ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΑΘΗΝΑΣ

Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Πληροφορικής

Πτυχιακή εργασία

«Σχεδιασμός και ανάπτυξη web & mobile εφαρμογής για on-line κρατήσεις»

Σπουδαστές:
Νικήτας Βαφειάδης
Βασίλης Δανιγγέλης
Νίκος Μαφρέδας

Επιβλέπων: κ. Ι. Βογιατζής

Αθήνα - Ιανουάριος 2012

Ευχαριστίες

Βασίλης, Νικήτας, Νίκος

Αρχικά, θα θέλαμε να ευχαριστήσουμε όλους όσους στάθηκαν και στήριξαν την προσπάθειά μας, από την αρχή της φοίτησή μας μέχρι και σήμερα. Τις οικογένειές μας που πρόσφεραν απλόχερα ότι καλύτερο καθώς και τα κοντινότερά μας πρόσωπα.

Ιδιαίτερες ευχαριστίες θα θέλαμε να απευθύνουμε στον κύριο Ι. Βογιατζή, καθηγητή του τμήματος και επιβλέπων της πτυχιακής, τόσο για την ανάθεση όσο και για την καθοδήγηση και υποστήριξη, καθ' όλη την διάρκεια διεκπεραίωσης, της παρούσας πτυχιακής.

Τέλος, ένα ακόμα ευχαριστώ στα μέλη της εξεταστικής επιτροπής που μας έκαναν την τιμή να αξιολογήσουν την προσπάθειά μας.

Αθήνα, Ιανουάριος 2012

Περίληψη

Η ραγδαία ανάπτυξη του διαδικτύου σε συνδυασμό με την εξέλιξη των έξυπνων κινητών συσκευών, δημιούργησαν προηγμένες ψηφιακές υπηρεσίες οι οποίες αντικατέστησαν παραδοσιακές μεθόδους. Στα πλαίσια αυτά, τα σύγχρονα συστήματα ενοποιούν διαφορετικές τεχνολογίες βασισμένες σε ετερογενή περιβάλλοντα και προσφέρουν ολοκληρωμένες λύσεις σε θέματα οργάνωσης, διαχείρισης και συλλογής στατιστικών στοιχείων.

Η παρούσα πτυχιακή ασχολείται με τη μελέτη, το σχεδιασμό και την υλοποίηση ενός συστήματος κρατήσεων. Το σύστημα που υλοποιήθηκε αποτελείται από την διαδικτυακή, Android και iPhone εφαρμογή. Η διαδικτυακή εφαρμογή βασίζεται σε σύγχρονες τεχνολογίες όπως είναι η PHP και η SQL. Η Android εφαρμογή βασίζεται στο Android SDK κάνοντας χρήση της γλώσσας προγραμματισμού JAVA, ενώ η iPhone εφαρμογή βασίζεται στο iOS SDK και κάνει χρήση της γλώσσας προγραμματισμού Objective-C.

Με βάση τα παραπάνω, το σύστημα κρατήσεων που αναπτύχθηκε είναι φιλικό προς το χρήστη και πληροί όλες τις προϋποθέσεις ευχρηστίας και ασφάλειας προσωπικών δεδομένων. Η σημαντικότητα του συστήματος έγκειται στη κοινή διαχείριση και στην ασφαλή αποθήκευση των πληροφοριών σε μια βάση δεδομένων. Επίσης, δόθηκε μεγάλη έμφαση στη δημιουργία ενός εύχρηστου περιβάλλοντος διαχείρισης το οποίο μπορεί να χειριστεί ο χρήστης χωρίς να απαιτούνται ιδιαίτερες γνώσεις πληροφορικής.

Η πτυχιακή αυτή είναι δομημένη σε 16 κεφάλαια. Σκοπός τους είναι να παρουσιάσουν τις διαφορετικές τεχνολογίες και τον τρόπο με τον οποίο ενοποιούνται για την επίτευξη ενός κοινού στόχου. Αρχικά περιγράφει την διαδικτυακή εφαρμογή, την μεθοδολογία ανάπτυξης και τα εργαλεία που χρησιμοποιήθηκαν. Ειδικά για την διαδικτυακή εφαρμογή, δόθηκε έμφαση στις τεχνικές προώθησης μέσω των μηχανών αναζήτησης, καθώς και στην ασφάλεια των προσωπικών δεδομένων. Επίσης παρουσιάζονται και αναλύονται οι Android και iPhone εφαρμογές, τα χαρακτηριστικά και τα εργαλεία ανάπτυξής τους.

Κεφάλαιο 1ο

Εισαγωγή

1.1 Πρόλογος

Το 2010 χαρακτηρίστηκε σε παγκόσμιο επίπεδο από μια άνευ προηγουμένου ανάπτυξη της χρήσης «έξυπνων» συσκευών κινητών επικοινωνιών (smartphones). Για πρώτη φορά, οι πωλήσεις smartphones ξεπέρασαν τις πωλήσεις προσωπικών υπολογιστών, προλαβαίνοντας τις σχετικές προβλέψεις κατά 3 περίπου έτη. Στη διεθνή αγορά, οι «έξυπνες» συσκευές εκτιμάται ότι θα ανέλθουν από 225 εκατομμύρια το 2008 σε περισσότερα από 1.8 δισεκατομμύρια το 2013, με μέσο ετήσιο ρυθμό αύξησης που ξεπερνά το 50%.

Παράλληλα με την επέκταση της χρήσης τους, οι νέες συσκευές κινητών επικοινωνιών αλλά και οι υπολογιστές-ταμπλέτες, προσφέρουν εξαιρετικά μεγάλες και νέες δυνατότητες, όπως χαρακτηριστικά:

- την πρόσβαση στο Διαδίκτυο και σε ψηφιακές υπηρεσίες πρακτικά από οπουδήποτε,
- τη διάθεση σημαντικά εμπλουτισμένων ψηφιακών υπηρεσιών που λαμβάνουν υπόψη το γεωγραφικό στίγμα (geo-location services, localized services),
- τη δυνατότητα αμφίδρομης ανταλλαγής δεδομένων όχι μόνο μέσω κειμένου, αλλά και φωτογραφιών, βίντεο και ήχου,
- τη δυνατότητα για διαρκή σύνδεση των χρηστών σε ψηφιακές υπηρεσίες και μέσα κοινωνικής δικτύωσης.

Οι παραπάνω δυνατότητες καταδεικνύουν την πολύ μεγάλη δυναμική που έχουν αποκτήσει οι «έξυπνες» συσκευές κινητών επικοινωνιών σε παγκόσμιο επίπεδο, οι οποίες αποτελούν πλέον τις πιο δυναμικές πλατφόρμες επιχειρηματικής επέκτασης και δραστηριοποίησης. Χαρακτηριστικά, η Gartner εκτιμά ότι λαμβάνοντας υπόψη μόνο το έτος 2011, η παγκόσμια αγορά εφαρμογών και διαφημίσεων σε έξυπνες κινητές συσκευές, θα αγγίξει τα 15 δισεκατομμύρια δολάρια. Η Mason Analysis

εκτιμά περαιτέρω, ότι έως το 2015 το 95% του συνολικού όγκου «κυκλοφορίας» στα δίκτυα κινητών επικοινωνιών, θα αντιπροσωπεύουν τα ψηφιακά δεδομένα.

Η κατάσταση στην Ελλάδα

Στην Ελλάδα, η χρήση των συσκευών κινητής επικοινωνίας κάθε τύπου, παρουσιάζει υψηλά ποσοστά διείσδυσης στον πληθυσμό. Το ποσοστό πληθυσμιακής κάλυψης από δίκτυα κινητών επικοινωνιών αγγίζει το 100%, ενώ το ποσοστό του πληθυσμού που κάνει χρήση κινητών φθάνει το 82%. Οι ενεργές συνδέσεις κινητής επικοινωνίας φθάνουν σε 125% επί του πληθυσμού, έναντι 122% στην Ευρωπαϊκή Ένωση.

Είναι χαρακτηριστικό, ότι στην Ελλάδα το 50% των εξερχόμενων λεπτών ομιλίας γίνεται πλέον από κινητές συσκευές. Παρ' όλη αυτή τη δυναμική, υπάρχει πολύ σημαντική υστέρηση στην αξιοποίηση των «έξυπνων» συσκευών κινητής επικοινωνίας για την πρόσβαση στο Διαδίκτυο και σε καινοτόμες ψηφιακές εφαρμογές, τόσο από τους πολίτες όσο και από τις ελληνικές επιχειρήσεις. Σύμφωνα με την ΕΕΤΤ, κατά το 4ο τρίμηνο του 2010 ο αριθμός των ενεργών συνδρομητών κινητών επικοινωνιών τρίτης γενιάς (3G) δεν μεταβλήθηκε, παραμένοντας στα επίπεδα των 2.786.000. Εντούτοις, παρά τη διαθεσιμότητα σύνδεσης 3G, η αξιοποίηση και ιδιαιτέρως η χρήση καινοτόμων εφαρμογών παραμένει χαμηλή.

1.2 Σκοπός πτυχιακής

Στόχος της παρούσας πτυχιακής είναι η οργάνωση της διαδικασίας κράτησης σε κέντρα διασκέδασης. Πιο συγκεκριμένα το σύστημα υποστηρίζει:

- την οργάνωση και διαχείριση κρατήσεων,
- την διαχείριση καταστημάτων και πελατών,
- την προώθηση μέσω κοινωνικών δικτύων,
- την άμεση εξυπηρέτηση και ενημέρωση των πελατών.

1.3 Δομή Εργασίας

Η παρούσα πτυχιακή αποτελείται από τα ακόλουθα μέρη:

Πρώτο μέρος: Υλοποίηση Ιστοσελίδας και Συστήματος Διαχείρισης Περιεχομένου.

Δεύτερο μέρος: Υλοποίηση Android εφαρμογής.

Τρίτο μέρος: Υλοποίηση iPhone εφαρμογής.

Προχωρώντας σε μία σύντομη επισκόπηση των κεφαλαίων :

Στο 2ο κεφάλαιο περιγράφονται οι μεθοδολογίες ανάπτυξης και τα εργαλεία προγραμματισμού που χρησιμοποιήθηκαν για την ανάπτυξη της Ιστοσελίδας.

Στο 3ο κεφάλαιο γίνεται η ανάλυση των απαιτήσεων για τις ανάγκες του συστήματος, η υλοποίηση της βάσης δεδομένων καθώς και η διαγραμματική της απεικόνιση.

Στο 4ο κεφάλαιο γίνεται αναλυτική παρουσίαση της ιστοσελίδας. Αναλύονται οι τεχνολογίες που χρησιμοποιήθηκαν για την ανάπτυξή της και περιγράφεται η διαδικασία κράτησης.

Στο 5ο κεφάλαιο παρουσιάζεται το Σύστημα Διαχείρισης Περιεχομένου και αναλύονται οι εσωτερικές του λειτουργίες.

Στο 6ο κεφάλαιο αναλύονται όλες οι σύγχρονες τεχνικές και διαδικασίες που αφορούν την προώθηση της ιστοσελίδας σε υψηλές θέσεις στις μηχανές αναζήτησης (Search Engine Optimization).

Στο 7ο κεφάλαιο περιγράφονται οι τρόποι αντιμετώπισης των ευπαθειών των δικτυακών εφαρμογών (PHP Sessions, SQL injection, .htaccess κλπ.).

Στο 8ο κεφάλαιο αναλύεται το Android σαν λειτουργικό σύστημα κινητών συσκευών. Γίνεται αναφορά στις εκδόσεις και στα χαρακτηριστικά του, στα εργαλεία ανάπτυξης και στην αρχιτεκτονική μιας Android εφαρμογής.

Στο 9ο κεφάλαιο περιγράφεται η υλοποίηση της Android εφαρμογής και αναλύονται οι επιμέρους κλάσεις της.

Στο 10ο κεφάλαιο περιγράφεται η επικοινωνία του Android application με τον Server. Αναλύεται το JSON-RPC 2.0 πρωτόκολλο σαν κύριο μοντέλο επικοινωνίας.

Στο 11° κεφάλαιο αναλύεται η δομή του iOS. Παρουσιάζονται τα εργαλεία προγραμματισμού και το πρότυπο ανάπτυξης.

Στο 12° κεφάλαιο παρουσιάζεται η τεχνολογία Push Notification. Αναλύεται η επικοινωνία με την υπηρεσία της apple.

Στο 13° κεφάλαιο αναλύεται η επικοινωνία με το server. Παρουσιάζεται η κλήση των web services.

Στο 14° κεφάλαιο παρουσιάζεται η υλοποίηση της iPhone εφαρμογής και αναλύεται η δομή της.

Στο 15° κεφάλαιο περιγράφονται τα πρωτόκολλα επικοινωνίας που χρησιμοποιούνται μεταξύ Server και mobile εφαρμογών, JSON-RPC 2.0 για Android και XML-Parsing για iPhone.

ΜΕΡΟΣ 1ο - ΙΣΤΟΣΕΛΙΔΑ

Κεφάλαιο 2ο

Μεθοδολογία Ανάπτυξης - Τεχνολογίες και Εργαλεία

2.1 Εισαγωγή

Η υλοποίηση της εργασίας έγινε με χρήση της γλώσσας προγραμματισμού HTML, PHP, της βάσης δεδομένων MySql, της Style Sheet Language CSS για τον σχεδιασμό της εμφάνισης της ιστοσελίδας και της γλώσσας προγραμματισμού Javascript.

2.2 Τι είναι η HTML

Η HTML είναι το ακρωνύμιο των λέξεων HyperText Markup Language, δηλαδή Γλώσσα Χαρακτηρισμού Υπερ-Κειμένου και βασίζεται στη γλώσσα SGML, Standard Generalized Markup Language, που είναι ένα πολύ μεγαλύτερο σύστημα επεξεργασίας εγγράφων. Τα αρχεία της HTML είναι απλά αρχεία κειμένου , τα οποία χρησιμοποιούν ετικέτες για την περιγραφή της δομής και της παρουσίασης μιας ιστοσελίδας , η οποία μπορεί να περιέχει κείμενο, εικόνα, φόρμες, συνδέσεις κ.α.

Για να δημιουργήσουμε ένα αρχείο HTML αρκεί ένας απλός συντάκτης κειμένου. Η HTML ορίζει ένα σύνολο κοινών στυλ για τις Web σελίδες, όπως τίτλοι (*titles*), επικεφαλίδες (*headings*), παράγραφοι (*paragraphs*), λίστες (*lists*) και πίνακες (*tables*). Ορίζει επίσης στυλ χαρακτήρων, όπως η έντονη γραφή (*boldface*) και οι ενότητες κώδικα.

Κάθε στοιχείο έχει ένα όνομα και περιέχεται μέσα στα σύμβολα < >, που αποκαλούνται *tags* (ετικέτες). Όταν γράφουμε μια Web σελίδα με την HTML, στην ουσία δίνουμε τίτλους στα διάφορα στοιχεία της σελίδας μ' αυτά τα tags. Οι φυλλομετρητές, μαζί με τη δυνατότητά τους να ανακτούν σελίδες από το Web, λειτουργούν επίσης και σαν μορφοποιητές για την HTML. Όταν διαβάζουμε μια σελίδα γραμμένη με την HTML σ' έναν φυλλομετρητή, ο φυλλομετρητής διαβάζει

(διερμηνεύει) τα tags της HTML και μορφοποιεί το κείμενο και τις εικόνες στην οθόνη.

Διαφορετικοί φυλλομετρητές, οι οποίοι τρέχουν σε διαφορετικούς υπολογιστές, μπορεί να αντιστοιχίζουν διαφορετικά στυλ σε κάθε στοιχείο μιας σελίδας. Αυτό σημαίνει ότι οι σελίδες που δημιουργούμε με την HTML μπορεί να δείχνουν εντελώς διαφορετικές από σύστημα σε σύστημα και από φυλλομετρητή σε φυλλομετρητή. Δηλαδή, οι πραγματικές πληροφορίες και οι σύνδεσμοι που περιέχουν οι σελίδες μας θα είναι πάντα εκεί, αλλά η εμφάνιση των σελίδων στην οθόνη θα είναι διαφορετική.

2.2.1 Συγκεντρωτικός πίνακας των Tags της HTML

Η παρακάτω λίστα περιέχει τα πιο συνηθισμένα HTML tags.

Tag	Περιγραφή
<html>	Το αρχικό tag για κάθε σελίδα HTML
<head>	Η επικεφαλίδα της σελίδας
<meta>	Στοιχεία για τις μηχανές αναζήτησης
<title>	Ο τίτλος της σελίδα που θα εμφανιστεί στο Browser.
<body>	Το κύριο σώμα της σελίδας
<font>	Μορφοποίηση γραμματοσειράς
** **	Αλλαγή γραμμής (δεν έχει tag τέλους)
<h?>	Επικεφαλίδα (? από 1 έως 6)
<b>	Δημιουργία έντονων χαρακτήρων (bold)
<u>	Υπογράμμιση χαρακτήρων (underline)
<tt>	Χαρακτήρες γραφομηχανής (Typewriter)
<I>	Δημιουργία πλάγιων χαρακτήρων (italic)
<strike>	Διακριτή διαγραφή (strike)
<a>	Σύνδεσμος (link)
<img>	Εισαγωγή εικόνας
<small>	Μικρά γράμματα
<big>	Μεγάλα γράμματα

<p>	Αλλαγή παραγράφου
<hr>	Οριζόντια γραμμή
<sub>	Δείκτης
<sup>	Εκθέτης
<!--	Σχόλιο (δεν εμφανίζονται στον Browser)

```
<html>
	<head>
		<title>Welcome to HTML</title>
	</head>
	<body>
		This is a HTML document!
	</body>
</html>
```

2.3 Cascading Style Sheets (CSS)

Τα Cascading Style Sheets (CSS) αποτελούν ένα πολύ καλό εργαλείο για να μπορούμε να αλλάζουμε την εμφάνιση και τη διάταξη (layout) των ιστοσελίδων μας. Μπορούν να μας γλυτώσουν από πολύ χρόνο και κόπο και μας δίνουν τη δυνατότητα να σχεδιάζουμε τις ιστοσελίδες μας με μια εντελώς καινούργια φιλοσοφία. Η κατανόηση των CSS απαιτεί να υπάρχει κάποια βασική εμπειρία με την HTML.

Για να δουλέψουμε με τα CSS μπορούμε να χρησιμοποιήσουμε κάποιο πρόγραμμα δημιουργίας ιστοσελίδων, όπως είναι τα γνωστά FrontPage και DreamWeaver. Επίσης, ένας επεξεργαστής κειμένου, όπως είναι το Σημειωματάριο (Notepad) των Windows ή το Notepad++ μπορούν να βοηθήσουν στην δημιουργία ή την επεξεργασία ενός CSS αρχείου. Μπορούμε να χρησιμοποιήσουμε όποιον φυλλομετρητή (browser) επιθυμούμε για να βλέπουμε πώς θα εμφανίζονται οι ιστοσελίδες που δημιουργούμε.

2.3.1 Πλεονεκτήματα CSS

- Πολύ μεγαλύτερη ευελιξία. Το CSS κατέστησε εφικτές μορφοποιήσεις οι οποίες ήταν αδύνατες ή πολύ δύσκολες με την κλασσική HTML.

- Ευκολότερη συντήρηση των ιστοσελίδων. Η εμφάνιση μιας ολόκληρης ιστοσελίδας μπορεί να ελέγχεται από ένα μόνο εξωτερικό αρχείο CSS. Έτσι, κάθε αλλαγή στο στυλ της ιστοσελίδας μπορεί να γίνεται με μια μοναδική αλλαγή σε αυτό το αρχείο, αντί για την επεξεργασία πολλών σημείων σε κάθε σελίδα που υπάρχει στο site.
- Μικρότερο μέγεθος αρχείου, δεδομένου ότι ο κάθε κανόνας μορφοποίησης γράφεται μόνο μια φορά και όχι σε κάθε σημείο που εφαρμόζεται.
- Καλύτερο SEO (Search Engine Optimization). Οι μηχανές αναζήτησης δεν «μπερδεύονται» ανάμεσα σε περιεχόμενο και τη μορφοποίησή του, αλλά έχουν πρόσβαση στο περιεχόμενο σκέτο, οπότε είναι πολύ ευκολότερο να το καταγράψουν και να το αρχειοθετήσουν (indexing).
- Γρηγορότερες σελίδες. Όταν χρησιμοποιούμε εξωτερικό αρχείο CSS, ο φυλλομετρητής την πρώτη φορά που θα φορτώσει κάποια σελίδα της ιστοσελίδα μας το αποθηκεύει στην μνήμη (cache), οπότε δεν χρειάζεται να το κατεβάσει την επόμενη φορά που ο χρήστης θα ανοίξει κάποια άλλη σελίδα της ιστοσελίδα μας.

Ακολουθεί παράδειγμα χρήσης για τον ορισμό του tag **<body>**:

```
body {
        width: 1024px;
        background-color:#333;
}
```

2.4 JavaScript

Η γλώσσα προγραμματισμού *JavaScript* αναπτύχθηκε από την εταιρεία Netscape, σε συνεργασία με την Sun Microsystems και η πρώτη της έκδοση δημοσιεύτηκε το 1995. Ακολούθησε η αντίστοιχη γλώσσα της Microsoft η οποία ονομάστηκε Jscript και η επόμενη έκδοση της JavaScript που είχε το όνομα ECMAScript, που αργότερα όμως καθιερώθηκε με το όνομα που είναι γνωστό μέχρι σήμερα. Η JavaScript είναι μία διερμηνευμένη (*interpreted*) γλώσσα προγραμματισμού με ιδιότητες αντικειμενοστραφούς γλώσσας προγραμματισμού, χωρίς όμως να μπορεί να χαρακτηριστεί ως πλήρης αντικειμενοστραφής. Η γλώσσα αυτή, κτίστηκε ουσιαστικά

πάνω στο πρότυπο των γλωσσών C και C++. Από την άλλη όμως έχει μία πολύ σημαντική διαφορά στο ότι διαχειρίζεται τους τύπους δεδομένων πιο χαλαρά (*loosely typed*) σε σχέση με τη σφικτή διαχείριση τύπων δεδομένων (*strongly typed*) που γίνεται στις προαναφερόμενες γλώσσες.

Στην JavaScript οι μεταβλητές δεν είναι απαραίτητο να έχουν ένα συγκεκριμένο τύπο, ή ακόμη είναι δυνατόν να αλλάζουν τύπο κατά τη διάρκεια της ζωής τους. Επίσης, δεν πρέπει να συγχέεται η JavaScript με την Java της Sun Microsystems. Δεν υπάρχει συνάφεια μεταξύ των δύο γλωσσών. Η χρήση του ονόματος JavaScript έγινε για λόγους προώθησης της γλώσσας σε μία εποχή που η εξάπλωση της Java ήταν πολύ μεγάλη.

2.4.1 Χρήσεις της JavaScript

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

- *Λιγότερος φόρτος των server.* Ο έλεγχος και η επικύρωση των δεδομένων που εισάγονται από τους χρήστες γίνεται από τη μεριά του φυλλομετρητή κι έτσι δεδομένα τα οποία δεν είναι σε κατάλληλη μορφή δεν αποστέλλονται στον διακομιστή (*server*). Αυτό όμως δεν σημαίνει ότι ο έλεγχος δεν πρέπει να γίνεται και στη μεριά των εξυπηρετητών, καθώς κάποιος χρήστης μπορεί να μην έχει ενσωματωμένη την JavaScript στον φυλλομετρητή του ή υπάρχει πιθανότητα να την έχει απενεργοποιήσει.
- *Άμεση αλληλεπίδραση με τους χρήστες.* Με την χρήση της JavaScript για τον έλεγχο των δεδομένων μειώνονται οι χρόνοι αναμονής του χρηστών, αφού αυτοί δεν χρειάζεται να περιμένουν μεγάλα χρονικά διαστήματα επαναφόρτωσης της σελίδας, σε περίπτωση που έχουν ξεχάσει να εισάγουν κάποιο δεδομένο ή έχουν εισάγει κάτι λάθος.

- *Αυτόματη διόρθωση λαθών.* Ένα παράδειγμα που μπορεί να κάνει περισσότερο κατανοητό το πώς μπορεί να χρησιμοποιηθεί η JavaScript με αυτόν τον τρόπο είναι αυτό της ημερομηνίας. Πολλά συστήματα βάσεων δεδομένων αποθηκεύουν δεδομένα ημερομηνιών σε μορφή dd-mm-yyyy. Αν κάποιος χρήστης εισάγει κάποια ημερομηνία σε μορφή dd/mm/yyyy τότε κάτι τέτοιο θα μπορούσε να ανιχνευτεί αυτόματα από τον φυλλομετρητή και να μετατραπεί στην σωστή μορφή πριν τα δεδομένα αποσταλούν στον διακομιστή.
- *Αυξημένη χρηστικότητα.* Αυτό επιτυγχάνεται επιτρέποντας στον χρήστη την αλλαγή και αλληλεπίδραση με το γραφικό περιβάλλον χωρίς την επαναφόρτωση της σελίδας. Ένα τέτοιο παράδειγμα είναι τα πτυσσόμενα μενού.
- *Αυξημένη δυνατότητα αλληλεπίδρασης.* Κάτι τέτοιο επιτυγχάνεται στα μενού τα οποία αλληλεπιδρούν όταν ο χρήστης περάσει το mouse πάνω από αυτά – *η λειτουργία hover* – κάτι το οποίο έχει ως αποτέλεσμα να δημιουργεί μία σειρά από γεγονότα τα οποία έχουν προγραμματιστεί να λειτουργούν με έναν συγκεκριμένο τρόπο.

```
function checkReservation(Val)
{
    var student = "student" + Val;

    if (document.getElementById(student).checked == true){
        var checked = "NAI";
    }
    else
    {
        var checked = "OXI";
    }
}
```

2.5 jQuery

Η *jQuery* είναι μια βιβλιοθήκη (*framework*) JavaScript που χρησιμοποιείται από προγραμματιστές για τη ταχεία ανάπτυξη ιστοσελίδων και διαδικτυακών εφαρμογών που χρειάζονται μεγάλη ευχρηστία και διαδραστικότητα (*interactivity*). Η jQuery πρωτοεμφανίστηκε τον Ιανουάριο του 2006 στο BarCamp από τον John Resig.

Πρόκειται για μια βιβλιοθήκη Javascript ανοιχτού κώδικα, υπό τις άδειες MIT License και την GNU General Public License.

Με τη χρήση της jQuery απλοποιείται αρκετά το *client-sidescripting* (δηλαδή ο κώδικας ο οποίος εκτελείται από τον client και όχι από τον server). Ορισμένα από τα χαρακτηριστικά της jQuery είναι: η χρήση στοιχείων DOM, η χρήση AJAX, η κατασκευή animation και visualeffects, λειτουργικότητα σε όλους τους φυλλομετρητές κ.α.

2.5.1 Πλεονεκτήματα της jQuery

- *Ακολουθεί την αρχή KISS (Keep It Simple Stupid):* Η βιβλιοθήκη *jQuery* προσπαθεί να υπεραπλουστεύσει τον προγραμματισμό σε Javascript, προσφέροντας πραγματικά απλούς μηχανισμούς και εντολές μέσω του framework της.
- *Παρέχει πλήρη και αναλυτικότατη τεκμηρίωση* που συμπληρώνεται από την εκτεταμένη παρουσία ηλεκτρονικών βοηθημάτων. Εκτός από την πολύ καλοδουλεμένη τεκμηρίωσή του jQuery, οι ενδιαφερόμενοι μπορούν να ανατρέξουν και στην σελίδα του Visual JQuery όπου μπορούν να βρουν μία εναλλακτική αλλά πολύ βολική, από άποψη δομής, τεκμηρίωση.
- *Υποστηρίζεται από μία πάρα πολύ ενεργή κοινότητα*: Όπως για τα περισσότερα open source έργα λογισμικού, έτσι και για το jQuery, η ύπαρξη μιας κατά το μέγιστο δυνατό ενεργής κοινότητας, αποτελεί τον ακρογωνιαίο λίθο για την ανάπτυξη και ευημερία του.
- *Μικρό μέγεθος*: Το γεγονός ότι το βασικό πακέτο της jQuery είναι μόλις 20Kb αφενός επιβεβαιώνει την πρώτη παρατήρηση, ότι δηλαδή η φιλοσοφία της έγκειται στην απλότητα και αφετέρου κάνει πολύ εύκολη την κατανόηση της αρχιτεκτονικής της.
- *Ποικιλία χαρακτηριστικών*: Η jQuery δίνει τη δυνατότητα στον χρήστη να χρησιμοποιήσει σχεδόν το σύνολο των δυνατοτήτων που προσφέρει η γλώσσα JavaScript. Από απλά χαρακτηριστικά που σχετίζονται με βασικές λειτουργίες εμφάνισης /απόκρυψης ως Ajax κλήσεις και σύνθετα εφέ.

- *Επεκτασιμότητα*: Η λογική με την οποία είναι φτιαγμένη η jQuery είναι απλή, πράγμα που αντικατοπτρίζεται και στον ίδιο της τον κώδικα. Αυτό κάνει πολύ εύκολη την επέκτασή / τροποποίησή της.

2.6 AJAX

Η *AJAX* (asynchronous java and xml) είναι ένας συνδυασμός τεχνολογιών ιστού. Χρησιμοποιείται για την κατασκευή εφαρμογών διαδικτύου (*client-sided*), οι οποίες επικοινωνούν με τον διακομιστή ασύγχρονα (η επικοινωνία γίνεται στο background χωρίς να επηρεάζει την προβολή της υπάρχουσας σελίδας). Για την επικοινωνία χρησιμοποιούμε κυρίως XML μηνύματα (συχνό είναι και το JSON).

Η Ajax είναι ένας από τους πρόσφατους και σημαντικότερους τρόπους βελτίωσης της online εμπειρίας των χρηστών και δημιουργίας νέας και καινοτομικής λειτουργικότητας web. Επιτρέποντας σε συγκεκριμένα τμήματα μιας ιστοσελίδας να προβάλλονται χωρίς ανανέωση ολόκληρης της σελίδας, βελτιώνει σημαντικά την εμπειρία των εφαρμογών ιστού (*Web*). Επιτρέπει επίσης στους προγραμματιστές Web να δημιουργούν διαισθητικές και καινοτομικές διαδραστικές διαδικασίες.

2.7 PHP

Η *PHP* (Hypertext Preprocessor) είναι μια γλώσσα προγραμματισμού για τη δημιουργία σελίδων Web με δυναμικό περιεχόμενο. Μια σελίδα PHP περνά από επεξεργασία από ένα συμβατό διακομιστή του Παγκόσμιου Ιστού (π.χ. Apache), ώστε να παραχθεί σε πραγματικό χρόνο το τελικό περιεχόμενο, που θα σταλεί στο πρόγραμμα περιήγησης των επισκεπτών σε μορφή κώδικα HTML.

Η PHP είναι μια γλώσσα script από την πλευρά του διακομιστή, σχεδιασμένη ειδικά για το Web. Μέσα σε μια HTML σελίδα μπορούμε να ενσωματώσουμε PHP κώδικα, που θα εκτελείται κάθε φορά που θα επισκεπτόμαστε τη σελίδα. Ο PHP κώδικας μεταφράζετε στο Web διακομιστή και δημιουργεί HTML.

Η PHP δημιουργήθηκε το 1994 και ήταν αρχικά η δουλειά ενός ατόμου, του Rasmus Lerdorf. Υιοθετήθηκε και από άλλα ταλαντούχα άτομα και έχει περάσει από

τρεις βασικές εκδόσεις. Η PHP είναι ένα προϊόν ανοιχτού κώδικα. Μπορούμε να έχουμε πρόσβαση στον κώδικα προέλευσης, να τον χρησιμοποιήσουμε, να τον τροποποιήσουμε και να τον αναδιανείμουμε χωρίς χρέωση.

Η PHP αρχικά σήμαινε *Personal Home Page* (προσωπική αρχική σελίδα), αλλά άλλαξε σύμφωνα με την σύμβαση GNU και τώρα σημαίνει *PHP Hypertext Preprocessor* (προεπεξεργαστής κειμένου PHP). Η τρέχουσα βασική της έκδοση είναι η PHP 5.


```
<html>
        <head>
                <title>PHP Script</title>
        </head>
        <body>
                <? php echo "This is my first PHP Script!" ?>
        </body>
</html>
```

2.7.1 Πλεονεκτήματα της PHP

Κάποιοι από τους βασικούς ανταγωνιστές της PHP είναι ο Perl, Microsoft Active Server Pages (ASP) , Java Server Pages (JSP) και Allaire Cold Fusion.

Σε σύγκριση με αυτά τα προϊόντα, η PHP έχει πολλά πλεονεκτήματα όπως:

- Υψηλή απόδοση,
- Διασυνδέσεις με πολλά διαφορετικά συστήματα βάσεων δεδομένων,
- Ενσωματωμένες βιβλιοθήκες για πολλές συνηθισμένες Web διαδικασίες,
- Χαμηλό κόστος,
- Ευκολία μάθησης και χρήσης,
- Μεταφερσιμότητα,
- Διαθεσιμότητα του κώδικα προέλευσης.

2.7.2 Τι μπορεί να κάνει η PHP

Η PHP επικεντρώνεται κυρίως στο server-side scripting. Στο πιο βασικό επίπεδο, μπορεί να κάνει ό,τι και τα άλλα προγράμματα της τεχνολογίας CGI, όπως επεξεργασία των δεδομένων μιας φόρμας, δημιουργία δυναμικού περιεχομένου ιστοσελίδων ή αποστολή και λήψη cookies.

Υπάρχουν τρεις κύριοι τομείς που χρησιμοποιείται ένα PHP script.

- *Server-side scripting.* Αυτό είναι το πιο παραδοσιακό και το κύριο πεδίο για την PHP. Απαιτούνται τρία πράγματα για να δουλέψει αυτό. Τον PHP μεταγλωτιστή (parser) (CGI ή server module), έναν εξυπηρετητή σελίδων (webserver) και ένα φυλλομετρητή.
- *Command line scripting.* Κάνοντας χρήση μόνο του PHP μεταγλωττιστή μπορούμε να δημιουργήσουμε ένα PHP script και να το εκτελέσουμε χωρίς server ή browser. Αυτός ο τύπος είναι ιδανικός για scripts που εκτελούνται συχνά με τη χρήση της *Cron* (σε *nix ή Linux) ή με τον *Task Scheduler* (στα Windows). Αυτά τα scripts μπορούν να χρησιμοποιηθούν για απλές εργασίες επεξεργασίας κειμένου.
- *Εγγραφή client-side GUI εφαρμογών (Γραφικά περιβάλλοντα χρηστών).* Η PHP δεν ευνοεί την ανάπτυξη παραθυρικών εφαρμογών, ωστόσο χρησιμοποιώντας την βιβλιοθήκη PHP-GTK γίνεται δυνατή η ανάπτυξη cross-platform εφαρμογών. Το PHP-GTK δεν συμπεριλαμβάνεται στην κύρια διανομή της.

Η PHP μπορεί να χρησιμοποιηθεί σε όλα τα κύρια λειτουργικά συστήματα, συμπεριλαμβανομένου του Linux, πολλών εκδοχών του Unix (HP-UX, Solaris και OpenBSD), Microsoft Windows, Mac OS X, RISC OS και πιθανώς σε άλλα. Η PHP υποστηρίζει επίσης τους Apache, Microsoft Internet Information Server, Personal Web Server, Netscape και iPlanet servers, Oreilly Website Pro server, Caudium, Xitami, OmniHTTPd και πολλούς άλλους webserver.

Οι δυνατότητες της PHP περιλαμβάνουν την εξαγωγή εικόνων, αρχείων PDF και ταινίες Flash χρησιμοποιώντας βιβλιοθήκες όπως libswf και Ming. Επίσης παρέχει την δυνατότητα εξαγωγής οποιουδήποτε κειμένου σε XML αρχείο. Η PHP δημιουργεί αυτόματα τα παραπάνω αρχεία και τα αποθηκεύει στο σύστημα αρχείων, αντί να τα εκτυπώνει, αποτελώντας έτσι μια server-side cache για το δυναμικό μας

περιεχόμενο. Ένα από τα πιο δυνατά και σημαντικά χαρακτηριστικά της PHP είναι η υποστήριξη που έχει για ένα μεγάλο σύνολο βάσεων δεδομένων. Οι βάσεις δεδομένων που υποστηρίζονται είναι οι εξής:

Adabas D	Ingres	Oracle (OCI7 and OCI8)
dBase	InterBase	Ovrimos
Empress	FrontBase	PostgreSQL
FilePro (read-only)	mSQL	Solid
Hyperwave	Direct MS-SQL	Sybase
IBM DB2	MySQL	Velocis
Informix	ODBC	Unix dbm

Επιπροσθέτως, η PHP υποστηρίζει την επικοινωνία μεταξύ υπηρεσιών χρησιμοποιώντας πρωτόκολλα όπως LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (στα Windows) και πολλά άλλα.

2.7.3 Επικοινωνία της PHP με βάσεις δεδομένων

Η PHP θεωρείται η καλύτερη επιλογή για την δημιουργία δυναμικών εφαρμογών που επικοινωνούν με βάσεις δεδομένων. Ένας βασικός λόγος για να την επιλέξουμε είναι ότι διαθέτει ενσωματωμένες συναρτήσεις που επικοινωνούν με μεγάλο αριθμό εμπορικών συστημάτων βάσεων δεδομένων.

Το παρακάτω παράδειγμα παρουσιάζει τις συναρτήσεις της PHP, που χρησιμοποιούνται για τη σύνδεση με μια βάση δεδομένων στη MySQL, τη δημιουργία επερωτήσεων και την ανάκτηση των αποτελεσμάτων για περαιτέρω επεξεργασία τους από την εφαρμογή.

```
<?php
//Δημιουργία σύνδεσης προς τον MySQL Server
$connection = mysql_connect("localhost","username","passwd");

//Εμφάνιση κωδικού σφάλματος σε περίπτωση αποτυχημένης σύνδεσης
if (!$connection)
      die('Could not connect: ' . mysql_error());

//Επιλογή μιας βάσης δεδομένων
```

```
mysql_select_db("mydb", $connection);

//Δημιουργία μιας επερώτησης (query) προς τη βάση
$result = mysql_query ("SELECT * FROM mytable", $connection);

// Επιστροφή των αποτελεσμάτων του query
while ($row = mysql_fetch_array($result, MYSQL_NUM))
{
      // επεξεργασία των αποτελεσμάτων
}

//Κλείσιμο της σύνδεσης με τον Server
mysql_close ($connection);
?>
```

- *mysql_connect()* χρησιμοποιείται για τη σύνδεση με τον MySQL Server και δέχεται τις εξής παραμέτρους: το *hostname* ή την *IP* διεύθυνση του server, το όνομα του χρήστη που έχει πρόσβαση στον server και τον κωδικό ασφαλείας του. Η τιμή που επιστρέφει η συνάρτηση χρησιμοποιείται σαν παράμετρος στις μετέπειτα συναρτήσεις που καλούνται προς τον MySQL Server.
- *mysql_error()* επιστρέφει το μήνυμα λάθους σε περίπτωση αποτυχίας της προηγούμενης ενέργειας που έγινε προς τον MySQL Server.
- *mysql_select_db()* διαλέγει μια συγκεκριμένη βάση δεδομένων η οποία έχει ήδη δημιουργηθεί στον MySQL Server. Οι παράμετροι που εμπεριέχονται είναι το όνομα της βάσης και η τιμή που επιστράφηκε από την mysql_connect().
- *mysql_query()* χρησιμοποιείται για την αποστολή μιας επερώτησης (*query*) στη βάση δεδομένων που ορίζει η δεύτερη παράμετρος $connection. Η πρώτη παράμετρος είναι το query που θα εκτελεστεί.
- *mysql_fetch_array()* επιτρέπει την ανάκτηση των εγγραφών (rows) που επέστρεψε το προηγούμενο query προς τη βάση.
- *mysql_close()* χρησιμοποιείται για να κλείσει την σύνδεση που ορίζει η παράμετρος $connection, αν και αυτό δεν είναι απαραίτητο αφού όλες οι συνδέσεις προς την βάση δεδομένων κλείνουν αυτόματα όταν τελειώνει το script.

2.8 MySQL

Η *MySQL* είναι ένα πολύ γρήγορο και δυνατό, σύστημα διαχείρισης βάσεων δεδομένων. Μια βάση δεδομένων επιτρέπει την αποθήκευση, την αναζήτηση, την ταξινόμηση και δίνει τη δυνατότητα να ανακαλέσει ο προγραμματιστής τα δεδομένα πιο αποτελεσματικά. Ο MySQL διακομιστής ελέγχει την πρόσβαση στα δεδομένα μας, για να μπορούν να δουλεύουν πολλοί χρήστες ταυτόχρονα, για να παρέχει γρήγορη πρόσβαση και να διασφαλίζει ότι μόνο πιστοποιημένοι χρήστες μπορούν να έχουν πρόσβαση.

Συνεπώς η MySQL είναι ένας πολυνηματικός διακομιστής πολλαπλών χρηστών. Χρησιμοποιεί την SQL (Structured Query Language) την τυπική γλώσσα ερωτημάτων για βάσεις δεδομένων, παγκόσμια. Η MySQL είναι διαθέσιμη από το 1996 αλλά η ιστορίας της ξεκινά από το 1979.

Ένα από τα μεγαλύτερα πλεονεκτήματα της MySQL είναι η δυνατότητα που δίνει για σύνδεση σε πολλές διαφορετικές βάσεις δεδομένων. Οι βάσεις δεδομένων που υποστηρίζονται περιλαμβάνουν τις: Adabas D, InterBase, PostgreSQL, dBase, FrontBase, SQLite, Empress, mSQL, Solid, FilePro (read-only), Direct MS-SQL, Sybase, Hyperwave, MySQL, Velocis, IBM, ODBC.


```
SELECT
       *
FROM
       mytable
WHERE
       user_id = '1';
```

2.8.1 Πλεονεκτήματα MySQL

Μερικοί από τους κύριους ανταγωνιστές της MySQL είναι οι PostgreSQL, Microsoft SQL και Oracle. Παρακάτω αναλύονται τα πλεονεκτήματά της:

- *Απόδοση* : Η MySQL ευνοεί την γρήγορη ευελιξία.
- *Χαμηλό κόστος* : Η MySQL διατίθεται είτε δωρεάν, με άδεια ανοικτού κώδικα (Open Source), είτε με χαμηλό κόστος αν απαιτείται από την εφαρμογή μας με εμπορική άδεια.
- *Ευκολία Χρήσης* : Οι περισσότερες μοντέρνες βάσεις δεδομένων χρησιμοποιούν SQL. Αν έχει χρησιμοποιηθεί κάποιο άλλο σύστημα διαχείρισης βάσεων δεδομένων, δεν θα υπάρχει πρόβλημα να προσαρμοστεί σε αυτό.
- *Μεταφερσιμότητα* : Η MySQL μπορεί να χρησιμοποιηθεί σε πολλά διαφορετικά συστήματα Unix όπως και στα Microsoft Windows.
- *Κώδικας Προέλευσης* : Αντιστοίχως με την PHP, ο κώδικας προέλευσης της MySQL μπορεί να τροποποιηθεί.
- *Νέα έκδοση* : Η νέα έκδοση MySQL 5 περιλαμβάνει νέες λειτουργίες υποστηρίζοντας projects με υψηλή αξιοπιστία.

2.8.2 Τρόπος λειτουργίας MySQL Βάσης Δεδομένων

Ο τρόπος λειτουργίας της MySQL είναι ίδιος με αυτόν που ακολουθούν όλες οι Web βάσεις δεδομένων. Τα βήματα λειτουργίας των αρχιτεκτονικών Web Βάσεων Δεδομένων μπορούν να συνοψιστούν στο παρακάτω παράδειγμα:

- Έστω ότι ο χρήστης, χρησιμοποιώντας τον web browser που κάνει HTTP αίτηση, επιδιώκει την αναζήτηση όλων των χρηστών μιας ιστοσελίδας. Για πρακτικούς λόγους θεωρούμε ότι η σελίδα των αποτελεσμάτων αναζήτησης ονομάζεται *results.php*.
- Ο web διακομιστής λαμβάνει την αίτηση για τη σελίδα *results.php*, ανακαλεί το αρχείο και το περνά στην μηχανή PHP για επεξεργασία.
- Η μηχανή PHP αρχίζει την ανάλυση του script. Μέσα στο script, υπάρχει μια εντολή που συνδέει την βάση δεδομένων και εκτελεί ένα ερώτημα (την

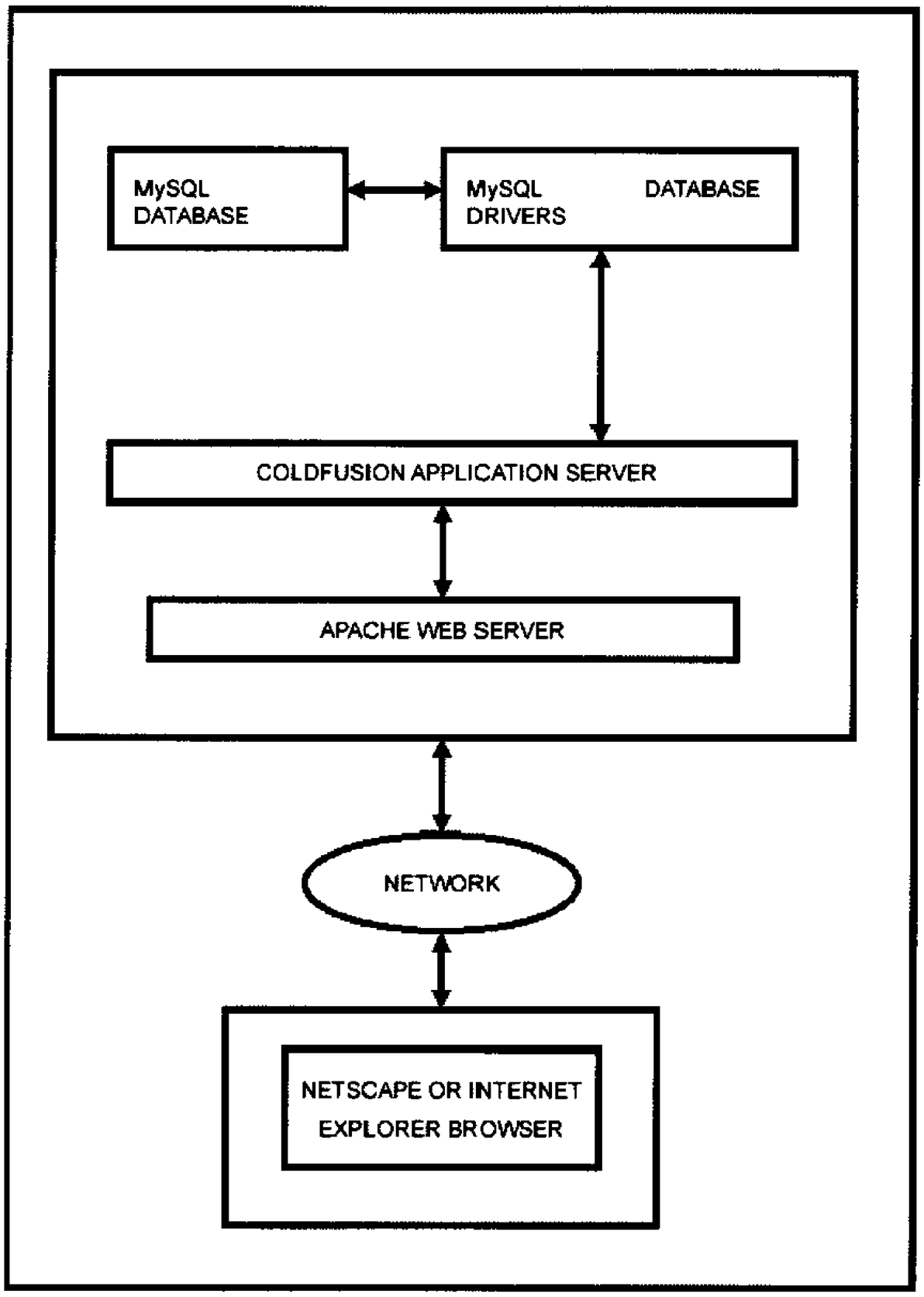
αναζήτηση των χρηστών). Η PHP ανοίγει μια σύνδεση με τον MySQL διακομιστή και στέλνει το κατάλληλο ερώτημα.

- Ο MySQL διακομιστής λαμβάνει το ερώτημα της βάσης δεδομένων, το επεξεργάζεται και στέλνει τα αποτελέσματα, μια λίστα χρηστών, στη μηχανή PHP.
- Η μηχανή PHP σταματά την εκτέλεση του script, που συνήθως περιλαμβάνει την μορφοποίηση των αποτελεσμάτων του ερωτήματος σε HTML. Τέλος επιστρέφει την τελική HTML σελίδα στον web διακομιστή.
- Ο web διακομιστής περνά την HTML σελίδα ξανά στον browser, όπου ο χρήστης μπορεί να δει τη λίστα των χρηστών που ζήτησε.

Το προαναφερθέν παράδειγμα περιέχει τρία επίπεδα επεξεργασίας, που αποτελούν παράλληλα και χαρακτηριστικό των περισσοτέρων Web εφαρμογών (three-tier client-server). Τα επίπεδα αυτά είναι τα εξής:

1. Βάση Δεδομένων,
2. Server,
3. Client.

Το παρακάτω σχεδιάγραμμα (*Εικόνα 2.1*) περιγράφει την διαδικασία που περιγράψαμε παραπάνω και τον τρόπο επικοινωνίας μεταξύ των τριών επιπέδων.



Εικόνα 2.1: *three-tier client-server*

2.9 Apache Web Server

Ο Apache Web Server είναι ένας δημοφιλής διακομιστής διαδικτύου που διανέμεται ελεύθερα στο διαδίκτυο. Αναπτύχθηκε και συντηρείται από μια ομάδα εθελοντών, που ήθελαν να υλοποιήσουν έναν εμπορικό και εύρωστο κώδικα με πολλά χαρακτηριστικά.

Ο Apache όπως έχει αποδειχτεί είναι ταχύτατος, σταθερός, ασφαλής και υποστηρίζει τα περισσότερα χαρακτηριστικά από οποιονδήποτε άλλο διακομιστή διαδικτύου. Είναι εγκατεστημένος στο 80% των διακομιστών παγκοσμίως (πάνω από 6 εκατομμύρια διακομιστές). Φιλοξενεί εκατομμύρια sites τα οποία δέχονται εκατομμύρια hits καθημερινά, χωρίς όμως να παρουσιάζεται κάποιο πρόβλημα.

Σήμερα θεωρείται ένας από τους πιο σταθερούς διακομιστές διαδικτύου που κυκλοφορούν. Αρκετοί εμπορικοί διακομιστές διαδικτύου όπως ο HTTP Server της IBM χρησιμοποιούν τον πυρήνα του Apache.

2.10 XAMPP

Για την ανάπτυξη και διανομή διαδικτυακών εφαρμογών είναι απαραίτητο κάποιο εργαλείο, το οποίο να υποστηρίζει την εκτέλεση δυναμικών ιστοσελίδων. Δηλαδή, να διαθέτει έναν web server που να μπορεί να εκτελεί server side scripts καθώς και να υποστηρίζει τη χρήση βάσεων δεδομένων

Η εφαρμογή *XAMPP 1.7.7*, η οποία χρησιμοποιήθηκε για την ανάπτυξη της εφαρμογής , αποτελείται από τα παρακάτω πακέτα λογισμικού :

1. *Τον Διαδικτυακό Εξυπηρετητή Apache 2.2.21*

   Στον web server αποθηκεύονται όλα τα αρχεία (HTML,PHP) μιας ιστοσελίδας και είναι υπεύθυνος για την αλληλεπίδραση με τα προγράμματα περιήγησης των επισκεπτών.

2. *Την γλώσσα σεναρίων PHP 5.3.8*

   Με το συγκεκριμένο πακέτο μπορεί να εκτελεί σενάρια στον εξυπηρετητή και να αποστέλλει τα αποτελέσματα στον επισκέπτη.

3. *Το εργαλείο για την διαχείριση βάσεων δεδομένων phpMyAdmin 3.4.5*

   Με αυτή την εφαρμογή, δίνεται η δυνατότητα διαχείρισης των διάφορων βάσεων δεδομένων, με εύκολο τρόπο μέσω web browser και χωρίς την πληκτρολόγηση εντολών.

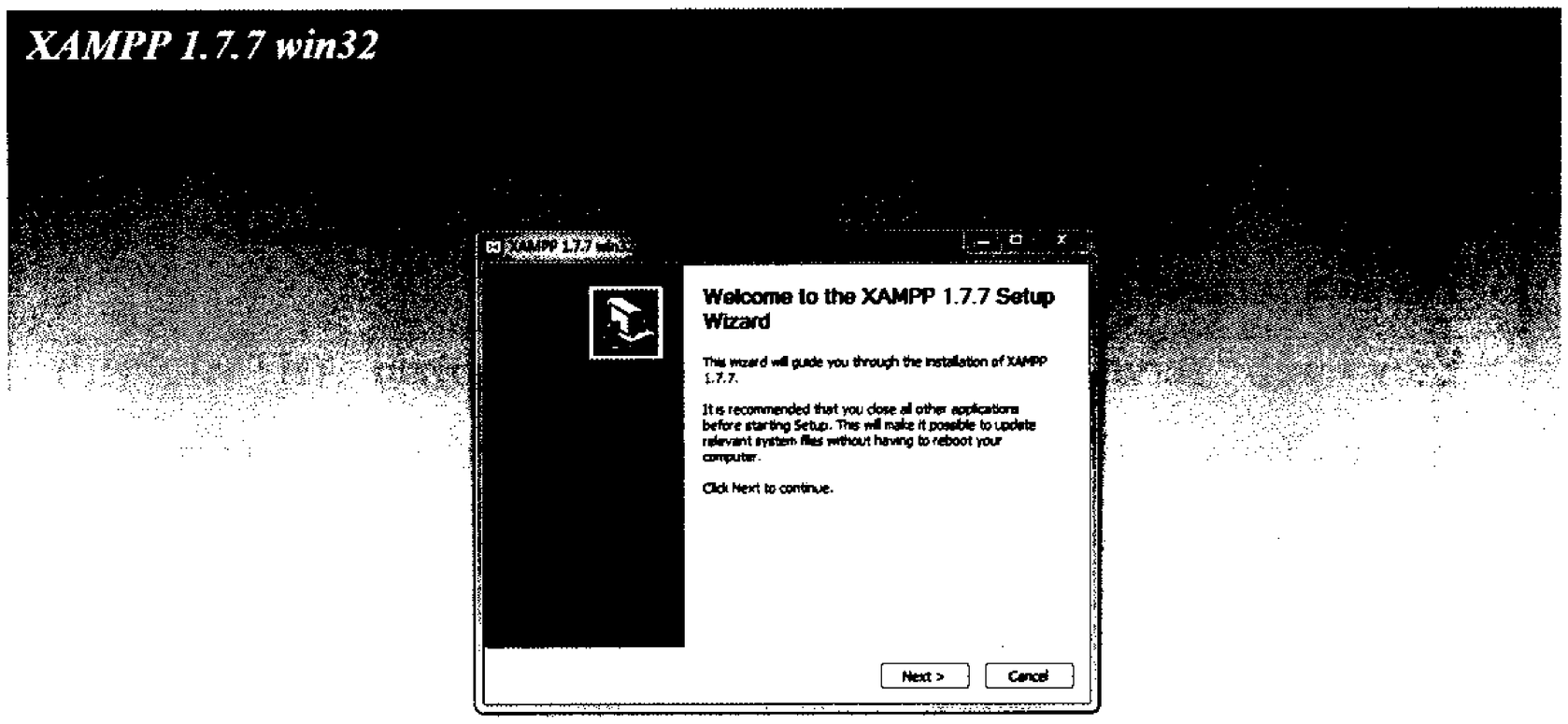
4. *Τον εξυπηρετητή βάσεων δεδομένων MySQL 5.5.16*

2.10.1 Εγκατάσταση xampp

Εκτελούμε το αρχείο XAMPP 1.7.7 και ακολουθούμε τα βήματα εγκατάστασης όπως φαίνονται στις παρακάτω εικόνες.

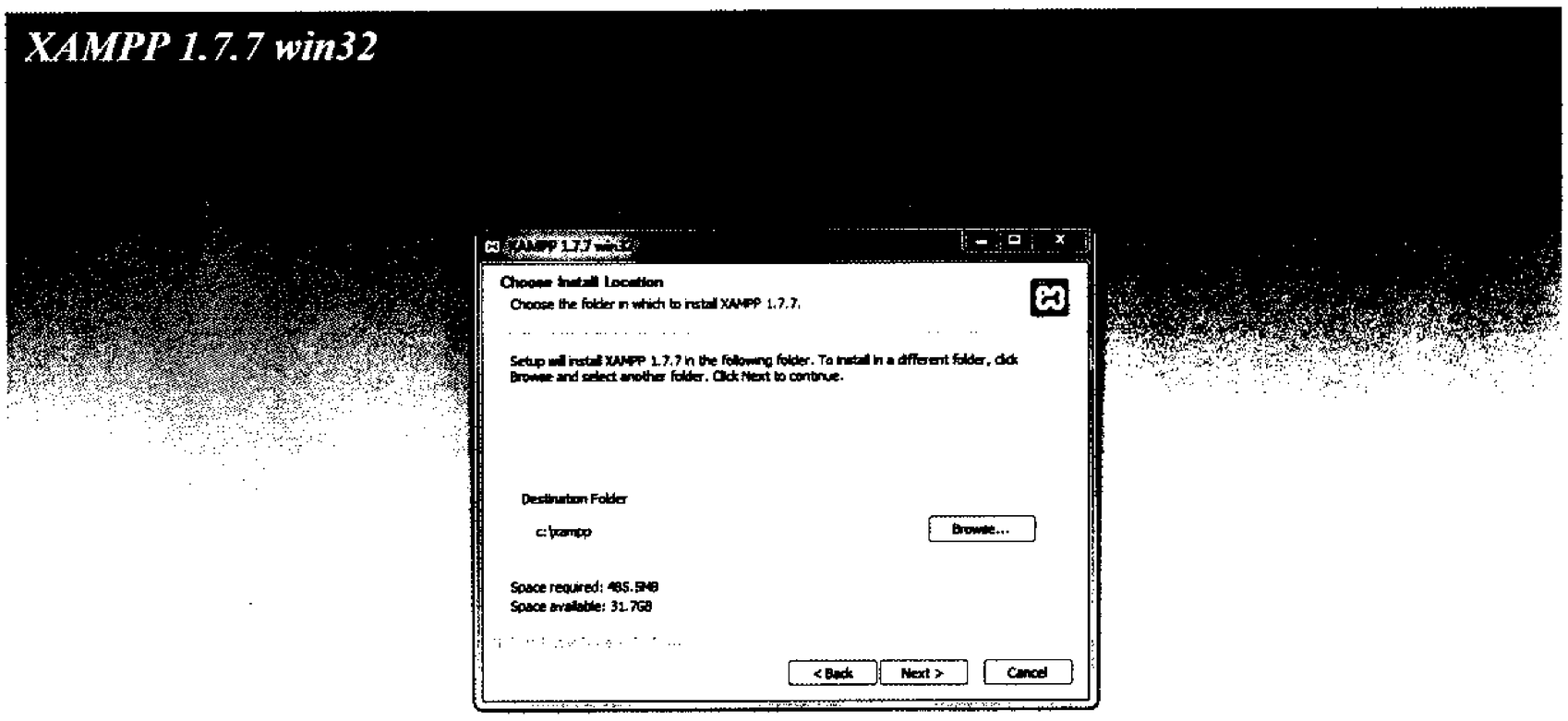
Εικόνα 2.2: *Xampp(Επιλογή γλώσσας)*

Πατώντας *OK*, εμφανίζεται η οθόνη καλωσορίσματος (*Εικόνα 2.3*)



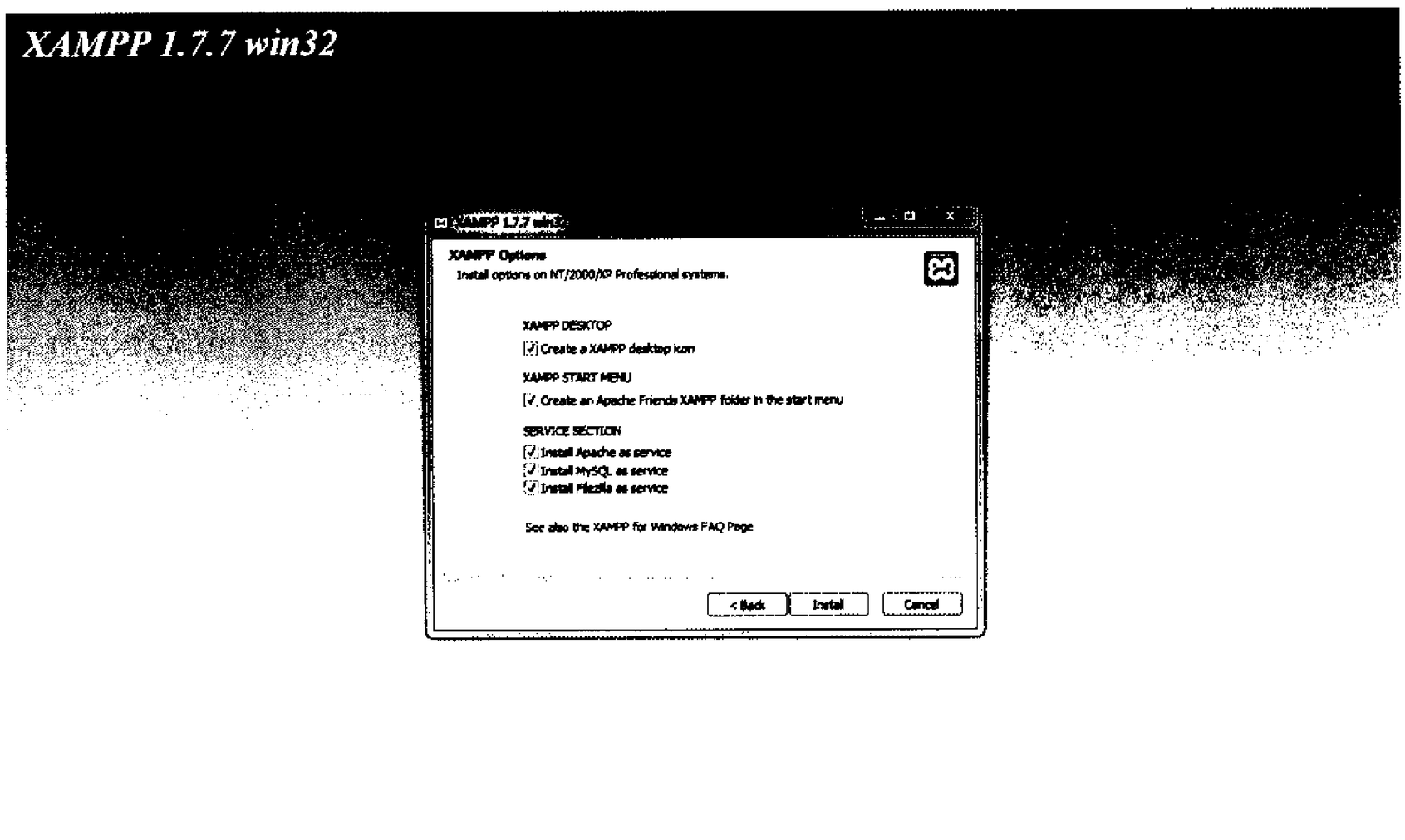
Εικόνα 2.3: *Xampp (Καλωσόρισμα)*

Στην επόμενη οθόνη ζητείται η διαδρομή στην οποία θα εγκατασταθεί η εφαρμογή (*Εικόνα 2.4*). Διατηρούμε τη θέση που προτείνει το πρόγραμμα, κάτω από αυτή τη θέση θα εγκατασταθούν όλες οι εφαρμογές αλλά και ο φάκελος στον οποίο θα ανεβαίνουν οι εφαρμογές.



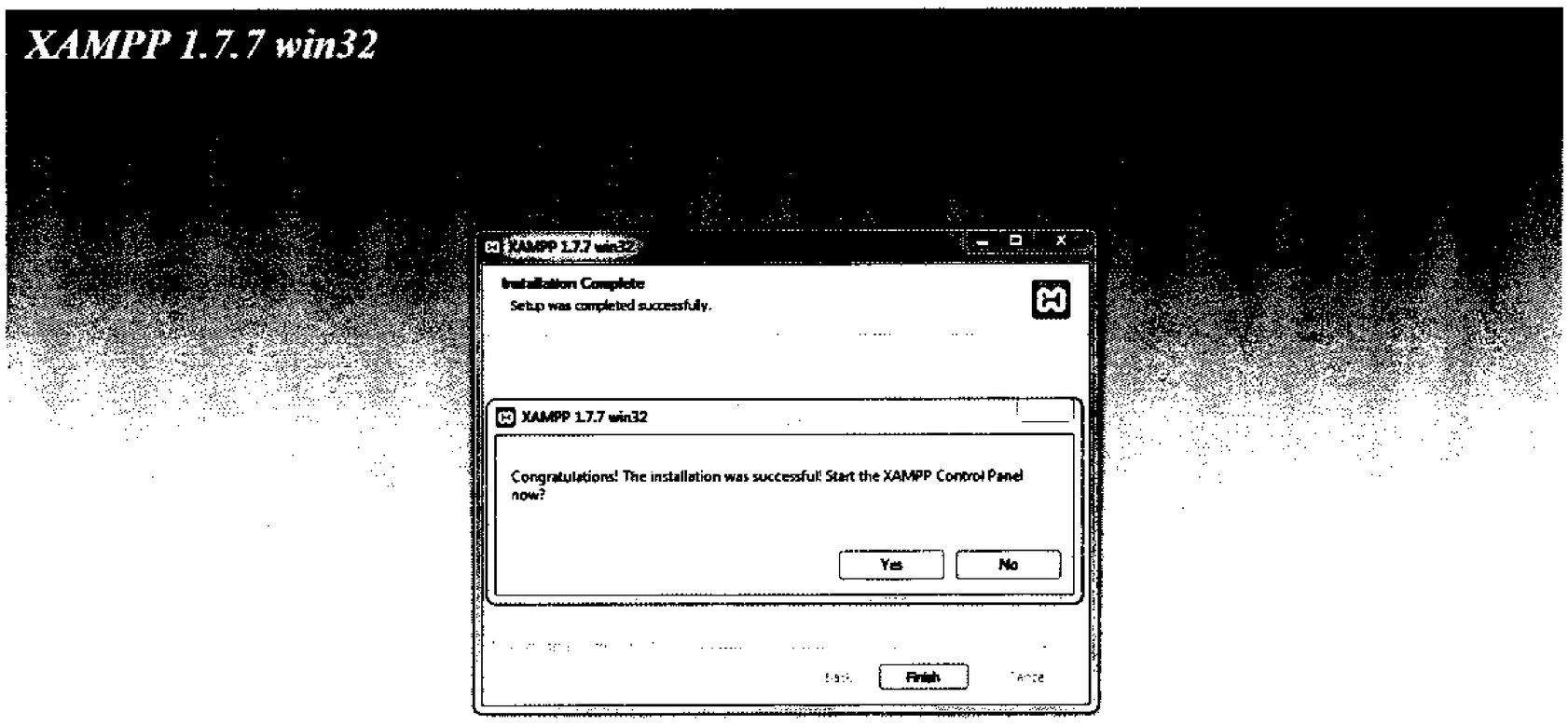
Εικόνα 2.4: *Xampp(θέση αποθήκευσης)*

Πατώντας *Next* εμφανίζονται οι επιλογές σχετικά με τα πρόσθετα πακέτα εγκατάστασης (*Εικόνα 2.5*).



Εικόνα 2.5: *Xampp (Επιλογή Installation)*

Πατώντας *Install* ξεκινάει η εγκατάσταση των εφαρμογών, η οποία διαρκεί μερικά λεπτά. Η εγκατάσταση ολοκληρώνεται με επιτυχία (*Εικόνα 2.6*).

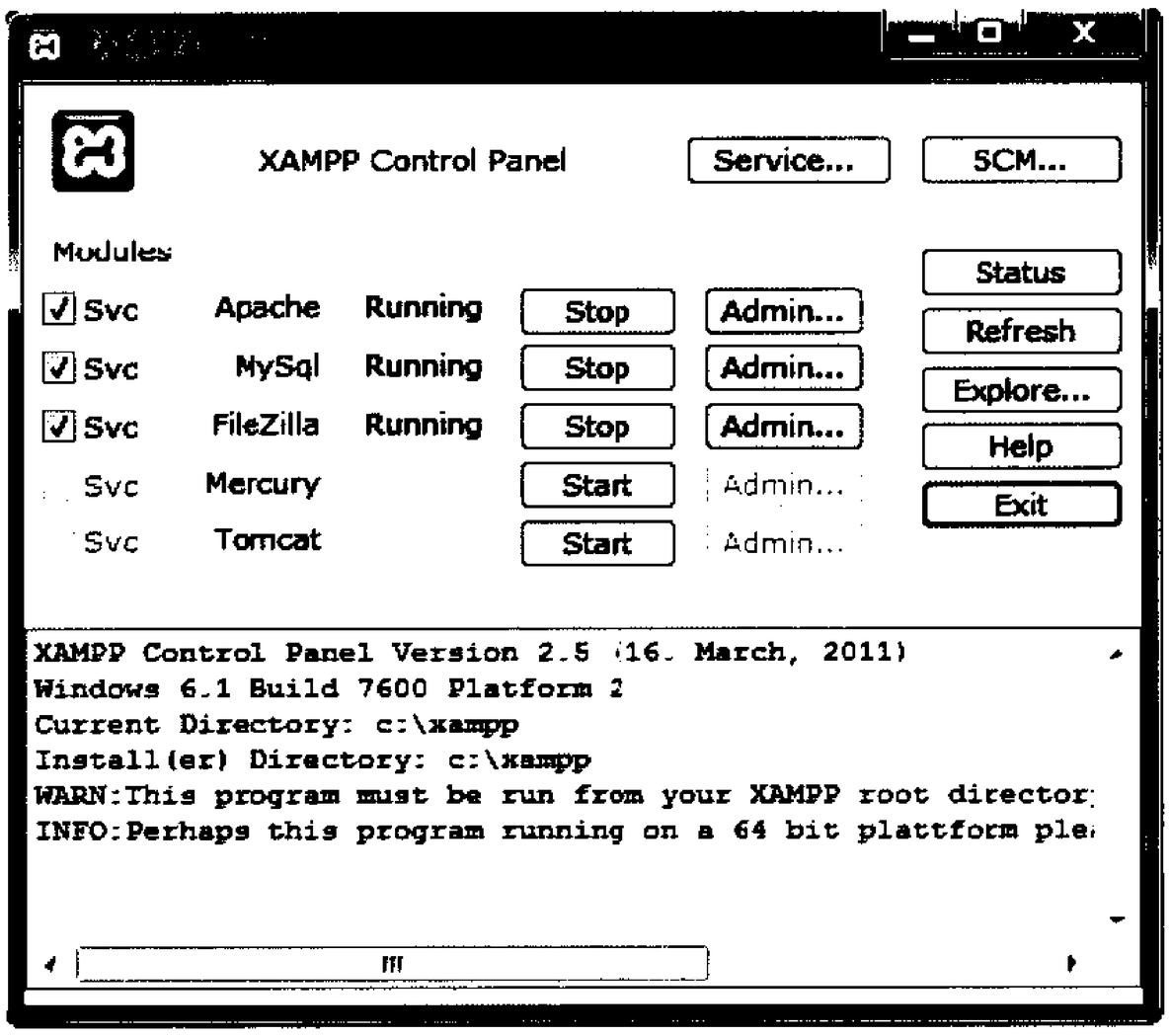


Εικόνα 2.6: *Xampp (Finish)*

Σημαντικό στοιχείο είναι ότι μαζί με το xampp εγκαθίσταται αυτόματα και η εφαρμογή *phpMyAdmin* για τη διαχείριση βάσεων δεδομένων MySQL.

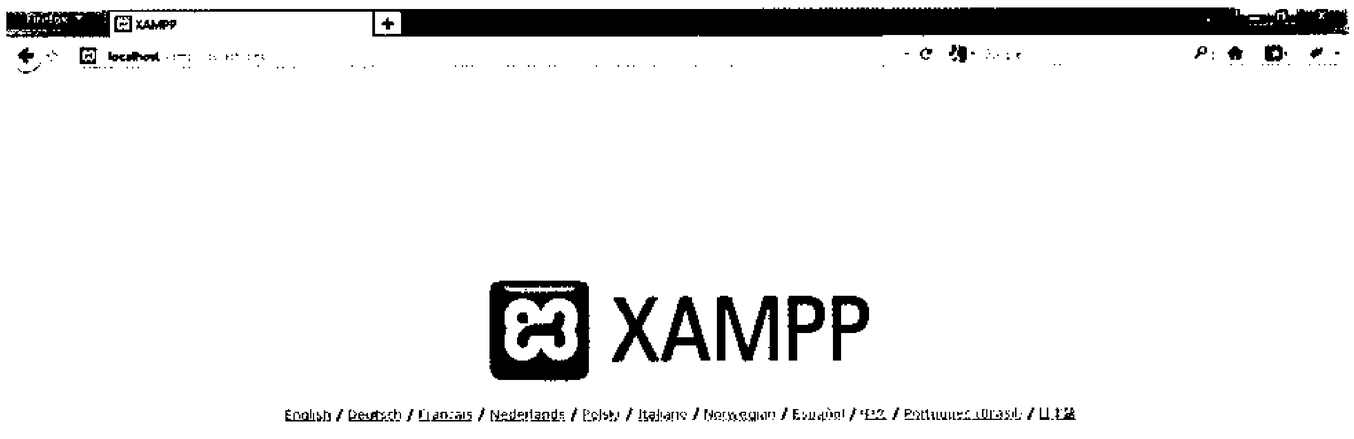
2.10.2 Ρυθμίσεις xampp

Για την εκκίνηση της λειτουργίας του προγράμματος εκτελούμε το αρχείο *xampp-control.exe* (*Εικόνα 2.7*).



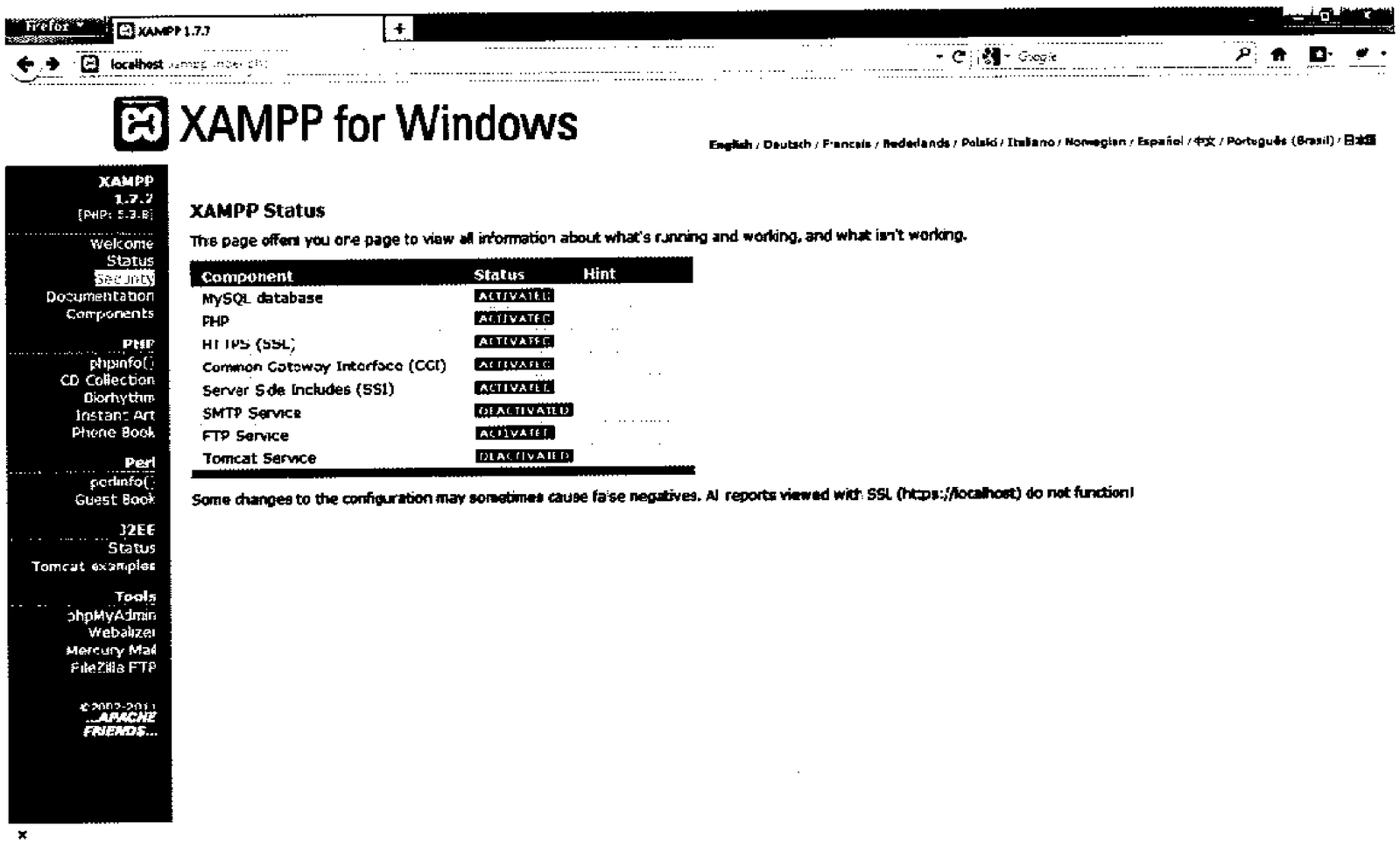
Εικόνα 2.7: *Xampp (Running)*

Στο επόμενο βήμα ανοίγουμε τον φυλλομετρητή (*Εικόνα 2.8*) και μεταφερόμαστε στην ιστοσελίδα *http://localhost* ή στο διαχειριστικό κομμάτι του προγράμματος:



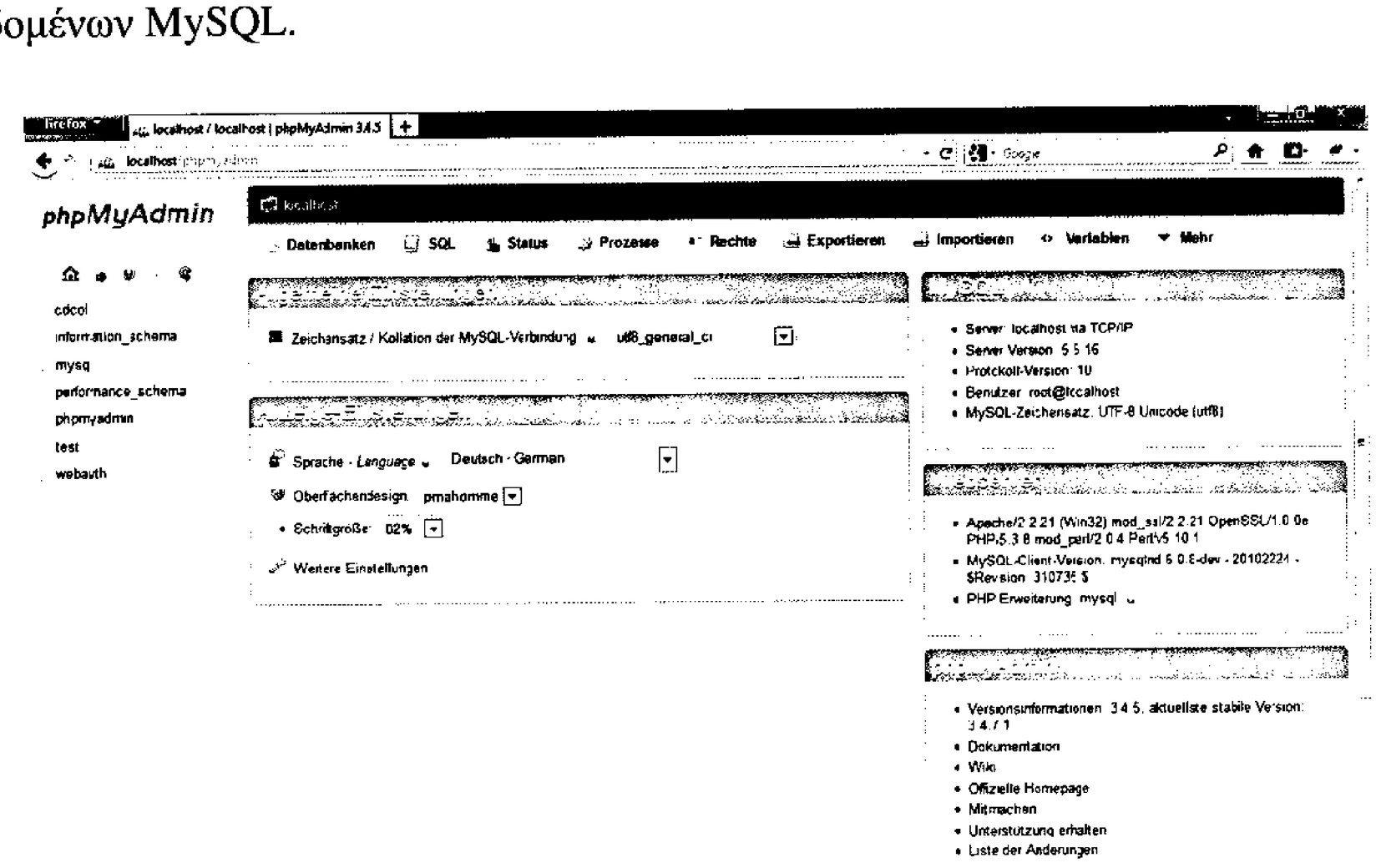
Εικόνα 2.8: *Xampp(http://localhost)*

Επιλέγουμε την επιθυμητή γλώσσα και ελέγχουμε εάν στο *xampp status* οι υπηρεσίες είναι *activated* όπως φαίνεται στη παρακάτω εικόνα (*Εικόνα 2.9*),



Εικόνα 2.9: *Xampp(Activate)*

Έπειτα ελέγχουμε τα περιεχόμενα του *phpMyAdmin* για τη διαχείριση βάσεων δεδομένων MySQL.



Εικόνα 2.10: *Xampp(phpMyAdmin)*

Η εγκατάσταση ολοκληρώνεται επιτυχώς.

2.10.3 PhpMyAdmin

Το *PhpMyAdmin* είναι ένα ελεύθερο λογισμικό ανοιχτού κώδικα με το οποίο ο χρήστης/προγραμματιστής έχει την δυνατότητα να διαχειρίζεται την MySQL στο διαδίκτυο. Μπορεί να χειρίζεται πλήρως βάσεις δεδομένων, πίνακες, πεδία πινάκων αλλά και ολόκληρο τον MySQL Server. Υποστηρίζει 47 γλώσσες μεταξύ των οποίων και τα Ελληνικά.

Οι δυνατότητες του PhpMyAdmin είναι οι εξής:

- Να δημιουργεί και να διαγράφει βάσεις δεδομένων,
- Να δημιουργεί, τροποποιεί, διαγράφει, αντιγράφει και μετονομάζει πίνακες σε μια υπάρχουσα βάση,
- Να κάνει την απαραίτητη συντήρηση της βάσης,
- Να προσθέτει, διαγράφει και τροποποιεί τα πεδία των πινάκων,
- Να εκτελεί SQL ερωτήματα προς την βάση,
- Να δημιουργεί αυτόματα πολύπλοκα ερωτήματα χρησιμοποιώντας το QBE (Query-by-example), που είναι σε στυλ συμπλήρωσης φορμών,
- Να κάνει εύρεση δεδομένων, γενικά στη βάση ή ειδικότερα σε κάποια υποδιαίρεσή της (πίνακα ή πεδίο),
- Να διαχειρίζεται κλειδιά των διαφόρων πεδίων των πινάκων της βάσης,
- Να φορτώνει αρχεία κειμένου σε πίνακες της βάσης,
- Να εισάγει και να εξάγει δεδομένα σε μορφή CVS, Latex και XML,
- Να διαχειρίζεται πολλαπλούς διακομιστές,
- Να διαχειρίζεται τους χρήστες MySQL και τα δικαιώματά τους,
- Να ελέγχει την αναφορική ακεραιότητα των δεδομένων των πινάκων της βάσης,
- Να δημιουργεί την γραφική αναπαράσταση της βάσης δεδομένων σε μορφή PDF.

phpMyAdmin

Κεφάλαιο 3ο

Υλοποίηση Βάσης Δεδομένων

3.1 Εισαγωγή στην ανάλυση των απαιτήσεων

Η ανάλυση απαιτήσεων είναι μια διαδικασία κατάρτισης μιας λίστας, όπου αναφέρονται οι προδιαγραφές που πρέπει να πληροί η εφαρμογή που πρόκειται να δημιουργηθεί. Οι προδιαγραφές που προσδιορίζονται μπορεί να είναι τεχνολογικές, επιχειρηματικές, λειτουργικές, να σχετίζονται με τη μορφή, το κόστος, τη διάρκεια ή ακόμη και το χρόνο απόσβεσης.

Η λίστα που καταρτίζεται χρησιμεύει τόσο σε αυτούς που θα αναπτύξουν την εφαρμογή όσο και σ' εκείνους που θα τη χρησιμοποιήσουν. Η ανάλυση απαιτήσεων είναι μία συνεργατική διαδικασία όπου διαφορετικά άτομα με διαφορετικές αφετηρίες συναντιούνται, αλληλεπιδρούν, διαφωνούν και συμφωνούν γύρω από το ίδιο αντικείμενο, *το έργο*.

3.2 Ανάλυση απαιτήσεων της ιστοσελίδας

- Προσθήκη - Επεξεργασία καταστημάτων

  Ο χρήστης έχει τη δυνατότητα εισαγωγής νέου καταστήματος καταχωρώντας όλα τα απαραίτητα στοιχεία που χρειάζεται κάθε κατάστημα (ημέρες λειτουργίας, διεύθυνση, καλλιτέχνες, φωτογραφίες, page title κλπ.), τα οποία αποθηκεύονται στη Βάση Δεδομένων και στον server.
- Προσθήκη - Επεξεργασία Χρήστη

  Ο χρήστης μπορεί να εισάγει νέους χρήστες για την διαχείριση του CMS απονέμοντάς τους δικαιώματα Administrator ή User. Τα στοιχεία του κάθε χρήστη αποθηκεύονται στην Βάση Δεδομένων. Ο κάθε χρήστης έχει τη δυνατότητα να επεξεργάζεται τα στοιχεία του λογαριασμού του.
- Προσθήκη - Επεξεργασία Media

Η επιλογή αυτή θα δίνει τη δυνατότητα στο χρήστη να διαχειριστεί τις διαφημίσεις (ανά κατηγορία) που θα προβάλλονται μέσω της ιστοσελίδας, με απώτερο σκοπό την προσθήκη, επεξεργασία και διαγραφή τους. Επιπρόσθετη επιλογή αποτελεί η διεργασία που δίνει την δυνατότητα στον χρήστη να επιλέξει την διαφήμιση που θα είναι ενεργοποιημένη ή όχι. Τα χαρακτηριστικά κάθε διαφήμισης θα καταγράφονται στη Βάση Δεδομένων.

- Προσθήκη Καθημερινών Προσφορών - News

  Αποτελεί την επιλογή όπου ο χρήστης θα μπορεί να καταχωρήσει, να απενεργοποιήσει και να διαγράφει μια προσφορά. Οι προφορές θα καταχωρούνται στη Βάση Δεδομένων και θα προβάλλονται στην ιστοσελίδα με απώτερο σκοπό την ενημέρωση των επισκεπτών.

- Προσθήκη - Αποστολή Ειδοποιήσεων (Notification)

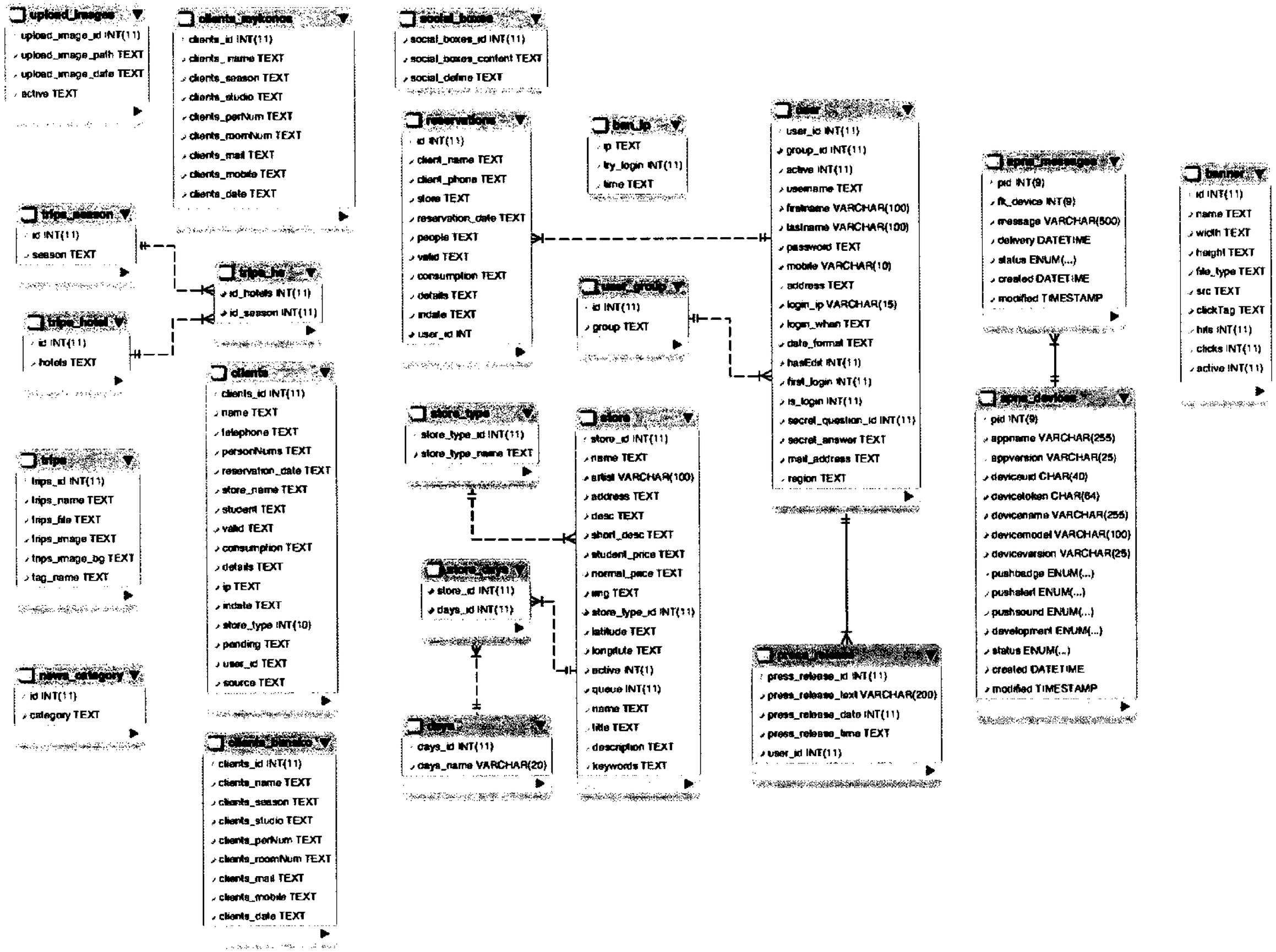
  Η κατηγορία των ειδοποιήσεων αποτελεί την διαδικασία κατά την οποία ο χρήστης θα μπορεί να ενημερώνει του χρήστες των smartphones για τις προφορές της ημέρας. Η διαδικασία θα περιλαμβάνει την καταχώρηση της προσφοράς στη Βάση Δεδομένων και την προγραμματισμένη αποστολή της στις εφαρμογές.

- Προβολή Στατιστικών

  Η Προβολή των στατιστικών αποτελεί την επιλογή όπου ο χρήστης θα μπορεί να ενημερώνεται σχετικά με τα καθημερινά και συνολικά downloads των εφαρμογών iPhone και Android.

- Προβολή - Επεξεργασία Κρατήσεων

  Ο χρήστης θα μπορεί να ενημερωθεί για τις κρατήσεις που έχουν πραγματοποιηθεί και θα έχουν καταχωρηθεί στη Βάση Δεδομένων. Η διαδικασία θα παρέχει τη δυνατότητα προσθήκης, επεξεργασίας και διαγραφής των κρατήσεων.


upload_images
upload_image_id INT(11)
upload_image_path TEXT
upload_image_date TEXT
active TEXT
clients_mykonos
clients_id INT(11)
clients_ name TEXT
clients_season TEXT
clients_studio TEXT
clients_perNum TEXT
clients_roomNum TEXT
clients_mail TEXT
clients_mobile TEXT
clients_date TEXT
social_boxes
social_boxes_id INT(11)
social_boxes_content TEXT
social_define TEXT
reservations
id INT(11)
client_name TEXT
client_phone TEXT
store TEXT
reservation_date TEXT
people TEXT
valid TEXT
consumption TEXT
details TEXT
indate TEXT
user_id INT
ban_ip
ip TEXT
try_login INT(11)
time TEXT
user
user_id INT(11)
group_id INT(11)
active INT(11)
username TEXT
firstname VARCHAR(100)
lastname VARCHAR(100)
password TEXT
mobile VARCHAR(10)
address TEXT
login_ip VARCHAR(15)
login_when TEXT
date_format TEXT
hasEdit INT(11)
first_login INT(11)
is_login INT(11)
secret_question_id INT(11)
secret_answer TEXT
mail_address TEXT
region TEXT
user_group
id INT(11)
group TEXT
trips_season
id INT(11)
season TEXT
trips_ho
id_hotels INT(11)
id_season INT(11)
trips_hotel
id INT(11)
hotels TEXT
clients
clients_id INT(11)
name TEXT
telephone TEXT
personNums TEXT
reservation_date TEXT
store_name TEXT
student TEXT
valid TEXT
consumption TEXT
details TEXT
ip TEXT
indate TEXT
store_type INT(10)
pending TEXT
user_id TEXT
source TEXT
store_type
store_type_id INT(11)
store_type_name TEXT
store_city
store_id INT(11)
days_id INT(11)
days
days_id INT(11)
days_name VARCHAR(20)
store
store_id INT(11)
name TEXT
artist VARCHAR(100)
address TEXT
desc TEXT
short_desc TEXT
student_price TEXT
normal_pace TEXT
img TEXT
store_type_id INT(11)
latitude TEXT
longitute TEXT
active INT(1)
queue INT(11)
name TEXT
title TEXT
description TEXT
keywords TEXT
trips
trips_id INT(11)
trips_name TEXT
trips_file TEXT
trips_image TEXT
trips_image_bg TEXT
tag_name TEXT
news_category
id INT(11)
category TEXT
clients_mykonos
clients_id INT(11)
clients_name TEXT
clients_season TEXT
clients_studio TEXT
clients_perNum TEXT
clients_roomNum TEXT
clients_mail TEXT
clients_mobile TEXT
clients_date TEXT
press_release
press_release_id INT(11)
press_release_text VARCHAR(200)
press_release_date INT(11)
press_release_time TEXT
user_id INT(11)
apns_messages
pid INT(9)
fk_device INT(9)
message VARCHAR(500)
delivery DATETIME
status ENUM(...)
created DATETIME
modified TIMESTAMP
apns_devices
pid INT(9)
appname VARCHAR(255)
appversion VARCHAR(25)
deviceuid CHAR(40)
devicetoken CHAR(64)
devicename VARCHAR(255)
devicemodel VARCHAR(100)
deviceversion VARCHAR(25)
pushbadge ENUM(...)
pushalert ENUM(...)
pushsound ENUM(...)
development ENUM(...)
status ENUM(...)
created DATETIME
modified TIMESTAMP
banner
id INT(11)
name TEXT
width TEXT
height TEXT
file_type TEXT
src TEXT
clickTag TEXT
hits INT(11)
clicks INT(11)
active INT(11)

3.4 Περιγραφή Βάσης Δεδομένων της Εφαρμογής

Για τις ανάγκες της διαδικτυακής εφαρμογής δημιουργήθηκε μια Βάση Δεδομένων στην οποία αποθηκεύονται οι απαραίτητες πληροφορίες. Με την καταχώρηση των δεδομένων επιτυγχάνεται δυναμικό περιεχόμενο. Με τον τρόπο αυτό γίνεται πιο εύκολη η διαχείριση και η προβολή του περιεχομένου της εφαρμογής.

Η Βάση Δεδομένων που δημιουργήθηκε ονομάζεται **kratisi** και αποτελείται από 23 πίνακες. Οι πίνακες αυτοί προέκυψαν από την ανάλυση απαιτήσεων του συστήματος και παρουσιάζονται συνοπτικά παρακάτω.

3.5 Περιγραφή Πινάκων της Βάσης Δεδομένων

Πίνακας «user»

Στον πίνακα user αποθηκεύονται οι χρήστες που έχουν πρόσβαση στο περιβάλλον διαχείρισης περιεχομένου (CMS). Το πεδίο password έχει τον κωδικό του χρήστη σε κρυπτογραφημένη μορφή κάνοντας χρήση του MD5 αλγόριθμου.

Πίνακας «user_group»

Σε αυτόν τον πίνακα ορίζονται ομάδες χρηστών. Κάθε ομάδα έχει διαφορετικά επίπεδα πρόσβασης και επεξεργασίας. Κάθε χρήστης μπαίνει σε μια προκαθορισμένη ομάδα.

Πίνακας «store»

Στον πίνακα store αποθηκεύονται τα κέντρα διασκέδασης. Ο πίνακας περιέχει γενικές πληροφορίες κάθε καταστήματος όπως τοποθεσία, κόστος, καλλιτέχνες κ.α. Αποθηκεύει επίσης πληροφορίες που βοηθούν στο Search Engine Optimization.

Πίνακας «store_type»

Ορίζονται οι κατηγορίες των κέντρων διασκέδασης.

Πίνακας «store_days»

Αποθηκεύονται οι διαθέσιμες ημέρες για κάθε κέντρο διασκέδασης

Πίνακας «days»

Αποθηκεύονται τα λεκτικά των εφτά ημερών της εβδομάδας μαζί με ένα id. Οι τιμές των ids είναι από 0-6 για τις ημέρες από Κυριακή-Σάββατο.

Πίνακας «clients»

Κάθε κράτηση που γίνεται αποθηκεύεται σε αυτόν τον πίνακα. Μερικά από τα στοιχεία που εισάγονται είναι το όνομα στο οποίο θα γίνει η κράτηση, ο αριθμός των ατόμων που θα παρευρεθούν, η ημερομηνία κράτησης κ.α

Πίνακας «press_release»

Στον πίνακα press_release εισάγονται οι ειδοποιήσεις και τα δελτία τύπου που αποστέλλονται στις συσκευές iPhone μέσω της τεχνολογία Push Notification.

Πίνακας «apns_devices»

Στον πίνακα apns_devices αποθηκεύονται όλες οι απαραίτητες πληροφορίες για τις συσκευές που έχουν εγκαταστήσει την iPhone εφαρμογή. Οι πληροφορίες αυτές είναι απαραίτητες για την αποστολή των notifications στην εφαρμογή. Περιλαμβάνει πεδία όπως το όνομα της συσκευής, τον μοναδικό κωδικό της συσκευής, την έκδοση του λειτουργικού κάθε συσκευής κ.α

Πίνακας «apns_messages»

Περιέχει πληροφορίες που σχετίζονται με την αποστολή των notifications στις iPhone συσκευές, όπως αναφορά παράδοσης, ώρα παράδοσης κ.α

Πίνακας «banner»

Στον πίνακα banner καταχωρούνται οι παράμετροι και οι ρυθμίσεις των διαφημιστικών banner που εμφανίζονται στην ιστοσελίδα όπως μέγεθος, τύπος, clicks πάνω στο banner κ.α

Πίνακας «ban_ip»

Στον πίνακα ban_ip αποθηκεύονται οι ip διευθύνσεις που έχουν μπλοκαριστεί, μετά από κάποιο αριθμό ανεπιτυχών προσπαθειών εισόδου στο Σύστημα Διαχείρισης Περιεχομένου (CMS).

Πίνακας «upload_images»

Ο πίνακας upload_images υποστηρίζει το module που έχει αναπτυχθεί στην αρχική σελίδα το οποίο προβάλλει φωτογραφίες που ανεβάζουν οι επισκέπτες.

Πίνακας «trips»

Στον πίνακα trips αποθηκεύονται οι εκδρομές που προβάλλονται στην σελίδα «trips» και περιέχονται φωτογραφίες για το background της σελίδας και κάποιες αλφαριθμητικές τιμές.

Πίνακας «trips_hotel»

Περιλαμβάνει τα διαθέσιμα ξενοδοχεία για τις εκδρομές που διοργανώνονται.

Πίνακας «trips_season»

Στον πίνακα αυτόν, αποθηκεύονται οι συγκεκριμένες ημερομηνίες για τα ξενοδοχεία

Πίνακας «trips_hs»

Συνδέει τους δύο προηγούμενους πίνακες. Δηλαδή τα ξενοδοχεία με τις αντίστοιχες διαθέσιμες ημερομηνίες για κάθε ξενοδοχείο.

Πίνακας «clients_trips»

Αποθηκεύονται οι πελάτες ταξιδιών. Περιλαμβάνονται πληροφορίες όπως όνομα, αριθμός ατόμων, ξενοδοχείο, ημερομηνίες κράτησης κ.α

Πίνακας «news_category»

Υποστηρίζει τα δυο module που έχουν αναπτυχθεί στην αρχική σελίδα για την προβολή των offers και των gossip. Περιέχει το id και το λεκτικό της κάθε κατηγορίας.

Πίνακας «news»

Υποστηρίζει τα δυο module που έχουν αναπτυχθεί στην αρχική σελίδα για την προβολή των offers και των gossip. Αποθηκεύει το περιεχόμενο των offers και gossip.

Πίνακας «social_boxes»

Περιλαμβάνει το περιεχόμενο των module που έχουν αναπτυχθεί για την προβολή του facebook, newsletter, photo gallery.

Πίνακας «newsletterer»

Στον πίνακα newsletter αποθηκεύονται τα emails από το module που υπάρχει στην αρχική σελίδα και προτρέπει τον χρήστη να κάνει εγγραφή στην υπηρεσία newsletter.

Κεφάλαιο 4ο

Παρουσίαση Ιστοσελίδας

Στο κεφάλαιο αυτό θα γίνει μία αναλυτική παρουσίαση της ιστοσελίδας από τη σκοπιά του επισκέπτη, ενώ παράλληλα με κάθε βήμα θα αναλύονται οι εσωτερικές λειτουργίες του συστήματος και θα αιτιολογούνται οι επιλογές που έχουν γίνει.

4.1 Γενική περιγραφή του Web-Site

Κάθε ιστοσελίδα απαρτίζεται από τρία βασικά μέρη: header, κύριο μέρος ιστοσελίδας και footer.

Το «header» της ιστοσελίδας (*Εικόνα 4.1 περιοχή 1*) αποτελείται από:

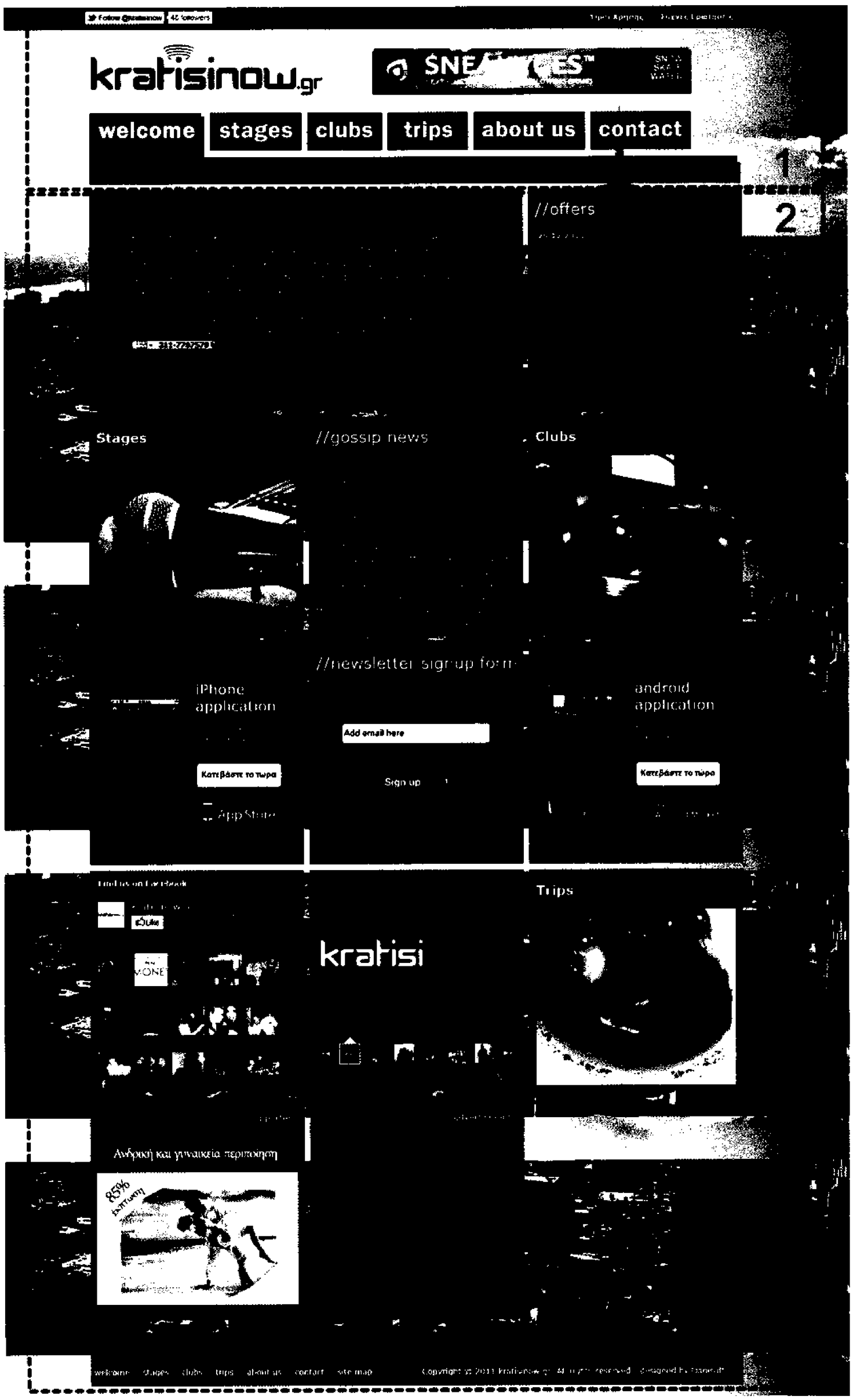
- ένα οριζόντιο menu που περιέχει το twitter follow και δύο συνδέσμους που παραπέμπουν στους όρους χρήσης και στις συχνές ερωτήσεις,
- το logo διάστασης 347x80,
- έναν χώρο φιλοξενίας και προβολής διαφημιστικών banner διάστασης 468x60,
- το κύριο menu του site, που επιτρέπει στον χρήστη να περιηγηθεί εύκολα και γρήγορα στις σελίδες welcome, stages, clubs, trips, about us, contact. Κάθε σελίδα αποτελεί ένα ξεχωριστό PHP αρχείο με το δικό της CSS, υπεύθυνο για την διαμόρφωση και μορφοποίησή της.

Το «κύριο μέρος της ιστοσελίδας» (*Εικόνα 4.1 περιοχή 2*) αποτελείται από:

- ένα κείμενο καλωσορίσματος για τον χρήστη,
- τετράγωνα κάθε ένα εκ των οποίων δίνει την δυνατότητα στον χρήστη είτε να αλληλεπιδράσει με το site (newsletter form, photo album, facebook like, iphone application, android application), είτε να περιηγηθεί σε μία από τις βασικές σελίδες του site (stages, clubs, trips) , είτε να ενημερωθεί για νέα ή τρέχουσες προσφορές (gossip news, offers).

Το «footer» της ιστοσελίδας (*Εικόνα 4.1 περιοχή 2*) αποτελείται από:

- ένα σύντομο menu πλοήγησης προς τις υπόλοιπες σελίδες του site,
- ένα διακριτικό για την κατοχύρωση των πνευματικών δικαιωμάτων.



Εικόνα 4.1: *Αρχική Σελίδα*

4.2 Ανάλυση σελίδων

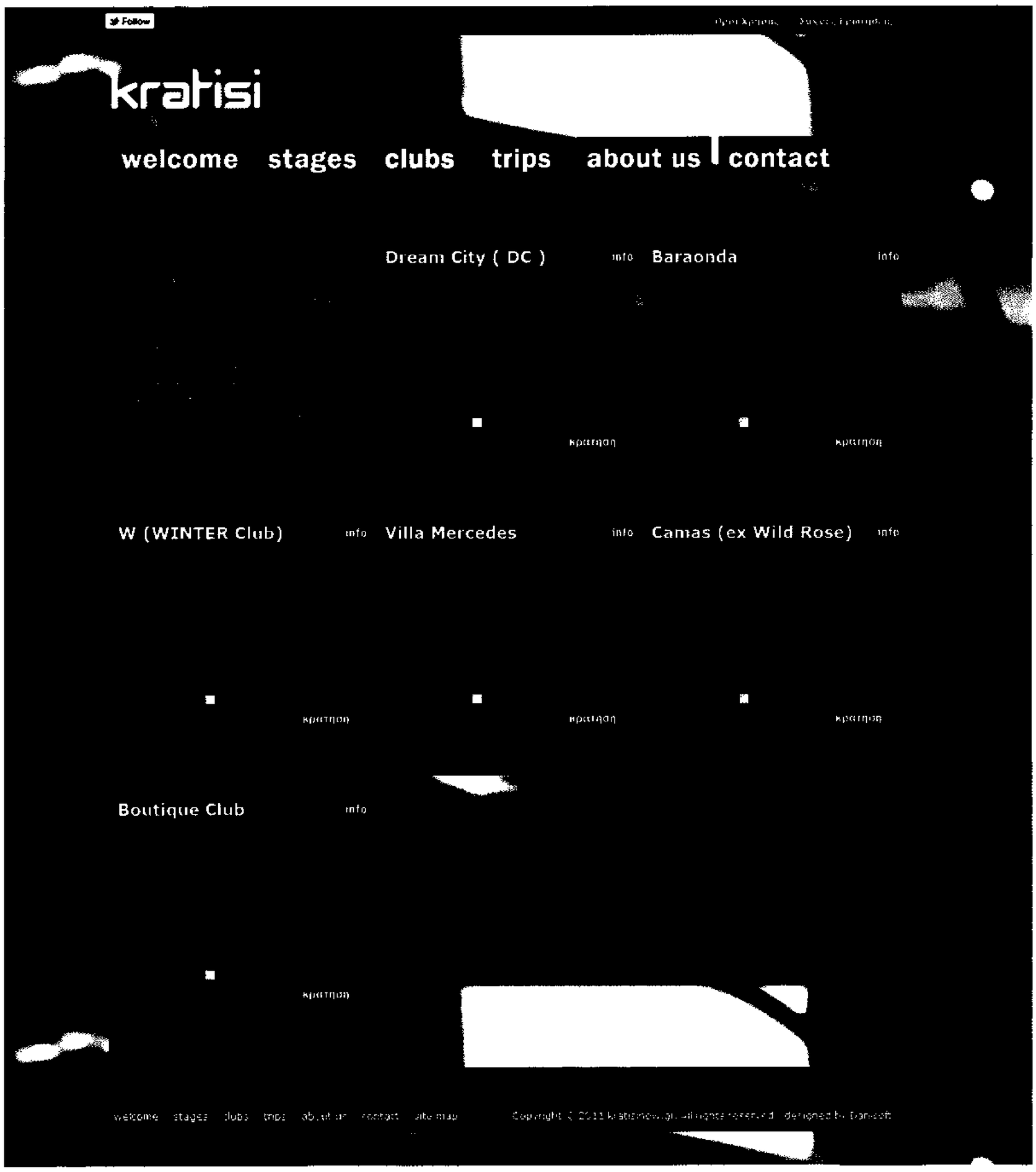
Η ενότητα αυτή περιλαμβάνει την περιγραφή των εσωτερικών σελίδων της ιστοσελίδας και τις τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση τους.

4.2.1 Clubs και Stages

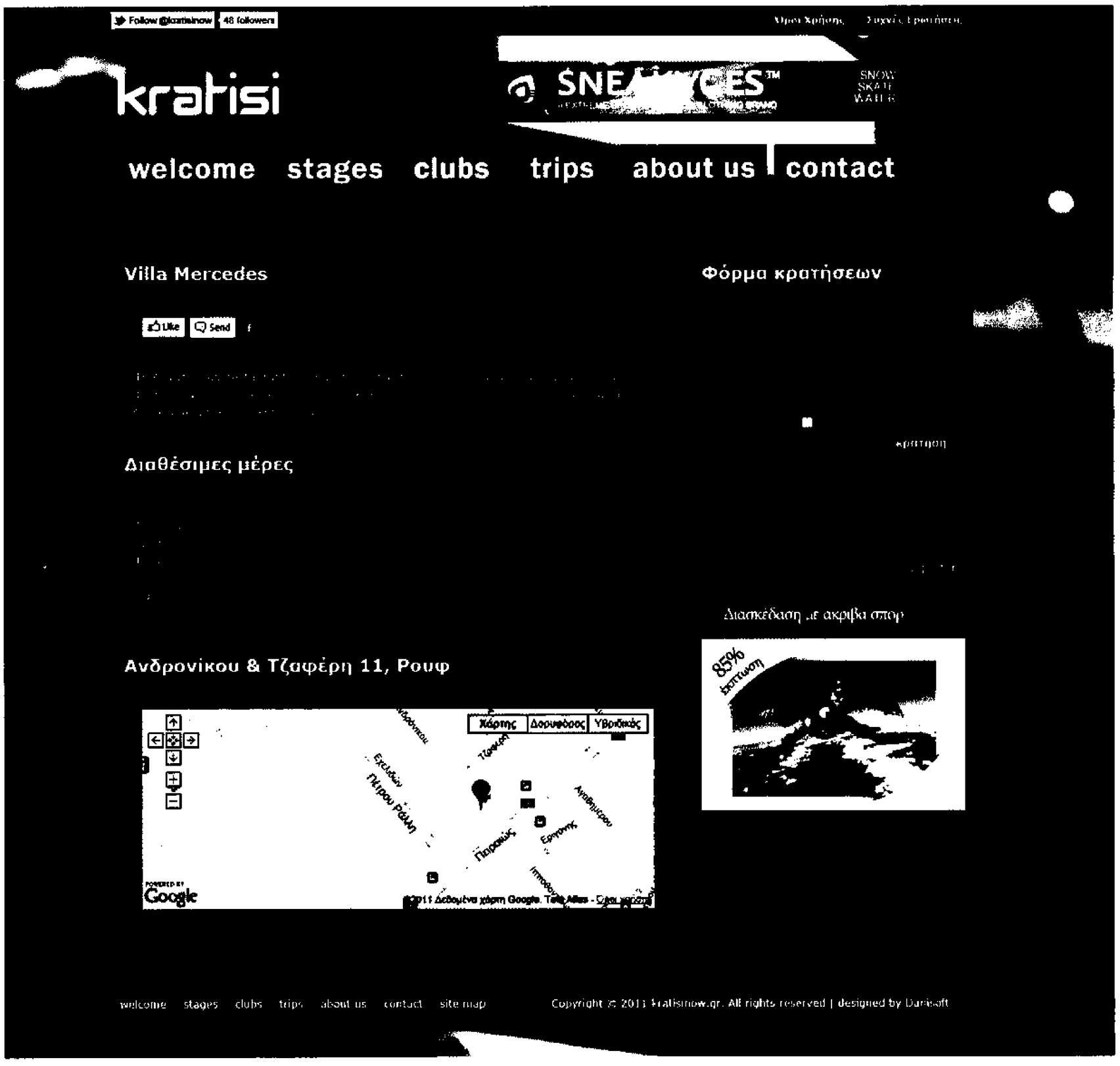
Η σελίδα με τα clubs ή stages αποτελείται από ξεχωριστά τετράγωνα με δομή δυσδιάστατου πίνακα (*Εικόνα 4.2*). Κάθε κελί του πίνακα είναι ένα τετράγωνο box που περιέχει:

- όνομα καταστήματος,
- ονόματα καλλιτεχνών,
- μία HTML φόρμα εκδήλωσης ενδιαφέροντος. Στην φόρμα αυτή ο χρήστης εισάγει τα προσωπικά του στοιχεία για την αποστολή της κράτησης. Τα στοιχεία αυτά είναι το ονοματεπώνυμο, τηλέφωνο, άτομα, ημερομηνία, επιλογή φοιτητικού και αποτελούν υποχρεωτικά πεδία για την ολοκλήρωση της διαδικασίας. Πιο συγκεκριμένα, για την επιλογή της ημερομηνίας γίνεται χρήση του component datePicker της βιβλιοθήκης JQuery. Για τον έλεγχο του τηλεφωνικού αριθμού η επιτρεπτή τιμή είναι της μορφής 69XXXXXXXX και για τον αριθμό ατόμων η επιτρεπτή τιμή είναι 2-30 άτομα.

Επίσης υπάρχει ένα κουμπί «info» το οποίο προβάλλει μια σύντομη περιγραφή του καταστήματος καθώς και το κόστος κράτησης ανά ημέρα. Η εναλλαγή των πληροφοριών γίνεται με το εφέ slideDown της JQuery βιβλιοθήκης. Τέλος, στο κάτω μέρος υπάρχει ένα κουμπί «read more» που παραπέμπει τον χρήστη σε μία άλλη σελίδα, η οποία ανάλογα με το μαγαζί περιέχει πληροφορίες όπως διαθέσιμες ημέρες και χάρτη με την διεύθυνση του καταστήματος (*Εικόνα 4.3*).



Εικόνα 4.2: *Φόρμες εκδήλωσης ενδιαφέροντος ανά κατηγορία*



Εικόνα 4.3: *Εσωτερική Σελίδα Καταστήματος*

4.2.2 Contact

Η συγκριμένη σελίδα αφορά στην επικοινωνία του χρήστη με τους διαχειριστές του site. Ο χρήστης συμπληρώνει τα προσωπικά του στοιχεία (ονοματεπώνυμο, email address, θέμα, μήνυμα) για να αποστείλει το μήνυμά του (*Εικόνα 4.4*).

Εικόνα 4.4: *Σελίδα Επικοινωνίας*

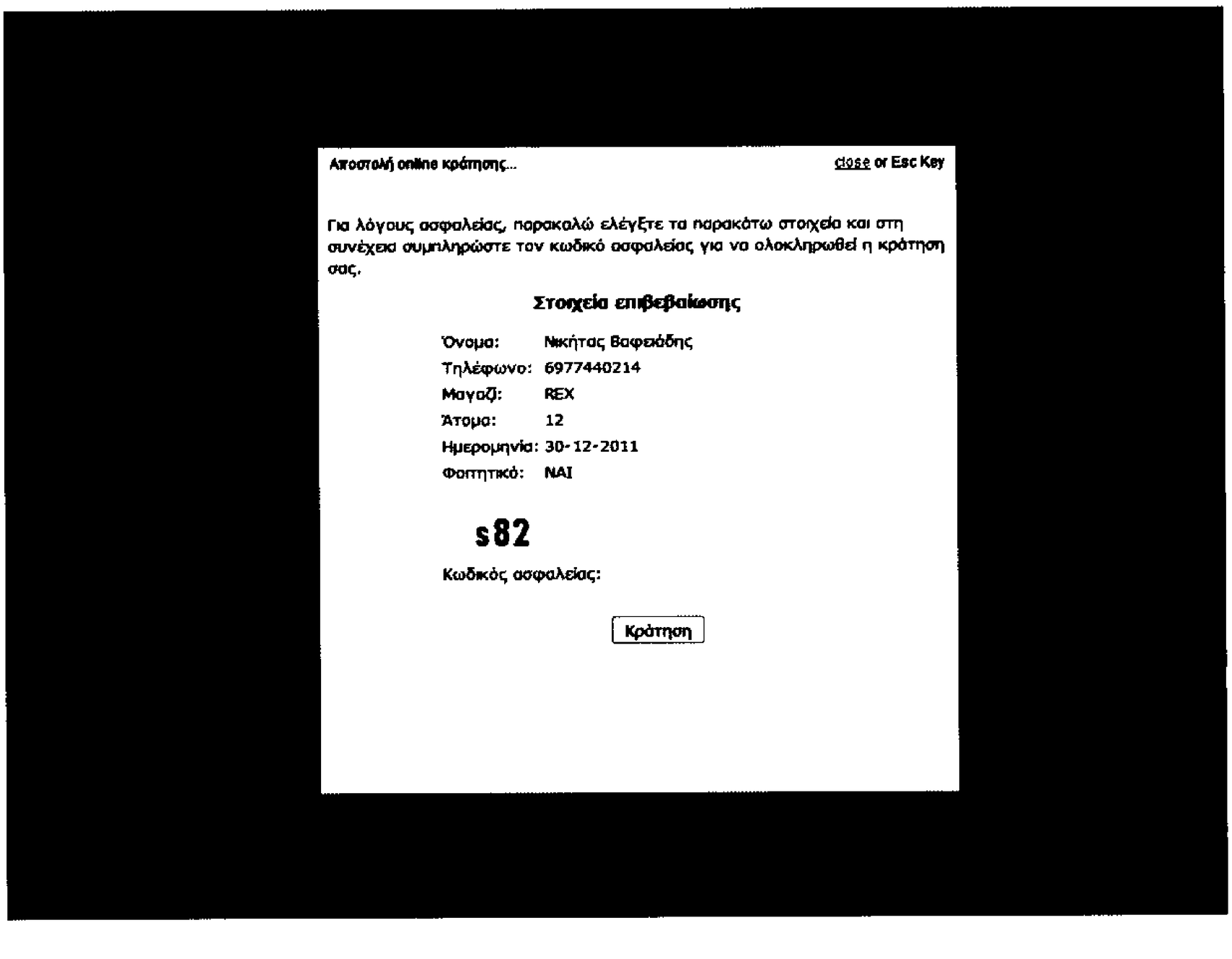
4.2.3 Sitemap

Πρόκειται για μια λίστα με τις υπάρχουσες σελίδες όλου του web site, προσβάσιμη από τους χρήστες και crawlers. Η λίστα αυτή οργανώνει τις σελίδες ανάλογα με την κατηγορία στην οποία ανήκουν ακολουθώντας ιεραρχική δομή. Αυτό βοηθάει τους επισκέπτες και τα search engine bots να ανιχνεύουν όλες τις σελίδες του site. Συνεπώς το sitemap είναι ιδιαίτερα κρίσιμο για τις αρχές του SEO (search engine optimization).

4.2.4 Περιγραφή Κράτησης

Κατά την διαδικασία της κράτησης ο χρήστης καλείται να συμπληρώσει τα προσωπικά του στοιχεία. Εφόσον πραγματοποιηθεί με επιτυχία ο έλεγχος των στοιχείων, τότε συνεχίζει στο δεύτερο και τελικό βήμα.

Στο δεύτερο βήμα εμφανίζεται στον χρήστη ένα pop up παράθυρο προκειμένου να επιβεβαιώσει τα στοιχεία κράτησης που συμπλήρωσε στο προηγούμενο βήμα (*Εικόνα 4.5*). Για να ολοκληρωθεί αυτό θα πρέπει να συμπληρώσει έναν extra μηχανισμό ασφαλείας που ονομάζεται captcha - Completely Automated Public Turing test to tell Computers and Humans Apart, που πρακτικά περιγράφει ένα είδος αυτόματου τεστ με την δυνατότητα να ξεχωρίζει αν ο χρήστης είναι άνθρωπος ή μηχανή. Μετά την ολοκλήρωση της επιβεβαίωσης, ο χρήστης λαμβάνει ένα απαντητικό SMS, ο υπεύθυνος κρατήσεων λαμβάνει ένα ενημερωτικό SMS και ένα email με τα στοιχεία της κράτησης.



Εικόνα 4.5: *Φόρμα Επιβεβαίωσης στοιχείων*

4.2.4.1 SMS API

Το σύστημα υποστηρίζει αποστολές μηνυμάτων SMS προς κινητά τηλέφωνα κάνοντας χρήση ενός Application Programming Interface (API). Το API αυτό υποστηρίζεται από την εταιρεία clickatell.com και χρησιμοποιεί το HTTP πρωτόκολλο. Μπορεί να χρησιμοποιηθεί είτε με την μέθοδο POST είτε με την μέθοδο GET.

Για την αποστολή των SMS, το σύστημα αρχικά χρειάζεται να αυθεντικοποιήσει τον αποστολέα, στην συγκεκριμένη περίπτωση πρόκειται για το ίδιο το kratisinow.gr.

Η βασική δομή της εντολής για την επικοινωνίας με το SMS API είναι η ακόλουθη:

http://api.clickatell.com/http/sendmsg?api_id=xxxx&user=xxxx&password=xxxx&to=xxxx&text=xxxx

Προκειμένου να γίνει η παράδοση του μηνύματος, ο server πρέπει να πιστοποιήσει ότι η κλήση γίνεται από έγκυρο αποστολέα. Οι παράμετροι που χρησιμοποιούνται για την αυθεντικοποιήση του αποστολέα είναι:

- *api_id*: Αλφαριθμητικός κωδικός που εκδίδεται από την υπηρεσία
- *user*: Αλφαριθμητικό αναγνωριστικό χρήστη
- *password*: Κωδικός χρήστη

Παράμετροι για SMS

- *to*: Παραλήπτης
- *text*: Το περιεχόμενο του μηνύματος. Κάποιοι χαρακτήρες προσμετρούνται σαν δύο λόγω του GSM encoding.

Σημείωση: Το πρότυπο GSM υποστηρίζει 7bit character set.

Κεφάλαιο 5ο

Παρουσίαση Συστήματος Διαχείρισης Περιεχομένου

5.1 Εισαγωγή

Στο κεφάλαιο αυτό θα γίνει μία αναλυτική παρουσίαση του Συστήματος Διαχείρισης περιεχομένου (CMS) από τη σκοπιά του χρήστη του συστήματος, ενώ παράλληλα με κάθε βήμα θα αναλύονται οι εσωτερικές λειτουργίες του συστήματος και θα αιτιολογούνται οι επιλογές που έχουν γίνει.

5.2 Τι είναι το Σύστημα Διαχείρισης Περιεχομένου (CMS)

Το *Σύστημα Διαχείρισης Περιεχομένου* (Content Management System CMS), είναι ένα πρόγραμμα ειδικά σχεδιασμένο για τη διαχείριση ιστοτόπων. Δημιουργείται και εγκαθίσταται από τους σχεδιαστές ιστοσελίδων, αλλά προορίζεται για χρήση από τελικούς χρήστες. Αρχικά, προσφέρει έναν εύκολο, και εύχρηστο τρόπο ενημέρωσης περιεχομένου. Αυτό συνήθως γίνεται με τη χρήση ενός συστήματος πλοήγησης (browser). Ο χρήστης απλά εισάγει το νέο κείμενο και το αποθηκεύει. Η ιστοσελίδα ενημερώνεται αμέσως. Το ίδιο απλό είναι να προστεθούν νέες σελίδες, να διαγραφούν παλαιές, ή να αναδιαμορφωθεί μια ιστοσελίδα ώστε να συμβαδίζει με νέες απαιτήσεις ή προδιαγραφές. Το Σύστημα Διαχείρισης Περιεχομένου αυτοματοποιεί διάφορες διαδικασίες όπως τη διατήρηση εμφάνισης σελίδων σε όλο τον ιστόχωρο καθώς και την δημιουργία σχετικών μενού, συνδέσμων κλπ. Το *Content Management* είναι ουσιαστικά η διαχείριση του περιεχομένου.

5.3 Λειτουργίες CMS

Τα Σύστημα Διαχείρισης Περιεχομένου διαφοροποιούνται μεταξύ τους σε αρκετά σημεία, όλα όμως έχουν κοινό στόχο και θα πρέπει οπωσδήποτε να υποστηρίζουν

κάποιες βασικές λειτουργίες. Έτσι διακρίνονται κάποια βασικά υποσυστήματα όπως τα παρακάτω:

- Σύστημα σύνταξης (authoring),
- Σύστημα διαχείρισης (management),
- Σύστημα αυτοματοποίησης κύκλου εργασιών (workflow automation),
- Σύστημα έκδοσης.

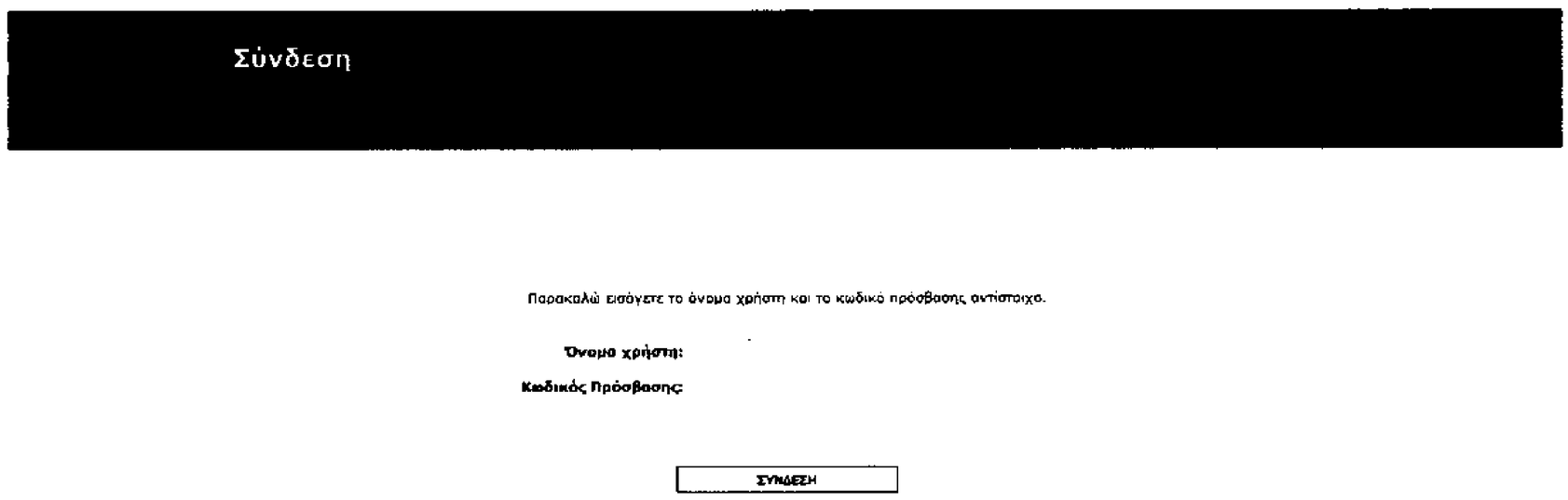
Για τις ανάγκες της εργασίας μας υλοποιήθηκαν τα ακόλουθα.

5.3.1 Είσοδος Χρήστη

Όπως φαίνεται παρακάτω (*Εικόνα 5.1*), κατά την είσοδο του χρήστη στο σύστημα η αρχική σελίδα που εμφανίζεται είναι η σελίδα εισόδου, στην οποία ο χρήστης θα πρέπει να εισάγει έγκυρα στοιχεία προκειμένου να συνδεθεί στο σύστημα.

Τα στοιχεία που καταχωρούνται στην φόρμα αναζητούνται στην Βάση Δεδομένων και συγκεκριμένα στον πίνακα *user*. Αν τα στοιχεία που εισάγει ο χρήστης υπάρχουν στον πίνακα *users*, το σύστημα του επιτρέπει να συνδεθεί και να μεταφερθεί στην κεντρική σελίδα διαχείρισης, αλλιώς τον παραπέμπει να ξαναπροσπαθήσει έως ότου δώσει τα έγκυρα στοιχεία.

Ο χρήστης έχει τη δυνατότητα να προβεί σε τρείς μη έγκυρες προσπάθειες σύνδεσης στο σύστημα. Στην τέταρτη προσπάθεια του απαγορεύεται η είσοδος για κάποιο χρονικό διάστημα. Η διεργασία αυτή αποσκοπεί στην προφύλαξη του συστήματος από τυχόν κακόβουλες προσπάθειες.



Εικόνα 5.1: *CMS - Είσοδος χρήστη*

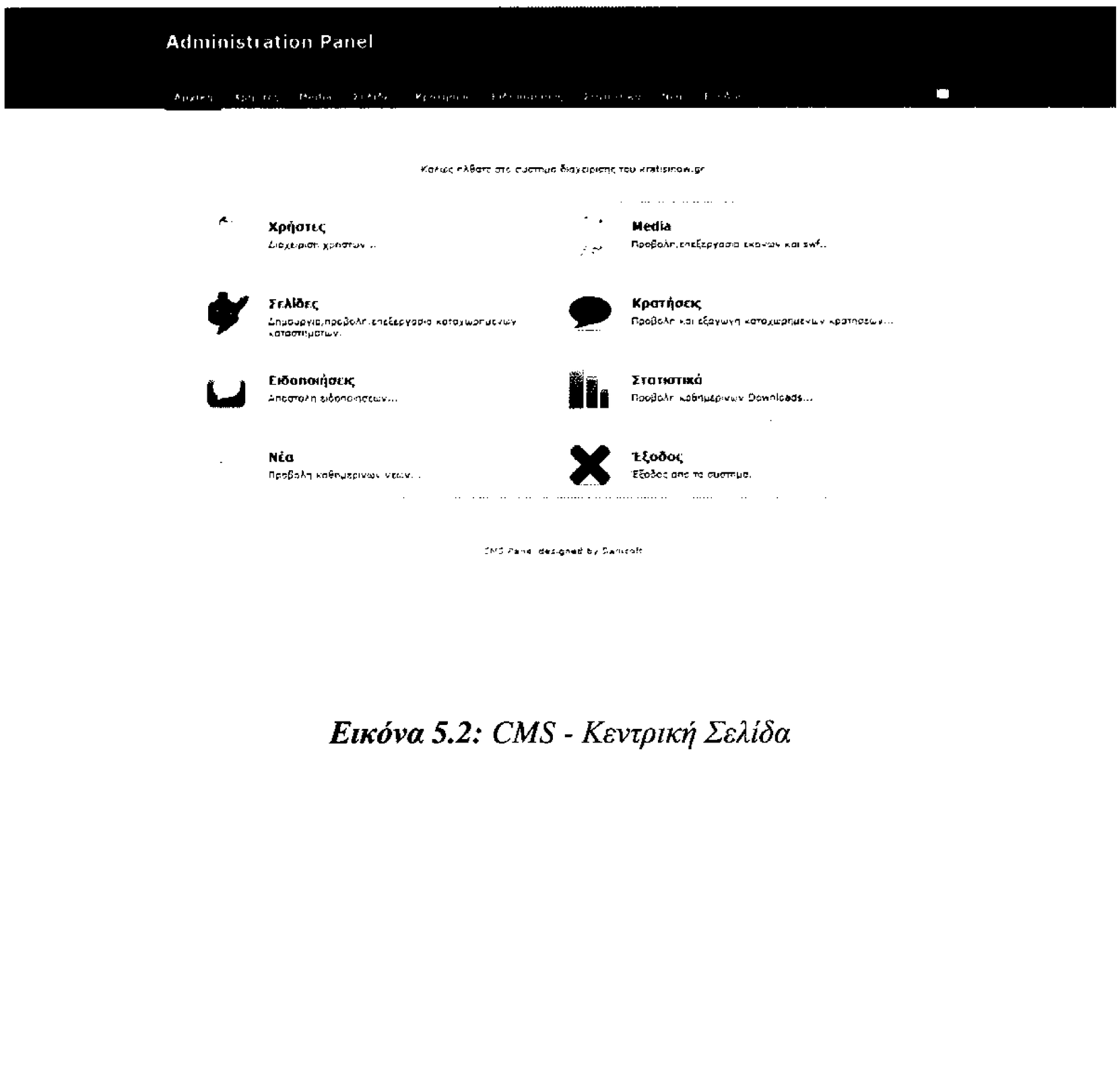
Ο έλεγχος που γίνεται για τον αριθμό των αποτυχημένων προσπαθειών σύνδεσης, έχει ως αποτέλεσμα την απαγόρευση του χρήστη στο σύστημα και πραγματοποιείται με το παρακάτω κώδικα php.

```
<?php

$max_logins = 3;

$ban_ip = $DB->query("SELECT
                *
                FROM
                ban_ip
                WHERE
                (ip = '" . $_SERVER['REMOTE_ADDR'] . "')");

if ($DB->numRows($ban_ip))
{
        $ip = $DB->fetchArray($ban_ip);

        if($ip['try_login'] >= $max_logins)
        {
                if ($ip['time'] + 3600 < time())
                {
                        $DB->query("UPDATE
                                        ban_ip
                                        SET
                                        try_login = '0',
                                        time = UNIX_TIMESTAMP()
                                        WHERE
                                        ip = '" . $_SERVER['REMOTE_ADDR'] .
"'");
```

```
                }
                else
                {
                        header('HTTP/1.0 401 Unauthorized');
                        echo "<h1>Access Deinied</h1>";
                        echo "You must enter a valid user name and password
                        to access the requested resource.";
                        exit;
                }
        }

}

?>
```

5.3.2 Κεντρική Σελίδα

Ύστερα από την επιτυχημένη σύνδεση του χρήστη στο σύστημα εμφανίζεται η κεντρική σελίδα προβάλλοντας όλες τις επιλογές και διεργασίες που μπορεί να εκτελέσει ο χρήστης.



Εικόνα 5.2: *CMS - Κεντρική Σελίδα*

Όπως φαίνεται παραπάνω (*Εικόνα 5.2*). η σελίδα χωρίζεται σε δύο βασικά μέρη:

1) Την κεφαλίδα, η οποία περιλαμβάνει το βασικό μενού πλοήγησης του CMS. Το μενού αυτό εμφανίζεται σε όλες τις σελίδες για ευκολότερη πρόσβαση του χρήστη στις διάφορες υπηρεσίες.

2) Τον υποδοχέα στο κέντρο της σελίδας, που περιλαμβάνει κάθε φορά τη βασική πληροφορία.

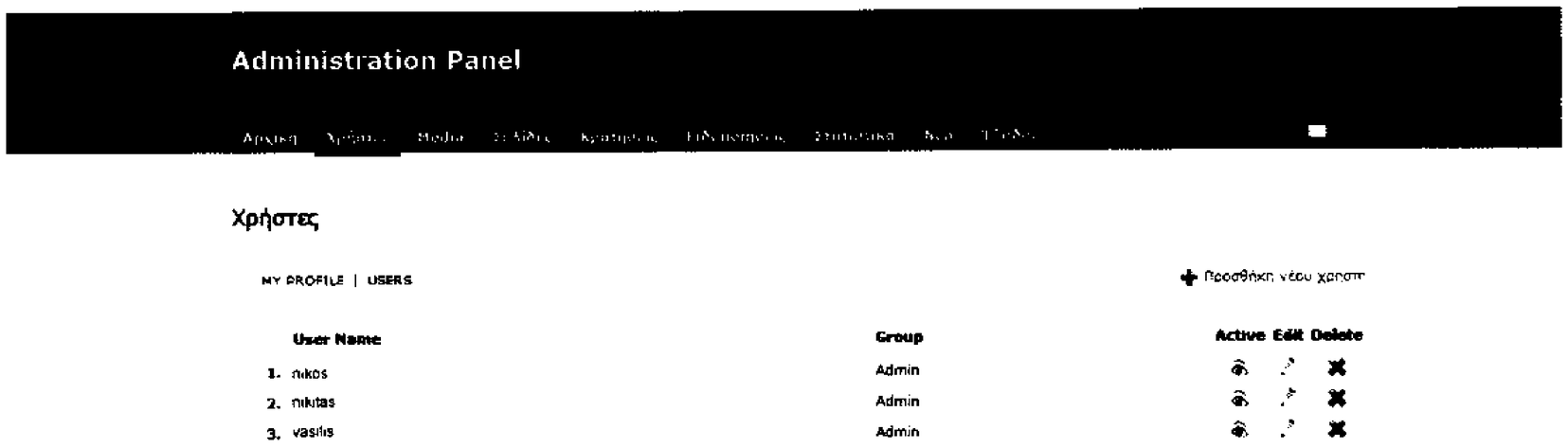
Το σύστημα μας δίνει τη δυνατότητα να ορίζουμε τι μπορεί να βλέπει ο κάθε χρήστης παραχωρώντας του τα αντίστοιχα δικαιώματα. Τα δικαιώματα αυτά ορίζονται κατά την δημιουργία του χρήστη και μπορούν να τροποποιηθούν στη συνέχεια από τον Administrator του συστήματος. Η συγκεκριμένη διεργασία αναλύεται παρακάτω.

Τα δικαιώματα που παρέχονται σε κάθε κατηγορία χρηστών καθώς και ο τύπος τους αρχικοποιούνται από την *class Permissions* (permissions.php).

5.3.3 Χρήστες Συστήματος

Με αυτή την επιλογή ο εκάστοτε χρήστης μπορεί να προβεί σε επεξεργασία των προσωπικών του στοιχείων. Ανάλογα με τα δικαιώματα που κατέχει, μπορεί να εκτελέσει ενέργειες όπως δημιουργία, διαγραφή ακόμα και απενεργοποίηση ενός προφίλ.

Στην παρακάτω εικόνα (*Εικόνα 5.3*). εμφανίζονται όλοι οι χρήστες που έχουν δικαίωμα σύνδεσης στο σύστημα. Σε κάθε χρήστη έχουν αποδοθεί τα αντίστοιχα δικαιώματα χρήσης του συστήματος.



Εικόνα 5.3: *CMS - Χρήστες Συστήματος*

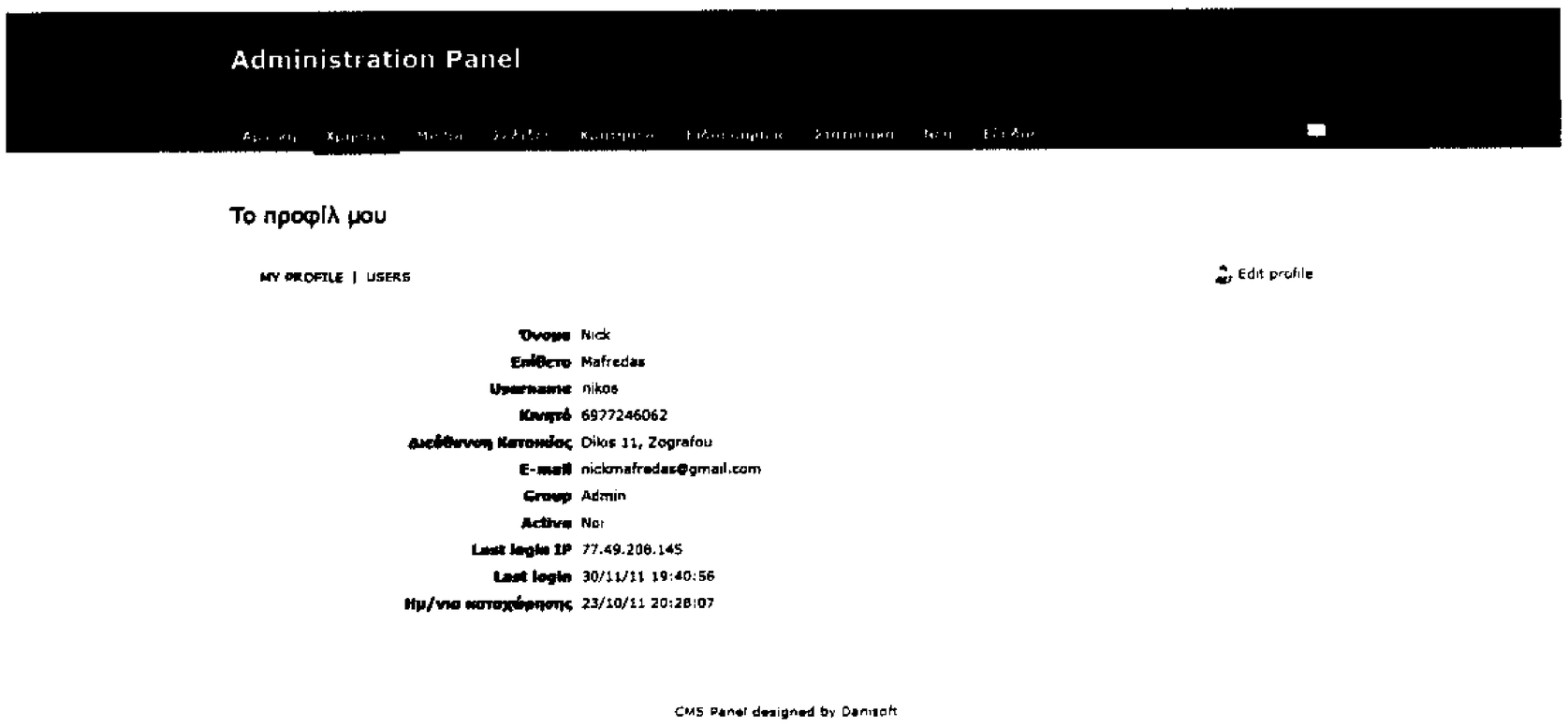
Η προβολή των καταχωρημένων χρηστών γίνεται με την χρήση της συνάρτησης *users_list.php* . Αφού αναζητηθούν και βρεθούν οι χρήστες που είναι καταχωρημένοι στη Βάση Δεδομένων προβάλλονται 10 ανά σελίδα. Η σελιδοποίηση γίνεται με τη χρήση της βιβλιοθήκης *Pager.*

5.3.3.1 Διαγραφή Χρήστη

Η διαδικασία της διαγραφής ενός χρήστη γίνεται ασύγχρονα με την βοήθεια της τεχνολογίας AJAX και της βιβλιοθήκης jQuery. Η jQuery παρέχει μια πλούσια γκάμα από συναρτήσεις για τον προγραμματισμό και την χρήση της AJAX, ενώ ταυτόχρονα δίνει τη δυνατότητα να πραγματοποιούνται αιτήσεις TXT, HTML, XML ή JSON σε έναν απομακρυσμένο server χρησιμοποιώντας είτε το *HTTP Get* είτε το *HTTP Post*. Έτσι μπορούν να φορτωθούν απομακρυσμένα δεδομένα απευθείας σε κάποιο στοιχείο της ιστοσελίδας ή να οριστούν ιδιότητες σε HTML στοιχεία (element π.χ εικόνα).

5.3.3.2 Προβολή - Επεξεργασία Χρήστη

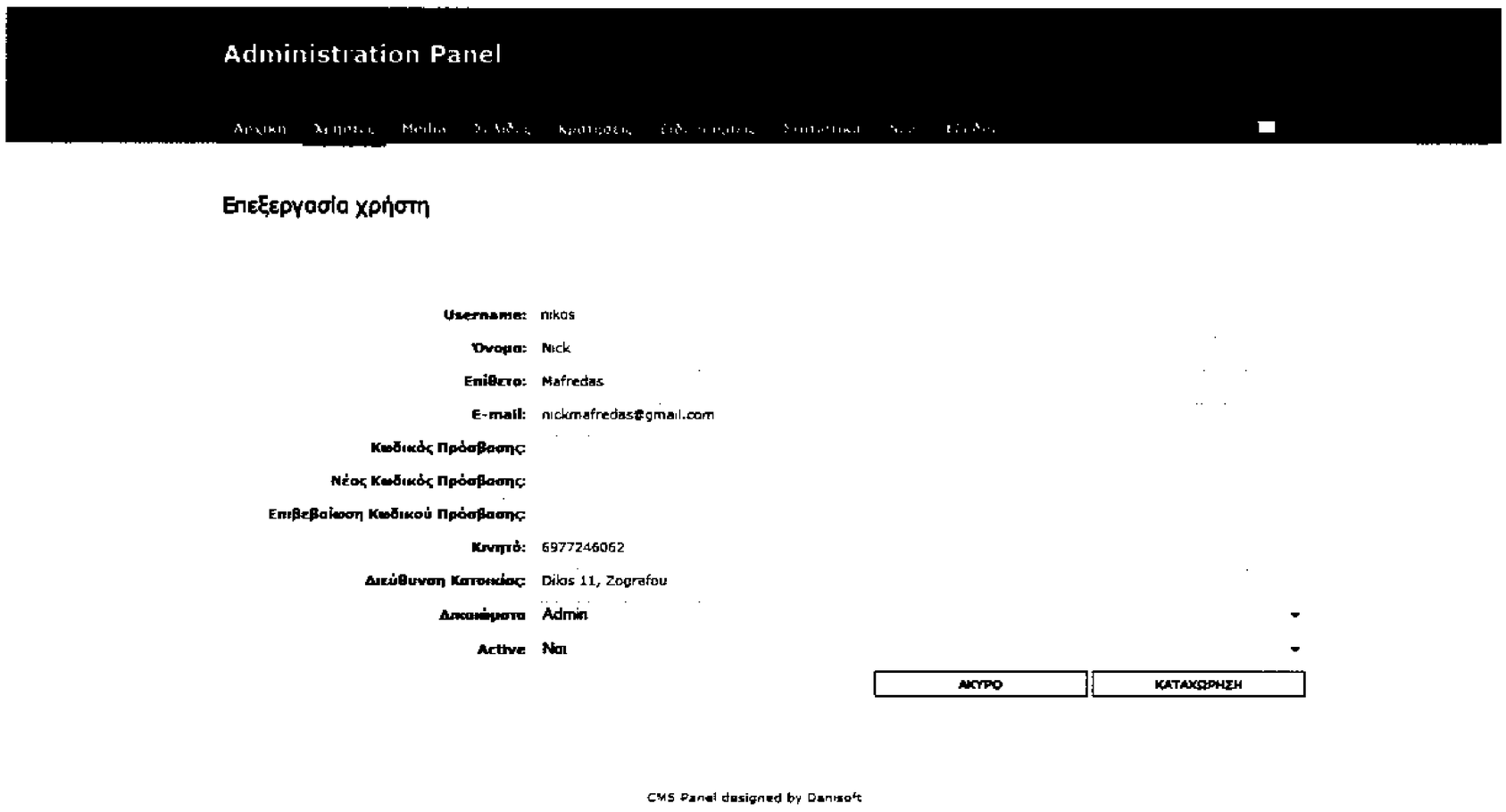
Επιλέγοντας έναν χρήστη προβάλλονται οι αντίστοιχες πληροφορίες του (*Εικόνα 5.4*). Τα στοιχεία που καταγράφονται στη Βάση Δεδομένων είναι το *Όνομα*, *Επίθετο*, *Username*, *Password*, *Κινητό*, *Διεύθυνση Κατοικίας*, *E-mail*, *Group*, *Active*, *Last login IP*, *Last login* και *Ημ/νια καταχώρησης*.



Εικόνα 5.4: *CMS - Προβολή του προφίλ*

Η προβολή των στοιχείων γίνεται με τη χρήση της συνάρτησης *load_my_profile()*. Η *load_my_profile()* αναζητά στοιχεία ενός συγκεκριμένου χρήστη και επιστρέφει έναν πίνακα που περιέχει όλες εκείνες τις πληροφορίες που είναι προς επεξεργασία - προβολή.

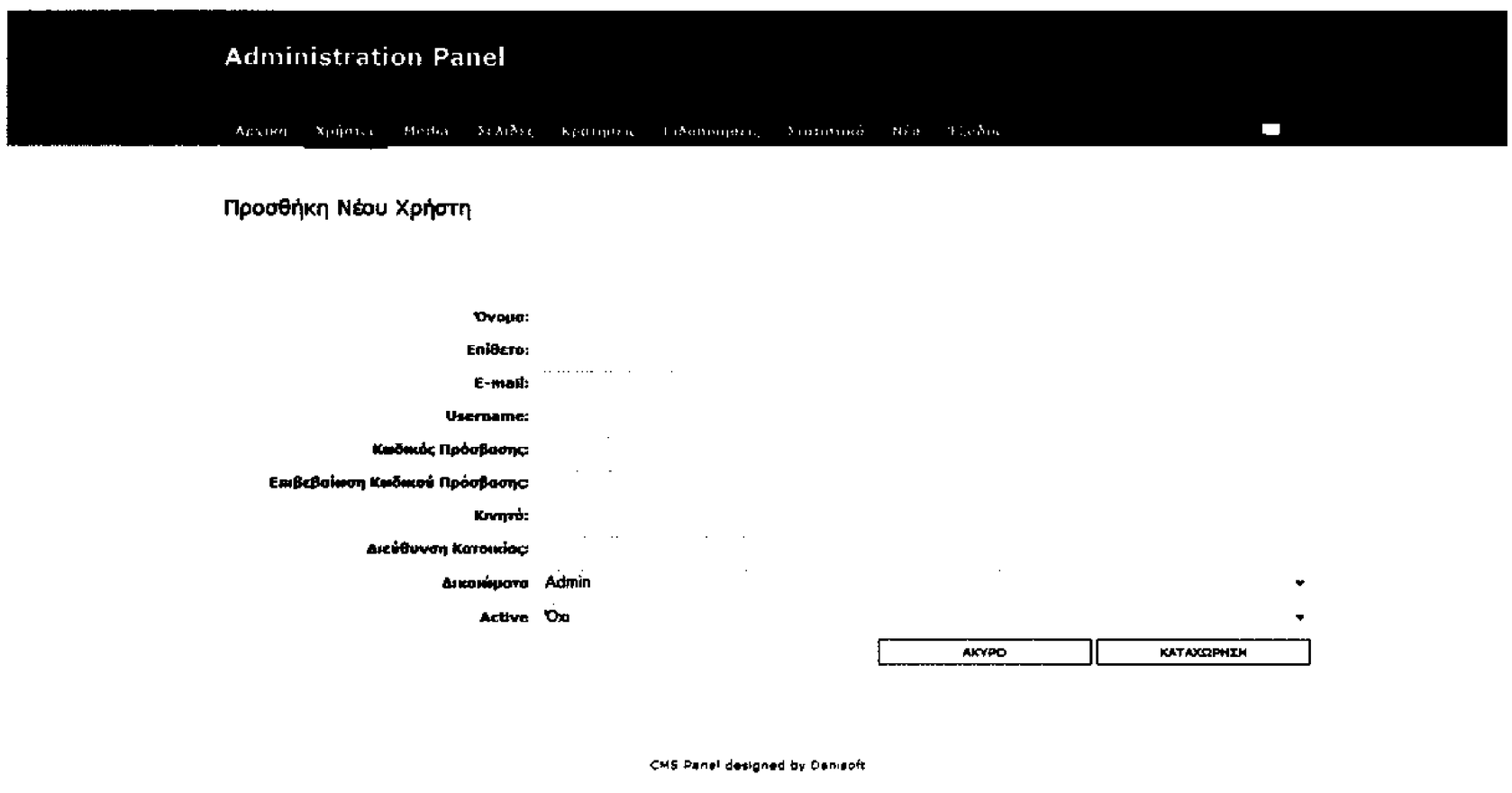
Παρακάτω εμφανίζεται η φόρμα με τα στοιχεία που είναι ήδη καταχωρημένα στη Βάση Δεδομένων (*Εικόνα 5.5*). Τα πεδία είναι ενεργά να τα επεξεργαστεί ο χρήστης και πατώντας το κουμπί καταχώρηση ενημερώνει τη βάση με τα καινούργια στοιχεία. Αφού γίνει η τροποποίηση εμφανίζεται μήνυμα που ενημερώνει τον χρήστη για την επιτυχή εκτέλεση.



Εικόνα 5.5: *CMS - Επεξεργασία χρήστη*

5.3.3.3 Προσθήκη Χρήστη

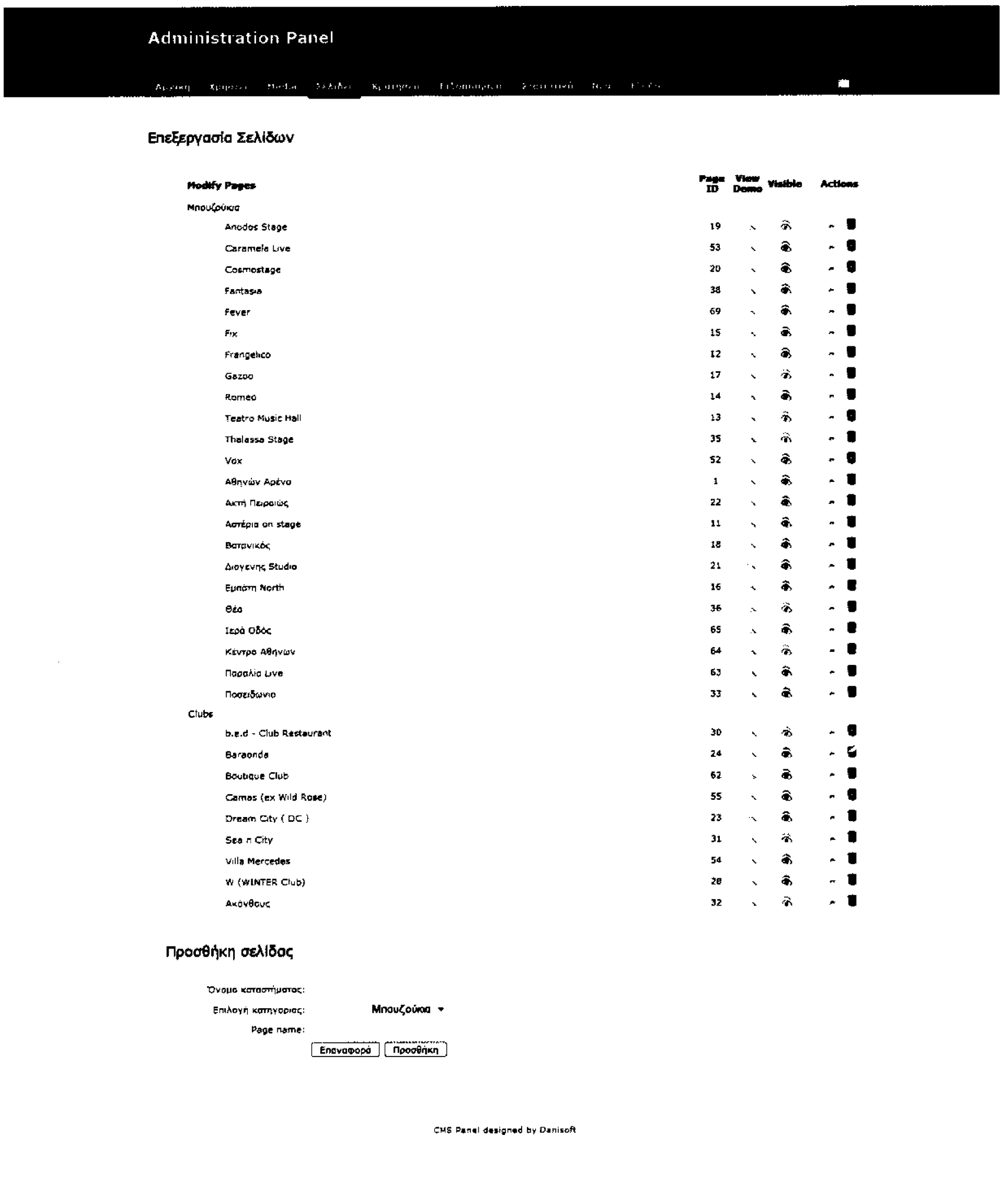
Στην εικόνα 5.6 εμφανίζεται η φόρμα όπου ο διαχειριστής του συστήματος έχει τη δυνατότητα να προσθέσει έναν νέο χρήστη στη βάση δεδομένων. Ο διαχειριστής καταχωρεί τα στοιχεία που απαιτούνται για την συγκεκριμένη διαδικασία όπως Όνομα, Επίθετο, E-mail, Username, Password, Κινητό τηλέφωνο και Διεύθυνση Κατοικίας. Επιλέγει επίσης τι δικαιώματα θα αποδώσει στον νέο χρήστη (Administrator ή User) και αν το νέο προφίλ θα είναι ενεργοποιημένο ή απενεργοποιημένο. Κατά την διαδικασία επιλογής του κωδικού πρόσβασης έχουν γίνει οι απαραίτητες διεργασίες, ώστε να ελέγχεται κατά πόσο ο κωδικός που εισάγεται είναι ισχυρός ή όχι..



Εικόνα 5.6: *CMS - Προσθήκη νέου χρήστη*

5.3.4 Σελίδες

Σε αυτό το σημείο ο χρήστης μπορεί να προσθέσει, να επεξεργαστεί, να διαγράψει, να απενεργοποιήσει και να ενημερωθεί με τα διαθέσιμα καταστήματα (ανά κατηγορία) που είναι καταχωρημένα μέσα στο σύστημα (*Εικόνα 5.7*).



Εικόνα 5.7: *CMS – Προβολή Καταστημάτων*

Η προβολή των καταχωρημένων καταστημάτων γίνεται με την χρήση της συνάρτησης *load_all_pages()*.

Η διαδικασία της προσθήκης ισχύει μόνο για την εισαγωγή ενός νέου καταστήματος. Κατά την προσθήκη, ελέγχεται αν το νεοεισαχθέν κατάστημα είναι ήδη καταχωρημένο στην βάση δεδομένων ή όχι. Σε περίπτωση που το όνομα που εισάγεται υφίσταται, ο χρήστης ενημερώνεται με ένα μήνυμα ανεπιτυχούς εισαγωγής.

Επιπλέον, ο χρήστης έχει τη δυνατότητα απενεργοποίησης – ενεργοποίησης ενός καταστήματος, έχοντας σαν αποτέλεσμα την άμεση προβολή ή μη του αντίστοιχου καταστήματος στην κύρια ιστοσελίδα.

Μια επιπρόσθετη ενέργεια που μπορεί να εκτελέσει ο χρήστης είναι αυτή της διαγραφής ενός καταστήματος. Η διαδικασία της διαγραφής κάνει χρήση της ασύγχρονης τεχνολογίας AJAX που αναλύθηκε παραπάνω.

5.3.4.1 Προσθήκη – Επεξεργασία Καταστημάτων

Σε αυτό το σημείο ο χρήστης ενημερώνει το κατάστημα με την αντίστοιχη φωτογραφία του, τα απαραίτητα κείμενα, την διεύθυνση, τις διαθέσιμες ημέρες όπως επίσης και τα ονόματα των καλλιτεχνών.

Στην διαδικασία της προσθήκης – επεξεργασίας καταστήματος, για την επεξεργασία των κειμένων χρησιμοποιείται ο *CKEditor*. Ο CKEditor αποτελεί έναν open source επεξεργαστή κειμένου που παρέχει την δυνατότητα μορφοποίησης κείμενων είτε με τη χρήση ενός απλού HTML editor, είτε με ένα πιο σύνθετο editor. Για να φορτωθεί ο επεξεργαστής κειμένου στην εκάστοτε σελίδα γίνεται χρήση της συνάρτησης *Editor()*.

Για την εύρεση της διεύθυνσης ενός καταστήματος χρησιμοποιείται η υπηρεσία που παρέχει η Google, το *Google maps*. Το *Google maps* κατατάσσεται μέσα στα σύγχρονα εργαλεία αναζήτησης για την ακριβή εύρεση συντεταγμένων μιας περιοχής – διεύθυνσης. Κάθε σημείο πάνω στο χάρτη χαρακτηρίζεται από δύο τιμές, το *longitude* και το *latitude*. Η τιμή latitude χαρακτηρίζει το γεωγραφικό πλάτος ενός σημείου πάνω στην επιφάνεια της γης, ενώ το longitude το γεωγραφικό μήκος.

Για τις ανάγκες της άμεσης εύρεσης των συντεταγμένων κάθε καταστήματος, μελετώντας το Google Maps API που διατίθεται από την Google, αναπτύχθηκε ένα

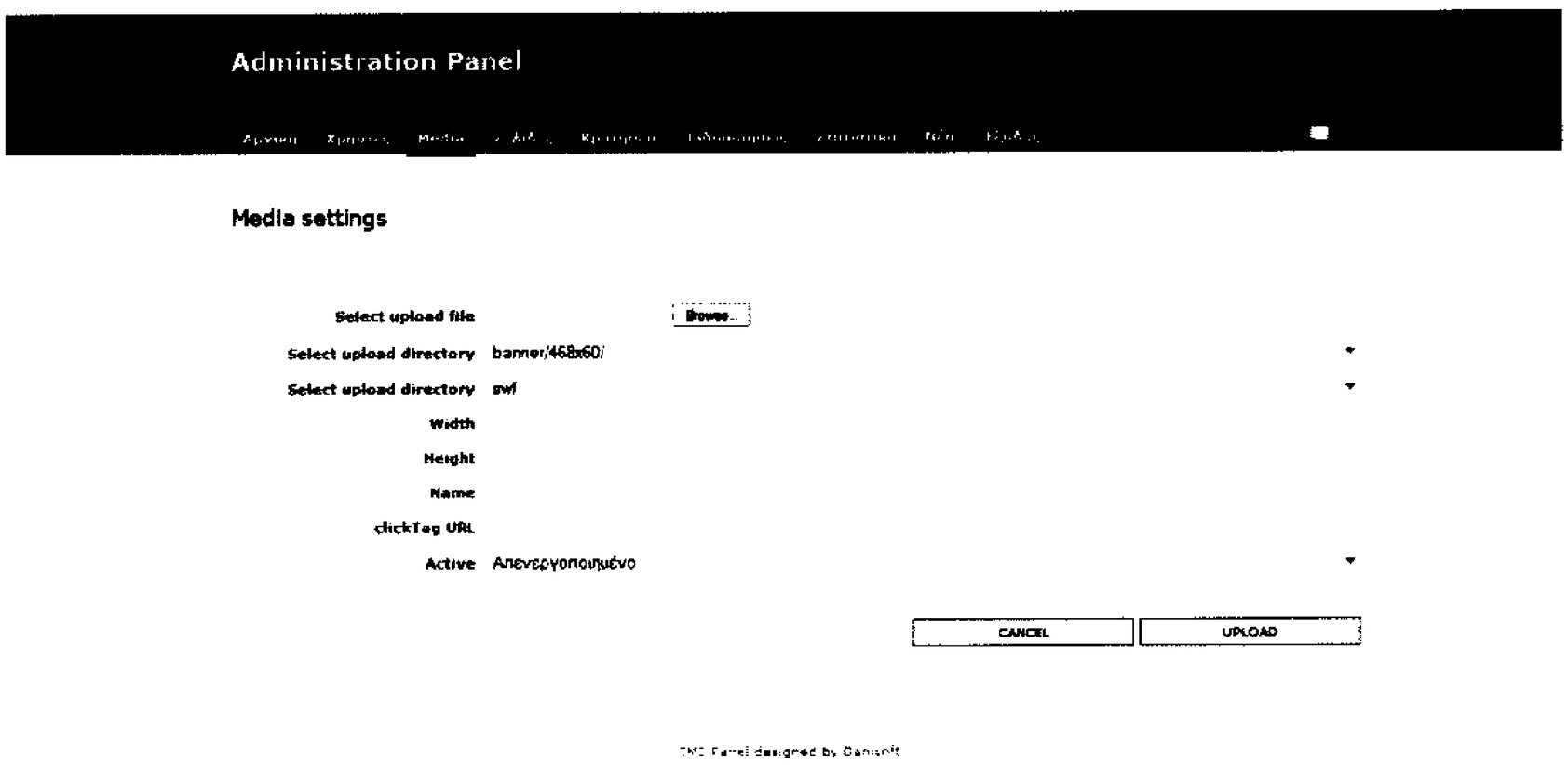
σύστημα αυτόματης εύρεσης συντεταγμένων πάνω στο χάρτη. Η αρχικοποίηση του χάρτη πραγματοποιείται με το script *geocoder.js*.

Ο χρήστης έχει τη δυνατότητα, είτε να πληκτρολογήσει την διεύθυνση του καταστήματος και να εμφανιστεί το αντίστοιχο point πάνω στο χάρτη, είτε να μετακινήσει το mark point του χάρτη στο σημείο που θέλει και να εμφανιστεί αυτόματα η αντίστοιχη διεύθυνση.

5.3.5 Media

Στην κατηγορία «Media» ο χρήστης διαχειρίζεται τις διαφημίσεις που προβάλλονται στα κύρια σημεία της ιστοσελίδας. Οι διαθέσιμες ενέργειες αφορούν την προσθήκη, ενεργοποίηση – απενεργοποίηση, επεξεργασία και διαγραφή των διαφημίσεων ανά περιοχή.

Οι διαθέσιμες περιοχές προβολής μέσω τις ιστοσελίδας τηρούν τις παγκόσμιες αρχές των διαστάσεων προβολής τους. Πιο συγκεκριμένα, η ιστοσελίδα παρέχει τη δυνατότητα προβολής διαφημίσεων σε διαστάσεις 468x69, 728x90 και 336x280. Επίσης, σε κάθε περιοχή προβάλλονται τυχαία μια ή περισσότερες από τις ενεργοποιημένες διαφημίσεις.



Εικόνα 5.8: *CMS – Προσθήκη media*

5.3.6 Κρατήσεις

Σε αυτήν την κατηγορία ο διαχειριστής του συστήματος έχει τη δυνατότητα να διαχειριστεί τις κρατήσεις που έχουν πραγματοποιηθεί ανά κατηγορία μέσα από την ιστοσελίδα, τις smartphone εφαρμογές ή μέσω facebook. Οι κύριες ενέργειες είναι η επεξεργασία, η διαγραφή και η προσθήκη.

Όταν ο επισκέπτης εκδηλώσει ενδιαφέρον μέσω της ιστοσελίδας ή των εφαρμογών, η κράτηση αυτή καταχωρείται στη βάση δεδομένων. Τα στοιχεία που καταγράφονται για κάθε κράτηση ποικίλουν αναλόγως την κατηγορία. Ο διαχειριστής του συστήματος έχει τη δυνατότητα να βλέπει στην κατηγορία «Εκκρεμότητες» τις κρατήσεις που έχουν πραγματοποιηθεί αλλά δεν έχουν επικυρωθεί. Μόλις μια κράτηση επικυρωθεί τότε από την κατηγορία «Εκκρεμότητες» αυτόματα μεταφέρεται στην αντίστοιχη κατηγορία (πχ clubs).

Για κάθε κράτηση που πραγματοποιείται καταγράφονται τα παρακάτω στοιχεία: *Όνομα πελάτη, Τηλέφωνο πελάτη, Κατάστημα, Ημέρα κράτησης, Αριθμός ατόμων, Εγκυρότητα, Φοιτητικό, Κατανάλωση, Ημέρα εκδήλωσης, Λεπτομέρειες, Πηγή προέλευσης.*

5.3.7 Ειδοποιήσεις

Η υπηρεσία ειδοποιήσεων παρέχει την δυνατότητα στον χρήστη να αποστέλλει ενημερωτικά notifications στους χρήστες κινητών συσκευών. Προβάλλεται το ιστορικό των ειδοποιήσεων (*Εικόνα 5.9*) που έχει αποσταλεί στο παρελθόν και δίνεται η δυνατότητα δημιουργίας νέων ειδοποιήσεων (*Εικόνα 5.10*). Για κάθε νέα ειδοποίηση ο χρήστης εκτός από την εισαγωγή του μηνύματος, έχει τη δυνατότητα να προγραμματίσει την ημερομηνία και ώρα αποστολής.

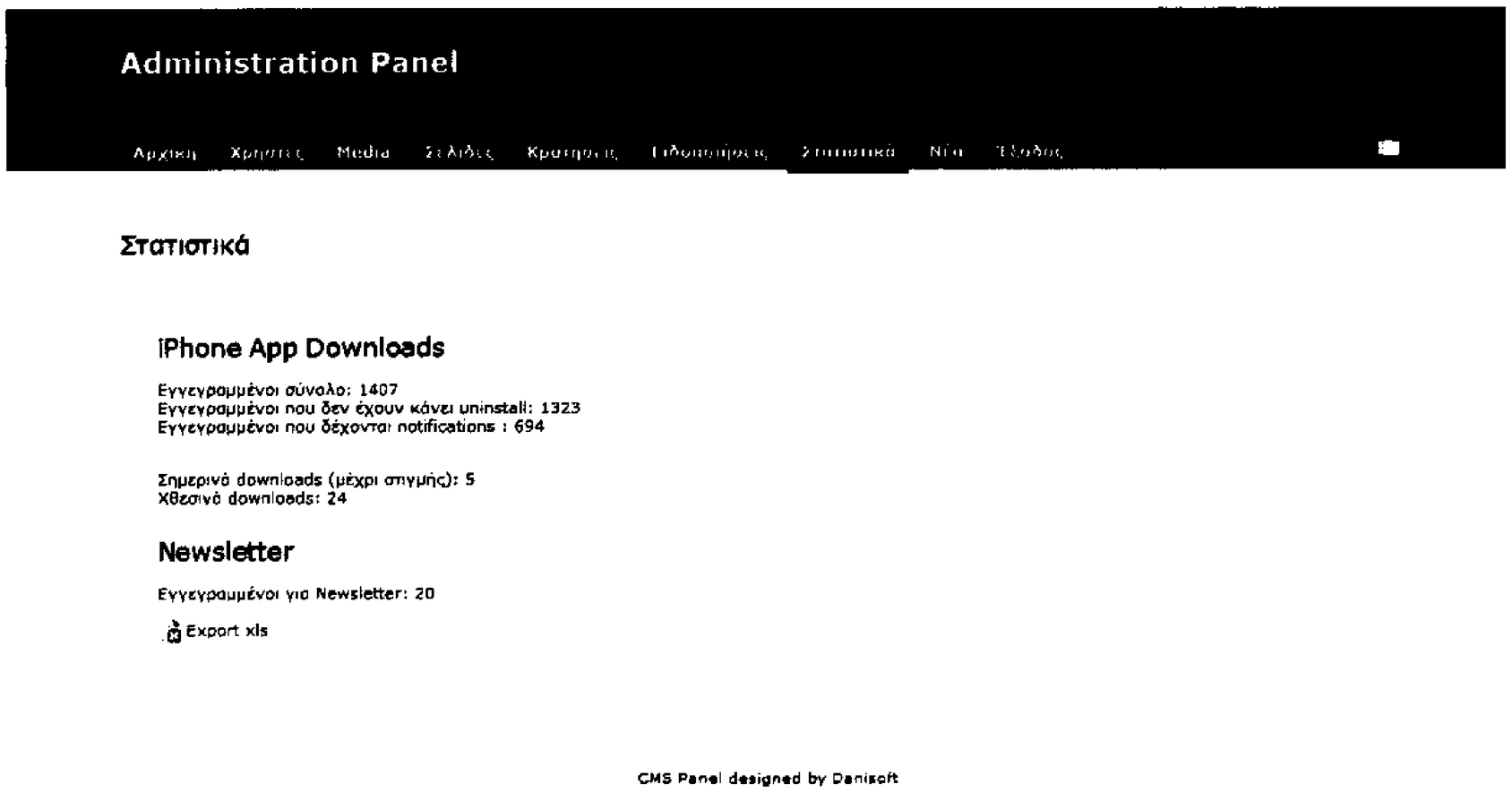
Η επικοινωνία και η διαδικασία αποστολής των ειδοποιήσεων στα smartphones αναλύονται σε ξεχωριστό κεφάλαιο.

Εικόνα 5.9: *CMS – Προβολή ιστορικού ειδοποιήσεων*

Εικόνα 5.10: *CMS – Προσθήκη ειδοποίησης (notification)*

5.3.8 Στατιστικά

Στην κατηγορία «Στατιστικά» ο χρήστης ενημερώνεται σχετικά με τα συνολικά downloads της iPhone εφαρμογής (*Εικόνα 5.11*). Ειδικότερα, ενημερώνεται σχετικά με τον αριθμό των χρηστών που έχουν διαγράψει την εφαρμογή, τον αριθμό των χρηστών που δέχονται notifications αλλά και τα ημερήσια downloads που έχουν πραγματοποιηθεί.



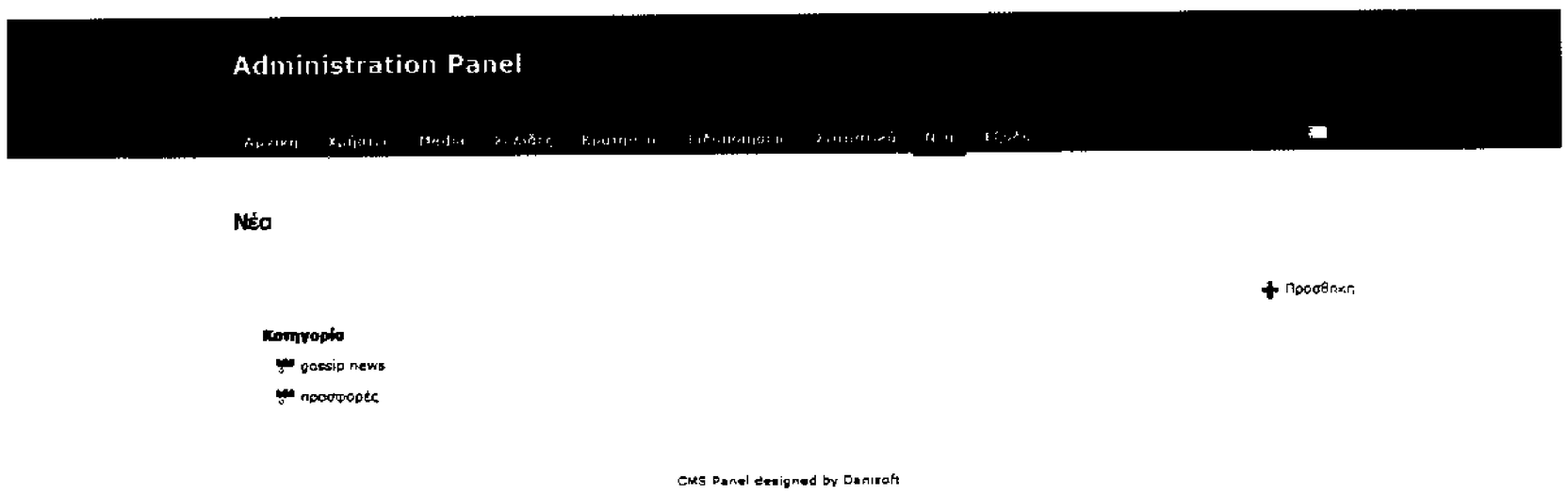
Εικόνα 5.11: *CMS - Προβολή Στατιστικών*

Τέλος, παρέχεται η δυνατότητα παρακολούθησης των συνολικά εγγεγραμμένων χρηστών στην υπηρεσία newsletter. Ο χρήστης μπορεί να εξάγει την λίστα των χρηστών που έχουν εγγραφεί στην υπηρεσία σε αρχείο .xls.

Η εξαγωγή της λίστας σε xls πραγματοποιείται με το κάλεσμα του αρχείου *xls_genarator.php*, το οποίο κάνει χρήση της βιβλιοθήκης Spreadsheet που αναλύθηκε παραπάνω.

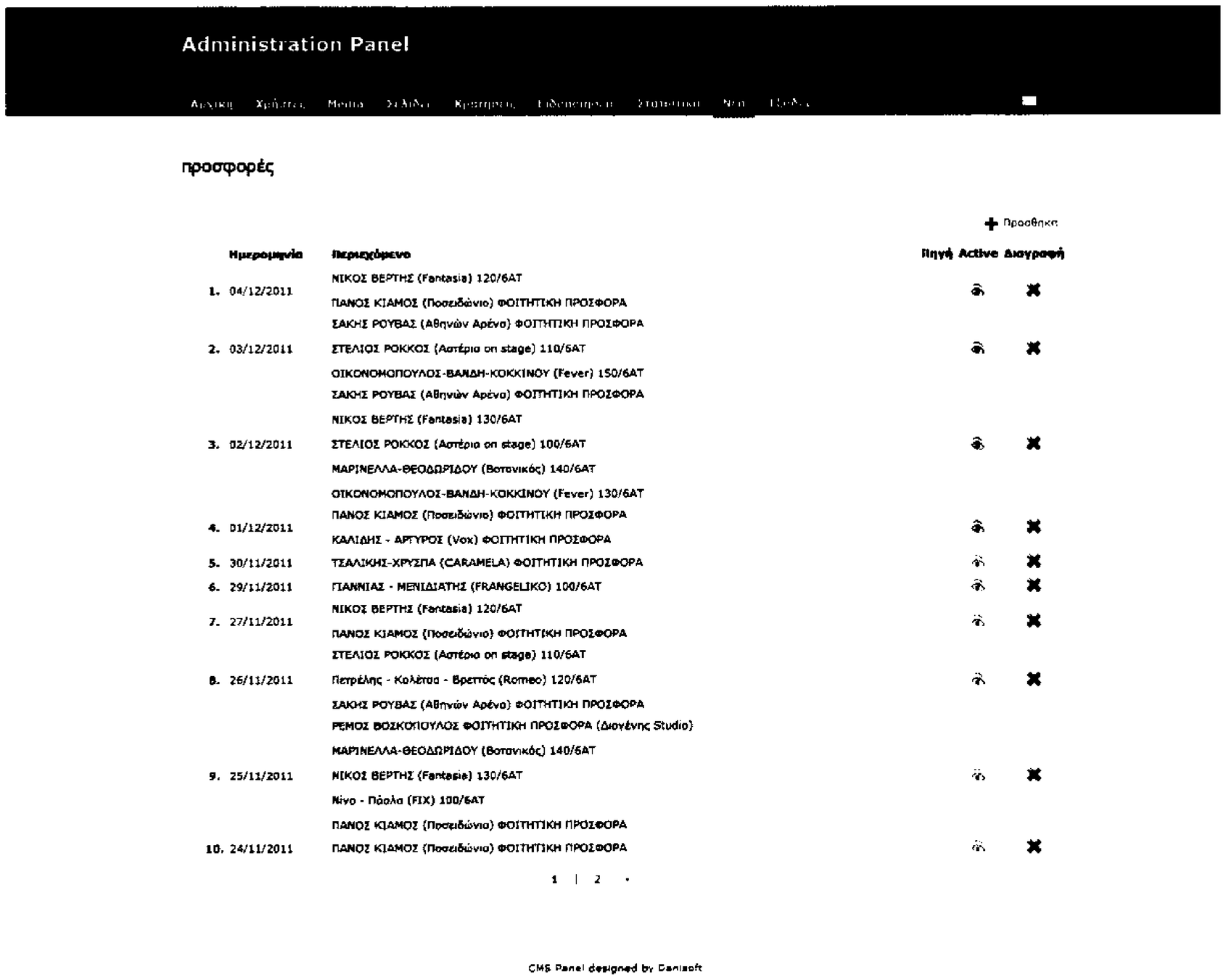
5.3.9 Νέα

Σε αυτή την επιλογή ο χρήστης μπορεί να διαχειριστεί τα «Νέα» και τις «Προσφορές» που προβάλλονται στην ιστοσελίδα (*Εικόνα 5.12*). Η προσθήκη, η ενεργοποίηση – απενεργοποίηση και η διαγραφή είναι οι ενέργειες που μπορεί να εκτελέσει ο χρήστης σε κάθε κατηγορία (*Εικόνα 5.13*).



Εικόνα 5.12: *CMS - Προβολή Κατηγοριών*

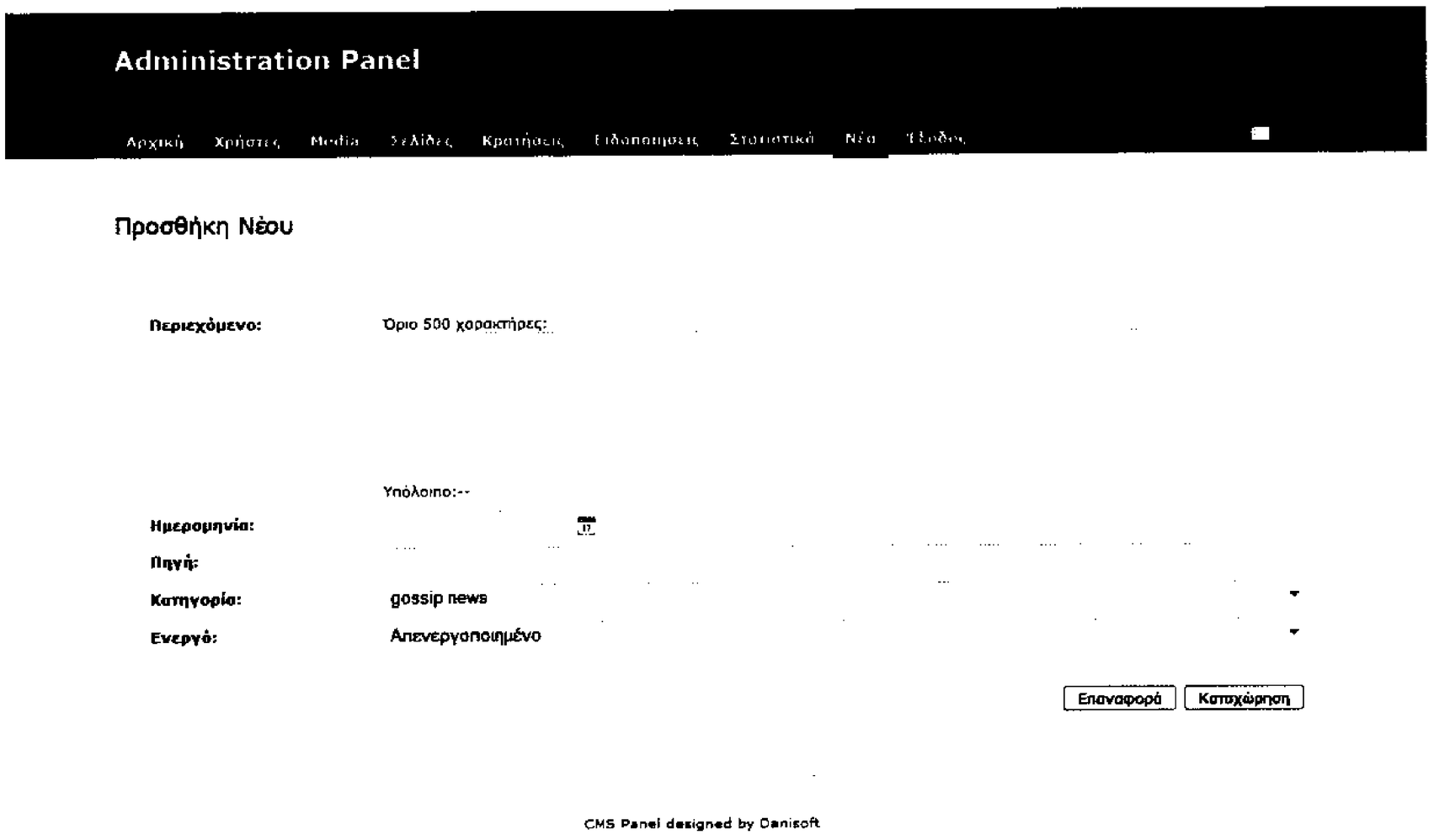
Για την προβολή των εγγραφών έχει χρησιμοποιηθεί η βιβλιοθήκη *Pager*, όπου μας βοηθάει στην σελιδοποίηση.



Εικόνα 5.13: *CMS - Προβολή Προσφορών*

Η ταξινόμηση των εγγραφών γίνεται με χρονολογική σειρά, με τις πιο πρόσφατες να εμφανίζονται πρώτες στη λίστα. Ο χρήστης έχει τη δυνατότητα να απενεργοποιήσει ή να διαγράψει μια εγγραφή.

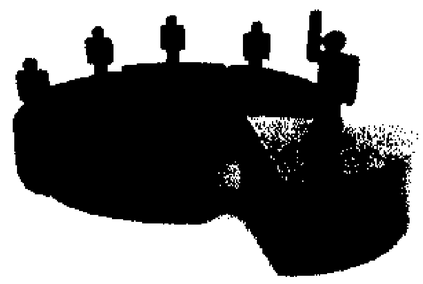
Τέλος, η διαδικασία προσθήκης «νέων» αποτελεί ενέργεια ενημέρωσης με νέες εγγραφές των κατηγοριών «Νέα» και «Προσφορές». Ο χρήστης έχει δυνατότητα να προσθέσει Περιεχόμενο, να επιλέξει Ημερομηνία Εισαγωγής, να αναφέρει την Πηγή πληροφορίας (αφορά την κατηγορία των Νέων), να επιλέξει κατηγορία και να θέσει αν η νέα εγγραφή θα είναι ενεργοποιημένη και διαθέσιμη προς τους επισκέπτες της ιστοσελίδας. Στην εικόνα που ακολουθεί προβάλλεται η φόρμα προσθήκης.



Εικόνα 5.14: *CMS – Προσθήκη Εγγραφής*

Κεφάλαιο 6ο

Search Engine Optimization



6.1 Εισαγωγή

Το *Search Engine Optimization* (SEO) είναι όλες εκείνες οι σύγχρονες τεχνικές και διαδικασίες που χρησιμοποιούμε ώστε να επιτύχουμε την εμφάνιση ενός website σε μια υψηλή θέση στα κύρια (οργανικά) αποτελέσματα στις μηχανές αναζήτησης.

Το SEO είναι μία διαδικασία που εφαρμόζεται σε ένα website κατά την οποία υλοποιούνται διάφορες μέθοδοι ανάλυσης και κατασκευής των επιμέρους σελίδων αλλά και ολόκληρου site, με σκοπό την βελτιστοποίησή τους, έτσι ώστε να μπορούν οι μηχανές αναζήτησης να τα ανιχνεύσουν, να τα αναλύσουν και να τα καταχωρήσουν (*indexing*).

Όταν ένας χρήστης ψάχνει για μια πληροφορία σε μια μηχανή αναζήτησης εμφανίζονται χιλιάδες αποτελέσματα. Σημασία έχουν τα websites που εμφανίζονται μόνο στην κορυφή των αποτελεσμάτων. Ένας μέσος χρήστης δεν πρόκειται να ψάξει πέρα από τα 20-30 πρώτα αποτελέσματα.

Ένα *καλοσχεδιασμένο SEO* μπορεί να κάνει το περιεχόμενο των ιστοσελίδων πιο σχετικό, πιο ελκυστικό αλλά και πιο φιλικό προς τους μηχανισμούς ανάγνωσης των μηχανών αναζήτησης έτσι ώστε με μεγαλύτερη ευκολία και ταχύτητα να σαρώσουν ολόκληρο το site και να το ευρετηριάσουν.

Το Search Engine Optimization (SEO) έχει αποδειχτεί ότι είναι μια από τις καλύτερες τεχνικές ανάπτυξης και online προώθησης περιεχομένου.

6.2 Συστατικά Επιτυχίας μιας καμπάνιας Search Engine Optimization (SEO)

Η επιτυχία στις μηχανές αναζήτησης δεν είναι ποτέ τυχαία. Οι μηχανές αναζήτησης ταξινομούν τα websites (*search engine ranking*) που βρίσκονται στη βάση τους, με συγκεκριμένους αλγόριθμους.

Αυτοί οι αλγόριθμοι περιοδικά ανανεώνονται πράγμα που σημαίνει ότι μια υψηλή θέση χρειάζεται αρκετή έρευνα και προσπάθεια για να επιτευχθεί.

Επιπλέον αν επιτευχθεί μια υψηλή θέση μέσα από μια καμπάνια Search Engine Optimization (SEO) δεν διασφαλίζεται ότι το website μας θα βρίσκεται για πάντα «ψηλά». Η κατάταξη αλλάζει συνεχώς, οπότε ο μόνος τρόπος για την κορυφή είναι η συνεχής παρακολούθηση των εξελίξεων, του ανταγωνισμού καθώς και η βελτίωση του website μας (*fine-tuning*).

Παρόλα αυτά υπάρχουν κάποιοι βασικοί παράμετροι που βοηθούν στην επίτευξη υψηλών θέσεων στις μηχανές αναζήτησης.

Παράγοντες Επιτυχίας

- *Επιλογή Λέξεων-Κλειδιά* (Keyword Research & Analysis). Η σωστή επιλογή των κατάλληλων keywords είναι κρίσιμη.
- *Βελτιστοποίηση* (Optimization). Ένα επιτυχημένο website πρέπει να είναι βελτιστοποιημένο ως προς τα επιλεγμένα keywords καθώς και φιλικό προς τις μηχανές αναζήτησης.
- *Ανάπτυξη Περιεχομένου* (Content Development). Οι μηχανές αναζήτησης ενδιαφέρονται για "πλούσιο" και συχνά ανανεωμένο περιεχόμενο. Μια στρατηγική για τη διασφάλιση και την ανάπτυξη νέου περιεχομένου είναι απαραίτητη τόσο για την εξασφάλιση υψηλών θέσεων όσο και για την ικανοποίηση των επισκεπτών μας.
- *Popularity* (Δημοτικότητα Website). Οι περισσότερες μηχανές αναζήτησης δίνουν ιδιαίτερη έμφαση στο πόσο «δημοφιλές» είναι το website μας. Σε γενικές γραμμές ως Popularity εννοούμε τον αριθμό και την ποιότητα των website που κάνουν link στο δικό μας. Όσο μεγαλύτερο Popularity διαθέτει το website μας τόσο αυξάνονται οι πιθανότητες να επιτύχουμε μια υψηλότερη θέση στις μηχανές αναζήτησης.

6.3 Οι θεμελιώδης έννοιες του SEO

Οι θεμελιώδης έννοιες πάνω στις οποίες στηρίζετε το SEO είναι οι λέξεις κλειδιά (keywords) και οι σύνδεσμοι (links).

6.3.1 Οι λέξεις κλειδιά (keywords)

Οι μηχανές αναζήτησης χρησιμοποιούν τα *bot*, τα οποία είναι *robot* με αυτοματοποιημένο λογισμικό με τα οποία σαρώνουν όλους τους δικτυακούς τόπους. Σαρώνοντας ο *crawler* τις ιστοσελίδες, καθορίζει το περιεχόμενο ενός site ανάλογα με τις λέξεις-κλειδιά (*keywords*) που περιέχει. Έτσι ανάλογα με το τι λέξεις-κλειδιά που βρίσκονται στο περιεχόμενο του site, ανάλογη θα είναι η κίνηση *(traffic)* και το είδος των επισκεπτών του. Αν για παράδειγμα στο περιεχόμενο των άρθρων μιας ιστοσελίδας υπάρχουν λέξεις-κλειδιά για «πόρτα, παράθυρο, κουφώματα, κατασκευή, τοποθέτηση, αλουμίνιο, υαλοπίνακας κλπ, κάνει προφανές στο robot, ότι επισκέπτεται μία σελίδα που είναι για κατασκευή και τοποθέτηση κουφωμάτων. Ως εκ τούτου το site θα επισημανθεί από τη μηχανή αναζήτησης σαν ένα site με πληροφορίες «κατασκευής και τοποθέτησης κουφωμάτων». Οι μηχανές αναζήτησης έχουν τον εξειδικευμένο μηχανισμό να προσδιορίζουν το είδος του περιεχομένου μιας ιστοσελίδας από ένα ελάχιστο αριθμό λέξεων-κλειδιών.

Η *Google* έχει κατορθώσει να καθορίσει ένα πρότυπο είδος τεχνολογίας για τον προσδιορισμό του περιεχομένου μιας ιστοσελίδας. Δίνοντας ένα μικρό αριθμό από σχετικές λέξεις-κλειδιά, η Google μπορεί να εντοπίσει και να προσδιορίσει το θέμα και να δώσει περισσότερες λέξεις σχετικές με αυτές που δόθηκαν. Ως εκ τούτου για τον καθορισμό και εφαρμογή μιας στρατηγικής SEO οι λέξεις-κλειδιά είναι τα πρώτα τα οποία πρέπει ληφθούν υπόψη και να καθοριστούν. Για τον προσδιορισμό των κατάλληλων λέξεων-κλειδιών θα πρέπει να σκεφτούμε σαν ένας χρήστης που ψάχνει με μια μηχανή αναζήτησης ένα site παρόμοιο με το δικό μας. Αυτές θα πρέπει να είναι και οι λέξεις τις οποίες θα πρέπει να χρησιμοποιήσουμε στο περιεχόμενο των κειμένων μας, έτσι ώστε να είναι σχετικό με τις *σελίδες αποτελεσμάτων* των μηχανών αναζήτησης SERP (*Search Engine Results Pages*).

6.3.1.2 Οι λέξεις-κλειδιά στο SEO και η σωστή χρήση τους στα άρθρα

Σε αυτή την ενότητα θα αναλύσουμε την χρήση των keywords και ποία είναι η κατάλληλη θέση τους μέσα στα κείμενα μας για αποτελεσματικότητα του SEO.

- Στο url της ιστοσελίδας θα πρέπει να υπάρχει η *λέξη-κλειδί (keyword)*
- Οι *λέξεις-κλειδιά* που μας ενδιαφέρουν θα πρέπει να βρίσκονται απαραίτητα στον τίτλο του άρθρου
- Η *λέξη-κλειδί* θα πρέπει να βρίσκεται στην πρώτη ή την δεύτερη πρόταση του άρθρου
- *Η λέξη-κλειδί* θα πρέπει να βρίσκεται στο μέσο της πρώτης παραγράφου
- Μια σχετική εικόνα θα είναι καλό να τοποθετηθεί στην πρώτη παράγραφο με μία ALT tag, η οποία θα περιέχει την *λέξη-κλειδί* ενώ η ονομασία του αρχείου αυτού θα πρέπει να περιέχει αντίστοιχα την *λέξη-κλειδί*
- Η *λέξη-κλειδί* ή συναφείς λέξεις-κλειδιά (keywords) θα πρέπει να βρίσκονται απαραίτητα και στην δεύτερη και τρίτη παράγραφο του άρθρου
- Η λέξη-κλειδί θα πρέπει να υπάρχει και στην τελευταία πρόταση του άρθρου
- Οι *λέξεις-κλειδιά* μέσα στο άρθρο θα πρέπει να έχουν την σωστή αναλογία στην *πυκνότητα*, *συχνότητα*, και *γειτνίαση* μεταξύ τους για καλύτερο SEO, την σωστή παρουσίαση του άρθρου στον επισκέπτη αλλά και την αποφυγή να θεωρηθεί από τις μηχανές αναζήτησης σαν spam

6.3.1.3 Οδηγίες και τεχνικές SEO βελτιστοποίησης

Όπως αναφέρθηκε ένας σημαντικός παράγοντας αξιολόγησης είναι το περιεχόμενο των σελίδων. Η Google αναλύει όχι μόνο τα meta tags αλλά το σύνολο του περιεχομένου της ιστοσελίδας. Λαμβάνει υπόψη της πολλές παραμέτρους, όπως το μέγεθος και χρωματισμό των γραμματοσειρών, τους τίτλους, τις παραγράφους, τις λέξεις και τη συνάφειά τους με το κείμενο αλλά και το υπόλοιπο περιεχόμενο των άλλων ιστοσελίδων.

Πιο αναλυτικά, ακολουθούν οδηγίες και τεχνικές βελτιστοποίησης του SEO που θα πρέπει να τηρούνται απο μια ιστοσελίδα, ώστε να επιτευχθεί η μέγιστη δυνατή υψηλή θέση στα κύρια (οργανικά) αποτελέσματα των μηχανών αναζήτησης.

Τεχνικές βελτιστοποίησης

- *Η λέξη-κλειδί (keyword) μέσα στον Τίτλο, title tag* : Τα αποτελέσματα αναζήτησης εμφανίζουν πάντα τον τίτλο *(title tag)* σαν τίτλο της ιστοσελίδας. Συνεπώς ο τίτλος θα πρέπει να είναι σύντομος και να περιέχει 6-12 λέξεις, το ιδανικό είναι το σύνολο 65 χαρακτήρων μαζί με τα κενά διαστήματα. Η λέξη-κλειδί μπορεί να βρίσκεται οπουδήποτε αλλά κατά προτίμηση στην αρχή (η σημασία του title tag αναλύεται παρακάτω σε ξεχωριστή ενότητα)
- *Η λέξη-κλειδί (keyword) στις παραγράφους (headings)* : Η τοποθέτηση των λέξεων-κλειδιών στην παράγραφο (heading) *H1 tag*, αποτελεί σημαντικό παράγοντα στο SEO, αλλά θα πρέπει και το σύνολο του περιεχομένου του άρθρου να έχει συνάφεια με αυτές τις λέξεις-κλειδιά (keywords)
- *Η λέξη-κλειδί (keyword) μέσα στο domain name*: Ένας πολύ σημαντικός παράγοντας για το SEO είναι να υπάρχει η λέξη-κλειδί (keyword) μέσα στο domain name, αυτό βοηθάει πολύ στην κατάταξη στα αποτελέσματα αναζήτησης αλλά και στην ευρετηρίαση των σελίδων για τις λέξεις-κλειδιά που μας ενδιαφέρουν
- *Η πυκνότητα (density)*: Ένας πολύ βασικός παράγοντας στο SEO είναι η πυκνότητα που έχουν οι λέξεις-κλειδιά μέσα στο περιεχόμενο του άρθρου. Θα πρέπει η *πυκνότητα (keyword density)* που θα έχουν οι λέξεις-κλειδιά, να κυμαίνεται κάτω από το 10%. Για 2-3 στοχευμένες φράσεις λέξεις-κλειδιά ιδανική αναλογία είναι 6-7% και για τις δευτερεύουσες συναφείς λέξεις-κλειδιά 1-2%
- *Οι λέξεις-κλειδιά (keywords) στο ALT tags*: Οι μηχανές αναζήτησης δεν έχουν την δυνατότητα για την ώρα να διαβάσουν κείμενο μέσα από φωτογραφίες και για αυτό οι απαραίτητοι χαρακτηρισμοί και περιγραφές για τις εικόνες του κειμένου τοποθετούνται στις ετικέτες *ALT tags*, ένας επίσης πολύ σημαντικός παράγοντας του SEO τον οποίο οι μηχανές αναζήτησης εξετάζουν και λαμβάνουν υπόψη τους. Σε κάθε εικόνα είναι απαραίτητο να χρησιμοποιούνται 2-3 στοχευμένες φράσεις *λέξεις-κλειδιά*
- *Η εγγύτητα ή γειτνίαση (proximity) για τις λέξεις-κλειδιά (keywords)*: Το πόσο κοντά θα βρίσκονται 2 ή περισσότερες από τις *στοχευμένες φράσεις λέξεις-κλειδιά (keywords)* μέσα στο κείμενο λαμβάνεται σοβαρά υπόψη από τις

μηχανές αναζήτησης. Όσο πιο κοντά βρίσκονται οι λέξεις-κλειδιά τόσο καλύτερα αποδίδουν στην κατάταξη. Αν μπορούν να βρίσκονται δίπλα μεταξύ τους, χωρίς να παρεμβάλετε άλλη λέξη είναι το ιδανικό του συντελεστή της γειτνίασης

- *Το όνομα του αρχείου των εικόνων και οι λέξεις-κλειδιά (keywords)*: Είναι πολύ σημαντικό τα ονόματα των αρχείων των εικόνων να είναι συναφή με τις *στοχευμένες φράσεις λέξεις-κλειδιά* του κειμένου π.χ keywords.jpg
- *Χρήση μεγάλων (long tail keywords) λέξεων-κλειδιών*: Έχει παρατηρηθεί ότι οι μεγάλες και μακρόσυρτες λέξεις, οι οποίες έχουν και χαμηλότερο ανταγωνισμό αποδίδουν καλύτερα και επιτυγχάνουν υψηλότερες θέσεις στην κατάταξη
- *Συνώνυμες λέξεις-κλειδιά (keywords)*: Η χρήση *συνωνύμων λέξεων* είναι ένας πολύ αποδοτικός τρόπος για την επίτευξη υψηλών κατατάξεων ιδιαίτερα σε Αγγλόφωνες ιστοσελίδες

6.3.2 Σύνδεσμοι (incoming links)

Χρησιμοποιώντας τις κατάλληλες λέξεις-κλειδιά όπως αναφέραμε παραπάνω, οι μηχανές αναζήτησης παίρνουν μια πολύ καλή ιδέα για το περιεχόμενο μιας ιστοσελίδας και που περίπου αναφέρεται. Τι θα μπορούσε όμως να προσδιορίσει ακριβέστερα και να καθορίσει μια ιστοσελίδα; Οι *Διασυνδέσεις (Links) με άλλα website*. Είναι κανόνας ότι όσο περισσότερα link για μια ιστοσελίδα υπάρχουν σε άλλα site, τόσο πιο σημαντικό είναι το περιεχόμενό της και οι πληροφορίες για το θέμα. Όσο περισσότεροι εισερχόμενοι σύνδεσμοι (*incoming links)* υπάρχουν προς το την ιστοσελίδα, έχει ως αποτέλεσμα της εμφάνισης της σε υψηλότερη θέση στην αναζήτηση. Επίσης εάν ένα πολύ σημαντικό και δημοφιλές site έχει link προς τη δική μας ιστοσελίδα, αυτό προσδίδει κύρος, μεγαλύτερη σημασία και βάρος στην ιστοσελίδα μας και μετράει θετικά, περισσότερο ακόμα και αν υπάρχουν link σε 10 «όχι σημαντικά» site. Το πώς και ποιο site κατατάσσετε και θεωρείτε ως σημαντικό (*Page Rank*) θα αναλυθεί παρακάτω.

Τα link από άλλα μεγάλα *site «υψηλού ειδικού βάρους»* πέρα από το ότι βοηθούν στην *οργανική κίνηση επισκεπτών* από τις μηχανές αναζήτησης (*organic traffic*),

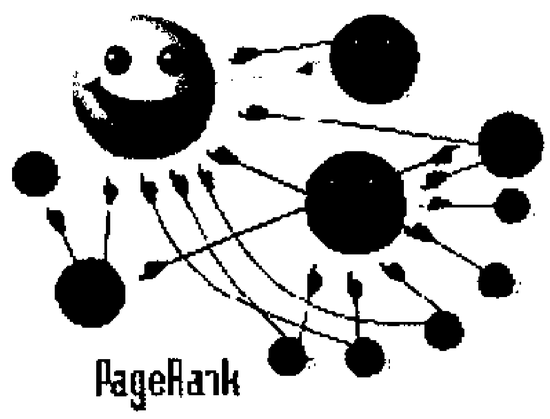
έχουν ενδιαφέρον διότι είναι εγγύηση για *άμεση επισκεψιμότητα* (*direct traffic*). Αυτό φυσικά πάλι είναι σχετικό διότι θα πρέπει να διαθέτει η ιστοσελίδα μας και το ανάλογο ποιοτικό περιεχόμενο.

Συνεπώς αυτά τα δύο στοιχεία Links και Keywords είναι αλληλένδετα και απαραίτητα για την επιτυχία του SEO μιας ιστοσελίδας.

6.4 PageRank

Μία έννοια την οποία συναντούμε και ακούμε πολύ συχνά στο χώρο του *Search Engine Optimization* (SEO) είναι το *PageRank* (PR) .

Το Pagerank είναι μία αριθμητική τιμή, η οποία αντιπροσωπεύει το πόσο σημαντική και σπουδαία είναι μία *ιστοσελίδα στο διαδίκτυο*, κατά την αντικειμενική κρίση της μηχανής αναζήτησης Google.



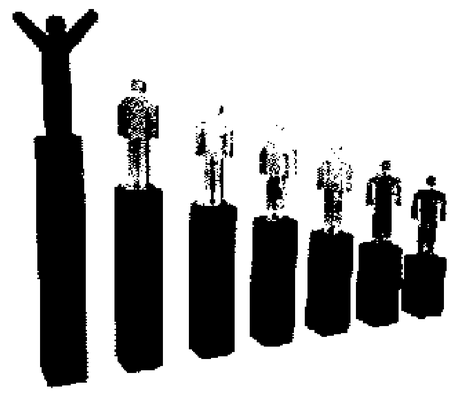
Το Pagerank επηρεάζει σημαντικά, μαζί με κάποιους επιπλέον παράγοντες, την κατάταξη στα αποτελέσματα αναζήτησης. Για τον υπολογισμό και προσδιορισμό του Pagerank, η Google χρησιμοποιεί ένα *πολύπλοκο αλγόριθμο* με εκατομμύρια μεταβλητές και όρους και ο υπολογισμός του γίνετε με αδιάβλητο αυτοματοποιημένο τρόπο.

Κύρια λογική της όμως στην αξιολόγηση είναι η σύνδεση (*link)* π.χ. μιας Ιστοσελίδας Α με μία Ιστοσελίδα Β. Θεωρεί τον σύνδεσμο (*link*) αυτό ότι είναι θετική ψήφος για την Ιστοσελίδα Β από την Ιστοσελίδα Α. Συνεπώς οι διασυνδέσεις των ιστοσελίδων και εσωτερικά και εξωτερικά λαμβάνοντα θετικά υπόψη, διότι όσο πιο πολλά links έχει μία ιστοσελίδα από άλλες, καθώς και η σπουδαιότητά τους, θεωρούνται πολλές θετικές ψήφοι, άρα η ιστοσελίδα χαρακτηρίζετε σημαντική και σπουδαία.

Εάν δηλαδή ο ψήφοι προς την ιστοσελίδα προέρχονται από σημαντικές ιστοσελίδες μεγάλης σπουδαιότητας, η αξία η οποία προσδίδετε είναι ακόμα μεγαλύτερη.

Η *κλίμακα διαβάθμισης του Pagerank* είναι από το 1 έως το 10 (PR1 - PR10). Όσο μεγαλύτερο Pagerank προσλαμβάνει μια ιστοσελίδα, τόσο υψηλότερη θέση έχει στην κατάταξη στα αποτελέσματα αναζήτησης (*SERP*).

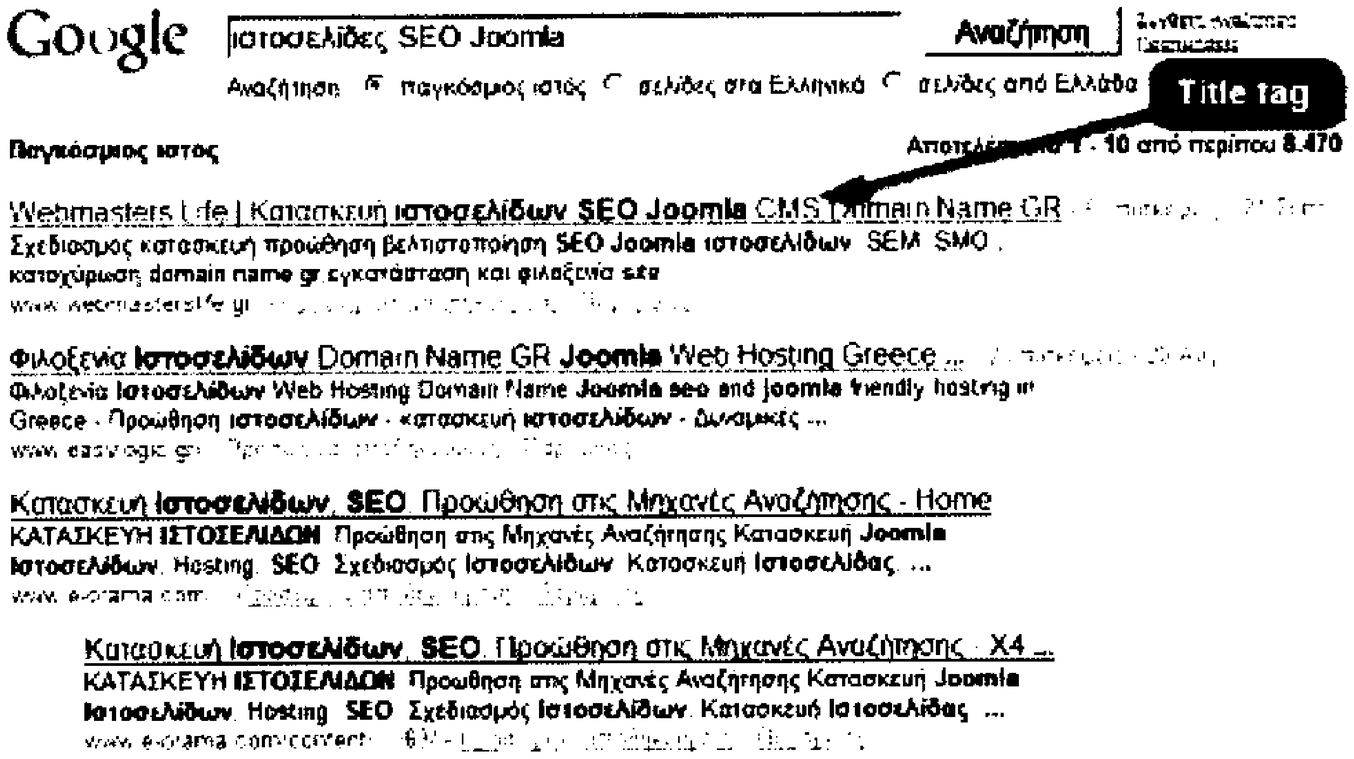
6.5 Title tag - On Page optimization



Το *On Page optimization* περιλαμβάνει όλες εκείνες τις τεχνικές του *SEO*, οι οποίες χρησιμοποιούνται σε μια ιστοσελίδα για την βελτίωση της θέσης της στην κατάταξη των *SERP (Search Engine Results Pages)*, χωρίς την βοήθεια από άλλα site. Υπάρχουν πολλά στοιχεία του *On Page optimization* τα οποία μπορούμε να διορθώσουμε και να βελτιώσουμε, το σημαντικότερο όμως από αυτά είναι το *Title tag*, δηλαδή ο Τίτλος της ιστοσελίδας.

Ο Τίτλος *(Title)* μιας ιστοσελίδας είναι το πρώτο αλλά και το πιο εμφανές στοιχείο στην ιστοσελίδα και εμφανίζεται συνήθως στην επάνω μπάρα του προγράμματος πλοήγησης. Στο κώδικα *HTML* της ιστοσελίδας εμφανίζεται μέσα στο <title> tag στο τμήμα του <head>.

Λόγω αυτής της σπουδαιότητας και προβεβλημένης θέσης που έχει, θεωρείται από τις μηχανές αναζήτησης μεγάλης σημασίας, εμφανίζεται στα αποτελέσματα αναζήτησης και κατά συνέπεια θα πρέπει να φροντίζουμε για όσο το δυνατόν καλύτερη βελτιστοποίησή του. Ένα καλός *SEO Title* θα πρέπει να περιέχει τις κατάλληλες *λέξεις-κλειδιά (keywords)* με κατάλληλη *γειτνίαση* και *πυκνότητα*, να έχει το σωστό μήκος σε χαρακτήρες, να είναι *σύντομος* και *περιγραφικός* αλλά θα πρέπει να προκαλεί και το ενδιαφέρον του επισκέπτη.



Εικόνα 6.1: Title

6.5.1 Η σπουδαιότητα του Title tag στο SEO

Χρησιμοποιείται στα αποτελέσματα αναζήτησης

Όταν η μηχανή αναζήτησης δίνει τις σελίδες με τα αποτελέσματα από μια αναζήτηση *(SERP),* χρησιμοποιεί τον *Τίτλο της ιστοσελίδας* που χρησιμεύσει ως σύνδεσμος με τη συγκεκριμένη ιστοσελίδα. *Ο τίτλος* θα πρέπει να είναι με τέτοιο τρόπο γραμμένος ώστε να προκαλέσει την περιέργεια του επισκέπτη για να κάνει κλικ στην ιστοσελίδα. Με λίγα λόγια, ο τίτλος της ιστοσελίδας είναι αυτός που κάνει την πρώτη εντύπωση στον επισκέπτη.

Εμφανίζεται στο πρόγραμμα πλοήγησης

Κάθε πρόγραμμα πλοήγησης (internet explorer, mozilla firefox, opera, safari κλπ), εμφανίζει τον τίτλο της ιστοσελίδας στο επάνω μέρος του παραθύρου του. Ο τίτλος δίνει σε μία γραμμή τις απαραίτητες πληροφορίες στον επισκέπτη για το περιεχόμενο της ιστοσελίδας και βοηθά στην ευκολία πλοήγησης, αν το παράθυρο του browser έχει πολλές καρτέλες ανοικτές ταυτόχρονα.

Η Google δίνει μεγάλη σημασία στο Title tag

Ο αλγόριθμος της Google δίνει μεγάλη σημασία στην ετικέτα του τίτλου. Η επιλογή του σωστού και κατάλληλου τίτλου, χρησιμοποιώντας τις κατάλληλες *λέξεις-κλειδιά (keywords),* μπορεί να μεταβάλει πολύ δραστικά την κατάταξη της ιστοσελίδας στα αποτελέσματα αναζήτησης.

Καταχώρηση της ιστοσελίδας σε καταλόγους (directories)

Οι περισσότεροι από τους καταλόγους αναζήτησης χρησιμοποιούν τον τίτλο της ιστοσελίδας για την καταχώρησή της τοποθετώντας ανάλογο σύνδεσμο (link) προς την ιστοσελίδα.

6.5.2 Τεχνικές SEO για βελτιστοποίηση του Title tag

- Το μήκος του τίτλου δεν πρέπει να είναι περισσότερο από 120 χαρακτήρες, αν και θεωρείται ότι οι 85 χαρακτήρες είναι ένα καλό μήκος για τον τίτλο της ιστοσελίδας.
- Οι συστάσεις του W3C θεωρούν σαν ιδανικό μήκος τίτλου τους 64 χαρακτήρες, συμπεριλαμβανομένων των διαστημάτων. Η Google προβάλει 66 χαρακτήρες μαζί με τα διαστήματα.
- Ο τίτλος της ιστοσελίδας θα πρέπει να είναι καλογραμμένος, με σωστή γραμματική και ορθογραφία και με τις κατάλληλες *λέξεις-κλειδιά*. Ο τίτλος θα πρέπει να προκαλεί το ενδιαφέρον του επισκέπτη.
- Κάθε ιστοσελίδα πρέπει να έχει διαφορετικό τίτλο, διότι ο αλγόριθμος της Google βαθμολογεί αρνητικά την ύπαρξη σελίδων με ίδιο τίτλο.
- Κάθε ιστοσελίδα πρέπει να περιέχει το όνομα της επιχείρησης ή των προσφερομένων υπηρεσιών εξασφαλίζοντας το branding της εταιρίας.
- Δεν θα πρέπει ο τίτλος να είναι φορτωμένος με επαναλαμβανόμενες λέξεις-κλειδιά.

Κεφάλαιο 7ο

Ασφάλεια

7.1 Εισαγωγή

Στις μέρες μας οι διαδικτυακές συναλλαγές έχουν αυξηθεί πάρα πολύ, με ανάλογη αύξηση και στον αριθμό και στον τύπο των επιθέσεων που δέχεται η ασφάλεια των ηλεκτρονικών συστημάτων. Κάποιες επιθέσεις έχουν στόχο το λογισμικό των καλαθιών αγοράς και κάποιες άλλες χρησιμοποιούν ευπάθειες που είναι κοινές σε οποιαδήποτε δικτυακή εφαρμογή, όπως τη χρήση της SQL ή τη συγγραφή τμημάτων κώδικα και την παράθεσή τους σε διάφορα σημεία του site. Στην ενότητα αυτή θα εξεταστούν κάποιες κοινές ευπάθειες ασφαλείας καθώς και οι τρόποι αντιμετώπισής τους.

7.2 Αλλαγή των δικαιωμάτων των αρχείων

Όλα τα στοιχεία που περιέχονται σε μια ιστοσελίδα (φάκελοι, αρχεία) έχουν ορισμένα δικαιώματα χρήσης. Τα δικαιώματα αυτά, καθορίζουν τι ενέργειες που επιτρέπεται να κάνει ο κάθε χρήστης με το εκάστοτε στοιχείο.

Υπάρχουν τρεις τύποι δικαιωμάτων:

- Read
- Write
- Execute

Τα δικαιώματα είναι ξεχωριστά για τον κάθε χρήστη και ορίζονται από ένα τριψήφιο αριθμό. Τον πιο αυστηρό περιορισμό τον δηλώνει ο αριθμός ***000***, σύμφωνα με τον οποίο κανένας χρήστης δεν έχει δικαίωμα για την εκτέλεση οποιασδήποτε ενέργειας. Εν αντιθέτως, ο τριψήφιος αριθμός *777* δίνει το ελεύθερο για όλες τις χρήσεις.

Το πρώτο ψηφίο του αριθμού αντιπροσωπεύει τα δικαιώματα του ιδιοκτήτη/ δημιουργού(*owner*), το δεύτερο των υπόλοιπων εξουσιοδοτημένων χρηστών(*group*) και το τρίτο, τα δικαιώματα των τρίτων(*all*).

Για την ασφάλεια αλλά και τη χρηστικότητα των στοιχείων ενός ιστοτόπου, τα δικαιώματα των φακέλων θα πρέπει να οριστούν σε 755 και των αρχείων σε 644. Η αλλαγή των δικαιωμάτων πραγματοποιείται μέσω του chmod.

7.3 Χρησιμοποιώντας το αρχείο htaccess.txt

Κατά την εγκατάσταση του Apache HTTP Server δημιουργήθηκε και το αρχείο ***htaccess.txt***, το οποίο όσο βρίσκεται σε αυτή τη μορφή δεν έχει καμία επίπτωση στον διαδικτυακό τόπο. Αν μετονομαστεί σε ***.htaccess*** προκύπτει ένα πολύ ισχυρό εργαλείο, το οποίο με τις κατάλληλες ρυθμίσεις μπορεί:

- να κρατήσει μακριά τους «ανεπιθύμητους επισκέπτες» ή να τους παραπέμψει αλλού
- να προστατέψει τις ιστοσελίδες και τους καταλόγους με κωδικούς πρόσβασης
- να κάνει την ιστοσελίδα φιλική στις μηχανές αναζήτησης (***SEF urls***)

Στη συνέχεια αναλύονται μερικές τεχνικές για το πως μπορεί να αξιοποιηθεί το αρχείο *.htaccess*, ώστε παραμετροποιώντας το, να αυξηθεί η ασφάλεια της ιστοσελίδας.

7.3.1 Αποκλεισμός της IP ενός ανεπιθύμητου επισκέπτη

Σε γενικές γραμμές, μια έγκυρη IP θα πρέπει να έχει τη μορφή *xxx.xxx.xxx.xxx*, όπου "*xxx*" είναι ένας αριθμός μεταξύ *0-255*. Εισάγοντας ένα τμήμα μιας ανεπιθύμητης IP στο αρχείο .htaccess, αποκλείονται όλες οι IP που περιέχουν το εν λόγω τμήμα μέσα σε αυτό το εύρος. Η παραπάνω ρύθμιση εφιστά την απαραίτητη προσοχή, γιατί μπορεί να αποκλειστούν χρήσιμοι και υγιή επισκέπτες (οι «κακές» IP συνήθως είναι *Blacklisted*).

Ας υποθέσουμε πως θέλουμε να αποκλείσουμε τις παρακάτω IPs (οι IPs είναι τυχαίες):

- 147.95.40.13 (Αποκλείει μια συγκεκριμένη διεύθυνση IP)
- 215.153.42. (Αποκλείει όλες τις IPs μέσα στην περιοχή 215.153.42.xxx)
- 93.32. (Αποκλείει όλες τις IPs μέσα στην περιοχή 93.32.xxx.xxx)
- 81.158.3 (Αποκλείει όλες τις IPs μέσα στην περιοχή 81.158.3xx.xxx)

Ο αντίστοιχος κώδικάς των παραπάνω παραδειγμάτων που θα πρέπει να ενσωματωθεί στο αρχείο .htaccess έχει την ακόλουθη μορφή:

```
## USER IP BANNING - BLACKLISTED
<Limit GET POST>
 order allow,deny
 deny from 147.95.40.13
 deny from 215.153.42.
 deny from 93.32.
 deny from 81.158.3
 allow from all
</Limit>
```

7.3.2 Αποκλεισμός ανεπιθύμητου spam traffic από Site Referrers

Σύνηθες φαινόμενο είναι το γεγονός που αρκετές ιστοσελίδες καταγράφουν στο *web site referrer log* (Webalizer) του server τους επισκέψεις από άλλες σελίδες. Οι ιστοσελίδες αυτές αναφέρονται ως *spam sites* και συνηθίζουν να εκτελούν τέτοιες ενέργειες για να καταγράφουν το URL της επίσκεψής τους στο Webalizer του server με απώτερο στόχο να είναι αναγνωρίσιμο από τις μηχανές αναζήτησης σαν *backlink*.

Τα spam sites είναι ανεπιθύμητα γιατί βομβαρδίζουν τον server με επισκέψεις, οι οποίες συνήθως γίνονται με κάποιο *bot* με αποτέλεσμα να μειώνουν το ωφέλιμο *bandwidth* του server. Μέσω του .htaccess μπορεί να απαγορευτεί η είσοδος των bot στην ιστοσελίδα.

Αν για παράδειγμα θέλουμε να αποκλείσουμε τα παρακάτω domains ή τις IPs (και τα domains και οι IPs είναι τυχαίες):

- verybadsite.com (Απαγόρευση επισκέψεων από το verybadsite.com)
- verybadsite. (Απαγόρευση επισκέψεων από όλες τις μορφές verybadsite.xxx, όπως verybadsite.com, verybadsite.net κλπ)
- sub.verybadsite.com (Απαγόρευση επισκέψεων από το sub.verybadsite.com)
- 32.173.21.187 (Απαγόρευση επισκέψεων προερχόμενων από συγκεκριμένη IP του site)

ο κώδικάς με τον οποίο πρέπει να ενημερωθεί το αρχείο .htaccess έχει την παρακάτω μορφή:

```
##SITE REFERRER BANNING
RewriteEngine on
# Options +FollowSymlinks

RewriteCond %{HTTP_REFERER} verybadsite\.com [NC,OR]
RewriteCond %{HTTP_REFERER} verybadsite\. [NC,OR]
RewriteCond %{HTTP_REFERER} sub\.verybadsite\.com [NC,OR]
RewriteCond %{HTTP_REFERER} 32\.173\.21\.187 [NC]
RewriteRule .* - [F]
```

7.3.3 Απενεργοποίηση και απαγόρευση του Hotlinking

Η απενεργοποίηση του *hot linking* είναι μια απαγόρευση για συνηθισμένα κοινά αρχεία από άλλες ιστοσελίδες, έτσι ώστε μόνο το domain(s) μας να μπορεί να αναφέρεται ή να έχει πρόσβαση σε αυτά. Για παράδειγμα, με την απενεργοποίηση του *hot linking* για τα αρχεία .jpg, οποιοδήποτε ιστοσελίδα η οποία δεν είναι μέσα στη λίστα των επιτρεπόμενων domain για την προβολή των αρχείων .jpg που βρίσκονται στο server, παίρνει μια αναφορά ανύπαρκτου αρχείου εικόνας.

Στον παρακάτω κώδικα που χρησιμοποιείται για την απαγόρευση των hot linkingg, τοποθετούνται τα domain και οι τύποι των αρχείων για τα οποία θα γίνεται το *hot linking*. Αν υποθέσουμε ότι θέλουμε να επιτρέψουμε κάποια Domains ή IPs για

κάποιους τύπους αρχείων και να απαγορεύσουμε κάποια αρχεία για *hot linking* από όλα τα υπόλοιπα (οι παρακάτω IPs και τα Domains είναι τυχαία), τότε ο κώδικάς θα έχει την παρακάτω μορφή:

Για "Επιτρεπόμενα Domains/ IPs"

```
## DISABLE HOTLINKING
RewriteEngine on
# Options +FollowSymlinks
RewriteCond %{HTTP_REFERER} !^$
RewriteCond %{HTTP_REFERER} !^http://(www\.)?teiath.gr/.*$ [NC]
RewriteCond %{HTTP_REFERER} !^http://(www\.)?209.62.45.34/.*$ [NC]
```

- teiath.gr (Επιτρέπετε σε αυτό το domain η πρόσβαση στους συγκεκριμένους τύπους αρχείων)
- 209.62.45.34 (Επιτρέπετε σε αυτή την IP η πρόσβαση στους συγκεκριμένους τύπους αρχείων)

Για τη "Λίστα τύπου Αρχείων"

```
RewriteRule \.(gif|jpg|png|css|js)$ - [F]
```

- gif (Απαγόρευση του hot linking στα .gif αρχεία σε αυτόν τον server)
- jpg (Απαγόρευση του hot linking στα .jpg αρχεία σε αυτόν τον server)
- png (Απαγόρευση του hot linking στα .jpg αρχεία σε αυτόν τον server)
- css (Απαγόρευση του hot linking στα css αρχεία σε αυτόν τον server, αντί αυτού να εμφανίζετε κενό αρχείο)
- js (Απαγόρευση του hot linking στα .js αρχεία σε αυτόν τον server, αντί αυτού να εμφανίζετε κενό αρχείο)

7.3.4 SEF urls

Η URL διεύθυνση είναι ένα από τα σημαντικότερα στοιχεία κάθε ιστοσελίδας. Σπάνια δίνεται ιδιαίτερη προσοχή στην σωστή επιλογή και διαμόρφωση της. Στην υποενότητα αυτή θα αναλυθεί η σημασία των URL διευθύνσεων και οι τρόποι για την

καλύτερη αξιοποίηση τους τόσο για την προσέλκυση επισκεπτών όσο και για την καλύτερη κατάταξη στα αποτελέσματα των μηχανών αναζήτησης.

Δυναμικές & στατικές URL διευθύνσεις

Υπάρχουν δύο κατηγορίες URL διευθύνσεων : οι δυναμικές και οι στατικές. Οι στατικές URL διευθύνσεις είναι οι διευθύνσεις εκείνων των ιστοσελίδων στις οποίες το περιεχόμενο παραμένει το ίδιο μέχρι την στιγμή που κάποιος θα πραγματοποιήσει αλλαγές στον HTML κώδικα. Ένα παράδειγμα στατικής διεύθυνσης είναι:

http://www.kratisinow.gr/reviews/trips/bansko.htm

Οι δυναμικές URL διευθύνσεις είναι οι διευθύνσεις εκείνων των ιστοσελίδων που παράγονται με αυτόματο τρόπο από κάποια βάση δεδομένων. Σε αυτή την κατηγορία ανήκουν οι ιστοσελίδες των ηλεκτρονικών καταστημάτων, των forums, των blogs και των συστημάτων διαχείρισης περιεχομένου, μεταξύ άλλων. Ένα παράδειγμα δυναμικής διεύθυνσης είναι :

http://search.barnesandnoble.com/booksearch/isbnInquiry.asp**?userid=**OB1JMx5DkB**&isbn=**0385504209**&itm=**1

Οι URL διευθύνσεις των ιστοσελίδων θα πρέπει να είναι σύντομες και περιγραφικές. Σύντομες για να μπορούν οι χρήστες να τις απομνημονεύουν ευκολότερα και περιγραφικές για να δίνουν με μια πρώτη ματιά στους επισκέπτες στοιχεία για το περιεχόμενο τους. Παράλληλα θα πρέπει να είναι διαμορφωμένες με τέτοιον τρόπο ώστε να διευκολύνουν την διαδικασία της καταγραφής και καταχώρισης από τις μηχανές αναζήτησης.

Προβλήματα με τις δυναμικές URL διευθύνσεις

Εκεί που υπάρχει το μεγαλύτερο πρόβλημα τόσο στους επισκέπτες όσο και στις μηχανές αναζήτησης, είναι στις URL διευθύνσεις των δυναμικών ιστοσελίδων. Όταν ένας επισκέπτης βλέπει μια URL διεύθυνση όπως η

http://www.kratisinow.gr/reviews/trips/bansko.htm

γνωρίζει ότι η ιστοσελίδα αυτή είναι πολύ πιθανό να περιέχει σχόλια και παρατηρήσεις για τον τουριστικό προορισμό «Bansko». Αν συγκρίνουμε αυτή την URL διεύθυνση με την παρακάτω διεύθυνση:

http://search.barnesandnoble.com/booksearch/isbnInquiry.asp**?userid**=OB1JMx5DkB**&isbn**=0385504209**&itm**=1

αντιλαμβανόμαστε ότι το παραπάνω URL δεν είναι τόσο κατανοητό όσο αυτό του πρώτου παραδείγματος. Το σύμβολο «**?**» δηλώνει ότι πρόκειται για δυναμική ιστοσελίδα ενώ οι «**userid=**», «**isbn=**» και «**itm=**» είναι δυναμικοί παράμετροι που «τραβάνε» στοιχεία από κάποια ή κάποιες βάσεις δεδομένων.

Οι crawlers των μηχανών αναζήτησης όταν συναντάνε τέτοιου είδους URL διευθύνσεις γνωρίζουν ότι πρόκειται για δυναμικές ιστοσελίδες. Αυτό που δεν γνωρίζουν είναι εάν το website έχει 100 ή 1.000.000 ιστοσελίδες που να δημιουργούνται με στοιχεία που προέρχονται από τις βάσεις δεδομένων. Γι αυτό και κάποιες μηχανές αναζήτησης αντιμετωπίζουν προβλήματα με την καταγραφή και την καταχώριση αυτών των ιστοσελίδων αφού τόσο ο χρόνος όσο και δυνατότητα επεξεργασίας που αφιερώνουν σε κάθε website δεν μπορεί να είναι απεριόριστη.

Παρά το γεγονός ότι με το πέρασμα των χρόνων οι μηχανές αναζήτησης γίνονται περισσότερο ικανές στην αντιμετώπιση τέτοιων περιπτώσεων, ο γενικός κανόνας είναι ότι δεν θα πρέπει να χρησιμοποιούνται περισσότερες από δύο δυναμικές παραμέτρους.

Επίσης είναι προτιμότερο να μην υπάρχουν τα σύμβολα «**?**» και «**&**» στις δυναμικές URL διευθύνσεις. Αυτό μπορεί να γίνει εφικτό με την χρήση της τεχνικής που ονομάζεται **URL Rewritting**. Η συγκεκριμένη τεχνική εφαρμόζεται στον web server που φιλοξενεί τις ιστοσελίδες και ουσιαστική λειτουργεί ως μεταφραστής των δυναμικών URL διευθύνσεων σε διευθύνσεις που μοιάζουν να είναι στατικές και είναι πολύ περισσότερο φιλικές στις μηχανές αναζήτησης.

URL Rewritting

Το mod_rewrite πρόκειται για μία λειτουργική μονάδα/ένθεμα (module) του διακομιστή Apache για χειρισμό ζητούμενων υπερσυνδέσμων σε επίπεδο διαμοιραστή (server-side). Τα εισερχόμενα URLS φιλτράρονται μέσω διαφόρων κανόνων και όταν γίνει η σύγκριση σε επίπεδο server, τότε αντικαθίσταται ο χαρακτήρας του URL με τον επιθυμητό (URL Rewritting).

Η ενεργοποίηση του mod_rewrite ή οποιουδήποτε module του apache πρέπει να γίνει από το αρχείο καθολικών μεταβλητών httpd.conf (global configuration file). Οι περισσότεροι hosting providers έχουν ενεργοποιημένη την υπηρεσία.

Ο απλούστερος τρόπος να ελέγξουμε αν είναι ενεργοποιημένο είναι μέσω της παρακάτω php εντολής.

```
<?php phpinfo(); ?>
```

Στην περιοχή των ενθεμάτων που τρέχουν (Loaded_modules) πρέπει να εμφανίζεται το mod_rewrite *(Εικόνα 7.1)*.

apache

APACHE_INCLUDE	
APACHE_TARGET	
Apache Version	Apache/1.3.20
Apache Release	10320100
Apache API Version	19990320
Hostname:Port	midas:80
User/Group	wwwrun(30)/65534
Max Requests	Per Child: 100 - Keep Alive: on - Max Per Connection: 100
Timeouts	Connection: 300 - Keep-Alive: 15
Server Root	/usr/local/httpd
Loaded Modules	mod_userdir, mod_php4, mod_perl, mod_setenvif, mod_so, mod_unique_id, mod_usertrack, mod_headers, mod_expires, mod_cern_meta, mod_digest, mod_auth_db, mod_auth_dbm, mod_auth_anon, mod_auth, mod_access, mod_rewrite, mod_alias, mod_proxy, mod_speling, mod_actions, mod_imap, mod_asis, mod_cgi, mod_dir, mod_autoindex, mod_include, mod_info, mod_status, mod_negotiation, mod_mime, mod_mime_magic, mod_log_referer, mod_log_agent, mod_log_config, mod_define, mod_env, mod_vhost_alias, mod_mmap_static, http_core

Εικόνα 7.1: *Apache modules*

Πλεονεκτήματα του mod_rewrite

Η βασική λειτουργία του mod_rewrite παρέχει τη δυνατότητα της ανακατεύθυνσης, χωρίς να το γνωρίζει ο χρήστης. Συνηθέστερα, χρησιμοποιείται τόσο για να μετατρέψει δύσχρηστα και ακατανόητα URLS σε πιο ευανάγνωστα και κατανοητά, όσο και στην ανακατανομή παλιών διευθύνσεων URL σε νέες διευθύνσεις. Σαν άμεσο αποτέλεσμα, τα URLS αυτά είναι περισσότερο φιλικά (Search Engine Friendly (SEF)) τόσο για τους αναγνώστες όσο και για τις μηχανές αναζήτησης.

- *Όχι τόσο φιλικό URL*: http://example.gr/user.php?id=5476
- *Πιο φιλικό*: http://example.gr/user/5476/
- *Ακόμα καλύτερα*: http://example.gr/user/george/

Το τελευταίο βελτιωμένο URL δεν είναι μόνο πιο κατανοητό στον απλό χρήστη αλλά έχει και σημασιολογική έννοια.

Επεκτείνοντας το παραπάνω παράδειγμα, πρέπει να επισημανθεί πως ο μετασχηματισμός φιλικών URLs μέσω του mod_rewrite συμβάλλει στην ασφάλεια του ιστοτόπου. Ας υποθέσουμε ότι έχουμε το παρακάτω URL:

```
http://example.gr/user.php?id=XY
```

Θεωρητικά, ένας κακόβουλος χρήστης μπορεί να περάσει την παράμετρο XY στο id, αποσκοπώντας λανθασμένη λειτουργία του scipt που εκτελείται. Κάνοντας όμως χρήση του mod_rewrite, το URL που παράγεται δεν θα μπορούσε να χρησιμοποιηθεί και να τροποποιηθεί για τυχόν κακόβουλες ενέργειες.

```
http://example.com/user/XY/
```

Όπως αναφέρθηκε, για να δημιουργηθούν τα SEF URLs θα πρέπει να ενημερωθεί ο Server ώστε να μεταφράζει τα URL από τη μια μορφή στην άλλη. Αυτή η διαδικασία είναι λίγο δυσνόητη επειδή απαιτεί γνώση των Regular Expressions. Τα Regular Expressions είναι ένας τρόπος να εντοπίζονται μοτίβα χαρακτήρων μέσα σε σειρές κειμένων ή/και σε σώματα κειμένων. Στους υπολογιστές, τα Regular expressions, που επίσης αποκαλούνται και regex ή regexp παρέχουν ένα σύντομο και

ευέλικτο τρόπο για τον εντοπισμό μοτίβων μέσα σε σώματα κειμένων, όπως για παράδειγμα συγκεκριμένους χαρακτήρες, λέξεις, ή συνδυασμούς χαρακτήρων. Τα Regular Expressions γράφονται σε μια τυποποιημένη γλώσσα η οποία μπορεί να μεταφραστεί από ένα επεξεργαστή για Regular Expressions.

Ακολουθεί μια λίστα με κλάσεις ειδικών χαρακτήρων.

\ : Χαρακτήρας διαφυγής. Χρησιμοποιείται για να εντοπιστούν σύμβολα που είναι ειδικοί χαρακτήρες

. : Η τελεία χρησιμοποιείται για να εντοπίσει κάθε είδους χαρακτήρα εκτός του χαρακτήρα νέας γραμμής

x : Εντοπίζει κάθε στιγμιότυπο του χαρακτήρα **x**

^x : Εντοπίζει κάθε χαρακτήρα εκτός του **x**

[x] : Εντοπίζει κάθε χαρακτήρα που είναι μέσα στις αγκύλες – [adfpq] εντοπίζει κάθε στιγμιότυπο των a, d, f, p και q

| : Τελεστής OR – [a|g] εντοπίζει κάθε στιγμιότυπο του a ή του g

() : Χρησιμοποιείται για να ομαδοποιήσει ακολουθίες χαρακτήρων ή αποτελεσμάτων. Επίσης στον Apache αυτό δημιουργεί και μια μεταβλητή της μορφής $1 με αριθμούς από 1 μέχρι 9. Έτσι αν σε ένα regular expression υπάρχουν δυο παρενθέσεις, αυτόματα παράγονται και δυο μεταβλητές της μορφής $1 και $2, όπου μπορούν να χρησιμοποιηθούν μέσα στο ίδιο regular expression.

{} : Χρησιμοποιείται για να περιορίσει το πλήθος των χαρακτήρων που αναζητούνται.

{x} : Το αποτέλεσμα θα πρέπει να έχει μήκος ακριβός **x** χαρακτήρες.

{x,} : Το αποτέλεσμα θα πρέπει να έχει τουλάχιστον **x** χαρακτήρες

{x,y} : Το αποτέλεσμα θα πρέπει να έχει τουλάχιστον **x** χαρακτήρες αλλά όχι περισσότερους από **y**

? : Το προηγούμενο ταίριασμα είναι προαιρετικό ή θα πρέπει να εμφανίζετε μόνο μια φορά. Ισοδύναμο με το {0,1}

* : Εντοπίζει 0 ή και περισσότερους χαρακτήρες από το προηγούμενο ταίριασμα. Ισοδύναμο με το {0,}

+ : Εντοπίζει 1 ή και περισσότερους χαρακτήρες από το προηγούμενο ταίριασμα. Ισοδύναμο με το {1,}

^ : Εντοπίζει την αρχή κάθε γραμμής.

$: Εντοπίζει το τέλος κάθε γραμμής.

Παρακάτω παρουσιάζονται ορισμένα παραδείγματα Regular Expressions:

Διευθύνσεις IP : ([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})\.([0-9]{1,3})
Domain names : ^[a-zA-Z]([a-zA-Z0-9-]{0,61}[a-zA-Z0-9])?\.[a-zA-Z]([a-zA-Z0-9-]{0,61}[a-zA-Z0-9])?(\.[a-zA-Z]([a-zA-Z0-9-]{0,61}[a-zA-Z0-9])?)?$
Διευθύνσεις email : {^[A-Za-z0-9._-]+@[[A-Za-z0-9.-]+$}

Για να τεθεί σε ισχύ το mod_rewrite πρέπει να ενημερωθεί το αρχείο .htaccess. Μέσα σε αυτό καταγράφονται οι κατάλληλες εντολές για την μετατροπή των SEF URL σε URL που μόνο η εφαρμογή μπορεί να διαχειριστεί.

Αρχικά πρέπει να ενεργοποιηθεί το mod_rewrite. Αυτό γίνεται με την εντολή:

```
RewriteEngine on
```

Στη συνέχεια με την εντολή RewriteRule μπορεί να μετατραπούν τα URLs σε SEF URLs. Η εντολή RewriteRule δέχεται τρία ορίσματα. Το πρώτο όρισμα είναι το Regular Expression που θα εντοπίσει τη μορφή του URL που είναι προς επεξεργασία, το δεύτερο όρισμα είναι το αρχείο το οποίο θα επεξεργαστεί το SEF URL και το τρίτο όρισμα είναι προαιρετικό και δέχεται διάφορα Flags για επεξεργασία του Rewrite. Έτσι το RewriteRule παίρνει την παρακάτω μορφή:

```
RewriteRule RegularExpression FileToRedirect.ext Flags
```

Στο παράδειγμα που ακολουθεί

```
RewriteEngine On
RewriteRule ^clubs/?$ clubs.php [L]
```

Το Regular Expression αρχίζει με το σύμβολο ^ και την λέξη clubs ενωμένα. Αυτό ενημερώνει τον Apache διακομιστή πως πρέπει να δώσει προσοχή σε κάθε URL που αρχίζει με τη λέξη clubs.

Στη συνέχεια το Regular Expression τελειώνει με ένα ερωτηματικό και ένα σύμβολο δολαρίου. Τα δυο αυτά σύμβολα μας υποχρεώνουν να μην ακολουθεί άλλος χαρακτήρας μετά τον χαρακτήρα slash (/) . Για παράδειγμα το clubs/example θα δημιουργούσε ένα σφάλμα 404 γιατί αυτό το URL δεν ταιριάζει με το Regular Expression.

Με αυτό τον τρόπο, το URL http://www.example.com/clubs/ αντιστοιχεί στο http://www.example.com/clubs.php, που έχει σαν αποτέλεσμα την δημιουργία ενός φιλικού URL.

7.4 Captcha

Το όνομα αποτελεί ακρωνύμιο για το ***C**ompletely **A**utomated **P**ublic **T**uring test to tell **C**omputers and **H**umans **A**part* που πρακτικά περιγράφει ένα είδος αυτόματου τεστ με την δυνατότητα να ξεχωρίζει αν ο χρήστης είναι άνθρωπος ή μηχανή.

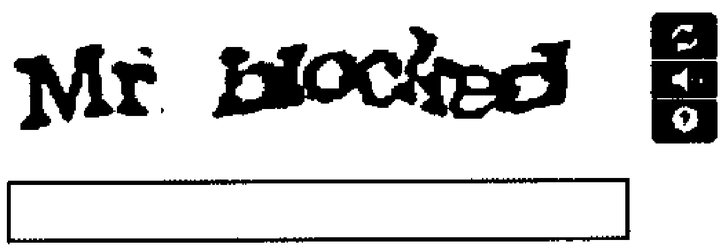
Η εξέλιξη και η διάδοση του internet είχαν ως αποτέλεσμα το γνωστό φαινόμενο του *spamming*, ανεπιθύμητων ηλεκτρονικών μηνυμάτων που στέλνονται κατά χιλιάδες και βομβαρδίζουν τους χρήστες με ανούσιες πληροφορίες και διαφημίσεις κάθε φορά που διαβάζουν email ή σχόλια σε blog.

Συνήθως η αποστολή *spam* απαιτεί ένα λογαριασμό *email* και ένα πρόγραμμα που στέλνει καθημερινά χιλιάδες μηνύματα. Οι spammers ανακάλυψαν τις δωρεάν υπηρεσίες email (*Yahoo!*, *Hotmail*, *Gmail* κλπ) και είδαν ότι με την χρήση κατάλληλου λογισμικού εύκολα μπορούσαν να δημιουργούν χιλιάδες δωρεάν λογαριασμούς και να αποστέλλουν αυτόματα χιλιάδες μηνύματα.

Η λύση βρέθηκε το 2000 με την δημιουργία και χρήση των πρώτων *captcha*, αυτόματων τεστ στα οποία το σημερινό λογισμικό αποτυγχάνει ενώ οι άνθρωποι

παιρνούν με ευκολία. Ο χρήστης καλείται να αναγνωρίσει και να πιστοποιήσει πληροφορία που παρέχει το τεστ συπληρώνοντας συνήθως κάποια φόρμα.

Τα *captcha* εμφανίζονται στο internet ως παραμορφωμένες εικόνες που παρουσιάζουν συνδυασμούς γραμμάτων και αριθμών σε διάφορα μεγέθη, χρώματα κλπ *(Εικόνα 7.2)*. Όταν ο χρήστης θέλει να δημιουργήσει ένα νέο email ή να σχολιάσει άρθρα σε κάποιο blog, καλείται να αναγνωρίσει την εικόνα και το περιεχόμενο της και να συμπληρώσει σωστά τη φόρμα. Διαδικασία απλή, σχεδόν για όλους τους ανθρώπους, όχι όμως και για τις δυνατότητες των σύγχρονων προγραμμάτων στα χέρια των spammers.



Εικόνα 7.2: *Captcha*

7.5 PHP Sessions

Όταν μια ιστοσελίδα γίνεται όλο και πιο πολύπλοκη, γίνεται πολύπλοκος και ο κώδικας που την υποστηρίζει. Μια κανονική ιστοσελίδα HTML δεν παρέχει την δυνατότητα μεταφοράς δεδομένων από τη μία σελίδα στην άλλη. Αυτό δημιουργεί πρόβλημα για εργασίες όπως ένα καλάθι αγορών, το οποίο απαιτεί τη μεταφορά δεδομένων (επιλεγμένο προϊόν του χρήστη) για την ολοκλήρωση της διαδικασίας.

Όταν η ιστοσελίδα απαιτεί τη μεταφορά δεδομένων του χρήστη από τη μία σελίδα στην άλλη, τότε γίνεται λόγος για την χρήση των PHP Sessions. Τα Sessions δημιουργούν ένα μοναδικό κωδικό (Unique, IDentifier) για κάθε επισκέπτη, έτσι ώστε να σωθούν συγκεκριμένες μεταβλητές.

Σύνταξη των PHP Sessions

Η εκκίνηση των Sessions γίνεται με την εντολή *session_start()*. Η λειτουργία *session_start()* τοποθετείται στην αρχή του κώδικα PHP, πριν να σταλεί κάποιο HTML ή άλλο κείμενο.

Παρακάτω φαίνεται η απλή δέσμη ενεργειών που θα πρέπει να τοποθετηθεί στην αρχή του κώδικα PHP για να ξεκινήσει ένα PHP Session.

```
<?php
session_start(); // start up your PHP session!
?>
```

Αυτό το κομμάτι του κώδικα καταχωρεί το Session του χρήστη στον browser και επιτρέπει την αποθήκευση πληροφοριών και την εκχώρηση ενός UID (μοναδικό αναγνωριστικό αριθμό) για τη συνεδρία του συγκεκριμένου χρήστη.

Δήλωση ενός Session

Για την αποθήκευση και την ανάκτηση των δεδομένων ενός χρήστη σε μια σύνοδο χρησιμοποιείται το *$_SESSION*.

```
<?php
session_start();
$_SESSION['user_id'] = 1; // store session data
echo "User ID = ". $_SESSION['user_id']; //retrieve data
?>
```

Χρήση της λειτουργίας isset της PHP

Κάθε φορά που απαιτείται η χρήση των Session από πολλαπλές σελίδες, απαιτείται και ο έλεγχος της εγκυρότητά τους. Η λειτουργία *isset()* της PHP παρέχει τη δυνατότητα να λαμβάνει κάθε μεταβλητή που χρησιμοποιήθηκε και να ελέγχει αν της έχει ήδη ανατεθεί μια τιμή.

Εμπλουτίζοντας το προηγούμενο παράδειγμα, στη συνέχεια δημιουργήθηκε μια πολύ απλή προβολή σελίδας με τη χρήση της *isset()*. Αρχικά, ελέγχοντας αν στη μεταβλητή user_id έχει καταχωρηθεί μια τιμή, τότε προσαυξάνεται κατά μια μονάδα, εναλλακτικά καταχωρείται η τιμή ένα.

```php
<?php
session_start();
if(isset($_SESSION['user_id']))
{
	$_SESSION['user_id'] = $_SESSION['user_id']+ 1;
}
else
{
	$_SESSION['user_id'] = 1;
}
echo "User ID = ". $_SESSION['user_id'];
?>
```

Η πρώτη φορά που εκτελείτε αυτήν τη δέσμη ενεργειών σε ένα πρόσφατα ανοιγμένο browser, η δήλωση if θα αποτύχει διότι καμία μεταβλητή Session δεν έχει ορισθεί ακόμα. Ωστόσο, μετά την ανανέωση της σελίδας, η συνθήκη if θα είναι αληθής και ο μετρητής θα αυξηθεί κατά ένα. Κάθε φορά που επαναλαμβάνεται αυτό το σενάριο η τιμή του user_id αυξάνεται κατά ένα.

Καθαρισμός και Καταστροφή της Session

Αν και τα δεδομένα Session είναι προσωρινά και δεν χρειάζεται να πραγματοποιηθεί διαγραφή τους, ορισμένες διεργασίες απαιτούν την άμεση καταστροφή τους. Η διαδικασία αυτή πραγματοποιείται με την εντολή *unset()*.

```php
<?php
session_start();
if(isset($_SESSION['user_id']))
	unset($_SESSION['user_id']);
?>
```

Τέλος, με την κλήση της λειτουργίας *session_destroy()* μπορούν να καταστραφούν εξολοκλήρου τα Session που έχουν δημιουργηθεί από την εκάστοτε διεργασία που εκτελείται.

```php
<?php
session_start();
session_destroy();
?>
```

7.6 SQL injection

Σήμερα παρατηρείται μια πληθώρα εφαρμογών ιστού που καλύπτουν ένα ευρύ φάσμα δραστηριοτήτων. Ενδεικτικά μπορούν να αναφερθούν συναλλαγές ηλεκτρονικού εμπορίου (e-commerce), ηλεκτρονικές τραπεζικές συναλλαγές (e-banking), υπηρεσίες ηλεκτρονικής διακυβέρνησης (e-government), ηλεκτρονικές ψηφοφορίες (e-voting), ή ηλεκτρονικές υπηρεσίες υγείας (e-health).

Στην ουσία οι εφαρμογές ιστού στηρίζονται σε μια αρχιτεκτονική πελάτη (client) - διακομιστή (server) που αλληλεπιδρούν μέσω του πρωτοκόλλου HTTP (Hypertext Transfer Protocol). Από την πλευρά του πελάτη υπάρχει τυπικά ένας φυλλομετρητής ιστού (web browser), ενώ από την πλευρά του διακομιστή υπάρχουν κατανεμημένοι διακομιστές εφαρμογών (application servers), που συνδέονται με πολλαπλές πηγές δεδομένων. Ο χρήστης αλληλεπιδρά με την εφαρμογή ιστού, στέλνοντας τις επιλογές ή τα δεδομένα του. Η αλληλεπίδραση αυτή μπορεί να κυμανθεί από μια απλή αναζήτηση δεδομένων, έως μεγάλες ενδοεπιχειρησιακές εφαρμογές που εκτελούν σε πραγματικό χρόνο πωλήσεις και διαχείριση απογραφής.

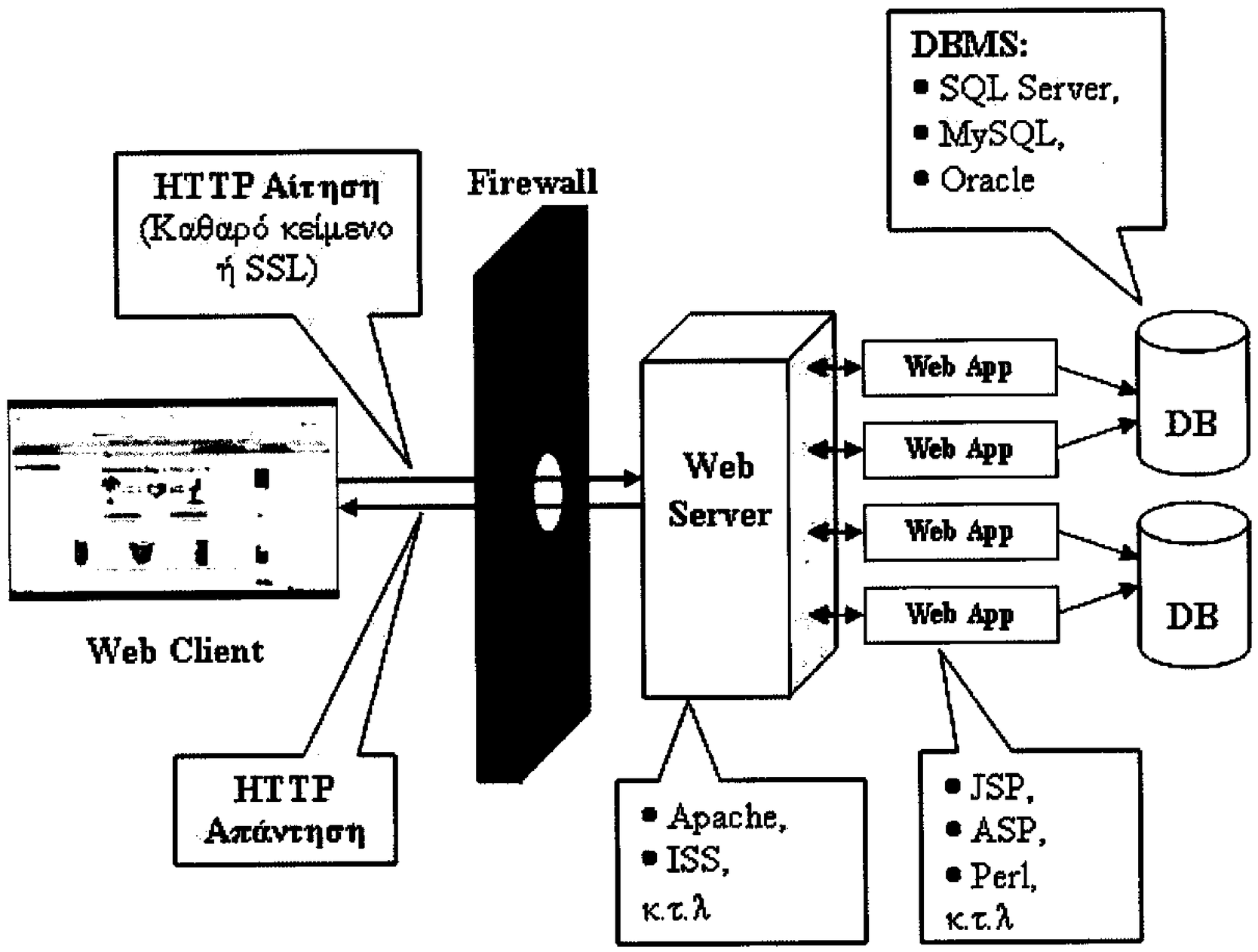
Στην εικόνα 7.3 παρουσιάζεται ένα τυπικό παράδειγμα εφαρμογής ιστού, όπου ο πελάτης αλληλεπιδρά με μια εφαρμογή, που βρίσκεται εγκατεστημένη στην πλευρά του διακομιστή, στέλνοντας αιτήσεις και αναμένοντας την απάντηση της εφαρμογής. Η εφαρμογή ιστού αλληλεπιδρά με την υποκείμενη βάση δεδομένων για να ανακτήσει τα δεδομένα που προκύπτουν από την επεξεργασία του αιτήματος και στη συνέχεια επιστρέφει τα δεδομένα αυτά, ώστε να εμφανιστούν στο περιβάλλον του χρήστη. Η πρόσβαση στη βάση δεδομένων γίνεται με την εκτέλεση δυναμικών SQL ερωτημάτων που δημιουργούνται κατά το χρόνο εκτέλεσης και διαμορφώνονται ανάλογα με τα δεδομένα του χρήστη σε πεδία εισόδου της εφαρμογής.

Οι επιτιθέμενοι μπορούν να εκμεταλλευθούν τη διαδικασία αυτή, τροποποιώντας συντακτικά ή σημασιολογικά τα γνήσια SQL ερωτήματα που περνούν προς τη βάση δεδομένων, ενσωματώνοντας κακόβουλο SQL κώδικα στα πεδία εισόδου. Αυτό το είδος επίθεσης ονομάζεται *δηλητηρίαση SQL κώδικα (SQL code poisoning)* ή *SQL έγχυση (SQL injection)* και μπορεί να οδηγήσει στην έκθεση του διακομιστή βάσεων δεδομένων.

Μια ευπαθής εφαρμογή ιστού επιτρέπει στον επιτιθέμενο να εκμεταλλευτεί ευπάθειες στην ασφάλεια μιας εφαρμογής ιστού με αποτέλεσμα να έχει πρόσβαση

στις βάσεις δεδομένων, να μεταβάλει και να διαγράψει πληροφορίες, ή ακόμα να θέσει την εφαρμογή εκτός λειτουργίας, προκαλώντας τεράστια προβλήματα με πολύ μικρή προσπάθεια και στοιχειώδης τεχνικές γνώσεις. Ως μέθοδος επίθεσης δεν είναι νέα ούτε πολύπλοκη.

Παραδείγματα τέτοιων επερωτήσεων είναι η προβολή (SELECT) δεδομένων που δεν έχει προβλεφθεί να προβάλλονται στους χρήστες, η ανεξέλεγκτη εισαγωγή (INSERT) δεδομένων στη βάση και η διαγραφή (DELETE) δεδομένων από τη βάση.



Εικόνα 7.3: *Τυπική αρχιτεκτονική εφαρμογής ιστού*

Οι επιθέσεις δηλητηρίασης SQL κώδικα είναι ανεξάρτητες από τη γλώσσα ανάπτυξης της εφαρμογής και το σύστημα διαχείρισης βάσεων δεδομένων. Συνεπώς, δεν έχει σημασία αν η εφαρμογή υλοποιήθηκε με χρήση Active Server Pages (ASP), Java Server Pages (JSP), PHP ή Perl, ούτε αν χρησιμοποιείται Oracle, Microsoft SQL Server, IBM DB2, MySQL ή Informix ως βάση δεδομένων. Η εφαρμογή μπορεί να είναι ευπαθής σε επιθέσεις αυτού του τύπου άσχετα από την τεχνολογία που χρησιμοποιεί, αν και κάποια προϊόντα θεωρούνται πιο ανθεκτικά σε σύγκριση με κάποια άλλα.

Παρά το αυξημένο ρίσκο, θεωρείται σήμερα μεγάλος ο αριθμός των συστημάτων τα οποία είναι ευάλωτα στις επιθέσεις SQL Injections. Το ευχάριστο είναι ότι ως απειλή εύκολα μπορεί να διερευνηθεί σε μια εφαρμογή και με κατάλληλη πρόβλεψη να αποφευχθεί.

Τρόποι εκδήλωσης

Οι τρόποι εκδήλωσης μιας τέτοιας επίθεσης μέσω SQL injections που έχουν καταγραφεί μέχρι σήμερα είναι πολλαπλοί. Ο κάθε τύπο επίθεσης μπορεί να προκαλέσει διαφορετικές συνέπειες σε κάθε σύστημα που βρίσκεται στο στόχο της επίθεσης. Πρέπει να σημειωθεί ότι γενικά οι διάφοροι τύποι επίθεσης δεν χρησιμοποιούνται μεμονωμένα. Η συνήθης πρακτική ορίζει δύο ή περισσότεροι τύποι επίθεσης δηλητηρίασης SQL κώδικα να συνδυάζονται ώστε να πετύχουν το επιθυμητό από τον επιτιθέμενο αποτέλεσμα.

Στα SQL injections ένας κακόβουλος χρήστης προσπαθεί να εκμεταλλευτεί την ελλιπή ή λανθασμένη επαλήθευση (validation) των δεδομένων εισόδου (input data) μιας εφαρμογής. Τυπικά η ευπάθεια των SQL injections μπορεί να εμφανίζεται και σε μη εφαρμογές ιστού, όμως εκεί είναι σαφώς πιο σπάνιο. Συνήθεις τρόποι όπου οι χρήστες δίνουν δεδομένα εισόδου σε μια εφαρμογή ιστού είναι οι φόρμες (με τις μεθόδους HTTP GET και POST) και τα links (μέθοδος HTTP GET). Στην περίπτωση ελλιπούς επαλήθευσης των δεδομένων εισόδου είναι σε κάποιες περιπτώσεις δυνατόν να περαστούν extra εντολές SQL προς εκτέλεση στην εφαρμογή ή να αλλαχτούν μέρη μιας επερώτησης SQL με τρόπο που δεν έχει προβλεφθεί.

Χαρακτηριστικά, οι επιτιθέμενοι θα καθορίσουν αρχικά εάν μια περιοχή είναι τρωτή σε μια τέτοια επίθεση με την αποστολή ενός χαρακτήρα ενιαίου-αποσπάσματος ('). Τα αποτελέσματα από μια επίθεση στην SQL σε μια τρωτή περιοχή μπορούν να κυμανθούν από ένα λεπτομερές μήνυμα λάθους, το οποίο αποκαλύπτει την τεχνολογία που χρησιμοποιείται, ή επιτρέποντας στον επιτιθέμενο να έχει πρόσβαση στις μη προσβάσιμες περιοχές της εφαρμογής ιστού με την παραποίηση του ερωτήματος σε μια πάντα-αληθή Boolean τιμή. Ακόμα μπορεί και να επιτρέψει την εκτέλεση εντολών διαχείρισης συστημάτων.

Για να γίνει πιο αντιληπτή η παραπάνω περιγραφή, παρουσιάζεται το πρόβλημα μέσω μερικών πρακτικών παραδειγμάτων (για λόγους απλότητας τα παραδείγματα δίνονται σε PHP):

Παράδειγμα 1

Ας θεωρήσουμε την περίπτωση προβολής στοιχείων ενός χρήστη από έναν πίνακα με όνομα «clients» σε μια βάση δεδομένων MySQL.

Το παρακάτω (ευπαθές) block κώδικα PHP εκτελεί μια επερώτηση SELECT στη βάση δεδομένων, με σκοπό την ανάγνωση από τη βάση δεδομένων των στοιχείων ενός συγκεκριμένου πελάτη.

```
<?php
$qry = "SELECT clientid, fullname, mobile, details FROM clients "
.
       "WHERE clientid =" . $_GET[' clientid '];
$result = mysql_query($qry);
?>
```

Σκοπός του προγραμματιστή εδώ είναι η εκτέλεση ερωτήσεων της μορφής:

```
SELECT clientid, fullname, mobile, details FROM clients WHERE
clientid = 3
SELECT clientid, fullname, mobile, details FROM clients WHERE
clientid = 352
SELECT clientid, fullname, mobile, details FROM clients WHERE
clientid = 590
```

όπου το clientid (πρωτεύον κλειδί στον πίνακα clients) είναι τιμή που δίνεται στην πράξη από τον χρήστη της εφαρμογής (μέσω του φυλλομετρητή, με χρήση της μεθόδου GET του HTTP). Για παράδειγμα, στην τυπική περίπτωση, το clientid μπορεί να δίνεται με ένα link της μορφής:

```
http://www.example.com/clients.php?clientid=3
```

Το πρόβλημα είναι ότι η τιμή της παραμέτρου GET «clientid» που δίνεται στο URL, ΔΕΝ επαληθεύεται επαρκώς πριν την εκτέλεση της επερώτησης από τον κώδικα.

Έτσι ένας κακόβουλος χρήστης μπορεί να γράψει το εξής URL (χειρονακτικά) στον φυλλομετρητή:

```
http://www.example.com/clients.php?client=3 OR 1=1
```

όπου θα έχει ως αποτέλεσμα να εκτελεστεί στη βάση δεδομένων η εξής επερώτηση:

```
SELECT clientid, fullname, mobile, details FROM clients WHERE
clientid=3 OR 1=1
```

έτσι όμως ο κλάδος WHERE θα ισχύει για κάθε εγγραφή του πίνακα clients, οπότε επιστρέφονται όλες οι εγγραφές με αποτέλεσμα ο κακόβουλος χρήστης ενδέχεται να δει πληροφορίες που δεν είναι σκόπιμο.

Παράδειγμα 2

Το επόμενο (ευπαθές) block κώδικα PHP, αφορά την φόρμα login μιας web εφαρμογής όπου χρησιμοποιείται από τους χρήστες δίνοντας τα στοιχεία username και password για την είσοδο τους στο σύστημα:

```
<?php
$username = $_POST['username'];
$password = $_POST['password'];
$qry = "SELECT userid FROM users" .
       " WHERE username='$username' AND password='$password'";
$result = mysql_query($qry);
if (mysql_numrows($result) > 0) {
    //log in user...
}
?>
```

Οι τιμές username και password δίνονται με χρήση μιας τυπικής web login φόρμας (με τη μέθοδο HTTP POST). Αν ο κακόβουλος χρήστης στο password πεδίο δώσει την τιμή:

```
bar' OR 1=1 OR username='
```

τότε από τον παραπάνω κώδικα PHP έχουμε την εκτέλεση της επερώτησης:

```
SELECT userid FROM users WHERE
username='foo' AND password='bar' OR 1=1 OR username='';
```

η οποία όμως ισχύει για κάθε εγγραφή του πίνακα (αφού πάντοτε 1=1) με αποτέλεσμα να επιστρέφονται πάντοτε εγγραφές, ανεξάρτητα από το τι έχει εισαχθεί

στα πεδία username και password. Έτσι ενδέχεται η πρόσβαση στην εφαρμογή (login) από άτομα που δεν είναι εξουσιοδοτημένα για κάτι τέτοιο.

Στα δύο προηγούμενα παράδειγμα αναλύθηκε πως ένας κακόβουλος χρήστης μπορεί να προσπαθήσει να αλλάξει μια SQL επερώτηση με τρόπο που δεν έχει προβλεφθεί από τον προγραμματιστή. Φυσικά η επερώτηση θα μπορούσε να αλλαχθεί με διαφορετικούς τρόπους από ότι στα συγκεκριμένα παραδείγματα (τυπική περίπτωση είναι για παράδειγμα η προσπάθεια χρήσης κλάδων UNION της SQL).

Παράδειγμα 3

Το παράδειγμα αυτό, εκμεταλλεύεται τη δυνατότητα εκτέλεση πολλαπλών εντολών SQL ως μια επερώτηση στο σύστημα. Οι εντολές διαχωρίζονται μεταξύ τους με τον χαρακτήρα ; (semicolon – ελληνικό ερωτηματικό). Η εκτέλεση πολλαπλών εντολών ως μια επερώτηση είναι μια standard δυνατότητα στο χώρο των βάσεων δεδομένων, όμως αποτελεί τον μεγαλύτερο κίνδυνο στην περίπτωση των SQL Injections.

Χρησιμοποιώντας το (ευπαθές) block κώδικα του παραδείγματος 1:

```
<?php
$qry = " SELECT clientid, fullname, mobile, details FROM clients
" .
       "WHERE clientid  =" . $_GET['clientid'];
$result = pg_query($qry);
?>
```

ο προγραμματιστής αναμένει δεδομένα εισόδου από links της μορφής:

```
http://www.example.com/clients.php?clientid=3
```

που θα έχει ως αποτέλεσμα την εκτέλεση της επερώτησης:

```
SELECT clientid, fullname, mobile, details FROM clients WHERE
clientid = 3
```

αυτή τη φορά όμως ο κακόβουλος χρήστης δίνει την εξής URL (χειρονακτικά) στον browser:

```
http://www.example.com/clients.php?clientid=3;DELETE FROM clients;
```

και έτσι θα εκτελεστούν οι εξής 2 εντολές:

```
SELECT clientid, fullname, mobile, details FROM clients WHERE
clientid = 3;
DELETE FROM clients;
```

όπου θα έχει ως αποτέλεσμα να διαγραφούν όλα τα δεδομένα του πίνακα clients από τη βάση!

Φυσικά αντί του DELETE του παραδείγματος, οποιαδήποτε άλλη SQL εντολή θα μπορούσε στην περίπτωση αυτή να εκτελεστεί (και αυτό είναι και ένα καλό παράδειγμα του γιατί δεν είναι ασφαλές να συνδέεται η εφαρμογή μας στη βάση δεδομένων με τα στοιχεία ενός χρήστη της βάσης που έχει αυξημένα δικαιώματα πρόσβασης – με αυξημένα δικαιώματα πρόσβασης κάποιος θα μπορούσε να διαγράψει έως και άλλες βάσεις δεδομένων του συστήματος μη σχετιζόμενες με τη συγκεκριμένη εφαρμογή ιστού).

Στα δυο πρώτα παραδείγματα παρουσιάστηκαν μέθοδοι αλλαγής μιας επερώτησης SQL ενώ στο τρίτο παράδειγμα αναλύθηκε πως είναι δυνατόν να περαστούν νέες ολοκληρωμένες εντολές SQL προς εκτέλεση.

Αντιμετώπιση

Για την προστασία των εφαρμογών ιστού από αυτές τις ευπάθειες μπορεί να υιοθετηθούν διάφοροι μηχανισμοί ελέγχου. Αυτοί οι μηχανισμοί ελέγχου είναι υπεύθυνοι για την προφύλαξη όπως:

- Έλεγχος και έγκριση δεδομένων. Χρειάζεται ένας μηχανισμός που θα ελέγχει όλα τα εισερχόμενα δεδομένα για το μήκος τους, τον τύπο τους και τη σύνταξή τους, προτού γίνουν αποδεκτά για αποθήκευση ή παρουσίαση. Πρέπει να απορρίπτονται όλες οι εντελώς άκυρες εισαγωγές δεδομένων, παρά να γίνεται προσπάθεια για την εξυγίανση των επικίνδυνων δεδομένων.
- Να επιβάλλονται όσο το δυνατόν λιγότερα προνόμια στους χρήστες όταν συνδέονται στη βάση του συστήματος.
- Να αποφεύγονται τα λεπτομερή μηνύματα λαθών, τα οποία μπορούν να βοηθήσουν τον επιτιθέμενο.

- Να μη χρησιμοποιούνται απλές μέθοδοι διαφυγής, όπως η addslashes() ή μέθοδοι αντικατάστασης χαρακτήρων όπως η str_replace(“’”, “””). Δεν είναι ασφαλείς κα μπορούν να τις εκμεταλλευτούν οι επιτιθέμενοι. Προτείνεται η χρήση της mysql_real_escape_string() όταν χρησιμοποιείται PHP με MySQL, ή η χρήση της PDO(PHP Data Objects).
- Όταν χρησιμοποιούνται απλοί μηχανισμοί διαφυγής, να σημειωθεί ότι απλές μέθοδοι διαφυγής δεν μπορούν να διαφύγουν από ονόματα πινάκων.

Google

android

Κεφάλαιο 8ο

Android

8.1 Τι είναι το Android

Ο όρος Android παραπέμπει σε ένα λειτουργικό σύστημα ανοικτού κώδικα για συσκευές κινητής τηλεφωνίας τελευταίας γενιάς (smartphones), το οποίο τρέχει τον πυρήνα του λειτουργικού Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την Open Handset Alliance. Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας την συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων από την Google. Στη συντριπτική πλειοψηφία τους οι συσκευές που χρησιμοποιούν Android είναι εξοπλισμένες με οθόνη αφής, αφού το ίδιο το UI (user interface) του λειτουργικού είναι προσαρμοσμένο για χρήση με αφή.

Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής hardware (μερικές από τις οποίες είναι η Google, η Motorola, η T-Mobile ,η HTC, η Samsung, η LG, η Qualcomm κ.α), οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές κινητής τηλεφωνίας. Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της Apache License, μιας ελεύθερης άδειας λογισμικού.

8.2 Ιστορικά - Εκδόσεις και Χαρακτηριστικά

Στις 5 Νοεμβρίου του 2007 ανακοινώθηκε για πρώτη φορά από τα Google Groups ότι:

"[Η Πλατφόρμα] Android – είναι πιο σπουδαία και φιλόδοξη από ένα μόνο τηλέφωνο."

Μια εβδομάδα αργότερα παρουσιάστηκε η πρώτη έκδοση του Android SDK, η οποία χαρακτηρίστηκε όχι σαν ένα τελικό προϊόν, αλλά σαν ένα "First Look SDK", πράγμα που πολλοί δεν είχαν συνειδητοποιήσει. Αρκετοί ήταν αυτοί που ισχυρίστηκαν ότι το Android είναι γεμάτο bugs και έχει έλλειψη τεκμηρίωσης. Η πλειοψηφία όμως παραδέχτηκε ότι το Android δεν είναι πιο προβληματικό από άλλα λογισμικά, σε αντίστοιχα πρώιμο επίπεδο.

Έτσι το Σεπτέμβριο του 2008, η T-Mobile ανακοινώνει την διαθεσιμότητα του T-Mobile G1, του πρώτου έξυπνου κινητού τηλεφώνου βασισμένο στην πλατφόρμα Android. Λίγες μέρες αργότερα (Οκτώβριος 2008), η Google ανακοινώνει την απελευθέρωση του SDK Release Candidate 1.0.

Το κομμάτι του κώδικα που έμεινε κλειστό, αποτελεί και το πακέτο λογισμικού που κυκλοφορεί ανά διαστήματα και συνοδεύεται από ένα «γλυκό» προσωνύμιο, προσφέροντας αρκετά νέα χαρακτηριστικά στους χρήστες. Η δυνατότητα ή όχι της αναβάθμισης σε νεότερη έκδοση είναι καθαρά θέμα κατασκευαστή, αφού η «καθαρή» έκδοση, πρέπει να προσαρμοστεί από τον εκάστοτε κατασκευαστή για τις συσκευές του.

Μέχρι τώρα υπάρχουν 8 επιβεβαιωμένες εκδόσεις Android:

- Έκδοση 1

  Ημερομηνία κυκλοφορίας: 23 Σεπτεμβρίου 2008

- Έκδοση 1.1

  Ημερομηνία κυκλοφορίας: 9 Φεβρουαρίου 2009

- Έκδοση 1.5 με κωδικό όνομα *CupCake*

  Ημερομηνία κυκλοφορίας : 30 Απριλίου 2009

Αλλαγές:

- Δυνατότητα εγγραφής και παρακολούθησης video στο CamCorder mode,
- Ανέβασμα video στο youtube και εικόνες στο Picasa απευθείας από το τηλέφωνο,
- Εικονικό πληκτρολόγιο με πρόβλεψη λέξεων,
- Υποστήριξη Bluetooth A2DP και AVRCP,
- Δυνατότητα αυτόματης σύνδεσης ακουστικών headset σε συγκεκριμένη απόσταση,
- Νέα Widgets και δυνατότητα προσθήκης φακέλων στην αρχική οθόνη,
- Κινούμενα Screen Transitions.

➢ Έκδοση 1.6 με κωδικό όνομα *Donut*

Ημερομηνία κυκλοφορίας : 15 Σεπτεμβρίου 2009

Αλλαγές:

- Βελτιωμένο Android Market,
- Δυνατότητα πολλαπλής επιλογής φωτογραφιών για επεξεργασία από τη Gallery,
- Αναβαθμισμένο voice Search,
- Αναβαθμισμένες δυνατότητες αναζήτησης από την κεντρική οθόνη που πλέον περιλαμβάνει bookmarks, ιστορικό και επαφές,
- Υποστήριξη CDMA/EVDO, 802.1x, VPNs, και text-to-speech.7,
- Υποστήριξη οθόνες αναλύσεων WVGA,
- Βελτίωση ταχύτητας στις εφαρμογές αναζήτησης και την κάμερα,
- Gesture framework και εργαλείο ανάπτυξης GestureBuilder,
- Δωρεάν turn-by-turn πλοήγηση από τη Google.

➢ Έκδοση 2.0, 2.1 με κωδικό όνομα *Eclair*

Ημερομηνία κυκλοφορίας :

- 26 Οκτωβρίου 2009 (2.0)
- 3 Δεκεμβρίου 2009 (2.0.1)
- 12 Ιανουαρίου 2010 (2.1)

Αλλαγές:

- Βελτιστοποίηση ταχύτητας hardware,
- Υποστήριξη για μεγαλύτερες αναλύσεις και μεγέθη οθονών,
- Ανανεωμένο UI,
- Νέο Browser UI και υποστήριξη HTML5,
- Νέα λίστα επαφών,
- Καλύτερο Contrast Ratio για τα Backgrounds,
- Google Maps 3.1.2,
- Microsoft Exchange Server, για υποστήριξη ActiveSync 2.5,
- Ενσωματωμένη υποστήριξη για Flash στην κάμερα,
- Ψηφιακό ζουμ,
- Δυνατότητα αντίληψης Multitouch,
- Βελτιωμένο εικονικό πληκτρολόγιο,
- Bluetooth 2.1,
- Live Wallpapers.

➢ Έκδοση 2.2 με κωδικό όνομα *Froyo*

Ημερομηνία κυκλοφορίας : 20 Μαΐου 2010

Αλλαγές:

- Βελτιστοποίηση στην ταχύτητα του OS, της διαχείριση μνήμης και την γενική απόδοση,
- Ενσωμάτωση του Chrome V8 JavaScript στα Browsers applications,
- Αναβαθμισμένη υποστήριξη Microsoft Exchange,
- Βελτιωμένος Application Launcher με συντομεύσεις για τις εφαρμογές τηλεφώνου και Browser,
- USB Tethering και λειτουργία Wi-Fi Hotspot,
- Επιλογή απενεργοποίησης για δεδομένα μέσω κινητών δικτύων. (Data Access over Mobile Network),
- Αναβαθμισμένο Market με δυνατότητα αυτόματων updates,
- Γρήγορη μετάβαση ανάμεσα στις γλώσσες του πληκτρολογίου,
- Υποστήριξη για αριθμητικούς και αλφαριθμητικούς κωδικούς,
- Επιλογή εγκατάστασης εφαρμογών στην κάρτα μνήμης,
- Υποστήριξη Adobe Flash 10.1,

- Υποστήριξη για οθόνες μεγάλου dpi όπως οθόνες 4 ιντσών 720p.

➢ Έκδοση 2.3 με κωδικό όνομα *GingerBread*

Ημερομηνία Κυκλοφορίας : 6 Δεκεμβρίου 2010

Αλλαγές:

- Ενημερωμένο UI Design,
- Υποστήριξη για πολύ μεγάλα μεγέθη οθονών και αναλύσεων (WXGA και μεγαλύτερες),
- Προεγκατεστημένη υποστήριξη για VoIP telephony,
- Υποστήριξη για WebM/VP8 video playback αλλά και AAC audio encoding,
- Νέα ηχητικά εφέ όπως τα reverb, equalization, headphone virtualization, και bass boost,
- Επανασχεδιασμένο Multi-touch πληκτρολόγιο,
- Λειτουργίες Copy-paste σε όλο το λειτουργικό,
- Αυξημένη υποστήριξη για development,
- Βελτιώσεις ήχου και γραφικών για τους προγραμματιστές παιχνιδιών,
- Προεγκατεστημένη υποστήριξη για περισσότερους αισθητήρες (όπως γυροσκόπιο και βαρόμετρο),
- Download manager για κατέβασμα μεγάλων αρχείων,
- Βελτιωμένη διαχείριση ενέργειας και έλεγχος των εφαρμογών,
- Προεγκατεστημένη υποστήριξη για πολλαπλές κάμερες,
- Μετατόπιση από το YAFFS σε ext4 filesystem,
- Υποστήριξη NFC (Near Field Communication).

➢ Έκδοση 3.0/3.1 με κωδικό όνομα *Honeycomb*

Η Honeycomb είναι η έκδοση 3.0/3.1 αποκλειστικά για ταμπλέτες, έφερε αλλαγές κυρίως στο γραφικό περιβάλλον και πρόσθεσε υποστήριξη πολλαπλών πυρήνων μαζί με βελτιωμένα γραφικά.

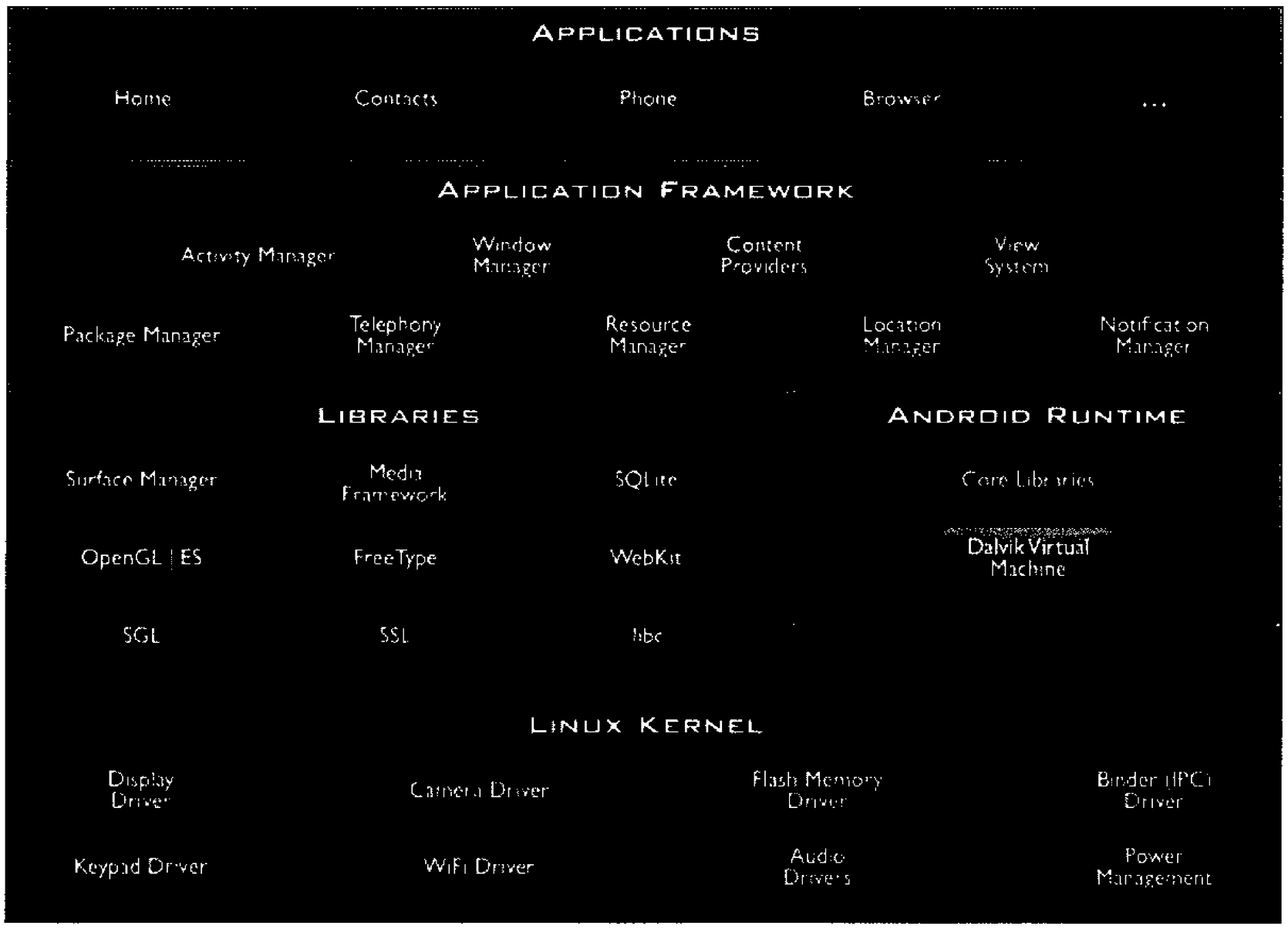
- Έκδοση 4.0 με κωδικό όνομα *Ice Cream Sandwich*
  Η Ice Cream Sandwich έχει ανακοινωθεί ότι σκοπός της είναι να "ενώσει" τις εκδόσεις για ταμπλέτες/κινητά και να προσθέσει υποστήριξη για την Google TV.

8.3 Εργαλεία ανάπτυξης

Ο πιο εύκολος και διαδεδομένος τρόπος ανάπτυξης Android εφαρμογών είναι χρησιμοποιώντας το ολοκληρωμένο περιβάλλον ανάπτυξης εφαρμογών Eclipse IDE και το Android SDK, σε συνδυασμό με το Android Development Tools (ADT) Plugin. Το plugin αυτό έχει σκοπό να επεκτείνει τις δυνατότητες του Eclipse, ώστε ο προγραμματιστής να μπορεί να δημιουργήσει Android projects, να κάνει αποσφαλμάτωση (debug) στον κώδικα, να δημιουργήσει το γραφικό περιβάλλον της εφαρμογής, να εξάγει την τελική εφαρμογή για να την μεταφέρει στην κινητή συσκευή. Όλα αυτά όμως έχουν σαν βασική προϋπόθεση να έχουμε εγκαταστήσει πρώτα στον υπολογιστή μας την JAVA, JDK (Java Development Kit).

Προκειμένου να γίνει ευκολότερη η διαδικασία της ανάπτυξης και αποσφαλμάτωσης μιας εφαρμογής, το Android SDK περιλαμβάνει έναν εξομοιωτή μιας εικονικής κινητής συσκευής η οποία τρέχει το λειτουργικό του Android. Έτσι δεν είναι αναγκαία η ύπαρξη πραγματικής κινητής συσκευής για την εκτέλεση και δοκιμή των εφαρμογών.

8.4 Αρχιτεκτονική του Android



Εικόνα 8.1: *Αρχιτεκτονική Android*

Κάθε επίπεδο στην αρχιτεκτονική του Android *(Εικόνα 8.1)* χρησιμοποιεί τις υπηρεσίες που του προσφέρονται από τα πιο κάτω επίπεδα. Αναλυτικότερα:

- **Applications**

Το Android κυκλοφορεί με ένα σύνολο βασικών εφαρμογών, περιλαμβάνοντας ένα email client, μία εφαρμογή SMS, ημερολόγιο, χάρτες, φυλλομετρητή, επαφές κ.α. Όλες οι εφαρμογές βρίσκονται στο application layer και αναπτύσσονται χρησιμοποιώντας τη γλώσσα προγραμματισμού Java.

- **Application Framework**

Με την παροχή μίας ανοιχτής πλατφόρμας ανάπτυξης, το Android προσφέρει στους developers τη δυνατότητα να κατασκευάσουν εξαιρετικά πλούσιες και καινοτόμες εφαρμογές. Οι developers μπορούν να επωφεληθούν από το υλικό της

συσκευής, να έχουν πληροφορίες για την τοποθεσία, να εκτελούν υπηρεσίες στο παρασκήνιο, να προσθέτουν ειδοποιήσεις στη μπάρα κατάστασης και πολλά ακόμα.

Οι developers έχουν πλήρη πρόσβαση στα ίδια APIs που χρησιμοποιήθηκαν από τις βασικές εφαρμογές. Η αρχιτεκτονική της εφαρμογής έχει σχεδιαστεί για να απλοποιήσει την επαναχρησιμοποίηση των components (συστατικών). Κάθε εφαρμογή μπορεί να δημοσιεύσει τις δυνατότητες της με σκοπό οποιαδήποτε άλλη εφαρμογή να τις χρησιμοποιήσει (με την επιφύλαξη περιορισμών ασφαλείας που επιβάλλονται από το framework). Ο ίδιος μηχανισμός επιτρέπει στο χρήστη να αντικαταστήσει components.

Στη βάση όλων των εφαρμογών βρίσκεται ένα σύνολο από υπηρεσίες και συστήματα, συμπεριλαμβανομένων των εξής:

- Ένα πλούσιο και επεκτάσιμο σύνολο από Views που μπορούν να χρησιμοποιηθούν για τη δημιουργία μιας εφαρμογής, όπως λίστες (lists), πλέγματα (grids), πλαίσια κειμένου (text boxes), κουμπιά (buttons), ακόμα και έναν embeddable web browser.
- Content Providers (Παρόχους Περιεχομένου) που επιτρέπουν στις εφαρμογές να έχουν πρόσβαση σε δεδομένα από άλλες εφαρμογές (όπως οι Επαφές) ή να μοιράζονται τα δικά τους δεδομένα.
- Έναν Resource Manager (Διαχειριστή Πόρων) που παρέχει πρόσβαση σε πηγές χωρίς κώδικα (non-code resources) όπως strings με βάση την τοποθεσία, γραφικά και αρχεία διάταξης (layout files).
- Έναν Notification Manager (Διαχειριστή Ειδοποιήσεων) που επιτρέπει σε όλες τις εφαρμογές να απεικονίζουν προσαρμοσμένες ειδοποιήσεις στη μπάρα κατάστασης.
- Έναν Activity Manager (Διαχειριστή Δραστηριοτήτων) που διαχειρίζεται τον κύκλο ζωής (lifecycle) των εφαρμογών (Σχήμα 1.4) και παρέχει ένα συνηθισμένο ιστορικό μετάβασης (navigation backstack).

- **Libraries**

Το Android περιλαμβάνει ένα σύνολο από βιβλιοθήκες C/C++ που χρησιμοποιούνται από διάφορα components του συστήματος. Αυτές οι δυνατότητες εκτίθενται στους developers μέσω του Android application framework. Από την έκδοση Donut και μετά, οι κατασκευαστές έχουν την δυνατότητα να γράψουν τις δικές τους βιβλιοθήκες κάνοντας χρήση της Εργαλειοθήκης NDK (Native Development Kit). Μερικές από τις βασικές βιβλιοθήκες αναφέρονται παρακάτω:

- *System C library* – μία υλοποίηση προερχόμενη από το BSD, της επίσημης βιβλιοθήκης συστήματος C (libc), βελτιστοποιημένη για ενσωματωμένες συσκευές που βασίζονται στο Linux.
- *Media Libraries* – βασισμένες στο OpenCORE της PacketVideo. Οι βιβλιοθήκες υποστηρίζουν την αναπαραγωγή και καταγραφή πολλών δημοφιλών μορφών ήχου και βίντεο, καθώς και στατικών αρχείων εικόνας. Συμπεριλαμβάνονται τα MPEG4, H.264, MP3, AAC, AMR, JPG, και PNG.
- *Surface Manager* – διαχειρίζεται την πρόσβαση στο υποσύστημα απεικόνισης και συνθέτει στρώματα 2D και 3D γραφικών από πολλαπλές εφαρμογές.
- *LibWebCore* – μία σύγχρονη μηχανή web browser η οποία χρησιμοποιείται και από τον Android browser και από τον embeddable web browser.
- *SGL* – η βασική μηχανή 2D γραφικών.
- *3D libraries* – μια υλοποίηση βασισμένη στα APIs του OpenGL ES 1.0. Οι βιβλιοθήκες χρησιμοποιούν είτε την επιτάχυνση υλικού για 3D (όπου είναι διαθέσιμο) ή το πολύ καλά βελτιστοποιημένο λογισμικό απεικόνισης 3D (rasterizer).
- *FreeType* – bitmap και vector γραμματοσειρά φωτοσκίασης.
- *SQLite* – μία ισχυρή και ελαφριά σχεσιακή βάση δεδομένων, διαθέσιμη σε όλες τις εφαρμογές.

- **Android Runtime**

  - Core Libraries - Το Android περιλαμβάνει ένα σύνολο βασικών βιβλιοθηκών που παρέχουν τις περισσότερες από τις διαθέσιμες λειτουργίες των βασικών βιβλιοθηκών της Java.
  - Dalvik Virtual Machine - Η Dalvik είναι μία εικονική μηχανή διερμηνέας, η οποία εκτελεί αρχεία της μορφής *.dex (Dalvik Executable), μια μορφή που είναι βελτιστοποιημένη για αποδοτική αποθήκευση και εκτέλεση με χαρτογραφημένη μνήμη (memory-mappable). Η εικονική μηχανή βασίζεται σε καταχωρητές και μπορεί να τρέξει κλάσεις που μεταγλωττίστηκαν από έναν Java compiler και έχουν μετασχηματιστεί στη δική της φυσική μορφή, χρησιμοποιώντας το παρεχόμενο εργαλείο "dx". Η VM τρέχει πάνω στον πυρήνα του Linux 2.6, στον οποίο βασίζεται για την υποκείμενη λειτουργικότητα (όπως η διαχείριση threads και η διαχείριση μνήμης σε χαμηλό επίπεδο). Κάθε εφαρμογή Android τρέχει στη δική της διαδικασία (process), με το δικό της στιγμιότυπο (instance) της Dalvik VM. Η DalvikVM βελτιστοποιήθηκε επίσης για να τρέχει σε πολλαπλά στιγμιότυπα με πολύ μικρή χρήση μνήμης. Μια σειρά από VM προστατεύουν μια εφαρμογή από το να υπολειτουργεί εξαιτίας μιας άλλης εφαρμογής που «κόλλησε».

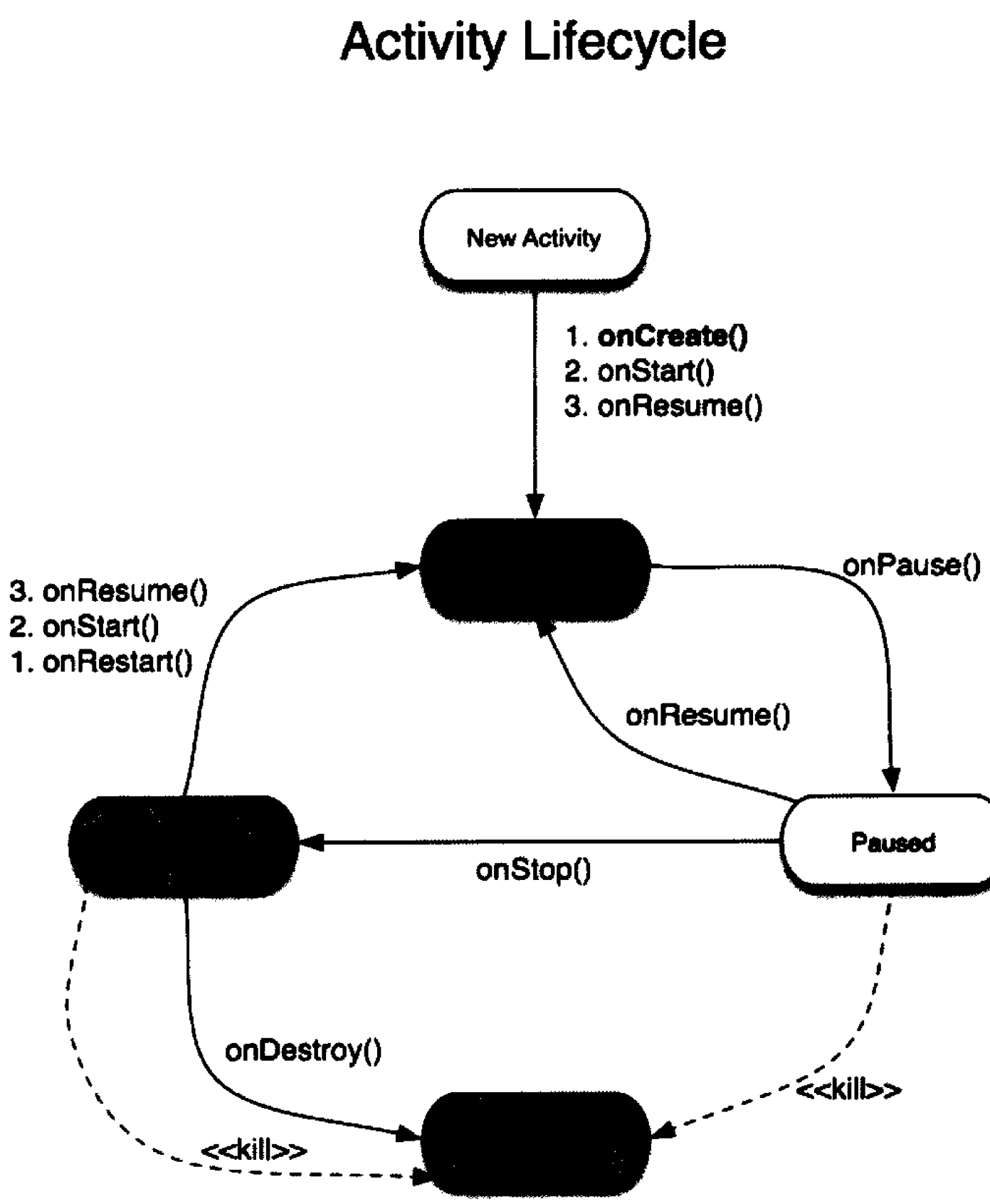
Διαφορές από μια κανονική JavaVM

Η JavaVM, που είναι πλέον εγκατεστημένη σχεδόν σε όλους τους προσωπικούς υπολογιστές, είναι βασισμένη σε «στοίβες» (stack based). Η DalvikVM από την άλλη είναι βασισμένη σε καταχωρητές (registered based), γιατί οι επεξεργαστές για συσκευές όπως τα κινητά είναι βελτιστοποιημένοι για εκτέλεση εφαρμογών με χρήση καταχωρητών.

Επίσης οι VM που βασίζονται σε καταχωρητές επιτρέπουν γρηγορότερη εκτέλεση συχνά σε βάρος προγραμμάτων που είναι μεγαλύτερα σε μέγεθος μετά την μεταγλώττισή τους (compilation).

- **Linux Kernel**

Το Android βασίζεται στον πυρήνα του Linux 2.6 για τις βασικές υπηρεσίες του συστήματος όπως η ασφάλεια, η διαχείριση μνήμης, η διαχείριση διαδικασιών, η στοίβα δικτύου (network stack) και οι οδηγοί (drivers). Επίσης ο πυρήνας λειτουργεί ως ένα αφαιρετικό επίπεδο (abstraction layer) μεταξύ του υλικού και της υπόλοιπης στοίβας λογισμικού.



Εικόνα 8.2: *Ο κύκλος ζωής μιας Δραστηριότητας Android*

8.5 Ανατομία μιας Android Εφαρμογής

Υπάρχουν 4 τμήματα κατασκευής (building blocks) σε μια εφαρμογή Android:

- Activity (Δραστηριότητα),
- Intent Receiver (Δέκτης Πρόθεσης),
- Service (Υπηρεσία),
- Content Provider (Πάροχος Περιεχομένου).

Δεν είναι απαραίτητο να υπάρχουν και τα 4 τμήματα σε μια εφαρμογή, αλλά κάθε εφαρμογή χρησιμοποιεί ένα συνδυασμό αυτών. Από την στιγμή που θα αποφασιστεί ποια τμήματα χρειάζονται για την εφαρμογή, θα πρέπει να οριστούν σε ένα αρχείο που ονομάζεται *AndroidManifest.xml*. Αυτό είναι ένα XML αρχείο όπου δηλώνονται τα τμήματα της εφαρμογής και ποιες είναι οι δυνατότητες και οι απαιτήσεις τους.

➢ Activity (Δραστηριότητα)

Τα Activities χρησιμοποιούνται περισσότερο από τα υπόλοιπα τμήματα που προαναφέρθηκαν. Μια δραστηριότητα είναι συνήθως μια απλή οθόνη της εφαρμογής. Κάθε δραστηριότητα υλοποιείται σαν μια κλάση που επεκτείνει (extends) την βασική κλάση Δραστηριότητα (Activity base class). Γενικά η κλάση προβάλει μια διεπαφή χρήστη (user interface) αποτελούμενη από Εικόνες (Views) και απαντά σε Συμβάντα (Events).

Οι περισσότερες εφαρμογές αποτελούνται από πολλαπλές οθόνες. Για παράδειγμα, μια εφαρμογή ανταλλαγής γραπτών μηνυμάτων, θα μπορούσε να έχει μια οθόνη που δείχνει μια λίστα με τις επαφές, μια δεύτερη οθόνη για σύνταξη μηνύματος και άλλες οθόνες προβολής μηνυμάτων και ρυθμίσεων. Κάθε μια από αυτές τις οθόνες θα υλοποιούταν σαν μια δραστηριότητα. Η μετάβαση σε άλλη οθόνη επιτυγχάνεται με την έναρξη μιας νέας δραστηριότητας. Σε μερικές περιπτώσεις μια δραστηριότητα μπορεί να επιστρέφει μια τιμή σε μια προηγούμενη – για παράδειγμα μια δραστηριότητα που επιτρέπει στο χρήστη να επιλέξει μια φωτογραφία θα επέστρεφε την επιλεγμένη φωτογραφία στην δραστηριότητα που την κάλεσε.

Όταν μια νέα οθόνη προβάλλεται, η προηγούμενη οθόνη μπαίνει σε μια στοίβα που κρατά το ιστορικό (history stack). Ο χρήστης μπορεί να πλοηγηθεί πίσω σε οθόνες που άνοιξε προηγουμένως μέσω του ιστορικού. Οι οθόνες μπορούν επίσης να αφαιρεθούν από το ιστορικό σε περιπτώσεις που δεν χρειάζεται να παραμείνουν προσβάσιμες. Το Android διατηρεί ιστορικό για κάθε εφαρμογή που εκκίνησε από την κεντρική οθόνη (home screen).

- **Intent and Intent Filters (Πρόθεση και Φίλτρα Πρόθεσης)**

Το Android χρησιμοποιεί μια ειδική κλάση που λέγεται Πρόθεση (Intent) για να κινείται από οθόνη σε οθόνη. Η Πρόθεση περιγράφει τι θέλει η εφαρμογή να γίνει στη συνέχεια. Τα δυο πιο σημαντικά μέρη της δομής δεδομένων της Πρόθεσης είναι η δράση (action) και τα δεδομένα βάσει των οποίων αυτή θα εκτελεστεί. Τυπικές τιμές για μια δράση είναι η MAIN (η κεντρική είσοδος της εφαρμογής), η VIEW, η PICK, η EDIT κλπ. Τα δεδομένα εκφράζονται ως URI (Uniform Resource Indicator). Για παράδειγμα, για να δείτε μια ιστοσελίδα στον browser, θα δημιουργούσατε ένα Intent με δράση VIEW και τα δεδομένα ως ένα website-URI.

```
new Intent(android.content.Intent.VIEW_ACTION,
ContentURI.create("http://anddev.org"));
```

Υπάρχει μια σχετική κλάση που λέγεται Φίλτρο Πρόθεσης (IntentFilter). Ενώ μια Πρόθεση περιγράφει ένα αίτημα για να γίνει κάτι, το Φίλτρο Πρόθεσης είναι μια περιγραφή του τι είναι δυνατόν να διαχειριστεί ένας Δέκτης Πρόθεσης (intent receiver). Μια δραστηριότητα που είναι σε θέση να προβάλει πληροφορίες επικοινωνίας για ένα άτομο θα ανακοίνωνε με ένα Φίλτρο Πρόθεσης (IntentFilter) ότι γνωρίζει πως να διαχειριστεί την VIEW_ACTION, όταν τα δεδομένα αντιπροσωπεύουν ένα άτομο. Οι δραστηριότητες ανακοινώνουν τα Φίλτρα Πρόθεσης στο αρχείο AndroidManifest.xml.

Η πλοήγηση από οθόνη σε οθόνη επιτυγχάνεται με Προθέσεις. Για να πλοηγηθείτε προς τα εμπρός, μια δραστηριότητα καλεί την startActivity(myIntent). Το σύστημα τότε κοιτά στα Φίλτρα Προθέσεων (intent filters) για όλες τις εγκατεστημένες εφαρμογές και διαλέγει την δραστηριότητα που τα Φίλτρα Πρόθεσης ταιριάζουν καλύτερα με την myIntent παράμετρο της κλήσης. Η νέα Δραστηριότητα

ενημερώνεται για την Πρόθεση και ξεκινά. Η διαδικασία της υλοποίησης των Προθέσεων συμβαίνει κατά τον χρόνο εκτέλεσης της εφαρμογής όταν καλείται η startActivity, πράγμα που προσφέρει 2 βασικά πλεονεκτήματα:

- Οι Δραστηριότητες μπορούν να επαναχρησιμοποιούν κάποια λειτουργικότητα από άλλα τμήματα του κώδικα, απλά κάνοντας ένα αίτημα υπό την μορφή μιας Πρόθεσης.
- Οι Δραστηριότητες μπορούν να αντικατασταθούν οποιαδήποτε στιγμή από μια νέα Δραστηριότητα με ένα αντίστοιχο Φίλτρο Πρόθεσης.

➢ **Intent Receiver (Δέκτης Πρόθεσης)**

Ένα IntentReceiver χρησιμοποιείται όταν θέλουμε η εφαρμογή μας να εκτελεστεί σε απάντηση ενός εξωτερικού συμβάντος (external event), για παράδειγμα όταν το τηλέφωνο χτυπά, ή όταν το δίκτυο είναι διαθέσιμο, ή όταν είναι μεσάνυχτα. Οι Δέκτες Πρόθεσης δεν προβάλουν μια διεπαφή χρήστη (UI), ωστόσο μπορούν να προβάλουν Ειδοποιήσεις (Notifications) για να ειδοποιήσουν τον χρήστη για κάτι σημαντικό που συνέβη. Οι Δέκτες Πρόθεσης είναι επίσης καταχωρημένοι στο AndroidManifest.xml, αλλά μπορούμε επίσης να τους καταχωρήσουμε από τον κώδικα χρησιμοποιώντας την Context.registerReceiver(). Η εφαρμογή μας δεν χρειάζεται να τρέχει για να κληθούν οι Δέκτες Πρόθεσης που έχει. Το σύστημα θα εκκινήσει την εφαρμογή, αν χρειαστεί, όταν ένας Δέκτης Πρόθεσης ενεργοποιηθεί. Οι εφαρμογές μπορούν επίσης να στέλνουν τις δικές τους Ανακοινώσεις Πρόθεσης (intent broadcasts) σε άλλους με την Context.broadcastIntent().

➢ **Service (Υπηρεσία)**

Ένα Service είναι κώδικας που τρέχει για μεγάλο χρονικό διάστημα και χωρίς διεπαφή χρήστη (UI). Ένα καλό παράδειγμα είναι μια εφαρμογή που αναπαράγει μουσική από μια λίστα μουσικών κομματιών (media player). Σε μια τέτοια εφαρμογή, θα υπήρχαν κατά πάσα πιθανότητα μία ή και παραπάνω Δραστηριότητες που επιτρέπουν στον χρήστη να επιλέξει τραγούδια και να τα αναπαράξει. Ωστόσο, η

αναπαραγωγή από μόνη της δεν θα έπρεπε να διαχειρίζεται από την Δραστηριότητα γιατί ο χρήστης θα περίμενε την μουσική να συνεχίσει να παίζει ακόμη και μετά την πλοήγησή του σε μια νέα οθόνη. Σε αυτή τη περίπτωση, η Δραστηριότητα της αναπαραγωγής μουσικής θα ξεκινούσε μια υπηρεσία χρησιμοποιώντας την Context.startService() για να τρέξει στο background και να συνεχίσει η μουσική να παίζει. Το σύστημα τότε θα κρατά την υπηρεσία αναπαραγωγής ενεργή μέχρι να τελειώσει το κομμάτι. Να σημειώσουμε ότι μπορούμε να συνδεθούμε σε μια υπηρεσία (και να την εκκινήσουμε αν δεν έχει ξεκινήσει) με την μέθοδο Context.bindService(). Όταν συνδεθούμε σε μια υπηρεσία, μπορούμε να επικοινωνήσουμε με αυτή μέσω μιας διεπαφής που προσφέρεται από την υπηρεσία. Για την υπηρεσία μουσικής, αυτό θα μας επέτρεπε να κάνουμε παύση, να πάμε πίσω στο κομμάτι (rewind) κλπ.

- **Content Provider (Πάροχος Περιεχομένου)**

Οι εφαρμογές μπορούν να σώσουν τα δεδομένα τους σε αρχεία, σε μια SQLite βάση δεδομένων, σε προτιμήσεις (preferences) ή σε οποιοδήποτε άλλο μηχανισμό μπορούν. Ένας Πάροχος Περιεχομένου είναι χρήσιμος αν θέλουμε τα δεδομένα της εφαρμογής μας να είναι διαθέσιμα και σε άλλες εφαρμογές. Επίσης είναι μια κλάση που υλοποιεί μια συγκεκριμένη ομάδα μεθόδων, που επιτρέπει σε άλλες εφαρμογές να αποθηκεύουν και να επανακτούν δεδομένα του τύπου που διαχειρίζεται ο Πάροχος Περιεχομένου.

8.6 Ασφάλεια και Άδειες

Το Android είναι ένα σύστημα πολλαπλών διαδικασιών (multi-process) στο οποίο κάθε εφαρμογή και μέρη του συστήματος τρέχουν στη δική τους διαδικασία. Η περισσότερη ασφάλεια μεταξύ των εφαρμογών και του συστήματος γίνεται σε επίπεδο διαδικασιών μέσω των διευκολύνσεων του Linux, όπως τα user και group IDs που εκχωρούνται στις εφαρμογές. Επιπρόσθετα χαρακτηριστικά ασφαλείας παρέχονται μέσω ενός μηχανισμού «Αδειών» (Permission), ο οποίος θέτει τους

περιορισμούς για ειδικές λειτουργίες που μια συγκεκριμένη διαδικασία μπορεί να εκτελέσει και τα δικαιώματα που έχει καθένα URI για την επί τούτου χορήγηση πρόσβασης σε συγκεκριμένα τμήματα δεδομένων.

8.6.1 Η Αρχιτεκτονική της Ασφάλειας

Ένα κύριο σημείο στην αρχιτεκτονική της ασφάλειας του Android είναι ότι καμία εφαρμογή, από προεπιλογή, δεν έχει άδεια να κάνει οποιεσδήποτε λειτουργίες οι οποίες θα έχουν αρνητικές επιπτώσεις σε άλλες εφαρμογές, στο λειτουργικό σύστημα, ή στον χρήστη. Περιλαμβάνονται η ανάγνωση ή η εγγραφή των προσωπικών δεδομένων του χρήστη (όπως οι επαφές ή τα e-mail), η ανάγνωση ή η εγγραφή αρχείων άλλης εφαρμογής, η πρόσβαση στο δίκτυο, η διατήρηση αναμμένης της οθόνης της συσκευής, κλπ.

Η διαδικασία μιας εφαρμογής είναι ένα ασφαλές «κουτί» (sandbox). Δεν μπορεί να διαταράξει άλλες εφαρμογές, εκτός εάν δηλώσει ρητά τα δικαιώματα που χρειάζεται για πρόσθετες δυνατότητες που δεν παρέχονται από το βασικό «κουτί». Αυτά τα δικαιώματα που ζητάει μπορούν να διεκπεραιωθούν από το λειτουργικό με διάφορους τρόπους, συνήθως αυτόματα επιτρέποντας ή όχι βάση πιστοποιητικών, ή προτρέποντας τον χρήστη να αποφασίσει. Τα δικαιώματα που χρειάζονται από μία εφαρμογή δηλώνονται στατικά στην εφαρμογή, έτσι ώστε να είναι γνωστά κατά την εγκατάστασή της και δεν μπορούν να αλλάξουν μετέπειτα.

8.6.2 Υπογραφή της Εφαρμογής

Όλες οι Android εφαρμογές (αρχεία .apk) πρέπει να υπογράφονται με ένα πιστοποιητικό του οποίου το ιδιωτικό κλειδί κρατείται από τον developer τους. Το πιστοποιητικό δεν χρειάζεται να έχει υπογραφεί από μία αρχή πιστοποίησης και έτσι είναι απολύτως επιτρεπτό και σύνηθες για τις Android εφαρμογές, να χρησιμοποιούν αυτό-υπογραφόμενα πιστοποιητικά. Το πιστοποιητικό αυτό χρησιμοποιείται μόνο για την επίτευξη σχέσεων εμπιστοσύνης μεταξύ των εφαρμογών και όχι για το γενικό έλεγχο του κατά πόσο μία εφαρμογή μπορεί να εγκατασταθεί. Οι πιο σημαντικοί

τρόποι που οι υπογραφές επιδρούν στην ασφάλεια είναι με το να προσδιορίζεται ποιος μπορεί να έχει πρόσβαση στις άδειες βάση πιστοποιητικού και ποιος μπορεί να μοιράζει ID χρήστη.

8.6.3 ID Χρήστη και Πρόσβαση Αρχείων

Κάθε αρχείο Android package (.apk) που εγκαθίσταται στη συσκευή, λαμβάνει το δικό του μοναδικό Linux ID χρήστη, δημιουργείται γι' αυτό ένα «κουτί» και εμποδίζεται από το να έχει επαφή με άλλες εφαρμογές (ή οι άλλες εφαρμογές να έχουν επαφή μ' αυτό). Αυτό το ID χρήστη του απονέμεται κατά την εγκατάσταση της εφαρμογής στη συσκευή και παραμένει σταθερό κατά τη διάρκεια της ζωής του σ' αυτή τη συσκευή.

Επειδή η επιβολή ασφάλειας συμβαίνει στο επίπεδο διαδικασίας, ο κώδικας οποιονδήποτε δύο packages δεν μπορεί να εκτελεστεί κανονικά στην ίδια διαδικασία, δεδομένου ότι πρέπει να τρέχουν ως διαφορετικοί χρήστες Linux. Μπορεί να χρησιμοποιηθεί το χαρακτηριστικό sharedUserId στην ετικέτα manifest του AndroidManifest.xml καθενός package, για να λάβουν έτσι το ίδιο ID χρήστη. Με τον τρόπο αυτό, για λόγους ασφαλείας τα δύο πακέτα αντιμετωπίζονται στη συνέχεια σαν να είναι η ίδια εφαρμογή, με το ίδιο ID χρήστη και τα ίδια δικαιώματα αρχείου. Αξίζει να σημειωθεί ότι για να διατηρηθεί η ασφάλεια, μόνο δύο εφαρμογές που υπογράφηκαν με το ίδιο πιστοποιητικό (και ζήτησαν το ίδιο sharedUserId) θα μπορέσουν να πάρουν το ίδιο ID χρήστη.

Στα δεδομένα που αποθηκεύονται από μία εφαρμογή, απονέμεται το ID χρήστη της εφαρμογής και δεν είναι προσβάσιμα από άλλα packages. Κατά τη δημιουργία ενός νέου αρχείου με τις getSharedPreferences(String, int), openFileOutput(String, int), ή την openOrCreateDatabase(String, int, SQLiteDatabase.CursorFactory), μπορούν να χρησιμοποιηθούν τα flags MODE_WORLD_READABLE και/ή MODE_WORLD_WRITEABLE για να επιτρέψουν σε οποιοδήποτε package να διαβάσει/γράψει στο αρχείο. Όταν τίθενται αυτά τα flags, το αρχείο ανήκει ακόμα στην εφαρμογή αλλά τα δικαιώματα ανάγνωσης/εγγραφής έχουν τεθεί κατάλληλα έτσι ώστε οποιαδήποτε εφαρμογή θέλει μπορεί να έχει πρόσβαση σ' αυτό.

8.6.4. Χρησιμοποιώντας τις Άδειες

Μία βασική εφαρμογή Android δεν έχει άδειες-δικαιώματα που να συνδέονται μ' αυτή, που σημαίνει ότι δεν μπορεί να κάνει τίποτα που να επηρεάσει αρνητικά την εμπειρία του χρήστη ή οποιαδήποτε δεδομένα στη συσκευή. Για να γίνει χρήση των προστατευόμενων λειτουργιών της συσκευής, θα πρέπει να συμπεριληφθούν στο αρχείο AndroidManifest.xml μία ή περισσότερες ετικέτες <uses-permission> δηλώνοντας τα δικαιώματα που χρειάζεται η εφαρμογή.

Για παράδειγμα, μία εφαρμογή που χρειάζεται να παρακολουθεί τα εισερχόμενα μηνύματα SMS θα ορίζει τα εξής:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="com.android.app.myapp" >
<uses-permission android:name="android.permission.RECEIVE_SMS" />
</manifest>
```

Κατά την εγκατάσταση της εφαρμογής, οι άδειες που ζητήθηκαν απ' αυτή παραχωρήθηκαν από το πρόγραμμα εγκατάστασης πακέτων, με βάση τον έλεγχο του πιστοποιητικού της εφαρμογής και/ή προτρέποντας το χρήστη να αποφασίσει. Από τη στιγμή που τρέχει η εφαρμογή δεν γίνονται έλεγχοι αλληλεπίδρασης με το χρήστη και έτσι είτε χορηγήθηκε μια συγκεκριμένη άδεια όταν εγκαταστάθηκε και μπορεί να χρησιμοποιηθεί η αντίστοιχη λειτουργία, είτε η άδεια απορρίφθηκε και οποιαδήποτε προσπάθεια να χρησιμοποιηθεί εκείνη η λειτουργία να αποτύχει αφού δεν εγκρίθηκε από το χρήστη.

Πολλές φορές μία αποτυχία άδειας καταλήγει σε εξαίρεση ασφαλείας (SecurityException) που επιστρέφεται στην εφαρμογή. Ωστόσο, δεν υπάρχει εγγύηση ότι θα συμβεί παντού. Για παράδειγμα, η μέθοδος sendBroadcast(Intent) ελέγχει τα δικαιώματα όσο τα δεδομένα παραλαμβάνονται από κάθε παραλήπτη, αφότου η κλήση της μεθόδου έχει επιστρέψει, έτσι ώστε να μην ληφθεί εξαίρεση εάν υπάρχουν αποτυχίες άδειας. Όμως, σε όλες σχεδόν τις περιπτώσεις, μία αποτυχία άδειας θα τυπωθεί στο αρχείο καταγραφής του συστήματος.

Οι άδειες που παρέχονται από το σύστημα του Android μπορούν να βρεθούν στο Manifest.permission. Κάθε εφαρμογή μπορεί επίσης να καθορίσει και να επιβάλλει τα

δικά της δικαιώματα, έτσι ώστε να μην ζητείται μία πλήρης λίστα με όλες τις πιθανές άδειες.

Μία συγκεκριμένη άδεια μπορεί να επιβάλλεται σε διάφορα τμήματα της εφαρμογής κατά τη διάρκεια εκτέλεσης της, όπως:

- Κατά τη διάρκεια μιας κλήσης στο σύστημα, για να αποφευχθεί μία εφαρμογή από το να εκτελέσει κάποιες λειτουργίες.
- Κατά την εκκίνηση ενός Activity, για να εμποδιστούν οι εφαρμογές από το να εκτελέσουν Activities άλλων εφαρμογών.
- Τόσο κατά την αποστολή όσο και κατά την λήψη Broadcasts, για να ελεγχθεί ποιος μπορεί να στείλει ή να λάβει Broadcasts σε/από ποιόν.
- Κατά την πρόσβαση και τον χειρισμό ενός Content Provider.
- Συνδέοντας (Binding) ή αρχίζοντας ένα Service.

8.6.5. Επιβολή Αδειών στο AndroidManifest.xml

Υψηλού επιπέδου άδειες που να περιορίζουν την πρόσβαση σε ολόκληρα τμήματα του συστήματος ή σε εφαρμογές, μπορούν να εφαρμοστούν μέσω του AndroidManifest.xml. Το μόνο που απαιτείται είναι να συμπεριληφθεί ένα χαρακτηριστικό android:permission στο επιθυμητό τμήμα του συστήματος και να καθοριστεί η άδεια που θα χρησιμοποιηθεί για να ελέγχει την πρόσβαση σ' αυτό.

Άδειες τύπου *Activity* (εφαρμόζονται με την ετικέτα <activity>) περιορίζουν ποιός μπορεί να αρχίσει το σχετικό Activity. Η άδεια ελέγχεται κατά την κλήση των Context.startActivity() και Activity.startActivityForResult(). Εάν αυτός που τις καλεί δεν έχει την απαιτούμενη άδεια τότε επιστρέφεται μία SecurityException.

Άδειες τύπου *Service* (εφαρμόζονται με την ετικέτα <service>) περιορίζουν ποιός μπορεί να αρχίσει ή να συνδεθεί στο σχετικό Service. Η άδεια ελέγχεται κατά την κλήση των Context.startService(), Context.stopService() και Context.bindService(). Εάν αυτός που τις καλεί δεν έχει την απαιτούμενη άδεια τότε επιστρέφεται μία SecurityException.

Άδειες τύπου *BroadcastReceiver* (εφαρμόζονται με την ετικέτα <receiver>) περιορίζουν ποιός μπορεί να στείλει broadcasts στο σχετικό receiver. Η άδεια ελέγχεται μετά την επιστροφή της Context.sendBroadcast(), καθώς το σύστημα προσπαθεί να παραδώσει το broadcast στο σχετικό receiver. Ως αποτέλεσμα, μία

αποτυχία άδειας δεν θα επιστρέψει μία εξαίρεση (exception) σ' αυτόν που την κάλεσε, απλά δεν θα παραδώσει το intent. Με τον ίδιο τρόπο, μία άδεια μπορεί να τροφοδοτήσει την Context.registerReceiver() για να ελέγχεται ποιός μπορεί να στείλει broadcasts σ' έναν καταχωρημένο (registered) receiver. Ένας άλλος τρόπος, είναι μία άδεια να τροφοδοτήσει την Context.sendBroadcast() για να περιορίσει ποιά αντικείμενα BroadcastReceiver επιτρέπεται να λαμβάνουν το broadcast.

Άδειες τύπου *ContentProvider* (εφαρμόζονται με την ετικέτα <provider>) περιορίζουν ποιός μπορεί να έχει πρόσβαση στα δεδομένα ενός ContentProvider. (Οι Content Providers έχουν μία πρόσθετη δυνατότητα ασφαλείας, τις άδειες URI). Σε αντίθεση με τα προηγούμενα, εδώ υπάρχουν δύο ξεχωριστά χαρακτηριστικά άδειας που μπορούν να τεθούν:

- Το *android:readPermission* το οποίο περιορίζει ποιός μπορεί να διαβάσει από τον provider και
- το *android:writePermission* το οποίο περιορίζει ποιός μπορεί να γράψει σ' αυτόν.

Κεφάλαιο 9ο

Υλοποίηση Android Εφαρμογής

Η Android εφαρμογή kratisinow.gr αποτελεί μέρος του διαδικτυακού χώρου www.kratisinow.gr και αφορά στην πραγματοποίηση online κρατήσεων σε νυχτερινά κέντρα διασκέδασης. Για την πραγματοποίηση και την ολοκλήρωση μιας κράτησης επιβάλλεται τηλεφωνική επικοινωνία, επομένως η εφαρμογή αυτή αφορά στην ενημέρωση των χρηστών για τα διαθέσιμα μαγαζιά, σχήματα και ημερομηνίες καθώς και στην αποστολή αίτησης εκδήλωσης ενδιαφέροντος.

9.1 Ανάλυση Εφαρμογής

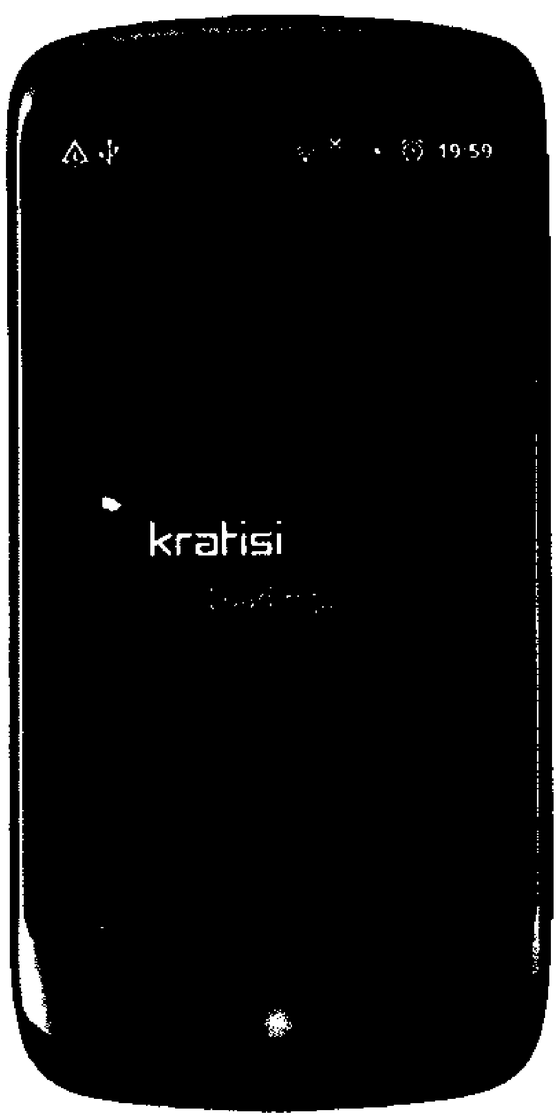
Για την δημιουργία της εφαρμογής και την προβολή των κύριων δραστηριοτήτων της χρησιμοποιήθηκε το control TabHost, ένα control που χρησιμοποιεί καρτέλες στο πάνω μέρος της οθόνης σε συνδυασμό με ετικέτες (labels). Έτσι λοιπόν η εφαρμογή αποτελείται από πέντε καρτέλες αναπαριστώντας το κύριο menu, κάθε μία εκ των οποίων περιέχει και μια διαφορετική δραστηριότητα (activity).

Οι δραστηριότητες αυτές είναι οι εξής:

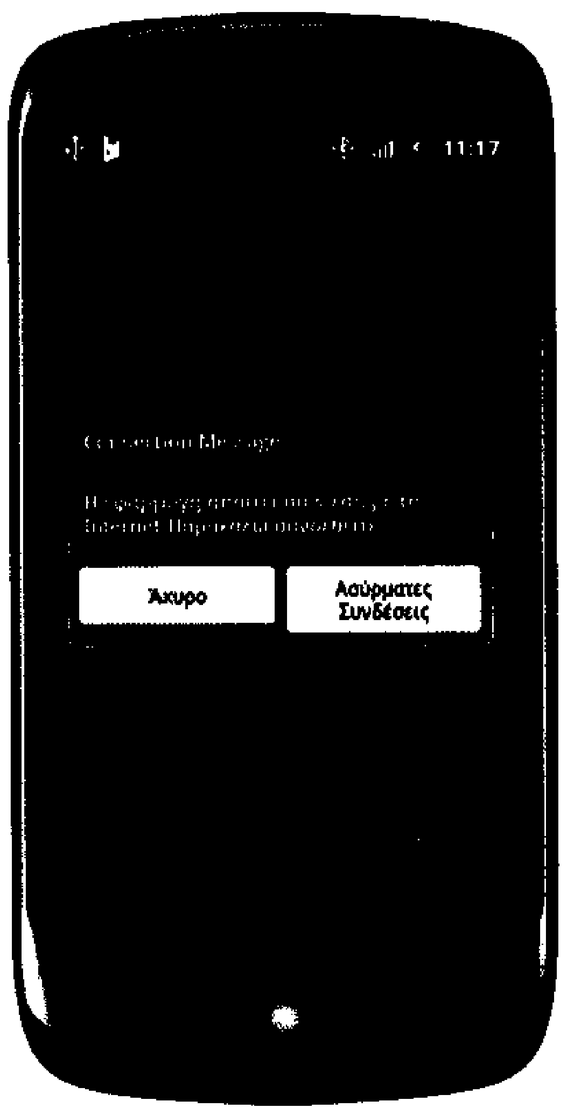
- tab1.java,
- Stores.java,
- Trips.java,
- Notifications.java,
- Contact.java

SplashScreen.java

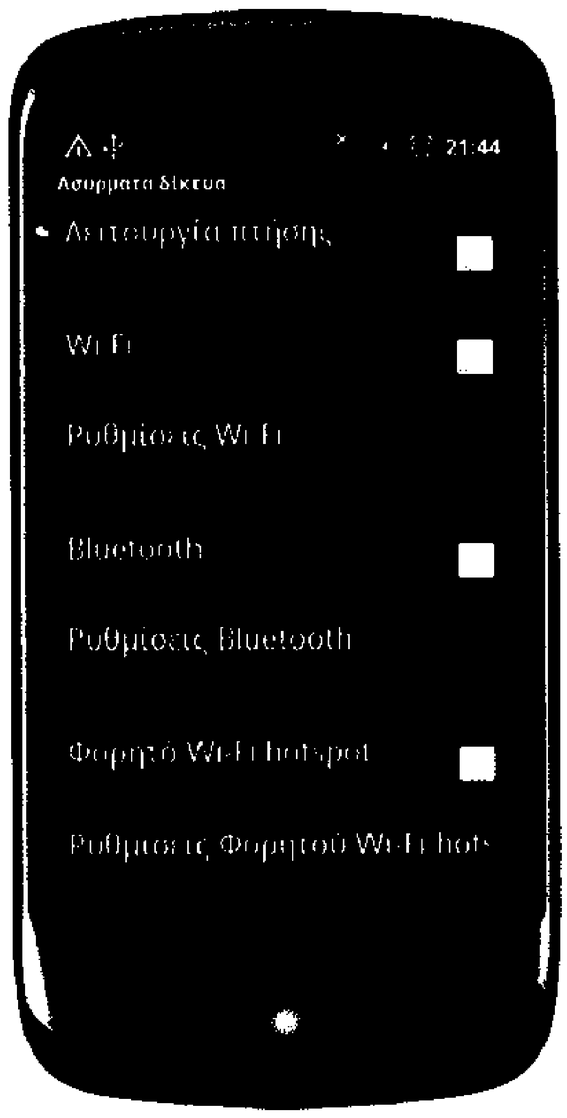
Κατά το άνοιγμα της εφαρμογής ο χρήστης αντικρίζει μια εισαγωγική οθόνη *(Εικόνα 9.1 - SplashScreen)* για 3 δευτερόλεπτα κατά την οποία γίνεται αρχικοποίηση των μεταβλητών του συστήματος και έλεγχος για σύνδεση δικτύου. Σε περίπτωση ύπαρξης Wifi ή 3G σύνδεσης, καλείται η κύρια κλάση της εφαρμογής kratisinow.java, ενώ σε αντίθετη περίπτωση εμφανίζεται στον χρήστη σχετικό μήνυμα *(Εικόνα 9.2)*. Επιλέγοντας ο χρήστης το κουμπί «Άκυρο» η εφαρμογή τερματίζεται, ενώ με την επιλογή «Ασύρματες Συνδέσεις» εμφανίζεται στον χρήστη το αντίστοιχο μενού ασύρματων συνδέσεων του κινητού τηλεφώνου *(Εικόνα 9.3)*.



Εικόνα 9.1: *SplashScreen*

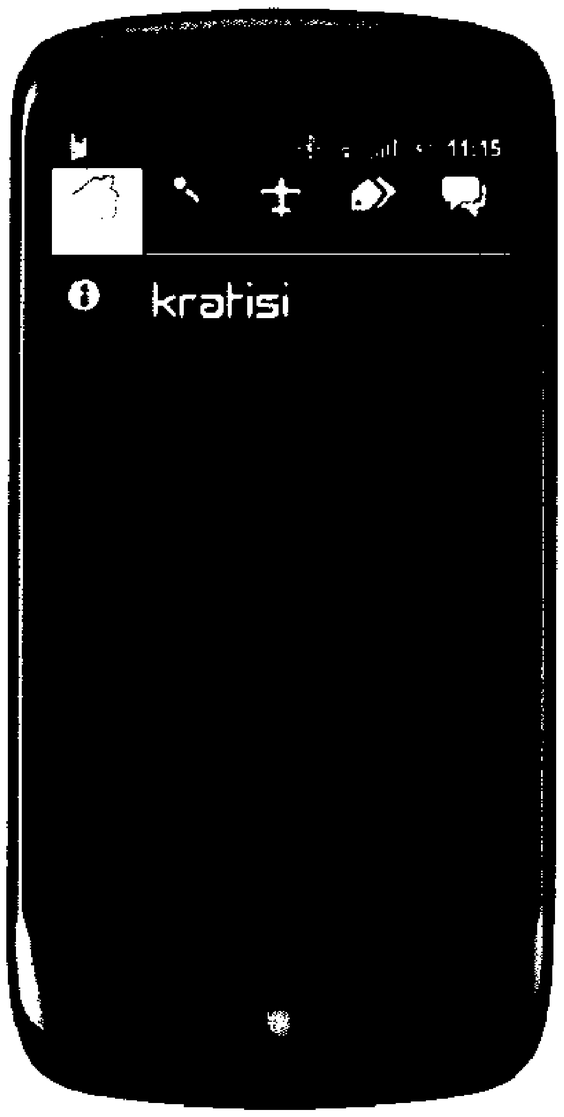


Εικόνα 9.2: *Μήνυμα αδυναμίας σύνδεσης στο internet*



Εικόνα 9.3: *Ασύρματες συνδέσεις*

Kratisinow.java



Πρόκειται για την κύρια δραστηριότητα (activity) της εφαρμογής που κάνει implement το TabHost που αναφέραμε παραπάνω.

Για κάθε καρτέλα ορίζουμε τρεις βασικές παραμέτρους:

α. το επιθυμητό εικονίδιο που θέλουμε να φαίνεται,

β. την ετικέτα που θα υπάρχει στο επάνω μέρος της καρτέλας,

γ. την δραστηριότητα (activity) που θα εκτελείται.

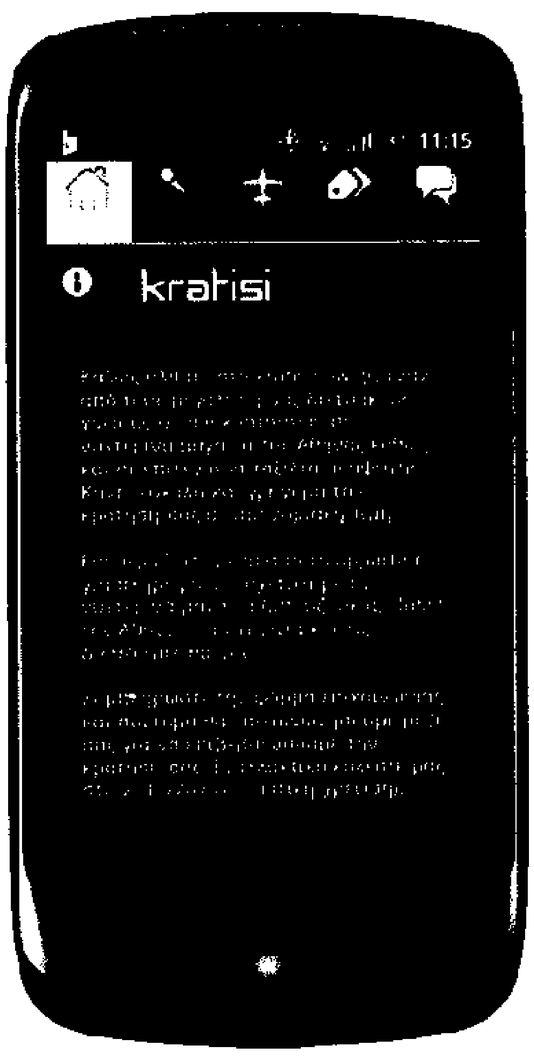
Με βάση το παραπάνω δημιουργούμε πέντε καρτέλες με τα εξής χαρακτηριστικά:

- Καρτέλα 1η με ετικέτα «Αρχική» και δραστηριότητα που εκτελεί tab1.class.
- Καρτέλα 2η με ετικέτα «Νυχ.Κέντρα» και δραστηριότητα που εκτελεί Stores.class.
- Καρτέλα 3η με ετικέτα «Ταξίδια» και δραστηριότητα που εκτελεί Trips.class.
- Καρτέλα 4η με ετικέτα «Προσφορές» και δραστηριότητα που εκτελεί Notifications.class.
- Καρτέλα 5η με ετικέτα «Επ/νία» και δραστηριότητα που εκτελεί Contact.class.

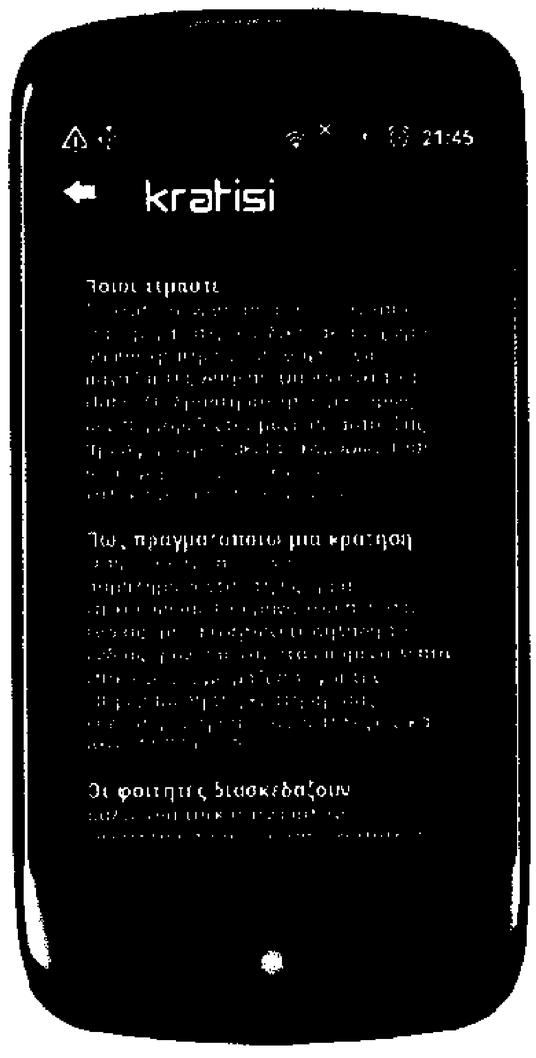
```
TabHost tabHost = getTabHost();


tabHost.addTab(tabHost.newTabSpec("Home")
.setIndicator("Αρχική",getResources().getDrawable(R.drawable.home_tab))
.setContent(new Intent(this, tab1.class)));


tabHost.addTab(tabHost.newTabSpec("Stores")
.setIndicator("Νυχ.Κέντρα",getResources().getDrawable(R.drawable.stages
_tab)).setContent(new Intent(this, Stores.class)));


tabHost.addTab(tabHost.newTabSpec("Trips")
```

```
.setIndicator("Ταξίδια",getResources().getDrawable(R.drawable.trips_tab
)).setContent(new Intent(this, Trips.class)));



tabHost.addTab(tabHost.newTabSpec("Notifications")
.setIndicator("Προσφορές",getResources().getDrawable(R.drawable.notific
ations_tab)).setContent(new Intent(this, Notifications.class)));



tabHost.addTab(tabHost.newTabSpec("Contact")
.setIndicator("Επ/νία",getResources().getDrawable(R.drawable.contact_ta
b)).setContent(new Intent(this, Contact.class)));
```

Tab1.java



Εικόνα 9.4: *Tab1.java*



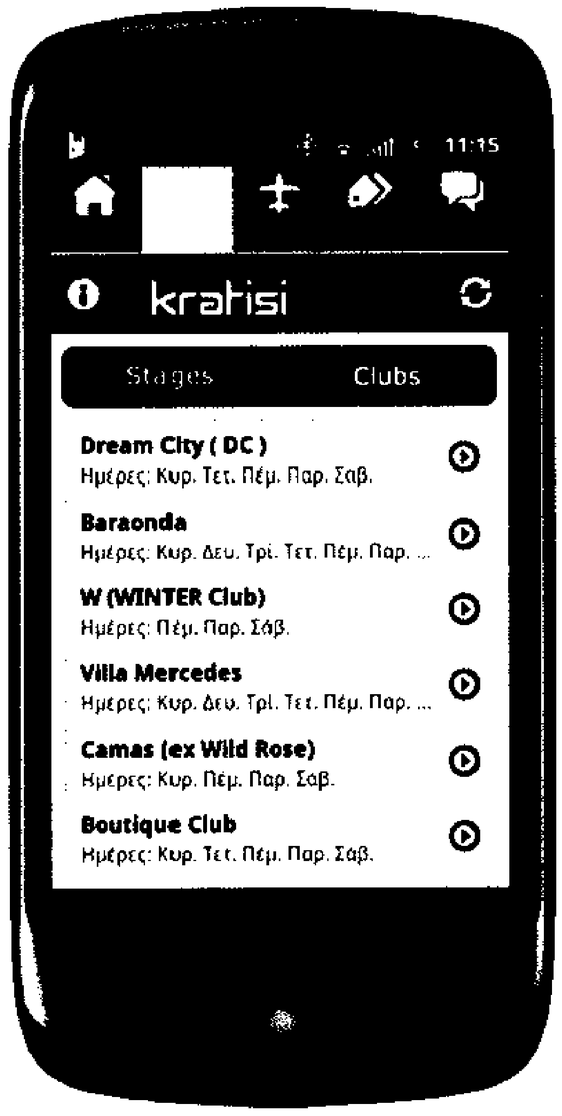
Εικόνα 9.5: *Info.java*

Η συγκεκριμένη κλάση εκτελείται πρώτη, αφού περικλείεται στην πρώτη καρτέλα της εφαρμογής. Εμφανίζεται στον χρήστη ένα μήνυμα καλωσορίσματος και κάποιες βασικές πληροφορίες για την εφαρμογή *(Εικόνα 9.4)*. Ο χρήστης έχει την δυνατότητα να επιλέξει το κουμπί που βρίσκεται δίπλα από το logo της εφαρμογής και να περιηγηθεί σε μία νέα activity με όνομα Info.java *(Εικόνα 9.5)*.

Η κλάση Info.java προβάλει πληροφορίες σχετικά με το kratisinow.gr και τον τρόπο με τον οποίο πραγματοποιείται μια κράτηση. Μπορεί να επιστρέψει στην αρχική activity επιλέγοντας το κουμπί back που βρίσκεται στο επάνω μέρος της οθόνης, αριστερά από το logo.

Stores.java

Εικόνα 9.6: *Stages*



Εικόνα 9.7: *Clubs*

Η συγκεκριμένη κλάση προβάλλει στον χρήστη όλα τα διαθέσιμα μαγαζιά, stages και clubs *(Εικόνα 9.6, 9.7)*. Για εξοικονόμηση χώρου έχουμε δημιουργήσει ένα custom control που συνδυάζει δύο διαφορετικά views με οριζόντια κίνηση κατά την εναλλαγή τους. Οι functions που χρησιμοποιήθηκαν για την κίνηση είναι οι inFromRightAnimation(), outToLeftAnimation(), inFromLeftAnimation(), outToRightAnimation().

```
private Animation inFromRightAnimation(){

        Animation inFromRight = new TranslateAnimation(

        Animation.RELATIVE_TO_PARENT,  +1.0f,
Animation.RELATIVE_TO_PARENT,  0.0f,

        Animation.RELATIVE_TO_PARENT,  0.0f,
Animation.RELATIVE_TO_PARENT,   0.0f

        );

        inFromRight.setDuration(300);

        inFromRight.setInterpolator(new DecelerateInterpolator());

        return inFromRight;

    }

    private Animation outToLeftAnimation(){

        Animation outtoLeft = new TranslateAnimation(

        Animation.RELATIVE_TO_PARENT,  0.0f,
Animation.RELATIVE_TO_PARENT,  -1.0f,

        Animation.RELATIVE_TO_PARENT,  0.0f,
Animation.RELATIVE_TO_PARENT,  0.0f

        );

        outtoLeft.setDuration(300);

        outtoLeft.setInterpolator(new DecelerateInterpolator());

        return outtoLeft;

    }


    private Animation inFromLeftAnimation() {

        Animation inFromLeft = new TranslateAnimation(

        Animation.RELATIVE_TO_PARENT,  -1.0f,
Animation.RELATIVE_TO_PARENT,  0.0f,

        Animation.RELATIVE_TO_PARENT,  0.0f,
Animation.RELATIVE_TO_PARENT,   0.0f

        );

        inFromLeft.setDuration(300);

        inFromLeft.setInterpolator(new DecelerateInterpolator());

        return inFromLeft;

    }
```

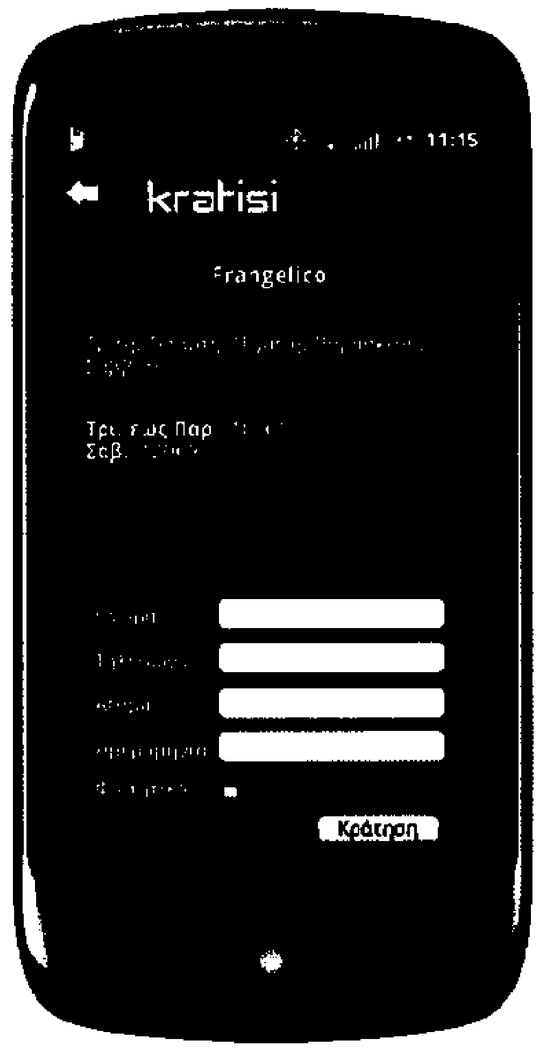
```
    private Animation outToRightAnimation() {

        Animation outtoRight = new TranslateAnimation(

        Animation.RELATIVE_TO_PARENT,  0.0f,
Animation.RELATIVE_TO_PARENT,  +1.0f,

        Animation.RELATIVE_TO_PARENT,  0.0f,
Animation.RELATIVE_TO_PARENT,   0.0f

        );

        outtoRight.setDuration(300);

        outtoRight.setInterpolator(new DecelerateInterpolator());

        return outtoRight;

    }
```

Το συγκεκριμένο control κάνει χρήση animations και δείχνει δύο διαφορετικά layouts. Κάθε layout αποτελείται από έναν πίνακα όπου παρουσιάζονται τα αντίστοιχα μαγαζιά. Όταν ο χρήστης επιλέξει κάποιο από τα μαγαζιά, γίνεται έναρξη μιας νέας δραστηριότητας η οποία δείχνει πληροφορίες (διαθέσιμες ημέρες, τιμές, περιγραφή) για το συγκεκριμένο μαγαζί *(Εικόνα 9.8)*.

Στο σημείο αυτό ο χρήστης έχει 3 επιλογές:

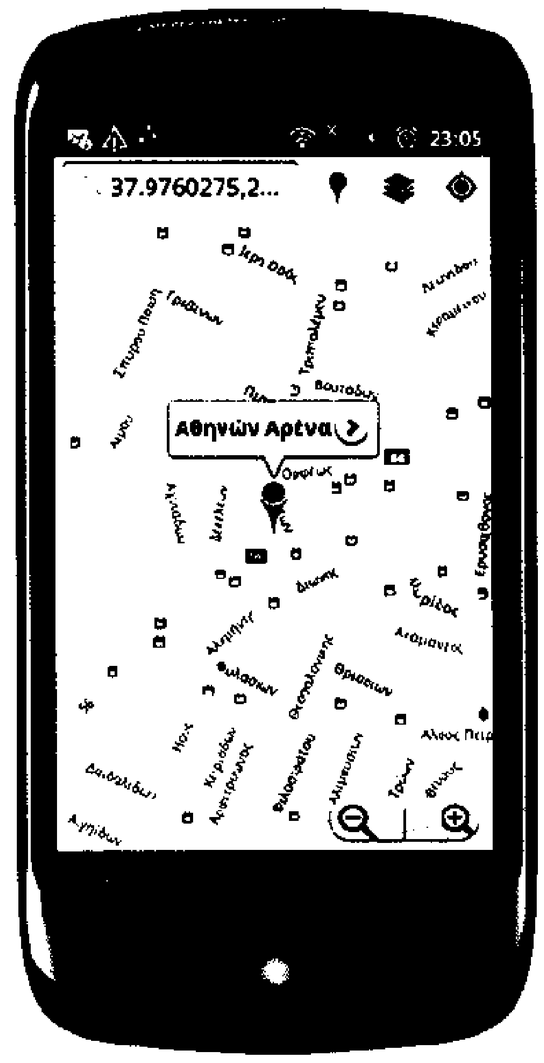
- Να προχωρήσει στην διαδικασία κράτησης, επιλέγοντας το κουμπί «Κράτηση» *(Εικόνα 9.9)*,
- Να επιλέξει το κουμπί «Share» ώστε να πραγματοποιήσει αποστολή στοιχείων μέσω μηνύματος, email, facebook, twitter κτλ. *(Εικόνα 9.10)*,
- Να ανοίξει τον χάρτη και να βρει τις συντεταγμένες του μαγαζιού καθώς και οδηγίες κατεύθυνσης προς το επιλεγμένο σημείο *(Εικόνα 9.11)*.

Εικόνα 9.8: *Πληροφορίες καταστήματος*



Εικόνα 9.9: *Φόρμα κράτησης*

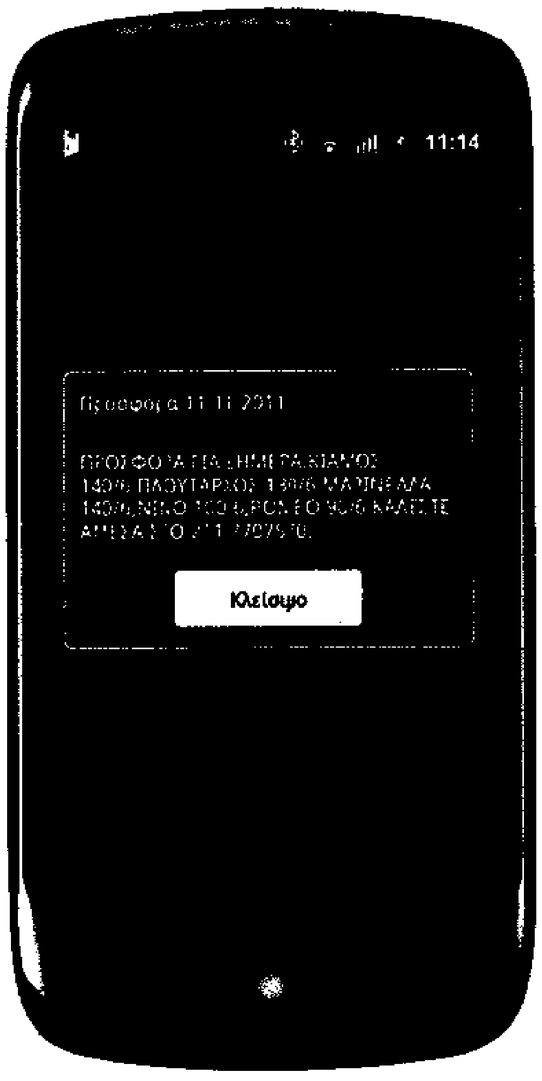
Εικόνα 9.10: *Επιλογές Share*



Εικόνα 9.11: *Διεύθυνση καταστήματος*

Notifications.java

Εικόνα 9.12: *Διαθέσιμες Προσφορές*



Εικόνα 9.13: *Περιγραφή Προσφοράς*

Στην συγκεκριμένη κλάση ο χρήστης έχει την δυνατότητα να ενημερωθεί για τις διαθέσιμες προσφορές *(Εικόνα 9.12)*. Στο πάνω μέρος της οθόνης υπάρχει η πιο πρόσφατη προσφορά ενώ στο κάτω μέρος οι παλιότερες. Επιλέγοντας μια προσφορά ανοίγει ένα pop up παράθυρο με την αναλυτική περιγραφή και την ημερομηνία της προσφοράς *(Εικόνα 9.13)*.

Contact.java

Στην συγκεκριμένη κλάση ο χρήστης έχει την δυνατότητα να στείλει email μέσω της φόρμας επικοινωνίας. Απαραίτητη προϋπόθεση είναι η συμπλήρωση του ονοματεπώνυμου και της διεύθυνσης email.

Κεφάλαιο 10ο

Επικοινωνία Android Εφαρμογής με Server

Το JSON RPC 2 (Remote Procedure Call) αποτελεί ένα «ελαφρύ» πρωτόκολλο επικοινωνίας μεταξύ κινητών συσκευών (clients) και απομακρυσμένων υπολογιστών-server.

Υπάρχουν δύο πρωτόκολλα του JSON-RPC, το JSON-RPC 1.0 και 2.0. Παρότι το δεύτερο αποτελεί την εξέλιξη του πρώτου, παρουσιάζουν κάποιες βασικές διαφορές. Το JSON-RPC 1.0 σχεδιάστηκε για να εξυπηρετεί peer-to-peer επικοινωνία χρησιμοποιώντας το TCP πρωτόκολλο, το HTTP πρωτόκολλο υποστηρίζεται αλλά με κάποιους περιορισμούς. Σε αντίθεση με το JSON-RPC 2.0 που υποστηρίζει κυρίως HTTP κλήσεις.

- JSON 1.0 αίτημα:

```
{"method":"add","params":[3,4],"id":0}
```

  - Το πεδίο «method» περιγράφει την απομακρυσμένη μέθοδο,
  - Το πεδίο «params» περιγράφει ένα πίνακα με RPC ορίσματα. Η σειρά των ορισμάτων επηρεάζει την αντίστοιχη σειρά στον απομακρυσμένο server,
  - Το πεδίο «id» είναι χαρακτηριστικό για κάθε κλήση και είναι αυτό που συνδέει τα αιτήματα από τον client με τις απαντήσεις από τον server.

- JSON 1.0 απάντηση:

```
{"error": null, "result": 5, "id": 0}
```

  - Το πεδίο «result» περιγράφει την τιμή που επιστρέφει ο server,
  - Το πεδίο «error» περιγράφει κάποιο σφάλμα που μπορεί να προέκυψε κατά την επικοινωνία.

Το JSON-RPC 2.0 επεκτείνει το JSON-RPC 1.0 με διάφορους τρόπους:

- Προκειμένου να παρέχει «προς τα πίσω» υποστήριξη, τα αιτήματα και οι απαντήσεις που στέλνονται και λαμβάνονται αντίστοιχα, περιλαμβάνουν το πεδίο "jsonrpc": "2.0",

  {"id":2,"jsonrpc":"2.0","method":"get_stores","params":[1]}
- Το πεδίο «error» δεν είναι πλέον αφηρημένο, αλλά συγκεκριμένης μορφής και περιέχει "code", "message" and "data".

10.1 JSON-RPC 2.0, Stores.java, Notifications.java

Αντίστοιχες JSON-RPC 2.0 κλάσεις του android sdk έχουν χρησιμοποιηθεί και για την αναπαράσταση των δεδομένων στην εφαρμογή μας, μερικές εκ των οποίων είναι η JSONArray, η JSONObject κτλ.

Στην εφαρμογή μας έγινε χρήση του παραπάνω πρωτοκόλλου δύο φορές, μία για την κλάση Stores.java και μία για την Notifications.java. Και στις δύο περιπτώσεις λαμβάνουμε δεδομένα σε συγκεκριμένη μορφή, αποθηκεύονται σε έναν πίνακα τύπου JSONArray και επεξεργάζονται αντίστοιχα.

Stores.java – αποστολή αιτήματος για την λήψη καταστημάτων

```
//create a Vector
specific_vec_store = new Vector<Integer>();
specific_vec_store.add(store_id);


NetEngine ne = NetEngine.getInstanceWithDelegate(this);
try{
   //Start Loading Bar
   String message =
   getResources().getString(R.string.loading_message);
   dialog.setMessage(message) ;
   dialog.setIndeterminate(true);
```

```
    dialog.setCancelable(true);

    dialog.show();


ne.executeJSONRPC("http://www.kratisinow.gr/android/json/message",
                  "specific_store",

                   specific_vec_store,

                   1);

    } catch (Exception e) {}
```

Βλέπουμε παραπάνω ότι έχει πραγματοποιηθεί μία JSON-RPC 2.0 κλήση με τέσσερα ορίσματα:

- *url*: http://www.kratisinow.gr/android/json/message

  Δηλώνει το url που γίνεται η κλήση.

- *method*: specific_store

  Δηλώνει την μέθοδο που υπάρχει στον server που εξυπηρετεί την συγκεκριμένη κλήση.

- *data*: specific_vec_store

  Δηλώνει τα στοιχεία που στέλνονται κατά την κλήση.

- *id κλήσης*: 1

  Δηλώνει το id της κλήσης σύμφωνα με το οποίο θα σταλεί η απάντηση.

ΜΕΡΟΣ 3ο - IPHONE

Κεφάλαιο 11ο

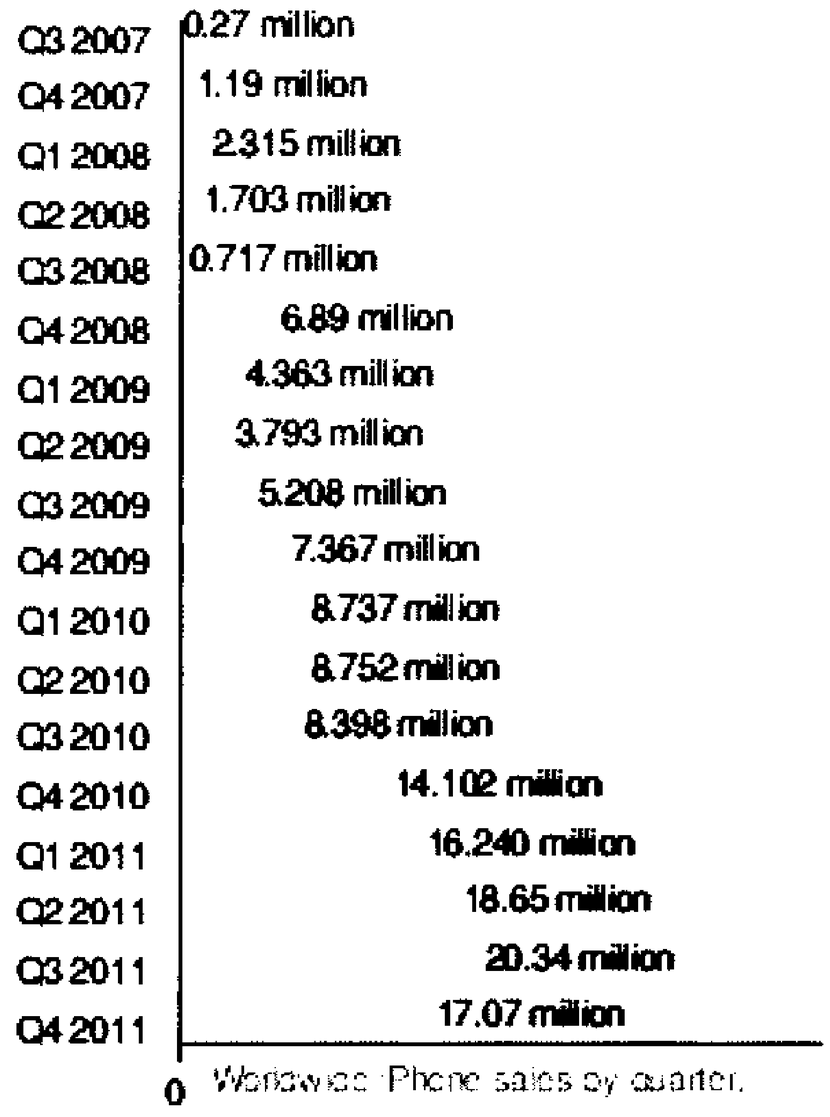
iPhone

11.1 Εισαγωγή – Ιστορικά Στοιχεία

Η εταιρία Apple παρουσίασε το πρώτο της Smartphone στις 9 Ιανουαρίου 2007 στο Macworld event, στο San Francisco. Η εμφάνιση της συσκευής στα καταστήματα της Αμερικανικής αγοράς, έγινε στις 29 Ιουνίου 2007. Μετά ακολούθησε το λανσάρισμα της συσκευής και σε άλλες μεγάλες αγορές του κόσμου, όπως Αγγλία, Γαλλία, Γερμανία.

Ένα χρόνο αργότερα, η Apple λάνσαρε το νέο της μοντέλο iPhone 3G. Οι πωλήσεις έφτασαν τα 6.1 εκατομμύρια συσκευές, μέσα σε 5 τετράμηνα. Οι πωλήσεις αυξήθηκαν σταθερά και στο τέλος του οικονομικού έτους 2010, οι συσκευές που είχαν πωληθεί, ανέρχονται στα 73.5 εκατομμύρια. Το έτος 2010/2011, το iPhone κατέχει μερίδιο στην αγορά που αγγίζει το 4% όλων το κινητών τηλεφώνων με την apple να κερδίζει περισσότερο από το 50% των κερδών που παράγει όλη η αγορά των κινητών.

Στις 6 Μαρτίου του 2008 η εταιρία ανακοίνωσε και παρουσίασε το iPhone SDK, που έδινε την δυνατότητα ανάπτυξης λογισμικού από τρίτους. Η διάθεση του Software Development Kit είναι δωρεάν και απαιτεί την εγγραφή στο Apple Developer Connection, καταβάλλοντας ετήσια συνδρομή. Η φιλοξενία των εφαρμογών γίνεται στο App Store. Το ηλεκτρονικό κατάστημα εφαρμογών αριθμεί 534,424 εφαρμογές με μέσο όρο κόστους τα 1,97$/application. Η εταιρεία έχει ανακοινώσει κέρδη της τάξης των 3.6$ δισεκατομμυρίων από την πώληση εφαρμογών *(Εικόνα 11.1)*.



Εικόνα 11.1: *Πωλήσεις συσκευών*

Στα κεφάλαια που ακολουθούν γίνεται αναφορά στην δομή του λειτουργικού και στα εργαλεία ανάπτυξης. Στο τέλος υπάρχει η παρουσίαση της εφαρμογής.

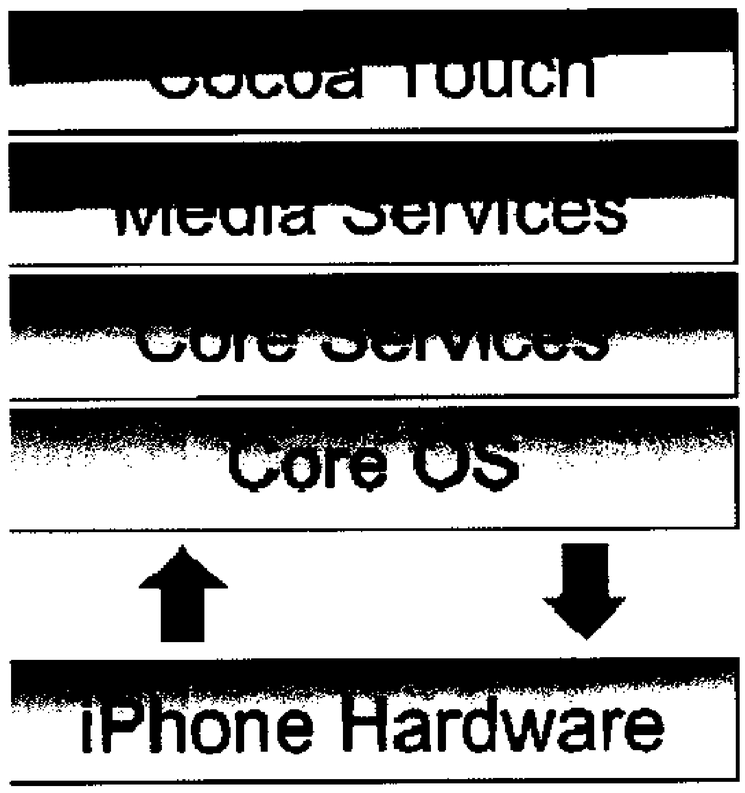
11.2 Δομή iOS

Το iOS SDK είναι το προγραμματιστικό εργαλείο που προσφέρει η Apple για την ανάπτυξη εφαρμογών στις κινητές πλατφόρμες της, δηλαδή στα iPhone, iPod Touch και iPad. Ακολουθεί μια πιο λεπτομερής παρουσίαση των χαρακτηριστικών του, καθώς και γνωριμία με τα βασικά προγραμματιστικά εργαλεία που περιέχονται σε αυτό.

Το iOS χρησιμοποίει ένα μέρος του πυρήνα XNU. Ο πυρήνας XNU αποτελεί την βάση του λειτουργικού συστήματος Mac OS X. Αρχικά αναπτύχτηκε από την εταιρία NeXT η οποία εξαγοράστηκε από την Apple που συνέχισε την ανάπτυξή του.

Το iOS χρησιμοποιεί το ίδιο λογισμικό για την ανάπτυξη εφαρμογών για iDevices που χρησιμοποιείται και στο Mac OS X, το οποίο ονομάζεται Xcode. Το iOS απαρτίζεται από διαφορετικά software layers *(Εικόνα 11.2)*, καθένα από τα οποία

παρέχει programming frameworks για την ανάπτυξη εφαρμογών που τρέχουν στην κορυφή του παρακάτω stack.



Εικόνα 11.2: Layers iOS

Cocoa Touch

Το Cocoa Touch είναι ένα API για τον σχεδιασμό λογισμικού σε iDevices και είναι γραμμένο στην γλωσσά προγραμματισμού Objective-C. Το Cocoa Touch χειρίζεται διάφορα μέρη των εφαρμογών όπως:

- Χειρίζεται τα γεγονότα από την χρήση πολλαπλών σημείων αφής στην οθόνη του κινητού. Για παράδειγμα με αυτόν τον τρόπο γίνεται εφικτός προγραμματισμός διαφορετικών συμπεριφορών της εφαρμογής ανάλογα με το πόσα δάκτυλα ακούμπησε ο χρήστης στην οθόνη ή αν έκανε κίνηση προς τα πάνω ή προς τα κάτω,
- Τα γεγονότα στην αλλαγή κλίσης της συσκευής (Accelerometer),
- Την υποστήριξη της κάμερας από εφαρμογές,
- Ελέγχει την ιεραρχία των διαφόρων προβολών (views). Κάθε φορά που πρέπει για κάποιο λόγο να αλλάξει η προβολή στοιχείων στην οθόνη ή κάθε φορά που θέλουμε να προβάλλουμε κάτι διαφορετικό σε αυτήν,
- Την μετατροπή των δεδομένων ανάλογα με την γεωγραφική τοποθεσία που βρίσκεται ο χρήστης ή ανάλογα την χωρά, αλλάζει η προβολή κάποιων δεδομένων όπως η ημερομηνία κλπ.

Media Services

Η διαχείριση των πολυμέσων γίνεται από διάφορα API τα οποία περιλαμβάνονται στο Media κομμάτι του iOS SDK. Αυτά είναι:

- Το OpenAL (open audio library) το οποίο είναι ένα λογισμικό που χρησιμοποιείται σε πολλές διαφορετικές πλατφόρμες. Είναι σχεδιασμένο έτσι ώστε να αποδίδει ποιοτικό ήχο, να ελέγχει την μείξη και εγγραφή ήχου, να υποστηρίζει την αναπαραγωγή βίντεο και να υποστηρίζει την προβολή διαφορετικών format εικόνας από το κύκλωμα γραφικών,
- Το Core Animation το οποίο είναι ένα API που παράγει κινούμενα περιβάλλοντα χρήσης,
- Το Quartz, γραφικό περιβάλλον της Apple το οποίο υποστηρίζει την σχεδίαση δισδιάστατων γραφικών και την δημιουργία κώδικα για την υλοποίησή τους,
- Το OpenGL ES που αποτελεί ένα μέρος του OpenGL 3D API και έχει σχεδιαστεί για κινητές πλατφόρμες. Χρησιμοποιείται για την παραγωγή δισδιάστατων και τρισδιάστατων γραφικών.

Core Services

Τα Core Services περιέχουν API που έχουν να κάνουν με την διαχείριση του δικτύου αλλά και των δεδομένων. Αυτά είναι τα ακόλουθα:

- Οι λειτουργίες Δικτύου (Networking). Ελέγχει όλες τις λειτουργίες δικτύου από την σύνδεση με αυτό μέχρι την αποστολή και λήψη δεδομένων,
- Η ενσωματωμένη βάσης δεδομένων SQLite,
- Το Core Location που ελέγχει όλες τις λειτουργίες εύρεσης τοποθεσίας μέσω του ενσωματωμένου GPS και των κεραιών κινητής τηλεφωνίας,
- Την εκτέλεση διάφορων Threads,
- Το Core Motion.

Core OS

Ο πυρήνας του OS αποτελείται από τα ακόλουθα:

- Το TCP/IP για την διασύνδεση των εφαρμογών με το διαδίκτυο,
- Τα Sockets,
- Την διαχείριση ενέργειας,
- Το σύστημα αρχείων,
- Την ασφάλεια των δεδομένων.

11.3 Τα εργαλεία προγραμματισμού του iOS SDK

Τα εργαλεία που περιλαμβάνονται στο iOS SDK είναι αρκετά. Υπάρχουν εργαλεία τα οποία μπορούν να χρησιμοποιηθούν για την δημιουργία οποιουδήποτε είδους εφαρμογής. Σύμφωνα και με την ίδια την Apple, τα εργαλεία ανάπτυξης που χρησιμοποιήθηκαν για την κατασκευή των native εφαρμογών είναι τα ίδια με αυτά που παρέχονται στους τρίτους κατασκευαστές λογισμικού.

Το Xcode

Το Xcode είναι το περιβάλλον ανάπτυξης των εφαρμογών (I.D.E) *(Εικόνα 11.3)*. Στο Xcode βρίσκεται ουσιαστικά όλος ο κώδικας της εφαρμογής. Εδώ είναι το μέρος που δημιουργούμε τις κλάσεις του προγράμματος μας, δηλώνουμε τις μεταβλητές, και γενικότερα προγραμματίζουμε όλη την συμπεριφορά της εφαρμογής.

Το πρώτο πράγμα που βλέπουμε όταν ανοίγουμε το Xcode είναι ένα παράθυρο που μας καλωσορίζει. Από εδώ διαλέγουμε αν θέλουμε να ξεκινήσουμε μια νέα εφαρμογή, εάν θέλουμε να ανοίξουμε κάποια πρόσφατη εφαρμογή που έχουμε ξεκινήσει ή αν θέλουμε βοήθεια στην εκμάθηση του προγράμματος. Εάν επιλέξουμε τη δημιουργία μιας νέας εφαρμογής τότε θα μας ζητηθεί να επιλέξουμε το πρότυπο που θα χρησιμοποιήσουμε και αργότερα θα πρέπει να την ονομάσουμε.

Τα πρότυπα είναι στην ουσία έτοιμες κλάσεις που μας δίνει απευθείας το Xcode ανάλογα με το είδος της εφαρμογής που θέλουμε να δημιουργήσουμε. Υπάρχουν 6 διαφορετικά πρότυπα για προγραμματισμό στο iPhone:

- Το πρότυπο πλοήγησης (Navigation-based) το οποίο δίνει τις βασικές κλάσεις και τον κώδικα για να διευκολύνει την δημιουργία μιας εφαρμογής που σκοπό θα έχει την πλοήγηση απο τη μια προβολή δεδομένων στην επόμενη,
- Το πρότυπο OpenGL ES που δίνει ένα αρχικό στάδιο μιας εφαρμογής που χρησιμοποιεί το OpenGL στην αρχική προβολή,
- Το πρότυπο με μπάρα επιλόγων (Tab Bar) έτσι ώστε να δημιουργήσουμε μια εφαρμογή που να εναλλάσσεται σε πολλές προβολές μεσώ μιας μπάρας επιλογής στο κάτω μέρος της εφαρμογής,
- Το πρότυπο μιας προβολής που δημιουργεί μια κλάση και με μια κεντρική προβολή,
- Το πρότυπο μιας εφαρμογής παραθύρου που απλά δίνει τα πιο βασικά αρχεία για την δημιουργία μιας οποιαδήποτε εφαρμογής.

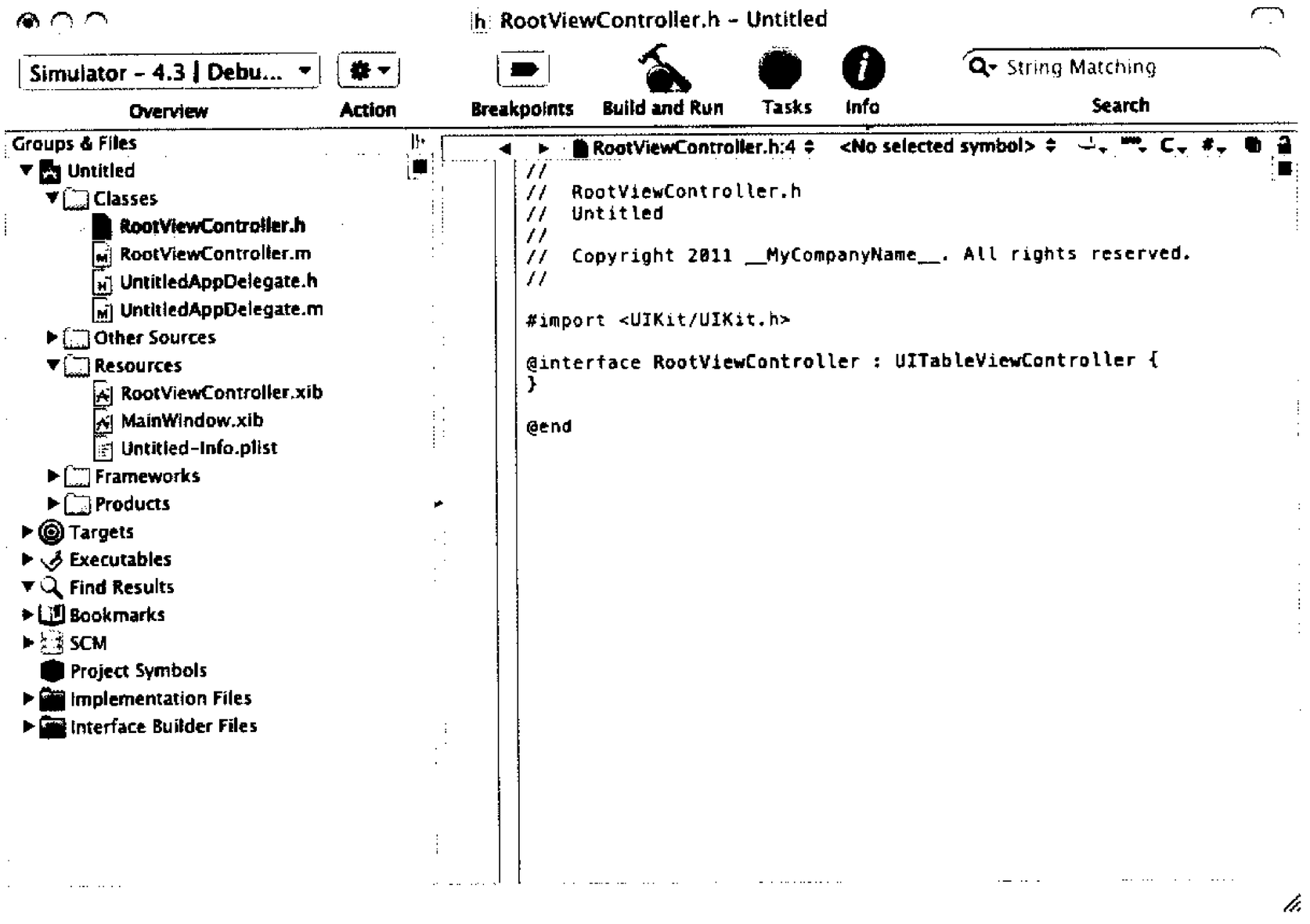
Αφού επιλεγεί κάποιο πρότυπο, το Xcode θα δημιουργήσει τις κατάλληλες κλάσεις και θα προσθέσει τα αρχεία που χρειάζονται καθώς και τα απαραίτητα Frameworks.

Στο αριστερό μέρος του παραθύρου μπορούμε να διακρίνουμε το υπό-παράθυρο με τίτλο Groups & Files. Εδώ εμφανίζονται οι κλάσεις που χρησιμοποιεί η εφαρμογή και αν επιλέξουμε κάποια από αυτές τότε μπορούμε να προσθέσουμε κώδικα στο δεξί μέρος του παραθύρου, οπού και προβάλλονται τα περιεχόμενα της. Ένας άλλος πολύ σημαντικός φάκελος που μπορούμε να δούμε τα περιεχόμενα του με τον ίδιο τρόπο, είναι ο φάκελος Resources, που περιέχει την προβολή της εφαρμογής μας. Είναι το αρχείο με τύπο .xib. Πρόκειται για αρχείο που ανοίγει προς επεξεργασία μέσω του δεύτερου προγράμματος που παρουσιάζεται στη συνέχεια, του Interface Builder. Όμως στο φάκελο αυτό υπάρχει και ένα άλλο σημαντικό αρχείο, το -info.plist, το οποίο περιέχει μερικές χρήσιμες ρυθμίσεις για την εφαρμογή, όπως ποιο θα είναι το όνομα της εφαρμογής όταν εγκαθίσταται στη συσκευή αλλά και ποιο θα είναι το εικονίδιο της.

Στη περίπτωση που θέλουμε να δημιουργήσουμε μια νέα κλάση, τότε θα πρέπει να πατήσουμε δεξί κλικ πάνω στον φάκελο που θέλουμε να την προσθέσουμε (συνήθως στον φάκελο Classes) και να επιλέξουμε Add και στη συνέχεια New File. Θα εμφανιστεί ένα νέο παράθυρο που μας δίνει διάφορα πρότυπα ανάλογα με το τι ακριβώς θέλουμε να κάνει η νέα κλάση μας.

Στο πάνω αριστερό μέρος του παραθύρου υπάρχει ένα αναδυόμενο μενού στο οποίο αναγράφεται η λέξη Simulator. Μέσω αυτού του μενού μπορούμε να διαλέξουμε που θέλουμε να εκτελέσουμε το πρόγραμμα μας. Αυτό μπορεί να γίνει είτε στην συσκευή που είναι συνδεδεμένη με τον υπολογιστή μας ή στον iPhone Simulator. Με το κουμπί στη Build and Run μπορούμε να δοκιμάσουμε την εφαρμογή μας.

Το Xcode έχει αρκετά εργαλεία που μπορούν να μας βοηθήσουν στο να γράψουμε κώδικα. Το μενού Help δινει χρήσιμες πληροφορίες για τις μεθόδους που μπορούμε να χρησιμοποιήσουμε ενώ υπάρχει και η δυνατότητά να πατήσουμε το κουμπί Command και να κάνουμε διπλό κλικ σε οποιαδήποτε λέξη κλειδί ή μέθοδο και να δούμε την δήλωσή της στο documentation της Apple. Επίσης εμφανίζει με κόκκινη γραμματοσειρά τις λέξεις κλειδιά αλλά και τις μεθόδους και με ανοικτό μπλε τις μεταβλητές, ενώ ανάλογα με το τι ξεκινούμε να γράφουμε, μας προσφέρει αυτόματη συμπλήρωση λέξεων.



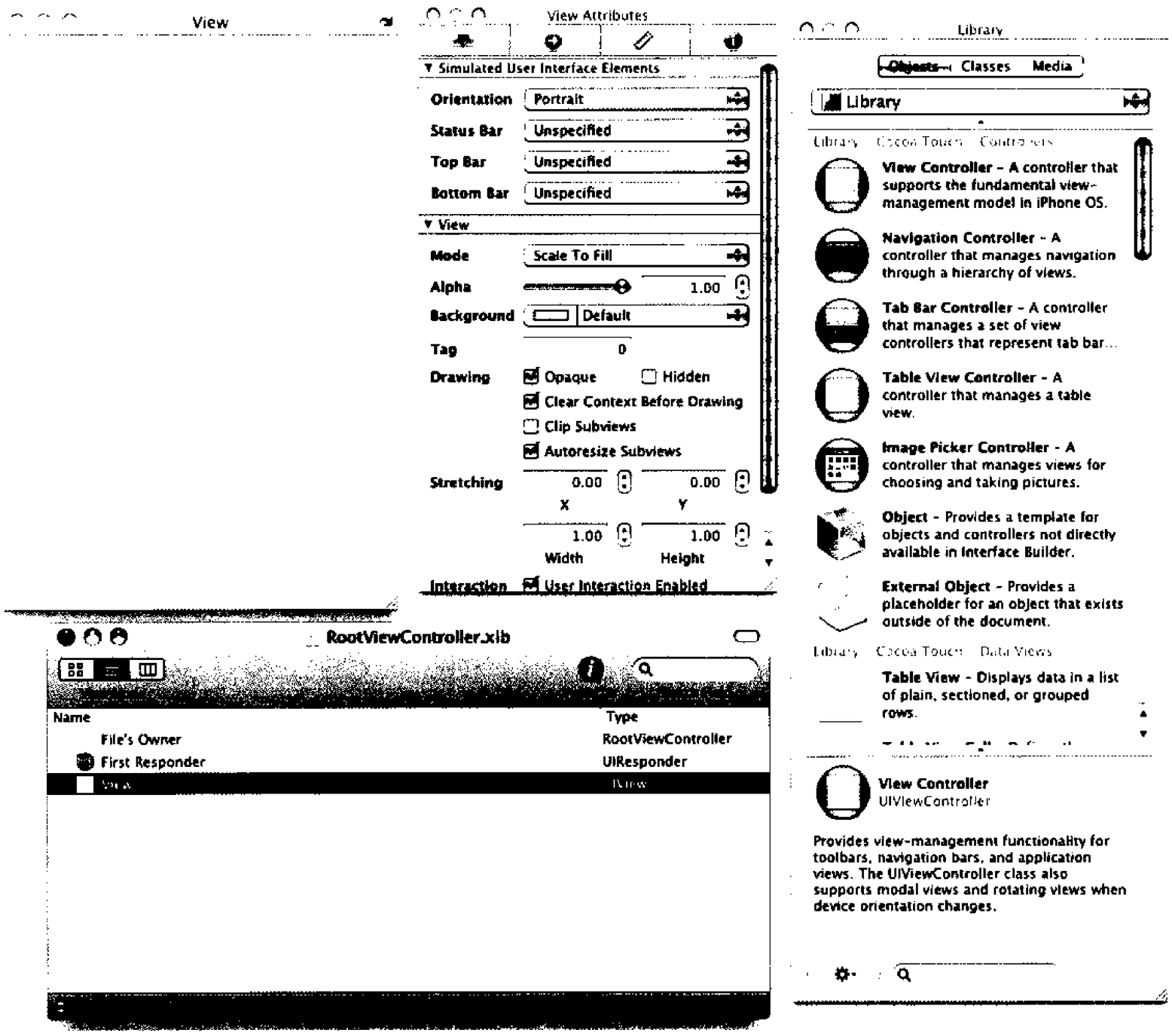
Εικόνα 11.3: *Περιβάλλον χρήσης του Xcode*

Ο Interface Builder

Για να ξεκινήσει ο Interface Builder, πρέπει να γίνει διπλό κλικ στο αρχείο με τύπο .xib που συνήθως βρίσκεται στον φάκελο Resources του υπό-παραθύρου Groups & Files του Xcode. Ο Interface Builder είναι ένα πρόγραμμα που ασχολείται με την επεξεργασία του User Interface *(Εικόνα 11.4)*. Μέσω αυτού τοποθετούνται κουμπιά, τίτλοι, μονοδιάστατοι πίνακες, εικόνες και γενικότερα οποιοδήποτε component προσφέρει το iOS. Ο Interface Builder αποτελείται από τέσσερα διαφορετικά παράθυρα. Το πιο βασικό είναι το κεντρικό παράθυρο, το οποίο περιέχει το όνομα του αρχείου που έχουμε ανοίξει. Σε αυτό το παράθυρο μπορούμε να δούμε τα αντικείμενα που περιέχει η προβολή μας. Το αντικείμενο File's Owner περιέχει όλες τις μεθόδους που έχουμε περάσει στην κλάση με την οποία συνδέεται η προβολή μας και μέσω αυτού μπορούμε να συνδέσουμε τα αντικείμενα που χρησιμοποιούμε στο παράθυρο προβολής με τις αντίστοιχες μεθόδους.

Το επόμενο παράθυρο με τίτλο View, περιέχει την προβολή. Πάνω στην προβολή, τοποθετούνται τα αντικείμενα, τα οποία συνθέτουν το User Interface. Αυτά τα αντικείμενα επιλέγονται από το τρίτο παράθυρο, που ονομάζεται Library.

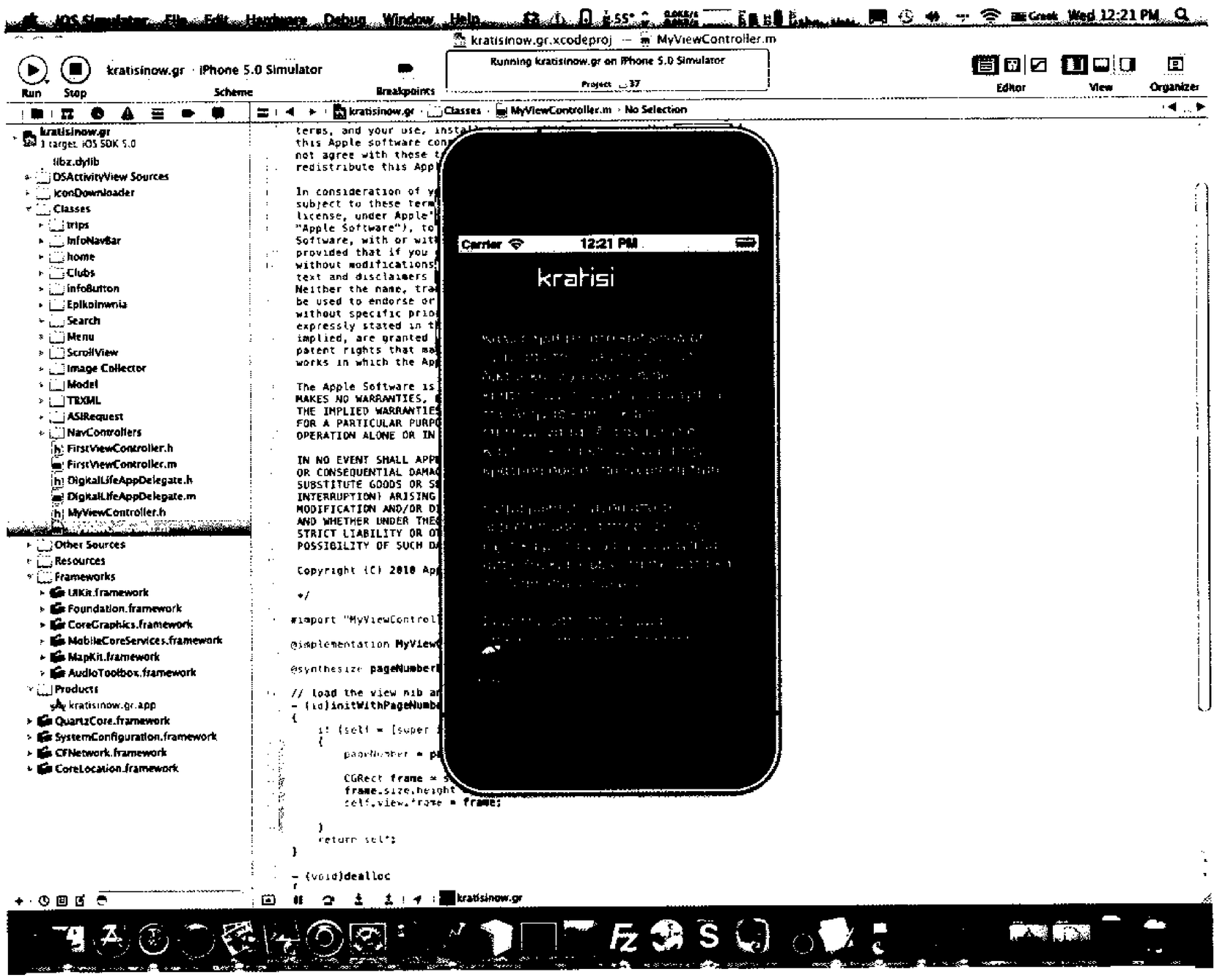
Εκεί υπάρχουν όλα τα αντικείμενα, ομαδοποιημένα με διάφορους τρόπους. Μπορούμε να τα τοποθετήσουμε στο παράθυρο προβολής κάνοντας τα drag and drop. Το τελευταίο παράθυρο είναι ο Inspector. Μέσω αυτού δίνονται οι ιδιότητες των αντικειμένων. Χωρίζεται σε τέσσερα μέρη: το μέρος με τα Attributes, το οποίο περιέχει τις ιδιότητες που είναι διαφορετικές ανάλογα το αντικείμενο που έχουμε επιλέξει, το Connections που περιέχει τις συνδέσεις του αντικείμενου με τις μεθόδους, το Size που ελέγχει το μέγεθος του παράθυρου και την τοποθέτηση του στην προβολή και τέλος το Identity, όπου περιέχει το όνομα της κλάση, από την οποία το αντικείμενο κληρονομεί τις ιδιότητές του.



Εικόνα 11.4: *Το περιβάλλον χρήσης του Interface Builder*

Ο iPhone Simulator

Οποιαδήποτε στιγμή μπορεί να γίνει debugging της εφαρμογή, πατώντας το κουμπί Build and Run και να προσωμοιώνοντας την λειτουργία της εφαρμογής, στον Simulator *(Εικόνα 11.5)*. Το ποντίκι του υπολογιστή, προσομειώνει τα δακτυλά μας πάνω στην οθόνη του εικονικού iPhone. Πατώντας το κουμπί Option απο το πληκρολόγιο, προσομειώνονται οι κινήσεις pitch in και out των δακτύλων μας και εκτελούνται οι ενέργειες zoom-in και zoom-out, ενώ πατώντας το κουμπί Command μαζί με τα βελάκια αλλάζει η γωνία της συσκευής κατά 90 μοίρες (οριζόντια προβολή).



Εικόνα 11.5: iPhone Simulator

11.4 Η γλώσσα προγραμματισμού Objective-C

Είναι μια γλώσσα κατασκευασμένη από τους Brad Cox και Tom Love το 1986. Είναι μια αντικειμενοστραφής γλώσσα προγραμματισμού και έχει μεγάλες επιρροές από την SmallTalk και την C. Κυρίως χρησιμοποιείται για προγραμματισμό συσκευών της Apple και γενικά συστημάτων που βασίζονται στο OpenStep. Εκτός από τα μηχανήματα της Apple μπορεί να γίνει μεταγλώττιση και σε πολλά άλλα λειτουργικά κυρίως σε αυτά που βασίζονται στο Unix και έχουν τη δυνατότητα εκτέλεσης του γνωστού μεταγλωττιστή gcc ή και του Clang.

Πάνω στη Objective-C βασίζονται πολλές αντικειμενοστραφείς γλώσσες μεταξύ των οποίων και η Java. Ο τρόπος σύνταξής μοιάζει με την SmallTalk και την C++, αλλά η λογικής βασίζεται καθαρά στην αντικειμενοστρέφια. Για παράδειγμα:

object->methodName(argument) C++

[object methodName:argument] Objective-C

object.methodName(argument) Java

11.4.1 Δήλωση και ορισμός κλάσεων

Όπως ισχύει και σε σχεδόν όλες τις αντικειμενοστραφείς γλώσσες προγραμματισμού, στην Objective-C οι κλάσεις παρέχουν τα θεμέλια που επιτρέπουν την λήψη δεδομένων και την χρησιμοποίηση μεθόδων για την επεξεργασία τους. Τα αντικείμενα είναι συγκεκριμένες καταστάσεις μιας κλάσης και περιέχουν τα δικά τους δεδομένα ανά περίπτωση αλλά και τους δείκτες (pointers) οι οποίοι δείχνουν στις μεθόδους που περιλαμβάνονται σε κάθε κλάση. Οι κλάσεις χωρίζονται σε 2 κατηγορίες: το Interface (διεπαφή ή προβολή δεδομένων) και το implementation (εφαρμογή). Το Interface περιέχει τη δήλωση της κλάσης και συνήθως βρίσκεται σε ένα αρχείο που έχει κατάληξη .h. Το implementation περιέχει τον κώδικα που καθορίζει μια κλάση και συνήθως βρίσκεται σε ένα αρχείο που έχει κατάληξη .m. Ας τα δούμε πιο αναλυτικά.

Δήλωση μιας κλάσης

Ας υποθέσουμε ότι θέλουμε να φτιάξουμε μια κλάση η οποία θα δέχεται μια εντολή από τον χρήστη και θα τυπώνει στην οθόνη "Hello World". Θα πρέπει κατ' αρχήν να δηλώσουμε αυτήν την κλάση. Η δήλωση της ξεκινάει με την λέξη κλειδί @interface ακολουθούμενη από το όνομα της κλάσης που δηλώνουμε και στη συνέχεια με το σύμβολο ":" το οποίο ακολουθείται από το όνομα της κλάσης από την οποία κληρονομεί:

```
@interface HelloWorldViewController : UIViewController
```

Μια κλάση στην Objective-C δεν μπορεί να κληρονομήσει από πολλές κλάσεις αλλά η κλάση από την οποία κληρονομεί μπορεί με την σειρά της να κληρονομεί από μια άλλη. Στη δική μας περίπτωση η κλάση μας HelloWorldViewController κληρονομεί από την UIViewController η οποία και αυτή με την σειρά της κληρονομεί από την UIResponder, η οποία τελικά κληρονομεί από την NSObject, που είναι η κλάση ρίζα στις περισσότερες ιεραρχίες κλάσεων στην Objective-C. Μετά από αυτή

την πρώτη γραμμή οι δηλώσεις των μεταβλητών βρίσκονται μέσα σε αγκύλες. Μετά ακολουθούν οι δηλώσεις των μεθόδων και των properties που σχετίζονται με την κλάση. Τα properties είναι ένας εύκολος τρόπος να δημιουργούμε τα set και τα get για κάθε μεταβλητή. Τέλος κλείνουμε την δήλωση της κλάσης μας με την λέξη κλειδί @end:

```
#import <UIKit/UIKit.h>

@interface HelloWorldViewController : UIViewController {

      UIButton *button; UILabel *label;

}

@property (nonatomic, retain) IBOutlet UILabel *label;

-(IBAction)sayHello:(id) sender;

@end
```

Συνεχίζοντας με το ίδιο παράδειγμα, πρέπει πια να υλοποιήσουμε με κώδικα ότι έχουμε δηλώσει στο αρχείο .h. Η εφαρμογή του κώδικά μας ξεκινάει με την λέξη κλειδί @implementation και τελειώνει με την λέξη κλειδί @end:

@implementation HelloWorldViewController ... @end

Στην αρχή πρέπει να χρησιμοποιήσουμε την λέξη κλειδί @synthesize έτσι ώστε να δημιουργηθούν για μας τα set και get των properties και στην συνέχεια να εφαρμόσουμε τις μεθόδους που δηλώσαμε παραπάνω:

```
#import "HelloWorldViewController.h"

@implementation HelloWorldViewController

@synthesize label;

-(IBAction) sayHello:(id) sender {

      label.text = @"Hello World";

}

- (void)didReceiveMemoryWarning {

      [super didReceiveMemoryWarning];

}
```

```
- (void)viewDidUnload { }

(void)dealloc { [label release];

        [button release];

        [super dealloc];

} @end
```

Δήλωση αντικειμένων

Όταν οι ίδιες οι μεταβλητές είναι αντικείμενα, παραδείγματος χάριν όταν η κλάση HelloWorldViewController δηλώνει μια UIButton και μια UILabel μεταβλητή, πρέπει πάντα να χρησιμοποιείται ένας τύπος δείκτη. Όμως με την Objective-C συμβαίνει κάτι ενδιαφέρον: υποστηρίζει και σαφείς δηλώσεις και ασαφείς δηλώσεις αντικείμενων. Ακολουθεί μια σαφής δήλωση:

UIButton *button; Εδώ δηλώνουμε το είδος του αντικείμενου, δηλώνοντας ξεκάθαρα τι είναι.

Παρακάτω όμως κάνουμε μια ασαφή δήλωση, δηλώνοντας ότι το αντικείμενο είναι ένα id: id button

Ο τύπος id είναι ένας γενικός C τύπος τον οποίο η Objective-C χρησιμοποιεί για να δείξει ένα αυθαίρετο αντικείμενο. Είναι ένας γενικός τύπος που αντιπροσωπεύει οποιοδήποτε αντικείμενο ανεξαρτήτως κλάσης και μπορεί να χρησιμοποιηθεί ως τρόπος αποθήκευσης τόσο μιας κλάσης όσο και μιας αναφοράς προς ένα αντικείμενο. Όλα τα αντικείμενα οπότε είναι τύπου id. Αυτό μπορεί να αποδειχτεί πολύ χρήσιμο καθώς αν θέλουμε να φτιάξουμε μια γενική κλάση που θα περιείχε μια συνδεδεμένη λίστα, ο τύπος κάθε στοιχείου της, μπορεί να είναι τύπου id και έτσι θα μπορούμε να αποθηκεύσουμε σε αυτήν οποιοδήποτε τύπου αντικείμενα.

Properties

Η δήλωση των properties χρησιμοποιώντας την λέξη κλειδί @property είναι ένας εύχρηστος τρόπος να αποφύγουμε την δήλωση και συνήθως και την εφαρμογή των μεθόδων προσπέλασης για τις μεταβλητές-μελή μιας κλάσης. Μπορούμε να θεωρήσουμε την δήλωση ενός property ως αντίστοιχη της δήλωσης μεθόδων Set και Get. Μπορούμε επίσης να δηλώσουμε πως οι αυτοματοποιημένες αυτές μέθοδοι θα συμπεριφέρονται, δηλώνοντας συγκεκριμένες ιδιότητες. Στη κλάση HelloWorldViewController δηλώθηκε το property ως (nonatomic, retain):

@property (nonatomic, retain) IBOutlet UILabel *label;

Μπορούμε επίσης να δηλώσουμε και τα 2 properties ως IBOutlet. Αν και στην πραγματικότητα δεν είναι μέρος των ιδιοτήτων για μια δήλωση property, το IBOutlet δηλώνει ότι αυτό το αντικείμενο είναι ένα στοιχείο το οποίο μπορούμε να το συνδέσουμε με άλλα μέσω του Interface Builder.

Synthesize

Όταν δηλώνουμε ένα @property στο αρχείο .h, πρέπει επίσης να χρησιμοποιούμε το @synthesize στο αρχείο .m, εκτός αν υλοποιήσουμε εμείς τα set και get, όπως κάναμε και για το property label της κλάσης HelloWorldViewController:

Με την εντολή@synthesize label; ο compiler δημιουργεί τις μεθόδους προσπέλασης σύμφωνα με τις ιδιότητες που δηλώθηκαν στο property και έτσι μειώνεται σημαντικά ο κώδικα που πρέπει να γράψει ο προγραμματιστής.

Η σύνταξη με την χρήση τελείας

Όταν δηλώνεται μια μεταβλητή-μέλος ως property και στην γίνεται το synthesize, τότε είναι δυνατή η χρησιμοποίηση κάποιων ευκολιών που προσφέρει η Objective-C. Η σύνταξη με την χρήση τελείας (the dot syntax) βοηθά στη άμεση χρησιμοποίηση

των μεταβλήτών, χωρίς να χρειάζεται η κλήση μεθόδων. Για παράδειγμα μας επιτρέπει να γράψουμε το ακόλουθο:

label.text = @"hello World"

Αντί για:

[label setText:@"Hello World"];

Ο πρώτος τρόπος γραφής είναι σίγουρα πιο απλός και ευκολοδιάβαστος.

Δηλώνοντας Μεθόδους

Στο παράδειγμα της με την κλάση HelloWorldViewController, δηλώθηκε μια μέθοδος με όνομα sayHello:

```
#import <UIKit/UIKit.h>

@interface HelloWorldViewController : UIViewController {

      UILabel *label;

      UIButton *button;

}

@property (nonatomic, retain)

IBOutlet UILabel *label;

-(IBAction)sayHello:(id) sender;

@end
```

Το μείον (-) στην αρχή της δήλωσης της μεθοδου δείχνει τον τύπο της, που σε αυτή την περίπτωση είναι μια κατάσταση. Ένα συν (+) ορίζει μια κλάση-μέθοδο όπως παρακάτω:

+(void)aMethod:(id) anObject;

Η μέθοδος sayHello: παίρνει ένα αντικείμενο id ως είσοδο και σημειώνεται ως IBAction για τον Interface Builder. Όταν το IBAction περάσει από τον compiler, τότε αντικαθείσταται από ένα void, όπως επίσης το ίδιο συμβαίνει στο IBOutlet, αφού και

τα 2 αυτά αντικείμενα χρησιμοποιούνται για να δείχνουν στον Interface Builder ότι μπορούν να συνδεθούν. Αυτή η μέθοδος δέχεται ως είσοδο ένα αντικείμενο id καί εκτελείται όταν κάνει μια ενέργεια ο χρήστης, χωρίς να περιοριζόμαστε ως προς το ποια ενέργεια θα είναι αυτή. Στο περιβάλλον χρήστη, θέλουμε ένα κουμπί που όταν ο χρήστης το πατήσει, τότε θα εκτελεστεί η μέθοδος. Επομένως, στη προκειμένη περίπτωση το id θα είναι ένα UIButton που ο χρήστης χρησιμοποίησε, ώστε να τρέξει την μέθοδος. Μπορούμε να ανακτήσουμε το αντικείμενο UIButton χρησιμοποιώντας το αντικείμενο sender σε ένα UIButton:

```
UIButton theButton = (UIButton *) sender;
```

Είναι συνηθισμένη πρακτική στην Objective-C να καλούμε τέτοια αντικείμενα sender. Μπορούμε να μάθουμε το ακριβή τύπο δεδομένων ενός αντικειμένου id, χρησιμοποιώντας την μέθοδο isKindOfClass:

```
if([thisObject isKindOfClass:[anotherObject class]]) { ... }
```

Καλώντας Μεθόδους

Εάν θέλουμε να καλέσουμε μια μέθοδο μέσω ενός αντικείμενου, μπορούμε να το κάνουμε αυτό στέλνοντας ένα μήνυμα σε αυτό το αντικείμενο. Το μήνυμα πρέπει να περιέχει το όνομα της μεθοδου όπως επίσης και διάφορες παραμέτρους της. Τα μηνύματα πρέπει να κλείνονται από παρενθέσεις τύπου []. Το αντικείμενο που δέχεται το μήνυμα βρίσκεται στα αριστερά και οι παράμετροι στα δεξιά, με την παράμετρο να ακολουθείται από άνω κάτω τελείες. Εάν η μέθοδος δέχεται περισσότερα από ένα δεδομένα, είναι υποχρεωτικό να χρησιμοποιηθούν και αυτά, έτσι οι επόμενες παράμετροι ακολουθούνται και αυτοί με άνω κάτω τελείες. Αυτό επιτρέπει πολλαπλές μεθόδους με το ίδιο όνομα , οι οποίες έχουν διαφορετικά ορίσματα.

```
[anObject someMethod: anotherObject];
[anObject someMethod: anotherObject withAnotherArgument: yetAnotherObject];
```

Στον κώδικα που προηγήθηκε έχουμε δυο μεθόδους με ονόματα someMethod: και someMethod:withAnotherArgument:.

Οι μέθοδοι μπορούν να επιστρέψουν το αποτέλεσμα τους όπως φαίνεται παρακάτω:

output = [anObject someMethodWithOutput: anotherObject];

Μπορούν να χρησιμοποιούνται εμφωλεθμένες κλήσεις:

output = [anObject someMethodWithOutput: [anotherObject someOtherMethod]];

Στην Objective-C το αντικείμενο nil είναι σε λειτουργία ισάξιο με τον δείκτη NULL που υπάρχει σε άλλες C γλώσσες. Όμως αντίθετα από ότι σε περισσότερες από αυτές, είναι δυνατόν να καλέσουμε μεθόδους με nil χωρίς να καταρρεύσει η εφαρμογή μας. Εάν καλέσουμε μια μέθοδο με αντικείμενο το τύπο nil, τότε θα μας επιστρέψει ένα nil.

11.4.2 Διαχείριση μνήμης

Ο τρόπος με τον οποίο η Objective-C στο iOS διαχειρίζεται την μνήμη, διαφέρει απο γλώσες που έχουν managed memory όπως η Java. Στο iPhone είμαστε περιορισμένοι να χρησιμοποιούμε κάτι που ονομάζεται αναφορά μέτρησης (Reference Counting). Αυτό ακολουθεί μερικούς απλούς κανόνες, και μας δίνει την δυνατότητα να διαχειριζόμαστε την μνήμη που διαθέτουμε (allocate).

Δημιουργώντας Αντικείμενα

Μπορούμε να δημιουργήσουμε ένα αντικείμενο με 2 τρόπους. Όπως φαίνεται και στον κώδικα που ακολουθεί, μπορούμε να διαθέσουμε μνήμη απευθείας με την λέξη alloc και να την αρχικοποιήσουν με την λέξη init ή την κατάλληλη initWith μέθοδο (π.χ. Ένα αντικείμενο NSString χρησιμοποιεί την μέθοδο initWithString):

NSString *string = [[NSString alloc] init]; NSString *string = [[NSString alloc] initWithString:@"This is a string"];

Εναλλακτικά μπορούμε να χρησιμοποιήσουμε μια μέθοδο-κατασκευαστή. Για παράδειγμα η κλάση NSString έχει μια μέθοδο stringWithString η οποία επιστρέφει

ένα αντικείμενο NSString: NSString *string = [NSString stringWithString:@"This is a string"];

Με τους 2 πρώτους τρόπους είμαστε υποχρεωμένοι να απελευθερώσουμε την μνήμη που διαθέσαμε με την λέξη alloc. Εάν δημιουργήσουμε ένα αντικείμενο με το alloc, πρέπει να τον απελευθερώσουμε (release) μετά. Αντιθέτως με τον τρίτο τρόπο το αντικείμενο θα απελευθερώσει την μνήμη που χρησιμοποιεί αυτόματα (autorelease). Δεν πρέπει ποτέ να απελευθερώνουμε μόνοι μας ένα τέτοιο αντικείμενο καθώς κάτι τέτοιο θα προκαλέσει κατάρρευση της εφαρμογής μας. Ένα τέτοιο αντικείμενο, τις περισσότερες φορές, θα απελευθερωθεί στο τέλος της χρήσης του εκτός εάν έχουμε χρησιμοποιήσει την λέξη retain (διατήρηση).

Αυτόματη απελευθέρωση μνήμης

Αυτόματη απελευθέρωση μνήμης (autorelease pool) ονομάζουμε την διαδικασία κατά την οποία στέλνεται ένα μήνυμα απελευθέρωσης μνήμης σε ένα αντικείμενο, αφού αυτό πρώτα έχει παύσει να χρησιμοποιείται από την μέθοδο από την οποία δημιουργήθηκε. Κάθε αντικείμενο που δημιουργείται με αυτόν τον τρόπο αποθηκεύεται μαζί με άλλα τέτοιου είδους αντικείμενα σε ένα συγκεκριμένο μέρος (pool) και με το πέρας της χρησιμοποίησης όλων αυτών των στοιχείων, όλα μαζί απελευθερώνονται. Όλες οι εφαρμογές iOS είναι υποχρεωμένες να έχουν μια autorelease pool και για αυτό ο Xcode δημιουργεί μια αυτόματα μέσα στο αρχείο main

```
int main(int argc, char *argv[]) {

    NSAutoreleasePool * pool = [[NSAutoreleasePool alloc] init];

    int retVal = UIApplicationMain(argc, argv, nil, nil);

    [pool release];

    return retVal;

}
```

Η autorelease pool δημιουργείται πριν μπει στην κεντρικό βρόγχο και στη συνέχεια απελευθερώνει την μνήμη, αφού έχει βγει από τον βρόγχο. Μια επιπλέον

εσωτερική autorelease pool δημιουργείται στην αρχή κάθε κύκλου γεγονότων και απελευθερώνεται στο τέλος του.

Η ανάγκη και η ύπαρξη της αυτόματης απελευθέρωσης μνήμης έχει περισσότερο νόημα μόλις αντιληφθούμε τον στόχο της, ο οποίος είναι να μεταφέρει τον έλεγχο του κύκλου ζωής ενός αντικείμενου από την ιδιοκτησία ενός αντικειμένου, σε ένα άλλο χωρίς να απελευθερώνει αμέσως το αντικείμενο.

Η διάθεση, διατήρηση, αντιγραφή και απελευθέρωση μνήμης

Αν και η αυτόματη απελευθέρωση μνήμης είναι χρήσιμη, πρέπει να είμαστε προσεχτικοί κατά την χρησιμοποίηση της καθώς μπορεί με αυτόν τον τρόπο να επεκτείνουμε χωρίς λόγο τον χρόνο που ένα αντικείμενο έχει δεσμεύσει μνήμη και κατά συνέπεια να μεγαλώνει η απαίτηση μνήμης της εφαρμογής μας.

Στην πραγματικότητα η ίδια η Apple στους οδηγούς χρήσης της, επισήμως αποθαρρύνει την χρήση autorelease αντικείμενων στον προγραμματισμό, καθώς στο iOS όταν μια εφαρμογή εκτελείται σε ένα τέτοιο σύστημα που διαθέτει περιορισμένη μνήμη, η χρήση τέτοιων αντικειμένων δεν πρέπει να γίνεται σε μεθόδους ή μέρη κώδικα οπού η εφαρμογή δημιουργεί πολλά αντικείμενα. Αντί για αυτό, πρέπει να απελευθερώνουμε τα αντικείμενα όσο μπορούμε πιο σύντομα.

Όταν θέλουμε να διαχειριστούμε την μνήμη μόνοι μας, χρησιμοποιώντας τον κύκλο διάθεση, διατήρηση και απελευθέρωση (alloc, retain και release cycle, δείτε εικόνα) δεν πρέπει να απελευθερώνουμε αντικείμενα που δεν έχουμε δημιουργήσει εμείς. Πρέπει επίσης να είμαστε σίγουροι ότι οι φόρες που έχουμε διατηρήσει ένα αντικείμενο είναι ίσες με τις φορές που το απελευθερώσαμε. Αντικείμενα που έχουμε εμείς δημιουργήσει, είναι αυτά που έχουμε χρησιμοποιήσει την λέξη alloc για να τους διαθέσουμε μνήμη ή την λέξη retain έτσι ώστε να τα διατηρήσουμε. Αντιθέτως αντικείμενα που έχουν δημιουργηθεί με την βοήθεια μεθόδων- κατασκευαστών όπως η stringWithString, δεν τα έχουμε δημιουργήσει εμείς, οπότε και δεν πρέπει να τα απελευθερώσουμε.

Όταν απελευθερώνουμε ένα αντικείμενο έχουμε την επιλογή να στείλουμε είτε ένα μήνυμα χρησιμοποιώντας την λέξη release είτε την λέξη autorelease:

[anObject release];

[anObject autorelease];

Όταν στείλουμε ένα μήνυμα με την λέξη release, τότε απελευθερώνουμε αμέσως την μνήμη που το αντικείμενο χρησιμοποιεί, εφόσον με αυτό το μήνυμα η αναφορά μέτρησης (Reference Count) του γίνεται μηδέν, ενώ άμα στείλουμε ένα μήνυμα με την λέξη autorelease τότε΄ το αντικείμενο προστίθεται στην autorelease pool. Έτσι το αντικείμενο απελευθερώνεται όταν απελευθερωθούν όλα τα αντικείμενα που την απαρτίζουν, κάτι που συνήθως συμβαίνει στο τέλος της κάθε κλάσης. Εάν το αντικείμενο είναι εκπρόσωπος (delegate) ενός άλλου αντικειμένου τότε, πριν απελευθερώσουμε το αρχικό αντικείμενο ,πρέπει να το θέσουμε ίσο με nil.

Η μέθοδος dealloc

Η μέθοδος dealloc καλείται όταν ένα αντικείμενο απελευθερώνεται. Δεν πρέπει ποτέ να καλούμε αυτήν την μέθοδο άμεσα αλλά αντί αυτού να στέλνουμε ένα μήνυμα με την λέξη release στο αντικείμενο καθώς σε διαφορετική περίπτωση το αντικείμενο μπορεί να περιέχει αναφορές σε άλλα αντικείμενα τα οποία δεν πρέπει να απελευθερωθούν (deallocated). Όπως κάναμε και με την κλάση HelloWorldViewController, πρέπει πάντα να προσθέτουμε στη μέθοδο dealloc τα αντικείμενα που εμείς δημιουργήσαμε:

```
- (void)dealloc {
      [label release];
      [button release];
      [super dealloc];
}
```

Σε αυτήν την μέθοδο απελευθερώνουμε το αντικείμενο label καθώς και το αντικείμενο button. Στη συνέχεια καλούμε την μέθοδο dealloc της υπέρ-κλάσης. Επιτρέπεται επίσης να στείλουμε ένα μήνυμα με την λέξη release σε ένα nil αντικείμενο.

Απαντώντας σε ειδοποιήσεις μνήμης

Ο κώδικάς μας πρέπει να μπορεί να απαντάει σε ειδοποιήσεις μνήμης. Ας δούμε ξανά λίγο το παράδειγμα μας στην εφαρμογή της κλάσης HelloWorldViewController. Περιέχει την μέθοδο didReceiveMemoryWarning:

```
- (void)didReceiveMemoryWarning {

      [super didReceiveMemoryWarning];

}
```

Εδώ είναι το κατάλληλο σημείο να απελευθερώσουμε μεγάλα μέρη μνήμης, όπως για παράδειγμα φωτογραφίες ή ότι μπορεί να έχουμε αποθηκεύσει από τον διαδίκτυο. Εάν αγνοήσουμε μια ειδοποίηση μνήμης τότε η εφαρμογή μας μπορεί να καταρρεύσει. Το iPhone δεν διαθέτει κάποιο είδος εικονικής μνήμης οπότε, όποτε η συσκευή δεν έχει χώρο στη μνήμη της, τότε δεν υπάρχει άλλη μνήμη που να μπορεί να διαθέσει το λειτουργικό. Είναι δυνατόν, να δοκιμάσουμε την εφαρμογή μας και να εξομοιώσουμε μια ειδοποίηση μνήμης στον iPhone Simulator.

11.4.3 Θεμελιώδη σχεδιαστικά πρότυπα

Όταν γράφουμε κώδικα, τις περισσότερες φορές χρησιμοποιούμε κάποια πρότυπα. Ένα σχεδιαστικό πρότυπο είναι μια εύκολα επαναχρησιμοποιήσιμα λύση, ένα σχέδιο, που σκοπό έχει να προσεγγίσει ένα σύνηθες πρόβλημα.

Ένα τέτοιο πρότυπο δεν είναι έτοιμος κώδικας, αλλά η περιγραφή του πώς πρέπει να μοντελοποιήσουμε την εφαρμογή μας σε σχέση με τις κλάσεις που χρησιμοποιούμε και πως αυτές πρέπει να κατασκευαστούν. Ακόμα περιγράφονται και οι συνδέσεις και τοι σχέσεις που πρέπει να έχουν μεταξύ τους.

Το πλαίσιο Cocoa Touch που υπάρχει στα κατώτερα στρώματα των εφαρμογών μας στο iOS, είναι βασισμένο σε ένα από τα πιο παλιά σχεδιαστικά πρότυπα. Το πρότυπο Model-View- Controller (MVC), το οποίο υπάρχει από το 1970. Το πρότυπο MVC χρησιμοποιείται για να διαχωρίζει την προγραμματιστική λογική από το

περιβάλλον χρήστη (User Interface, UI) και είναι ένας ευρέως διαδεδομένος τρόπος με τον οποίο μπορούμε να κατασκευάσουμε εφαρμογές για το iOS. Από την στιγμή που χρησιμοποιείται τόσο πολύ από τα πλαίσια (Framework) που μας προμηθεύει η Apple, και στα οποία περιλαμβάνεται και το UIKit Framework, είναι δύσκολο να γράψουμε μια εφαρμογή χωρίς να χρησιμοποιήσουμε αυτό το σχεδιαστικό πρότυπο στην υλοποίηση του κώδικά μας.

Το πρότυπο Model-View-Controller

Το πρότυπο MVC χωρίζει την εφαρμογή μας σε τρία λειτουργικά μέρη.

- Το Model (μοντέλο): Το model διαχειρίζεται την κατάσταση της εφαρμογής μας (και τα σχετιζόμενα δεδομένα) και συνήθως δουλεύει αδιάκοπα. Είναι εντελώς ανεξάρτητο απο το UI
- Το View (προβολή): Το view ουσιαστικά είναι αυτό που βλέπει ο χρήστης .Επιτρέπει στον χρήστη να αλληλεπιδρά με το μοντέλο και απαντά ή δημιουργεί γεγονότα. Σε εφαρμογές στο iOS το view κατασκευάζεται συνήθως στο Interface Builder και όχι προγραμματιστικά.
- Ο Controller (ρυθμιστής): Ο controller συντονίζει τις αλλαγές στο view σε σχέση με το model, όταν ο χρήστης αλληλεπιδρά με το view αυτός κάνει αλλαγές στο model και ανάποδα. Εδώ είναι συνήθως το μέρος οπού υπάρχει η μεγαλύτερη λογική της εφαρμογής. Στο παράδειγμά μας, την εφαρμογή HelloWorld, δημιουργήσαμε το view με το Interface Builder και η κλάση HelloWorldViewController στην ουσία είναι αυτή που θα διαχειριζόταν το view. Αυτή η εφαρμογή ήταν πολύ απλή και για αυτό το model της συμπεριλήφθηκε στη ίδια κλάση. Αν θέλαμε όμως να ακολουθήσουμε αυστηρά το σχεδιαστικό πρότυπο MVC τότε θα έπρεπε να είχαμε δημιουργήσει μια κλάση που θα περιελάμβανε την μέθοδο sayHello:.

Views και View Controllers

Έχουμε αναφερθεί και στα views αλλά και στα view controllers, Μπορούμε να δημιουργήσουμε ένα view προγραμματιστικά, όμως ο Interface Builder κάνει τα πράγματα αρκετά πιο εύκολα και είναι ο πιο απλός και εύχρηστος τρόπος να

δημιουργήσουμε ένα view. Όταν χρησιμοποιούμε τον Interface Builder στη ουσία δημιουργούμε ένα αρχείο με κώδικα XML. Με αυτό τον τρόπο η δημιουργία αντικειμένων και η δήλωση των σχέσεων μεταξύ τους, μας ελευθερώνει από μεγάλα μέρη κώδικα που αλλιώς θα χρειαζόταν να γράψουμε μονοί προκειμένου να διαχειριστούμε το view.

Τα Delegates και το πρότυπο DataSource

Τα delegates (εκπρόσωποι) είναι πρωτόκολλα που επιτρέπουν σε ένα αντικείμενο να δρα εκ μέρους ενός άλλου αντικειμένου. Για να δεχτούμε μια ειδοποίηση ενός γεγονότος στο οποίο πρέπει να απαντήσουμε, η κλάση delegate πρέπει να υλοποιεί μια μέθοδο notification η οποία να έχει δηλωθεί ως πρωτόκολλο delegate το οποίο να συνδέεται με αυτό το γεγονός. Το πρωτόκολλο μπορεί, και συνήθως το κάνει, να προσδιορίσει έναν αριθμό μεθόδων που η κλάση delegate πρέπει να περιλαμβάνει.

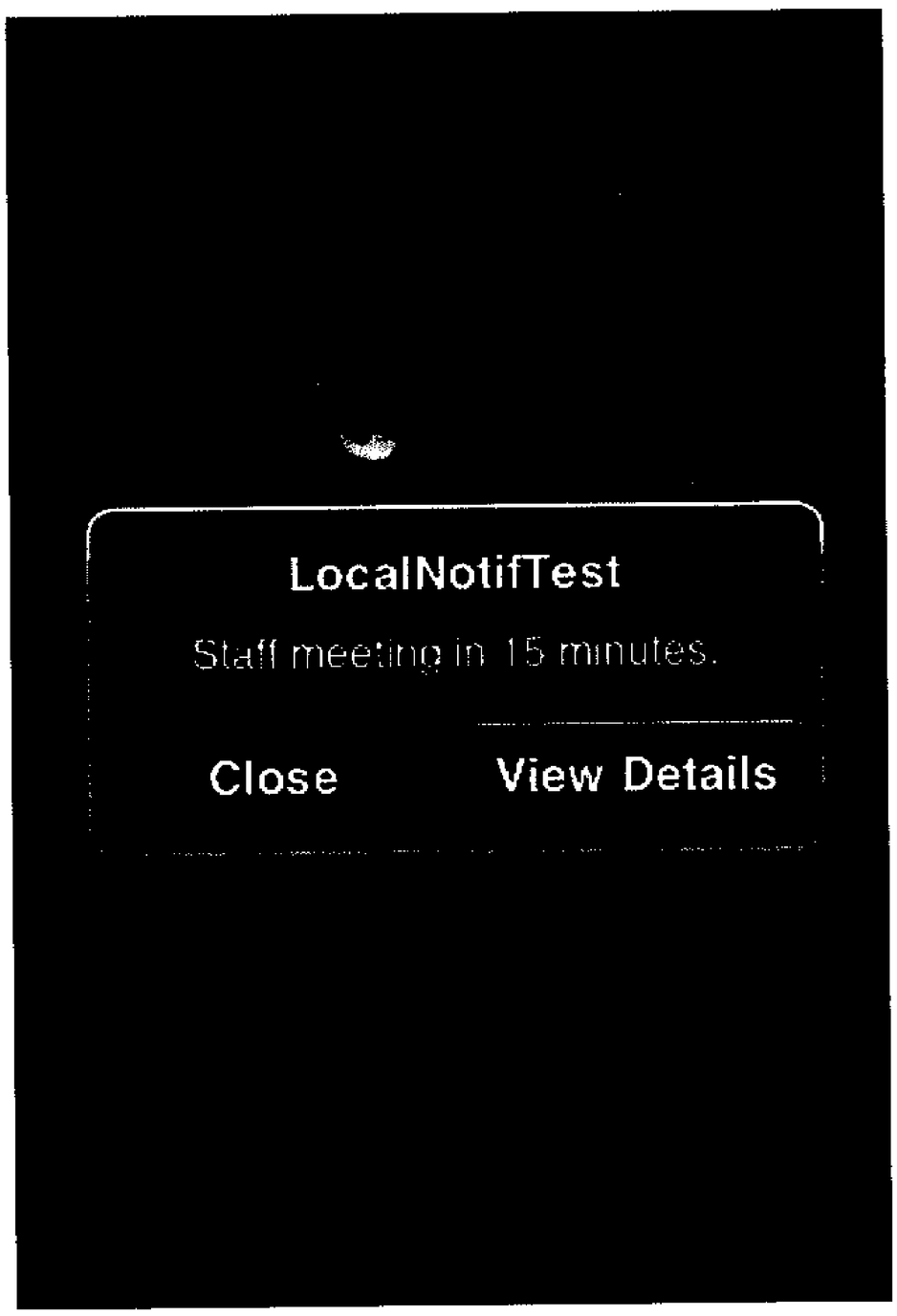
Συμπεράσματα

Σε αυτό το υποκεφάλαιο καλύψαμε ένα μεγάλο μέρος των βασικών λειτουργιών της γλώσσας καθώς και το πρότυπο MVC, το οποίο είναι ένα πολύ βασικό στοιχείο ώστε να καταλάβουμε πως μπορούμε να χρησιμοποιήσουμε την Objective-C, καθώς και τα δυνατά της σημεία.

Κεφάλαιο 12^{o}

Push Notifications

Η τεχνική Push notifications δίνει την δυνατότητα σε μια εφαρμογή που δεν τρέχει στο προσκήνιο, να ενημερώνει τον χρήστη ότι υπάρχει κάποια διαθέσιμη πληροφορία γι' αυτόν. Η πληροφορία αυτή μπορεί να είναι ένα μήνυμα κειμένου, ένα γεγονός στο ημερολόγιο της συσκευής ή νέα δεδομένα σε κάποιο remote server που «περιμένουν» . Η όψη του notification είναι ίδια σε όλες τις περιπτώσεις. Ο χρήστη ενημερώνεται με ένα alert message ή με ένα νούμερο πάνω στο εικονίδιο της εφαρμογής, που σηματοδοτεί οτι υπάρχουν τόσα notifications, όσα δείχνει το νούμερο.



Εικόνα 12.1: *Push Nortification - Alert Message*

Όταν ο χρήστης ειδοποιείται από την εφαρμογή ότι έχει ένα μήνυμα, ένα event, ή οποιαδήποτε άλλα δεδομένα διαθέσιμα, τότε μπορεί να ανοίξει το application και να δει τις λεπτομέρειες. Επίσης μπορεί να αγνοήσει την ειδοποίηση και στην περίπτωση η εφαρμογή δεν ενεργοποιείται.

12.1 Το πρόβλημα που επιλύει η τεχνολογία Push Notification

Μόνο ένα application μπορεί να βρίσκεται στο προσκήνιο κάθε φορά. Πολλές εφαρμογές λειτουργούν σε time-based ή interconnected περιβάλλον. Time based περιβάλλον είναι μια εφαρμογή όπως ένα calendar και interconnected είναι μια εφαρμογή που συνδέεται με κάποιο server που έχει τα δεδομένα απομακρυσμένα. Έτσι λοιπόν, οι εφαρμογές παράγουν ειδοποιήσεις όταν η εφαρμογή δεν βρίσκεται στο προσκήνιο. Τα push notifications επιτρέπουν στις εφαρμογές να ειδοποιούν τους χρήστες όταν συμβαίνει κάποιο γεγονός.

Όταν το λειτουργικό σύστημα της συσκευής λάβει ένα push notification και η εφαρμογή δεν τρέχει στο προσκήνιο, τότε εμφανίζεται η ειδοποίηση (κείμενο, εικονίδιο με το νούμερο των ειδοποιήσεων που εκκρεμούν, ήχος). Εάν υπάρχει push notification και ο χρήστης πατήσει το action button ή σύρει το action slider, τότε η εφαρμογή ανοίγει και καλεί μια μέθοδο στην οποία περνάει το notification payload. Εάν το application τρέχει στο προσκήνιο και έρθει ένα push notification, τότε το λαμβάνει ο application delegate της εφαρμογής.

Apple Push Notification Service είναι η πύλη για τα Push Notifications

Η υπηρεσία της apple, ονόματι Apple Push Notification service (APNs), μοιράζει push notifications σε συσκευές που έχουν εφαρμογές οι οποίες έχουν εγγραφεί προκειμένου να λαμβάνουν ειδοποιήσεις. Κάθε συσκευή εγκαθιδρύει μια διαπιστευμένη και κρυπτογραφημένη σύνδεση μέσω πρωτοκόλλου IP και λαμβάνει notifications μέσω αυτής της μόνιμης σύνδεσης. Οι Providers(developers της εφαρμογής), συνδέονται με την υπηρεσία της apple (APNs), μέσω μόνιμης και ασφαλούς σύνδεσης. Ο provider ετοιμάζει τα δεδομένα που θέλει να αποστείλει και

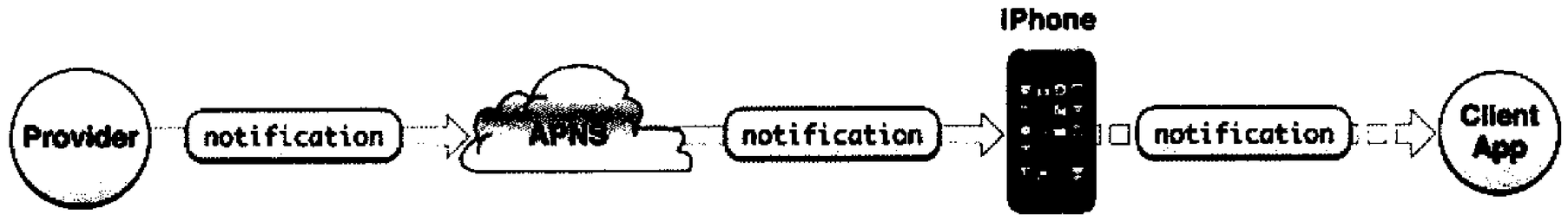
τα στέλνει στην υπηρεσία της apple(APNs) η οποία προωθεί τα δεδομένα στις συσκευές.

Η υπηρεσία της apple (APNs) εκτός του ότι είναι ασφαλής και υψηλής χωρητικότητας, περιλαμβάνει Quality-Of-Service συστατικά με δυνατότητα store-and-forward. Θα περιγραφεί αναλυτικά στην συνέχεια.

12.2 Τα Push Notifications και η διαδρομή

Η Apple Push Notification service μεταφέρει και δρομολογεί ένα notification από τον provider(developer) σε μια συγκεκριμένη συσκευή. Μια ειδοποίηση είναι στην ουσία ένα σύντομο μήνυμα που περιλαμβάνει δύο μέρη: το device token και το payload. Το device token είναι ανάλογο του τηλεφωνικού αριθμού και περιλαμβάνει πληροφορίες που δίνουν την δυνατότητα στην υπηρεσία της apple, να εντοπίσει την συσκευή στην οποία έχει εγκατασταθεί το application. Επίσης η υπηρεσία της apple χρησιμοποιεί το device token για να αυθεντικοποιήσει την δρομολόγηση του notification. Το payload είναι πληροφορίες σε JSON format που περιγράφουν πώς θα ειδοποιηθεί ο χρήστης (text, ήχος, icon badge).

Η πορεία του push notification είναι one-way *(Εικόνα 12.2)*. Ο provider, συνθέτει το πακέτο που περιλαμβάνει το device token για το client application και το payload. Ο provider στέλνει το notification στον υπηρεσία της apple και αυτή με την σειρά της, το αποστέλλει στην συσκευή.



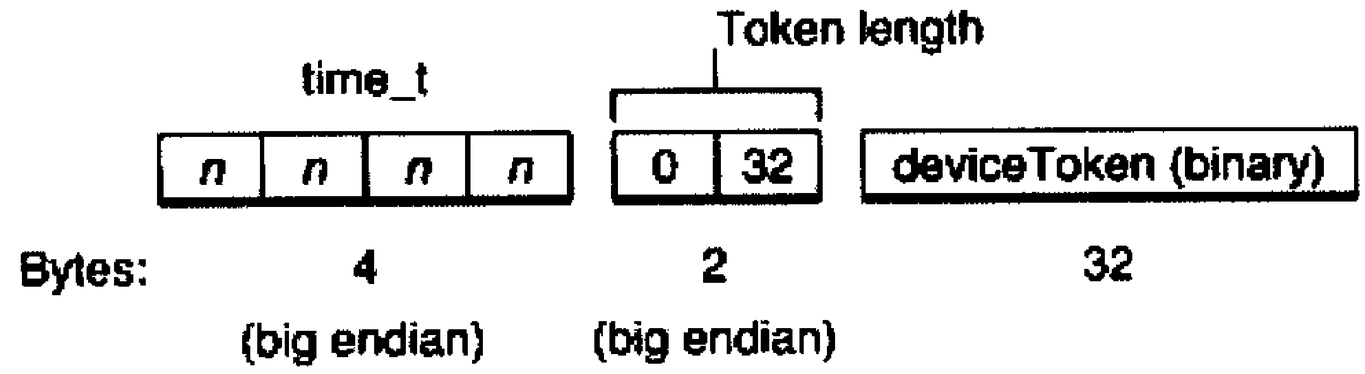
Εικόνα 12.2: *Διαδρομή Push Notification*

Η εικόνα 1 αποτελεί μια απλουστευμένη απεικόνιση του virtual APNs netowrk που καθιστά δυνατή την επικοινωνία μεταξύ provider και συσκευής

12.3 Feedback Service

Μερικές φορές η υπηρεσία της apple, ενδέχεται να προσπαθήσει να παραδώσει το notification σε μια συσκευή, αλλά η συσκευή μπορεί να αρνηθεί κατ' επανάληψη την παράδοση, επειδή δεν υπάρχει πλέον το application εγκατεστημένο. Αυτό συμβαίνει όταν ο χρήστης έχει απεγκαταστάσεις την εφαρμογή. Σε αυτή την περίπτωση, η υπηρεσία της apple ενημερώνει τον provider μέσω μιας κατάλληλης υπηρεσίας (feedback service). Η υπηρεσία αυτή συντηρεί μια λίστα με τις συσκευές για τις οποίες υπήρχαν πρόσφατες επανειλημμένες και αποτυχημένες προσπάθειες παράδοσης του notification. Ο provider οφείλει να λαμβάνει αυτή την λίστα και να μην στείλει άλλο notification στο μέλλον.

Η πρόσβαση στην υπηρεσία αυτή (feedback service) γίνεται μέσω ενός binary interface *(Εικόνα 12.3)*, παρόμοιο με αυτό που χρησιμοποιείται για την αποστολή των notifications. Η υπηρεσία βρίσκεται στην τοποθεσία feedback.push.apple.com και «ακούει» στην πόρτα 2196. Για να εγκαθιδρυθεί μια ασφαλής σύνδεση χρησιμοποιείται TLS(ή SSL) πρωτόκολλο. Μόλις γίνει η σύνδεση με την υπηρεσία, η αποστολή της λίστας που περιγράψαμε παραπάνω ξεκινά να αποστέλλεται στον provider, χωρίς να χρειαστεί επιπλέον εντολή. Ο provider ξεκινά να διαβάζει το stream που «γράφει» η υπηρεσία της apple, μέχρι το τέλος των δεδομένων. Τα δεδομένα έχουν την παρακάτω μορφή



Εικόνα 12.3: *Μορφή Feedback*

Timestamp	Το timestamp σηματοδοτεί πότε η υπηρεσία της Apple διαπίστωσε ότι το application δεν είναι εγκατεστημένο στην συσκευή
Token length	Το μέγεθος του device token
Device token	Το device token se binary format

12.4 Quality of Service - Security Architecture

Quality of Service

Η υπηρεσία των notifications περιλαμβάνει QoS components που υλοποιούν store-and-forward λειτουργικότητα. Εαν η υπηρεσία προσπαθήσει να στείλει ένα notification και η συσκευή είναι offine, τότε το notification αποθηκεύεται. Η υπηρεσία κράτα το τελευταίο notification για κάθε συσκευή. Όταν η συσκευή τελικά συνδεθεί, τότε το αποθηκευμένο μήνυμα προωθείται στην συσκευή. Το notification αποθηκεύεται για περιορισμένο χρονικό διάστημα πριν διαγραφεί.

Security Architecture

Για να πραγματοποιηθεί επικοινωνία μεταξύ του provider και της συσκευής, η υπηρεσία της apple χρειάζεται δυο διαφορετικά επίπεδα ασφαλείας. Αυτά είναι γνωστά ως connection trust και token trust.

Connection trust δηλώνει την βεβαιότητα ότι από την μια πλευρά, η υπηρεσία της apple έχει συνδεθεί με κάποιον διαπιστευμένο (authorized) provider με τον οποίο η apple έχει συμφωνήσει να παραδίδει notifications. Στην πλευρά της συσκευής, η υπηρεσία της apple, πρέπει να επικυρώνει ότι η σύνδεση είναι με μια νόμιμη συσκευή.

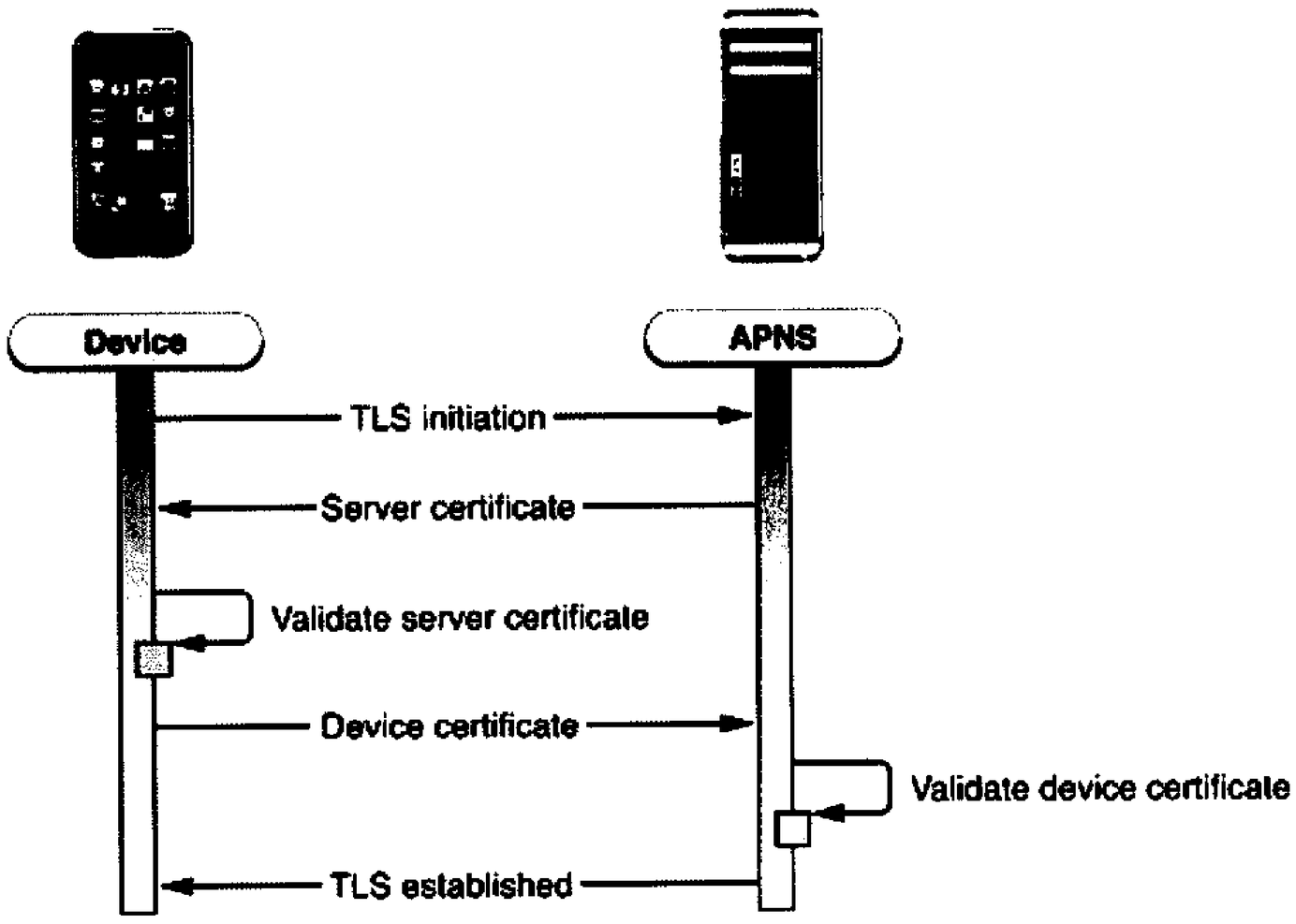
Αφού εγκαθιδρύσει trust connection, θα πρέπει στην συνέχεια να διασφαλίσει ότι μεταφέρει τα notification, μόνο σε αυτούς για τους οποίους προορίζονται. Για να γίνει αυτό, θα πρέπει να επικυρώσει την δρομολόγηση των μηνυμάτων που ταξιδεύουν. Μόνο η συσκευή που είναι ο τελικός παραλήπτης πρέπει να παραλάβει το μήνυμα.

Η διασφάλιση για την ακριβή δρομολόγηση(token trust) γίνεται εφικτή μέσω του device token. Το device token είναι ένα αδιαφανές αναγνωριστικό της συσκευής που η υπηρεσία της apple το δίνει στην συσκευή την πρώτη φορά που συνδέεται μαζί της.

Η συσκευή μοιράζει το device token στον provider. Στη συνέχεια, αυτό το token συνοδεύει κάθε notification από τον provider. Είναι η βάση για την διασφάλιση της εμπιστευτικότητας. Κατά μία έννοια, το device token, λειτουργεί όπως ο τηλεφωνικός αριθμός που προσδιορίζει τον προορισμό μιας τηλεφωνικής κλήσης).

12.5 Service-to-Device Connection Trust

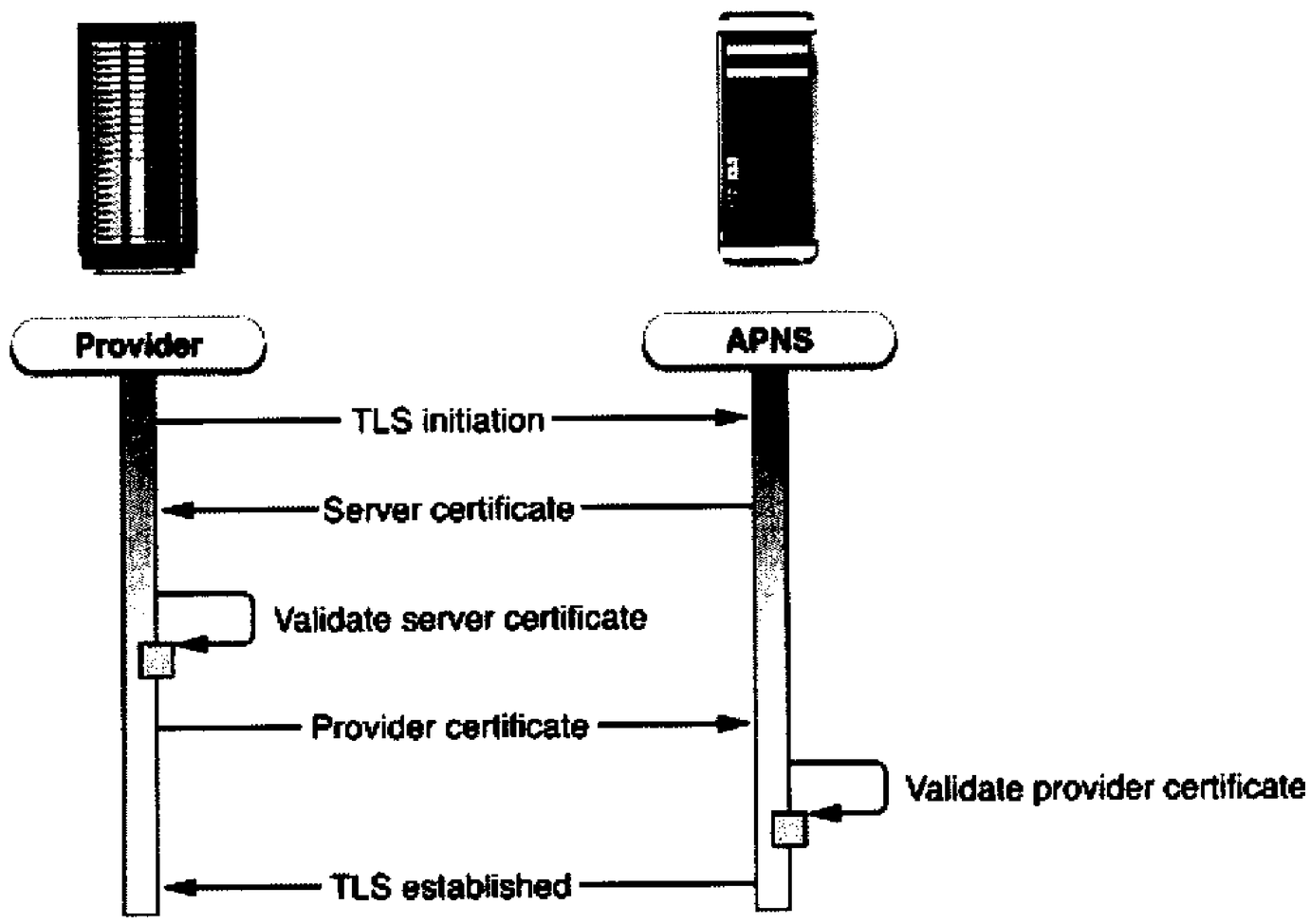
Η υπηρεσία της apple ελέγχει την ταυτότητα της σύνδεσης με την συσκευή μέσω TLS peer-to-peer authentication. (Το iOS αναλαμβάνει αυτή την διαδικασία του connection trust και όχι ο developer). Κατά την διάρκεια αυτής της διαδικασίας, μια συσκευή ξεκινά μια σύνδεση TLS με την υπηρεσία της apple (APNs) και η apple επιστρέφει το πιστοποιητικό του server. Η συσκευή υπογράφει το πιστοποιητικό που έλαβε και με την σειρά της, στέλνει στην apple το device certificate. Το device certificate το λαμβάνει ο server της apple και το επικυρώνει.



Εικόνα 12.4: Service-to-Device Connection

12.6 Provider-to-Service Connection Trust

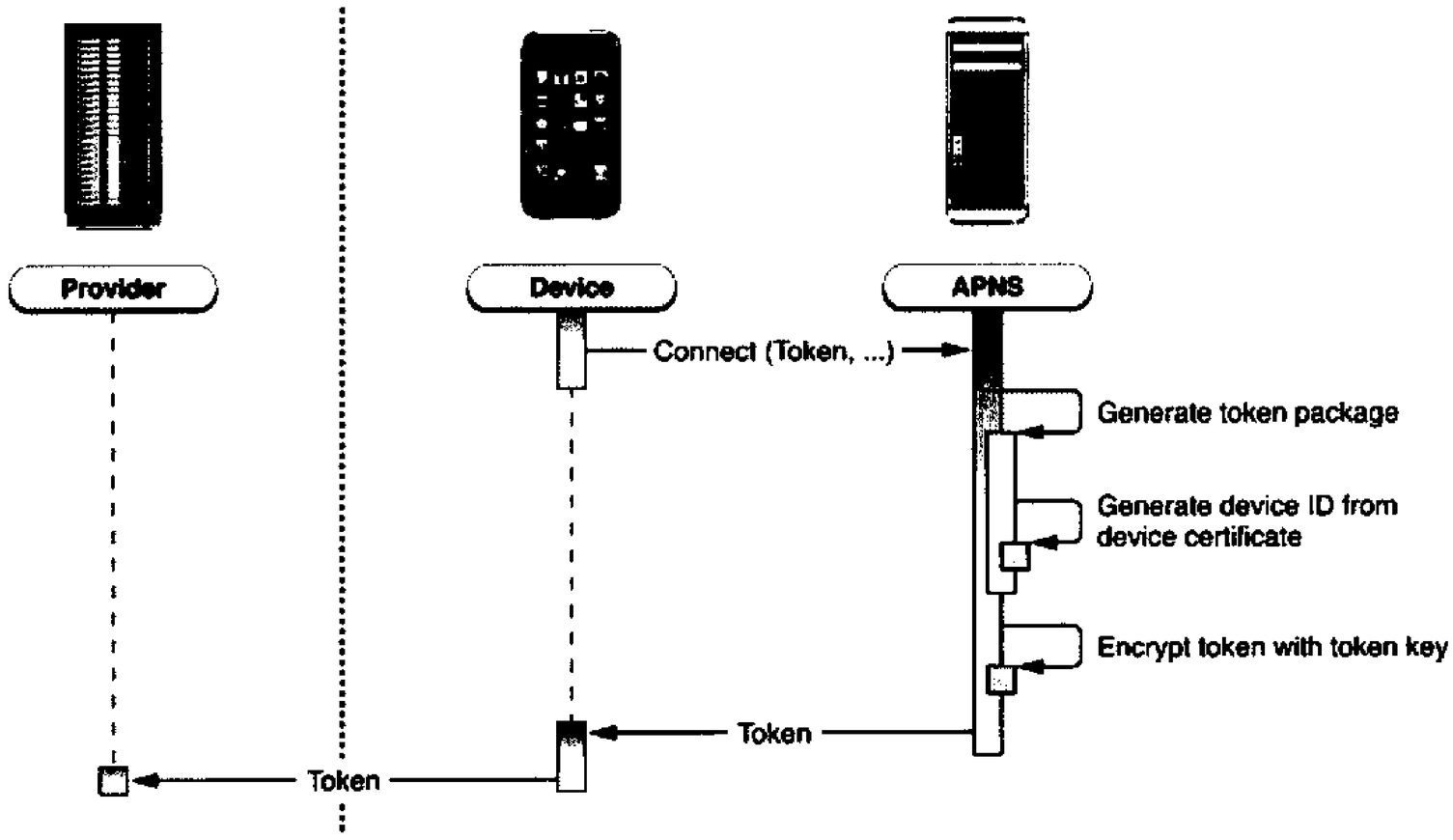
Η σύνδεση μεταξύ του provider και της υπηρεσίας της apple γίνεται και αυτή μέσω TLS connection. Η διαδικασία είναι παρόμοια με αυτή που περιγράφηκε παραπάνω. Ο provider ξεκινά μια TLS σύνδεση, λαμβάνει το server certificate από την υπηρεσία της apple και το επικυρώνει. Στην συνέχεια ο provider στέλνει στον server της apple το πιστοποιητικό του. Ο server το επικυρώνει και εγκαθιδρύεται μια ασφαλής σύνδεση(TLS).



Εικόνα 12.5: Provide-to-Service Connection

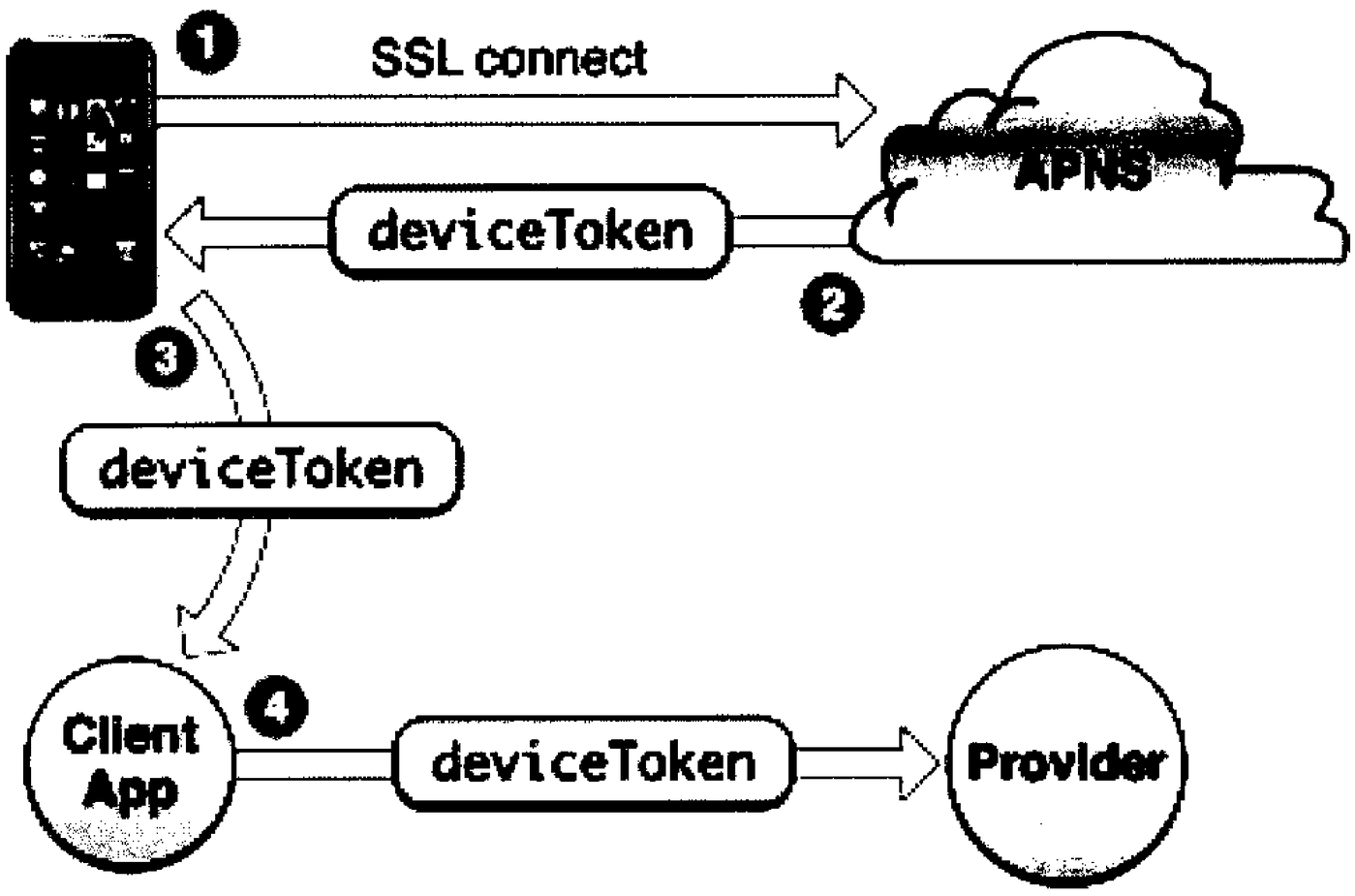
12.7 Δημιουργία device token και διαμοιρασμός του

Κάθε application θα πρέπει να εγγραφεί προκειμένου να λαμβάνει push notifications. Αυτό γίνεται συνήθως ακριβώς αφού η εφαρμογή, εγκατασταθεί στην συσκευή. Το λειτουργικό λαμβάνει το αίτημα για εγγραφή από το application, συνδέεται με την υπηρεσία της apple και προωθεί το αίτημα. Η apple παράγει το device token χρησιμοποιώντας πληροφορίες που περιέχονται στο device certificate. Το device token περιέχει ένα αναγνωριστικό της συσκευής.



Εικόνα 12.6: *Αποστολή Device Token*

Το device token κρυπτογραφείται χρησιμοποιώντας ένα token key και το κρυπτογραφημένο token επιστρέφεται στην συσκευή. Στην συνέχεια το device token επιστρέφει στην συσκευή και πιο συγκεκριμένα στο application που το ζήτησε. Το device token επιστρέφει σαν NSData object. Το application με την σειρά του, επιστρέφει το device token στον provider σε binary or hexadecimal format.

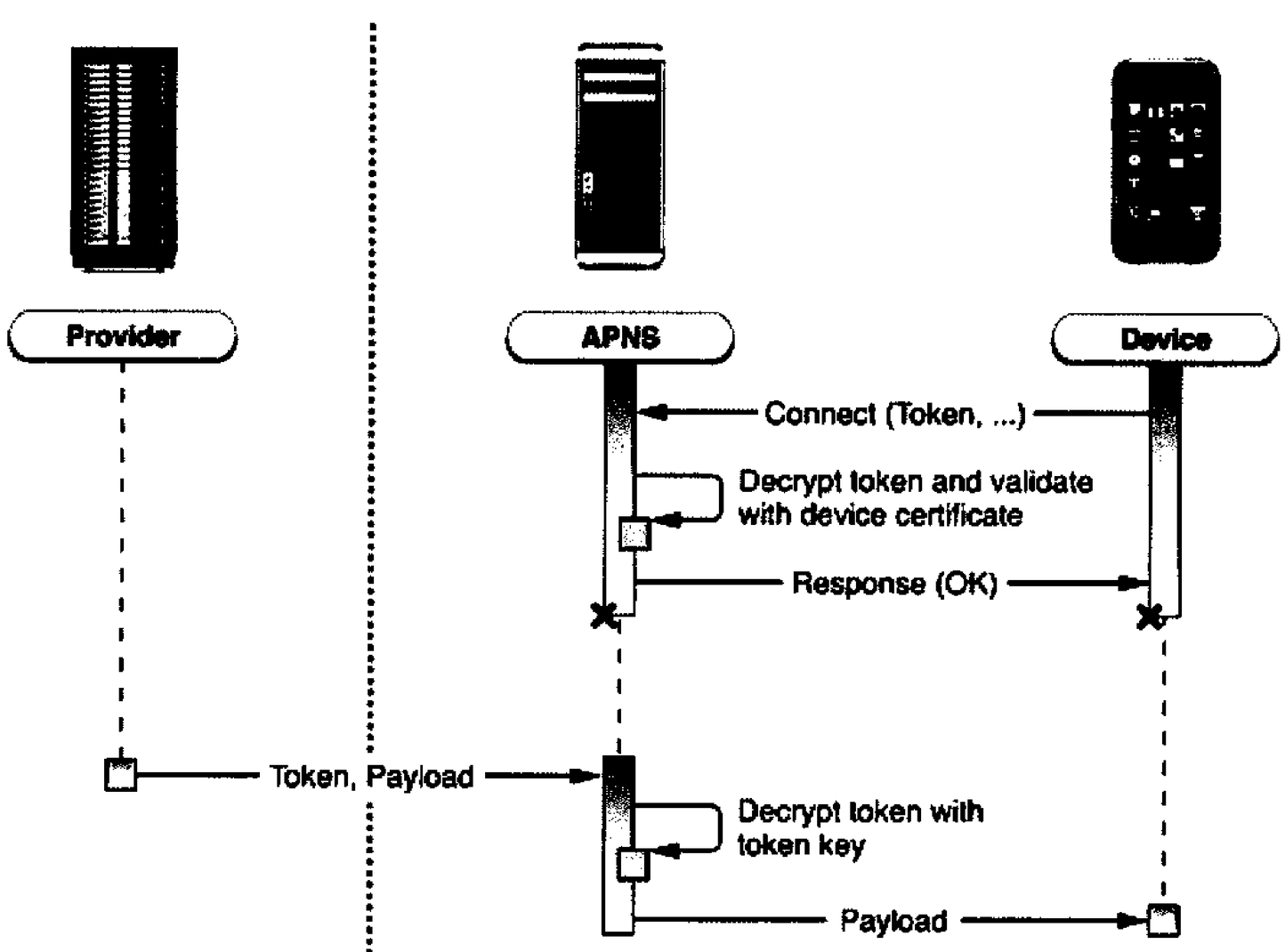


Εικόνα 12.7: *Διαμοιρασμός token*

12.8 Token Trust (Notification)

Αφού το λειτουργικό σύστημα παραλάβει το device token από την υπηρεσία της apple, όπως παρουσιάστηκε στην παραπάνω παράγραφο, ο provider πρέπει να στέλνει το συγκεκριμένο token κάθε φορά που συνδέεται με την υπηρεσία της apple. Ο server της apple αποκρυπτογραφεί το device token και πιστοποιεί ότι το token δημιουργήθηκε από την ίδια την συσκευή. Για να γίνει η πιστοποίηση, η υπηρεσία της apple βεβαιώνεται ότι το device identifier που περιέχεται στο token, ταιριάζει με το device identifier που υπάρχει στο device certificate.

Κάθε notification που στέλνει ο provider στον server της apple για να το προωθήσει στην συσκευή, πρέπει να συνοδεύεται με το device token που το application έχει δημιουργήσει. Ο server της apple αποκρυπτογραφεί το token χρησιμοποιώντας το token key, ώστε να επιβεβαίωση ότι το notification είναι έγκυρο. Στη συνέχεια χρησιμοποιεί το device id που είναι ενσωματωμένο στο device token προκειμένου να προσδιορίσει τον προορισμό του notification.



Εικόνα 12.8: Αποστολή Token στον Provider

Trust Componets

Για να υποστηριχτεί αυτό το μοντέλο ασφάλειας που θέτει η apple, ο provider και η συσκευή πρέπει να επεξεργαστούν πιστοποιητικά, certificate authority (CA) certificates και tokens.

- Provider: Κάθε provider χρειάζεται ένα unique provider certificate και ένα private key για την ασφαλή επικοινωνία με την υπηρεσία. Για κάθε notification, ο provider πρέπει να στείλει στην υπηρεσία της Apple και το device token το οποίο προσδιορίζει τον τελικό παραλήπτη, δηλαδή την συσκευή.
- Device: iOS(λειτουργικό σύστημα) χρησιμοποιεί ένα public server certificate που το λαμβάνει από την υπηρεσία της Apple, προκειμένου να αυθεντικοποιήσει την υπηρεσία στην οποία συνδέεται. Έχει ένα unique private key και ένα certificate που το χρησιμοποιεί για να αυθεντικοποιηθεί στην υπηρεσία και να εγκαθιδρύσει μια TLS σύνδεση. Η συσκευή παίρνει το device certificate και το private key κατά την διάρκεια του device activation και το αποθηκεύει. Το iOS αποθηκεύει επίσης το device token που το λαμβάνει κατά την διάρκεια της σύνδεσης. Κάθε εγγεγραμμένη εφαρμογή είναι υπεύθυνη για να ενημερώνει με το device token τον provider.

Η υπηρεσία της Apple έχει επίσης όλα τα απαραίτητα πιστοποιητικά, CA certificates και private-public keys, προκειμένου να πιστοποιεί τις συνδέσεις και τις ταυτότητες των providers και των συσκευών.

12.9 Notification payload

Κάθε notification περιέχει ένα κομμάτι που ονομάζεται payload. Το κομμάτι αυτό προσδιορίζει πώς θα ειδοποιηθεί ο χρήστης. Το μέγιστο μέγεθος που επιτρέπεται είναι 256bytes. Αν το payload είναι μεγαλύτερο από 256bytes, τότε η υπηρεσία της apple απορρίπτει το notification. Η παράδοση του notification είναι “best effort” και δεν είναι εγγυημένη.

Για κάθε notification, οι providers θα πρέπει να συνθέτουν ένα JSON dictionary object το οποίο να ακολουθεί το πρότυπο RFC 4627. Αυτό το dictionary, θα περιέχει ένα δεύτερο dictionary με key "aps". Το aps dictionary περιέχει ένα ή περισσότερα properties που προσδιορίζουν τις ακόλουθες ενέργειες:

- Το alert message που θα εμφανιστεί στον χρήστη,
- Το νούμερο του εικονιδίου που θα προστεθεί.,
- Τον ήχο που θα ηχήσει όταν εμφανιστεί το notification.

Εάν το application δεν τρέχει όταν έρθει το notification, τότε το alert message, ο ήχος, και το εικονίδιο με το νούμερο θα εμφανιστούν. Εάν το application τρέχει ήδη, το iOS παραδίδει το notification στον application delegate σας NSDictionary obect. Οι providers μπορούν να δημιουργούν και custom payload values που θα είναι εκτός του aps namespace. Οι custom τιμές πρέπει να κάνουν χρήση του JSON stracture και να έχουν primitive τύπο δεδομένων: dictionary(object), array, string, number και Boolean. Δεν θα πρέπει να περιέχονται πληροφορίες πελατών. Επειδή η παράδοση των notifications δεν είναι εγγυημένη, δεν θα πρέπει να στηριζόμαστε στην υπηρεσία αυτή για την αποστολή σημαντικών πληροφοριών. Επίσης δεν πρέπει να περιέχονται ευαίσθητα δεδομένα μέσα στο payload για λόγους ασφαλείας.

Ο παρακάτω πίνακας δείχνει τα keys και τα values για το aps dictionary

alert	string or dictionary	Εάν αυτό το property συμπεριληφθεί, τότε το iOS εμφανίζει το standard alert. Το κείμενο που εμφανίζεται έχει οριστεί σαν value στο property alert. Επίσης εμφανίζονται δύο buttons: Close και View. Εάν ο χρήστης πατήσει το View τότε το application ξεκινά
badge	number	Ο αριθμός που θα εμφανιστεί πάνω στο εικονίδιο της εφαρμογής και δείχνει πόσες ειδοποιήσεις έχουν φτάσει.
sound	string	Το όνομα αρχείου ήχου που υπάρχει μέσα στο application bundle. Αυτό το αρχείο μουσικής παίζει σαν ήχος ειδοποίησης όταν έρχεται κάποιο alert. Αν το αρχείο αυτό δεν βρεθεί ή οριστεί η τιμή default, τότε ο default ήχος ενεργοποιείται.

Παραδείγματα JSON Payload

Το παράδειγμα που ακολουθεί, ορίζει ένα alert message με τιμή "Message received from Bob", τοποθετεί στο εικονίδιο της εφαρμογής τον αριθμό ένα, και όταν το notification εμφανιστεί θα παίξει ο ήχος με όνομα αρχείου bingbong.aiff που είναι ήδη αποθηκευμένος στα αρχεία της εφαρμογής.

```
{
      "aps" : { "alert" : "Message received from Bob",
                  "badge" : 1,
                  "sound" : "bingbong.aiff"
       }
}
```

12.10 Επικοινωνία Provider με την υπηρεσία Apple Push Notification

Σε αυτή την ενότητα θα περιγράψουμε το τρόπο επικοινωνίας που χρησιμοποιεί ο provider προκειμένου να επικοινωνήσει με την υπηρεσία της apple, καθώς και τις λειτουργίες που πρέπει να υλοποιηθούν.

Γενικές απαιτήσεις

Ο provider επικοινωνεί με την υπηρεσία μέσω ενός binary interface. Αυτό το interface είναι high-speed, high-capacity και χρησιμοποιεί streaming TCP socket design σε συνδυασμό με binary πληροφορίες. Το binary interface είναι ασύγχρονο.

Το interface είναι διαθέσιμο στην τοποθεσία gateway.push.apple.com, στην πόρτα 2195. Οι γρήγορες επανειλημμένες συνδέσεις μπορεί να θεωρηθούν denial-of-services attacks και η υπηρεσία της κόβει.

Οι providers είναι υπεύθυνοι για τα ακόλουθα. Πρέπει να:

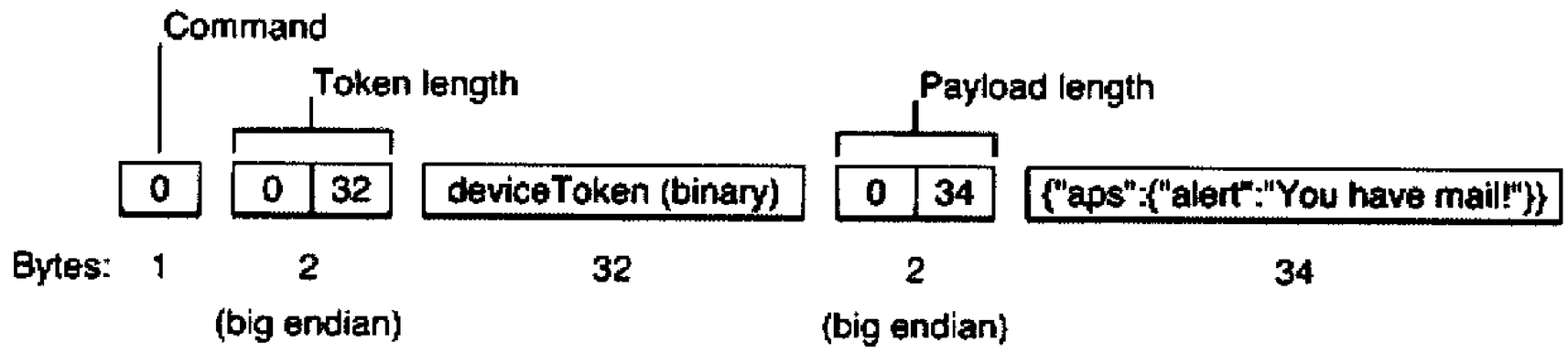
- συνθέτουν το notification payload,
- ορίζουν τον αριθμό που θα εμφανίζεται πάνω στο εικονίδιο της εφαρμογής,

- συνδέονται με τον feedback web server και να λαμβάνουν την τρέχουσα λίστα με τις συσκευές που έχουν επανειλημμένα δηλώσει failed-delivery.

Binary Interface και Notification Format

Το binary interface χρησιμοποιεί ένα plain TCP socket για binary περιεχόμενο.

Ακολουθεί το notification format



Το πρώτο byte αντιπροσωπεύει το command value και έχει τιμή μηδέν. Ακολουθεί το μέγεθος του device token. Τα επόμενα 32 Bytes κρατάνε το ίδιο το device token. Χρησιμοποιούνται 2 Bytes για το μέγεθος του payload και τέλος ακολουθεί το payload με μέγεθος που δεν μπορεί να ξεπεράσει τα 256 bytes. Στο παράδειγμα της εικόνας το payload έχει μέγεθος 34 Bytes.

Ακολουθεί κώδικας σε C που δείχνει την αποστολή notification με το format που περιγράφηκε παραπάνω.

```
static bool sendPayload(SSL *sslPtr, char *deviceTokenBinary,
                                        char *payloadBuff, size_t
payloadLength)
{
    bool rtn = false;
    if (sslPtr && deviceTokenBinary && payloadBuff && payloadLength)
    {
        uint8_t command = 0; /* command number */
        char binaryMessageBuff[sizeof(uint8_t) + sizeof(uint16_t) +
```

```
            DEVICE_BINARY_SIZE + sizeof(uint16_t) +
MAXPAYLOAD_SIZE];

        /* message format is,
|COMMAND|TOKENLEN|TOKEN|PAYLOADLEN|PAYLOAD| */

        char *binaryMessagePt = binaryMessageBuff;

        uint16_t networkOrderTokenLength =
htons(DEVICE_BINARY_SIZE);

        uint16_t networkOrderPayloadLength = htons(payloadLength);

         /* command */

        *binaryMessagePt++ = command;

        /* token length network order */

        memcpy(binaryMessagePt, &networkOrderTokenLength,
sizeof(uint16_t));

        binaryMessagePt += sizeof(uint16_t);

         /* device token */

        memcpy(binaryMessagePt, deviceTokenBinary,
DEVICE_BINARY_SIZE);

        binaryMessagePt += DEVICE_BINARY_SIZE;

         /* payload length network order */

        memcpy(binaryMessagePt, &networkOrderPayloadLength,
sizeof(uint16_t));

        binaryMessagePt += sizeof(uint16_t);

         /* payload */

        memcpy(binaryMessagePt, payloadBuff, payloadLength);

        binaryMessagePt += payloadLength;



        if (SSL_write(sslPtr, binaryMessageBuff, (binaryMessagePt -
binaryMessageBuff)) > 0)

            rtn = true;

    }

    return rtn;

}
```

Κεφάλαιο 13ο

Επικοινωνία με Server

Όλα τα δεδομένα της εφαρμογής είναι δυναμικά και μεταβάλλονται συχνά. Επίσης, η εφαρμογή χρησιμοποιεί κοινά δεδομένα με το web site. Όλες οι πληροφορίες βρίσκονται αποθηκευμένες σε κεντρικό σημείο και διαμοιράζονται στους clients. Οι τεχνολογίες που χρησιμοποιούνται στην πλευρά του server, είναι διαφορετικές από αυτές που χρησιμοποιούνται στην πλευρά της εφαρμογής-client. Επομένως οι πληροφορίες που αποστέλλονται πρέπει να είναι ανεξάρτητες από την πλατφόρμα και το σύστημα. Οι γενικευμένες γλώσσες έχουν τέτοια χαρακτηριστικά προσφέροντας ευέλικτους και ανοικτούς τρόπους αναπαράστασης πληροφοριών. Μια τέτοια γλώσσα είναι η XML.

13.1 Υλοποίηση επικοινωνίας με τον server

Κάθε φορά που η εφαρμογή- client ζητάει δεδομένα από τον server, τότε πραγματοποιεί ένα HTTP request προς τον server. Η απάντηση αυτής της αίτησης, περιλαμβάνει όλες τις πληροφορίες που είναι αναγκαίες, σε XML αναπαράσταση.

Δημιουργία HTTP request – κλήση web service

Για την δημιουργία του HTTP request, έγινε χρήση της κλάσης NSURLConnection. Η συγκεκριμένη native κλάση δίνει την δυνατότητα για την ασύγχρονη φόρτωση ενός URL request.

Ο NSURLConnection delegate περιλαμβάνει μεθόδους που επιτρέπουν σε κάποιο αντικείμενο, να λαμβάνει callbacks που σχετίζονται με την φόρτωση του αιτήματος. Οι μέθοδοι που ορίζονται στο interface του delegate είναι:

- Κανένα ή περισσότερα μηνύματα connection:willSendRequest:redirectResponse: θα σταλούν στον delegate εφόσον πρέπει να γίνει redirect σε νέα τοποθεσία.

- Κανένα ή περισσότερα μηνύματα connection:didReceiveAuthenticationChallenge: θα σταλούν στο delegate εάν είναι απαραίτητη η αυθεντικοποίηση του χρήστη προκειμένου να δεχθεί απάντηση και εφόσον η κλάση NSURLConnection δεν έχει authenticated credentials.
- Κανένα ή περισσότερα μηνύματα connection:didCancelAuthenticationChallenge: θα σταλούν στο delegate εάν η σύνδεση ακυρώσει την διαδικασία αυθεντικοποίησης λόγω σφάλματος.
- Κανένα η περισσότερα μηνύματα connection:didReceiveResponse: θα σταλούν στον delegate πριν λάβει μήνυμα connection:didReceiveData:
- Κανένα ή περισσότερα μηνύματα connection:didReceiveData: θα σταλούν στον delegate πριν από κάθε άλλο μήνυμα τύπου: connection:willCacheResponse:, connectionDidFinishLoading:, connection:didFailWithError:.

Το ακόλουθο παράδειγμα δείχνει την αρχικοποίηση της κλάσης NSURLConnection μέσω της οποίας γίνεται το HTTP request προς τον server.

```
NSMutableURLRequest *request = [NSMutableURLRequest
                                requestWithURL:[NSURL
                                URLWithString:

@"http://www.kratisinow.gr/data.php"]];
[request setHTTPMethod:@"POST"];
[[NSURLConnection alloc] initWithRequest:request delegate:self];
```

Απόκριση web service προς τον client

Εφόσον γίνει η κλήση του web service όπως περιγράφηκε παραπάνω, ο server ανακτά τις πληροφορίες από την βάση δεδομένων και απαντά σε XML αναπαράσταση.

Ακολουθεί μια XML αναπαράσταση των δεδομένων που στέλνει ο server προς την εφαρμογή.

```
<?xml version="1.0" encoding="UTF-8"?>
<stores>
	<store>
		<store_id>23</store_id>
		<store_name>Dream City - DC </store_name>
		<store_artist>Dj Alex</store_artist>
		<store_address>Ιερά Οδός 30</store_address>
		<store_student_price>110€/6</store_student_price>
		<store_normal_price>140€/6</store_normal_price>
		<store_img>dc.png</store_img>
		<store_latitude>37.979</store_latitude>
		<store_longtitude>23.714</store_longtitude>
	</store>
<stores>
```

Οι παραπάνω πληροφορίες περιλαμβάνουν tags με

- το id του καταστήματος,
- την ονομασία,
- τους καλλιτέχνες,
- την διεύθυνση,
- το κόστος,
- το όνομα αρχείου της φωτογραφίας που είναι αποθηκευμένη στον server,
- τις συντεταγμένες του καταστήματος.

Κεφάλαιο 14°

Υλοποίηση iPhone εφαρμογής

14.1 Ανάλυση εφαρμογής

Η εφαρμογή ακολουθεί την βασική δομή που ακολουθεί και η ιστοσελίδα. Το περιεχόμενο χωρίζεται σε 5 κατηγορίες. Για τις ανάγκες της εφαρμογής χρησιμοποιήθηκε η κλάση UITabBarController. Αυτή η κλάση δημιουργεί ένα tab bar interface που περιλαμβάνει tabs στο κάτω μέρος της οθόνης, προκειμένου ο χρήστης να πλοηγείτε στις διαφορετικές κατηγορίες. Τα tabs που ενσωματώθηκαν είναι:

- home
- stages
- clubs
- trips
- contact

Home tab

Εικόνα 14.1: *Home Tab*

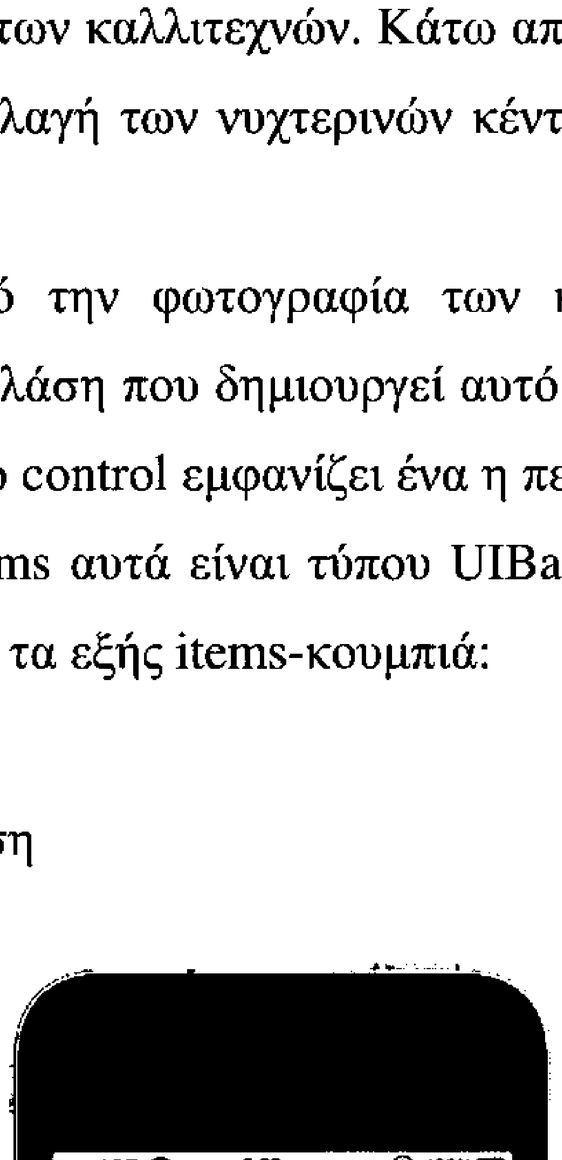
Είναι το πρώτο tab και εμφανίζεται με το που ανοίγει η εφαρμογή. Περιλαμβάνει ένα κείμενο καλωσορίσματος με κάποιες χρήσιμες πληροφορίες.

Stages και Clubs tabs

Σε αυτά τα tabs εμφανίζονται τα αντίστοιχα κέντρα διασκέδασης. Όλες οι πληροφορίες που εμφανίζονται είναι δυναμικές και λαμβάνονται από τον server με τον τρόπο που έχει ήδη περιγραφεί. Στο κεντρικό μέρος της οθόνης εμφανίζεται η φωτογραφία των καλλιτεχνών. Κάτω από την φωτογραφία αναγράφονται τα ονόματα τους. Η εναλλαγή των νυχτερινών κέντρων, γίνεται με slide gesture είτε δεξιά, είτε αριστερά.

Πάνω από την φωτογραφία των καλλιτεχνών, υπάρχει ένα toolbar με τρία κουμπιά. Η κλάση που δημιουργεί αυτό το control ονομάζεται UIToolbar. Γενικά, το συγκεκριμένο control εμφανίζει ένα η περισσότερα κουμπιά, που ονομάζονται toolbar items. Τα items αυτά είναι τύπου UIBarButtonItem. Στην συγκεκριμένη υλοποίηση, προστέθηκαν τα εξής items-κουμπιά:

- Info
- Κράτηση
- Share

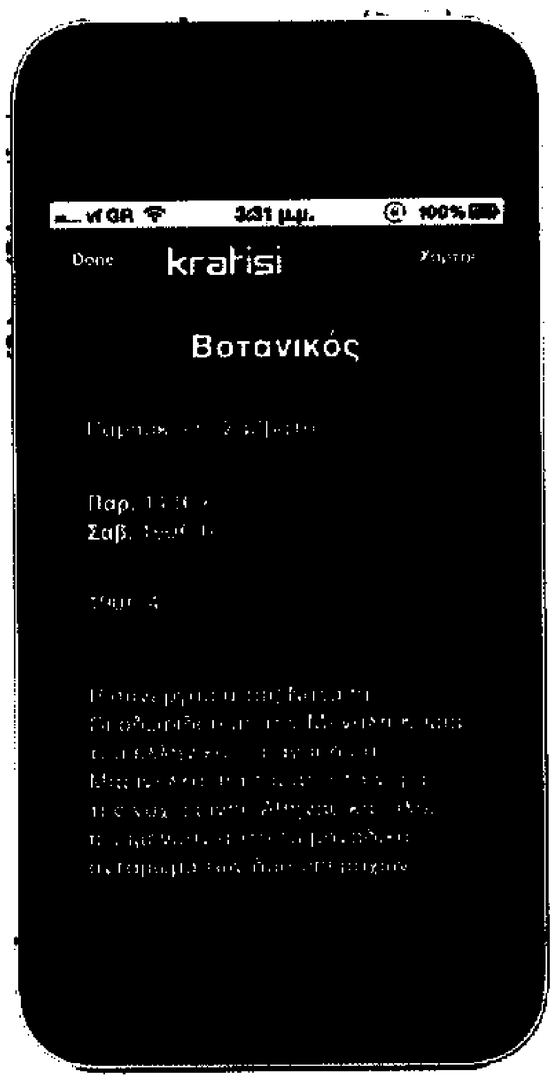


Slide finger

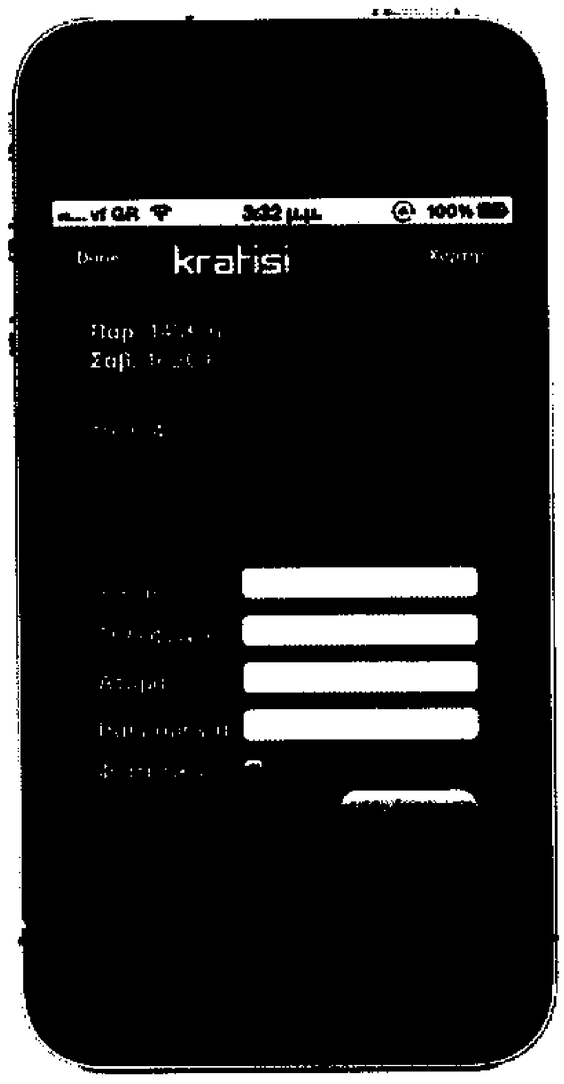
Εικόνα 14.2: *Stages-Clubs tabs*

Κουμπί Info

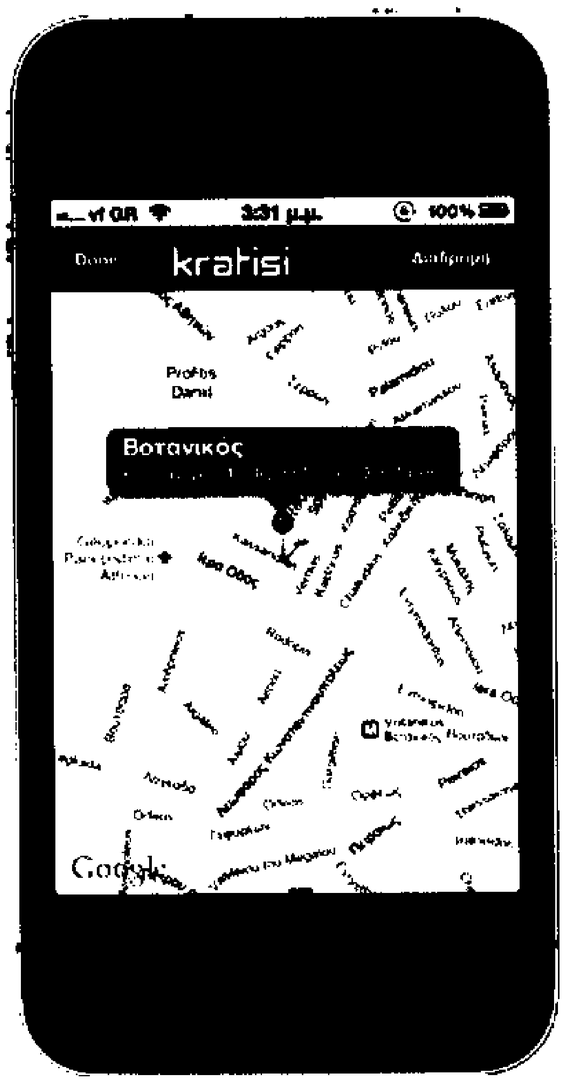
Το κουμπί Info περιλαμβάνει πληροφορίες για το συγκεκριμένο μαγαζί και τους καλλιτέχνες που το πλαισιώνουν *(Εικόνα 14.3)*. Το κουμπί κράτηση περιλαμβάνει πληροφορίες όπως τις ημέρες που λειτουργεί το συγκεκριμένο κέντρο διασκέδασης, το κόστος της κράτησης και τέλος περιλαμβάνει την φόρμα κράτησης που πρέπει να συμπληρώσει ο χρήστης *(Εικόνα 14.4)*. Τέλος υπάρχει η δυνατότητα προβολής του κέντρου ψυχαγωγίας πάνω σε χάρτη *(Εικόνα 14.5)*. Δίνεται η δυνατότητα εύρεσης διαδρομής από την τρέχουσα θέση του χρήστη, προς την τοποθεσία του κέντρου ψυχαγωγίας. Η λειτουργία αυτή παρέχεται μέσω της native εφαρμογής "Maps" που υπάρχει εγκατεστημένη στην συσκευή και χρησιμοποιεί Google Maps.



Εικόνα 14.3: *Πληροφορίες καταστήματος*



Εικόνα 14.4: *Φόρμα Κράτησης*



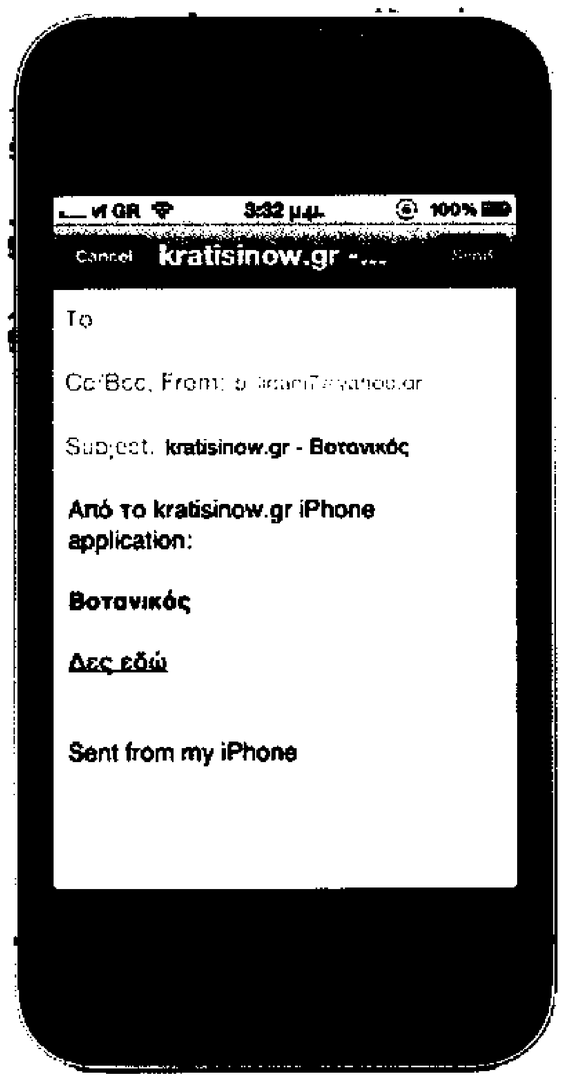
Εικόνα 14.5: *Διεύθυνση καταστήματος*

Κουμπί share

Το τρίτο και τελευταίο κουμπί είναι το «Share». Ο χρήστης έχει την δυνατότητα να μοιράζει το περιεχόμενο της εφαρμογής με τους φίλους του, επιλέγοντας ανάμεσα σε δύο τρόπους κοινοποίησης. Ο πρώτος είναι μέσω αποστολής email ενώ ο δεύτερος τρόπος κάνει χρήση του facebook API και δημοσιεύει στον τοίχο του χρήστη τις πληροφορίες. Τα δεδομένα που διαμοιράζονται και στις δυο περιπτώσεις έχουν να κάνουν με πληροφορίες που αφορούν συγκεκριμένα κέντρα διασκέδασης.

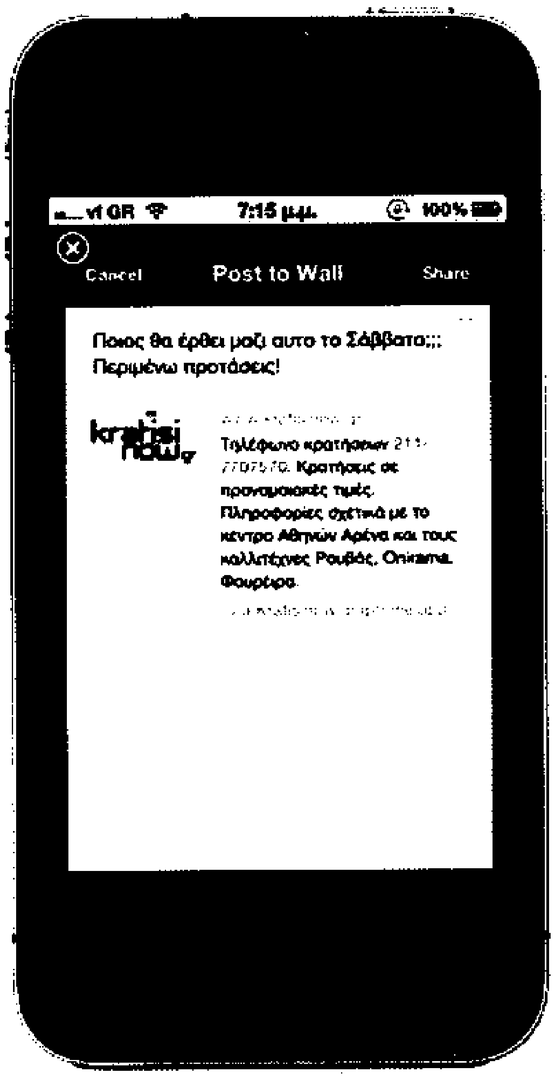
Εικόνα 14.6: *Share Button*

Όταν ο χρήστης επιλέξει την κοινοποίηση μέσω email, τότε ανοίγει ο προκαθορισμένος mail client της συσκευής με συμπληρωμένο το θέμα και και το περιεχόμενο του email που πρόκειται να σταλεί. Το σώμα του email περιλαμβάνει τον σύνδεσμο (link) του συγκεκριμένου κέντρου και παραπέμπει στο site. Ο χρήστης μπορεί να προσθέσει κείμενο και να επιλέξει τους παραλήπτες.

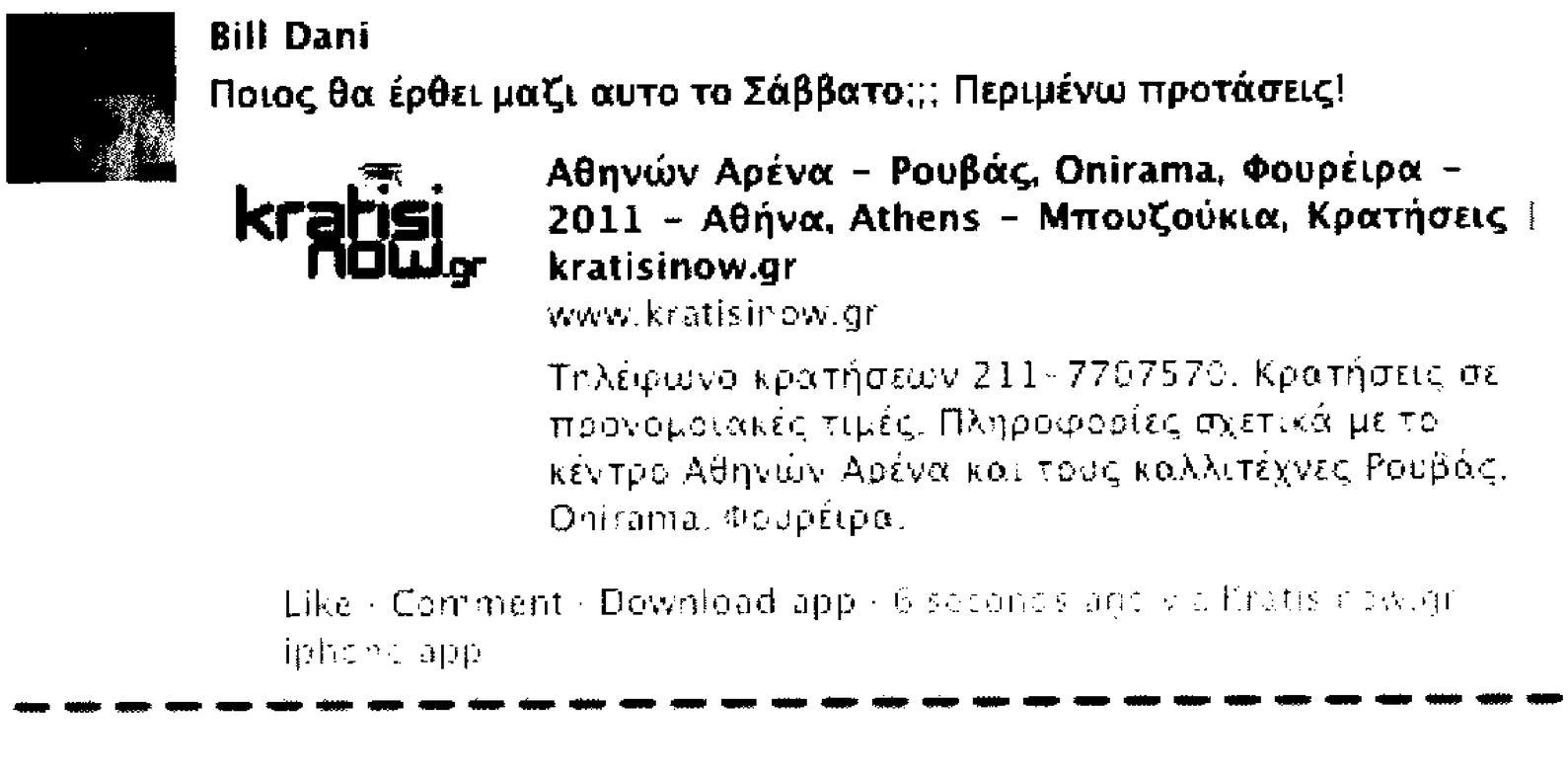


Εικόνα 14.7: *Αποστολή e-mail*

Όταν ο χρήστης επιλέξει την κοινοποίηση μέσω facebook, τότε το πρώτο βήμα σε αυτή την διαδικασία, είναι η αυθεντικοποίηση του χρήστη από το facebook. Σε αυτό το σημείο η διαδικασία γίνεται πολύ απλή για τον χρήστη, επειδή χρησιμοποιείται η τεχνική single-sign-on. Η τεχνική αυτή δίνει την δυνατότητα σε εφαρμογές που χρησιμοποιούν το facebook api, να αυθεντικοποιούν τους χρήστες χωρίς να χρειάζεται εκείνοι να δίνουν τα στοιχεία τους. Αυτό γίνεται εφόσον οι χρήστες είναι ήδη συνδεδεμένοι στο facebook μέσω της επίσημης εφαρμογής. Έτσι λοιπόν η εφαρμογή kratisinow.gr αλληλεπιδρά με την επίσημη εφαρμογή του facebook κατά την διάρκεια της αυθεντικοποίησης. Σε περίπτωση που δεν υπάρχει εγκατεστημένη η εφαρμογή του facebook στην συσκευή, τότε ο χρήστης δίνει τα στοιχεία του σε ένα pop-up dialog. Όταν ο χρήστης κάνει log-in είτε με τον αυτόματο, είτε με τον χειροκίνητο τρόπο, τότε εμφανίζεται ένα pop-up dialog στο οποίο ο χρήστης μπορεί να εισάγει το κείμενο που θα κοινοποιηθεί στην σελίδα του στο facebook *(Εικόνα 14.8)*. Εκτός από αυτό το κείμενο, θα κοινοποιηθεί το λογότυπο της εφαρμογής, ο σύνδεσμος του συγκεκριμένου κέντρου που παραπέμπει στην σελίδα του κέντρου στο διαδίκτυο, ένα προκαθορισμένο κείμενο με σύντομες πληροφορίες επικοινωνίας με το kratisinow.gr και τέλος ένας σύνδεσμος προς το app store για απευθείας κατέβασμα της εφαρμογής. Το αποτέλεσμα όλης της διαδικασίας εμφανίζεται στην εικόνα 14.9.



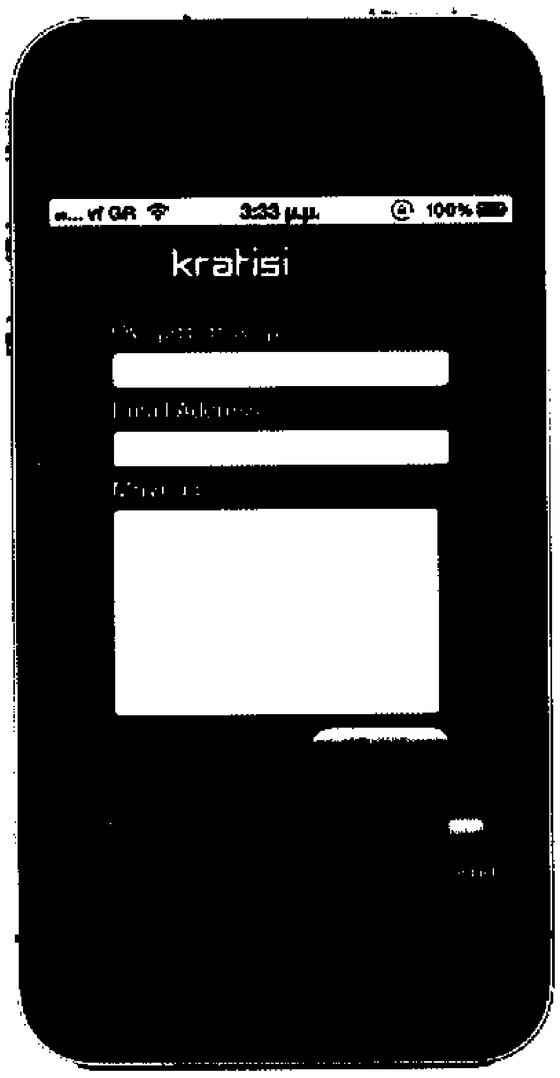
Εικόνα 14.8: *Facebook wall post*



Εικόνα 14.9: *Post στο Facebook Profile*

Contact tab

Το τελευταίο tab είναι η επικοινωνία. Εκεί υπάρχει μια φόρμα με τρία υποχρεωτικά πεδία. Πριν σταλεί το μήνυμα πραγματοποιείται το validation των εισόδων. Εφόσον ο έλεγχος βρει αποδεκτές τιμές, τότε το μήνυμα αποστέλλεται μέσω email στους διαχειριστές του kratisnow.



Εικόνα 14.10: *Contact tab*

Κεφάλαιο 15^{o}

Επικοινωνία Διακομιστή με Android Εφαρμογή

15.1 Επικοινωνία Server - Android Client

Όπως αναφερθήκαμε παραπάνω για την επικοινωνία μεταξύ server και κινητών συσκευών Android έγινε χρήση του JSON-RPC 2.0 πρωτοκόλλου, το οποίο είναι υπεύθυνο για τη μεταφορά δεδομένων.

Για την εξυπηρέτηση των κλήσεων που δέχεται ο server κατασκευάστηκε μια κλάση με το όνομα *json.class.php*, η οποία είναι υπεύθυνη για την επεξεργασία των HTTP κλήσεων που δέχεται ο server αλλά και την αποστολή των αντίστοιχων δεδομένων. Για την εκτέλεση των μεθόδων και την αναζήτηση των αντίστοιχων αποτελεσμάτων στη βάση δεδομένων δημιουργήθηκε η κλάση *kratisinow.class.php*, η οποία περιέχει όλες εκείνες τις μεθόδους που μπορεί να καλεστούν από τον client, και η κλάση *db.class.php*, η οποία διαχειρίζεται τη σύνδεση στη βάση δεδομένων και είναι η υπεύθυνη για την εκτέλεση των SQL ερωτημάτων.

Πιο αναλυτικά, το πρωτόκολλο HTTP υποστηρίζει ποικίλους μεθόδους επικοινωνίας όπως GET, HEAD, POST, PUT κλπ. Η μέθοδος επικοινωνίας που χρησιμοποιήθηκε για να εξυπηρετηθεί ένα αίτημα από το server είναι η μέθοδος POST. Επίσης, για την επιτυχή επεξεργασία του αιτήματος το περιεχόμενο(content) που αποστέλλεται πρέπει να έχει την ακόλουθη μορφή JSON:

{"id":2,"jsonrpc":"2.0","method":"get_stores","params":[1]}

- Το πεδίο «*id*» είναι χαρακτηριστικό για κάθε κλήση και είναι αυτό που συνδέει τα αιτήματα από client με τις απαντήσεις από τον server,
- Το πεδίο «*jsonrpc*» γνωστοποίει την έκδοση του πρωτοκόλλου που χρησιμοποιείται,
- Το πεδίο «*method*» περιγράφει την απομακρυσμένη μέθοδο, που θα εξυπηρετηθεί από την κλάση *kratisinow.class.php,*

- Το πεδίο *«params»* περιγράφει ένα πίνακα με RPC ορίσματα. Η σειρά των ορισμάτων επιρεάζει την αντίστοιχη σειρά στον απομακρυσμένο server.

Κατά την διαδικασία της αίτησης από τον client στο server, ο client χρησιμοποιεί ένα URL το οποίο δηλώνει που θα γίνει η κλήση και έχει την παρακάτω μορφή:

http://127.0.0.1:8080/Server/json/class_name

Πιο συγκεκριμένα,

- Το *http://127.0.0.1:8080/* είναι η διεύθυνση του server
- Το *Server/json/* είναι το directory του server που εξυπηρετεί τα αιτήματα που χρησιμοποιούν το πρωτόκολλο JSON-RPC 2.0.
- Το *class_name* περιγράφει το όνομα της κλάσης που περιέχει μέσα τις μεθόδους που χρησιμοποιούνται από τον client.

Αφού πραγματοποιηθεί το αίτημα, το οποίο πρέπει να ακολουθεί τις παραπάνω προδιαγραφές που αναλύθηκαν, η μορφή του αποτελέσματος που επιστρέφεται είναι JSON και περιγράφεται ως έξης:

{"jsonrpc":"2.0","result":[{"store_id":"23"}],"id":2},

- Το πεδίο «jsonrpc» γνωστοποίει την έκδοση του πρωτοκόλλου που χρησιμοποιείται,
- Η μεταβλητή «result» περιέχει το αποτέλεσμα της μεθόδου,
- Το πεδίο «id» είναι χαρακτηριστικό για κάθε κλήση και είναι αυτό που συνδέει τα αιτήματα από τον client με τις απαντήσεις από τον server.

Σε περίπτωση λάθους ορίσματος, το αποτέλεσμα που επιστρέφεται έχει την παρακάτω μορφή:

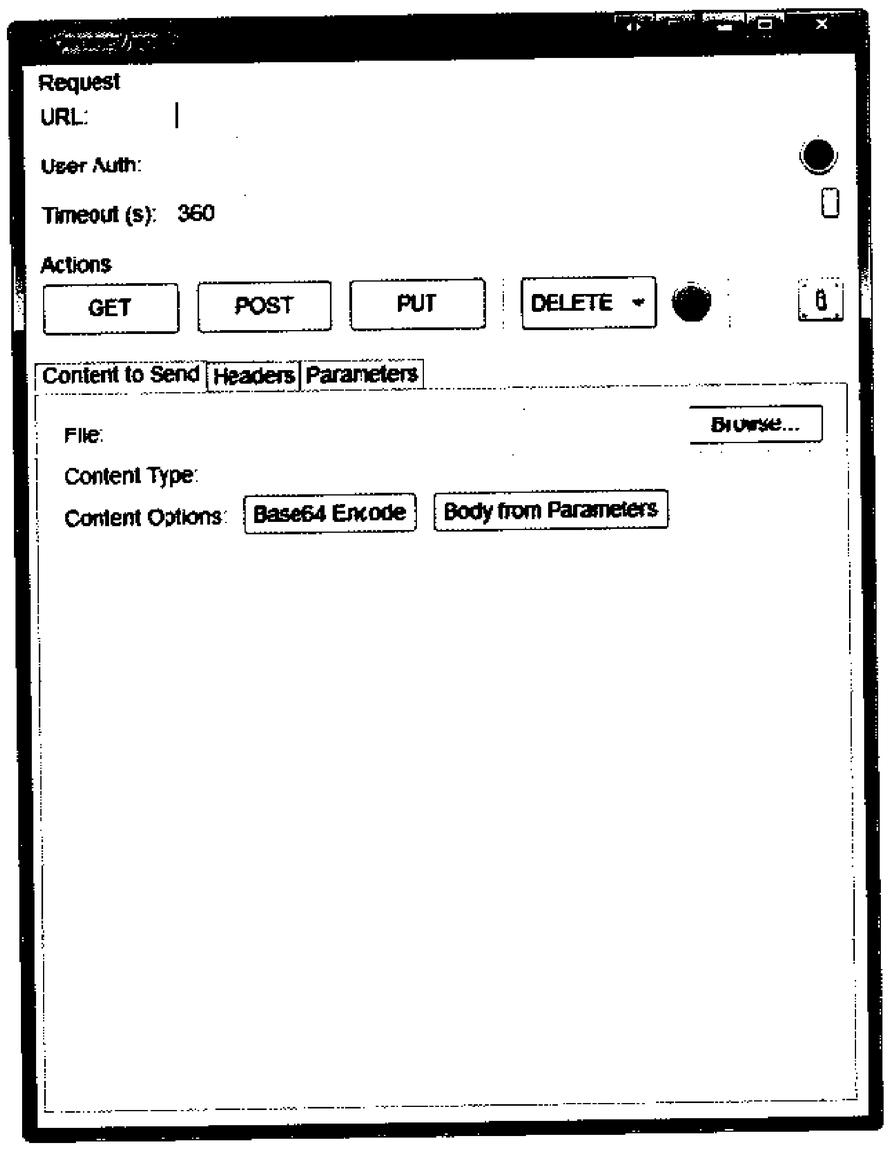
{"jsonrpc":"2.0","error":{"code":-32601,"message":"Method not found.", "data":null
}, "id":null}

Όπου το πεδίο «*error*» περιγράφει το σφάλμα που προέκυψε κατά την επικοινωνία.

15.2 Περιγραφή Αιτήματος JSON-RPC με τη χρήση του POSTER

Για την καλύτερη κατανόηση, θα περιγράψουμε την παραπάνω διαδικασίας χρησιμοποιώντας το POSTER, addon του διακομιστή Firefox. Το POSTER δίνει τη δυνατότητα να ορίσουμε

- ένα URL, το οποίο δηλώνει το URL του server που θα μας εξυπηρετήσει,
- την μέθοδο των HTTP κλήσεις που θα χρησιμοποιήσουμε,
- και το περιεχόμενο που θα στείλουμε προς επεξεργασία.

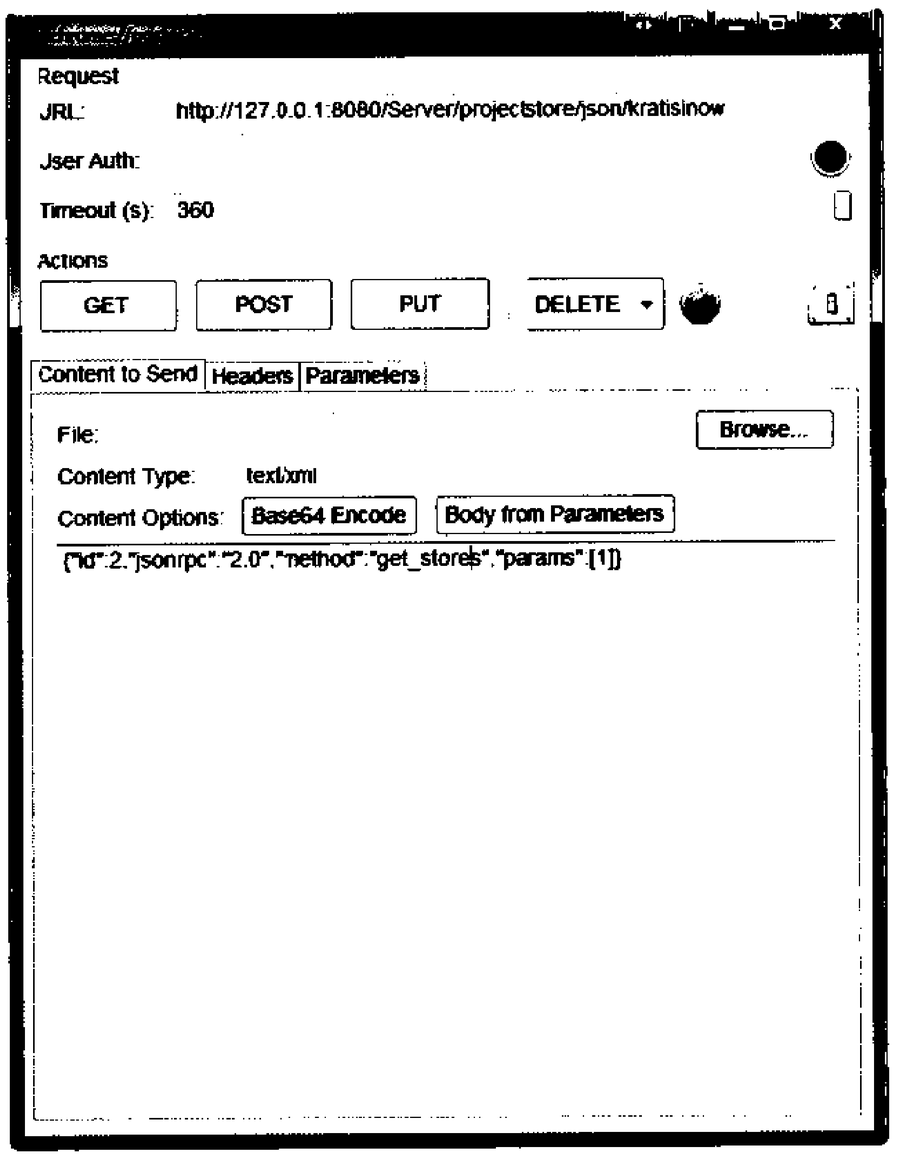


Εικόνα 15.1: *Firefox Addon - Poster*

Αρχικοποιώντας λοιπόν τις μεταβλητές μας, όπως φαίνεται στην παρακάτω εικόνα, ορίζουμε ως:

URL: http://127.0.0.1:8080/Server/projectstore/json/kratisinow

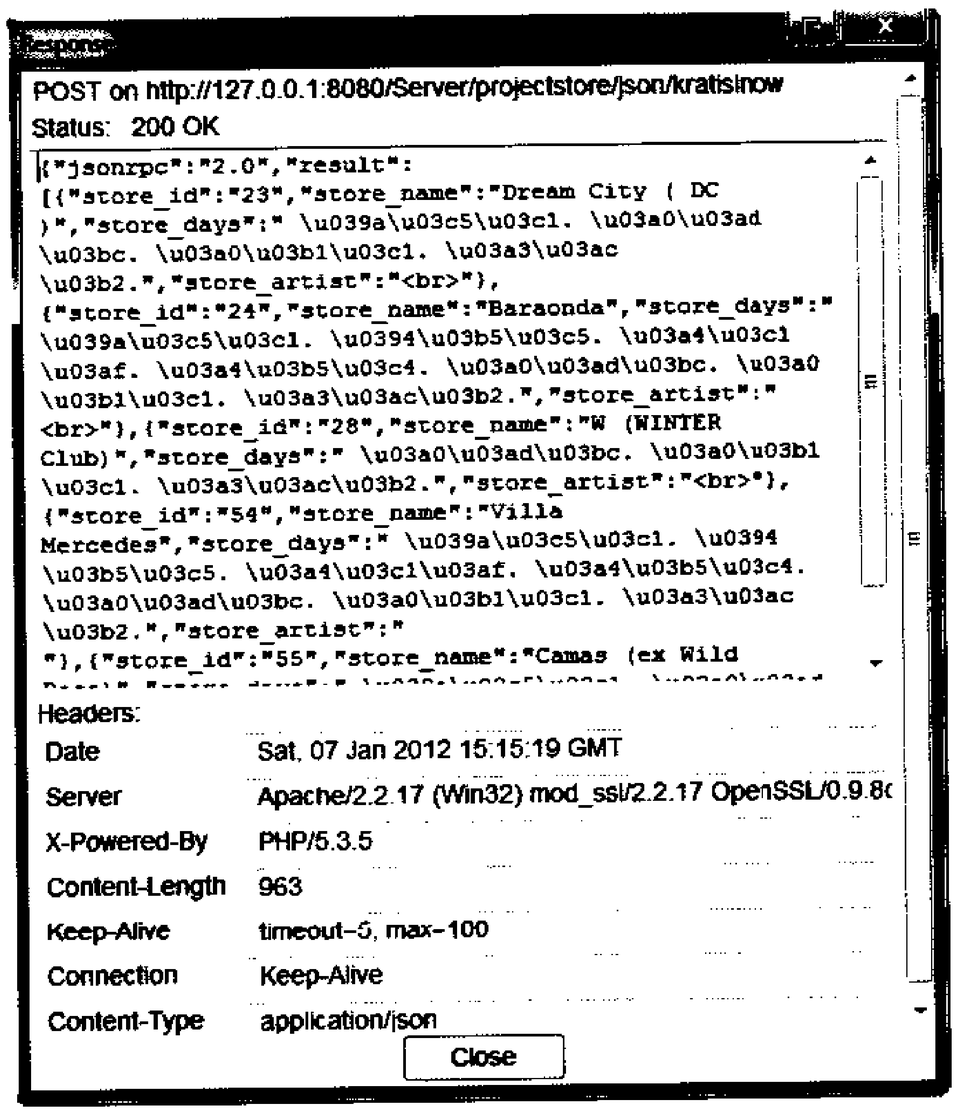
Content: {"id":2,"jsonrpc":"2.0","method":"get_stores","params":[1]}



Εικόνα 15.2: *Αρχικοποίηση μεταβλητών*

Στο URL αναφέρουμε το όνομα της κλάσης που θέλουμε να χρησιμοποιήσουμε, στην προκειμένη περίπτωση *kratisinow*, και στο Content ζητάμε να εκτελεστεί η μέθοδο *get_stores* με παράμετρο 1, δηλαδή να μας επιστρέψει όλα τα καταστήματα της κατηγορίας με id=1.

Στη συνέχεια πατάμε το κουμπί POST. Το αποτέλεσμα της διαδικασίας, αφού επεξεργαστεί, μας επιστρέφεται σε μορφή JSON και περιέχει όλες τις πληροφορίες για τις οποίες είναι υπεύθυνη η μέθοδο *get_stores* να επιστρέφει. Στην παρακάτω εικόνα εμφανίζεται το αποτέλεσμα του αιτήματος το οποίο είναι έτοιμο προς επεξεργασία από τον client και την εμφάνιση του στις κινητές συσκευές android.



Εικόνα 15.3: Αποτελέσματα Κλήσης

15.3 Μέθοδοι εξυπηρέτησης αιτημάτων

Για την εξυπηρέτηση των απαιτήσεων της εφαρμογής για τις κινητές συσκευές android υλοποιήθηκαν οι μέθοδοι που περιγράφονται παρακάτω. Η κάθε μέθοδος εξυπηρετεί διαφορετικά αιτήματα και περιέχονται στην κλάση *kratisinow.class.php*.

15.3.1 Μέθοδος «get_stores»

Η μέθοδος *get_store* επιστρέφει τα διαθέσιμα καταστήματα ανά κατηγορία. Οι πληροφορίες που ανακτώνται από το κάλεσμά της είναι το *id* του κάθε καταστήματος, το όνομα του κάθε καταστήματος, οι ημέρες που είναι διαθέσιμο το κατάστημα και οι καλλιτέχνες που εμφανίζονται στο κατάστημα αυτό.

15.3.2 Μέθοδος «specific_store»

Η μέθοδος specific_store, σε συνδυασμό με το id του καταστήματος που στέλνεται στον server, επιστρέφει στην Android εφαρμογή πληροφορίες για το συγκεκριμένο κατάστημα.

15.3.3 Μέθοδος «get_notifications»

Η μέθοδος *get_notification* επιστρέφει όλες τις προσφορές που έχουν αποσταλεί προς τους χρήστες των κινητών συσκευών. Για κάθε προσφορά οι πληροφορίες που ανακτώνται είναι το *id* της προσφοράς, το περιεχόμενο και η ημερομηνία ανακοίνωσής της.

Κεφάλαιο 16°

Επικοινωνία Διακομιστή με iPhone Εφαρμογή

16.1 Επικοινωνία Server - iPhone Client

Τα δεδομένα βρίσκονται στην πλευρά του διακομιστή αποθηκευμένα σε βάση δεδομένων. Οι πληροφορίες πρέπει να είναι προσπελάσιμες από τον client. Σημειώνεται ότι οι τεχνολογίες που χρησιμοποιούνται στις δυο πλευρές είναι διαφορετικές. Το πρόβλημα της επικοινωνίας δύο ετερογενών συστημάτων, επιλύεται με την χρήση γενικευμένων γλωσσών. Οι γλώσσες αυτές προσφέρουν ευέλικτο και ανοικτό τρόπο αναπαράστασης δεδομένων. Στην βιβλιογραφία οι γλώσσες αυτές αναφέρονται και γλώσσες σήμανσης. Μια από αυτές είναι και η XML που χρησιμοποιήθηκε για τις ανάγκες της παρούσας πτυχιακής εργασίας.

16.2 Η Γλώσσα Σήμανσης XML

Η XML (Extensible Markup Language) είναι μία γλώσσα σήμανσης, που περιέχει ένα σύνολο κανόνων για την ηλεκτρονική κωδικοποίηση κειμένων. Ορίζεται, κυρίως, στην προδιαγραφή XML 1.0 (XML 1.0 Specification), που δημιούργησε ο διεθνής οργανισμός προτύπων W3C (World Wide Web Consortium), αλλά και σε διάφορες άλλες σχετικές προδιαγραφές ανοιχτών προτύπων.

Η XML σχεδιάστηκε δίνοντας έμφαση στην απλότητα, τη γενικότητα και τη χρησιμότητα στο Διαδίκτυο. Είναι μία μορφοποίηση δεδομένων κειμένου, με ισχυρή υποστήριξη Unicode για όλες τις γλώσσες του κόσμου. Αν και η σχεδίαση της XML εστιάζει στα κείμενα, χρησιμοποιείται ευρέως για την αναπαράσταση αυθαίρετων δομών δεδομένων, που προκύπτουν για παράδειγμα στις υπηρεσίες ιστού.

Υπάρχει μία ποικιλία διεπαφών προγραμματισμού εφαρμογών, που μπορούν να χρησιμοποιούν οι προγραμματιστές, για να προσπελαύνουν δεδομένα XML, αλλά και διάφορα συστήματα σχημάτων XML, τα οποία είναι σχεδιασμένα για να βοηθούν στον ορισμό γλωσσών, που προκύπτουν από την XML.

Έως το 2009, έχουν αναπτυχθεί εκατοντάδες γλώσσες που βασίζονται στην XML, συμπεριλαμβανομένων του RSS, του SOAP και της XHTML.

Βασική ορολογία

Το περιεχόμενο αυτής της ενότητας, βασίζεται στην προδιαγραφή XML 1.0 [5]. Δεν αποτελεί μία πλήρη λίστα όλων των όρων που υπάρχουν στη γλώσσα XML. Είναι μία εισαγωγή στα βασικά στοιχεία, που συναντώνται στην καθημερινή της χρήση.

Χαρακτήρας Unicode

Εξ ορισμού, ένα κείμενο XML είναι μία ακολουθία χαρακτήρων. Σχεδόν κάθε χαρακτήρας Unicode μπορεί να εμφανίζεται σε ένα κείμενο XML.

Επεξεργαστής και Εφαρμογή

Είναι το λογισμικό που επεξεργάζεται ένα κείμενο XML. Είναι αναμενόμενο, ότι ένας επεξεργαστής δουλεύει για μία εφαρμογή. Υπάρχουν μερικές πολύ συγκεκριμένες απαιτήσεις, σχετικά με το τι μπορεί και τι δεν μπορεί να κάνει ένας επεξεργαστής XML, αλλά καμία, όσον αφορά στη συμπεριφορά της εφαρμογής. Ο επεξεργαστής (όπως ονοματίζεται από την προδιαγραφή), αναφέρεται συχνά, με τον αγγλικό όρο *XML parser*.

Σήμανση και Περιεχόμενο

Οι χαρακτήρες που απαρτίζουν ένα κείμενο XML, αποτελούν είτε τη *σήμανση* είτε το *περιεχόμενό* του. Η σήμανση και το περιεχόμενο, μπορούν να επισημανθούν και να διακριθούν, ύστερα από την εφαρμογή κάποιων απλών συντακτικών κανόνων. Όλα τα αλφαριθμητικά που συνιστούν τη σήμανση, είτε ξεκινούν με το χαρακτήρα "<" και καταλήγουν στο χαρακτήρα ">", είτε ξεκινούν με το χαρακτήρα "&" και καταλήγουν

στο χαρακτήρα ";". Ακολουθίες χαρακτήρων που δε συνιστούν τη σήμανση, αποτελούν το περιεχόμενο ενός κειμένου XML.

Ετικέτα (Tag)

Ένα στοιχείο σήμανσης που ξεκινά με το χαρακτήρα "<" και καταλήγει στο χαρακτήρα ">". Υπάρχουν τρία είδη ετικέτας: *ετικέτες-αρχής*, για παράδειγμα <section>, *ετικέτες-τέλους*, για παράδειγμα </section>, και *ετικέτες-χωρίς-περιεχόμενο*, για παράδειγμα <line-break/>.

Στοιχείο (Element)

Ένα λογικό απόσπασμα ενός κειμένου, που είτε ξεκινά με μία ετικέτα-αρχής και καταλήγει σε μία ετικέτα-τέλους, είτε αποτελείται μόνο από μία ετικέτα-χωρίς-περιεχόμενο. Οι χαρακτήρες που υπάρχουν, αν υπάρχουν, μεταξύ μιας ετικέτας-αρχής και μιας ετικέτας-τέλους, συνιστούν το *περιεχόμενο* του στοιχείου, το οποίο μπορεί να περιέχει σήμανση, συμπεριλαμβανομένων και άλλων στοιχείων, που ονομάζονται *στοιχεία-παιδιά*. Ένα παράδειγμα ενός στοιχείου είναι το <Greeting>Hello, world.</Greeting>. Ένα άλλο είναι το <line-break/>.

Χαρακτηριστικό (Attribute)

Ένα στοιχείο σήμανσης που αποτελείται από ένα ζευγάρι *όνομα/τιμή*, το οποίο υπάρχει μέσα σε μία ετικέτα-αρχής ή σε μία ετικέτα-χωρίς-περιεχόμενο. Στο παράδειγμα παρακάτω, το στοιχείο *img* έχει δύο χαρακτηριστικά, τα *src* και *alt*: . Ένα άλλο παράδειγμα θα ήταν το <store id="3">Connect A to B.</step>, όπου το όνομα του χαρακτηριστικού είναι "id" και η τιμή του είναι "3".

XML Declaration

Τα κείμενα XML μπορούν να αρχίζουν, με τη δήλωση κάποιων πληροφοριών σχετικών με αυτά, όπως στο ακόλουθο παράδειγμα:

<?xml version="1.0" encoding="UTF-8" **?>**

Escaping

Η XML προσφέρει χαρακτήρες διαφυγής, προκειμένου να συμπεριλαμβάνονται χαρακτήρες που δεν μπορούν να χρησιμοποιηθούν απευθείας.

Για παράδειγμα:

- Ο χαρακτήρας < και & είναι key syntax markers και δεν μπορούν να εμφανίζονται στο περιεχόμενο ενός Tag
- Μερικές κωδικοποιήσεις υποστηρίζουν μόνο ένα υποσύνολο από το Unicode. Π.χ., είναι επιτρεπτό να κωδικοποιηθεί ένα XML έγγραφο σε ASCII, όμως το ASCII δεν περιλαμβάνει κάποια στοιχεία όπως το "é".
- Μερικοί χαρακτήρες έχουν σύμβολα που δεν μπορούν να ξεχωρίσουν από κάποιους άλλους χαρακτήρες:
    - Non-breaking-space ( ) σε σχέση με το space ( ),
    - Κυριλικό κεφαλαίο А (А) σε σχέση με το Λατινικό κεφαλαίο A (a)

Predefined entities

- < συμβολίζει "<"
- > συμβολίζει ">"
- & συμβολίζει "&"
- ' συμβολίζει '
- " συμβολίζει "

16.3 Υλοποίηση web-service στην πλευρά του διακομιστή

Ο διακομιστής αναλαμβάνει να μοιράζει τα δεδομένα στις client εφαρμογές. Για τον λόγο αυτό, αναπτύχθηκε η υπηρεσία που τροφοδοτεί τις client εφαρμογές με δεδομένα, σε XML αναπαράσταση. Για την υλοποίηση της υπηρεσίας χρησιμοποιήθηκε η PHP που είναι server side script γλώσσα.

Οι ενέργειες του web-service, είναι να ανασύρει τα δεδομένα από τον SQL Server και κατόπιν να τα συντάσσει σύμφωνα με το πρότυπο που αναλύθηκε παραπάνω. Για την σύνταξη των δεδομένων σε XML αναπαράσταση, χρησιμοποιήθηκε η κλάση DOMDocument.

Αναπτύχτηκαν συνολικά δύο web-services, ένα για κάθε κατηγορία καταστήματος. Η υπηρεσία για τα clubs βρίσκεται στην τοποθεσία kratisinow.gr/fetch-clubs.php, ενώ για τα stages στην τοποθεσία kratisnow.gr/fetch-stages.php.

16.4 Αποστολή Notifications

Η εντολή *Cron* είναι στην πραγματικότητα ένας time-based χρονοπρογραμματιστής σε Unix περιβάλλοντα. Η cron επιτρέπει στους χρήστες να προγραμματίζουν εργασίες (εντολές ή shell scripts), προκειμένου να ενεργοποιούνται σε συγκεκριμένες χρονικές στιγμές. Συνήθως χρησιμοποιείται για να αυτοματοποιήσει διαδικασίες συντήρησης και διαχείρισης, όμως ειναι γενικού σκοπού που σημαίνει οτι μπορεί να χρησιμοποιηθεί και για άλλες λειτουργίες όπως για σύνδεση στο διαδίκτυο και λήψη των emails.

16.4.1 Η εντολή Cron

Η Cron οδηγείται απο ένα crontab αρχείο (cron table) που είναι στην ουσία ένα configuration αρχείο που ορίζει shell commands που εκτελούνται περιοδικά. Τα crontab αρχεία αποθηκεύονται εκεί που βρίσκεται η λίστα με τις διεργασίες και τις οδηγίες προς το cron deamon. Οι χρήστες μπορούν να έχουν τα δικά τους crontab αρχεία και συνήθως υπάρχει ένα crontab του συστήματος που μόνο ο διαχειριστής μπορεί να το επεξεργαστεί

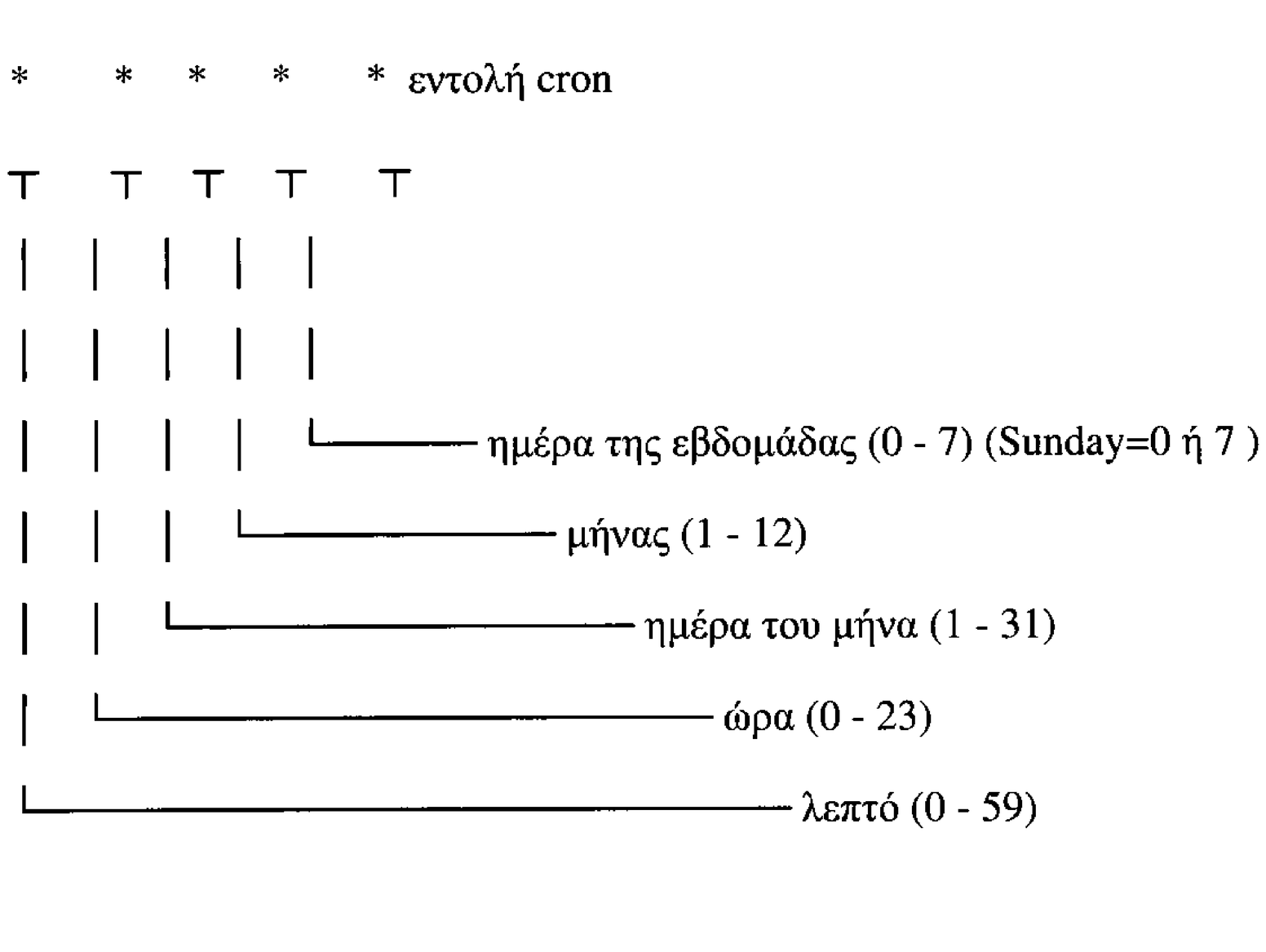
Κάθε γραμμή του crontab file αντιπροσωπεύει μια εργασία και συντάσεται με CRON expression ακολουθούμενο απο το shell command που πρόκειται να εκτελεστεί. Μερικές υλοποιήσεις της εντολής cron υποστηρίζουν έξι πεδία στο format της εντολής: το username του λογαριασμού που θα εκτελέσει το cron job.

Για την ημέρα της εβδομάδας (πεδίο 5), η τιμές 0 ή 7 θεωρούνται ως Κυριακή, αν και μερικές εκδόσεις Unix όπως το AIX δεν υποστηρίζουν την τιμή 7.

Προκαθορισμένες εντολές

Υπάρχουν αρκετές προκαθορισμένες εντολές που μπορούν να αντικαταστήσουν τα CRON expressions.

Entry	Description	Equivalent To
@yearly (or @annually)	Εκτελείται κάθε χρόνο(μεσάνυτα), Jan. 1st	0 0 1 1 *
@monthly	Εκτελείται κάθε μήνα, μεσάνυχτα, την πρώτη του μήνα	0 0 1 * *
@weekly	Εκτελείται μια φορα την εβομάδα, μεσάνυχτα Κυριακής	0 0 * * 0
@daily	Εκτελείται κάθε μέρα μεσάνυχτα	0 0 * * *
@hourly	Εκτελείται κάθε ώρα	0 * * * *



Cron perimissions

Τα δύο αρχεία που ακολουθούν έχουν σημαντικό ρόλο:

- /etc/cron.allow – Εαν αυτό το αρχείο υπάρχει, τότε πρέπει να υπάρχει μέσα το όνομα χρήστη που θα μπορεί να χρησιμοποιεί την εντολή cron

- /etc/cron.deny – Εαν το cron.allow δεν υπάρχει αλλα υπάρχει το cron.deny, τότε όσα ονοματα χρηστών περιλαμβάνονται σε αυτό, δεν μπορούν να χρησιμοποιήσουν την εντολή cron.

Εάν δεν υπάρχει κάνενα απο τα δύο αρχεία, τότε μόνο ο super user μπορεί να εκτελέσει cron jobs ή όλοι χωρίς εξαιραίσεις.

Timezone handling

Οι περισσότερες υλοποιήσεις μεταφράζουν τις εγγραφές του crontab στη ζώνη ώρας(time zone) του συστήματος στην οποία τρέχει ο cron daemon. Αυτό γίνεται γιατί κάποιο μηχάνημα μπορεί να έχει πολλούς χρήστες με διαφορετικές χρονικές ζώνες. Γι'αυτό η εντολή cron υποστηρίζει την μεταβλητή "TZ=<timezone>".

Ειδικοί χαρακτήρες

Μπορούν να χρησιμοποιήθουν ακρετοί χαρακτήρες για να ορίσουν χρονικές περιόδους:

- Το κόμα(‘,’), ορίζει μια λίστα με τιμές. Π.χ: «1,3,6,8»
- Η παύλα (‘-‘), ορίζει ένα εύρος τιμών. Π.χ: «1-6»
- Ο αστερίσκος (‘*’), ορίζει όλες τις δυνατές τιμές του πεδίου. Π.χ ο αστερίσκος στην ώρα σημαίνει «κάθε ώρα».
- Η πισωκάθετος (‘/’), ορίζει μία τιμή που μπορεί να εξαιρεθεί.

16.4.2 Χρήση Cron Job στην λειτουργία των ειδοποιήσεων

Για την υλοποίηση της αποστολής των ειδοποιήσεων (push notification service), χρησιμοποιήθηκε η τεχνολογία cron. Όπως ήδη αναφέρθηκε, η εντολή cron χρονοπρογραμματίζει την εκτέλεση προγραμμάτων- script.

Κάθε φορά που εισάγεται μια νέα ειδοποίηση προς αποστολή, δημιουργούνται νέες εγγραφές στον αντίστοιχο πίνακα της βάσης δεδομένων. Οι εγγραφές αυτές δείχνουν ότι υπάρχουν ειδοποιήσεις σε εκκρεμότητα και πρέπει να αποσταλούν. Σε αυτό το σημείο βοηθάει η τεχνολογία cron. Στην πλευρά του server, έχει χρονοπρογραμματιστεί η εκτέλεση ενός script που ελέγχει κάθε δύο λεπτά για εκκρεμείς αποστολές. Ο έλεγχος του script περιλαμβάνει queries στον πίνακα με τις αποστολές, που βρίσκονται σε εκκρεμότητα. Εφόσον ανιχνευθούν ειδοποιήσεις, τότε γίνεται η αποστολή τους και μετά την επιτυχή ολοκλήρωση της διαδικασίας, οι εγγραφές σβήνονται. Για την σωστή διαχείριση των πόρων του server, η εκτέλεση των ελέγχων γίνεται κάθε 2 λεπτά και όχι συχνότερα.

Βιβλιογραφία

1. **Beginning PHP, Apache, MySQL Web Development**
   Michael K. Glass, Yann Le Scouarnec, Elizabeth Naramore, Gary Mailer, Jeremy Stolz, Jason Gerner
2. **Professional PHP5**
   Ed Lecky-Thompson, Heow Eide-Goodman, Steven D. Nowicki, Alec Cove
3. **Beginning CSS: Cascading Style Sheets for Web Design 2nd Edition**
   Richard York
4. **Google SEO Secrets - How to Get a Top Ranking With Search Engine Optimization**
   Dan Sisson
5. **Prosessional Android 2 – Application Development**
   Reto Meier

LINKS

Ιστοσελίδα

HMTL [http://el.wikipedia.org/wiki/HTML]
JavaScript [http://el.wikipedia.org/wiki/JavaScript]

PHP [http://www.php.net]

PHP [http://el.Wikipedia.org/wiki/PHP]

MYSQL [http://www.mysql.com/]

JQuery [http://en.wikipedia.org/wiki/JQuery]

Ajax [http://en.wikipedia.org/wiki/Ajax_%28programming%29]

SEO [http://en.wikipedia.org/wiki/Search_engine_optimization]

Htaccess [http://httpd.apache.org/docs/1.3/howto/htaccess.html]

Htaccess [http://en.wikipedia.org/wiki/Htaccess]

Rewrite engine [http://en.wikipedia.org/wiki/Rewrite_engine]

CAPTCHA [http://en.wikipedia.org/wiki/CAPTCHA]

SQL injection [http://en.wikipedia.org/wiki/SQL_injection]
JSON RPC [http://www.jsonrpc.org/spec.html]

JSON RPC [http://en.wikipedia.org/wiki/JSON-RPC]

CronJob [http://en.wikipedia.org/wiki/Crontab#CRON_expression]

CronJob [http://www.thesitewizard.com/general/set-cron-job.shtml]

CronJob [http://docs.phplist.com/SetupCronJob]

Android

Android Developers [http://developer.android.com]

Open Handset Alliance [http://www.openhandsetalliance.com]

Andbook [http://andbook.anddev.org]

iPhone

iOS Developer Library [http://developer.apple.com/library/ios/navigation/]

App Store Wikipedia [http://en.wikipedia.org/wiki/App_Store_(iOS)]

Objective-C Wikipedia [http://en.wikipedia.org/wiki/Objective-C]

Παράρτημα

permissions.php

```
<?php

class Permissions
{
    /*
     * Function to assign Permissions based on Group membership
     */

    public function permission($p)
    {
        //Split the levels string into an array
        $perm = str_split($p);

        //Permissions array
        $up = array(
            'tab_1' => 1,
            'tab_2' => 2,
            'tab_3' => 3,
            'tab_4' => 4,
            'tab_5' => 5,
            'tab_6' => 6,
            'tab_7' => 7,
            'tab_8' => 8,
            'tab_9' => 9,
            'tab_10' => 10,

        );
        $combine = array_combine(array_keys($up), $perm);

        return $combine;
    }

    public function findPermissionString($user,$hasEdit)
    {

        $menulevels = array(
            'User' => "000000111",
            'Admin' => "111111111",
        );


        foreach ($menulevels as $key => $value) {
            if ($key == $user)
                return $value;
        }
    }

}
?>
```

users_list.php

```
<?php
/* --------------- PAGER ------------------- */
require_once '../library/Pager/Pager.php';

$result = $DB->query("SELECT count(*) AS total FROM user");
$row = $DB->fetchArray($result);

/* Total number of rows in the logs table */
$totalItems = $row['total'];

$pager_options = array(
      //'itemData' => $num_rows ,
      'totalItems' => $totalItems,
      'perPage' => 10,
      'delta' => 3, // for 'Jumping'-style a lower number is better
      'separator' => ' | ',
      'mode' => 'Sliding', //try switching modes
);

/* Initialize the Pager class with the above options */
$pager = Pager::factory($pager_options);

list($from, $to) = $pager->getOffsetByPageId();

/*
The MySQL 'LIMIT' clause index starts from '0', so decrease the $from by 1
*/
$from = $from - 1;

/* The number of rows to get per query */
$perPage = $pager_options['perPage'];

//SEARCH possible cases
$result = $DB->query("SELECT * FROM user LIMIT $from , $perPage");

/* -------------- PAGER ------------------- */

if ($totalItems == 0)
{
      $array .='<tr class="sectionrow">';
      $array .='<td width="27%" align="center" colspan="12">' . $RESULTS .
'</td>';
      $array .='</tr>';
}
else
{
      $i = $counter-10;

      while ($row = $DB->fetchArray($result))
      {
            $i++;
            $table = 'user';

            if ($row[active]==1)
            {
```

```
                        $hidden = '<a href="#"
onClick="changeActive('.$row[active].','.$row[id].',\''.$table.'\'); return
false;"><img src="' . BASE_URL . BASE_URI . images/visible_16.png"
alt="Hidden/Vissible" title="Hidden/Vissible" /></a>';
                }
                else
                {
                        $hidden = '<a href="#"
onClick="changeActive('.$row[active].','.$row[id].',\''.$table.'\'); return
false;"><img src="' . BASE_URL . BASE_URI . images/hidden_16.png"
alt="Vissible/Hidden" title="Vissible/Hidden" /></a>';
                }

                $array .='<tr class="sectionrow" id=' . $row[id] . '>';
                $array .='<td ><strong>'.$i.'.</strong></td>';
                $array .='<td align="left">
                                        <a
href="edit.php?id='.$row[id].'">'.$row[username].'</a>
                                </td>';
                $array .='<td
align="left">'.ViewGroupsTag($row[group_id]).'</td>';
                $array .='<td align="center" >'.$hidden.'</td>';
                $array .='<td align="center">
<a href="edit.php?id='.$row[user_id].'"><img style="vertical-align:
middle;" src="' . BASE_URL . BASE_URI . images/edit-icon.png" /></a>
                        </td>';
                $array .='<td align="center">
<a href="#" class="delete"><img style="vertical-align: middle;" src="' .
BASE_URL . BASE_URI . images/delete_16.png" /></a>
                        </td>';
                $array .='</tr>';
        }
}
$array .= '<tr><td colspan="5" align="center"><br />'.$pager-
>links.'</td></tr>';

echo $array;

/* --------------- END PAGER ------------------- */

?>
```

load_my_profile

```
<?php

function load_my_profile()
{
    global $DB, $PROFILE;

    $sql = $DB->query("SELECT
                                *
                        FROM
                                user
                        WHERE
                                user_id = '" . $_SESSION['id'] . "'");
```

```
$user_profile = $DB->fetchAssoc($sql);

$ViewGroups = ViewGroupsTag($user_profile['group_id']);
$ViewActive = ViewActiveTag($user_profile['active']);

$user_profile_array = array(

                $PROFILE['NAME'] => $user_profile['firstname'],
                 $PROFILE['SURNAME'] => $user_profile['lastname'],
                 $PROFILE['USERNAME'] => $user_profile['username'],
                 $PROFILE['MOBILE'] => $user_profile['mobile'],
                 $PROFILE['ADDRESS'] => $user_profile['address'],
                 $PROFILE['EMAIL'] => $user_profile['mail_'address''],
                 $PROFILE['GROUP'] => $ViewGroups,
                 $PROFILE['ACTIVE'] => $ViewActive,
                 $PROFILE['IP'] => $user_profile['ip'],
                 $PROFILE['LAST_LOGIN'] => date("d/m/y
H:i:s",$user_profile['login_date']),
                  $PROFILE['INDATE'] => date("d/m/y
H:i:s",$user_profile['indate'])

                              );

    return $user_profile_array;
}

?>
```

load_all_pages()

```
<?php

function load_all_pages()
{
    global $DB;

    $query_user = $DB->query("SELECT * FROM store_type");
    while ($row_parent = $DB->fetchArray($query_user))
    {

        $store_type_id = $row_parent['store_type_id'];

        $content .= '<tr class="sectionrow">
                        <td width="3%"></td>
                        <td width="72%" align="left">' .
$row_parent['store_type_name'] . '</td>
                        <td  width="5%" align="center"></td>
                        <td  width="5%" align="center"></td>
                        <td  width="5%" align="center"></td>
                        <td  width="10%" align="center"></td>
                      </tr>';

        $result = $DB->query("SELECT * FROM store WHERE store_type_id =
$store_type_id ORDER BY name ASC");
```

```
        while ($row = $DB->fetchArray($result))
        {
            $table = 'store';

            if ($row['store_active']==1)
            {
                $hidden = '<a href="#"
onClick="changeActive('.$row['store_active'].','.$row['store_id'].',\''.$ta
ble.'\'); return false;"><img src="' . BASE_URL . BASE_URI .
images/visible_16.png" alt="Hidden/Vissible" title="Hidden/Vissible"
/></a>';
            }
            else
            {
                $hidden = '<a href="#"
onClick="changeActive('.$row['store_active'].','.$row['store_id'].',\''.$ta
ble.'\'); return false;"><img src="' . BASE_URL . BASE_URI .
images/hidden_16.png" alt="Vissible/Hidden" title="Vissible/Hidden"
/></a>';
            }
            $content .='<tr class="sectionrow" id=' . $row['store_id'] .
'>
                            <td width="3%"></td>
                            <td width="72%" align="left" style="padding-
left:50px">
                                <a href="modify.php?page_id=' .
$row['store_id'] . '" >'. $row['store_name'] . '</a>
                            </td>
                            <td  width="5%" align="center">' .
$row['store_id'] . '</td>
                            <td  width="5%" align="center"><a href="' .
BASE_URL .'section.php?store=' . $row['store_id'] . '" target="_blank"><img
src="' . BASE_URL . BASE_URI . images/demo.png" alt="view demo" title="View
Demo"/></a></td>
                            <td width="5%" align="center">' . $hidden .
'</td>
                            <td  width="10%" align="center">
                                <a href="settings.php?page_id=' .
$row['store_id'] . '" ><img src="' . BASE_URL . BASE_URI .
images/edit_16.png" alt="Page settings" title="Page settings"/></a>
                                <a href="#" class="delete"><img
style="padding-left:10px" src="' . BASE_URL . BASE_URI . images/trash-
icon_2.png" alt="Delete page" title="Delete page"/></a>
                            </td>
                        </tr>';
        }
    }

    return $content;
}

?>
```

geocoder.js

```
<script type="text/javascript">
        var geocoder;
        var map;
        var marker;
        function initialize(lat,longt){
            //MAP
            var latlng = new google.maps.LatLng(lat,longt);
            var options = {
                zoom: 16,
                center: latlng,
                mapTypeId: 'roadmap'
            };
            map = new google.maps.Map(document.getElementById("map_canvas"),
options);
            //GEOCODER
            geocoder = new google.maps.Geocoder();
            marker = new google.maps.Marker({
                position: latlng,
                map: map,
                draggable: true
            });
        }
</script>
```

editor()

```
<?php

function Editor($value, $section, $editor, $type)
{
	GLOBAL $sizeInputTextarea,$page_errors;

	$content ='<tr>
	<td width="170" class="insert" align="right" style="padding-left:
5px"><strong>'.$section.'</strong></td>
	<td align="left" class="value_input">
	<div id="alerts" style="margin:3px 0px 0px 3px;">
	<noscript>
	<p>
	<strong>CKEditor requires JavaScript to run</strong>. In a browser
with no JavaScript
	support, like yours, you should still see the contents (HTML data)
and you should
	be able to edit it normally, without a rich editor interface.
	</p>
	</noscript>
	</div>'.
	Return_create_form_input($editor, $type , $sizeInputTextarea,
$page_errors, $value)
	.'
	<script type="text/javascript">
	//<![CDATA[

	CKEDITOR.replace( \''.$editor.'\',
	{
	enterMode : CKEDITOR.ENTER_BR,
	entities: false,
	htmlEncodeOutput: false,
	fullPage : false,
	language:\'el\'
});
//]]>
</script>
</td>
</tr>';

	return $content;
}
?>
```