανακτάται από την βάση και το όνομά του εμφανίζεται στην οθόνη του Μέλους1. Το ζητούμενο είναι αν στην οθόνη του Μέλους1 εμφανιστεί το κουμπί αποστολής μηνύματος.

Το αποτέλεσμα του παρακάτω πολύπλοκου query είναι να επιλέγει μόνο εκείνους τους χρήστες οι οποίοι είναι φίλοι του Μέλους1, εκείνους τους χρήστες οι οποίοι

δεν βρίσκονται στη μαύρη λίστα του Μέλους1 και το id τους δεν είναι ίδιο με το id του χρήστη, εκείνους τους χρήστες για τους οποίους το Μέλος1 δεν βρίσκεται στην μαύρη λίστα τους και το id αυτών που βρίσκονται δεν είναι ίδιο με το id του Μέλους1, καθώς και όλους τους χρήστες οι οποίοι δεν είναι φίλοι του Μέλους1 αλλά έχουν ορίσει στις επιλογές των ρυθμίσεών τους (Settings) να δέχονται μηνύματα από όλους.

```
$query = "SELECT mId FROM Members WHERE mId !='$mId' AND
mId IN         SELECT mId FROM Members WHERE mId IN (SELECT
cFromMid FROM Contacts WHERE cFromMid = '$friendOfFriend'
AND cToMid = '$mId' AND confirmStatus LIKE 'y' UNION SELECT
cToMid FROM Contacts WHERE cFromMid = '$mId' AND cToMid =
'$friendOfFriend' AND confirmStatus LIKE 'y'))
          UNION SELECT mId FROM Members WHERE mId !='$mId'
AND mId NOT IN (SELECT bLFromMid FROM BlackList WHERE
bLFromMid = '$friendOfFriend' AND bLToMid = '$mId'
          UNION SELECT bLToMid FROM BlackList WHERE
bLFromMid = '$mId' AND bLToMid = '$friendOfFriend') AND mId
IN ( SELECT sMid FROM Settings WHERE acceptMessages =
'a');";
```

9.18 Διαγραφή φίλων

Για την διαγραφή ενός ή και περισσότερων φίλων πρέπει πρώτα κατά την εμφάνιση των φίλων να δηλωθεί ένα πεδίο τύπου checkbox δίπλα από τον κάθε μέλος – φίλο ώστε επιλέγοντάς το να επιλέγεται το id του:

```
echo '<input type="checkbox" name="checkFriends[]"
id="checkFriends[]" value="'.$row['cId'].'">';
```

Σημειώνεται ότι το checkbox αυτό είναι πίνακας που έχει τιμή το id της κάθε εγγραφής του πίνακα Friends. Έτσι, κάθε checkbox που επιλέγεται έχει ως αποτέλεσμα την αποθήκευση του ανάλογου id.

Η μέθοδος που διαγράφει τους επιλεγμένους φίλους είναι η παρακάτω:

Αυτό που γίνεται εδώ είναι ότι ο πίνακας με τους επιλεγμένους φίλους περνάει στον πίνακα $check κι έπειτα για όσο αυτός ο πίνακας έχει και άλλα στοιχεία ο βρόγχος συνεχίζει να εκτελείται και οι σχετικές εγγραφές – φιλίες διαγράφονται. Σημειώνεται ότι η συνάρτηση count() επιστρέφει τον αριθμό των στοιχείων ενός πίνακα που περνά σε αυτή ως όρισμα. Το ίδιο επίσης κάνει και η συνάρτηση sizeof().

```
function delete_friends($connection) {
     $check = $_POST['checkFriends'];
     for ($i=0; $i<count($check); $i++) {
          $del = $check[$i];
          $query = "SELECT confirmStatus FROM Contacts
WHERE cId = '$del';";
          $result = mysqli_query($connection, $query);
          confirm_query($result);
          $confirmStatus = 'n';
          $row = mysqli_fetch_array($result);
          $query2 = "DELETE FROM Contacts WHERE cId =
'$del';";
          $result2 = mysqli_query($connection, $query2);
          confirm_query($result2);
     }
if (count($check) > 0) {
          echo "<meta http-equiv=\"refresh\"
content=\"0;URL=home.php? mod=friends&ref=yourfriends\">";
     }
}
```

Η εντολή:

```
"<meta http-equiv=\"refresh\"
content=\"0;URL=home.php?mod=friends& ref=yourfriends\">
```

είναι μία μεταεντολή και αυτό που κάνει είναι να ανανεώνει την σελίδα που αναφέρεται στην ιδιότητα URL στον συγκεκριμένο χρόνο που περιγράφεται στην παράμετρο πριν την URL (σε αυτήν την εντολή ο χρόνος είναι 0 δευτερόλεπτα).

9.19 Έλεγχος αναφοράς σελίδας

Θεωρώντας ότι εκτελείται η εντολή:

```
"<meta http-equiv=\"refresh\"
content=\"0;URL=home.php?mod=friends& ref=yourfriends\">
```

περιγράφεται ο τρόπος με τον οποίο η εφαρμογή καταλαβαίνει ποια σελίδα πρόκειται να εκτελεστεί κάθε φορά. Φορτώνοντας την διεύθυνση:

```
home.php?mod=messages&ref=inbox&action=showmessages&view=showall
```

Θα ακολουθηθούν τα εξής βήματα από το πρόγραμμα κάτι το οποίο θα απαιτήσει πολλούς ελέγχους σε πολλά σημεία του προγράμματος:

1. Κλήση της συνάρτησης get_mod($_GET[‘mod’]).
   Η συνάρτηση αυτή έχει ως αποτέλεσμα να δει ότι βρίσκεται στο mod με τιμή “messages” ώστε να δώσει στη σελίδα το ανάλογο CSS και JavaScript αρχείο.
2. Κλήση της συνάρτησης get_ref($_GET[‘ref]).
   Η συνάρτηση αυτή καθώς βλέπει ότι βρίσκεται στο ref με τιμή “inbox” θα αρχίσει να εκτελεί την συνάρτηση show_inbox_messages().
3. Κλήση της συνάρτησης get_action($_GET[‘action]).
   Η συνάρτηση αυτή αρχίζει ελέγχοντας αν υπάρχουν άλλες παράμετροι που πρέπει να ελεγχθούν. Στην περίπτωση αυτή διαπιστώνει ότι υπάρχει η μεταβλητή $_GET[‘view’] και καθώς αυτή έχει την τιμή “showall” θα εμφανίσει στην οθόνη όλα τα εισερχόμενα μηνύματα είτε αυτά έχουν αναγνωστεί είτε όχι.

Σε περίπτωση που η μεταβλητή action είχε, για παράδειγμα, την τιμή deletemessage τότε θα εκτελούνταν η συνάρτηση delete_one_message($m) με παράμετρο το μήνυμα τα οποίο θα ήταν προς διαγραφή. Φυσικά η αλληλουχία των κλήσεων συνάρτησης θα μπορούσε να συνεχιστεί ανάλογα με την τιμή των μεταβλητών που αναγράφονται στην URL.

9.20 Έλεγχος δικαιώματος πρόσβασης σελίδας

Πρέπει να αναφερθεί το σημαντικό ζήτημα εξουσιοδότησης σχετικά με την πρόσβαση ενός χρήστη σε κάποια σελίδα. Αν και οι περισσότερες επιλογές δίνονται στον χρήστη με τη μορφή κουμπιών και συνδέσμων και οι τιμές που πρόκειται να γραφτούν στο πεδίο της URL είναι προκαθορισμένες από το σύστημα, τίποτα δεν εμποδίζει από έναν χρήστη να γράψει απευθείας στην γραμμή URL. Έτσι, σε περίπτωση που ο χρήστης με id=10 βρίσκεται στην σελίδα των εισερχόμενων μηνυμάτων, μπορεί να δει το περιεχόμενο του καθενός από αυτά απλά κάνοντας κλικ σε κάθε ένα από τα μηνύματά του. Αυτό θα έχει ως αποτέλεσμα να φορτωθεί, για παράδειγμα, η σελίδα:

```
home.php?mod=messages&ref=inbox&action=viewcontent&messagei
d=90
```

δηλαδή θα βλέπει το μήνυμα με messageid=90 το οποίο του το έχει στείλει κάποιος άλλος χρήστης. Μέχρι εδώ δεν υπάρχει κανένα πρόβλημα… μέχρι φυσικά που ο χρήστης αυτός αποφασίσει να δει και άλλα μηνύματα τα οποία δεν είναι δικά του. Αυτό μπορεί να γίνει πολύ εύκολα, πληκτρολογώντας:

```
home.php?mod=messages&ref=inbox&action=viewcontent&messagei
d=110
```

Το μήνυμα με messageid=110 δεν απευθύνεται σε αυτόν αλλά αυτό δεν τον εμποδίζει να το διαβάσει αφού η αλληλουχία εκτέλεσης των συναρτήσεων είναι σωστή και τα δεδομένα “καθαρά”. Έτσι, για κάθε σελίδα πρέπει να γίνεται έλεγχος δικαιωμάτων πρόσβασης κάτι το οποίο επιτυγχάνεται με τα αντίστοιχα queries.
Για παράδειγμα, για την σελίδα των μηνυμάτων το query είναι:

```
if (($row['fromMid'] != $_SESSION['id'] && $row['toMid']
!= $_SESSION['id']) || $read == "y") {
     echo "<p>You have no access here</p>";
     return;
}
```

Αυτό σημαίνει ότι κανένας χρήστης με id διαφορετικό από το id του χρήστη για τον οποίο προορίζεται το μήνυμα ή id διαφορετικό από το id του αποστολέα του μηνύματος δε μπορεί να διαβάσει το μήνυμα. Αντί για αυτό εκτυπώνεται στην οθόνη το μήνημα “ You have no access here” και σταματάει η εκτέλεση της αντίστοιχης συνάρτησης.

9.21 Ανάρτηση αρχείου

Κάθε χρήστης μπορεί να αναρτήσει κάποιο αρχείο ώστε να το μοιραστεί με τους υπόλοιπους. Ο κώδικας που απαιτείται για την ανάρτηση οποιαδήποτε αρχείου είναι ο παρακάτω:

```
If ($_FILES['userfile']['error'] > 0) {
     echo '<p>Problem</p>';
     case 1: echo '<p>File exceeded
upload_max_filesize</p>'; break;
     case 2: echo '<p>File exceeded max_file_size</p>';
break;
     case 3: echo '<p>File only partially uploaded</p>';
break;
     case 4: echo '<p>No file uploaded</p>'; break;
return;
}
```

```
if (preg_match("/.exe$|.bat|.come$/i",
$_FILES['userfile']['name'])) {
     echo '<p>The type of the file is not valid</p>';
}
$upfile = "upload/files/".$_FILES['userfile']['name'];
if (file_exists($upfile)) {
     echo "<p>The file already exists.</p>";
     return;
}
if (is_uploaded_file($_FILES['userfile']['tmp_name'])) {
     if
(!move_uploaded_file($_FILES['userfile']['tmp_name'],
$upfile)) {
          echo "<p>Could not move file to destination
directory.</p>";
          return;
     }
} else {
     echo "<p>Problem to file
".$_FILES['userfile']['name']."</p>";
     return;
}
```

Αν δεν υπάρξει κανένα πρόβλημα τότε ο πίνακας λαθών θα έχει μήκος ίσο με 0 οπότε ο κώδικας θα συνεχίσει να εκτελείται ελέγχοντας την κατάληξη του αρχείου. Αν το αρχείο αυτό είναι εκτελέσιμο τότε για λόγους ασφαλείας απορρίπτεται. Σε διαφορετική περίπτωση ελέγχεται αν το αρχείο αυτό υπάρχει ήδη οπότε δεν επιτρέπεται να αναρτηθεί πάλι. Τελικά, αν περάσει όλους τους παραπάνω ελέγχους τότε γίνεται αναρτάται. Αν όμως κάτι δεν γίνει σωστά (διακοπή ρεύματος, διακοπή σύνδεσης κτλ) τότε το αρχείο δε θα αναρτηθεί στον server οπότε θα εμφανιστεί το αντίστοιχο μήνυμα.

9.22 Download αρχείου

Το κατέβασμα αρχείων από τον server είναι πολύ απλό. Το μόνο που απαιτείται είναι μόνο ένας σύνδεσμος που να δείχνει στην θέση του αρχείου στον server:

```
echo "<a href=\"upload/files/".$row['fName']."\"><img
src=\"images/download.png\" alt=\"Download Button\"
/></a>";
```

Σημειώνεται ότι ο παραπάνω σύνδεσμος είναι μία εικόνα.

9.23 BlackList

Κάθε μέλος (Μέλος1) έχει την επιλογή να προσθέσει κάποιο άλλο μέλος (Μέλος2) στη μαύρη λίστα του. Αυτό σημαίνει ότι το Μέλος2 δε θα μπορεί να στείλει μηνύματα στο Μέλος1 αλλά ούτε και να δει το profile του.

Τα μέλη που ο κάθε χρήστης έχει τη δυνατότητα να προσθέσει στη μαύρη λίστα του είναι όλα αυτά με τα οποία έχει αλληλεπιδράσει μέχρι τώρα, δηλαδή μπορεί να επιλέξει οποιοδήποτε μέλος είναι με αυτό φίλος, ή έχει στείλει ή αποδεχτεί μήνυμα, ή έχει ο ίδιος προστεθεί στο παρελθόν στη μαύρη λίστα από κάποιο μέλος. Ο κώδικας είναι ο παρακάτω:

```
$query = "SELECT mId, name FROM Members WHERE mId IN (
     SELECT DISTINCT mId FROM Members WHERE mId IN (
          SELECT fromMid FROM Messages WHERE toMid = '$mId'
          UNION
          SELECT toMid FROM Messages WHERE fromMid = '$mId'
          UNION
          SELECT cFromMid FROM Contacts WHERE cToMid =
'$mId'
          UNION
          SELECT cToMid FROM Contacts WHERE cFromMid =
'$mId') AND mId NOT IN (
     SELECT bLToMid FROM BlackList WHERE bLFromMid =
'$mId')
          AND mId != '$mId');";
```

9.24 Διαγραφή μέλους

Η διαγραφή κάποιου μέλους μπορεί να γίνει είτε αν το ίδιο το μέλος επιθυμεί να διαγραφεί είτε αν το διαγράψει ο διαχειριστής του συστήματος. Και στις δύο περιπτώσεις θα ακολουθηθούν τα ίδια βήματα:

1. Προσθήκη του μέλους στον πίνακα deletedMembers.
2. Διαγραφή του μέλους από τον πίνακα Online.
3. Διαγραφή του μέλους από τον πίνακα BlackList.
4. Διαγραφή οποιασδήποτε φιλίας του από τον πίνακα Contacts.
5. Σε περίπτωση που το μέλος έχει φωτογραφία στο profile του τότε αυτή διαγράφεται.
6. Μετατροπή του email του στην καθορισμένη τιμή “d”.
7. Μετατροπή όποιου μηνύματος έστειλε σε διαγραμμένο από τον αποστολέα.
8. Μετατροπή όποιου μηνύματος έλαβε σε διαγραμμένο από τον παραλήπτη.
9. Διαγραφή των μηνυμάτων που έχουν και τα δύο πεδία inboxDelete και sentDelete από τον πίνακα Messages.
10. Αποσύνδεση από την εφαρμογή.

Τα αντίστοιχα τμήματα κώδικα είναι τα παρακάτω:

```
1. $query = "INSERT INTO deletedMembers (deletedMid)
   VALUES ('$mId');";
2. $query2 = "DELETE FROM Online WHERE onMid = '$mId';";
3. $query3 = "DELETE FROM BlackList WHERE bLFromMid =
   '$mId' OR bLToMid = '$mId';";
4. $query4 = "DELETE FROM Contacts WHERE cFromMid =
   '$mId' OR cToMid = '$mId';";
5. unlink("upload/userimages/".$mId.".jpg");
6. $query5 = "SELECT * FROM Members WHERE mId = '$mId';";
7. $query6 = "UPDATE Members SET email = 'd' WHERE mId =
   '$mId';";
8. $query7 = "UPDATE Messages SET sentDelete = 'y' WHERE
   fromMid = '$mId';";
9. $query8 = "UPDATE Messages SET inboxDelete = 'y' WHERE
   toMid = '$mId';";
10.    $query9 = "DELETE FROM Messages WHERE sentDelete
   = 'y' AND inboxDelete = 'y';";
echo "<meta http-equiv=\"refresh\"
```

9.25 Αποσύνδεση χρήστη

Κατά την αποσύνδεση του χρήστη, ο αντίστοιχος χρήστης διαγράφεται από τον πίνακα Online και καταστρέφεται το cookie με το αντίστοιχο όνομα session_name(). Τέλος, ο χρήστης μεταφέρεται στην αρχική σελίδα.

Ο κώδικας είναι ο εξής:

```
$mId = $_SESSION['id'];
$query = "DELETE FROM Online WHERE onMid = '$mId';";
$result = mysqli_query($connection, $query);
confirm_query($result);
$_SESSION = array();
if (isset($_COOKIE[session_name()])) {
     setcookie(session_name(), '', time()-42000, '/');
}
redirect_to("index.php");
```

9.26 Σελιδοποίηση αποτελεσμάτων

Πολύ σημαντικό στοιχείο είναι η σελιδοποίηση των αποτελεσμάτων ανά κάποιο συγκεκριμένο αριθμό ώστε αυτά να μην εμφανίζονται σε μία μόνο σελίδα. Για παράδειγμα στην σελίδα των αρχείων, εμφανίζονται όλα τα αρχεία σελιδοποιημένα ανά 10.

Πρώτα γίνεται καταμέτρηση των αρχείων που υπάρχουν στον πίνακα Files, ορίζεται ότι θα εμφανίζονται 10 αποτελέσματα ανά σελίδα, υπολογίζονται πόσες σελίδες απαιτούνται για την εμφάνιση όλων των αποτελεσμάτων και τέλος καθορίζεται η τρέχουσα σελίδα.

Στην συνέχεια επιλέγονται μόνο εκείνα τα αρχεία από την τιμή $offset και μετά.
Τέλος, καθορίζεται πόσοι σύνδεσμοι θα εμφανίζονται, στην προκειμένη περίπτωση 5, αλλά και η μορφή του τρέχων συνδέσμου αλλάζει και γίνεται μορφής παραγράφου.

```
$query5 = "SELECT COUNT(*) AS num FROM Files; ";
$result5 = mysqli_query($connection, $query5);
confirm_query($result5);
$row5 = mysqli_fetch_array($result5);
$numRows = $row5['num'];
$rowsPerPage = 10;
$totalPages = ceil($numRows / $rowsPerPage);
if (isset($_GET['currentpage']) &&
is_numeric($_GET['currentpage'])) {
     $currentPage = (int) $_GET['currentpage'];
} else {
     $currentPage = 1;
}
if ($currentPage > $totalPages) {
     $currentPage = $totalPages;}
if ($currentPage < 1) {
     $currentPage = 1;
}
$offset = ($currentPage - 1) * $rowsPerPage;
```

```
$query = "SELECT * FROM Files WHERE fUploadDate < '$today'
ORDER BY fUploadDate $order LIMIT $offset, $rowsPerPage;";
while ($row = mysqli_fetch_array($result)) {
     …
     …
}
```

```
$range = 5;
if ($currentPage > 1) {
     $prevPage = $currentPage - 1;
     echo " <a
href='home.php?mod=files&ref=upload&currentpage=
$prevPage#s'>Προηγούμενο</a> ";
}
for ($x = ($currentPage - $range); $x < (($currentPage +
$range) + 1); $x++) {
      if (($x > 0) && ($x <= $totalPages)) {
           if ($x == $currentPage) {
               echo " [<b>$x</b>] ";
           } else {
                echo " <a
href='home.php?mod=files&ref=upload&
currentpage=$x#s'>$x</a> ";
               }
          }
     }
if ($currentPage != $totalPages) {
     $nextPage = $currentPage + 1;
     echo " <a
href='home.php?mod=files&ref=upload&currentpage=
$nextPage#s'>Επόμενο</a> ";
}
```

9.27 Εμφάνιση αποτελεσμάτων

Τα αποτελέσματα εμφανίζονται σε διαδοχικές σειρές οι οποίες βρίσκονται μέσα σε HTML div. Παρακάτω εξηγείται ο κώδικας της εμφάνισης των αρχείων καθώς και οι αντίστοιχοι κανόνες CSS.

```
$cl = odd_even($mode);
echo '<div class="sameRow '.$cl.'" id="'.$row['fId'].'">';
     echo '<div class="picDiv"';
          echo …
     echo '</div>';
     echo '<div class="profileDiv">';
          echo …
     echo "</div>";
     echo '<div class="fileNameDiv">';
          …
     echo '</div>';
     echo '<div class="fileSubjectDiv">';
          …
     echo '</div>';
     echo '<div class="fileBranchDiv">';
          …
     echo '</div>';
     echo '<div class="downloadDiv">';
          …
     echo '</div>';
echo '</div>';
$mode++;
```

Για τη διαμόρφωση της κάθε σειράς αποτελεσμάτων πρέπει να υπάρχει κάποιος τρόπος διαχωρισμού τους. Αυτό γίνεται δίνοντας σε κάθε div σειράς την κλάση sameRow ακολουθούμενη από την κλάση η οποία κάθε φορά καθορίζεται από την τιμή της μεταβλητής $cl η οποία μπορεί να είναι odd ή even. Με τον διαχωρισμό αυτό διαδοχικές σειρές παίρνουν και διαφορετικό χρώμα.

```
.odd {
     background-color: #DFDFDF;
     border-bottom: 2px solid #94A3C4;
     border-right: 2px solid #94A3C4;}
.even {
     background-color: #CCC;
     border-bottom: 2px solid #94A3C4;
     border-right: 2px solid #94A3C4;}
```

Κατόπιν, κάθε σειρά διασπάται σε μικρότερα div τα οποία διακρίνονται από την δικιά τους κλάση.

```
.picDiv,   .profileDiv,   .fileNameDiv,   .fileSubjectDiv,
.fileBranchDiv, .fileUploadDateDiv, .downloadDiv {
     display: inline-block;
     text-align: center;
     vertical-align: middle;}
```

Εδώ πρέπει να σημειωθεί ότι ο Internet Explorer παρουσιάζει πρόβλημα στο να εμφανίζει inline-block κι έτσι αυτά θα εμφανιστούν λανθασμένα. Εναλλακτική λύση είναι αντί για διαφορετικά div να χρησιμοποιηθεί πίνακας με τα αντίστοιχα tr και td για την εμφάνιση των αποτελεσμάτων. Πάντως, η χρήση πίνακα κατά τον σχεδιασμό του layout δεν ενδείκνυται.

9.28 Έλεγχος δεδομένων με JavaScript

Ο έλεγχος των δεδομένων που εισάγει ο χρήστης για το πεδίο του θέματος του αρχείου που πρόκειται να αναρτηθεί γίνεται ως εξής:

Πρώτα ελέγχεται αν ο browser υποστηρίζει χρήση DOM. Αν δεν υποστηρίζει τότε η συνάρτηση επιστρέφει απευθείας true. Διαφορετικά, επιλέγεται το στοιχείο με id=”formUploadFile”, καταμετρούνται τα στοιχεία του και ελέγχεται το στοιχείο του με τύπο textarea. Αν η τιμή του στοιχείου αυτού έχει μήκος πάνω από 150 χαρακτήρες τότε η flag γίνεται false. Κατόπιν μετρούνται όλες οι παράγραφοι του στοιχείου με id=”data” και αν αυτές δεν ξεπερνάν τη 1 τότε σημαίνει ότι είναι η πρώτη φορά που εισήχθηκαν λανθασμένα δεδομένα κι έτσι δημιουργείται

αυτόματα μία νέα παράγραφος στην οποία δίνεται η κλάση Warning και προστίθεται το κείμενο txt.

Ο κώδικας για όλα τα προηγούμενα είναι ο παρακάτω:

```
function validate_upload_file() {
     if (!document.getElementsByTagName) return true;
     var allOk = true;
     var formUploadFile =
document.getElementById("formUploadFile");
     var formElements = formUploadFile.elements;
     var txt = "";

     for (var a=0; a < formElements.length; a++) {
          if (formUploadFile.elements[a].type ==
"textarea") {
               if (formUploadFile.elements[a].value.length
> 150) {
                    allOk = false;
                    txt = "Το πεδίο του θέματος δεν πρέπει
να ξεπερνάει τους 150 χαρακτήρες.";
                    break;
               }
          }
     }
     var data = document.getElementById("data");
     var allP = data.getElementsByTagName("p");
     if (allP.length == 1 && allOk == false) {
          var newP = document.createElement("p");
          data.appendChild(newP);
          newP.className = "warning";
          var txtNode = document.createTextNode(txt);
          data.lastChild.appendChild(txtNode);
     } else if (allOk == false) {
          data.lastChild.childNodes[0].nodeValue = txt;
```

Οδηγός Χρήσης Λογισμικού

Στο τμήμα αυτό δίνονται οδηγίες σχετικά με την περιήγηση του χρήστη στην εφαρμογή.

- Ο χρήστης μπορεί να εγγραφεί ή να συνδεθεί στην αρχική σελίδα, η οποία είναι:

http://kostptyx.net76.net/index.php

- Στην σελίδα “Home” μπορεί να δει πληροφορίες σχετικά με τις τελευταίες ανανεώσεις που συνέβηκαν σχετικά με αρχεία, φιλίες, αλλαγές profile και μηνύματα.
- Στην σελίδα “Profile” μπορεί να δει τις πληροφορίες του λογαριασμού του αλλά και να κάνει αλλαγές πάνω σε αυτόν.
- Στην σελίδα “Φίλοι” μπορεί να δει όλους τους φίλους του αλλά και να κάνει αναζήτηση επιλεγμένων φίλων.
- Στην σελίδα “Μηνύματα” μπορεί να αναγνώσει τα μηνύματα που έχει στείλει ή έχει δεχτεί καθώς επίσης και να στείλει νέα μηνύματα στους υπόλοιπους χρήστες.
- Στην σελίδα “Αναζήτηση” μπορεί να κάνει αναζήτηση στα μέλη της εφαρμογής.
- Η σελίδα “Αρχεία” δίνει την δυνατότητα στους χρήστες να αναρτήσουν τα δικά τους αρχεία αλλά και να αναζητήσουν και να κατεβάσουν ήδη υπάρχοντα.
- Στην σελίδα “Προκηρύξεις” οι χρήστες μπορούν να ενημερωθούν για τις καινούριες προκηρύξεις του ΑΣΕΠ.
- Στην σελίδα “Ρυθμίσεις” οι χρήστες μπορούν να καθορίσουν τις προσωπικές τους ρυθμίσεις όσον αφορά την ιδιοτικότητα των στοιχείων τους.

Σημειώνεται ότι πληκτρολογώντας στην αρχική σελίδα της εφαρμογής:

http://kostptyx.net76.net/index.php?admin=admin

δίνεται η δυνατότητα να εισέλθει ο διαχειριστής του συστήματος.

Βιβλιογραφία

Welling, L., Laura, T., (2005), *PHP and MySQL Web Development, 3rd Edition*, Sams Publishing, Indiana, USA.

Keith, J., (2005), *Web Design with JavaScript and the Document Object Model*, Apress, Berkeley, USA.

Heilmann, C., (2006), *Beginning JavaScript with DOM Scripting and Ajax: From Novice to Professional*, Apress, Berkeley, USA.

Croft, J., Lloyd, I., Rubin, D., (2006), *Pro CSS Techniques*, Apress, Berkeley, USA.

Griffiths, P., (2007), *The Best-Practice Guide to XHTML & CSS*, HTML Dog, Berkeley, USA.

Shultz, D., Cook, C., (2007), *Beginning HTML with CSS and XHTML: Modern Guide and Reference*, Apress, Berkeley, USA.

Pfaffenberger, B., Shcafer, S., White, C., Karow, B., (2004), *HTML, XHTML, and CSS Bible, 3rd Edition*, Wiley Publishing, Inc., Indiana, USA.