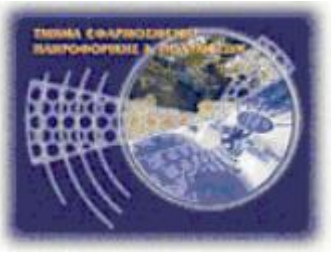
Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Εφαρμοσμένης Πληροφορικής και Πολυμέσων



Πτυχιακή εργασία

Μηχανογράφηση ενος φροντιστηρίου μέσης εκπαίδευσης

Εμμανουήλ Δραμουντάνης (Α.Μ. 2399)

Ευάγγελος Δερμιτζάκης (Α.Μ. 2503)

Επιβλέπων Καθηγητής: Νίκος Παπαδάκης

Abstract

The education sector is quite a fertile ground for the use of computer applications, given the large volume of data required to monitor the progress of students. In this paper we develop a computer application using a relational database for data storage and modern internet technologies to create a user interface management and data entry.

Furthermore we present the necessary procedures for the creation of such an application and issues encountered during their use.

Σύνοψη

Ο τομέας της εκπαίδευσης είναι ένα αρκετά πρόσφορο έδαφος για την χρήση εφαρμογών μηχανογράφησης, δεδομένου του μεγάλου όγκου δεδομένων που απαιτούνται για την παρακολούθηση της πορείας των μαθητών. Σε αυτή την εργασία αναπτύσσουμε μία εφαρμογή μηχανογράφησης με τη χρήση μίας σχεσιακής βάσης για την αποθήκευση των δεδομένων και σύγχρονων τεχνολογιών διαδικτύου για την δημιουργία ενός εύχρηστου περιβάλλοντος διαχείρισης και εισαγωγής των δεδομένων.

Επιπλέον παρουσιάζουμε τις απαραίτητες διαδικασίες για την δημιουργία μίας τέτοιας εφαρμογής και τα θέματα που συναντήσαμε κατά την χρήση τους.

Περιεχόμενα

1 Εισαγωγή

1.1 Περίληψη

Αυτή η εργασία ασχολείται με μία σελίδα μηχανοργάνωσης φροντιστηρίου. Η εφαρμογή που έχουμε φτιάξει είναι μία σελίδα με τη γλώσσα PHP που γράφει και διαβάζει τα δεδομένα της στη γλώσσα HTML. Θέλουμε να δείξουμε πώς γίνεται να φτιαχτεί μία τέτοια σελίδα και πώς μπορεί να χρησιμοποιηθεί η ίδια μέθοδος για την ανάπτυξη άλλων σελίδων.

1.2 Κίνητρο για την διεξαγωγή της εργασίας

Το internet είναι μία τεχνολογία που χρησιμοποιείται κατά κόρων στις μέρες μας από πάρα πολλούς ανθρώπους. Οι σελίδες σε αυτό χρησιμοποιούνται για πάρα πολλούς στόχους και επιτρέπουν πέρα από την προβολή πληροφοριών και την ανάπτυξη συστημάτων διαχείρισης και μηχανογράφησης.

Θελήσαμε να δούμε πώς μπορεί να εφαρμοστεί αυτή η τάση και στη διαχείριση ενός εκπαιδευτικού ιδρύματος.

1.3 Σκοπός και στόχοι εργασίας

Με την εργασία μας θέλουμε να δείξουμε πως φτιάξαμε την εφαρμογή μας και πώς χρησιμοποιήσαμε τις γλώσσες προγραμματισμού HTML, PHP και JavaScript για να πετύχουμε τον σκοπό μας. Θα δούμε διεξοδικά όλα τα βήματα που ακολουθήσαμε για να φτιάξουμε το τελικό αποτέλεσμα.

Έπειτα θα πούμε τα συμπεράσματά μας για αυτές τις τεχνολογίες και τι μπορεί να γίνει για να βελτιωθεί ακόμη περισσότερο η εφαρμογή μας από μελλοντικούς ερευνητές.

1.4 Δομή εργασίας

Η εργασία είναι διαρθρωμένη με τον ακόλουθο τρόπο:

- Το πρώτο κεφάλαιο είναι αυτή η εισαγωγή
- Το δεύτερο κεφάλαιο εισάγει κάποιες βασικές θεωρητικές έννοιες.
- Το τρίτο κεφάλαιο μιλάει για τις τεχνολογίες και τα εργαλεία που έχουν χρησιμοποιηθεί.
- Το τέταρτο κεφάλαιο αναλύει τη δομή της εφαρμογής.

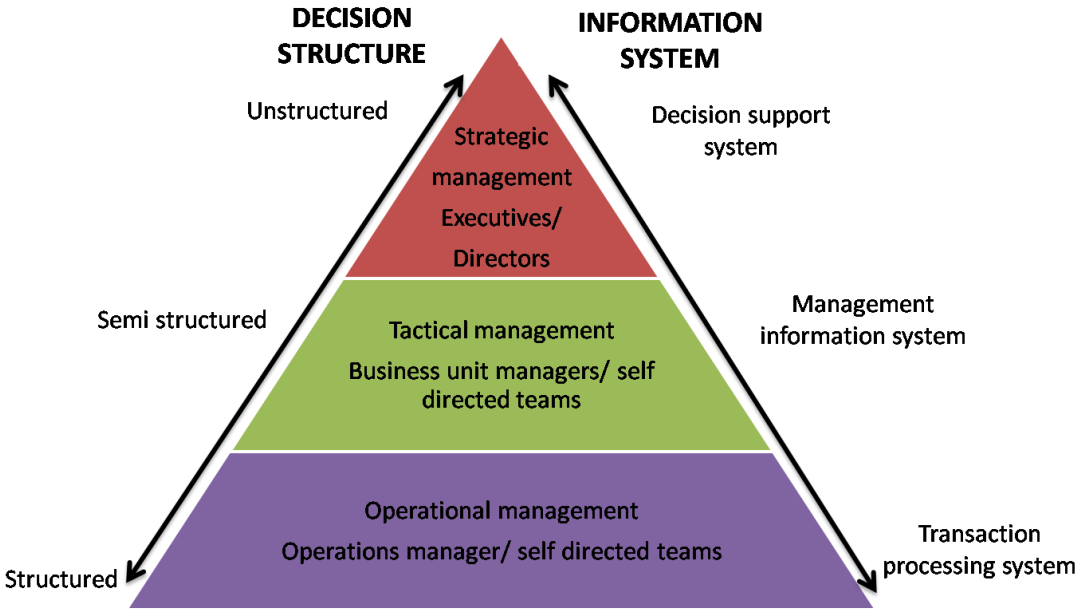
2 Γενικές και Βασικές Έννοιες

2.1 Μηχανογράφηση

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

Στην εποχή που ζουμε, η σπουδαιότητα μηχανογράφησης παίζει καθοριστικό ρόλο ως ανταγωνιστικό πλεονέκτιμα απέναντι τόσο μέσα στην ίδια την εταιρία είτε οργανισμό όσο και εξωτερικά. Η σωστή διαχείρηση της πληροφορίας αλλα και ο συνηφασμός της φυσης μας με την τεχνολογία μπορούν να δόσουν το καλυτερο αποτέλεσμα σχετικά με το πλάνο πορείας οργάνωσης και εξέλληξης. Η τεχνολογία προχωράει με γοργούς ρυθμούς και δεν επιτρέπει λάθος χειρισμούς τόσο για την σωστή ενημέρωση όσο και για την σωστή επεξεργασία των πόρων που υπάρχουν. Δεν χρειάζεται να γίνεται σπατάλη πόρων αλλά και καταπόνηση πνευματικής υγείας.

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



Εικόνα 2-1 Η πυραμύδα των πληροφοριακών συστημάτων.

2.2 Πληροφοριακό Σύστημα

Αρχικά σε επιχειρήσεις και άλλους οργανισμούς ,ο εσωτερικός έλεγχος γινόταν χειροκίνητα και μόνο περιοδικά ως παράγωγο του λογιστικού συστήματος και με κάποιες πρόσθετες στατιστικές πληροφορίες που δινόταν για την απόδοση της διαχείρισης καθυστερημένα και περιορισμένα. Τα δεδομένα οργανώνονταν με μη αυτόματο τρόπο και σύμφωνα με τις απαιτήσεις και τις ανάγκες του οργανισμού. Με την ανάπτυξη της πληροφορικής , η πληροφορία άρχισε να διαχωρίζεται από τα δεδομένα και αναπτύχθηκαν

συστήματα για την παραγωγή και την οργάνωση λήψεων ,περιλήψεων ,σχέσεων και γενικεύσεων βασισμένων στα δεδομένα.

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

Τα πληροφοριακά συστήματα αποτελούν το μέσο για την αρμονική συνεργασία ανθρώπινου δυναμικού, δεδομένων, διαδικασιών και τεχνολογιών πληροφορίας και επικοινωνιών. Προέκυψαν ως γέφυρα μεταξύ των πρακτικών εφαρμογών της επιστήμης υπολογιστών και του επιχειρηματικού κόσμου. Κάθε ειδικό πληροφοριακό σύστημα έχει ως στόχο την υποστήριξη των επιχειρήσεων, τη διαχείριση και λήψη αποφάσεων. Σε μια ευρεία έννοια, ο όρος χρησιμοποιείται για να αναφερθεί όχι μόνο στην τεχνολογία της πληροφορίας και της επικοινωνίας (ΤΠΕ), που ένας οργανισμός χρησιμοποιεί, αλλά στο τρόπο με τον οποίο οι άνθρωποι αλληλεπιδρούν με αυτή την τεχνολογία για την υποστήριξη των επιχειρηματικών διαδικασιών.

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

2.3 Πληροφοριακό Σύστημα Διοίκησης

Ένα Πληροφοριακό Σύστημα Διοίκησης παρέχει πληροφορίες που χρειάζονται για να διαχειρίζονται οι οργανισμοί αποδοτικά και αποτελεσματικά. Τα Πληροφοριακά Συστήματα Διοίκησης περιλαμβάνουν τρεις βασικές πηγές: ανθρώπους ,τεχνολογία και πληροφορία. Τα Πληροφοριακά Συστήματα Διοίκησης είναι διακριτά από τα άλλα πληροφοριακά συστήματα τα οποία χρησιμοποιούνται για την ανάλυση λειτουργικών λειτουργιών στον οργανισμό. Ακαδημαϊκά, ο όρος χρησιμοποιείται συνήθως για να αναφερθεί στην ομάδα των μεθόδων διαχείρισης πληροφοριών που είναι συνδεδεμένες με την αυτοματοποίηση ή στηρίζουν την ανθρώπινη λήψη αποφάσεων.

2.4 Βαση Δεδομένων

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

Ειδικότερα, στην επιστήμη της πληροφορικής και στην καθημερινή χρήση των ηλεκτρονικών υπολογιστών, με τον όρο βάσεις δεδομένων αναφερόμαστε σε οργανωμένες, διακριτές συλλογές σχετιζόμενων δεδομένων ηλεκτρονικά και ψηφιακά αποθηκευμένων, στο λογισμικό που χειρίζεται τέτοιες συλλογές (Σύστημα Διαχείρισης Βάσεων Δεδομένων, ή DBMS) και στο γνωστικό πεδίο που το μελετά. Πέρα από την εγγενή της ικανότητα να αποθηκεύει δεδομένα, η βάση δεδομένων παρέχει μέσω του σχεδιασμού και του τρόπου ιεράρχησης των δεδομένων, τα αποκαλούμενα συστήματα διαχείρισης περιεχομένου, δηλαδή τη δυνατότητα γρήγορης άντλησης και ανανέωσης των δεδομένων[4].

Εικόνα 2-2 Η ενοποίηση βάσεων δεδομένων.

2.5 Σύστημα Διαχείρισης Βάσης Δεδομένων

Με τον όρο γνωστό ως Database Management system (DBMS) εννοείται είτε κάποιο λογισμικό μέσω του οποίου γίνεται η δημιουργία, η διαχείριση, η συντήρηση και η χρήση μιας ηλεκτρονικής βάσης δεδομένων, ανάλογα με τον τύπο βάσης δεδομένων που επιλέγεται ή ένα σύνολο αλληλοσυσχετιζόμενων προγραμμάτων που τρέχουν και διαχειρίζονται τα δεδομένα μιας τέτοιας βάσης. Το λογισμικό χρησιμοποιεί στερεότυπες (standard) μεθόδους καταλογοποίησης, ανάκτησης, και εκτέλεσης ερωτημάτων σχετικών με τα δεδομένα. Το σύστημα διαχείρισης οργανώνει τα εισερχόμενα δεδομένα με τρόπους χρησιμοποιήσιμους από εξωτερικούς χρήστες

Ιδωμένο από μία άλλη οπτική γωνία, το σύστημα διαχείρισης βάσης δεδομένων είναι ένας διαχειριστής αρχείων (file manager) που διαχειρίζεται δεδομένα σε βάσεις δεδομένων παρά αρχεία σε συστήματα αρχείων, τα οποία είναι μία άλλη μορφή βάσης δεδομένων

3 Τεχνολογίες && Εργαλεία που χρησιμοποιήθηκαν

Οι Τεχνολογίες που θα παρουσιαστουν παρακάτω κρίθηκαν απαραίτητες για την υλοποιήση του πληροφοριακού συστήματος για την εκπαίδευση.

3.1 Γλώσσες Προγραμματισμού

3.1.1 HTML

3.1.1.1 Εισαγωγή

Η βασική ανάγκη που κάλυψε η γλώσσα HTML και η αιτία δημιουργίας της ήταν η προβολή απλών σελίδων κειμένου και την σύνδεση τους με άλλες. Στην συνέχεια με την εξάπλωση του διαδικτύου, η ανάγκη κατασκευής πιό εντυπωσιακών σελιδών που θα ήταν και πίο φιλικές προς το χρήστη καθώς και τα τελευταία χρόνια ολόκληρων εφαρμογών είτε πληροφοριακών συστημάτων, δυνάμωσε την ανάγκη χρήσης την γλώσσας αυτής και λόγω απαιτήσεων και αναγκών φυσικά υλοποίησης εξελήχθηκε σε νέες εκδόσεις ώστε να μπορει να υποστηρίζει τις απαιτήσεις υλοποιησης των προγραμματιστών και τις ανάγκες χρήστων.

Το πρότυπο της HTML αναθεωρήθηκε αρκετά με την τελευταία έκδοσή του 4.01 να έχει οριστικοποιηθεί το 1999. Καθώς οι απαιτήσεις από την γλώσσα αυξανόντουσαν αναπτύχθηκαν πολλές τεχνολογίες με σκοπό να καλύψουν τις ελλείψεις της γλώσσας οι οποίες με τη σειρά τους δημιούργησαν νέα προβλήματα, καθώς χρειάζεται να υποστηρίζονται από τα αντίστοιχα προγράμματα που χρησιμοποιούν οι χρήστες, κάτι που δεν ήταν δεδομένο.

Εξαιτίας ότι οι απαιτήσεις από την HTML αυξανόντουσαν όπως προαναφέρθηκε και παραπάνω, το πρότυπο της αναθεωρήθηκε αρκετα μετά την τελευταία έκδοση του 4.01 πριν το 1999 που οριστικοποιηθηκε. Η τεχνολογία και οι προγραμματιστικές απατήσεις επαίτειναν τη ν κατάσταση για επόμενη έκδοση ακόμα περισσότερο καθώς νέες τεχνολογίες αναπτύχθηκαν με σκοπό να καλύψουν τις ελλείψεις που δημιουργόντουσανώστε να γίνεται σωστά η υποστήριξη προγραμμάτων και εμφάνισης δεδομένων. Η επόμενη έκδοση ήταν η 5 και γνωστη σε όλους μας HTML5, , η οποία άρχισε να αναπτύσσεται το 2004 και αναμένεται να οριστικοποιηθεί μετά το 2015.

Η HTML5 είναι η νέα γλώσσα που προσφέρει έναν όγκο δυνατοτήτων με κύριο σκοπό να να μην είναι απαραίτητη η χρήση ξένων εργαλείων για την δημιουργία ιστοσελίδας και για την υποστήρηξη αυτών από φυλλομετρητες ιστοσελίδων .Έτσι η νέα έκδοση κάνει την δημιουργία ιστοσελίδων πολύ πιο εύκολη και πολλές από τις δυνατότητες της έχουν ήδη ενσωματωθεί στους υπάρχοντες φυλλομετρητές για την ευκολότερη υποστηρίξη και εμφάνιση διαδικτυακών εφαρμογων και ιστοσελίδων.

3.1.1.2 Ιστορία της HTML

Το ENQUIRE , το οποίο επινόησε ο φυσικός Τιμ Μπέρνερς Λι το 1980, αποτέλεσε τον μακρινό πρόγονο της HTML και ήταν ένα σύστημα που χρησιμοποιούσαν οι ερευνητές του CERN, και ήταν σύστημα χρήσης και διαμοιρασμού εγγράφων. Αργότερα, ο Μπέρνερς Λι, το 1989, πρότεινε ένα σύστημα βασισμένο στο διαδίκτυο που θα χρησιμοποιούσε υπερκείμενο. Έτσι, στα τέλη του 1990, φτιάχτηκε η προδιαγραφή της HTML και γράφτηκε ο browser και το λογισμικό εξυπηρετητή από τον ίδιο. Επίσης, τον ίδιο χρόνο, ο Μπέρνερς Λι σε συνεργασία με τον Robert Cailliau, μηχανικός συστημάτων πληροφορικής του CERN, προσπάθησαν να βρουν χρηματοδότηση από το CERN αλλά δυστυχώς το έργο δεν υιοθετήθηκε ποτέ επίσημα. Ο Μπέρνερς Λι αριθμεί «μερικές από τις πολλές χρήσεις του υπερκειμένου» στις προσωπικές του σημειώσεις από το 1990, και αναφέρει πρώτα από όλες μια εγκυκλοπαίδεια.

Εικόνα 3-1 Η HTML από σκοπιά ανάλυσης.

Ένα έγγραφο με το όνομα Ετικέτες HTML (HTML tags) ήταν η πρώτη δημόσια διαθέσιμη περιγραφή της HTML, καθώς πρωτοαναφέ, στα τέλη του 1991, ο Μπέρνερς Λι στο Διαδίκτυο. Το έγγραφο αυτό έδινε την περιγραφη των 20 στοιχειών τα οποία αποτελούσαν τον αρχικό και παράλληλα απλό σχεδιασμό της HTML. Όλες οι ετικέτες ήταν έντονα επηρεασμένες από την SGMLguid, μια μορφή δημιουργίας τεκμηρίωσης, φτιαγμένη στο CERN και βασισμένη στην SGML εκτός από την ετικέτα υπερσυνδέσμου. Αξιοσημείωτο αποτελεί οτι δεκατρία από εκείνα τα αρχικά στοιχεία υπάρχουν ακόμα σήμερα στην HTML 4.

Το πρότυπο SGML αποτελεί εκτός από μια απλή μίμηση της τυπογραφίας και αναπαραγωγή μερικών τεχνικών που χρησιμοποιούσαν οι τυπογράφοι και προσθέτει μια γενικευμένη σήμανση βασισμένη σε στοιχεία, τα οποία έχουν την δυνατότητα να μπορούν να εμφωλεύονται το ένα μέσα στο άλλο και να εμφωλεύουν τις ιδιότητες του «αντικειμένου» και ετικέτας. Ακόμα, Ο διαχωρισμός της δομής από το περιεχόμενο αποτελεί βασική λογική που είναι βασισμένο το SGML. Η η HTML, με τα CSS ακολούθησαν αυτή την αρχή ώς κύριο γνώμονα στους. Η τεχνική αναφορά ISO TR 9537, Techniques for using SGML (τεχνικές χρήσης της SGML) αποτελεί την έμπνευση πολλών στοιχείων του προτύπου. Τα TYPSET και RUNOFF είχαν αναπτυχθεί στις αρχές της δεκαετίας του 1960 για το λειτουργικό

σύστημα CTSS μέσω αυτής της τεχνικής και καλύπτει τα χαρακτηριστικά των πρώιμων γλωσσών μορφοποίησης κειμένου που χρησιμοποιούνταν από αυτά.

Ως μια υλοποίηση του SGML θεωρήθηκε η HTML από τον Μπέρνερς Λι. Με τη δημοσίευση της πρώτης πρότασης για μια προδιαγραφή της HTML, στα μέσα του 1993, επισημοποιήθηκε και ορίστηκε από το Internet Engineering Task Force (IETF). Η πρόταση αυτή περιλάμβανε και έναν ορισμό τύπου εγγράφου (DTD, Document Type Definition) της SGML, ο οποίος όριζε την γραμματική που ήταν βασιμένη. Μετά την πάροδο έξι μηνών, το πρόχειρο (draft) αυτό έληξε αλλά κάτι αξιοσημείωτο περιέχεται το οποόι ειναι ότι η αναγνώριση της ετικέτας του NCSA Mosaic για την ενσωμάτωση εικόνων μέσα στο κείμενο.Είναι αξιοσημείωτο γιατί αντικατοπτρίζει την φιλοσοφία του IETF για ενσωμάτωση μέσα στα πρότυπα επιτυχημένων πρωτότυπων. Το «HTML+ (Hypertext Markup Format)», το ανταγωνιστικό πρόχειρο του Dave Raggett, περιείχε κατι παρόμοιο, πρότεινε την προτυποποίηση μερικών ήδη υλοποιημένων δυνατοτήτων, όπως οι πίνακες και οι φόρμες.

Στις αρχές του 1994,το IETF δημιούργησε την Ομάδα Εργασίας για την HTML ,μετά που τα πρόχειρα HTML και HTML+ είχαν λήξει, η οποία το 1995 ολοκλήρωσε την «HTML 2.0», με την πρόθεση να αποτελέσει την πρώτη προδιαγραφή όπου οι μελλοντικές υλοποιήσεις θα βασίζονταν πάνω σε αυτή. Η HTML 2.0 περιείχε ιδέες από τα πρόχειρα HTML και HTML+. Δημοσιεύτηκε ως RFC 1866. Ο σκοπός της αρίθμησης 2.0 ήταν απλά για να ξεχωρίσει την νέα έκδοση από τα πρόχειρα που προηγήθηκαν.

Η HTML 3.0 προτάθηκε ως πρότυπο από το IETF. Περιείχε πολλές δυνατότητες, όπως τη ροή κειμένου γύρω από εικόνες, την υποστήριξη για πίνακες και την προβολή πολύπλοκων μαθηματικών τύπων. Πολλές δυνατότητες συμπεριλαμβάνονταν στην πρόταση του Raggett για την HTML+. Η πρόταση αυτή έληξε πέντε μήνες αργότερα χωρίς άλλη ενέργεια.

Η ανάπτυξη του Arena browser ως δοκιμαστική πλατφόρμα για την HTML 3 και για τα CSS ξεκίνησε από το W3C αλλά η HTML 3.0 δεν πέτυχε, για διάφορους λόγους. Αρχικά, αφενός το πρόχειρο θεωρήθηκε υπερβολικά μεγάλο καθώς αποτελούταν από 150 σελίδες αφεταίρου, ο ρυθμός ανάπτυξης του browser, καθώς και ο αριθμός των ενδιαφερομένων μερών υπερέβαιναν τις δυνατότητες του IETF. Έτσι οι εταιρείες που διέθεταν browser, όπως η Microsoft και η Netscape εκείνο τον καιρό, αποφάσισαν να εισάγουν τις δικές του επεκτάσεις στο πρόχειρο της HTML 3 καθώς επέλεξαν να υλοποιήσουν διαφορετικά υποσύνολα των δυνατοτήτων του πρόχειρου της . Οι επεκτάσεις αυτές επικεντρωνόταν σε στοιχεία που είχαν να κάνουν με τον έλεγχο εμφανισης και παρουσίασης των εγγράφων, όσο και αν εναντιωνόταν με την ακαδημαϊκή κοινότητα μηχανικών ότι στοιχεία όπως το χρώμα, το μέγεθος, ο τύπος της γραμματοσειράς και το παρασκήνιο, ήταν οπωσδήποτε έξω από το στόχος μιας γλώσσας και λεπτομέρειες καθώς η μοναδική πρόθεση ύπαρξης της γλώσσας αυτής ήταν να καθορίσει πώς οργανώνεται ένα έγγραφο. Βέβαια οι εταιρίες ΄και πιο ειδικά η Microsoft σχολιάστηκε για αυτή της την τροπή επέκτασης και πιο συγκεκριμένα ο Dave Raggett, συνεργάτης του W3C για πολλά χρόνια, σχολίαζε ότι «Μέχρι ενός σημείο, η Microsoft έκτισε την επιχειρηματική της δραστηριότητα στον Ιστό επεκτείνοντας τις δυνατότητες της HTML.»

Λόγω σύγκρουσης συμφερόντων που δημιουργήθηκαν, η περαιτέρω ανάπτυξη κάτω από την επίβλεψη του IETF καθυστέρησε. Οι προδιαγραφές της HTML τηρούνται, μαζί με ανάδραση από τους δημιουργούς λογισμικού, από το World Wide Web Consortium (W3C) από το 1996 και μετά. Εν το μεταξύ, το 2000 η HTML έγινε επίσης παγκόσμιο πρότυπο (ISO/IEC 15445:2000). Η τελευταία προδιαγραφή της HTML, η HTML 4.01 δημοσιεύτηκε από το W3C το 1999, και το 2001 δημοσιεύτηκαν επίσης και τα λάθη και οι παραλείψεις της (errata).

3.1.1.3 Εκδόσεις της XHTML

Η XHTML είναι ξεχωριστή γλώσσα η οποία ως αναδιαμόρφωση της HTML 4.01 με χρήση της XML 1.0 και η οποία συνεχίζει να αναπτύσσεται.

Η XHTML 1.0, δημοσιεύτηκε στις 26 Ιανουαρίου 2000, ως Σύσταση του W3C, μετά αναθεωρήθηκε και επανεκδόθηκε στις 1 Αυγούστου 2002. Προσφέρει τις ίδιες τρεις εκδοχές όπως η HTML 4.0 και 4.01, αναδιαμορφωμένες ως XML, με μικρούς περιορισμούς.

HTML

```
<html>
<title>Course Roster</title>
<body>
<center>
  <h1>Course Roster</h1>
  <h2>XML Programming</h2>
  <h3>Department: EECS</h3>
  <p>
  <table border=2>
    <tr>
      <th>Teacher</th>
      <td>Paul Thompson</td>
    </tr><tr>
      <th>Student<br>List</th>
      <td>Ron Jones<br>
          Uma Abingdon<br>
          Lindsay Garmon
      </td>
    </tr>
  </table>
</center>
</body>
</html>
```

XML

```
<?xml version="1.0"?>
<course>
  <name>Java Programming</name>
  <department>EECS</department>
  <teacher>
    <name>Paul Thompson</name>
  </teacher>
  <student>
    <name>Ron Jones</name>
  </student>
  <student>
    <name>Uma Abingdon</name>
  </student>
  <student>
    <name>Lindsay Garmon</name>
  </student>
</course>
```

Εικόνα 3-2 HTML versus XML.

Η XHTML 1.1, δημοσιεύτηκε στις 31 Μαΐου 2001, ως Σύσταση του W3C. Βασίζεται στην XHTML 1.0 Strict, αλλά περιέχει μικρές αλλαγές, μπορεί να παραμετροποιηθεί, μπορεί να αναμορφωθεί χρησιμοποιώντας αρθρώματα της XHTML, τα οποία δημοσιεύτηκαν στις 10 Απριλίου 2001, ως Σύσταση του W3C.

Για την XHTML 2.0 δεν υπάρχει πρότυπο, και αυτή τη στιγμή είναι ένα πρόχειρο έγγραφο και θεωρείται ακόμα έργο σε εξέλιξη. Η XHTML 2.0 δεν είναι συμβατή με την

XHTML 1.x και επομένως μπορεί πιο σωστά να χαρακτηριστεί ως μια νέα γλώσσα που είναι εμπνευσμένη από την XHTML παρά ως αναβάθμιση της υπάρχουσας XHTML 1.x.

Τον Οκτώβριο του 2009 το W3C σταμάτησε την εξέλιξη της XHTML με σκοπό να επικεντρωθεί στην ανάπτυξη της HTML5.

3.1.1.4 Σχετικά με τα πρότυπα

Οι εκδόσεις παίζουν καθοριστικό ρόλο καθώς με το πέρας των εκδόσεων γίνεται και η υιοθέτηση διαφόρων προτοτύπων ανάλογα με τις απαιτησεις που πρέπει να καλυφθούν.

3.1.1.4.1 Διαδικασία υιοθέτησης προτύπων

Η έννοια του προτύπου αφορά μία τεχνολογία η οποία πρέπει να προσαρτηθεί στην πράξη και να καθορίσει όλες τις λεπτομέρειες τόσο συμβατότητας με τις υπάρχουσες τεχνολογίες όσο και της λειτουργικότητας που προσφέρει η ίδια. Σε περίπτωση υιοθέτησης ενός προτύπου τότε έχουμε την εγγύηση συνεργασίας προιόντων από διαφορετικές εταιρίες.

Το Internet Engineering Task Force (IETF) είναι ο φορέας που είναι υπεύθυνος για την δημιουργία προτοτύπων σχετικα με τις τεχνολογίες που έχουν σχέση με το internet. Η δημιουργία ενός προχείρου (working draft) ξεκινάει όταν ξεκινήσει η συζήτηση για μία νέα έκδοση μίας τεχνολογίας και έτσι ανοίγεται ένα νέο πρόχειρο που αναφέρεται σε αυτή. Έγγραφα RFC (Request for Comments) καταθέτονται από εταιρείες, από ειδικούς που ασχολούνται με τον τομέα, καθώς και από άλλους σχετικούς φορείς, αυτά περιέχουν προτάσεις σχετικά με την τεχνολογία που τίθεται υπό συζήτηση. Κάθε έγγραφο RFC έχει έναν μοναδικό αριθμό σαν πρωτόκολο, όπως για παράδειγμα η έκδοση 2.0 της HTML είναι το έγγραφο RFC 1866. Τα έγγραφα RFC αυτά δεν καταλήγουν όλα να γίνονται πρότυπα, αφού κάποια από αυτά απορρίπτονται ή απορροφούνται από άλλα έγγραφα.

Στην συνέχεια, πραγματοποιείται μια συζήτηση σχετική με το έγγραφο που έχει κατατεθει για το περιεχομενο του και πιθανές βελτιώσεις. Ανάλογα με την τεχνολογία που περιέχεται οι φορείς που συμμετέχουν κάθε φορά στη συζήτηση διαφέρουν. Συνήθως για την νέα έκδοση της HTML αναλαμβάνει την συζήτηση το World Wide Web Consortium (W3C). Οποιοσδήποτε μπορεί να καταθέσει μία πρόταση ή να σχολιάσει μία ήδη υπάρχουσα, ενώ όταν οι συμμετέχοντες στη συζήτηση συμφωνήσουν, η πρόταση συνήθως εγκρίνεται ως πρότυπο. Όσο αφορά την έκδοση 5 της γλώσσας αυτής υπήρξε συνεργασία και με το Web Hypertext Application Technology Working Group (WHATWG), το οποίο είναι μία άλλη επιτροπή που ιδρύθηκε από στελέχη εταιρειών που ασχολούνται με την γλώσσα.

3.1.1.5 HTML5

3.1.1.5.1 Η «ιστορία» και η τυποποίηση της HTML5

Η HTML5 είναι μια υπό ανάπτυξη γλώσσα σήμανσης για τον Παγκόσμιο Ιστό που όταν ετοιμαστεί θα είναι η επόμενη μεγάλη έκδοση της HTML (Γλώσσα Υπερκειμένου, HyperText Markup Language). Με το όνομα Web Applications 1.0, η ομάδα Web Hypertext Application Technology Working Group (WHATWG) άρχισε δουλειά σε αυτή την έκδοση τον Ιούνιο του 2004. Σε κατάσταση "Last Call" στο WHATWG, ηταν ακόμη το πρότυπο, το Φεβρουάριο του 2010.

Η HTML5 πρόκειται να αντικατάστασει της HTML 4.01, της XHTML 1.0, και της DOM Level 2 HTML. Η μείωση της ανάγκης για ιδιόκτητα plug-in και πλούσιες διαδικτυακές εφαρμογές (RIA) όπως το Microsoft Silverlight, το Adobe Flash, το Apache Pivot και η Sun JavaFX είναι ο βασικός σκοπός. Η HTML5 εμπεριέχει το πρότυπο Web Forms 2.0 που είναι επίσης της WHATWG. Οι ιδέες πίσω από την HTML5 αρχισαν να εμφανίζονται αρχικά το 2004 από την ομάδα WHATWG.

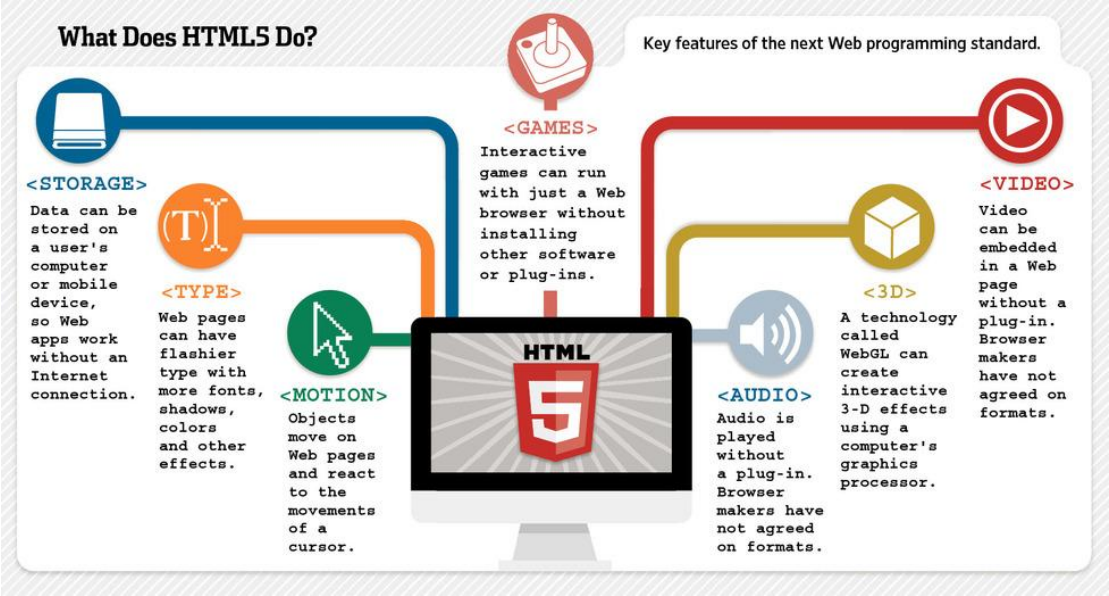
Το πρότυπο HTML5 υιοθετήθηκε ως αρχικό βήμα για τις εργασίες της νέας ομάδας εργασίας HTML του W3C το 2007. Αυτή η ομάδα εργασίας δημοσίευσε το Πρώτο Δημόσιο Working Draft του προτύπου στις 22 Ιανουαρίου 2008. Το πρότυπο είναι ακόμη υπό ανάπτυξη, μέρη της HTML5 θα τελειώσουν και θα υποστηριχτούν από περιηγητές πριν το όλο πρότυπο φτάσει στη τελική κατάσταση Recommendation και αναμένεται να παραμείνει έτσι για πολλά χρόνια.

3.1.1.5.2 Εύρος του προτύπου και δύναμη της HTML5

Η HTML5 περιλαμβάνει πολλά τμήματα με τα οποία θα μπορούν να δημιουργηθούν ανεξάρτητες και αρκετά πολύπλοκες εφαρμογές που απαιτουν οι συνθήκες εξέλιξης της παρούσας εποχής σε αντίθεση με τις προηγούμενες εκδόσεις της HTML οι οποίες ως τσόχο είχαν την δημιουργία μίας γλώσσας κατάλληλης για την περιγραφή της εμφάνισης των σελίδων και την βασική αλληλεπίδρασή τους με τον χρήστη.

Η HTML5, εξαιτίας του εύρους των θεμάτων που μπορεί να καλύψουν οι προτάσεις που αναπτύσσονται μπορεί να χαρακτηριστεί και σαν μία πλήρης γλώσσα προγραμματισμού. Η δύναμη της γλώσσας πλέον HTML, οφείλεται στο ότι παρέχει πλέον δυνατότητες που θα μπορεί να αντικαταστήσει ένα πλήθος τρίτων τεχνολογιών που σήμερα χρησιμοποιούνται στις υπάρχουσες ιστοσελίδες χωρίς να υπάρξει απώλεια δυνατότητων της ιστοσελίδας είτε τεχνολογίας. Επιπλέον πολλές από τις αλλαγές που έχουν προταθεί έχουν σαν βασικό μέλημα να διορθώσουν κενά της γλώσσας και άμεσες ελλείψεις.

Οι αλλαγές προφανώς επιρρεάζουν και προσθέτουν πολλά καινούργια στοιχεία και στη Javascript και στο CSS εκτος από την γλώσσα HTML. Το πρότυπο διατυπώνεται με μεγάλη ακρίβεια και λεπτομέρια ώστε να είναι ξεκάθαρος ο τρόπος υλοποίησής τους και να μην επαναληφθούν προβλήματα που παρουσιάστηκαν στο παρελθόν.



Εικόνα 3-3 Οι τεχνολογίες μέσω δυνατοτήτων που περιλαμβάνονται στο πρότυπο HTML5.

3.1.1.6 Περιορισμοί της HTML

Η τυποποίηση της HTML αναγκαστηκε να γίνεται παράλληλα με την χρήση της και την υλοποίηση της είτε μέχρι τελικής μορφής είτε τμημάτων τμημάτων της καθώς εάν ακολουθουνταν το τυπικό πρωτόκολλο ιυοθέτησης προτπυπων θα έπρεπε να ολοκληρώθει το 2020 όμως είναι πράγμα αναπόφευχθο. Οι εταιρείες και οι οργανισμοί αναλαμβ'ανουν την ευθύνη παράλληλης στήριξης και χρησης και υλοποίησης καθώς φυσικά χρειάζεται να γίνει και κάλυψη δικών τους αναγκών.

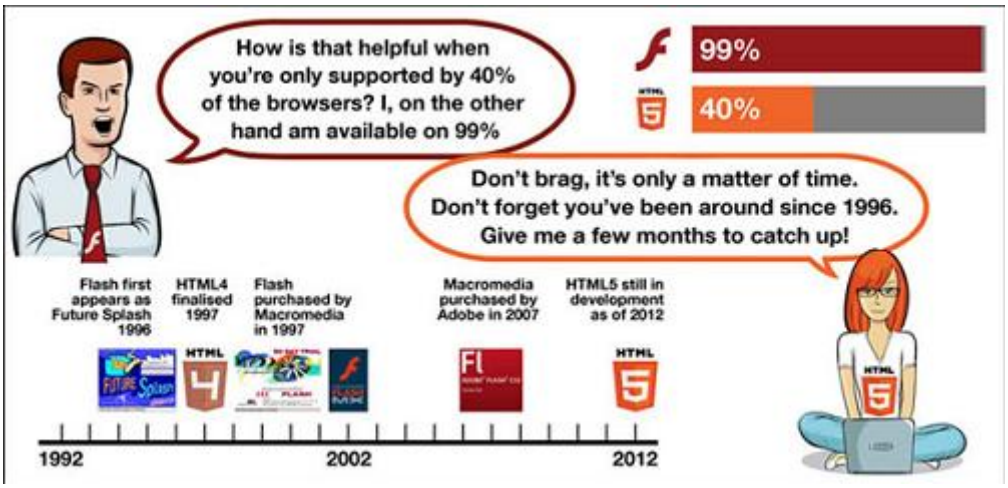
Όπως παραδείγματος χάριν συμβαίνει με τους browsers όπου κάθε διαφορετική έκδοση του ίδιου browser μπορεί να υποστηρίζει διαφορετικά στοιχεία σε διαφορετικό βαθμό ή να τα υλοποιεί με διαφορετικό τρόπο. Οι περισσότεροι σύγχρονοι browser αναβαθμίζονται συνεχώς, υιοθετώντας όλο και περισσότερα χαρακτηριστικά με κάθε νέα έκδοση.

Τα προβλήματα που υπάρχουν σήμερα για την κατασκευή σελίδων με τις προηγούμενες εκδόσεις της HTML χωρίζονται σε δύο βασικές κατηγορίες. Οι κατηγορίες αυτές ειναι η πληθώρα των τρίτων τεχνολογιών και η δεύτερη οι αποκλίσεις από το πρότυπο της.

Η αιτία δημιουργίας και ύπαρξης της πρώτης είναι η έλλειψη χαρακτηριστικών από την HTML, όπως για παράδειγμα η έλλειψη την δυσκολία εμφάνισης αρκετά πολύπλοκων γραφικών ή animations.

Στην προσπάθεια εξάλληψης αυτού του κενού που υπήρχε δημιουργήθηκε μια πληθώρα τρίτων τεχνολογιών με σκοπό την ενσωμάτωσή τους σε μία σελίδα. Η πιο γνωστή τέτοια τεχνολογία είναι το Flash της εταιρείας Adobe, το οποίο χρησιμοποιείται κατά κύριο

λόγο μεταξύ άλλων για την δημιουργία animations, παιχνιδιών και την προβολή βίντεο. Εκτός από το flash, η Silverlight από την εταιρεία Microsoft, το JavaFX που είναι βασισμένο στη γλώσσα Java έχουν επιπροσθεθεί σε σελίδες μαζι και άλλες μικρότερες εφαρμογές.



Εικόνα 3-4 Θέμα χρόνου για την HTML η τεχνολογία Adobe Flash.

Σε όλους τους χρήστες είναι γνωστό ότι για την εφικτή λειτουργία των τεχνολογιών αυτών είναι απαραίτητο να υπάρχει εγκαταστημένο πρόγραμμα που να ρυθμίζει να υποστηρίζει και ανα αναγνωρίζει την τεχνολογία αυτή εκτός από το προγραμμα περιήγησης που πρέπει να είναι συνεργάσιμο με την τεχνολογία αυτη.

Αυτό είναι προβληματικό από πολλές απόψεις καθώς αυτά τα προγράμματα αναπτύσσονται από τρίτες εταιρείες και δεν προσφέρουν εγγυήσεις σχετικά με την διαθεσιμότητα τους σε όλες τις πλατφόρμες, όπως για παράδειγμα το Silverlight το οποίο ποτέ δεν κυκλοφόρησε για λειτουργικά συστήματα Linux. Από την άλλη οπτική και διαθέσιμα να είναι για την πλατφόρμα που χρήσιμοποιεί ο χρήστης, ο χρήστης ειναι αναγκασμένος να εγκαταστήσει το προγραμμα και σε περίπτωση όπου δεν κατεχει τις απαραίτητες γνώσεις πάλι καταλληγουμε σοτο ίδιο παρονομαστή. Επιπλέον, οι τεχνολογίες αυτές μπορει να δημιουργούν προβληματα τόσο κενών ασφάλειας όσο και απαιτήσεις σε επεξεργαστική ισχύ και μνήμη και είδικά όταν προκειται για φορητές συσκευές αυτό προκαλει μεγάλη κατανάλωση ενέργειας, όπως για παράδειγμα το πρόγραμμα Flash Player που είναι υπεύθυνο για την αναπαραγωγή αντικειμένων τύπου Flash.

Σε περίπτωση που το δούμε από την οπτική γωνία του προγραμματιστή που είναι υπεύθυνος για την κατασκευή ιστοσελιδών που εμπεριέχουν γραφικά και animation είτε videos και ο χρήστης που θα επισκευθεί την ιστοσελίδα που δημιουργησε να μην μπορεί να δει το περιεχόμενο της ,αφού δεν υπάρχει κάποια εγγύηση πως θα έχει εγκαταστημένο το αντίστοιχο πρόσθετο, αυτόματα περιορίζει το κοινό της σελίδας του ή δεν χρησιμοποιεί την συγκεκριμένη τεχνολογία, άρα πάλι στην αβεβαιότητα της στιγμής.

Όσο αφορά τηνν δευτερη και σημαντική κατηγορία είναι απο την στιγμή που δεν υπάρχει σταθεροποίηση προτύπου έχουμε και αποκλισεις υλοποιήσης του αναλογα με το browser και έτσι πολλές σελίδες υποστηρίζονται με διαφορετικό τρόπο και εμφανίζονται ανάλογα με τον browser και όχι με την υλοποίηση της ιστοσελίδας που προήλθε από τον προγραμματιστή είτε λόγο διαφορετικής ερμηνείας των προδιαγραφών είτε λόγω εσωτερικών προβλημάτων (bugs).

Αυτό έχει σαν αποτέλεσμα ο κώδικας που λειτουργεί σωστά σε ένα πρόγραμμα περιήγησης να μην εμφανίζεται σε κάποιο άλλο. ΄Ετσι ο προγραμματιστής πρέπει να κάνει δοκιμές σε όλους τους πιθανούς browsers τον κώδικα μίας σελίδας και να παρεμβαίνει όπου δεν λειτουργουν σωστά. Κάτι τέτοιο όμως αυξάνει κατακόρυφα το κόστος και τον χρόνο ανάπτυξης. Σαν αποτέλεσμα πολλές σελίδες γράφονται στοχεύοντας μόνο τις πιο πρόσφατες εκδόσεις των browsers και δεν λειτουργούν σωστά στις παλιότερες, και συνήθως παροτρύνουν τον χρήστη να αναβαθμίσει το πρόγραμμα περιήγησης, πράγμα το οποιο για άλλη μιά φορά καταλίγουμε στο ότι ο χρήστης πρέπει κάτι να κάνει που ίσως και να αποτελεί μονο στο άκουσμα του κάτι σαν σπαζοκεφαλία.

Δυστηχώς είναι λάθη τα οποία επαναλαμβάνονται καθώς χαρακτηριστικό παράδειγμα κακής χρήσης των προτύπων παρουσιάστηκε κατά την έκδοση της HTML 3.0, η οποία ποτέ δεν επισημοποιήθηκε και τότε οι μεγάλοι κατασκευαστές browsers της εποχής ήταν οι εταιρείες Microsoft και Netscape, οι οποίες υιοθέτησαν διαφορετικά τμήματα της προδιαγραφής με αποτέλεσμα αρκετά πράγματα που λειτουργούσαν στα προϊόντα της μίας να μην λειτουργούν στα προϊόντα της άλλης. Προς αποφυγήν κάτι τέτοιου κατά την διαδικασία ανάπτυξης της HTML5 δίνεται αρκετά μεγάλο βάρος στην ακριβή διατύπωση της προδιαγραφής και στην παροχή διευκρινήσεων για όλες τις λεπτομέρειες που μπορεί να παρερμηνευτούν σε περίπτωση επέκτασης προτοτύπου και βαρύτητας γνώσεων.

3.1.1.7 Λειτουργία

Συνήθως οι ετικέτες HTML λειτουργούν ανά ζεύγη όπως για παράδειγμα **<h1>**Επικεφαλίδα1**</h1>**, με την πρώτη ετικέτα να ονομάζεται ετικέτα έναρξης και τη δεύτερη ετικέτα λήξης. Ανάμεσα στις ετικέτες τοποθετούνται αρχεία όπως κείμενα, βίντεο, εικόνες κτλ όπου ο browser ερμηνεύει το περιεχόμενο, χωρίς όμως να εμφανίζει τις ετικέτες HTML. Οι σελίδες που γράφονται σε αυτή τη μορφή είναι κείμενα και μπορούν να διαβαστούν από οποιονδήποτε χρησιμοποιεί απλό κείμενο. Δεν επηρεάζονται οι εντολές από το αν έχουν γραφτεί με κεφαλαία ή πεζά γράμματα. Τα αρχεία αυτά αποθηκεύονται ως *.html ή *.htm**.**

3.1.1.8 Μερικές συντάξεις και εντολές των στοιχείων της HTML

Content Tag							
a	abbr	acronym	address	applet	area	base	blockquote
body	br	button	caption	col	colgroup	del	dfn
dd	dir	div	dl	dt	embed	fieldset	frame
frameset	h1 through h6	head	html	hr	iframe	ilayer	img
input	ins	isindex	label	layer	legend	li	link
map	marquee	menu	meta	multicol	noembed	noframes	noscript
object	ol	optgroup	option	p	param	pre	samp
select	spacer	span	style	table	tbody	td	textarea
tfoot	th	thead	title	tr	ul	xmp	wbr

Εικόνα 3-5 Ετικέτες περιεχομένου

- Περιγραφή της ιστοσελίδας:

```
<html>

…

</html>
```

- Κεφαλίδα εγγράφου HTML:

```
<head>
…
</head>
```

- Καθορισμός του τίτλου της ιστοσελίδας, ο οποίος εμφανίζεται στο πάνω μέρος του browser:

```
<title>
The title
</title>
```

- Κύριο μέρος του αρχείου:

```
<body>
…
</body>
```

- Επικεφαλίδες:

```
<h1>
Επικεφαλίδα1
```

```
</h1>
<h2>
Επικεφαλίδα2
</h2>
<h3>
Επικεφαλίδα3
</h3>
```

- Πίνακες δεδομένων:
  <table>

  …

  Οι γραμμές του πίνακα καθορίζονται από την ετικέτα:

  <tr>

  …

  </tr>

  Οι στήλες του πίνακα καθορίζονται από την ετικέτα:

  <th>

  …

  Τα δεδομένα στηλών καθορίζονται από την ετικέτα:

  <td>

  …

  </td>

  </th>

  </table>

- Πλαίσια (frames):

  <frame>
  …

  </frame>

- Μη – αριθμημένη λίστα:

   Πρώτο στοιχείο

   Δεύτερο στοιχείο

```
</ul>
```

- Αριθμημένη λίστα:

```
<ol>
    <li> Πρώτο στοιχείο

    <li> Δεύτερο στοιχείο

</ol>
```

- Φόρμα:

```
<form>

…

        Σε περίπτωση εισαγωγής κειμένου:
        <input name=".." ...>

        Σε περίπτωση αποστολής δεδομένων συμπλήρωσης φόρμας μέσω κομπιού:
        <input type="submit" value="....">
</form>
```

Συνοψίζοντας τα πλεονεκτήματα της HTML :

- Εύκολο στη χρήση.
- Υποστηρίζεται από κάθε πρόγραμμα περιήγησης.
- Είναι δωρεάν. Δε χρειάζεται η αγορά κάποιου λογισμικού.
- Είναι εύκολο στη μάθηση και στη δημιουργία κώδικα.
- Χρησιμοποιείται ευρέως.

3.1.2 CSS

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

Η CSS έχει σχεδιαστεί για να επιτρέπει κυρίως τον διαχωρισμό του περιεχομένου ενός εγγράφου από την παρουσίαση του εγγράφου και για να προσφέρει περισσότερες δυνατότητες για μορφοποίηση από την HTML, συμπεριλαμβανομένων στοιχείων, όπως η διάταξη, τα χρώματα και τις γραμματοσειρές. Η πιο κοινή της εφαρμογή είναι σε

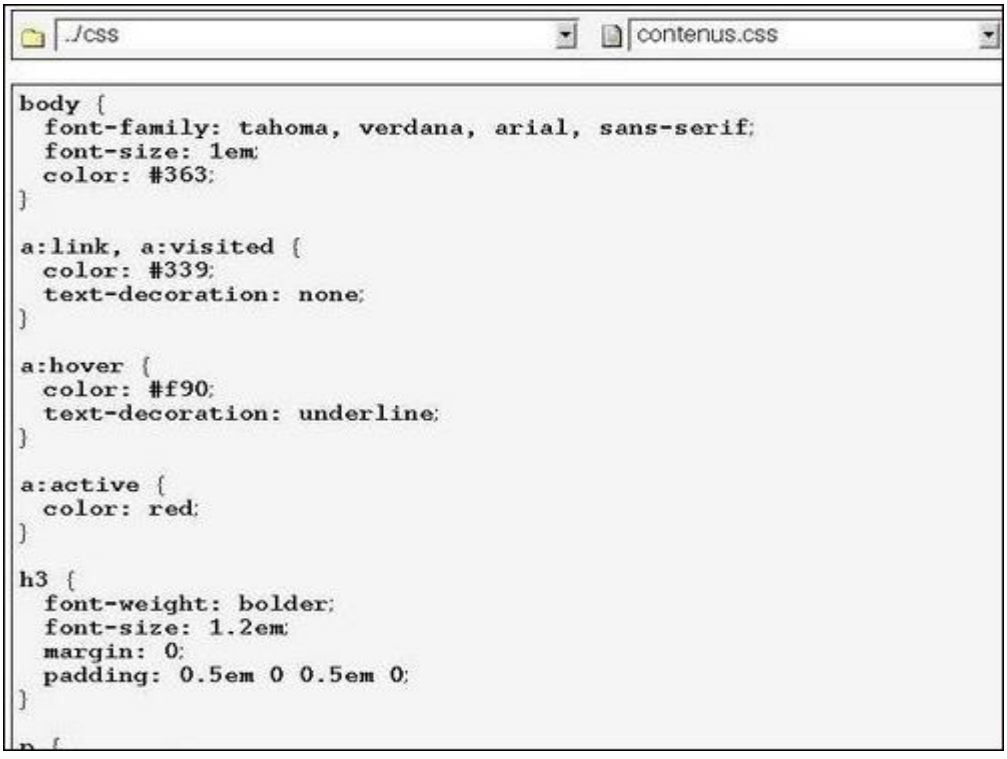
ιστοσελίδες γραμμένες σε HTML και XHTML, αλλά η CSS μπορεί επίσης να εφαρμοστεί σε οποιοδήποτε είδος εγγράφου XML, συμπεριλαμβανομένου του απλού XML, SVG και XUL.

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

Ένα φύλλο στυλ αποτελείται από μια λίστα κανόνων. Κάθε κανόνας ή σετ κανόνων αποτελείται από έναν ή περισσότερους επιλογείς και ένα μπλοκ δήλωσης. Οι επιλογείς χρησιμοποιούνται για να δηλώσουν σε ποιο μέρος της σήμανσης εφαρμόζεται το στυλ. Το τμήμα δήλωσης αποτελείται από μια λίστα δηλώσεων οι οποίες περικλείονται σε αγκύλες. Κάθε δήλωση αποτελείται από μια ιδιότητα, άνω και κάτω τελεία (:) και μια τιμή. Εάν υπάρχουν πολλές δηλώσεις σε ένα μπλοκ τότε εισάγεται ένα ερωτηματικό για το διαχωρισμό κάθε δήλωσης.

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

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



Εικόνα 3-6 CSS stylesheet

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

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

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

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

Δυστήχως εξαιτίας ότι δεν έχουν ακόμα οριστεί απο το πρότυπο εκφράσεις δεν μπορουν να υποστηριχθουν και απουσιάζουν εντελώς ακόμα και απλές εκφράσεις όπως το περιθώριο κέρδους. Ενέργιες που είχαν γίνει σχετικά με αυτό σταματησαν να υποστηρίζονται γιατί πρέπει να υπάρχει συνάφεια των προτοτύπων με τους browsers ώστε να μην υπάρχουν προβλήματα επίδοσης και ασφάλειας.

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

Δεν μπορεί να δηλώθει καινουργιο πεδίο, ανεξάρτητα από τη θέση του.

Δεν γίνεται υποστήριξη και έλεγχος ψευδο- δυναμικής συμπεριφοράς. Η CSS υλοποιεί ψευδο-κλάσεις που επιτρέπουν ως ένα βαθμό τα σχόλια των χρηστών με όρους εφαρμογής εναλλακτικών μορφών. Όπως για παράδειγμα, μια CSS ψευδο-class, «: hover», είναι δυναμική (ισοδύναμη με το «onmouseover» της JavaScript) και έχει τη δυνατότητα κατάχρησης- υπερκάλυψης (π.χ., στην εφαρμογή αναδυόμενα παράθυρα), αλλά η CSS δεν έχει καμία δυνατότητα για έναν πελάτη για να το απενεργοποιήσει. Δεν υπάρχει δυνατότητα απενεργοποίησης ή να περιορίσει τα αποτελέσματά της.

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

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

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

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

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

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

3.1.3 JavaScript

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

Καθώς οι σελίδες HTML παρουσιάζουν περιεχόμενο στον χρήστη, αρκετές φορές αυτό αρκεί. Όμως υπάρχουν περιπτώσεις που θέλουμε να κάνουμε μία σελίδα να αντιδράει στις επιλογές του χρήστη με διάφορους τρόπους. Κάτι τέτοιο δεν είναι απαραίτητο, όμως μπορεί να κάνει την σελίδα πολύ πιο εύκολη στη χρήση άλλες πάλι φορές είναι απαραίτητο καθως πρέπει να γίνουν και ενέργειες που αφορουν και λειτουργικά κομμάτια όπως συνδεση σε ένα server . Ο ρόλος της γλώσσας Javascript εμπείπτει ακριβώς εκεί.

Η JavaScript είναι μια δυναμική γλώσσα προγραμματισμού των ηλεκτρονικών υπολογιστών. Χρησιμοποιείται πιο συχνά ως μέρος των web browsers, των οποίων οι υλοποιήσεις επιτρέπουν client-side scripts για να αλληλεπιδράσει με το χρήστη, ελέγχουν το πρόγραμμα περιήγησης, να επικοινωνούν ασύγχρονα, και μεταβάλλει το περιεχόμενο

του εγγράφου που εμφανίζεται. Επίσης, χρησιμοποιείται για τον προγραμματισμό του δικτύου server-side με πλαίσια όπως Node.js, την ανάπτυξη παιχνιδιών και τη δημιουργία των desktop και mobile εφαρμογών.

Στην κατανόηση της γλώσσας αυτής θα εξηγήσουμε τον σημαντικό ρόλο που παιζει μέσω ενός παραδείγματος που σύγουρα όλοι μας έχουμε εντιμετωπίσει ως χρήστες του διαδικτύου. Το παταδειγμα μας είναι η ορθή συμπλήρωση φόρμας και αυτόματος έλεγχος των πεδίων συμπληρωσης σε κάποια ιστοσελίδα. Ας το κοιτασουμε αναλυτικά.

Ένα τυπικό σενάριο είναι μία φόρμα εγγραφής νέου χρήστη. Σε αυτή ο χρήστης συμπληρώνει τα στοιχεία του και υποβάλει την φόρμα, δηλαδή στέλνει αυτά τα στοιχεία στον server. Πριν γίνει η αποθήκευση των στοιχείων του νέου χρήστη χρειάζεται να γίνει έλεγχος των στοιχείων που στάλθηκαν, για παράδειγμα αν έχουν δοθεί έγκυρα στοιχεία και αν έχουν συμπληρωθεί όλα τα υποχρεωτικά πεδία, πχ σε ένα πεδίο που ζητάει το email του χρήστη έαν είναι σωστά γραμμένο. Αν τα στοιχεία δεν είναι έγκυρα τότε δεν μπορεί να γίνει αποθήκευση, οπότε είναι απαραίτητο να εμφανιστεί πάλι η φόρμα και να εμφανιστεί το κατάλληλο μήνυμα που να ειδοποιεί τον χρήστη για τα λάθη. Παρόλο που η παραπάνω προσέγγιση λειτουργεί, είναι κουραστική και δυσνόητη για τον χρήστη. Αφού αυτός συμπληρώνει την φόρμα εγγραφής περιμένει να δει το αποτέλεσμά της ενέργειάς του. Το να εμφανίζεται πάλι η ίδια φόρμα μπορεί να τον αποπροσανατολίσει, ειδικά αν δεν έχει εμπειρία, ενώ χρειάζεται να καταβάλει προσπάθεια για καταλάβει τι λάθος έκανε και ότι πρέπει να την συμπληρώσει ξανά.

Στο παραπάνω παράδειγμα η φόρμα θα μπορούσε να ελέγχεται με JavaScript πριν την υποβολή της. Με αυτόν τον τρόπο τα λάθη θα εμφανίζονταν μόλις ο χρήστης πατούσε το κουμπί υποβολής, χωρίς να ξαναφορτωθεί η σελίδα. Με αυτόν τον τρόπο μπορεί να καταλάβει αμέσως τι έχει συμβεί, αφού το λάθος εμφανίστηκε σαν αντίδραση στο πάτημα του κουμπιού, κάνοντας την εφαρμογή πιο φιλική, είτε θα μπορούσε με το που συμπληρώμε το κάθε κουτάκι να το τσεκάρι και να μην το κοκκινίζει όπως έχουμε δει σε πολλες σελίδες.

Εικόνα 3-7 Ο βοηθητικός ρόλος της Javascript

Δυστήχς θα πρέπει να παραθέσουμε κάποιους περιορισμους που μας ορίζει η γλώσσα αυτή που δυστηχώς μας προκαλει την εντύπωση και οχι μόνο, μιας γλωσσας περίπλοκης και δύσκολης συγκριτικα με τις συμβατικές γλωσσες που γνωρίζουμε αν και παρέχει παντοδύναμα πράγματα στην τροποποίηση περιεχομένου μιας σελίδας.

Ο πρώτος είναι πως για την εκτέλεση πολύπλοκων λειτουργιών ο κώδικας JavaScript που απαιτείται είναι αρκετά πιο μπερδεμένος σε σχέση με κάποιες παραδοσιακές γλώσσες προγραμματισμού.

Επιπλέον υπάρχουν αρκετές μικρές διαφοροποιήσεις στον ακριβή τρόπο λειτουργίας της JavaScript ανάμεσα σε διαφορετικούς browsers, κάτι το οποίο κάνει απαραίτητο τον έλεγχο της σελίδας σε κάθε έναν από αυτούς ώστε να τσεκάρουμε την συμβατότητα υποστήριξης της.

Ο βασικότερος περιορισμός της JavaScript είναι η αδυναμία άμεσης επικοινωνίας με βάση δεδομένων. Αν και είναι δυνατή η επικοινωνία με τον server μέσω AJAX, δεν γίνεται με κανέναν τρόπο να επικοινωνήσει με την βάση δεδομένων, εκτός και αν παρεμβάλλεται μία σελίδα γραμμένη σε κάποια γλώσσα που εκτελείται στον server.

Έτσι, αυτό που θα πρέπει να θυμόμαστε είναι ότι ο ρόλος της είναι βοηθητικός, στο να προσφέρει κάποια επιπλέον λειτουργικότητα, αλλά η δημιουργία της βασικής σελίδας γίνεται στον server.

3.1.3.1 JavaScript Frameworks

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

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

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

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

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

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

Εικόνα 3-8 Javascript FrameWorks

3.1.3.2 jQuery

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

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

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

```
function loadXMLDoc() {
   var xmlhttp;
   // code for IE7+, Firefox, Chrome, Opera, Safari
   if (window.XMLHttpRequest) {
      xmlhttp=new XMLHttpRequest();
   } else { // code for IE6, IE5
      xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
   }

   xmlhttp.onreadystatechange=function() {
      if (xmlhttp.readyState==4 && xmlhttp.status==200) {
         document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
      }
   }

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

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

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

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

```
});
```

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

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

3.1.4 PHP

Η γλώσσα PHP δημιουργήθηκε το 1995 από τον Rasmus Lerdoff, με σκοπό την επεξεργασία των στοιχείων μίας φόρμας που αποστέλλεται σε έναν web server, έναν ρόλο που έχει αποκτήσει πλέον η Javascript. Πλέον έχει εξελιχθεί σε μία γλώσσα προγραμματισμού γενικής χρήσης, χρησιμοποιείται κυρίως για την δημιουργία δυναμικών ιστοσελίδων.

Πρόκειται για μία interpreted γλώσσα προγραμματισμού, το οποίο σημαίνει ότι τα αρχεία της δεν γίνονται compile, αλλά διαβάζονται κάθε φορά από τον μεταφραστή της γλώσσας (interpreter) ο οποίος εκτελεί τις εντολές που περιέχουν. Αυτό το χαρακτηριστικό σημαίνει πως είναι γενικά πιο αργή από τις κλασικές (compiled) γλώσσες προγραμματισμού, όμως γενικά χρησιμοποιείται σε περιβάλλοντα που ο επιπλέον χρόνος εκτέλεσης δεν παίζει υπολογίσιμο ρόλο στις επιδόσεις του συστήματος. Επίσης υπάρχει μία σειρά βιβλιοθηκών και εργαλείων που αυξάνουν σημαντικά την ταχύτητα εκτέλεσης, με μία από τις επιλογές να είναι η μεταγλώττιση ενός προγράμματος PHP σε C++, εκμηδενίζοντας τις καθυστερήσεις που οφείλονται στον μεταφραστή.

Αν και αρχικά η γλώσσα ήταν καθαρά διαδικαστική (procedural), δηλαδή έκανε χρήση συναρτήσεων, με το πέρασμα του χρόνου προστέθηκε η δυνατότητα χρήσης κλάσεων και αρκετά χαρακτηριστικά των αντικειμενοστραφών γλωσσών προγραμματισμού. Καθώς ένας από τους στόχους κάθε νέας έκδοσης ήταν η συμβατότητα με τον κώδικα που έχει γραφτεί για παλιότερες εκδόσεις, η χρήση των νέων στοιχείων της γλώσσας είναι καθαρά προαιρετική και δίνεται αρκετή ελευθερία στον προγραμματιστή να επιλέξει το στυλ που ταιριάζει καλύτερα στις ανάγκες του. Αυτό πολλές φορές αναφέρεται σαν ένα από τα πλεονεκτήματα της PHP, καθώς δεν επιβάλλει στον προγραμματιστή μία συγκεκριμένη δομή στον κώδικα, σε αντίθεση με άλλες παρόμοιες τεχνολογίες, (πχ Java Servlets). Αντίθετα ένα project μπορεί να είναι πλήρως αντικειμενοστραφές ή να μην έχει καν συναρτήσεις.

3.1.4.1 Εκτέλεση προγραμμάτων PHP

Ένα αρχείο με κώδικα PHP (τα οποία λέγονται και PHP scripts) δεν χρειάζεται compile όπως αναφέραμε παραπάνω και μπορεί να εκτελεστεί κατευθείαν. Η εκτέλεση μπορεί να γίνει με την χρήση του interpreter ή μέσω κάποιου web server.

Αν χρησιμοποιηθεί ο interpreter, δηλαδή ένα εκτελέσιμο αρχείο (που συνήθως λέγεται php) τότε ένα πρόγραμμα μπορεί να τρέξει από τη γραμμή εντολών όπως μία κανονική γλώσσα. Όμως αυτή η ιδιότητα προστέθηκε πολύ αργότερα και δεν χρησιμοποιείται ιδιαίτερα.

Ο πιο συνηθισμένος τρόπος εκτέλεσης ενός προγράμματος PHP είναι σε συνδυασμό με έναν web server. Συγκεκριμένα ο server λαμβάνει μία αίτηση HTML και ένα πρόκειται για ένα αρχείο PHP το προωθεί σε ένα κατάλληλο άρθρωμα(module). Συνήθως τα αρχεία που επεξεργάζονται σαν PHP είναι αυτά που έχουν κατάληξη php ή phtml αλλά μέσω ρυθμίσεων μπορούν να κατασκευαστούν κανόνες τόσο για άλλες καταλήξεις όσο και για συγκεκριμένα ονόματα ή διαδρομές αρχείων.

Εικόνα 3-8 The benefits of PHP

Η προώθηση ενός αρχείου στο module PHP σημαίνει ότι εκτελείται το κατάλληλο module στο οποίο περνιούνται σαν παράμετροι οι παράμετροι που στέλνονται με το HTML request, καθώς και κάποιες επιπλέον πληροφορίες σχετικά με το περιβάλλον στο οποίο τρέχει ο server. Αυτό εκτελεί τον κώδικα του αρχείου κάνοντας χρήση του μεταφραστή PHP,

και παράγει το αποτέλεσμα. Σαν αποτέλεσμα αναφερόμαστε στο κείμενο που τυπώνει το αρχείο, όμως αυτό μπορεί παράλληλα να εκτελεί και άλλες ενέργειες, όπως για παράδειγμα να γράφει αρχεία, να επικοινωνεί με βάσεις δεδομένων, web services, κα.

Το αποτέλεσμα επιστρέφεται στον web server ο οποίος το στέλνει σαν απάντηση στον client (browser) που το ζήτησε. Τυπικά το αποτέλεσμα είναι κείμενο και επιστρέφεται σαν HTML, όμως το είδος του μπορεί να προσδιοριστεί προγραμματιστικά κατά την εκτέλεση ενός script, ώστε να είναι μία εικόνα, ένα έγγραφο XML, Json, ή οτιδήποτε άλλο.

Το παραπάνω σενάριο, δηλαδή ο συνδυασμός ενός web server μαζί με το κατάλληλο module, είναι μία τυπική εγκατάσταση. Ανάλογα με τα εργαλεία που χρησιμοποιούνται υπάρχουν πολλοί δυνατοί συνδυασμοί, όπως πχ η μεταγλώττιση ενός προγράμματος PHP σε κανονικό εκτελέσιμο πρόγραμμα και η απευθείας εκτέλεσή του από τον web server.

Πρακτικά, για την ανάπτυξη εφαρμογών PHP, χρειάζεται η εγκατάσταση ενός web server μαζί με το module PHP, της γλώσσας PHP και οι κατάλληλες ρυθμίσεις ώστε να συνεργάζονται μεταξύ τους. Έπειτα χρειάζεται να γραφτεί ένα αρχείο PHP και να τοποθετηθεί στο document root του web server (δηλαδή στον φάκελο στον οποίο ψάχνει τα αρχεία που του ζητούνται) και με έναν web browser να ζητηθεί το αρχείο αυτό από τη διεύθυνση του τοπικού υπολογιστή (localhost ή 127.0.0.1). Εάν οι ρυθμίσεις είναι σωστές θα εκτελεστεί το κατάλληλο αρχείο και το αποτέλεσμα θα επιστραφεί στον web browser ο οποίος θα το προβάλλει.

3.1.4.2 Μεταβλητές στην PHP

Ένα ιδιαίτερο χαρακτηριστικό της PHP είναι πως οι μεταβλητές δεν δηλώνονται ούτε έχουν προκαθορισμένο τύπο, αλλά χρησιμοποιούνται άμεσα. Το όνομα κάθε μεταβλητής ξεκινάει με το σύμβολο του δολαρίου ($), ενώ ο τύπος της καθορίζεται δυναμικά. Για παράδειγμα όταν η PHP συναντήσει την δήλωση $var = 1 αρχικά ελέγχει αν υπάρχει ήδη μεταβλητή με αυτό το όνομα και αν δεν υπάρχει την δημιουργεί. Έπειτα επειδή το 1 είναι ακέραιος καθορίζει ότι ο τύπος της μεταβλητής είναι ακέραιος και αναθέτει σε αυτή την τιμή 1. Αν παρακάτω συναντήσει την δήλωση $var = ‘test’ τότε κάνει τον τύπο της μεταβλητής string και βάζει την τιμή test.

Αν και η έλλειψη δηλώσεων και τύπων κάνει πολύ πιο απλή τη συγγραφή του κώδικα είναι εξίσου εύκολο να γίνουν λάθη είτε εξαιτίας ορθογραφικών λαθών είτε λόγο των μετατροπών τύπων που γίνονται συνεχώς, με αποτέλεσμα να χρειάζεται μεγάλη προσοχή σε πολύπλοκα κομμάτια κώδικα.

Κάτι άλλο που αξίζει να σημειωθεί είναι η έλλειψη διαφορετικών scope. Όλες οι μεταβλητές ενός προγράμματος (εκτός και αν βρίσκονται μέσα σε μία συνάρτηση ή μία κλάση) είναι ορατές σε ολόκληρο το πρόγραμμα από το σημείο που χρησιμοποιούνται και μετά, άσχετα από το αν έχουν δηλωθεί μέσα σε ένα block ή όχι. Έστω για παράδειγμα το παρακάτω τμήμα κώδικα σε Java:

```
if( bool > 0){
    int i = 1;
}

      i++;
```

Αυτό, πάντα στη Java (ή και στη C και στη C++) είναι λάθος καθώς το i δηλώνεται μέσα στο block του if και μόλις αυτό κλείσει παύει να υπάρχει, οπότε δεν μπορεί να χρησιμοποιηθεί στην επόμενη εντολή.

Αντίθετα στην PHP αυτό γράφεται ως εξής:

```
if( bool > 0){
    int $i = 1;
}

      $i++;
```

Λόγω της έλλειψης διαφορετικών scope ο κώδικας είναι έγκυρος και το i γίνεται δύο.

3.1.4.3 Μορφή των αρχείων PHP

Οτιδήποτε είναι γραμμένο μέσα σε ένα αρχείο PHP τυπικά θεωρείται ουδέτερο κείμενο το οποίο δεν επεξεργάζεται αλλά επιστρέφεται αυτούσιο σαν έξοδο. Αυτό σημαίνει, ειδικά στην περίπτωση των αρχείων τύπου HTML πως εάν μετονομάσουμε την κατάληξη ενός αρχείου HTML σε PHP τότε αυτό θα εμφανιστεί ακριβώς με τον ίδιο τρόπο.

Για να δηλώσουμε πως θέλουμε να γράψουμε κώδικα PHP πρέπει να χρησιμοποιήσουμε το ειδικό tag <?php για να δείξουμε την αρχή του κώδικα και το ?> για να δείξουμε το τέλος του κώδικα. Μέσω ρυθμίσεων μπορεί να ενεργοποιηθεί η υποστήριξη και διαφορετικών tags όπως τα <? και ?> ή <% και %>, αν και κάτι τέτοιο τυπικά αποφεύγεται γιατί η υποστήριξή τους δεν είναι καθολική αλλά διαφέρει ανάλογα με τις ρυθμίσεις κάθε web server, περιορίζοντας την συμβατότητα του παραγόμενου κώδικα.

Μέσα στα tags της PHP γράφεται ο κανονικός κώδικας PHP, το βασικό συντακτικό του οποίου μοιάζει με την C, δηλαδή τα κενά και οι αλλαγές γραμμών αγνοούνται (το οποίο σημαίνει πως η στοίχιση μπορεί να γίνει με οποιονδήποτε τρόπο ενισχύει την δυνατότητα ανάπτυξης). Κάθε εντολή χρειάζεται να τελειώνει με ερωτηματικό (;), ενώ οι αγκύλες {} υποδηλώνουν ένα block. Εναλλακτικά ένα block μπορεί να ξεκινάει με άνω κάτω τελεία (:) και να τελειώνει με την εντολή endif, endfor, end while, ανάλογα με την εντολή που ανήκει.

Ένα απλό αρχείο PHP έχει την παρακάτω μορφή:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>PHP Test</title>
    <head>
    <body>
    <?php
        echo '<p>Hello World</p>';
    ?>
    </body>
</html>
```

Ο κώδικας που βρίσκεται μέσα στα PHP tags (καθώς και τα ίδια τα tags) δεν επιστρέφεται αλλά εκτελείται. Αυτό που επιστρέφετε είναι το κείμενο που παράγουν τυχών

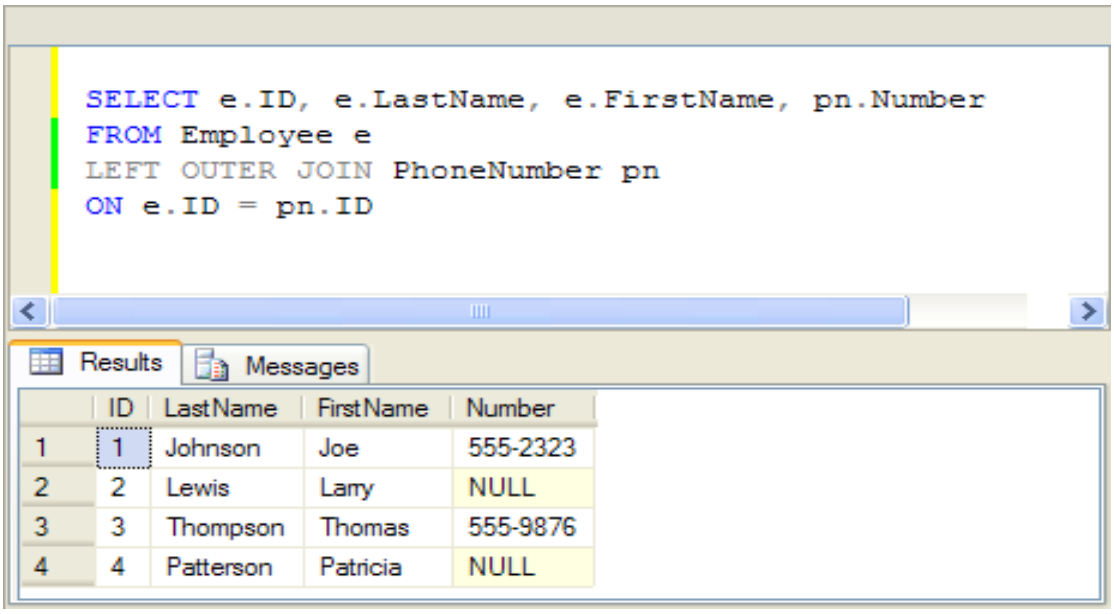
εντολές εκτύπωσης, δηλαδή εδώ η εντολή echo που τυπώνει ότι την ακολουθεί. Ο κώδικας που βλέπει ο browser είναι ο παρακάτω:

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>PHP Test</title>
    <head>
    <body>
        <p>Hello World</p></body>
</html>
```

Τα tags και η εντολή έχουν φύγει, ενώ φαίνεται μόνο η έξοδος της echo, χωρίς μετά να αλλάζει γραμμή αφού δεν έχουμε ζητήσει κάτι τέτοιο, γι' αυτό και είναι κολλημένο με το tag body. Ο browser δεν μπορεί να διαχωρίζει πιο τμήμα του κώδικα είναι στατικό και πιο παράγεται δυναμικά, αφού η απάντηση που παίρνει περιέχει μόνο HTML.

Το γεγονός πως για να γράψουμε κώδικα PHP χρειάζεται να ανοίξουμε τα κατάλληλα tags διευκολύνει πάρα πολύ την συγγραφή σελίδων HTML, καθώς σε μία τυπική σελίδα το μεγαλύτερο μέρος του αρχείου είναι tags HTML που καθορίζουν την μορφή της σελίδας, ενώ τα δυναμικά κομμάτια, δηλαδή αυτά που πρέπει να δημιουργηθούν προγραμματιστικά είναι σχετικά λίγα.

3.1.5 SQL



Εικόνα 3-9 Select Query with the answes!!!

Η SQL είναι μια γλώσσα προγραμματισμού που χρησιμοποιείται για την ανάκτηση εγγραφών ή τμημάτων εγγραφών από βάσεις δεδομένων και την εκτέλεση διαφόρων υπολογισμών πριν την εμφάνιση των αποτελεσμάτων. Η δομή της μοιάζει με φυσικής γλώσσας όμως το συντακτικό της είναι περιορισμένο και σταθερό. Η SQL είναι ιδιαίτερα κατάλληλη για αναζήτηση σε σχεσιακές βάσεις δεδομένων. (Britannica Concise

Encyclopedia). Σύμφωνα με τον ANSI (American National Standards Institute) η SQL είναι η πρότυπη γλώσσα για αυτά τα συστήματα διαχείρισης.

Οι δηλώσεις SQL χρησιμοποιούνται για την εκτέλεση εργασιών όπως η ενημέρωση των δεδομένων σε μια βάση δεδομένων ή η ανάκτηση δεδομένων από μια βάση. Μερικά κοινά συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων που χρησιμοποιούν την SQL είναι τα Oracle, Sybase, Microsoft SQL Server, Access, Ingres κ.α.

Αν και τα περισσότερα συστήματα βάσεων δεδομένων χρησιμοποιούν SQL, τα περισσότερα από αυτά έχουν επίσης δικές τους συμπληρωματικές ιδιόκτητες επεκτάσεις, που συνήθως χρησιμοποιούνται μόνο στο σύστημά τους . Οι βασικές εντολές της SQL, όπως οι "Select", "Insert", "Update", "Delete", "Create" και "Drop", μπορούν να χρησιμοποιηθούν για να πραγματοποιήσουν σχεδόν οτιδήποτε χρειάζεται να κάνει κάποιος με μια βάση δεδομένων.

Το σχεσιακό μοντέλο από το οποίο η SQL αντλεί μεγάλο μέρος του εννοιολογικού της πυρήνα, καθορίστηκε για πρώτη φορά επίσημα το 1970 από τον Dr E. F. Codd, έναν ερευνητή της IBM, σε μια δημοσίευση με τίτλο "A Relational Model of Data for Large Shared Data Banks" (« Ένα Σχεσιακό Μοντέλο Δεδομένων για Μεγάλες Κοινόχρηστες Τράπεζες Δεδομένων»). Το 1974 η IBM άρχισε το έργο System/R, και με το έργο του Donald Chamberlin και άλλων, όρισε τη SEQUEL (Structured English Query Language). Το System/R υλοποιήθηκε σε ένα πρωτότυπο της IBM που ονομάστηκε SEQUEL-XRM το 1974-75. Το System/ R ξαναγράφτηκε ολοκληρωτικά το 1976-1977 για να παρέχει multi-table και multi-user χαρακτηριστικά. Όταν το σύστημα αναθεωρήθηκε, εν συντομία μετονομάστηκε σε "SEQUEL/ 2", πριν τελικά μετονομαστεί σε "SQL" για νομικούς λόγους.

Το 1978 ξεκίνησαν μεθοδικές δοκιμές του System/R οι οποίες έδειξαν τόσο τη χρησιμότητα όσο και την πρακτικότητα του συστήματος. Ως αποτέλεσμα, η IBM άρχισε να αναπτύσσει εμπορικά προϊόντα που εφάρμοζαν την SQL βάση του πρωτοτύπου System/ R. Αρκετοί άλλοι προμηθευτές λογισμικού αποδέχθηκαν την άνοδο του σχεσιακού μοντέλου, και ανακοίνωσαν προϊόντα βασισμένα στην SQL. Αυτές οι εταιρείες περιλάμβαναν την Oracle (που νίκησε την IBM στην αγορά εντός δύο ετών με την κυκλοφορία του πρώτου εμπορικού RDBMS τους, το 1979),την Sybase και την Ingres. (Wha) (John Worsley and Joshua Drake) .

3.1.5.1 MySQL

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

Σαν βάση δεδομένων χρησιμοποιήσαμε την MySQL. Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων (Relational Database Management System - RDBMS), δηλαδή ένας server που μπορεί να περιέχει πολλές διαφορετικές μεταξύ τους βάσεις δεδομένων. Κάθε βάση έχει ένα μοναδικό όνομα και μπορεί να περιέχει πολλούς πίνακες, ενώ τα περιεχόμενα δύο διαφορετικών βάσεων δεν σχετίζονται μεταξύ τους.

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

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

Αν και η δημιουργία των πινάκων μπορεί να γίνει μέσω της PHP κάτι τέτοιο δεν έχει νόημα, γιατί πρόκειται για μία λειτουργία που θα γίνει μόνο μία φορά. Αντίθετα δημιουργήσαμε τη μορφή της βάσης με κάποιο έτοιμο εργαλείο και μέσω της εφαρμογής που γράψαμε τροποποιούμε τα δεδομένα της.

Το MySQL προέρχεται από το My, το όνομα της κόρης του ενός από τους ιδρυτές Michael 'Monty' Widenius (Μόντυ Βιντένιους), και το SQL αντιστοιχεί στο “Structured Query Language”, που σημαίνει «Δομημένη Γλώσσα Ερωτημάτων».

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

Ορισμένες εντολές της MySQL είναι για:

- **Διαχείριση χρηστών.**
  Δημιουργία:
  create user username@hostname *identified by* password;

  Διαγραφή:
  drop user username@hostname;

- **Διαχείριση βάσης δεδομένων.**
  Δημιουργία βάσης δεδομένων:
  create database όνομα_βάσης;

  Διαγραφή βάσης δεδομένων:
  drop database όνομα_βάσης;

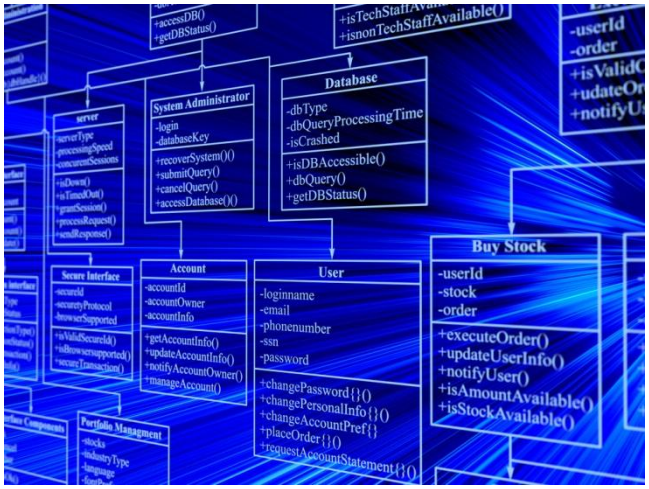
  Εμφάνιση των βάσεων που έχει δικαιώματα ο χρήστης:
  show databases;

- **Διαχείριση Δικαιωμάτων σε χρήστη.**
  Εκχώρηση δικαιωμάτων σε χρήστη:
  *grant [*privilege*] on[*db*][.*table*]to* user@host [*identified by* password] *[with grant option]*;

  Ανάκληση:
  revoke [privilege] *on* [db] [.table] [column] *from* user@host;

Συνοψίζοντας τα πλεονεκτήματα της MYSQL

- Είναι πολύ γρήγορο και δυνατό σύστημα διαχείρισης βάσεων δεδομένων.
- Μπορούν πολλές συνδέσεις με τη βάση να υπάρχουν ταυτόχρονα χωρίς να υπάρχουν πολλαπλά αντίγραφα της.
- Η απόδοση της είναι καλύτερη σε μεγαλύτερο όγκο βάσεων δεδομένων.
- Είναι οικονομική.
- Είναι λογισμικό ανοιχτού κώδικα.
- Είναι γρήγορη στην ανάκτηση δεδομένων.
- Παρέχει ευκολίες στο backup.



Εικόνα 3-10 huge Database which is supported by MYSQL!!!

Συνοψίζοντας τα μειονεκτήματα της MySQL

- Οι συναλλαγές δεν αντιμετωπίζονται αρκετά αποτελεσματικά.
- Δεν υποστηρίζει ένα μεγάλο μέγεθος της βάσης δεδομένων.
- Δεν υποστηρίζει ξένα κλειδιά.

3.2 Εργαλεία που χρησιμοποιήθηκαν για την υλοποίηση.

3.2.1 Notepad++

Το Notepad++ είναι ένας text editor ανοικτού κώδικα με προσανατολισμό για χρήση κατά των προγραμματισμό. Η κυριότερη δυνατότητα που έχει είναι η υποστήριξη συντακτικού χρωματισμού (syntax highlighting) για ένα πολύ μεγάλο πλήθος γλωσσών προγραμματισμού, καθώς και προηγμένες λειτουργίες αναζήτησης, αντικατάστασης κ.α.

Πρακτικά είναι ένα εργαλείο που λειτουργεί σαν απλός text editor αλλά παρουσιάζει τον κώδικα με ποιο κατανοητό τρόπο. Προφανώς αυτή η μορφοποίηση φαίνεται απλά στον προγραμματιστή και δεν αποθηκεύεται στα αρχεία.

4 Δομή εφαρμογής

4.1 Γενικά

Η εφαρμογή μας είναι ένα data driven web application, δηλαδή μία ιστοσελίδα η οποία αλληλοεπιδράει κατά κόρων με μία βάση δεδομένων για την παρουσίαση των περιεχομένων της.

Οι σελίδες είναι προφανώς γραμμένες σε HTML και CSS, με τη χρήση Javascript σε κάποια σημεία για την επιτέλεση κάποιων ιδιαίτερων λειτουργιών.

Ο server είναι ένας Apache συνδεδεμένος με την PHP, η οποία είναι και η γλώσσα που δημιουργεί τις σελίδες μας.

Η αποθήκευση και η ανάκτηση των δεδομένων γίνονται σε μία σχεσιακή βάση δεδομένων MySQL, με την οποία αναλαμβάνει να μιλήσει η PHP.

Η εγκατάσταση αυτών των εργαλείων είναι αρκετά δύσκολη και μπερδεμένη. Για αυτόν τον λόγο υπάρχουν έτοιμα πακέτα που εγκαθιστούν και αναλαμβάνουν την ρύθμισή τους. Εμείς προτιμήσαμε να χρησιμοποιήσουμε το WAMPServer. Η επιλογή αυτού του εργαλείου δεν έχει καμία επίπτωση στο αποτέλεσμα, και μπορεί να τρέχει σε οποιοδήποτε σύστημα υποστηρίζει αυτές τις τεχνολογίες.

4.2 Βάση δεδομένων

Η βάση δεδομένων που χρησιμοποιήσαμε είναι η MySQL. Είναι η πιο διαδεδομένη βάση δεδομένων, και χρησιμοποιείται κατά κόρων σε ιστοσελίδες και ειδικά σε εφαρμογές PHP.

Καθώς όλη η ιστοσελίδα μας περιστρέφεται γύρω από τα δεδομένα που αποθηκεύονται σε αυτή, το σημαντικότερο βήμα στην διαδικασία ανάπτυξης ήταν η δημιουργία των πινάκων που αποθηκεύουν αυτά τα δεδομένα. Παρακάτω θα παρουσιάσουμε αυτή τη διαδικασία και τα περιεχόμενα που αποθηκεύει κάθε πίνακας.

4.2.1 Πίνακας absence

Ο πίνακας absence είναι ο πίνακας στον οποίο αποθηκεύονται οι απουσίες των μαθητών. Συγκεκριμένα, κάθε γραμμή κρατάει μία συγκεκριμένη απουσία για έναν μαθητή.

Ο πίνακας περιέχει πέντε πεδία:

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**id**	int(11)			No	*None*	AUTO_INCREMENT
2	**date**	date			No	*None*	
3	**year**	int(11)			No	*None*	
4	**justified**	int(11)			No	*None*	
5	**student_id**	int(11)			No	*None*	

Οι ρόλοι τους είναι οι εξής:

- Πεδίο id: Αυτό είναι το μοναδικό αναγνωριστικό της εγγραφής.
- Πεδίο date: Αυτή είναι η ημερομηνία που έγινε η απουσία.
- Πεδίο year: Πρόκειται για την σχολική χρονιά που έγινε η απουσία. Συγκεκριμένα, για λόγους ευκολίας, κρατάμε κάθε σχολική χρονιά όχι με δύο νούμερα, πχ 2007 – 2008 αλλά μόνο με τον πρώτο χρόνο, πχ 2007. Με αυτόν τον τρόπο είναι πιο εύκολο να κατασκευάσουμε διάφορες επερωτήσεις που χρειαζόμαστε σε κάποια σημεία της εφαρμογής.
- Πεδίο justified: Αυτό το πεδίο είναι ένας Boolean που κρατάει το αν μία απουσία είναι δικαιολογημένη ή όχι.
- Πεδίο student_id: Είναι το id του μαθητή που έκανε την απουσία.

4.2.1.1 Κώδικας δημιουργίας

Ο κώδικας με τον οποίο δημιουργήθηκε αυτός ο πίνακας είναι ο εξής:

```
CREATE TABLE `absence` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `date` date NOT NULL,
  `year` int(11) NOT NULL,
  `justified` int(11) NOT NULL,
  `student_id` int(11) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=10 DEFAULT CHARSET=utf8;
```

4.2.1.2 Κώδικας εισαγωγής

Για την εισαγωγή μίας νέας εγγραφής μέσα στον πίνακα χρειάζεται η παρακάτω εντολή:

```
INSERT INTO `school`.`absence`
(`id`,
`date`,
`year`,
`justified`,
`student_id`)
```

VALUES

(<{id: }>,

<{date: }>,

<{year: }>,

<{justified: }>,

<{student_id: }>);

4.2.2 Πίνακας class

Ο πίνακας class περιέχει τις εγγραφές για ένα τμήμα, η οποία χαρακτηρίζεται από την χρονιά, την τάξη και το τμήμα.

Περιέχει τέσσερα πεδία:

1	**id**	int(11)	No	*None*	AUTO_INCREMENT
2	**grade**	int(11)	No	*None*	
3	**number**	int(11)	No	*None*	
4	**year**	int(11)	No	*None*	

Ο ρόλος τους είναι ο εξής:

- Πεδίο id: Αυτό είναι το μοναδικό αναγνωριστικό της εγγραφής.
- Πεδίο grade: Είναι η τάξη, πχ 1 για πρώτη λυκείου κοκ.
- Πεδίο number: Είναι ο αριθμός του τμήματος, ώστε να καλύψουμε τις περιπτώσεις που υπάρχουν πολλά τμήματα για την ίδια τάξη.
- Πεδίο year: Είναι η σχολική χρονιά για το τμήμα αυτό.

4.2.2.1 Κώδικας δημιουργίας

Ο κώδικας με τον οποίο δημιουργήθηκε αυτός ο πίνακας είναι ο εξής:

CREATE TABLE `class` (

`id` int(11) NOT NULL AUTO_INCREMENT,

`grade` int(11) NOT NULL,

`number` int(11) NOT NULL,

`year` int(11) NOT NULL,

```
  PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
```

4.2.2.2 Κώδικας εισαγωγής

Για την εισαγωγή μίας νέας εγγραφής μέσα στον πίνακα χρειάζεται η παρακάτω εντολή:

```
INSERT INTO `school`.`class`

(`id`,

`grade`,

`number`,

`year`)

VALUES

(<{id: }>,

<{grade: }>,

<{number: }>,

<{year: }>);
```

4.2.3 Πίνακας grade

Ο πίνακας grade περιέχει εγγραφές για τον βαθμό. Κάθε βαθμός είναι για έναν συγκεκριμένο μαθητή, σε ένα συγκεκριμένο μάθημα, σε μία συγκεκριμένη τάξη ένα συγκεκριμένο τετράμηνο.

Περιέχει έξι στήλες:

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**id**	int(11)			No	*None*	AUTO_INCREMENT
2	**student_id**	int(11)			No	*None*	
3	**lesson_id**	int(11)			No	*None*	
4	**class_id**	int(11)			No	*None*	
5	**semester**	int(11)			No	*None*	
6	**value**	int(11)			No	*None*	

Ο ρόλος τους είναι ο εξής:

- Πεδίο id: Αυτό είναι το μοναδικό αναγνωριστικό της εγγραφής.
- Πεδίο student_id: Είναι το id του μαθητή που πήρε τον βαθμό.
- Πεδίο lesson_id: Είναι το id του μαθήματος στο οποίο αναφέρεται ο βαθμός.
- Πεδίο class_id: Είναι το id της τάξης στην οποία φοιτούσε ο μαθητής όταν πήρε αυτόν τον βαθμό.
- Πεδίο semester: Είναι το τετράμηνο στο οποίο αναφέρεται ο βαθμός.
- Πεδίο value: Πρόκειται για τον ίδιο τον βαθμό.

4.2.3.1 Κώδικας δημιουργίας

Ο κώδικας με τον οποίο δημιουργήθηκε αυτός ο πίνακας είναι ο εξής:

```
CREATE TABLE `grade` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `student_id` int(11) NOT NULL,
  `lesson_id` int(11) NOT NULL,
  `class_id` int(11) NOT NULL,
  `semester` int(11) NOT NULL,
  `value` int(11) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=29 DEFAULT CHARSET=utf8;
```

4.2.3.2 Κώδικας εισαγωγής

Για την εισαγωγή μίας νέας εγγραφής μέσα στον πίνακα χρειάζεται η παρακάτω εντολή:

```
INSERT INTO `school`.`grade`
(`id`,
`student_id`,
`lesson_id`,
`class_id`,
```

`semester`,

`value`)

VALUES

(<{id: }>,

<{student_id: }>,

<{lesson_id: }>,

<{class_id: }>,

<{semester: }>,

<{value: }>);

4.2.4 Πίνακας lesson

Ο πίνακας lesson είναι ο πίνακας που κρατάει τις πληροφορίες για τα μαθήματα. Κάθε μάθημαδιδάσκεται σε μία συγκεκριμένη τάξη και είναι υποχρεωτικό ή προαιρετικό.

Πρόκειται για έναν πίνακα με τέσσερις στήλες.

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	**id**	int(11)			No	*None*	AUTO_INCREMENT
2	**name**	varchar(30)	utf8_general_ci		No	*None*	
3	**grade**	int(11)			No	*None*	
4	**type**	int(11)			No	*None*	

Ο ρόλος κάθε πεδίου είναι ο εξής:

- Πεδίο id: Αυτό είναι το μοναδικό αναγνωριστικό της εγγραφής.
- Πεδίο name: Το όνομα του μαθήματος.
- Πεδίο grade: η τάξη στην οποία διδάσκεται το μάθημα.
- Πεδίο type: Ο τύπος του μαθήματος, αν είναι υποχρεωτικό ή προαιρετικό. Είναι ακέραιος, με την τιμή 1 να σημαίνει υποχρεωτικό και την τιμή 2 να σημαίνει προαιρετικό.

4.2.4.1 Κώδικας δημιουργίας

Ο κώδικας με τον οποίο δημιουργήθηκε αυτός ο πίνακας είναι ο εξής:

```
CREATE TABLE `lesson` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(30) NOT NULL,
  `grade` int(11) NOT NULL,
  `type` int(11) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=17 DEFAULT CHARSET=utf8;
```

4.2.4.2 Κώδικας εισαγωγής

Για την εισαγωγή μίας νέας εγγραφής μέσα στον πίνακα χρειάζεται η παρακάτω εντολή:

```
INSERT INTO `school`.`lesson`
(`id`,
`name`,
`grade`,
`type`)
VALUES
(<{id: }>,
<{name: }>,
<{grade: }>,
<{type: }>);
```

4.2.5 Πίνακας student

Ο πίνακας student κρατάει τα στοιχεία κάθε μαθητή, μεταξύ άλλων αν είναι ενεργός και την τάξη στην οποία ανήκει.

Κάθε εγγραφή περιέχει εννέα πεδία:

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	id	int(11)			No	None	AUTO_INCREMENT
2	name	varchar(60)	utf8_general_ci		No	None	
3	address	text	utf8_general_ci		No	None	
4	father	varchar(60)	utf8_general_ci		No	None	
5	mother	varchar(60)	utf8_general_ci		No	None	
6	phone	varchar(14)	utf8_general_ci		No	None	
7	mobile	varchar(14)	utf8_general_ci		No	None	
8	active	int(11)			No	1	
9	class_id	int(11)			No	None	

Ο ρόλος τους είναι ο εξής:

- Πεδίο id: Αυτό είναι το μοναδικό αναγνωριστικό της εγγραφής.
- Πεδίο name: Το ονοματεπώνυμο του μαθητή.
- Πεδίο address: Η διεύθυνση του μαθητή.
- Πεδίο father: Το πατρώνυμο του μαθητή.
- Πεδίο mother: Το όνομα της μητέρας του μαθητή.
- Πεδίο phone: Το τηλέφωνο της οικίας του μαθητή.
- Πεδίο mobile: Ένα κινητό τηλέφωνο επικοινωνίας με τους γονείς του μαθητή.
- Πεδίο active: Έναν Boolean που αποθηκεύει το αν ο μαθητής είναι ενεργός ή όχι, για παράδειγμα σε περίπτωση που αποφοιτήσει ή αλλάξει σχολείο.
- Πεδίο lesson_id: Είναι το id του μαθήματος στο οποίο αναφέρεται ο βαθμός.
- Πεδίο class_id: Είναι το id της τάξης στην οποία φοιτούσε ο μαθητής όταν πήρε αυτόν τον βαθμό.

4.2.5.1 Κώδικας δημιουργίας

Ο κώδικας με τον οποίο δημιουργήθηκε αυτός ο πίνακας είναι ο εξής:

```
CREATE TABLE `student` (
 `id` int(11) NOT NULL AUTO_INCREMENT,
 `name` varchar(60) NOT NULL,
 `address` text NOT NULL,
 `father` varchar(60) NOT NULL,
 `mother` varchar(60) NOT NULL,
 `phone` varchar(14) NOT NULL,
```

```
  `mobile` varchar(14) NOT NULL,
  `active` int(11) NOT NULL DEFAULT '1',
  `class_id` int(11) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8;
```

4.2.5.2 Κώδικας εισαγωγής

Για την εισαγωγή μίας νέας εγγραφής μέσα στον πίνακα χρειάζεται η παρακάτω εντολή:

```
INSERT INTO `school`.`student`
(`id`,
`name`,
`address`,
`father`,
`mother`,
`phone`,
`mobile`,
`active`,
`class_id`)
VALUES
(<{id: }>,
<{name: }>,
<{address: }>,
<{father: }>,
<{mother: }>,
<{phone: }>,
<{mobile: }>,
```

<{active: 1}>,

<{class_id: }>);

4.3 Παρουσίαση κώδικα

4.3.1 Ορολογία

Πριν ξεκινήσουμε την αναλυτική παρουσίαση, αξίζει να σημειωθεί ότι εξαιτίας της φύσης της γλώσσας PHP χρησιμοποιούμε τους όρους αρχείο και σελίδα σαν να πρόκειται για το ίδιο πράγμα.

Η έκφραση να καλεστεί ένα αρχείο σημαίνει να πάμε σε μία συγκεκριμένη διεύθυνση στον browser, είτε πατώντας κάποιο link είτε μέσω κάποιας φόρμας.

Αν και η PHP δίνει την δυνατότητα να καλέσουμε απευθείας κάποιο αρχείο PHP, το οποίο σημαίνει ότι θα εκτελεστεί μέσω του κώδικα και όχι μέσω κλήσης από τον browser, στις περιπτώσεις που κάνουμε κάτι τέτοιο το αναφέρουμε ρητά.

4.3.2 Γενικά

Η εφαρμογή μας ακολουθεί έναν διαχωρισμό ανάμεσα στις σελίδες που παρουσιάζουν ένα αποτέλεσμα στον χρήστη και σε αυτές που κάνουν μία συγκεκριμένη ενέργεια.

Αυτό γίνεται για να μην επαναλαμβάνουμε πολλές φορές τον ίδιο κώδικα. Για παράδειγμα θα φέρουμε την διαδικασία διαγραφής ενός μαθητή.

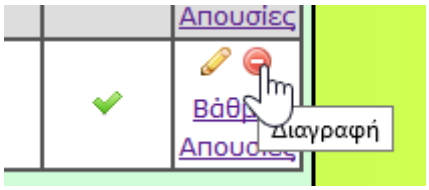
Για να μπορέσει ο χρήστης να σβήσει έναν μαθητή χρειάζεται να βλέπει ήδη τη λίστα των μαθητών, δηλαδή ένα αρχείο που να παρουσιάζει τη λίστα.

Λίστα Μαθητών

Εδώ εμφανίζονται οι καταχωρημένοι μαθητές και μπορείτε να αλλάξετε ή να σβήσετε τα στοιχεία τους.

ΑΜ	Όνομα	Τμήμα	Διεύθυνση	Όνομα πατέρα	Όνομα μητέρας	Τηλέφωνο	Κινητό	Ενεργός	
1	Ιωάννης Παπαδόπουλος	Α1-2007	Μίνωος 17, 71300, Ηράκλειο	Ιωάννης	Ιωάννα	+302810100200	+306910100200	✔	Βάθμοι Απουσίες
3	Ιωσήφ Μαύρος	Α2-2007	1821 12, 71200, Ηράκλειο	Ιωάννης	Μαρία	+302810100300	+306910100300	✔	Βάθμοι Απουσίες
4	Ιωάννης	Α1-2007	12	1	1	1	1	✔	Βάθμοι Απουσίες

Για να διαγράψει έναν μαθητή χρειάζεται να πατήσει στο κουμπί διαγραφής.



Αυτό χρειάζεται να κάνει τη διαγραφή του μαθητή και να παρουσιάσει πάλι τη λίστα με την αλλαγή που έγινε.

Η εγγραφή διαγράφηκε.

Λίστα Μαθητών

Εδώ εμφανίζονται οι καταχωρημένοι μαθητές και μπορείτε να αλλάξετε ή να σβήσετε τα στοιχεία τους.

ΑΜ	Όνομα	Τμήμα	Διεύθυνση	Όνομα πατέρα	Όνομα μητέρας	Τηλέφωνο	Κινητό	Ενεργός	
1	Ιωάννης Παπαδόπουλος	A1-2007	Μίνωος 17, 71300, Ηράκλειο	Ιωάννης	Ιωάννα	+302810100200	+306910100200	✔	Βάθμοι Απουσίες
3	Ιωσήφ Μαύρος	A2-2007	1821 12, 71200, Ηράκλειο	Ιωάννης	Μαρία	+302810100300	+306910100300	✔	Βάθμοι Απουσίες
4	Ιωάννης	A1-2007	12	1	1	1	1		

Η δεύτερη λίστα που παρουσιάζεται, με την αλλαγή της διαγραφής είναι όμοια με την πρώτη, με μοναδικές διαφορές ότι εμφανίζονται τα διαφορετικά στοιχεία και το μήνυμα της διαγραφής.

Ένας τρόπος για να γίνει αυτό είναι να υπάρχουν δύο αρχεία, ένα που παρουσιάζει τη λίστα και ένα δεύτερο, που θα καλείται όταν ο χρήστης επιλέγει να διαγράψει κάποιον μαθητή, το οποίο θα κάνει τη διαδικασία της διαγραφής και θα παρουσιάζει τη λίστα μαζί με το μήνυμα της διαγραφής. Όμως σε αυτή την περίπτωση ο κώδικας σχετικά με την παρουσίαση της λίστας θα πρέπει να μπει και στα δύο αρχεία.

Αντίθετα, στην υλοποίηση που έχουμε κάνει υπάρχει μόνο ένα αρχείο που παρουσιάζει την λίστα των μαθητών. Όταν ο χρήστης επιλέγει να διαγράψει κάποιον μαθητή τότε καλείται ένα αρχείο που περιέχει μόνο τον κώδικα της διαγραφής. Αφού εκτελέσει την διαγραφή ξαναστέλνει τον χρήστη στο (προηγούμενο) αρχείο, το οποίο αφού φορτώνει δυναμικά τα περιεχόμενα της λίστας κάθε φορά που καλείται θα δείξει την ανανεωμένη λίστα με τον μαθητή να έχει γίνει ανενεργός.

Για να λειτουργήσει ικανοποιητικά η παραπάνω διαδικασία χρειάστηκε να δημιουργήσουμε και έναν μηχανισμό μηνυμάτων, ώστε μία σελίδα που δεν έχει οπτική απεικόνιση να μπορεί να στείλει τον χρήστη σε μία σελίδα που 3

5 Οδηγίες χρήσης

5.1 Γενικά

Σε αυτή την ενότητα θα παρουσιάσουμε τις οδηγίες χρήσης της εφαρμογής. Συγκεκριμένα θα δούμε πώς μπορεί ο χρήστης να επιτελέσει τις διάφορες λειτουργίες αλλά και τις δυνατότητες που έχει στις διάφορες οθόνες της εφαρμογής.

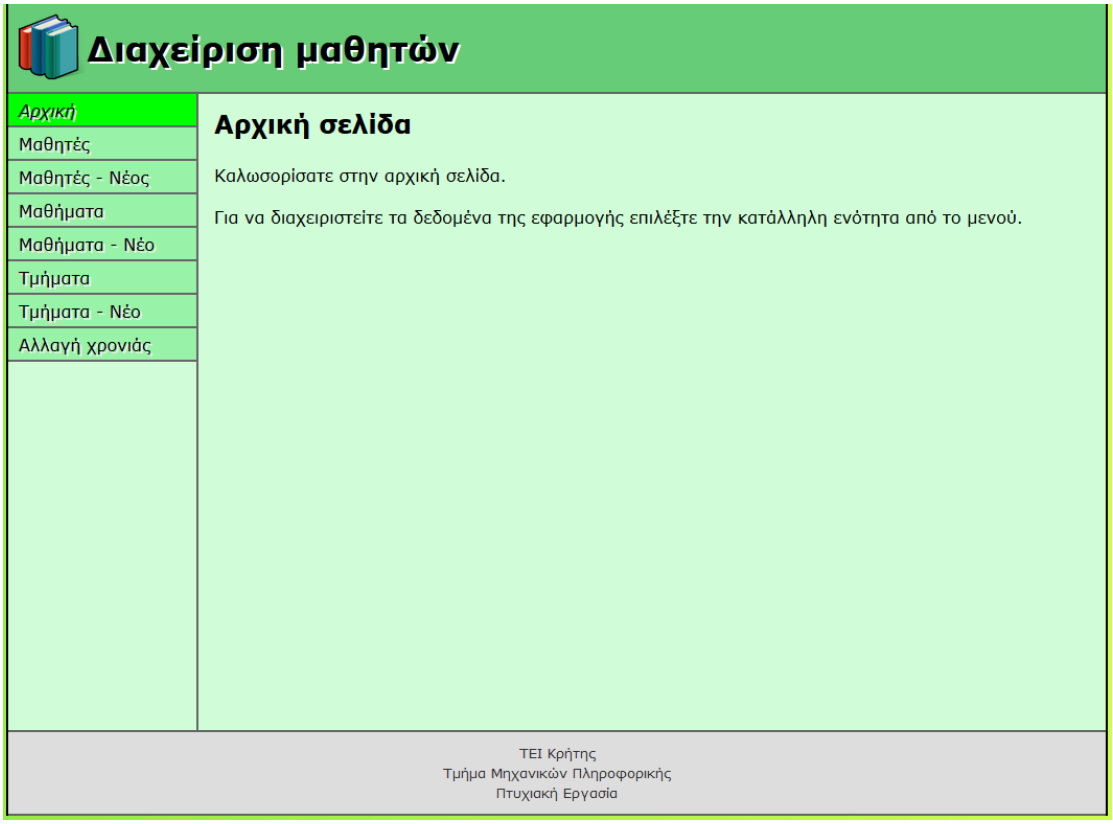
5.2 Αρχική οθόνη

Για να μπει ο χρήστης στην εφαρμογή χρειάζεται να ανοίξει μέσα από έναν web browser τη διεύθυνση:

http://localhost/school/

Προφανώς αν η εφαρμογή είναι εγκαταστημένη σε διαφορετικό υπολογιστή είναι απαραίτητο να αλλάξει το localhost με το όνομα ή τη διεύθυνση του υπολογιστή που βρίσκεται η εφαρμογή.

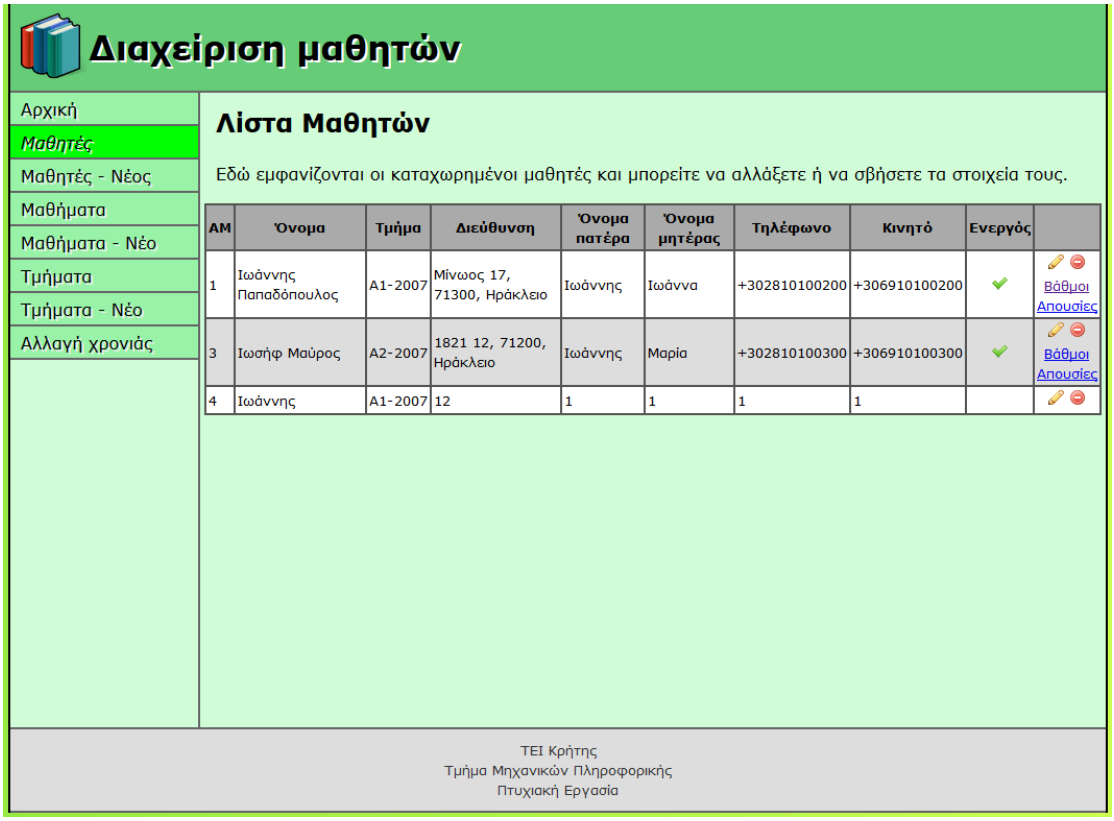
Οι λειτουργίες της εφαρμογής είναι προσβάσιμες από το κεντρικό μενού που βρίσκεται στα αριστερά.



Κάθε μία επιλογή είναι μία ενότητα που δίνει πρόσβαση στις οθόνες προβολής και επεξεργασίας των αντίστοιχων δεδομένων.

5.3 Οθόνη μαθητών

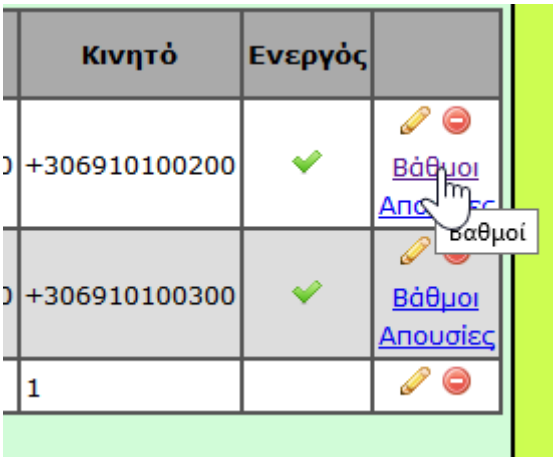
Όταν ο χρήστης πατήσει στην επιλογή μαθητές φαίνεται μία λίστα με τα ονόματα όλων των μαθητών που βρίσκονται καταχωρημένοι στο σύστημα.



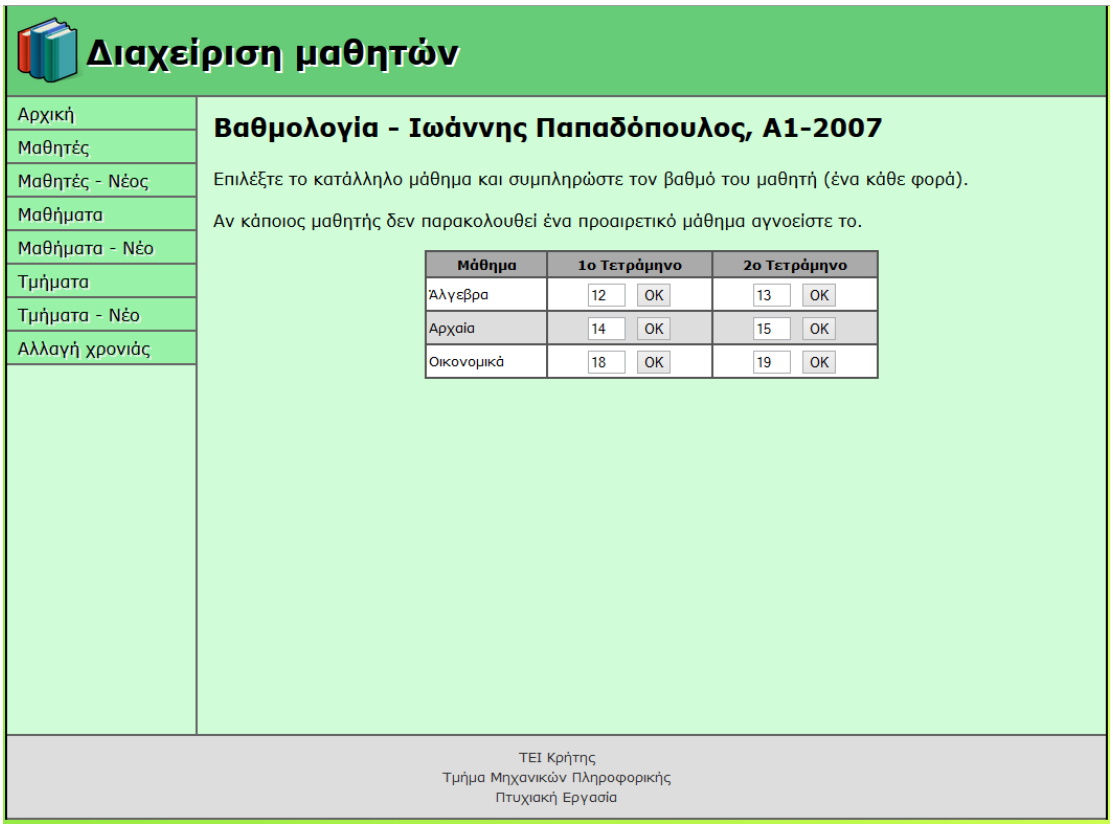
Σε αυτή φαίνονται τα στοιχεία του μαθητή και στην τελευταία στήλη έχει πρόσβαση σε διάφορες λειτουργίες που σχετίζονται με την επεξεργασία τους.

5.3.1 Εισαγωγή βαθμών μαθητή

Ενώ ο χρήστης βλέπει την οθόνη μαθητών μπορεί να πατήσει την επιλογή Βαθμοί που βρίσκεται στην τελευταία στήλη.



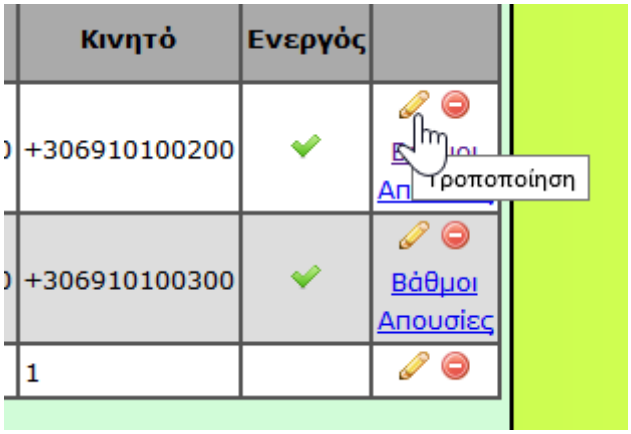
Αυτή θα εμφανίσει μία φόρμα με όλους τους βαθμούς ενός μαθητή.



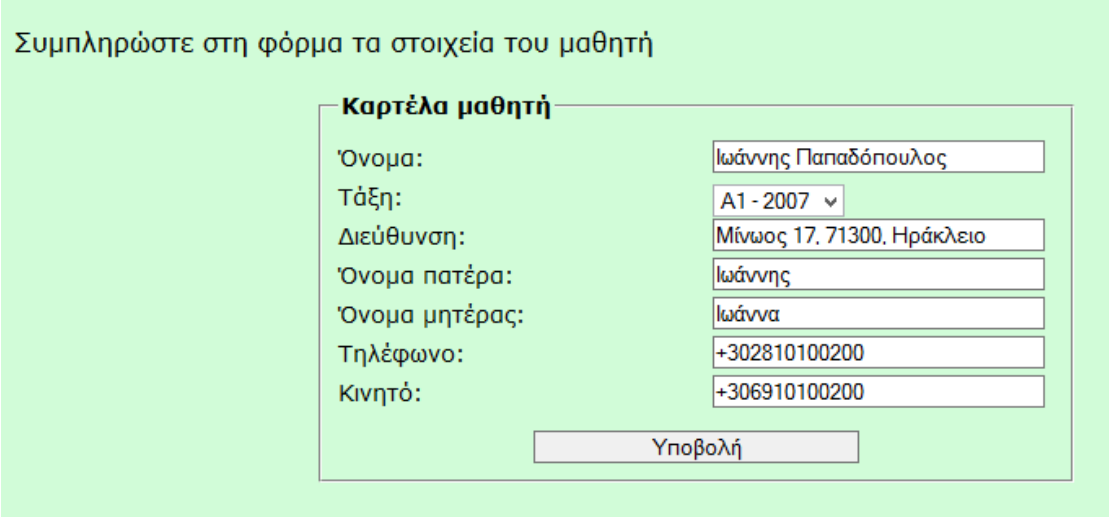
Από εκεί ο χρήστης μπορεί να δει όλους τους βαθμούς ενός μαθητή και να αλλάξει όποιον επιθυμεί. Για να το κάνει αυτό χρειάζεται να πατήσει πάνω στο πεδίο, να γράψει τον νέο βαθμό και να πατήσει OK.

5.3.2 Τροποποίηση στοιχείων μαθητή

Για να αλλάξει τα στοιχεία ενός μαθητή ο χρήστης πρέπει να πατήσει πάνω στο εικονίδιο με το μολύβι.



Αυτό εμφανίζει τη φόρμα με τα στοιχεία του.

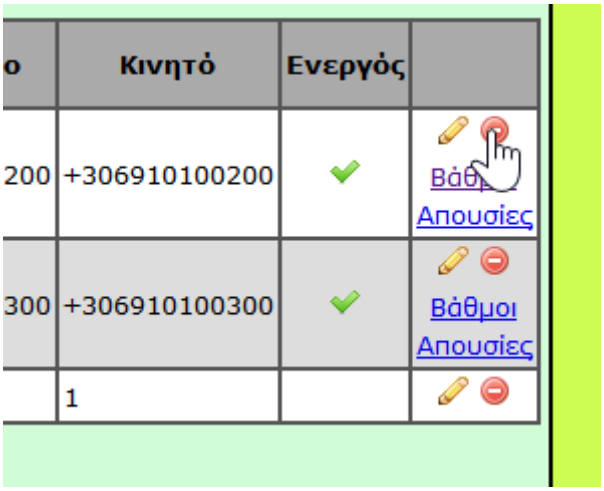


Από εκεί μπορεί να αλλάξει αυτά που θέλει και να πατήσει Υποβολή.

Ένας μαθητής μπορεί να μεταφερθεί ανάμεσα σε διαφορετικά τμήματα αλλά δεν μπορεί να αλλάξει τάξη, καθώς δεν έχει πάρει τα αντίστοιχα μαθήματα και έχει βαθμούς για τα μαθήματα της τάξης του.

5.3.3 Διαγραφή μαθητή

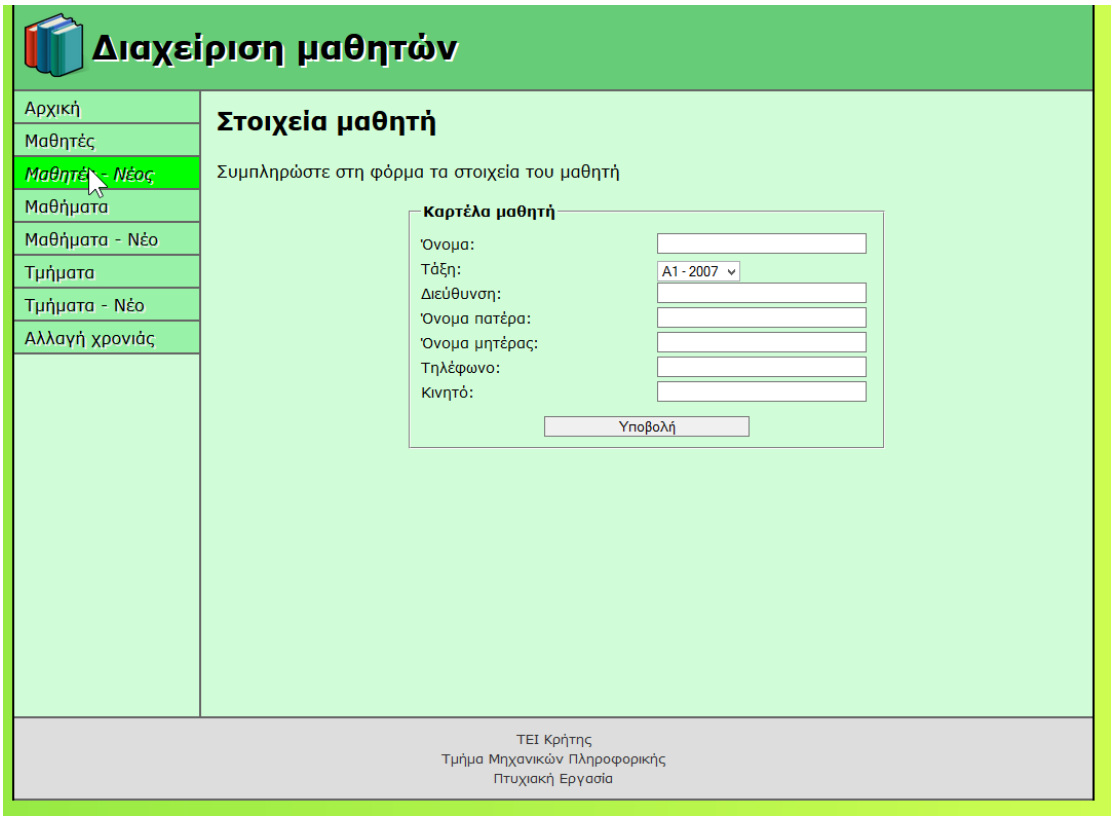
Για να διαγράψει έναν μαθητή ο χρήστης χρειάζεται να πατήσει το αντίστοιχο κουμπί στη στήλη του μαθητή.



Ένας μαθητής δεν διαγράφεται για πάντα αλλά γίνεται ανενεργός. Τα στοιχεία του συνεχίζουν να υπάρχουν.

5.3.4 Εισαγωγή μαθητή

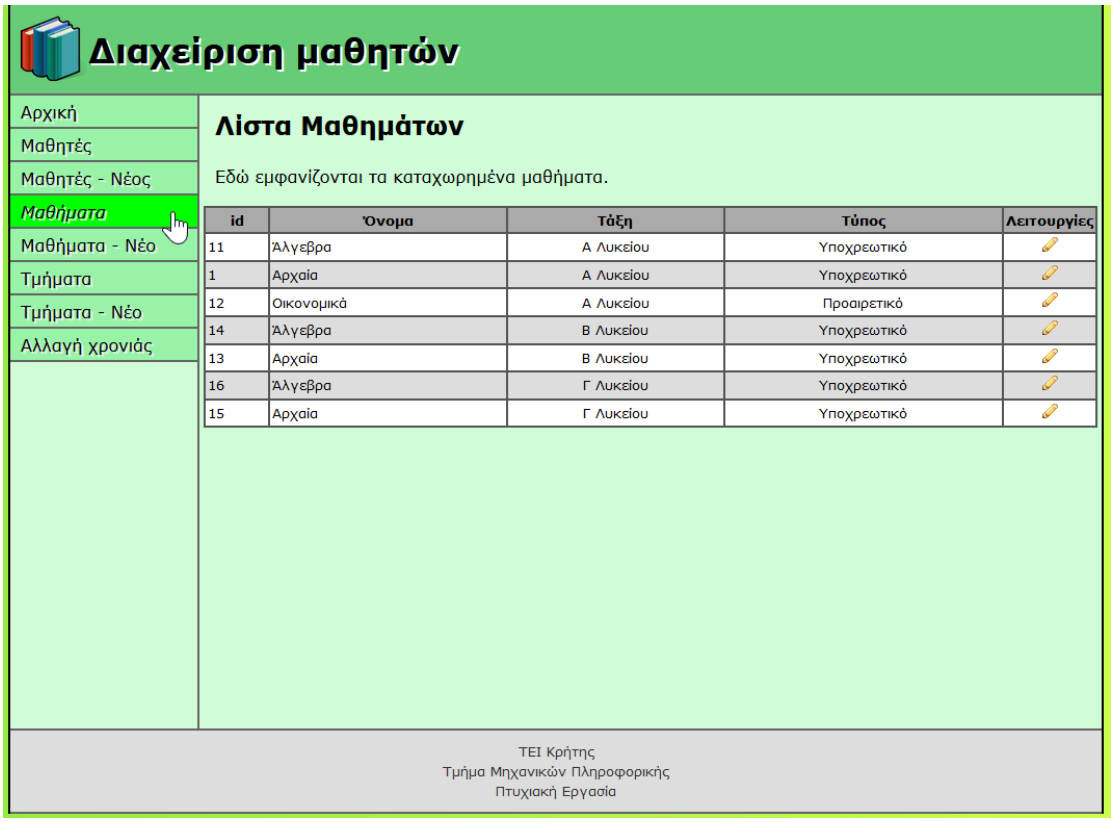
Για να προστεθεί ένας καινούργιος μαθητής χρειάζεται να πατηθεί στο αριστερό μενού η επιλογή μαθητές – Νέος.



Στη φόρμα που εμφανίζεται ο χρήστης προσθέτει τα στοιχεί που χρειάζονται. Ένας νέος μαθητής πρέπει να προστεθεί στα υπάρχοντα τμήματα.

5.4 Μαθήματα

Η λίστα με τα μαθήματα εμφανίζεται κάνοντας κλικ στην επιλογή μαθήματα στα αριστερά.

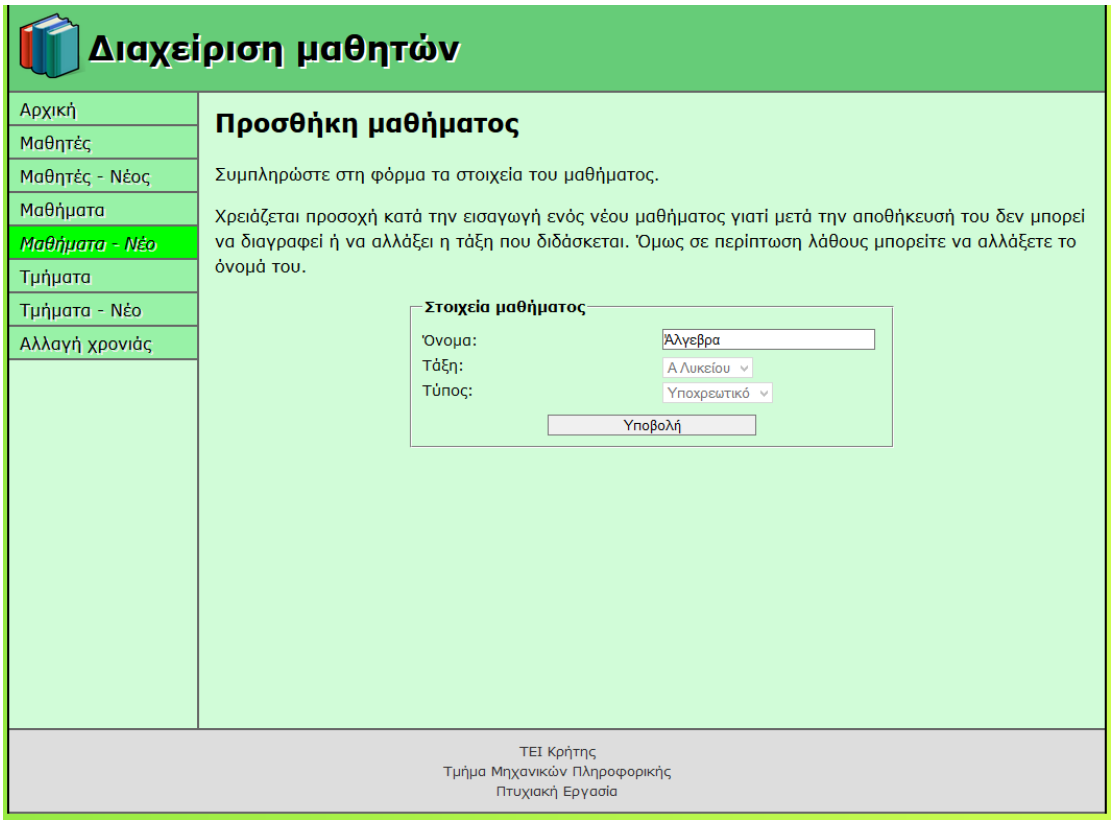


Φαίνεται το είδος του μαθήματος, η τάξη και το όνομά του.

5.4.1 Τροποποίηση στοιχείων μαθήματος

Για να αλλάξει τα στοιχεία ενός μαθήματος ο χρήστης κάνει κλικ στο στοιχείο με το μολύβι.

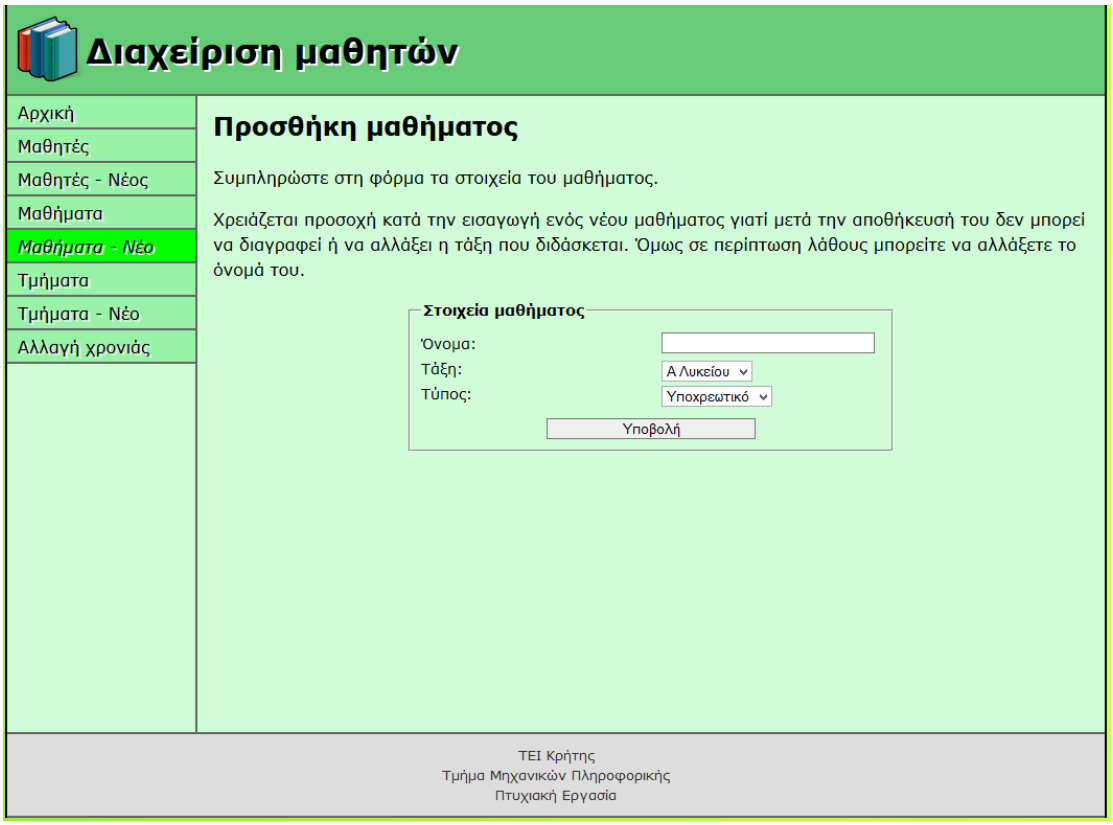
Αυτό εμφανίζει την αντίστοιχη φόρμα.



Καθώς τα μαθήματα παίζουν ρόλο στον υπάρχοντα βαθμό των μαθητών, τα στοιχεία τους δεν μπορούν να αλλάξουν, πέρα από το όνομά τους.

5.4.2 Εισαγωγή μαθήματος

Για την δημιουργία ενός νέου μαθήματος ο χρήστης πατάει στην επιλογή Μαθήματα – Νέο στην αριστερή στήλη.

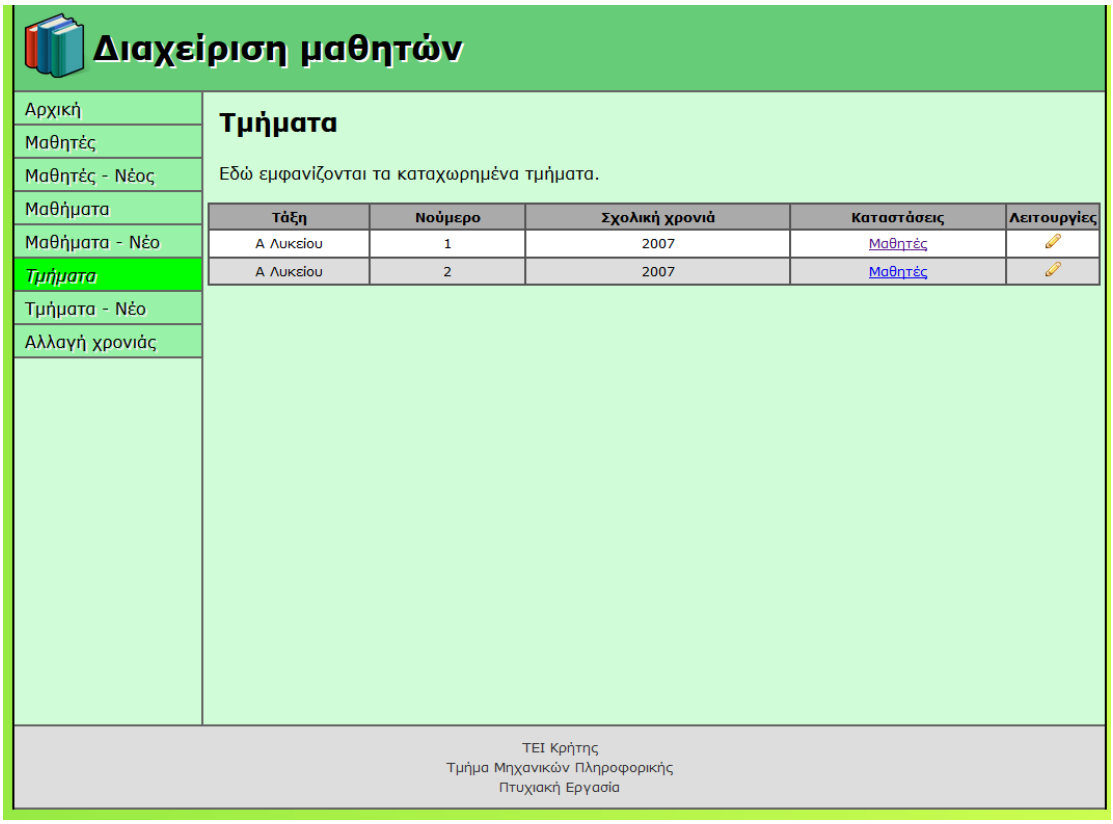


Από εκεί εισάγει τα στοιχεία που θέλει.

Καθώς με την εισαγωγή ενός μαθήματος αυτό λαμβάνεται αυτόματα υπόψη στον βαθμό χρειάζεται ιδιαίτερη προσοχή ώστε να μην γίνει κάποιο λάθος.

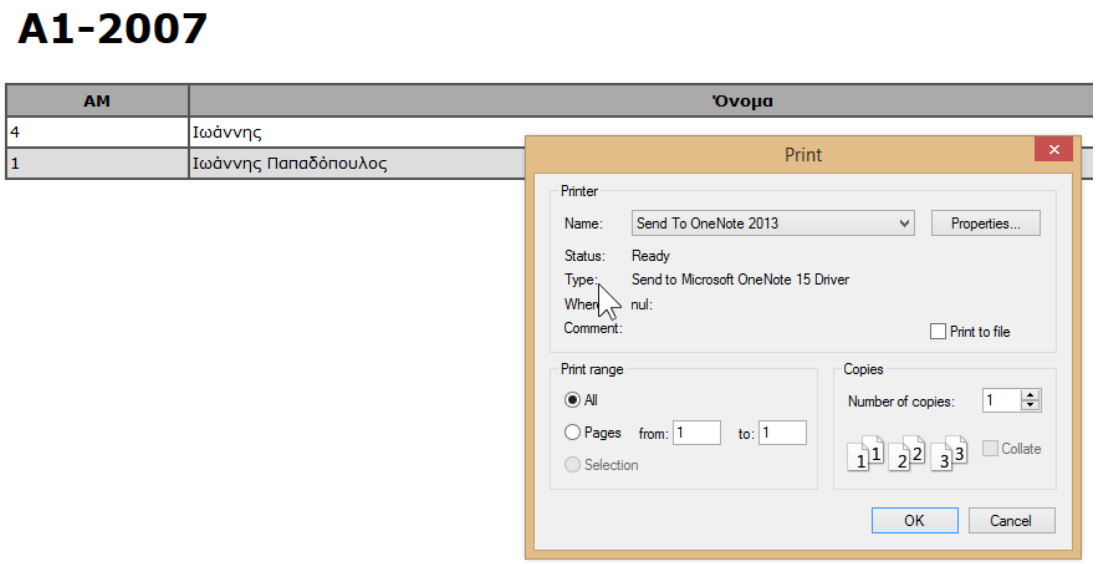
5.5 Τμήματα

Η λίστα με τα τμήματα φαίνεται πατώντας στην αντίστοιχη επιλογή στο αριστερό μενού.



5.5.1 Καταστάσεις

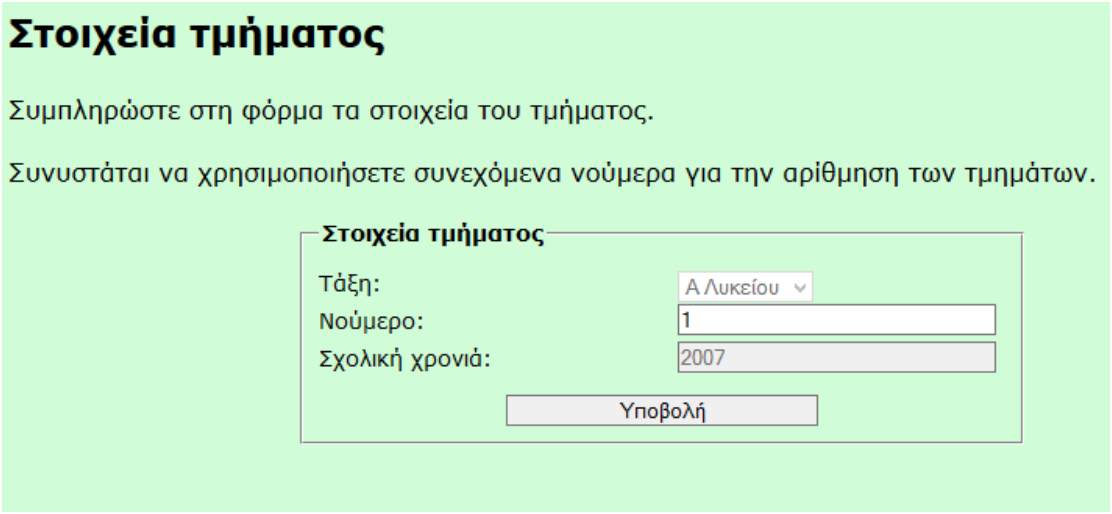
Στη στήλη καταστάσεις φαίνονται οι καταστάσεις που είναι διαθέσιμες. Μία κατάσταση εμφανίζεται σε νέο παράθυρο και τυπώνεται απευθείας.



5.5.2 Τροποποίηση Τμήματος

Για να αλλάξει τα στοιχεία ενός τμήματος ο χρήστης πρέπει να πατήσει στο κουμπί με το μολύβι.

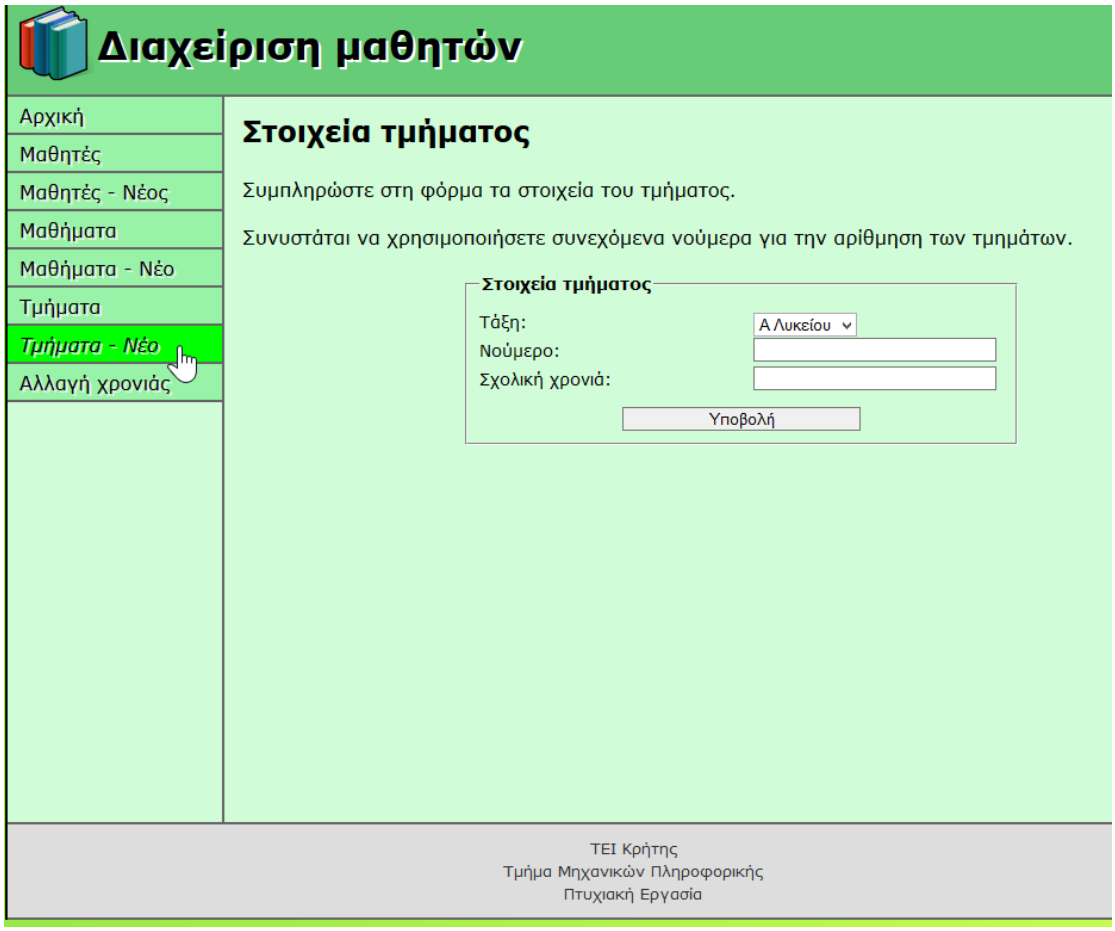
Αυτό εμφανίζει την φόρμα με τα στοιχεία του τμήματος.



Από αυτά μπορεί να αλλάξει μόνο το όνομά του, καθώς υπόλοιπες αλλαγές θα δημιουργούσαν πρόβλημα με τους μαθητές και τους βαθμούς τους.

5.5.3 Δημιουργία νέου τμήματος

Για να δημιουργηθεί ένα νέο τμήμα ο χρήστης πρέπει να το επιλέξει από το κεντρικό μενού.



Έπειτα συμπληρώνει τα στοιχεία του στη φόρμα.

6 Συμπεράσματα

Τα πληροφοριακά συστήματα μπορούν να βοηθήσουν πάρα πολύ αν χρησιμοποιηθούν σε τομείς που παραδοσιακά χειρίζονται μεγάλο όγκο πληροφοριών, μία διαδικασία που παραδοσιακά γίνεται με την χρήση καταστάσεων και αρχείων. Τα πλεονεκτήματα είναι διπλά, τόσο στην μείωση της απαιτούμενης εργασίας όσο και στην βελτίωση των παρεχόμενων υπηρεσιών, καθώς η αναζήτηση της πληροφορίας και η διόρθωσή της μπορούν να γίνουν άμεσα.

7 Κώδικας εφαρμογής

7.1 absence_add.php

```
<?php
    require_once('include/header.php');
    require_once('include/db.php');

    $id = $_REQUEST['id'];

    $q = "SELECT * FROM `student` WHERE `id` = $id";
    $student = query_row($q);

    $name = $student['name'];
    $class_id = $student['class_id'];

    $q = "SELECT * FROM `class` WHERE `id` = $class_id";
    $class = query_row($q);

    $class_name = mb_substr(name_by_id($GRADES, $class['grade']), 0,
1, "UTF-8").$class['number'].'-'.$class['year'];

        $date = date('Y-m-d');

        $q = "SELECT * FROM `absence` WHERE `student_id` = $id AND
`year`=$class[year] ORDER BY `date` ASC";
        $items = query($q);
?>

<h1>Απουσίες - <?php echo "$name, $class_name"?></h1>
<section>
    <p>Αν επιθυμείτε να εισάγετε μία νέα απουσία συμπληρώστε την
παρακάτω φόρμα.</p>

<form method="post" action="absence_add_action.php">
        <fieldset><legend>Εισαγωγή απουσίας</legend>

        <label>Ημερομηνία:</label><input    type="date"    name="date"
class="required" <?php echo "value=\"$date\""; ?> /><br>
        <label>Δικαιολογημένη:</label><input           type="checkbox"
name="justified" /><br/>

        <input type="hidden" name="student_id" value="<?php echo $id;
?>" />
        <input    type="hidden"    name="year"    value="<?php    echo
$class['year']; ?>" />

        <input type="submit" value="Υποβολή" />

        </fieldset>
</form>

<h2>Υπάρχουσες απουσίες</h2>
        <table class="list">
                <thead>
                        <th>Α/Α</th>
                        <th>Ημερομηνία</th>
                        <th>Δικαιολογημένη</th>
                        <th>Ενέργειες</th>
```

```
                        </thead>

                        <?php
                        $i = 0;

                        foreach($items as $item){
                                $i++;
                                if($i%2 == 0) {
                                    echo '<tr class="even">';
                                }else{
                                    echo '<tr class="odd">';
                                }
                                echo '<td>'.$i.'</td>';
                                echo '<td>'.$item['date'].'</td>';

                                if($item['justified'] == 1){
                                        echo    '<td    class="actions"><img
src="images/tick.png" /></td>';
                                }else{
                                        echo                                '<td
class="actions"> </td>';
                                }
                                echo '<td class="actions">'
                                        .'<a
href="absence_delete_action.php?id='.$item['id'].'"         onclick="return
confirm(\'Πρόκειται να διαγράψετε μία εγγραφή!\')"   title="Διαγραφή"><img
src="images/delete.png" /></a> '
                                                                .'</td>';

                                echo '</tr>';
                        }


                        ?>
                    </table>
        </section>
        <?php
            require_once('include/footer.php');
        ?>



        absence_add_action.php
        <?php
            session_start();

            require_once('include/db.php');

            $student_id = $_REQUEST['student_id'];
            $year = $_REQUEST['year'];

              if(isset($_REQUEST['justified'])){
                    $justified = 1;
              }else{
                    $justified = 0;
              }

              $date = $_REQUEST['date'];
              list($year, $month, $day) = explode("-", $date);
```

```
            $date = "$year-$month-$day";

            $q = "INSERT INTO `school`.`absence` (`id`, `date`, `year`,
`justified`, `student_id`)"
                        ." VALUES (NULL, '$date', '$year', $justified,
$student_id);";
            $result = update($q);

        $_SESSION['info'] = 'Τα στοιχεία αποθηκεύτηκαν.';
        header("Location: absence_add.php?id=$student_id");

    ?>



    absence_delete_action.php
    <?php
        session_start();

        require_once('include/db.php');

        connect();

        $id = $_REQUEST['id'];

        $q = "DELETE FROM `school`.`absence` WHERE `id` = $id";
        update($q);

        $_SESSION['info'] = 'Η εγγραφή διαγράφηκε.';
        header("Location: student_list.php");

    ?>
```

7.2 advance_year.php

```
    <?php
    require_once('include/header.php');
    ?>
    <script>
        $(document).ready(function() {
            $("form").validate({
                errorClass: "invalid",
                errorElement: ""
            });
        });
    </script>

    <h1>Αλλαγή χρονιάς</h1>
    <section>
        <p>Επιλέγοντας τον παρακάτω σύνδεσμο μπορείτε να αλλάξετε τη
σχολική χρονιά. Θα υπολογιστούν αυτόματα οι μαθητές που προάγονται και
          αυτοί που μένουν και θα μπουν στα αντίστοιχα τμήματα.</p>
        <p>Αυτή η διαδικασία δεν είναι αναστρέψιμη, οπότε χρειάζεται
μεγάλη προσοχή.</p>
          <p>Για να έχετε σωστά αποτελέσματα βεβαιωθείτε ότι όλοι οι
μαθητές έχουν καταχωρημένους τους βαθμούς για όλα τα μαθήματα που
παρακολουθούν.</p>
```

```
            <p><a                    href="advance_year_action.php">ΕΚΤΕΛΕΣΗ
ΛΕΙΤΟΥΡΓΙΑΣ</a></p>
      </section>
      <?php
      require_once('include/footer.php');
      ?>
```

7.3 advance_year_action.php

```
      <?php
          session_start();
            header('Content-Type: text/html; charset=utf-8');

          require_once('include/db.php');



          $year  =  query_single("SELECT  `year`  FROM  `student`,  `class`
WHERE  `class`.`grade`=1  AND  `student`.`class_id`  =  `class`.`id`  AND
`student`.`active` = 1 LIMIT 1");
            $next_year = $year+1;

            echo "<pre>";
            echo "Τρέχουσα χρονιά: $year, επόμενη: $next_year\n";

            $class1 = array();
            $class2 = array();
            $class3 = array();
            $class4 = array();

            ////////////////////////                          Α     λυκείου
//////////////////////
            $class_count  =  query_single("SELECT  COUNT(`id`)  FROM  `class`
WHERE `class`.`grade`=1 AND `class`.`year` = $year");
            $class_count1 = $class_count;
            echo "\nΑ λυκείου\n\n";
            echo "Σύνολο τμημάτων: $class_count\n";

            $students = query("SELECT `student`.`id`, `student`.`class_id`
FROM `student`, `class` "
                        ."WHERE           `class`.`grade`=1           AND
`student`.`class_id`    =    `class`.`id`    AND    `student`.`active`    AND
`class`.`year` = $year ORDER BY `student`.`name` ASC");

            $student_count = count($students);
            echo "Σύνολο μαθητών: $student_count\n";

            foreach ($students as $student) {
                  // Επιλογή μαθημάτων
                  $q    =    "SELECT    `lesson`.`id`,    `lesson`.`name`,
`lesson`.`type` FROM `lesson`, `class`"
                              ."      WHERE      `lesson`.`grade`      =
`class`.`grade`"
                              ."        AND        `class`.`id`        =
'$student[class_id]'"
            //            ." AND `lesson`.`type` = 1"
```

```
                                ."  ORDER  BY  `lesson`.`grade`,  `type`,
`name`";
                    $lessons = query($q);
                    //print_r($lessons);1

                    $sum = 0;
                    $count = 0;
                    $failed_count = 0;

                    foreach($lessons as $lesson) {
                          $q  =  "SELECT  `value`  FROM  `grade`  WHERE
`student_id`  =  $student[id]  AND  `class_id`  =  $student[class_id]  AND
`lesson_id` = $lesson[id] AND `semester` = 1";
                    $grade1 = query_single($q);

                          $q  =  "SELECT  `value`  FROM  `grade`  WHERE
`student_id`  =  $student[id]  AND  `class_id`  =  $student[class_id]  AND
`lesson_id` = $lesson[id] AND `semester` = 2";
                    $grade2 = query_single($q);

                    if($lesson['type']  ==  2  &&  (sizeof($grade1)  ==  0  ||
sizeof($grade2) == 0) ){
                        continue;
                    }

                          $lesson_avg = ($grade1 + $grade2) / 2.0;

                          $sum += $lesson_avg;
                          $count++;

                          if($sum < 0){
                                $failed_count++;
                          }
                    }

                    $total_grade = $sum / $count;

                    echo  "Μαθητής:  $student[id],  βαθμός:  $total_grade,
αποτυχίες: $failed_count\n";

                    $q  =  "SELECT  COUNT(`id`)  FROM  `absence`  WHERE
`student_id` = $student[id] AND `year` = $year";
                    $total_abs = query_single($q);

                    $q  =  "SELECT  COUNT(`id`)  FROM  `absence`  WHERE
`student_id` = $student[id] AND `year` = $year AND `justified` = 0";
                    $unjust_abs = query_single($q);

                    echo  "Μαθητής:  $student[id],  απουσίες:  $total_abs,
αδικαιολόγητες: $unjust_abs\n";

                    if($unjust_abs >= 70) {
                          echo  "Μαθητής:  $student[id],  απέτυχε  λόγω
αδικαιολόγητων απουσιών\n";
                          array_push($class1, $student);
                    }else if($total_grade < 10){
                          echo "Μαθητής: $student[id], απέτυχε λόγω μέσου
όρου\n";
                          array_push($class1, $student);
```

```
                  }else if($failed_count > 2){
                        echo "Μαθητής: $student[id], απέτυχε επειδή
έμεινε σε πολλά μαθήματα\n";
                        array_push($class1, $student);
                  }else if($failed_count == 2 && $total_grade < 12){
                        echo "Μαθητής: $student[id], απέτυχε επειδή
έμεινε σε 2 μαθήματα με Μ.Ο. < 12\n";
                        array_push($class1, $student);
                  }else if($failed_count == 1 && $total_grade < 11){
                        echo "Μαθητής: $student[id], απέτυχε επειδή
έμεινε σε 2 μαθήματα με Μ.Ο. < 11\n";
                        array_push($class1, $student);
                  }else{
                        echo "Μαθητής: $student[id], πέρασε στην επόμενη
τάξη\n";
                        array_push($class2, $student);
                  }
            }

            ////////////////////////                        Β     λυκείου
//////////////////////
            $class_count = query_single("SELECT COUNT(`id`) FROM `class`
WHERE `class`.`grade`=2 AND `class`.`year` = $year");
            $class_count2 = $class_count;

            echo "\nΒ λυκείου\n\n";
            echo "Σύνολο τμημάτων: $class_count\n";

            $students = query("SELECT `student`.`id`, `student`.`class_id`
FROM `student`, `class` "
                        ."WHERE          `class`.`grade`=1          AND
`student`.`class_id`   =   `class`.`id`   AND   `student`.`active`   AND
`class`.`year` = $year ORDER BY `student`.`name` ASC");

            $student_count = count($students);
            echo "Σύνολο μαθητών: $student_count\n";

            foreach ($students as $student) {
                  // Επιλογή μαθημάτων
                  $q   =   "SELECT   `lesson`.`id`,   `lesson`.`name`,
`lesson`.`type` FROM `lesson`, `class`"
                              ."     WHERE     `lesson`.`grade`     =
`class`.`grade`"
                              ."       AND       `class`.`id`       =
'$student[class_id]'"
            //            ." AND `lesson`.`type` = 1"
                              ."  ORDER  BY  `lesson`.`grade`,  `type`,
`name`";
                  $lessons = query($q);
                  //print_r($lessons);1

                  $sum = 0;
                  $count = 0;
                  $failed_count = 0;

                  foreach($lessons as $lesson) {
                        $q  =  "SELECT  `value`  FROM  `grade`  WHERE
`student_id`  =  $student[id]  AND  `class_id`  =  $student[class_id]  AND
`lesson_id` = $lesson[id] AND `semester` = 1";
```

```
                $grade1 = query_single($q);

                        $q = "SELECT `value` FROM `grade` WHERE
`student_id` = $student[id] AND `class_id` = $student[class_id] AND
`lesson_id` = $lesson[id] AND `semester` = 2";
                $grade2 = query_single($q);

                if($lesson['type'] == 2 && (sizeof($grade1) == 0 ||
sizeof($grade2) == 0) ){
                    continue;
                }

                        $lesson_avg = ($grade1 + $grade2) / 2.0;

                        $sum += $lesson_avg;
                        $count++;

                        if($sum < 0){
                                $failed_count++;
                        }
                }

                $total_grade = $sum / $count;

                echo "Μαθητής: $student[id], βαθμός: $total_grade,
αποτυχίες: $failed_count\n";

                $q = "SELECT COUNT(`id`) FROM `absence` WHERE
`student_id` = $student[id] AND `year` = $year";
                $total_abs = query_single($q);

                $q = "SELECT COUNT(`id`) FROM `absence` WHERE
`student_id` = $student[id] AND `year` = $year AND `justified` = 0";
                $unjust_abs = query_single($q);

                echo "Μαθητής: $student[id], απουσίες: $total_abs,
αδικαιολόγητες: $unjust_abs\n";

                if($unjust_abs >= 70) {
                        echo "Μαθητής: $student[id], απέτυχε λόγω
αδικαιολόγητων απουσιών\n";
                        array_push($class2, $student);
                }else if($total_grade < 10){
                        echo "Μαθητής: $student[id], απέτυχε λόγω μέσου
όρου\n";
                        array_push($class2, $student);
                }else if($failed_count > 2){
                        echo "Μαθητής: $student[id], απέτυχε επειδή
έμεινε σε πολλά μαθήματα\n";
                        array_push($class2, $student);
                }else if($failed_count == 2 && $total_grade < 12){
                        echo "Μαθητής: $student[id], απέτυχε επειδή
έμεινε σε 2 μαθήματα με Μ.Ο. < 12\n";
                        array_push($class2, $student);
                }else if($failed_count == 1 && $total_grade < 11){
                        echo "Μαθητής: $student[id], απέτυχε επειδή
έμεινε σε 2 μαθήματα με Μ.Ο. < 11\n";
                        array_push($class2, $student);
                }else{
```

```
                        echo "Μαθητής: $student[id], πέρασε στην επόμενη
τάξη\n";
                        array_push($class3, $student);
                    }
            }

            ////////////////////////                        Γ     λυκείου
//////////////////////
            $class_count = query_single("SELECT COUNT(`id`) FROM `class`
WHERE `class`.`grade`=3 AND `class`.`year` = $year");
            $class_count3 = $class_count;

            echo "\nΓ λυκείου\n\n";
            echo "Σύνολο τμημάτων: $class_count\n";

            $students = query("SELECT `student`.`id`, `student`.`class_id`
FROM `student`, `class` "
                        ."WHERE          `class`.`grade`=1          AND
`student`.`class_id`    =    `class`.`id`    AND    `student`.`active`    AND
`class`.`year` = $year ORDER BY `student`.`name` ASC");

            $student_count = count($students);
            echo "Σύνολο μαθητών: $student_count\n";

            foreach ($students as $student) {
                  // Επιλογή μαθημάτων
                  $q    =    "SELECT    `lesson`.`id`,    `lesson`.`name`,
`lesson`.`type` FROM `lesson`, `class`"
                              ."      WHERE      `lesson`.`grade`      =
`class`.`grade`"
                              ."        AND        `class`.`id`        =
'$student[class_id]'"
            //            ." AND `lesson`.`type` = 1"
                              ."  ORDER  BY  `lesson`.`grade`,  `type`,
`name`";
                  $lessons = query($q);
                  //print_r($lessons);1

                  $sum = 0;
                  $count = 0;
                  $failed_count = 0;

                  foreach($lessons as $lesson) {
                        $q  =  "SELECT  `value`  FROM  `grade`  WHERE
`student_id`  =  $student[id]  AND  `class_id`  =  $student[class_id]  AND
`lesson_id` = $lesson[id] AND `semester` = 1";
                 $grade1 = query_single($q);

                        $q  =  "SELECT  `value`  FROM  `grade`  WHERE
`student_id`  =  $student[id]  AND  `class_id`  =  $student[class_id]  AND
`lesson_id` = $lesson[id] AND `semester` = 2";
                 $grade2 = query_single($q);

                 if($lesson['type'] == 2 && (sizeof($grade1) == 0 ||
sizeof($grade2) == 0) ){
                     continue;
                 }

                        $lesson_avg = ($grade1 + $grade2) / 2.0;
```

```
                        $sum += $lesson_avg;
                        $count++;

                        if($sum < 0){
                              $failed_count++;
                        }
                  }

                  $total_grade = $sum / $count;

                  echo "Μαθητής: $student[id], βαθμός: $total_grade,
αποτυχίες: $failed_count\n";

                  $q = "SELECT COUNT(`id`) FROM `absence` WHERE
`student_id` = $student[id] AND `year` = $year";
                  $total_abs = query_single($q);

                  $q = "SELECT COUNT(`id`) FROM `absence` WHERE
`student_id` = $student[id] AND `year` = $year AND `justified` = 0";
                  $unjust_abs = query_single($q);

                  echo "Μαθητής: $student[id], απουσίες: $total_abs,
αδικαιολόγητες: $unjust_abs\n";

                  if($unjust_abs >= 70) {
                        echo "Μαθητής: $student[id], απέτυχε λόγω
αδικαιολόγητων απουσιών\n";
                        array_push($class3, $student);
                  }else if($total_grade < 10){
                        echo "Μαθητής: $student[id], απέτυχε λόγω μέσου
όρου\n";
                        array_push($class3, $student);
                  }else if($failed_count > 2){
                        echo "Μαθητής: $student[id], απέτυχε επειδή
έμεινε σε πολλά μαθήματα\n";
                        array_push($class3, $student);
                  }else if($failed_count == 2 && $total_grade < 12){
                        echo "Μαθητής: $student[id], απέτυχε επειδή
έμεινε σε 2 μαθήματα με Μ.Ο. < 12\n";
                        array_push($class3, $student);
                  }else if($failed_count == 1 && $total_grade < 11){
                        echo "Μαθητής: $student[id], απέτυχε επειδή
έμεινε σε 2 μαθήματα με Μ.Ο. < 11\n";
                        array_push($class3, $student);
                  }else{
                        echo "Μαθητής: $student[id], πέρασε στην επόμενη
τάξη\n";
                        array_push($class4, $student);
                  }
            }

            $size1 = count(class1);
            $size2 = count(class2);
            $size3 = count(class3);

            foreach ($class4 as $student) {
                  $q = "UPDATE `school`.`student` SET `active` = '0'
WHERE `student`.`id` = $student[id]";
```

```
                update($q);
            }

            $class_size3 = $size3 / $class_count3;
            for($i=0; $i<=$size3; $i++) {

                $q  =  "INSERT  INTO  `school`.`class`  (`id`,  `grade`,
`number`, `year`) VALUES (NULL, '3', '".($i+1)."', '$next_year');";
                $class_id = update($q);

                $start_pos = $i*$class_size3;
                $end_pos = ($i+1)*$class_size3;

                for($j=$start_pos; $j<$end_pos; $j++) {
                    $id = class3[$j]['id'];

                    $q = "UPDATE `school`.`student` SET"
                        ." `class_id` = '$class_id'"
                        ." WHERE `student`.`id` = $id;";
                    $result = update($q);
                }
            }

            $class_size2 = $size2 / $class_count2;
            for($i=0; $i<=$size2; $i++) {

                $q  =  "INSERT  INTO  `school`.`class`  (`id`,  `grade`,
`number`, `year`) VALUES (NULL, '2', '".($i+1)."', '$next_year');";
                $class_id = update($q);

                $start_pos = $i*$class_size3;
                $end_pos = ($i+1)*$class_size3;

                for($j=$start_pos; $j<$end_pos; $j++) {
                    $id = class2[$j]['id'];

                    $q = "UPDATE `school`.`student` SET"
                        ." `class_id` = '$class_id'"
                        ." WHERE `student`.`id` = $id;";
                    $result = update($q);
                }
            }

            $class_size1 = $size1 / $class_count1;
            for($i=0; $i<=$size1; $i++) {

                $q  =  "INSERT  INTO  `school`.`class`  (`id`,  `grade`,
`number`, `year`) VALUES (NULL, '1', '".($i+1)."', '$next_year');";
                $class_id = update($q);

                $start_pos = $i*$class_size1;
                $end_pos = ($i+1)*$class_size1;

                for($j=$start_pos; $j<$end_pos; $j++) {
                    $id = class1[$j]['id'];

                    $q = "UPDATE `school`.`student` SET"
                        ." `class_id` = '$class_id'"
                        ." WHERE `student`.`id` = $id;";
```

```
                        $result = update($q);
                }
        }

        echo "</pre>";

?>
```

7.4 class_add.php

```
<?php
    require_once('include/header.php');
    require_once('include/db.php');

    if(isset($_REQUEST['id'])){
            $id = $_REQUEST['id'];

            $q = "SELECT * FROM `class` WHERE `id` = $id";
            $item = query_row($q);

            $grade = $item['grade'];
            $number = $item['number'];
            $year = $item['year'];
    }
    else
            $id = 0;
?>
<script>
$(document).ready(function(){
        $("form").validate({
                        errorClass: "invalid",
                        errorElement: ""
                });
});
</script>
        <h1>Προσθήκη τμήματος</h1>
        <section>
                <p>Συμπληρώστε στη φόρμα τα στοιχεία του τμήματος.</p>
                <p>Συνυστάται να χρησιμοποιήσετε συνεχόμενα νούμερα για την αρίθμηση των τμημάτων.</p>

                <form method="post" action="class_add_action.php">
                <fieldset><legend>Στοιχεία τμήματος</legend>

                <label>Τάξη:</label>

                <?php
                    if($id == 0) {
                        echo '<select name="grade">';
                    } else {
                        echo '<select name="grade" disabled>';
                    }


                    foreach($GRADES as $item){
                        if($id == 0 || $grade != $item['id'])
```

```
                                    echo                                '<option
value="'.$item['id'].'">'.$item['name'].'</option>';
                                else
                                    echo   '<option   value="'.$item['id'].'"
selected>'.$item['name'].'</option>';
                                }
                        ?>
                        </select><br>

                        <label>Νούμερο:</label><input            type="text"
name="number" class="required" <?php if($id != 0) echo "value=\"$number\"";
?> /><br>

                        <label>Σχολική   χρονιά:</label><input   type="text"
name="year"  class="required"  <?php  if($id  !=  0)  echo  "value=\"$year\"
disabled"; ?> /><br>

                        <input  type="hidden"  name="id"  value="<?php  echo
$id; ?>" />

                        <input type="submit" value="Υποβολή" />

                        </fieldset>
                        </form>

                </section>
        <?php
            require_once('include/footer.php');
        ?>
```

7.5 class_add_action.php

```
        <?php
            session_start();

            require_once('include/db.php');

            $id = $_REQUEST['id'];
            $number = $_REQUEST['number'];

            if($id == 0){
                $grade = $_REQUEST['grade'];
                $year = $_REQUEST['year'];

                $q = "INSERT INTO `school`.`class` (`id`, `grade`, `number`,
`year`)"
                        ." VALUES (NULL, '$grade', '$number', '$year');";
                $result = update($q);
            }else{
                $q = "UPDATE `school`.`class` SET"
                        ." `number` = '$number' "
                        ." WHERE `id` = $id;";
                $result = update($q);
            }

            $_SESSION['info'] = 'Τα στοιχεία αποθηκεύτηκαν.';
            header("Location: class_list.php");
```

```
?>
```

7.6 class_list.php

```
<?php
      require_once('include/header.php');
?>
<h1>Τμήματα</h1>
<section>
        <p>Εδώ εμφανίζονται τα καταχωρημένα τμήματα.</p>

        <table class="list">
                <thead>
                        <th>Τάξη</th>
                        <th>Νούμερο</th>
                        <th>Σχολική χρονιά</th>
                        <th>Καταστάσεις</th>
                        <th>Λειτουργίες</th>
                </thead>

                <?php
                require_once('include/db.php');
                $i = 0;
                $q = "SELECT * FROM `class` ORDER BY `year` DESC";
                $items = query($q);
                foreach($items as $item){
                        $i++;


                        if($i%2 == 0) {
                            echo '<tr class="even">';
                        }else{
                            echo '<tr class="odd">';
                        }
                        echo                                            '<td
class="center">'.name_by_id($GRADES, $item['grade']).'</td>';
                        echo                                            '<td
class="center">'.$item['number'].'</td>';
                        echo                                            '<td
class="center">'.$item['year'].'</td>';

                        echo          '<td          class="center"><a
href="report_student_list.php?id='.$item['id'].'"
target="report">Μαθητές</a></td>';

                        echo         '<td         class="actions"><a
href="class_add.php?id='.$item['id'].'"><img src="images/edit.png" /></a> '
                                .'</td>';
                        echo '</tr>';
                }


                ?>

        </table>
```

```
	</section>
	<?php
		require_once('include/footer.php');
	?>
```

7.7 grade_add.php

```
	<?php
	    require_once('include/header.php');
	    require_once('include/db.php');

	    $id = $_REQUEST['id'];

	    $q = "SELECT * FROM `student` WHERE `id` = $id";
	    $student = query_row($q);

	    $name = $student['name'];
	    $class_id = $student['class_id'];

	    $q = "SELECT * FROM `class` WHERE `id` = $class_id";
	    $class = query_row($q);

	    $class_name = mb_substr(name_by_id($GRADES, $class['grade']), 0,
1, "UTF-8").$class['number'].'-'.$class['year'];


	    $q  =  "SELECT  `lesson`.`id`,  `lesson`.`name`  FROM  `lesson`,
`class`"
	            ." WHERE `lesson`.`grade` = `class`.`grade`"
	            ." AND `class`.`id` = '$class_id'"
	//            ." AND `lesson`.`type` = 1"
	            ." ORDER BY `lesson`.`grade`, `type`, `name`";
	    $items = query($q);

	?>
	<script>


	</script>
	<h1>Βαθμολογία - <?php echo "$name, $class_name"?></h1>
	<section>
	    <p>Επιλέξτε το κατάλληλο μάθημα και συμπληρώστε τον βαθμό του
μαθητή (ένα κάθε φορά).</p>
	      <p>Αν κάποιος μαθητής δεν παρακολουθεί ένα προαιρετικό μάθημα
αγνοείστε το.</p>

	    <table class="list narrow">
	        <thead>
	                <th>Μάθημα</th>
	                <th>1ο Τετράμηνο</th>
	                <th>2ο Τετράμηνο</th>
	        </thead>

	        <?php
	        $i = 0;

	        foreach($items as $item){
```

```
                    $i++;

                    $q = "SELECT `id`, `value` FROM `grade` WHERE
`student_id` = $id AND `class_id` = $class_id AND `lesson_id` = $item[id]
AND `semester` = 1";
                    $grade1 = query_row($q);
                    if(sizeof($grade1) > 0) {
                        $value1 = $grade1['value'];
                        $id1 = $grade1['id'];
                    }else{
                        $value1 = '';
                        $id1 = 0;
                    }

                    $q = "SELECT `id`, `value` FROM `grade` WHERE
`student_id` = $id AND `class_id` = $class_id AND `lesson_id` = $item[id]
AND `semester` = 2";
                    $grade2 = query_row($q);

                    if(sizeof($grade2) > 0) {
                        $value2 = $grade2['value'];
                        $id2 = $grade2['id'];
                    }else{
                        $value2 = '';
                        $id2 = 0;
                    }

                    if($i%2 == 0) {
                        echo '<tr class="even">';
                    }else{
                        echo '<tr class="odd">';
                    }
                    echo '<td>'.$item['name'].'</td>';

                ?>
                <td class="center">
                    <form            class="small"            method="post"
action="grade_add_action.php">
                        <input type="text" name="value" required <?php
if($id != 0) echo "value=\"$value1\""; ?> />

                        <input type="hidden" name="student_id" value="<?php
echo $id; ?>" />
                        <input type="hidden" name="class_id" value="<?php
echo $class_id; ?>" />
                        <input type="hidden" name="lesson_id" value="<?php
echo $item['id']; ?>" />
                        <input type="hidden" name="semester" value="1" />
                        <input type="hidden" name="id" value="<?php echo
$id1; ?>" />
                        <input type="submit" value="OK" />
                    </form>
                </td>

                <td class="center">
                    <form            class="small"            method="post"
action="grade_add_action.php">
                        <input type="text" name="value" required <?php
if($id != 0) echo "value=\"$value2\""; ?> />
```

```
                    <input type="hidden" name="student_id" value="<?php
echo $id; ?>" />
                    <input type="hidden" name="class_id" value="<?php
echo $class_id; ?>" />
                    <input type="hidden" name="lesson_id" value="<?php
echo $item['id']; ?>" />
                    <input type="hidden" name="semester" value="2" />
                    <input type="hidden" name="id" value="<?php echo
$id2; ?>" />
                    <input type="submit" value="OK" />
                </form>
            </td>
            <?php
            }
            ?>
        </table>

        </section>
    <?php
        require_once('include/footer.php');
    ?>
```

7.8 grade_add_action.php

```
    <?php
        session_start();

        require_once('include/db.php');

        $id = $_REQUEST['id'];
        $student_id = $_REQUEST['student_id'];
        $class_id = $_REQUEST['class_id'];
        $lesson_id = $_REQUEST['lesson_id'];
        $semester = $_REQUEST['semester'];
        $value = $_REQUEST['value'];

        if($id == 0){

            $q = "INSERT INTO `school`.`grade` (`id`, `student_id`,
`lesson_id`, `class_id`, `semester`, `value`)"
                    ." VALUES (NULL, '$student_id', '$lesson_id',
'$class_id', '$semester', '$value');";
            $result = update($q);
        }else if($value != 0){
            $q = "UPDATE `school`.`grade` SET"
                    ." `value` = '$value' "
                    ." WHERE `id` = $id;";
            $result = update($q);
        }else{
           $q = "DELETE FROM `school`.`grade` WHERE `grade`.`id` = $id";
           $result = update($q);
        }

        $_SESSION['info'] = 'Τα στοιχεία αποθηκεύτηκαν.';
        header("Location: grade_add.php?id=$student_id");
```

```
?>
```

7.9 index.php

```
<?php
require_once('include/header.php');
?>
<script>
    $(document).ready(function() {
        $("form").validate({
            errorClass: "invalid",
            errorElement: ""
        });
    });
</script>

<h1>Αρχική σελίδα</h1>
<section>
    <p>Καλωσορίσατε στην αρχική σελίδα.</p>
    <p>Για να διαχειριστείτε τα δεδομένα της εφαρμογής επιλέξτε την
κατάλληλη ενότητα από το μενού.</p>
</section>
<?php
require_once('include/footer.php');
?>
```

7.10 lesson_add.php

```
<?php
    require_once('include/header.php');
    require_once('include/db.php');

    if(isset($_REQUEST['id'])){
            $id = $_REQUEST['id'];

            $q = "SELECT * FROM `lesson` WHERE `id` = $id";
            $item = query_row($q);

            $name = $item['name'];
            $grade = $item['grade'];
            $type = $item['type'];
    }
    else
            $id = 0;
?>
<script>
$(document).ready(function(){
        $("form").validate({
                        errorClass: "invalid",
                        errorElement: ""
                });
});
</script>
        <h1>Προσθήκη μαθητή</h1>
```

```
            <section>
                    <p>Συμπληρώστε στη φόρμα τα στοιχεία του
μαθήματος.</p>
                    <p>Χρειάζεται προσοχή κατά την εισαγωγή ενός νέου
μαθήματος γιατί μετά την αποθήκευσή του δεν μπορεί να διαγραφεί ή να
αλλάξει η τάξη που διδάσκεται. Όμως σε περίπτωση λάθους μπορείτε να
αλλάξετε το όνομά του.</p>

                    <form method="post" action="lesson_add_action.php">
                    <fieldset><legend>Στοιχεία μαθήματος</legend>

                    <label>Όνομα:</label><input  type="text"  name="name"
class="required" <?php if($id != 0) echo "value=\"$name\""; ?> /><br>

                    <label>Τάξη:</label>

                    <?php
                        if($id == 0) {
                            echo '<select name="grade">';
                        } else {
                            echo '<select name="grade" disabled>';
                        }


                        foreach($GRADES as $item){
                            if($id == 0 || $grade != $item['id'])
                                echo                                '<option
value="'.$item['id'].'">'.$item['name'].'</option>';
                            else
                                echo   '<option   value="'.$item['id'].'"
selected>'.$item['name'].'</option>';
                            }
                    ?>
                    </select><br>

                    <label>Τύπος:</label>

                    <?php
                        if($id == 0) {
                            echo '<select name="type">';
                        } else {
                            echo '<select name="type" disabled>';
                        }

                        foreach($TYPES as $item){
                            if($id == 0 || $type != $item['id'])
                                echo                                '<option
value="'.$item['id'].'">'.$item['name'].'</option>';
                            else
                                echo   '<option   value="'.$item['id'].'"
selected>'.$item['name'].'</option>';
                            }
                    ?>
                    </select>

                    <input  type="hidden"  name="id"  value="<?php  echo
$id; ?>" />

                    <input type="submit" value="Υποβολή" />
```

```
                </fieldset>
                </form>

        </section>
<?php
    require_once('include/footer.php');
?>
```

7.11 lesson_add_action.php

```
<?php
    session_start();

    require_once('include/db.php');

    $id = $_REQUEST['id'];
    $name = $_REQUEST['name'];

    if($id == 0){
        $grade = $_REQUEST['grade'];
        $type = $_REQUEST['type'];

        $q = "INSERT INTO `school`.`lesson` (`id`, `name`, `grade`,
`type`)"
                ." VALUES (NULL, '$name', '$grade', '$type');";
        $result = update($q);
    }else{
        $q = "UPDATE `school`.`lesson` SET"
                ." `name` = '$name' "
                ." WHERE `id` = $id;";
        $result = update($q);
    }

    $_SESSION['info'] = 'Τα στοιχεία αποθηκεύτηκαν.';
    header("Location: lesson_list.php");

?>
```

7.12 lesson_list.php

```
<?php
      require_once('include/header.php');
?>
<h1>Λίστα Μαθημάτων</h1>
<section>
        <p>Εδώ εμφανίζονται τα καταχωρημένα μαθήματα.</p>

        <table class="list">
                <thead>
                        <th>id</th>
                        <th>Όνομα</th>
                        <th>Τάξη</th>
                        <th>Τύπος</th>
```

```
                        <th>Λειτουργίες</th>
                    </thead>

                    <?php
                    require_once('include/db.php');
                    $i = 0;
                    $q = "SELECT * FROM `lesson` ORDER BY grade, type,
name";
                    $items = query($q);
                    foreach($items as $item){
                        $i++;


                        if($i%2 == 0) {
                            echo '<tr class="even">';
                        }else{
                            echo '<tr class="odd">';
                        }
                        echo '<td>'.$item['id'].'</td>';
                        echo '<td>'.$item['name'].'</td>';
                        echo                                    '<td
class="center">'.name_by_id($GRADES, $item['grade']).'</td>';
                        echo                                    '<td
class="center">'.name_by_id($TYPES, $item['type']).'</td>';

                        echo          '<td          class="actions"><a
href="lesson_add.php?id='.$item['id'].'"><img  src="images/edit.png"  /></a>
'
                                .'</td>';
                        echo '</tr>';
                    }


                    ?>

            </table>

    </section>
    <?php
          require_once('include/footer.php');
    ?>
```

7.13 report_student_grade_list.php

```
    <?php
        session_start();

        require_once('include/db.php');

        $id = $_REQUEST['id'];

        $q = "SELECT * FROM `class` WHERE `id` = $id";
        $class = query_row($q);
        $class_name = mb_substr(name_by_id($GRADES, $class['grade']), 0,
1, "UTF-8").$class['number'].'-'.$class['year'];
```

```
            $q = "SELECT * FROM `student` WHERE `class_id`='$id' ORDER BY
`name`";
            $items = query($q);

            $q  =  "SELECT  `lesson`.`id`,  `lesson`.`name`  FROM  `lesson`,
`class`"
                    ." WHERE `lesson`.`grade` = `class`.`grade`"
                    ." AND `class`.`id` = '$id'"
                    ." AND `lesson`.`type` = 1"
                    ." ORDER BY `lesson`.`grade`, `type`, `name`";
            $lessons = query($q);

        ?>
        <!doctype html>
        <html>
        <head>
              <meta charset="utf-8">
              <title>Πτυχιακή εργασία</title>
                    <link          href="style.css"          rel="stylesheet"
type="text/css">

        </head>

        <body style="background: none">

        <div class="container" style="background: none; border: none">
            <h1><?php echo $class_name; ?></h1>
                <table class="list" >
                        <thead>
                                <th>ΑΜ</th>
                                <th>Όνομα</th>
                        <?php
                        foreach($lessons as $lesson){
                                        echo '<th>'.$lesson['name'].'</th>';
                        }
                                        echo '<th>Τελικός</th>';

                                  ?>
                        </thead>

                        <?php
                        $i = 0;
                        foreach($items as $item){
                                $i++;
                                if($i%2 == 0) {
                                    echo '<tr class="even">';
                                }else{
                                    echo '<tr class="odd">';
                                }
                                echo '<td>'.$item['id'].'</td>';
                                echo '<td>'.$item['name'].'</td>';

                        }


                        ?>

                </table>
```

```
        <script>
           window.print();
       </script>

       </div><!-- END .container -->
       </body>
       </html>
```

7.14 report_student_list.php

```
       <?php
           session_start();

           require_once('include/db.php');

           $id = $_REQUEST['id'];

           $q = "SELECT * FROM `class` WHERE `id` = $id";
           $class = query_row($q);
           $class_name = mb_substr(name_by_id($GRADES, $class['grade']), 0,
1, "UTF-8").$class['number'].'-'.$class['year'];

           $q = "SELECT * FROM `student` WHERE `class_id`='$id' ORDER BY
`name`";
           $items = query($q);


       ?>
       <!doctype html>
       <html>
       <head>
              <meta charset="utf-8">
              <title>Πτυχιακή εργασία</title>
                     <link          href="style.css"          rel="stylesheet"
type="text/css">

       </head>

       <body style="background: none">

       <div class="container" style="background: none; border: none">
           <h1><?php echo $class_name; ?></h1>
               <table class="list" >
                       <thead>
                              <th>ΑΜ</th>
                              <th>Όνομα</th>
                       </thead>

                       <?php
                       $i = 0;
                       foreach($items as $item){
                              $i++;
                              if($i%2 == 0) {
                                  echo '<tr class="even">';
                              }else{
                                  echo '<tr class="odd">';
                              }
```

```
                        echo '<td>'.$item['id'].'</td>';
                        echo '<td>'.$item['name'].'</td>';

                }


                ?>

        </table>

 <script>
    window.print();
</script>

</div><!-- END .container -->
</body>
</html>
```

7.15 student_add.php

```
<?php
    require_once('include/header.php');
    require_once('include/db.php');

    if(isset($_REQUEST['id'])){
            $id = $_REQUEST['id'];

            $q = "SELECT * FROM `student` WHERE `id` = $id";
            $student = query_row($q);

            $name = $student['name'];
            $address = $student['address'];
            $father = $student['father'];
            $mother = $student['mother'];
            $phone = $student['phone'];
            $mobile = $student['mobile'];
            $active = $student['active'];
            $class_id = $student['class_id'];
    }
    else
            $id = 0;
?>
<script>
$(document).ready(function(){
        $("form").validate({
                        errorClass: "invalid",
                        errorElement: ""
                });
});
</script>
<h1>Προσθήκη μαθητή</h1>
<section>
        <p>Συμπληρώστε στη φόρμα τα στοιχεία του μαθητή</p>

        <form method="post" action="student_add_action.php">
        <fieldset><legend>Καρτέλα μαθητή</legend>
```

```
            <label>Όνομα:</label><input      type="text"      name="name"
class="required" <?php if($id != 0) echo "value=\"$name\""; ?> /><br/>

            <label>Τάξη:</label>

            <?php
                $q  =  "SELECT  *  FROM  `class`  ORDER  BY  `year`  DESC,
`number` ASC";
                $classes = query($q);

                echo '<select name="class_id">';

                foreach($classes as $item){
                    if($id == 0 || $class_id != $item['id'])
                        echo                                          '<option
value="'.$item['id'].'">'.mb_substr(name_by_id($GRADES, $item['grade']), 0,
1, "UTF-8").$item['number'].' - '.$item['year'].'</option>';
                    else
                        echo       '<option       value="'.$item['id'].'"
selected>'.mb_substr(name_by_id($GRADES,  $item['grade']),  0,  1,  "UTF-
8").$item['number'].' - '.$item['year'].'</option>';
                }
            ?>
            </select><br>

            <label>Διεύθυνση:</label><input  type="text"  name="address"
class="required" <?php if($id != 0) echo "value=\"$address\""; ?> /><br/>
            <label>Όνομα πατέρα:</label><input type="text" name="father"
class="required" <?php if($id != 0) echo "value=\"$father\""; ?> /><br/>
            <label>Όνομα       μητέρας:</label><input       type="text"
name="mother" class="required" <?php if($id != 0) echo "value=\"$mother\"";
?> /><br/>
            <label>Τηλέφωνο:</label><input    type="text"    name="phone"
class="required" <?php if($id != 0) echo "value=\"$phone\""; ?> /><br/>
            <label>Κινητό:</label><input    type="text"    name="mobile"
class="required" <?php if($id != 0) echo "value=\"$mobile\""; ?> /><br/>

            <input type="hidden" name="id" value="<?php echo $id; ?>" />

            <input type="submit" value="Υποβολή" />

            </fieldset>
            </form>

        </section>
    <?php
        require_once('include/footer.php');
    ?>
```

7.16 student_add_action.php

```
    <?php
        session_start();

        require_once('include/db.php');

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

```
            $name = $_REQUEST['name'];
            $address = $_REQUEST['address'];
            $father = $_REQUEST['father'];
            $mother = $_REQUEST['mother'];
            $phone = $_REQUEST['phone'];
            $mobile = $_REQUEST['mobile'];
            $class_id = $_REQUEST['class_id'];

            if($id == 0){
                $q   =   "INSERT   INTO   `school`.`student`   (`id`,   `name`,
`address`, `father`, `mother`, `phone`, `mobile`, `class_id`)"
                        ."  VALUES  (NULL,  '$name',  '$address',  '$father',
'$mother', '$phone', '$mobile', '$class_id');";
                $result = update($q);
            }else{
                $q = "UPDATE `school`.`student` SET"
                        ." `name` = '$name',"
                        ." `address` = '$address',"
                        ." `father` = '$father',"
                        ." `mother` = '$mother',"
                        ." `phone` = '$phone',"
                        ." `mobile` = '$mobile',"
                        ." `class_id` = '$class_id'"
                        ." WHERE `student`.`id` = $id;";
                $result = update($q);
            }

            $_SESSION['info'] = 'Τα στοιχεία αποθηκεύτηκαν.';
            header("Location: student_list.php");

        ?>
```

7.17 student_delete_action.php

```
        <?php
            session_start();

            require_once('include/db.php');

            connect();

            $id = $_REQUEST['id'];

            $q = "UPDATE `school`.`student` SET"
                    ." `active` = '0'"
                    ." WHERE `student`.`id` = $id;";
            update($q);

            $_SESSION['info'] = 'Η εγγραφή διαγράφηκε.';
            header("Location: student_list.php");

        ?>
```

7.18 student_list.php

```
<?php
	require_once('include/header.php');
?>
<h1>Λίστα Μαθητών</h1>
<section>
	<p>Εδώ εμφανίζονται οι καταχωρημένοι μαθητές και μπορείτε να αλλάξετε ή να σβήσετε τα στοιχεία τους.</p>

	<table class="list">
		<thead>
			<th>ΑΜ</th>
			<th>Όνομα</th>
			<th>Τμήμα</th>
			<th>Διεύθυνση</th>
			<th>Όνομα πατέρα</th>
			<th>Όνομα μητέρας</th>
			<th>Τηλέφωνο</th>
			<th>Κινητό</th>
			<th>Ενεργός</th>
			<th> </th>
		</thead>

		<?php
		require_once('include/db.php');
		$i = 0;
		$q = "SELECT * FROM `student`";
		$items = query($q);

		foreach($items as $item){
			$i++;
			if($i%2 == 0) {
			    echo '<tr class="even">';
			}else{
			    echo '<tr class="odd">';
			}
			echo '<td>'.$item['id'].'</td>';
			echo '<td>'.$item['name'].'</td>';

			$q = "SELECT * FROM `class` WHERE `id` = $item[class_id]";
			$class = query_row($q);

			$class_name = mb_substr(name_by_id($GRADES, $class['grade']), 0, 1, "UTF-8").$class['number'].'-'.$class['year'];
			echo '<td>'.$class_name.'</td>';
			echo '<td>'.$item['address'].'</td>';
			echo '<td>'.$item['father'].'</td>';
			echo '<td>'.$item['mother'].'</td>';
			echo '<td>'.$item['phone'].'</td>';
			echo '<td>'.$item['mobile'].'</td>';
			if($item['active'] == 1){
				echo '<td class="actions"><img src="images/tick.png" /></td>';
			}else{
				echo '<td class="actions"> </td>';
			}
```

```
                                echo          '<td          class="actions"><a
href="student_add.php?id='.$item['id'].'"          title="Τροποποίηση"><img
src="images/edit.png" /></a> '
                                        .'<a
href="student_delete_action.php?id='.$item['id'].'"          onclick="return
confirm(\'Πρόκειται να διαγράψετε μία εγγραφή!\')"   title="Διαγραφή"><img
src="images/delete.png" /></a> ';
                                        if($item['active'] == 1) {
                                                                        echo
'<br><a href="grade_add.php?id='.$item['id'].'" title="Βαθμοί">Βάθμοι</a>'
                                                                        .'<br><a
href="absence_add.php?id='.$item['id'].'" title="Απουσίες">Απουσίες</a>';
                                                                    }
                                        echo '</td>';

                                echo '</tr>';
                        }


                        ?>

                 </table>

        </section>
        <?php
               require_once('include/footer.php');
        ?>
```