Πανεπιστήμιο Πατρών

Τμήμα Μηχανικών Η/Υ & Πληροφορικής

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη Κατανεμημένων Εφαρμογών σε Περιβάλλον Spring.io

Κακοσίμου Δήμητρα Α.Μ: 4454

Επιβλέπων: Καθηγητής Χρήστος Ζαρολιάγκης

Συνεπιβλέπων: Δρ. Ιωάννης Χατζηγιαννάκης

Πάτρα 2014

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον κ. Χρήστο Ζαρολιάγκης, Καθηγητή του Τμήματος Μηχανικών Ηλεκτρονικών Υπολογιστών και Πληροφορικής και επιβλέποντα της διπλωματικής μου, για την εμπιστοσύνη που μου έδειξε δίνοντας μου την ευκαιρία να ασχοληθώ με το αντικείμενο αυτής της εργασίας.

Επίσης, ένα θερμό ευχαριστώ στον Δρ. Ιωάννη Χατζηγιαννάκη, συνεπιβλέποντα της διπλωματικής μου, για την εμπιστοσύνη, τη βοήθεια και την καθοδήγηση του σε όλη τη διάρκεια εκπόνησης αυτής της εργασίας. Θέλω επίσης να ευχαριστήσω τον Δημήτριο Καραβία για την πολύτιμη βοήθεια του και τη συνεργασία που είχαμε κατά τη διάρκεια εκπόνησης αυτής της εργασίας.

Δήμητρα Κακοσίμου

Πάτρα 2014

Περιεχόμενα

Ι. Εισαγωγή

1.1 Κίνητρο και σημασία

Τα τελευταία χρόνια βιώνουμε μία επανάσταση των έξυπνων κινητών τηλεφώνων (smartphones), τα οποία ενσωματώνουν πλέον δυνατότητες, οι οποίες δεν υπάρχουν στα συμβατικά κινητά τηλέφωνα. Περιλαμβάνουν ολοκληρωμένο λειτουργικό σύστημα, ισχυρό επεξεργαστή ικανό για εκτέλεση πολύπλοκων και χρονοβόρων υπολογισμών, δυνατότητα σύνδεσης στο διαδίκτυο είτε μέσω ασύρματων δικτύων (Wi-Fi) είτε δικτύων τρίτης και τέταρτης γενιάς (3G - 4G networks), συστήματα εντοπισμού θέσης (GPS) κ.α. Τα λειτουργικά συστήματα των συσκευών αυτών ποικίλουν, με το Android της Google να περιλαμβάνεται τη στιγμή αυτή στο μεγαλύτερο ποσοστό των smartphones της αγοράς. Επίσημα στοιχεία δείχνουν ότι η ιστοσελίδα της Google, Android Market, που περιέχει όλες τις εφαρμογές για Android περιλαμβάνει αυτή την στιγμή περισσότερες από 425.000 εφαρμογές.

1.2 Στόχοι της διπλωματικής εργασίας

Οι στόχοι αυτής εργασίας περιστρέφονται γύρω από το σύστημα διαχείρισης ερευνητικών έργων ΝΕΔΑ που χρησιμοποιείται από το Ινστιτούτο Τεχνολογίας Υπολογιστών & Εκδόσεων (Ι.Τ.Υ.Ε.) για την παρακολούθηση των ερευνητικών αναπτυξιακών έργων.

Στο πλαίσιο αυτό, σκοπός της παρούσας διπλωματικής εργασίας είναι η ανάπτυξη μιας εφαρμογής, που προορίζεται για τα παραπάνω κινητά τηλέφωνα και όσες συσκευές περιλαμβάνουν το λειτουργικό σύστημα Android (tablets, pc). Μέσα από την εφαρμογή αυτή, ο χρήστης θα μπορεί να διαχειρίζεται τα ταξίδια του τόσο την εισαγωγή νέου ταξιδιού όσο και τον έλεγχο παλαιότερων ταξιδιών.

1.3 Συνεισφορά διπλωματικής

Η συνεισφορά της παρούσας διπλωματικής εργασίας αφορά τις ανάγκες του οργανισμού Ι.Τ.Υ.. Εργαζόμαστε πάνω στο πληροφοριακό σύστημα διαχείρισης έργων ΝΕΔΑ το οποίο χρησιμοποιείται από τον οργανισμό για την καλύτερη παρακολούθηση και οργάνωση των έργων και των εργαζομένων σε αυτά. Επεκτείνουμε τις δυνατότητές αυτού του πληροφοριακού συστήματος ώστε να δώσουμε την δυνατότητα καλύτερης παρακολούθησης των ταξιδιών που πραγματοποιούνται από τους εργαζόμενους του, καθώς και την δυνατότητα εισαγωγής νέας αίτησης ταξιδιού από συσκευές με λειτουργικό σύστημα Android. Κάνουμε την διαδικασία εγκρίσεων πιο εύκολη, ειδοποιώντας τον διαχειριστή για νέες αιτήσεις ώστε να εξυπηρετήσουμε τις

ανάγκες του οργανισμού, ενώ αυτόματα παρέχεται η δυνατότητα παρακολούθησης των ταξιδιών που χρεώνονται στα έργα του οργανισμού παρέχοντας άμεσο έλεγχο στις δαπάνες μετακίνησης που προκύπτουν σε κάθε έργο.

1.3 Δομή της διπλωματικής εργασίας

Η διπλωματική αυτή εργασία αναλύεται συνολικά σε 6 κεφάλαια.

Στο **1ο Κεφάλαιο** ασχολούμαστε με την εισαγωγή στο αντικείμενο που πραγματεύεται η διπλωματική εργασία. Εισάγουμε τον αναγνώστη στις νέες τεχνολογίες όσον αφορά τα κινητά τηλέφωνα.

Στο **2ο Κεφάλαιο** παρουσιάζονται τεχνολογίες, εργαλεία και τεχνικές που χρησιμοποιήθηκαν για την σχεδίαση και την υλοποίηση της πλατφόρμας.

Στο **3ο Κεφάλαιο** περιγράφεται το λειτουργικό σύστημα Android. Εξηγούνται τα προγραμματιστικά εργαλεία που χρειάζονται και αναλύονται τα βασικά βήματα που απαιτούνται για την ανάπτυξη μιας εφαρμογής που θα εκτελείται σε αυτό το λειτουργικό σύστημα. Περιγράφονται κάποιες προγραμματιστικές τεχνικές και αναλύονται τρόποι βελτιστοποίησης κάθε εφαρμογής ώστε να είναι λειτουργική και προσβάσιμη στο ευρύ κοινό. Αποτελεί λοιπόν ένα οδηγό που πρέπει να ακολουθήσει ο προγραμματιστής ώστε να αναπτύξει μια εφαρμογή Android.

Στο **4ο** και **5ο Κεφάλαιο** παρουσιάζεται η προτεινόμενη αρχιτεκτονική του συστήματος και αναλύονται τα διάφορα επίπεδα της.

Στο **6ο κεφάλαιο** παρουσιάζονται διεξοδικά όλες οι λειτουργίες της εφαρμογής και οι επιλογές που έχει ο χρήστης αλληλεπιδρώντας με αυτήν. Περιγράφεται κάθε διαφορετική οθόνη που συναντά καθώς και οι επιλογές που έχει όπως επίσης και ο τρόπος υπολοίησης αυτών των λειτουργιών στο προγραμματιστικό περιβάλλον. Όλα τα αποτελέσματα παρουσιάζονται σε εικόνες, οι οποίες εμφανίζουν ένα στιγμιότυπο από την αντίστοιχη οθόνη της συσκευής που εκτελεί την εφαρμογή και βοηθάνε τον αναγνώστη στην κατανόηση των παραπάνω λειτουργιών.

Στο **7ο Κεφάλαιο** αναφέρονται μελλοντικές κατευθύνσεις για την πλατφόρμα. Τέλος, παρουσιάζεται η Βιβλιογραφία που χρησιμοποιήθηκε για την συγγραφή του παρόντος κειμένου, καθώς και οι ιστοσελίδες και όλες οι πηγές που βοήθησαν στην ανάπτυξη της εφαρμογής.

II. Τεχνολογίες και εργαλεία

2.1 Spring

Η Spring είναι ένα ελεύθερο (open source) περιβάλλον εργασίας για εφαρμογές Java που δημιουργήθηκε από τον Rob Johnson και περιγράφηκε στο βιβλίο του ‘‘Expert One-on-One: J2EE Design and Development’’. Αναπτύχθηκε κατά κύριο λόγο, για να αντιμετωπίσει την πολυπλοκότητα ανάπτυξης επιχειρησιακών εφαρμογών. Η Spring κάνει εφικτή την χρήση απλών JavaBeans για την επίτευξη λειτουργιών που προηγουμένως μπορούσαν να γίνουν μόνο μέσω EJBs. Παρόλα αυτά, η Spring δεν είναι μόνο χρήσιμη για την ανάπτυξη server-side εφαρμογών. Η χρήση της μπορεί να βοηθήσει στην απλοποίηση του κώδικα, τον ευκολότερο και αποτελεσματικό έλεγχο και τη χαλαρή διασύνδεση (loose coupling) κάθε Java εφαρμογής. Η Spring εκτελεί πολλές διαφορετικές λειτουργίες, αλλά, αν αναλυθεί στα βασικά της τμήματα, θα φανεί ότι είναι ένας ελαφρύς (lightweight) aspect-oriented container ο οποίος εφαρμόζει dependency injection, ενώ παράλληλα αποτελεί και περιβάλλον εργασίας (framework). Πιο αναλυτικά:

- Lightweight (Ελαφρύς) — Ο όρος αυτός αναφέρεται τόσο στο μέγεθος της πλατφόρμας όσο και στον φόρτο εργασίας (overhead) που απαιτείται. Η διανομή του Spring Framework μπορεί να συμπεριληφθεί σε ένα απλό jar αρχείο μεγέθους 2,5 MB, ενώ ο επιπρόσθετος φόρτος εργασίας που απαιτείται είναι αμελητέος. Ο προγραμματιστής χρειάζεται να κάνει ελάχιστες, αν όχι καθόλου, αλλαγές στον κώδικα της εφαρμογής που αναπτύσσει έτσι ώστε να επωφεληθεί από τον πυρήνα της, ενώ μπορεί οποιαδήποτε στιγμή να σταματήσει να τη χρησιμοποιεί.

Βέβαια αυτή η δυνατότητα παρέχεται μόνο στον πυρήνα της Spring. Πολλά επιπλέον μέρη της, όπως είναι η πρόσβαση δεδομένων, απαιτούν πολύ στενότερη 'διασύνδεση' (coupling) από ότι το Spring framework. Εντούτοις, το κέρδος στις περισσότερες από αυτές τις περιπτώσεις είναι πολύ σημαντικό.

- Dependency Injection (Εξαρτώμενη Έγχυση) — Η Spring προάγει τη χαλαρή διασύνδεση χρησιμοποιώντας μία τεχνική που είναι γνωστή ως dependency injection (DI). Όταν εφαρμόζεται η DI, τα αντικείμενα λαμβάνουν παθητικά τις εξαρτήσεις τους (dependencies) αντί να τις δημιουργούν ή να τις αναζητούν μόνα τους. Είναι κάτι σαν ένα αντίστροφο Java Naming and Directory Interface (JN-DI) όπου αντί ένα αντικείμενο να ψάχνει μόνο του για τις εξαρτήσεις του σε έναν υποδοχέα (container), ο υποδοχέας δίνει τις εξαρτήσεις στο αντικείμενο χωρίς να περιμένει πρώτα να ερωτηθεί.

- Aspect - Oriented — Η Spring παρέχει πλούσια υποστήριξη σε Aspect - Oriented Programming (AOP). Αυτό έχει σαν αποτέλεσμα τη δημιουργία εφαρμογών με περισσότερη συνοχή ακόμη και αν είναι απαραίτητη η συνύπαρξη διαφορετικών λογικών λειτουργίας. Έτσι, κάθε αντικείμενο της εφαρμογής περιορίζεται στις δικές του λειτουργίες και δεν είναι υπεύθυνο (ενδεχομένως δε γνωρίζει καν) για λειτουργίες που έχουν να κάνουν με άλλα συστήματα. Τέτοιου είδους λειτουργίες μπορεί να είναι η υποστήριξη συνδιαλλαγών ή η καταγραφή των διαφόρων κινήσεων (logging).

- Container (Υποδοχέας) — Η Spring αποτελεί έναν container με την έννοια ότι περιέχει τα αντικείμενα της εφαρμογής και διαχειρίζεται τον κύκλο ζωής τους. Ο προγραμματιστής μπορεί να ορίσει τον τρόπο με

τον οποίο θέλει να διαμορφώνονται και να δημιουργούνται τα αντικείμενα, καθώς και ποιες θέλει να είναι οι σχέσεις μεταξύ τους.

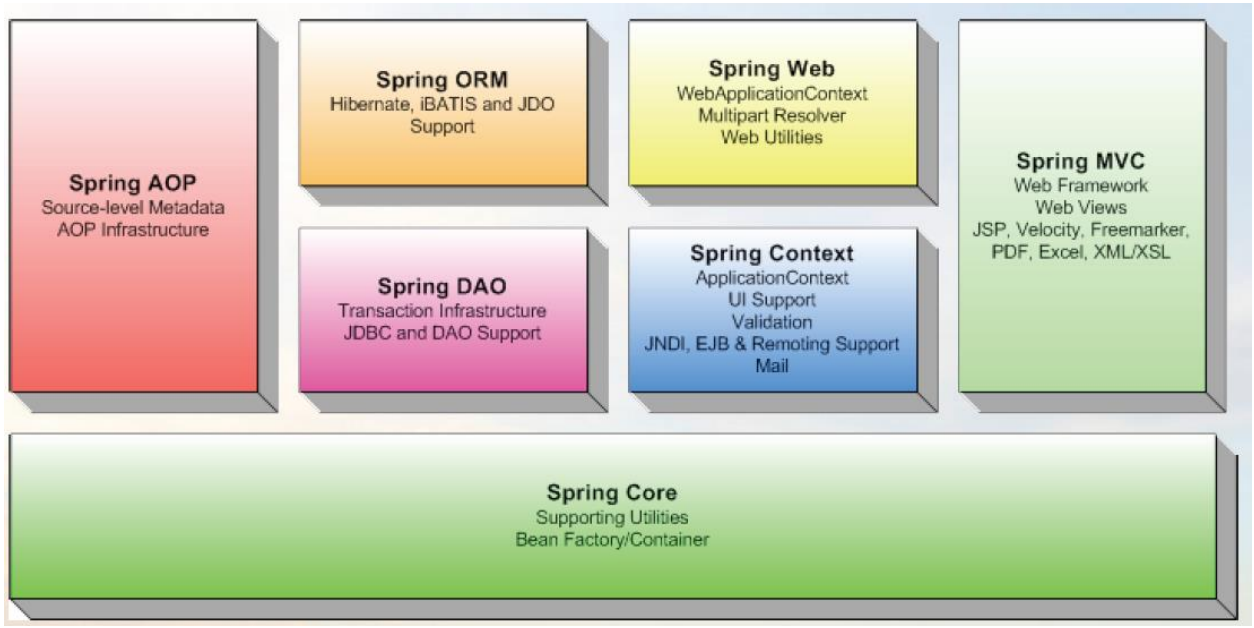
- Framework (Περιβάλλον Εργασίας) — Η Spring δίνει τη δυνατότητα να δημιουργηθούν πολύπλοκες εφαρμογές με το συνδυασμό άλλων, λιγότερο πολύπλοκων, κομματιών. Στη Spring τα αντικείμενα δημιουργούνται μέσω δηλώσεων σε απλά XML αρχεία. Επίσης παρέχει πολλές δομικές λειτουργίες, όπως η διαχείριση συναλλαγών κ.ά., επιτρέποντας στον προγραμματιστή να ασχοληθεί περισσότερο με τη λογική της εφαρμογής του.

Έτσι, μπορούμε να πούμε ότι η Spring είναι ένα περιβάλλον εργασίας το οποίο βοηθάει τον προγραμματιστή να δημιουργήσει εφαρμογές με χαλαρά διασυνδεδεμένο κώδικα. Ακόμη και αν αυτό ήταν το μόνο πλεονέκτημα αυτής της πλατφόρμας, τα οφέλη θα ήταν πάρα πολλά από άποψη συντηρησιμότητας και ελεγξιμότητας. Ωστόσο τα πλεονεκτήματα της Spring δεν περιορίζονται στα παραπάνω. Περιέχει κομμάτια που χρησιμοποιούν DI και AOP και έτσι συνθέτουν μία αρκετά αξιόλογη πλατφόρμα στην οποία μπορούν να αναπτυχθούν εφαρμογές.

2.1.1 Τα μέρη στης Spring

Το περιβάλλον εργασίας της Spring αποτελείται από αρκετά, καλά ορισμένα μέρη(modules), τα οποία ως σύνολο δίνουν στον προγραμματιστή ό, τι χρειάζεται για την ανάπτυξη εφαρμογών enterprise. Δεν χρειάζεται όλη η εφαρμογή να βασίζεται στη Spring. Ο προγραμματιστής είναι ελεύθερος να επιλέξει εκείνα τα μέρη που ταιριάζουν στην εφαρμογή του αλλά και να αναζητήσει άλλες επιλογές όταν η Spring δεν τον καλύπτει. Επίσης, παρέχει τρόπους διασύνδεσης με περιβάλλοντα εργασίας και βιβλιοθήκες, έτσι ώστε ο προγραμματιστής να μην χρειαστεί να τα αναπτύξει από την αρχή.

Οπως φαίνεται στο παρακάτω σχήμα, όλα τα μέρη της Spring είναι χτισμένα πάνω στον container του πυρήνα της. Ο πυρήνας καθορίζει τον τρόπο που δημιουργούνται, αρχικοποιούνται και χειρίζονται τα beans, δηλαδή οι κύριες λειτουργίες της Spring. ΄Ομως ο προγραμματιστής είναι πολύ πιθανό να ενδιαφέρεται για τα άλλα μέρη της Spring τα οποία χρησιμοποιούν τις υπηρεσίες που προσφέρει ο πυρήνας.



- **Spring Core**

Ο πυρήνας παρέχει τη βασική λειτουργικότητα του περιβάλλοντος εργασίας της Spring. Σε αυτό το μέρος περιλαμβάνεται το BeanFactory, το οποίο είναι ο θεμελιώδης container της Spring και η βάση στην οποία στηρίζεται το DI της Spring.

- **AOP module**

Η Spring παρέχει υποστήριξη για aspect-oriented programming μέσω του AOP module της.΄Οπως και το DI, το AOP υποστηρίζει τη χαλαρή διασύνδεση μεταξύ των αντικειμένων κάθε εφαρμογής. Ωστόσο, με το AOP, σημαντικά θέματα που αφορούν μια εφαρμογή (όπως οι συναλλαγές και οι ασφάλεια) χωρίζονται από τα αντικείμενα στα οποία εφαρμόζονται.

- **ORM module**

Για αυτούς που προτιμούν να χρησιμοποιούν ένα εργαλείο για object-relation mapping (ORM) αντί του JDBC, η Spring παρέχει το ORM module. Η υποστήριξη της Spring για ORM χτίζεται πάνω στο DAO παρέχοντας έναν αρκετά βολικό τρόπο για ανάπτυξη DAO για αρκετές λύσεις που προσφέρουν object-relational mapping. Η Spring δεν υλοποιεί κάποια λύση για ORM, υποστηρίζει αρκετά δημοφιλή περιβάλλοντα εργασίας, συμπεριλαμβανομένων των Hibernate, Java Persistence API, Java Data Objects και iBatis SQL Maps.

Επιπλέον, το σύστημα διαχείρισης συναλλαγών της Springυποστηρίζει και το JDBC.

- **DAO module**

Η ανάπτυξη με JDBC συχνά περιλαμβάνει τη συγγραφή αρκετού κώδικα για τη σύνδεση στη βάση δεδομένων, τη δημιουργία κάποιας εντολής (statement), την επεξεργασία του αποτελέσματος και το κλείσιμο της σύνδεσης με τη βάση. Μέσω του Data Access Object (DAO) module, η Spring απλοποιεί τη διαδικασία που περιγράφηκε, διατηρεί τον κώδικα που αφορά τη βάση δεδομένων απλό και κατανοητό, και περιορίζει τα προβλήματα που μπορεί να προκύψουν. Επιπλέον, χτίζει ένα επίπεδο με κατανοητές εξαιρέσεις πάνω από τα μηνύματα σφάλματος που στέλνουν οι εξυπηρετητές των βάσεων δεδομένων. Επιπρόσθετα, αυτό το module (χρησιμοποιώντας το AOP module) παρέχει υπηρεσίες διαχείρισης συναλλαγών για τα αντικείμενα των εφαρμογών που χρησιμοποιούν τη Spring.

- **Web module**

Η Spring παρέχει μία πλούσια συλλογή κλάσεων για τη δημιουργία εφαρμογών που στηρίζονται στο διαδίκτυο (web-based applications) μέσω του web module της. Υποστηρίζεται η χρήση του προτύπου σχεδίασης εφαρμογών Model/View/Controller, ο χειρισμός ηλεκτρονικής αλληλογραφίας, καθώς και η χρήση απομακρυσμένων υπηρεσιών (remote support). ΄Οσον αφορά τη χρήση του προτύπου σχεδίασης εφαρμογών Model/View/Controller (MVC), η Spring υποστηρίζει πλήρως το πιο γνωστό

MVC περιβάλλον εργασίας, το Apache Struts. Με αυτόν τον τρόπο δίνεται η δυνατότητα στον προγραμματιστή να χρησιμοποιήσει στις ήδη σχεδιασμένες κλάσεις του, τις αρχές του DI που προσφέρει η Java. Βέβαια, πέραν της υποστήριξης του Apache Struts, η Spring προσφέρει και τη δική της υλοποίηση για το πρότυπο MVC. Μέσω αυτού μπορεί να χρησιμοποιηθεί μια ευρεία γκάμα τεχνολογιών παρουσίασης, από σελίδες JSP και Apache Jakarta Velocity μέχρι Microsoft Excel και Adobe PDF.

Τέλος, για τη χρήση απομακρυσμένων υπηρεσιών, παρέχεται εκτεταμένη υποστήριξη μία μεγάλης συλλογής τεχνικών απομακρυσμένης πρόσβασης, για τη γρήγορη δημιουργία και πρόσβαση σε απομακρυσμένες υπηρεσίες. Τέτοιες τεχνικές είναι οι Java RMI, JAXRPC, Caucho Hessian και Caucho Burlap. Εκτός αυτών, η Spring παρέχει και το δικό της πρωτόκολλο, που χρησιμοποιεί το HTTP πρωτόκολλο επικοινωνίας και βασίζεται στο απλό Java serialization.

2.1.2 Παράδειγμα στην Spring

Το πιο βασικό πράγμα που κάνει η Java είναι dependency injection. Στη συνέχεια θα παρουσιαστεί μια απλή εφαρμογή, μια παραλλαγή του συνηθισμένου ‘‘Hello World’’, η οποία θα παρουσιάσει τα βασικά σημεία της Spring. Η πρώτη κλάση που χρειάζεται η Hello World εφαρμογή είναι μία service class, η οποία θα τυπώνει το γνωστό χαιρετισμό. Παρακάτω φαίνεται το interface GreetingService το οποίο ορίζει τις συναρτήσεις της κλάσης.

```
public interface GreetingService {
        public void sayGreeting ( ) ;

}
```

Ακολούθως παρουσιάζεται η κλάση GreetingServiceImpl η οποία υλοποιεί το παραπάνω interface. Η χρήση interface για τον ορισμό των ενεργειών που μπορούν να γίνουν με τα διάφορα αντικείμενα δεν είναι απαραίτητη, εντούτοις η πολιτική αυτή συνιστάται.

```
public class GreetingServiceImpl implements GreetingService {

    private String greeting ;

    public GreetingServiceImpl ( ) { }

    public GreetingServiceImpl ( String greeting ) {
      this. greeting = greeting ;
     }

    public void  sayGreeting ( ) {
      System . out . println ( greeting ) ;
    }

    public void  setGreeting ( String greeting ) {
      this. greeting = greeting ;
    }
}
```

Η κλάση GreetingServiceImpl έχει μία μόνο μεταβλητή μέλος, την greeting. Αυτή είναι ένα απλό String το οποίο θα κρατάει το χαιρετισμό που θα τυπωθεί όταν θα καλεστεί η μέθοδος sayGreeting(). Επιπλέον, έχει και δύο μεθόδους δημιουργούς, έναν κενό και έναν που παίρνει ως όρισμα το χαιρετισμό. Οποιοσδήποτε, από τους δύο αυτούς δημιουργούς μπορεί να κληθεί, ανάλογα με τις παραμέτρους που θα οριστούν. Παρακάτω φαίνεται το αρχείο

ρυθμίσεων (configuration file) hello.xml το οποίο ορίζει στον πυρήνα της Spring πως ακριβώς πρέπει να εκτελέσει την υπηρεσία που δημιουργήθηκε.

```
<?xml version=" 1.0 " encoding="UTF8" ?>

 <beans xmlns=" http://www. springframework . org/schema/beans "

       xmlns:xsi=" http://www.w3. org/2001/XMLSchema-instance "
       xsi:schemaLocation=" http://www. springframework . org/schema/beans
          http://www. springframework . org/schema/beans/spring-beans-2.0.xsd
">

     <bean id=" greetingService "
            class="com. springinaction . chapter01 . hello . GreetingServiceImpl
">
          <property name=" greeting " value="Buenos Dias ! " />
     </bean>
</beans>
```

Στο παραπάνω αρχείο δηλώνεται ένα στιγμιότυπο της GreetingServiceImpl στον container της Spring στο οποίο ορίζεται ότι η μεταβλητή μέλος (property) θα έχει την τιμή ‘‘Buenos Dias!’’. Αναλύοντας λίγο τον παραπάνω XML κώδικα βλέπουμε ότι το root στοιχείο είναι το <beans>, κάτι που ισχύει για οποιοδήποτε αρχείο ρυθμίσεων της Spring. Το στοιχείο <bean> λέει στον container για μία κλάση πώς αυτή πρέπει να αρχικοποιηθεί. Το χαρακτηριστικό (attribute) id του στοιχείου <bean> χρησιμοποιείται, για να ορίσει το όνομα του bean, ενώ το class χρησιμοποιείται για να οριστεί το ακριβές όνομα της κλάσης (classpath). Μέσα στο στοιχείο <bean> ορίζεται ένα στοιχείο <property> το οποίο χρησιμοποιείται για να οριστεί μία μεταβλητή μέλος, σε αυτή την περίπτωση, η greeting. Το στοιχείο property λέει στον πυρήνα να καλέσει την μέθοδο sayGreeting() δίνοντας της την τιμή ‘‘Buenos Dias!’’ κατά την αρχικοποίηση του bean.

Παρακάτω φαίνεται τι είναι αυτό που κάνει ακριβώς ο πυρήνας κατά την αρχικοποίηση της υπηρεσίας σύμφωνα με το XML αρχείο ρυθμίσεων.

```
GreetingServiceImpl greetingService =
    new GreetingServiceImpl();
greetingService.setGreeting("Buenos Dias!");
```

Αυτό που μένει είναι η δημιουργία μίας κλάσης η οποία να φορτώνει τον πυρήνα της Spring και να τον χρησιμοποιεί για την εκτέλεση της υπηρεσίας.

```
import org . springframework . beans . factory . BeanFactory ;

import org . springframework . beans . factory . xml . XmlBeanFactory ;
import org . springframework . core . i o . ClassPathResource ;

public class HelloApp {

    public static void main ( String [ ] args ) throws Exception {
        BeanFactory factory =
                new  XmlBeanFactory (new ClassPathResource ( " hello . xml" ) ) ;

        GreetingService greetingService =
              ( GreetingService ) factory . getBean ( " greetingService " ) ;

        greetingService . sayGreeting ( ) ;
    }
}
```

Η κλάση BeanFactory που χρησιμοποιείται παραπάνω αποτελεί τον πυρήνα της Spring. Μετά τη φόρτωση του hello.xml αρχείου από τον πυρήνα η main() καλεί την μέθοδο getBean(), ώστε να ανακτήσει τις εξαρτήσεις της υπηρεσίας. 'Εχοντας αυτές τις εξαρτήσεις καλείται τελικά η μέθοδος sayGreeting().΄Οταν εκτελεστεί η εφαρμογή, όπως αναμενόταν, τυπώνεται το:

Buenos Dias!

Η παραπάνω εφαρμογή αποτελεί το πιο απλό παράδειγμα χρήσης της Spring.

Παρά την απλότητά του, παρουσιάζει το βασικό τρόπο ρύθμισης και χρήσης κλάσεων μέσω της Spring. Όμως, λόγω της απλότητάς του, δε δείχνει πως μπορεί να ρυθμιστεί κάποιο bean, ώστε να περαστεί η τιμή ενός String στο πεδίο (injection). Η πραγματική δύναμη της Spring έγκειται στο πως μπορούν τα διάφορα beans να συνδεθούν με άλλα beans κάνοντας χρήση του DI.

2.2 Mysql

Η MySQL είναι το πιο δημοφιλές, open-source σχεσιακό σύστημα διαχείρισης βάσεων δεδομένων. Το όνομα προέκυψε από το όνομα της κόρης του συν-ιδρυτή Michael Widenius, My. Το ακρωνύμιο SQL προέρχεται από το Structured Query Language (Γλώσσα Δομημένων Αιτημάτων).

Η MySQL είναι κατά κύριο λόγο ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων (RDBMS) και παρέχεται χωρίς γραφικά εργαλεία για τη διαχείριση των βάσεων δεδομένων, ή των δεδομένων που βρίσκονται μέσα σε αυτές. Αντιθέτως η αλληλεπίδραση γίνεται με τη χρήση γραμμής εντολών. Ωστόσο υπάρχουν και γραφικά εργαλεία τα οποία μπορούν να χρησιμοποιήσουν οι χρήστες και τα οποία παρέχουν μεταξύ άλλων, δυνατότητες δημιουργίας και επεξεργασίας βάσεων δεδομένων, δημιουργία αντιγράφων ασφαλείας και έλεγχο κατάστασης βάσεων. Το επίσημο σετ γραφικών εργαλείων, το MySQL Workbench έχει αναπτυχθεί από την Oracle και είναι διαθέσιμο δωρεάν. Άλλα προγράμματα που παρέχουν γραφική

υποστήριξη είναι το phpMyAdmin, Adminer, OpenOffice.org Base και το Sequel Pro που απευθύνεται σε χρήστες λειτουργικού συστήματος Mac OS X.

Η MySQL είναι γραμμένη σε C και C++. Ο αναλυτής της SQL είναι γραμμένος σε yacc και ένα λεξιλογικό αναλυτή που έχει αναπτύξει η ίδια η εταιρεία. Πολλές γλώσσες προγραμματισμού χρησιμοποιούν βιβλιοθήκες για να αποκτήσουν πρόσβαση σε βάσεις δεδομένων της MySQL.

2.3 Android

Το Android είναι ένα λειτουργικό σύστημα για κινητές συσκευές, όπως τα smartphones και τα tablets και είναι βασισμένο στα Linux. Όλες οι λειτουργίες του κινητού τηλεφώνου ελέγχονται από τις εφαρμογές (apps), τις οποίες ο χρήστης μπορεί να κατεβάσει και να εγκαταστήσει ελεύθερα.

Αξίζει να σημειωθεί ότι το Android είναι ένα ελεύθερο λογισμικό ανοικτού κώδικα (free and open source software) πράγμα που επιτρέπει στους χρήστες την ανάπτυξη δωρεάν παιχνιδιών και κάθε είδους εφαρμογής. Αυτή ακριβώς η δυνατότητα συντέλεσε στην ταχεία διάδοση του. Επιπλέον, τα εργαλεία για να αναπτύξει εφαρμογές οποιοσδήποτε προγραμματιστής μπορεί να τα βρει δωρεάν στο Internet.

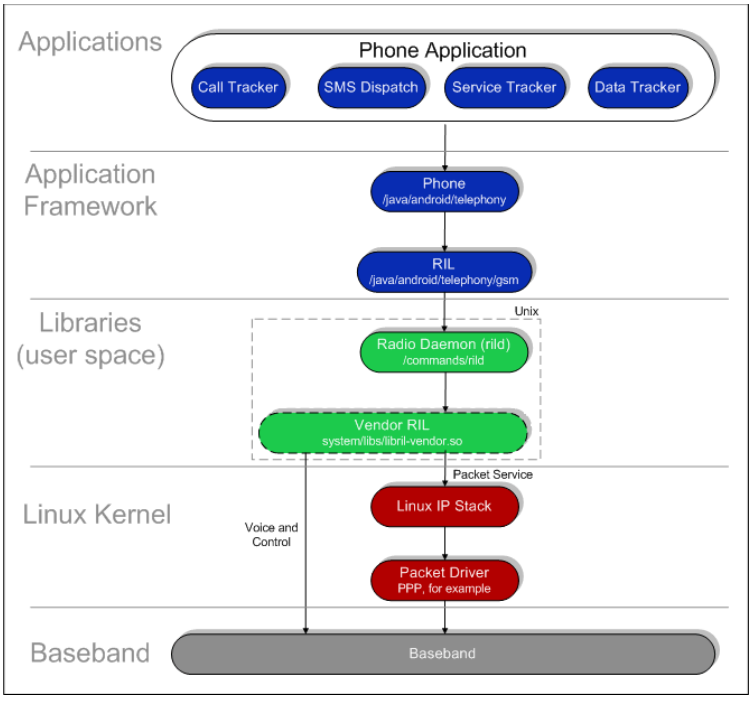
Το λειτουργικό σύστημα Android αναπτύχθηκε από την εταιρεία Android Inc. που εξαγοράστηκε πλήρως από την Google τον Αύγουστο του 2005.

Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας την συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων από την Google. Δύο χρόνια αργότερα, τον Νοέμβριο του 2007, έκανε την εμφάνισή της η Open Handset Alliance, μια κοινοπραξία 48 τηλεπικοινωνιακών εταιρειών, εταιρειών λογισμικού και κατασκευής hardware, ανάμεσα τους ονόματα όπως LG, Intel, HTC, Motorola και NVidia. Στόχος της ήταν η δημιουργία ανοικτών προτύπων για κινητές συσκευές. Παράλληλα, δημοσίευσαν το πρώτο τους προϊόν, το Android, μια πλατφόρμα βασισμένη στον πυρήνα του Linux (Linux kernel) 2.6. Η Google έδωσε στην δημοσιότητα και το μεγαλύτερο μέρος του πηγαίου κώδικα του Android υπό τους όρους της Apache License.

III. Προγραμματισμός σε περιβάλλον Android

3.1 Σχεδιασμός

Το Android αποτελείται από έναν kernel βασισμένο σε αυτόν του Linux με το middleware, τις βιβλιοθήκες και τα APIs να είναι γραμμένα σε C και το software των εφαρμογών που τρέχει πάνω σε ένα πλαίσιο (applications framework) που περιλαμβάνει βιβλιοθήκες συμβατές με την Java. Το Android χρησιμοποιεί την εικονική μηχανή Dalvik (Dalvik virtual machine). Η κύρια πλατφόρμα του hardware είναι η ARM αρχιτεκτονική που χρησιμοποιείται σε ευρέως σε 32-bit συστήματα.



Εικόνα 3.1: Διάγραμμα Αρχιτεκτονικής

3.2 Linux

Η αρχιτεκτονική του Linux είναι βασισμένη στις αρχές του λειτουργικού Unix αλλά έχει αναπτυχθεί εκ του μηδενός και δεν περιλαμβάνει κώδικα από το Unix. Η ανάπτυξη του Linux είναι χαρακτηριστικό παράδειγμα εθελοντικής συνεργασίας από διαδικτυακές κοινότητες, ενώ όλο το έργο είναι ανοικτού κώδικα και ελεύθερα προσβάσιμο από όλους για αντιγραφή, τροποποίηση ή αναδιανομή χωρίς περιορισμό. Το Linux είναι διαθέσιμο υπό άδειες όπως η GNU General Public License.

Δημιουργός του πυρήνα Linux είναι ο Linus Torvalds, από το όνομα του οποίου προήλθε και η ονομασία Linux. Ο Torvalds άρχισε να αναπτύσσει έναν kernel το 1991 εμπνευσμένος από το λειτουργικό MINIX και χρησιμοποιώντας πολλά προγράμματα και βιβλιοθήκες από το GNU του Richard Stallman. Πάνω στον αρχικό πυρήνα του Torvalds έχουν εργαστεί χιλιάδες χρήστες αλλά και εταιρείες. Λόγω των στενότατων σχέσεων μεταξύ Linux και GNU, πολλές φορές το σύστημα αυτό αναφέρεται ως GNU/Linux, ονομασία που είναι πιο ακριβής και την προτιμά και το Ίδρυμα Ελεύθερου Λογισμικού.

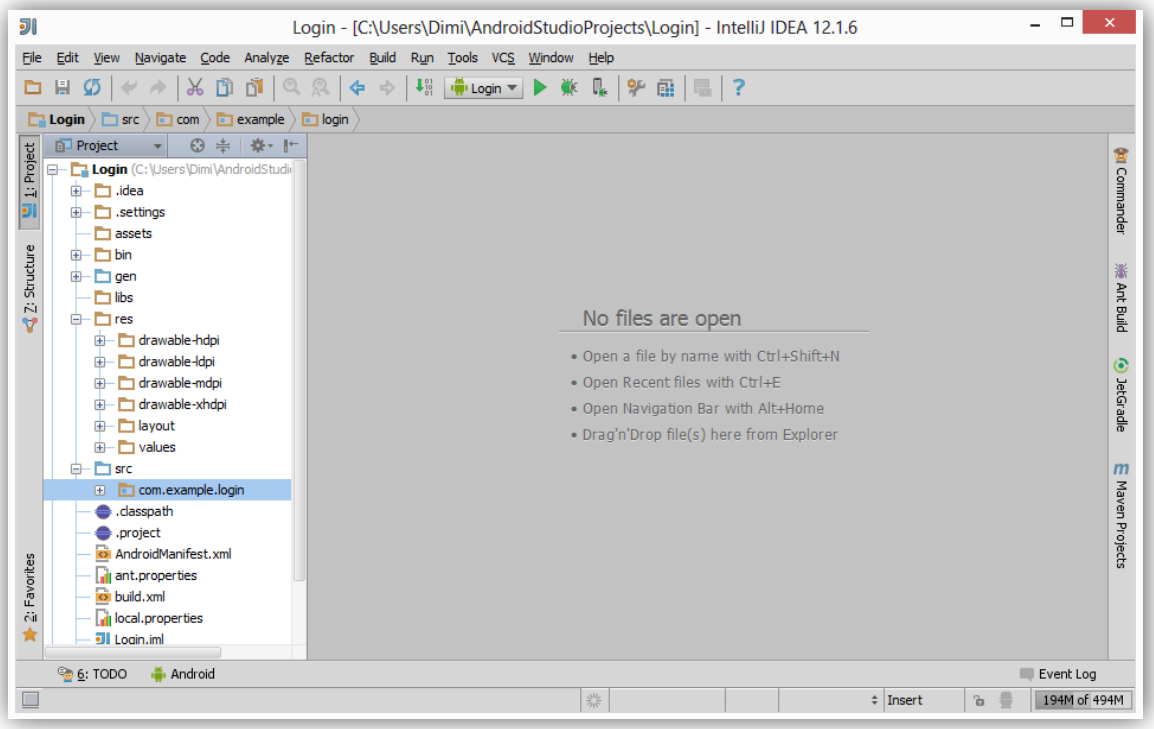
Ο kernel του Android είναι βασισμένος σε αυτόν του Linux αλλά έχει αρκετές αρχιτεκτονικές διαφορές κυρίως χάρη στην Google. Δεν υποστηρίζει το πλήρες σύνολο των καθιερωμένων βιβλιοθηκών GNU πράγμα που καθιστά δύσκολη την μεταφορά εφαρμογών και βιβλιοθηκών από το Linux στο Android. Ορισμένα χαρακτηριστικά που η Google έδωσε πίσω στον Linux kernel, ένα σύστημα διαχείρισης ενέργειας που ονομάζεται wake lock, απορρίφτηκε από τους προγραμματιστές του κυρίως kernel επειδή φάνηκε ότι δεν έδινε και πολλή σημασία στην δικιά τους δουλειά. Σήμερα πολλοί προγραμματιστές εργάζονται για τη τροποποίηση των υφιστάμενων kernel και ROM, για να δημιουργήσουν νέα που θα είναι συμβατά με το νέο λειτουργικό σύστημα Android. Αυτές οι προσπάθειες για το Android γίνονται συνήθως μέσω του XDA-developers και του androidforums.com.

3.3 Προγραμματιστικά Εργαλεία

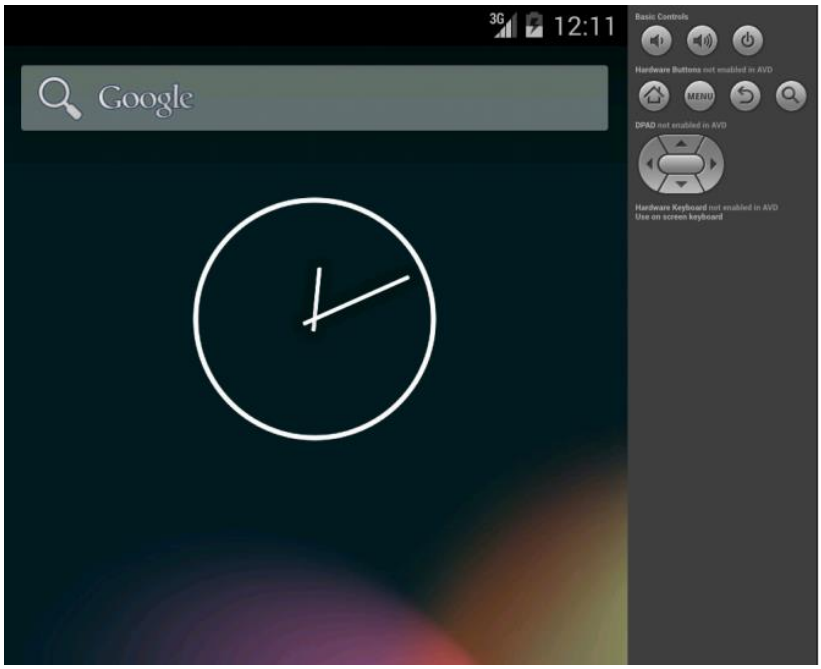
Η κύρια γλώσσα προγραμματισμού που χρησιμοποιείται για την ανάπτυξη εφαρμογών Android είναι η Java, ενώ τους τελευταίους μήνες γίνονται προσπάθειες για να συμπεριληφθούν και άλλες γλώσσες όπως η C και η C++. Οπότε βασική προϋπόθεση είναι να διαθέτουμε τα αντίστοιχα εργαλεία της γλώσσας προγραμματισμού και συγκεκριμένα το Java Development Kit (JDK). Ακόμη χρειαζόμαστε ένα ολοκληρωμένο περιβάλλον ανάπτυξης (integrated development environment ή IDE) για να μεταγλωττίσουμε και να τρέξουμε τα προγράμματα μας, όπως είναι για παράδειγμα το Intellij (Εικόνα 3.2).
Το βασικότερο εργαλείο αποτελεί το Android SDK (software development kit) το οποίο ουσιαστικά μας παρέχει τα επιπλέον εργαλεία ώστε να μπορούμε να γράψουμε κώδικα για την κατασκευή μιας εφαρμογής Android. Η σύνδεση του Android SDK με το γραφικό μας περιβάλλον (Intellij) γίνεται μέσω μιας επέκτασης (Android Development Tools ή ADT Plug-in) που κάνουμε εγκατάσταση στο Intellij, ώστε να μπορέσουμε να μεταγλωττίσουμε την εφαρμογή μας και έπειτα να την τρέξουμε.

Καθώς το λειτουργικό σύστημα Android κυκλοφορεί σε διάφορες εκδόσεις, όπως αναφέραμε παραπάνω, καθίσταται σαφές ότι η κάθε έκδοση θα χρησιμοποιεί και διαφορετικά προγραμματιστικά εργαλεία. Έτσι μέσα από το ADT Plug-in μπορούμε να εγκαταστήσουμε τα εργαλεία καθώς και την τεκμηρίωση (documentation) αλλά και διάφορα παραδείγματα για την υλοποίηση της εφαρμογής σε οποιαδήποτε έκδοση. Για παράδειγμα, αν θέλουμε η εφαρμογή μας να είναι συμβατή και σε παλαιότερες εκδόσεις του Android τότε θα πρέπει , μαζί με άλλες αλλαγές που θα παρουσιάσουμε στην συνέχεια της διπλωματικής (βλ. Android Manifest), να εγκαταστήσουμε και τα αντίστοιχα εργαλεία και να δοκιμάσουμε την εφαρμογή μας σε αυτές.

Τέλος, για να δοκιμάσουμε την εφαρμογή μας και να δούμε τα αποτελέσματα της θα χρειαστούμε κάποιον προσομοιωτή κινητού τηλεφώνου, αν δεν διαθέτουμε εμείς οι ίδιοι, στον υπολογιστή μας. Τη λύση μας δίνει μια εικονική συσκευή Android (Android Virtual Device ή AVD) η οποία ουσιαστικά αποτελεί προσομοιωτή τόσο software όσο και hardware ενός κινητού τηλεφώνου με λειτουργικό σύστημα Android (Εικόνα 3.3). Την συσκευή αυτή την εγκαθιστάμε μέσα από το ADT Plug-in, από όπου μπορούμε να ρυθμίσουμε πολλές παραμέτρους της, όπως τι έκδοση Android θα χρησιμοποιεί, το μέγεθος της οθόνης, το μέγεθος της εξωτερικής κάρτας αποθήκευσης και της cache, καθώς και άλλα χαρακτηριστικά που έχουν να κάνουν με την τοποθεσία (GPS), την δυνατότητα λήψης φωτογραφιών, κ.α.
Απαιτούνται αρκετοί υπολογιστικοί πόροι και μεγάλη έκταση μνήμης RAM για την εκτέλεση της AVD, πράγμα που καθιστά τις περισσότερες φορές αργή την εκτέλεση της εφαρμογής μας στη συσκευή αυτή. Φυσικά, κάθε φορά μπορούμε να εγκαθιστούμε και να ελέγχουμε τις εφαρμογές μας σε φυσική συσκευή, όπως ένα κινητό τηλέφωνο ή ένα tablet που χρησιμοποιούν λειτουργικό σύστημα Android, συνδέοντας το απλά σε μια θύρα USB του υπολογιστή μας.



Εικόνα 3.2: Περιβάλλον Ανάπτυξης λογισμικού Intellij



Εικόνα 3.3: Android Virtual Device

IV. Δομή και Αρχιτεκτονική

Στο κεφάλαιο που ακολουθεί αρχικά θα περιγράψουμε την εφαρμογή που αναπτύχθηκε για το λειτουργικό σύστημα Android, αναφέροντας όλες τις δυνατότητες που προσφέρει η εφαρμογή στο χρήστη. Έπειτα θα αναλύσουμε τη δομή της, τα αρχεία δηλαδή στα οποία οργανώνεται, καθώς και τη χρήση του καθενός.

4.1 Περιγραφή

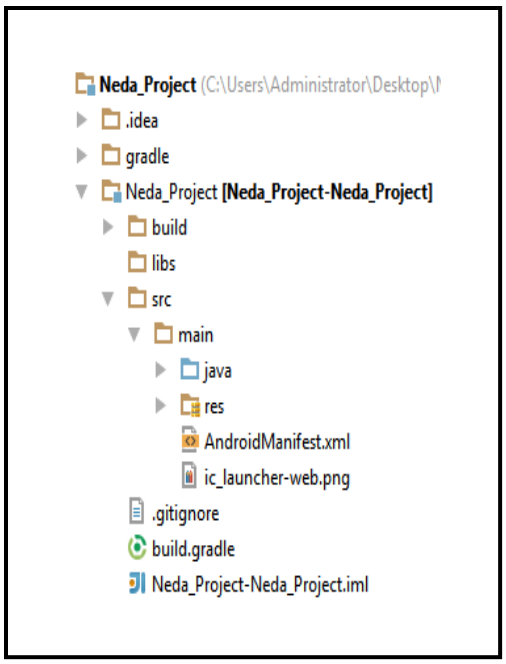
Η διπλωματική εργασία περιστρέφεται γύρω από το πληροφοριακό σύστημα διαχείρισης ερευνητικών έργων ΝΕΔΑ. Πρόκειται για ένα σύστημα σχεδιασμένο να καλύψει τις απαιτήσεις του Ερευνητικό Ακαδημαϊκό Ινστιτούτο τεχνολογίας Υπολογιστών & Εκδόσεων (Ι.Τ.Υ.Ε.) και βρίσκεται σε χρήση από το 2009.
Το παρόν σύστημα πρόκειται για ένα παράδειγμα πληροφοριακού συστήματος σχεδιασμένο με όλα τα πλέον σύγχρονα εργαλεία ανάπτυξης κώδικα. Είναι γραμμένο σε Java και κάνει ευρεία χρήση του περιβάλλοντος εργασίας Spring και τα modules που προσφέρει αυτό το περιβάλλον για να επεξεργάζεται τα αιτήματα των χρηστών του. Χρησιμοποιεί το περιβάλλον προγραμματισμού Android για το επίπεδο παρουσίασης της εφαρμογής στον χρήστη και το περιβάλλον εργασίας MySQL για την επικοινωνία του συστήματος με το σύστημα βάσεων δεδομένων που χρησιμοποιεί.

Κύριος σκοπός της εφαρμογής είναι η διαχείριση των ταξιδιών των ερευνητών του Ι.Τ.Υ.Ε. Οι επιλογές που έχει ο χρήστης είναι η αποστολή των στοιχείων

ενός ταξιδιού που πρόκειται να πραγματοποιήσει καθώς και ο έλεγχος του ιστορικού των ταξιδιών. Έτσι όταν ο χρήστης συνδεθεί με τα προσωπικά του στοιχεία στον λογαριασμό του θα εμφανίζονται στην οθόνη οι επιλογές "πρόσθεση νέου ταξιδιού" και "ιστορικό ταξιδιών".

4.2 Δομή

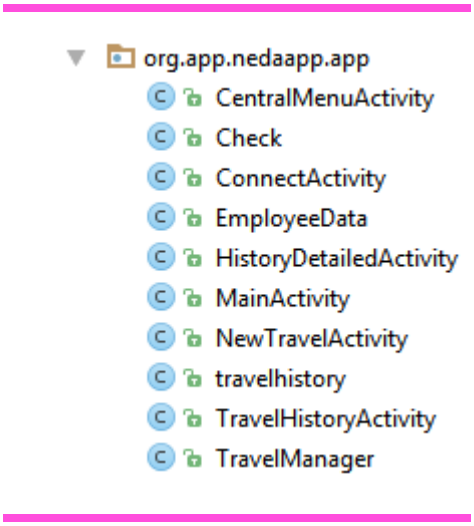
Πριν προχωρήσουμε στη λεπτομερή ανάλυση των διαφόρων λειτουργιών της εφαρμογής που αναπτύχθηκε, θα ήταν χρήσιμο να παρουσιάσουμε πρώτα τη δομή της, πώς δηλαδή οργανώνονται όλα τα αρχεία που περιέχονται σε αυτήν και ποια είναι η κύρια λειτουργία τους. Στο προγραμματιστικό περιβάλλον Intellij στο οποίο αναπτύχθηκε η εφαρμογή, το project εμφανίζεται με την παρακάτω δομή:



Εικόνα 4.1: Δομή της εφαρμογής

Ακολουθεί η επεξήγηση των παραπάνω φακέλων και αρχείων καθώς και του περιεχομένου τους:

- **src/**

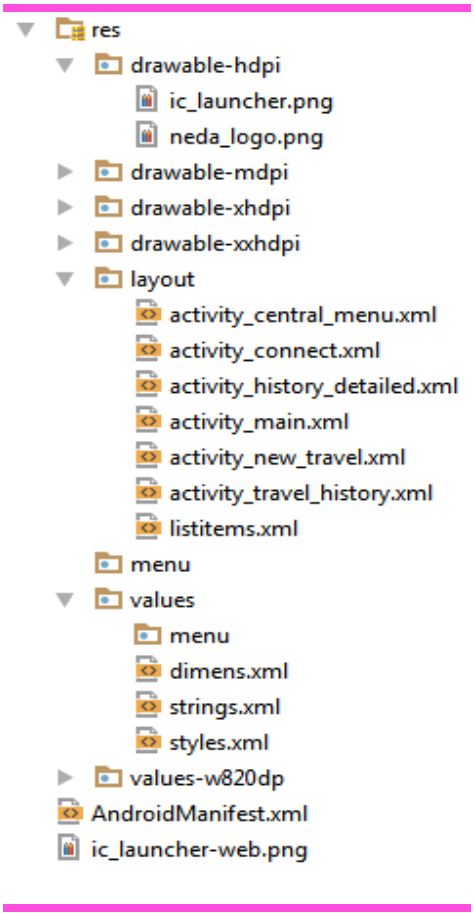


Εικόνα 4.2**:** Περιεχόμενα φακέλου src/

Ο φάκελος αυτός αρχικά περιέχει τα πακέτα τα οποία έχουμε δηλώσει στην εφαρμογή μας. Δηλαδή έχουμε μόνο ένα πακέτο, το "org.app.nedaapp.app" οπότε εμφανίζεται αυτό. Έπειτα στο πακέτο αυτό περιέχονται όλες οι κλάσεις της εφαρμογής μας. Οι κλάσεις αυτές χωρίζονται στις activities, δηλαδή στις διαφορετικές οθόνες της εφαρμογής καθώς και για την σύνδεση με το web service και την βάση δεδομένων.

- **res/**

Στο φάκελο res/ αποθηκεύονται όλα τα application resources, τα οποία χρησιμοποιούμε. Συγκεκριμένα, στο φάκελο drawable αποθηκεύουμε όλα τα αρχεία εικόνας που χρησιμοποιούνται στην εφαρμογή. Στο φάκελο layout αποθηκεύονται τα αρχεία xml στα οποία δηλώνουμε το user interface της κάθε οθόνης. Ο φάκελος values περιλαμβάνει το αρχείο strings.xml με όλα τα strings της εφαρμογής δηλωμένα.



Εικόνα 4.3: Όλα τα resources της εφαρμογής

Παρατηρούμε επίσης την δημιουργία του αρχείου AndroidManifest.xml, το οποίο δημιουργείται αυτόματα όταν ξεκινάμε καινούργιο project για μια

εφαρμογή Android. Το αρχείο αυτό περιέχει βασικές πληροφορίες σχετικά με την εφαρμογή, τις οποίες το λειτουργικό σύστημα πρέπει να γνωρίζει προτού τρέξει οποιδήποτε άλλο κώδικα. Οι σημαντικότες από αυτές τις πληροφορίες είναι οι εξής:

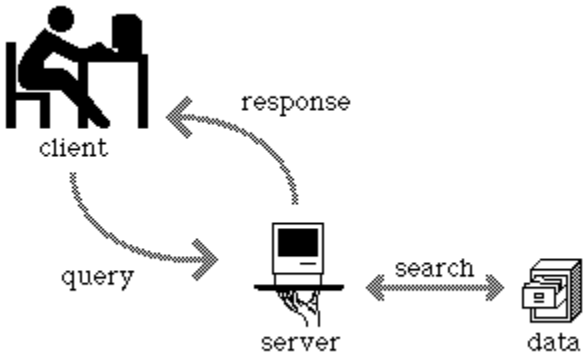
- Η ονομασία του πακέτου Java της εφαρμογής.
- Η έκδοση της εφαρμογής
- Η ελάχιστη έκδοση του λειτουργικού συστήματος Android που απαιτεί η εφαρμογή (min sdk version). Για παράδειγμα αν έχει δηλωθεί min sdk version ο αριθμός 19, που ισοδυναμεί με την έκδοση Android 4.4.2, τότε η εφαρμογή θα εκτελεστεί σε συσκευές με έκδοση Android μεγαλύτερη ή ίση της 4.4.2
- Το όνομα της εφαρμογής καθώς και το εικονίδιό της.
- Οι άδειες που απαιτούνται για να εκτελεστούν ορισμένες λειτουργίες της εφαρμογής.
- Όλα τα συστατικά στοιχεία της εφαρμογής. Για παράδειγμα, αν δημιουργήσουμε κάποια κλάση για activity της εφαρμογής, χωρίς να το δηλώσουμε στο AndroidManifest.xml, δε θα λειτουργήσει.
- Οι εξωτερικές βιβλιοθήκες που χρησιμοποιεί η εφαρμογή.

V. Σύστημα Διαχείρισης Ταξιδιών

Σε αυτό το κεφάλαιο θα αναλύσουμε το σύστημα διαχείρισης ταξιδιών που πραγματοποιούνται από τους εργαζόμενους του οργανισμού Ι.Τ.Υ.Ε. Ο οργανισμός αυτός ασχολείται με ερευνητικά έργα και συνεργάζεται με άλλους οργανισμούς τόσο του εσωτερικού όσο και του εξωτερικού. Οι εργαζόμενοι συχνά καλούνται να ταξιδέψουν ώστε να έρθουν σε επαφή με συνεργάτες, είτε για την παρακολούθηση συνεδρίων ή για την παρουσίαση. Για να πάει κάποιος ταξίδι χρειάζεται να κάνει αίτηση με τα στοιχεία του ταξιδιού και στην συνέχεια ο διαχειριστής να ελέγξει την αίτηση και να την εγκρίνει ή να την απορρίψει. Σε αυτό το κεφάλαιο θα δούμε τις οντότητες που χρησιμοποιούνται στο σύστημα και τον τρόπο που γίνεται η αίτηση του ταξιδιού και η έγκρισή του ή μη.

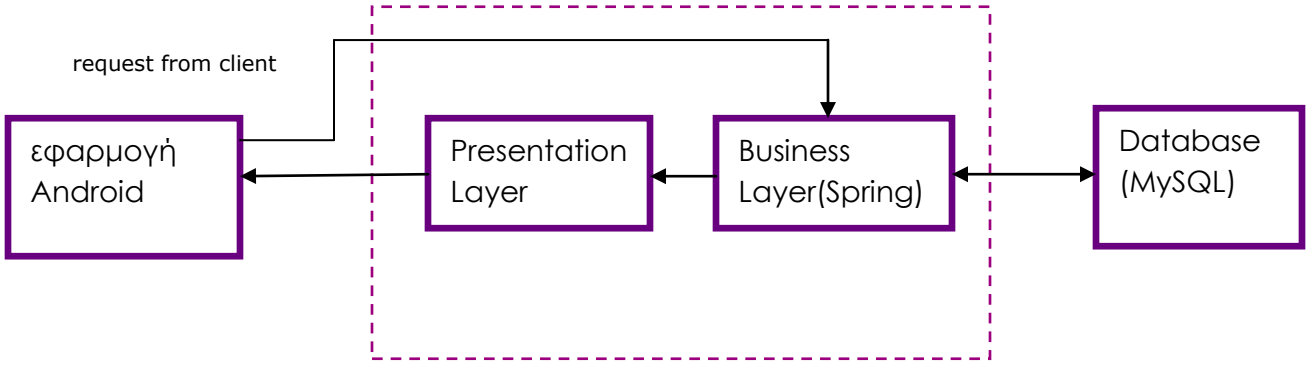
5.1 Δομή client - server model

Η ΝΕΔΑ είναι ένα κατανεμημένο σύστημα το οποίο βασίζεται στο μοντέλο πελάτη-εξυπηρετητή (client - server model).



Το σύστημα αποτελείται από τα εξής διαφορετικά επίπεδα, μία βάση δεδομένων για την αποθήκευση της πληροφορίας, το επίπεδο επικοινωνίας με την βάση δεδομένων, το επίπεδο λειτουργιών (business layer) το οποίο αναλαμβάνει την εξυπηρέτηση των αιτημάτων που πραγματοποιούν οι χρήστες,δηλαδή υλοποιεί τα βασικά ερωτήματα που χρειάζονται για την επικοινωνία με τη βάση, ερωτήματα ανάγνωσης δεδομένων στη βάση, εγγραφής και ανανέωσης. Στην συνέχεια είναι το επίπεδο παρουσίασης το οποίο τροφοδοτείται από το επίπεδο λειτουργίας με πληροφορίες τις οποίες αναλαμβάνει να τις εμφανίσει στον χρήστη σε μορφή αναγνώσιμη από τον την εφαρμογή. Στην συνέχεια υπάρχει το επίπεδο ασφαλείας το οποίο ελέγχει αν ο συγκεκριμένος χρήστης που κάνει το αίτημα στο σύστημα έχει το δικαίωμα να πραγματοποιήσει το συγκεκριμένο αίτημα. Μόνο αν έχει ο έλεγχος μεταφέρεται στα παρακάτω επίπεδα αλλιώς το αίτημα απορρίπτεται. Τέλος είναι το επίπεδο της εφαρμογής μέσω του οποίου ο χρήστης πραγματοποιεί αιτήματα.

Η διαδικασία λειτουργίας του συστήματος φαίνεται στην εικόνα 5.1. Ο χρήστης στέλνει ένα αίτημα, για παράδειγμα προβολή ενός ταξιδιού. Το σύστημα μόλις λάβει το αίτημα θα εξετάσει αν ο συγκεκριμένος χρήστης έχει δικαίωμα για την συγκεκριμένη πράξη. Αν ναι, τότε ο αρμόδιος για αυτό το αίτημα Controller της Spring θα εκτελέσει τις εντολές που χρειάζονται για να εξυπηρετηθεί το αίτημα. Αν η εξυπηρέτηση του αιτήματος χρειάζεται συναλλαγές με τη βάση, στο παράδειγμά μας θα χρειαστεί να ανακτήσει από την βάση δεδομένων τα στοιχεία του ταξιδιού, τότε ο Controller είναι υπεύθυνος για την επικοινωνία με την βάση δεδομένων. Στην συνέχεια ο έλεγχος γυρίζει στον Controller ο οποίος δίνει τις πληροφορίες στο επίπεδο παρουσίασης που χρειάζεται για να επιστρέψει τα στοιχεία στον χρήστη. Στο επίπεδο προβολής επίσης τα αποτελέσματα μπορούν να υποστούν φιλτράρισμα αναλόγως τα δικαιώματα που έχει ο χρήστης πάνω στο συγκεκριμένο αίτημα. Για παράδειγμα διαφορετική σελίδα γυρίζει στον χρήστη που ζητάει να δει τα στοιχεία ενός ταξιδιού του και διαφορετική σελίδα επιστρέφει στον διαχειριστεί που βλέπει τα στοιχεία του αιτήματος για να το εγκρίνει ή μη. Στην συνέχεια το αποτέλεσμα επιστρέφει σε μορφή αναγνώσιμη από την εφαρμογή στον χρήστη.



Εικόνα 5.1: Δομή συστήματος

5.2 Επιπεδο Database

Η σχεσιακή βάση δεδομένων που χρησιμοποιείται είναι η MySQL στην έκδοση 6.0. Στην συνέχεια ακολουθούν οι οντότητες της σχεσιακής βάσης όπως σχεδιάστηκαν για να καλύψουν τις ανάγκες του συστήματος.

Travel

Το ταξίδι είναι η βασική οντότητα του ταξιδιού και συγκεντρώνει τις βασικές πληροφορίες για ένα ταξίδι, οι ιδιότητες που έχει είναι οι εξής:

- **TravelID:** είναι ο κωδικός που χρησιμοποιείται από την βάση δεδομένων.
- **EmployeeID:** είναι ο κωδικός του εργαζόμενου που κάνει την συγκεκριμένη αίτηση.

- **ApprovalID:** είναι ο κωδικός του ταξιδιού που εγκρίθηκε ή απορρίφθηκε.

- **TravelPurpose:** είναι το κείμενο που ο χρήστης περιγράφει τον σκοπό του ταξιδιού. Στην βάση δεδομένων αποθηκεύεται ως varchar με μέγιστο όριο 500 χαρακτήρες.

- **DepartureCountry:** είναι η χώρα αφετηρίας του ταξιδιού.

- **DepartureCity:** είναι η πόλη αφετηρίας του ταξιδιού.

- **DestinationCountry**: είναι η χώρα προορισμού του ταξιδιού.

- **DestinationCity:** είναι η πόλη προορισμού του ταξιδιού.

- **DepartureDate:** Κάθε ταξίδι έχει ημερομηνία αναχώρησης, πρόκειται για την μέρα που ο ταξιδιώτης ξεκινάει το ταξίδι. Στην βάση αποθηκεύεται ως date.

- **ReturnDate:** Κάθε ταξίδι έχει και ημερομηνία επιστροφής, πρόκειτα για την ημερομηνία που ο ταξιδιώτης επιστρέφει. Όμοια με την ημερομηνία αναχώρησης στη βάση αποθηκεύεται ως date.

- **RegistrationFees:** είναι τα έξοδα εγγραφής εάν υπάρχουν.

- **AccommodationExpenses:** είναι τα έξοδα του ξενοδοχείου που θα μείνει ή κάποιας άλλης στέγασης.

- **TransportExpenses:** είναι όλα τα έξοδα μετακίνησης είτε με αεροπλάνο, πλοίο, λεωφορείο, ταξί, τρένο κτλ.

- **PersonalExpenses:** αναφέρεται στα προσωπικά του έξοδα κατά την διάρκεια του ταξιδιού.

Employees

Το employees είναι τα στοιχεία των εργαζομένων στον οργανισμό. Αναλυτικά:

- **EmployeeID:** είναι το αναγνωριστικό του κάθε εργαζόμενου που χρησινομοιείται από την βάση δεδομένων.
- **Surname:** είναι το επώνυμο του εργαζόμενου.
- **Name:** είναι το όνομα του εργαζόμενου.
- **Password:** είναι ο κωδικός του εργαζόμενου με το οποίο εισέρχεται το σύστημα.
- **Mail:** είναι το mail του εργαζόμενου με το οποίο εισέρχεται το σύστημα.
- **Phone:** είναι το τηλέφωνο του εργαζόμενου.
- **Address:** είναι η διεύθυνση επικοινωνίας του εργαζόμενου.
- **SectorID:** είναι το αναγνωριστικό της οντότητας sector που δηλώνει σε ποιον τομέα ανήκει ο εργαζόμενος.

Sector

Η οντότητα αυτή αναφέρεται στον τομέα που ανήκει ο εργαζόμενος καθώς και τον υπεύθυνο του κάθε τομέα.

- **SectorID:** είναι το αναγνωριστικό που χρησιμοποιείται από την βάση δεδομένων.
- **SectorName:** είναι το όνομα του τομέα στο οποίο εργάζεται ο χρήστης.
- **SectorDirector:** είναι ο υπεύθυνος του συγκεκριμένου τομέα.

Approval

Αναφέρεται στην έγκριση ή μη μιας αίτησης ταξιδιού καθώς και στις ημερομηνίες που υποβλήθηκε και που εγκρίθηκε ή απορρίφθηκε.

- **ApprovalID:** είναι το αναγνωριστικό που χρησιμοποιείται από την βάση δεδομένων.
- **DateOfRequest:** είναι η ημερομηνία που ο χρήστης υπέβαλε αίτηση ταξιδιού ως προς έγκριση.
- **IsApproved:** πέρνει δύο τιμές είτε εγκρίθηκε ή απορρίφθηκε.
- **DateOfApproval:** δηλώνει την ημερομηνία που ο διαχειριστής αποφάσισε για την τύχη του ταξιδιού.

5.3 Επίπεδο λειτουργιών – business layer

Είναι το επίπεδο στο οποίο αναπτύσσεται η λογική για την εξυπηρέτηση κάθε αιτήματος από τους χρήστες. Συγκεκριμένα για αυτό το επίπεδο χρησιμοποιείται η Spring. Όταν ένα αίτημα καταφθάνει αυτό περνάει μέσα από τον Dispacher Servlet, το αίτημα χαρακτηρίζεται από το URL και τις παραμέτρους. Ο dispatcher βρίσκει τον κατάλληλο Controller ο οποίος περιέχει την λογική για την εξυπηρέτηση του αιτήματος. Επίσης σε αυτό το επίπεδο υπάρχουν και οι διαχειριστές, κλάσεις οι οποίες υλοποιούν όλα τα ερωτήματα που χρειάζεται το σύστημα για να επικοινωνεί με την βάση.
Χρησιμοποιώντας την Spring ο τρόπος με τον οποίο εξυπηρετούνται τα αιτήματα των χρηστών είναι ξεκάθαρος και οργανωμένος, και κάνει την επέκταση ενός τέτοιου συστήματος αρκετά απλή υπόθεση αφήνοντας τον προγραμματιστή να ασχοληθεί αποκλειστικά με το πρόβλημα της εξυπηρέτησης ενός αιτήματος παρά με την συνοχή της αρχιτεκτονικής του συστήματος γενικότερα.

5.4 Επίπεδο παρουσίασης – presentation layer

Πρόκειται για το επίπεδο το οποίο αναλαμβάνει να παρουσιάσει το αποτέλεσμα ενός αιτήματος στον χρήστη. Αυτό το επίπεδο χρησιμοποιεί το module SpringMVC. Το αίτημα περνάει μέσα από τον Servlet, στο οποίο έχουμε ορίσει τους Controllers που θα εξυπηρετήσουν τα αντίστοιχα αιτήματα. Μόλις ο Controller ολοκληρώσει την λειτουργία του, επιστρέφει πίσω στον Servlet ένα View μαζί με τα δεδομένα της επεξεργασίας. Στην

συνέχεια μαζί με τα δεδομένα που έδωσε ο Controller συνθέτουν το αποτέλεσμα που βλέπει ο χρήστης στην οθόνη του.

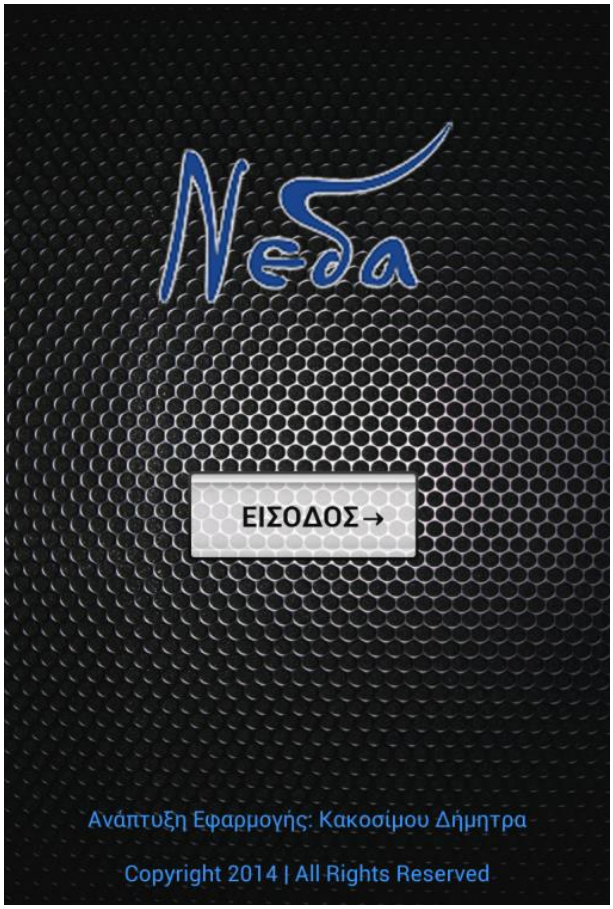
VI. Βασικές λειτουργίες εφαρμογής

6.1 Εισαγωγή

Αφού παρουσιάστηκε η δομή της εφαρμογής, μπορούμε να παρουσιάσουμε αναλυτικά όλες τις λειτουργίες που περιλαμβάνει η εφαρμογή που αναπτύχθηκε, καθώς και τον τρόπο υλοποίησης αυτών στο προγραμματιστικό περιβάλλον. Στο κεφάλαιο αυτό, λοιπόν, θα περιγράψουμε όλες τις διαφορετικές οθόνες της εφαρμογής, ο τρόπος σύνδεσης μεταξύ τους, οι κλάσεις στις οποίες υλοποιούνται, και όλες οι δυνατές επιλογές που έχει ο χρήστης όταν βρίσκεται σε συγκεκριμένη οθόνη. Τα διάφορα στιγμιότυπα της εφαρμογής παρουσιάζονται σε εικόνες και έπειτα εξηγούνται οι διάφορες λειτουργίες της, για να διευκολύνεται ο αναγνώστης στην κατανόηση των λειτουργιών αυτών.

6.2 Αρχική Οθόνη

Όταν εκτελεστεί η εφαρμογή εμφανίζεται στον χρήστη μία αρχική οθόνη στην οποία κάνει είσοδο στην εφαρμογή. Η οθόνη αυτή φαίνεται στην παρακάτω εικόνα:

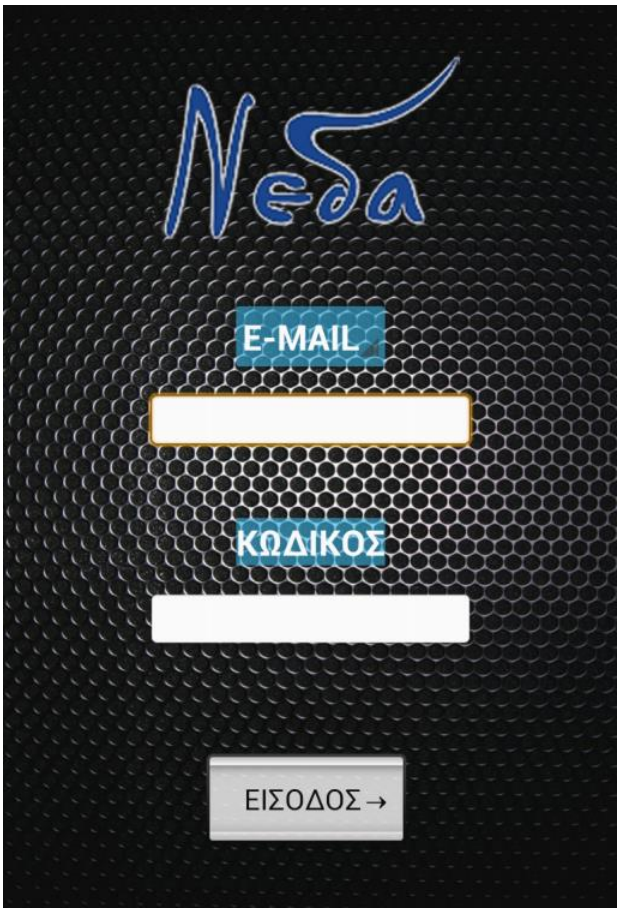


Εικόνα 6.1: Αρχική οθόνη εφαρμογής

Πρόκειται για την κλάση MainActivity.java την οποία έχουμε ορίσει ως κύρια activity στο AndroidManifest.xml ως εξής:

```
<activity   android:name="org.app.nedaapp.app.MainActivity"
android:label="@string/app_name" >

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

 </activity>
```

Ο χρήστης αφού εισέλθει στο σύστημα, του εμαφνίζεται η οθόνη με την επιλογή να συνδεθεί ως χρήστης του συστήματος βάζοντας το email και το password του. Η οθόνη φαίνεται παρακάτω:



Εικόνα 6.2: Οθόνη login

Ο χρήστης αφού συνδεθεί στο σύστημα, του εμφανίζεται η οθόνη με τις επιλογές είτε να εισάγει νέο ταξίδι είτε να ελέγξει το ιστορικό των ταξιδιών του.

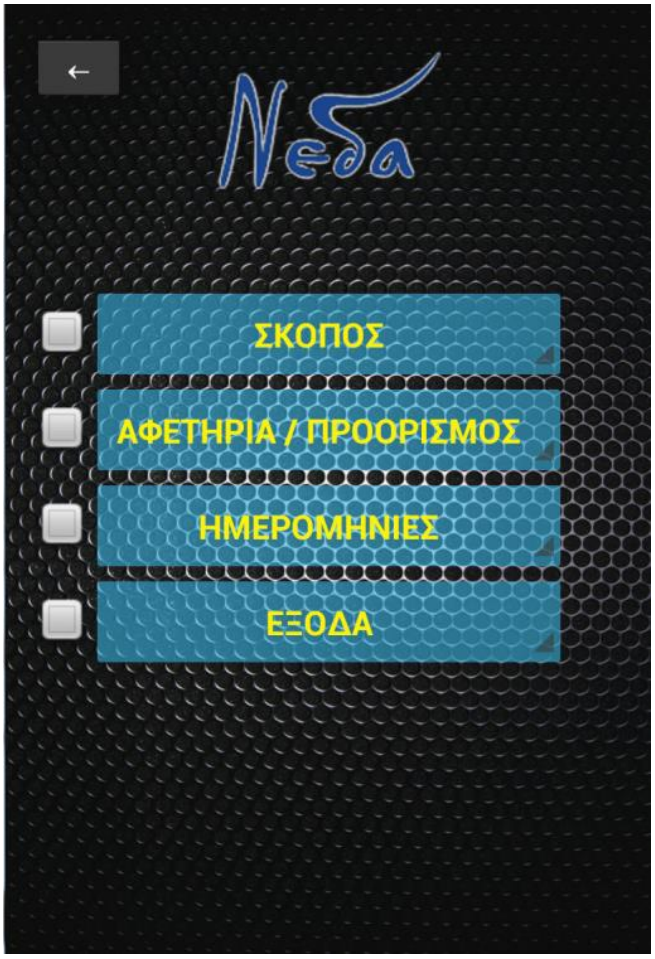
Εικόνα 6.3: Αρχική οθόνη επιλογών

Οι λειτουργίες της οθόνης αυτής ορίζονται στην activity CentralMenuActivity, ενώ το user interface της ορίζεται στο αρχείο activity_central_menu.xml.

6.3 Εισαγωγή Νέου Ταξιδιού – Add New Travel

Στην οθόνη αυτή, ο χρήστης έχει την δυνατότητα να προσθέσει τα στοιχεία για την αίτηση ενός νέου ταξιδιού. Τα στοιχεία που πρέπει να προσθέσει είναι

ο σκοπός του ταξιδιού, ο τόπος αναχώρησης και άφιξης, οι ημερομηνίες του ταξιδιού καθώς και τα έξοδά του.



Εικόνα 6.4: Οθόνη εισαγωγής στοιχεία νέου ταξιδιού

Οι λειτουργίες της οθόνης αυτής ορίζονται στην activity NewTravelActivity, ενώ το user interface της ορίζεται στο αρχείο activity_new_travel.xml.

6.3.1 Εισαγωγή Σκοπού

Στην οθόνη εισαγωγής νέου ταξιδιού, όταν ο χρήστης πατήσει το κουμπί "σκοπός" ανοίγει νεό παράθυρο με την φόρμα εισαγωγής του σκοπού του ταξιδιού, όπως φάινεται και στην εικόνα.

Εικόνα 6.5: Οθόνη εισαγωγής σκοπού

Αφού ολοκληρωθεί αυτή η ενέργεια έχει την δυνατότητα είτε να επιστρέψει στο μενού, είτε να προχωρήσει στη διαδικασία εισαγωγής στοιχείων του ταξιδιού, πατώντας το κουμπί "αφετηρία/προορισμός" πηγαίνοντας στην επόμενη ενέργεια.

6.3.2 Εισαγωγή Περιοχής

Αφού ο χρήστης εισάγει τον σκοπό του νέου ταξιδιού και πατήσει το κουμπί "αφετηρία/προορισμός" ανοίγει νεό παράθυρο με την φόρμα εισαγωγής της χώρας και πόλης που θα γίνει η αναχώρηση καθώς και αντίστοιχα του προορισμού, όπως φαίνεται στην παρακάτω εικόνα.

Εικόνα 6.6: Οθόνη εισαγωγής περιοχής

Αφού εισάγει τα απαραίτητα στοιχεία έχει την δυνατότητα είτε να επιστρέψει στο μενού, είτε να προχωρήσει στη διαδικασία εισαγωγής στοιχείων του ταξιδιού, πατώντας το κουμπί "ημερομηνίες" πηγαίνοντας στην επόμενη ενέργεια.

6.3.3 Εισαγωγή ημερομηνίας

Αφού ο χρήστης εισάγει την περιοχή του νέου ταξιδιού και πατήσει το κουμπί "ημερομηνίες" ανοίγει νέο παράθυρο με την φόρμα εισαγωγής των ημερομηνιών που θα γίνει η αναχώρηση και η επιστροφή, όπως φαίνεται στη παρακάτω εικόνα.

Εικόνα 6.7: Οθόνη εισαγωγής ημερομηνιών

Αφού εισάγει τα απαραίτητα στοιχεία έχει την δυνατότητα είτε να επιστρέψει στο μενού, είτε να προχωρήσει στη διαδικασία εισαγωγής στοιχείων του ταξιδιού, πατώντας το κουμπί "έξοδα" πηγαίνοντας στην επόμενη ενέργεια.

6.3.4 Εισαγωγή προβλεπόμενων εξόδων

Αφού ο χρήστης εισάγει τις ημερομηνίες του νέου ταξιδιού και πατήσει το κουμπί "έξοδα" ανοίγει νεό παράθυρο με την φόρμα εισαγωγής των εξόδων εγγραφής, μετακίνησης, διαμονής και προσωπικών, όπως φαίνεται παρακάτω.

Εικόνα 6.8: Οθόνη εισαγωγής εξόδων

Αφού εισάγει τα απαραίτητα στοιχεία έχει την δυνατότητα να επιστρέψει στο μενού, όπου πλέον όλες οι λειτουργίες εισαγωγής στοιχείων του ταξιδιού έχουν ολοκληρωθεί και εμφανίζεται η επιλογή "Αποστολή" με την οποία ο χρήστης βλέπει ξανά όλα τα στοιχεία που έχει εισάγει και έχει την δυνατότητα είτε να επιστρέψει να αλλάξει κάποια είτε να στείλει την αίτηση του νέου ταξιδιού για να πάρει την έγκριση. Οι οθόνες αυτών των επιλογών φαίνονται παρακάτω.

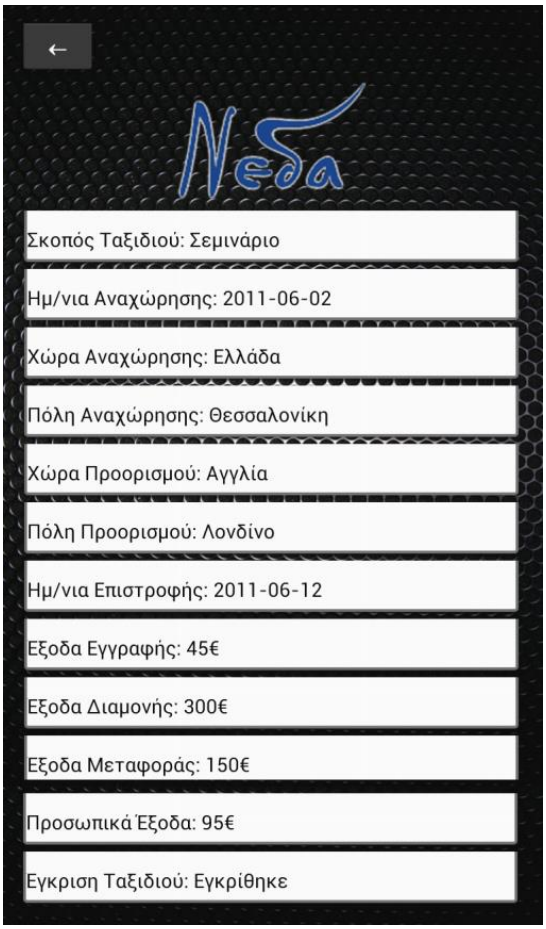
Εικόνα 6.9: Οθόνες αποστολής αίτησης ταξιδιού

6.4 Ιστορικό Ταξιδιών

Στην οθόνη αυτή ο χρήστης έχει την δυνατότητα να ελέγχει τα ταξίδια που έχει πραγματοποιήσει αλλά και τα ταξίδια που έχει δηλώσει και δεν έχουν εγκριθεί ακόμη. Έχει την δυνατότητα επιλέγοντας κάποιο ταξίδι να δει τις λεπτομέρειες του. Η αρχική οθόνη εμφάνισης του ιστορικού φαίνεται παρακάτω:

Εικόνα 6.10: Οθόνη εμφάνισης ιστορικού

Πατώντας πάνω σε κάποιο από τα ταξίδια, ο χρήστης μπορεί να δει τις λεπτομέρειες του ταξιδιού, εκείνες που είχε προσθέσει στην αίτηση για νέο ταξίδι, καθώς επίσης για το αν έχει εγκριθεί ή όχι. Η οθόνη που εμφανίζεται με τις λεπτομέρειες του ταξιδιού φαίνεται στην επόμενη εικόνα.



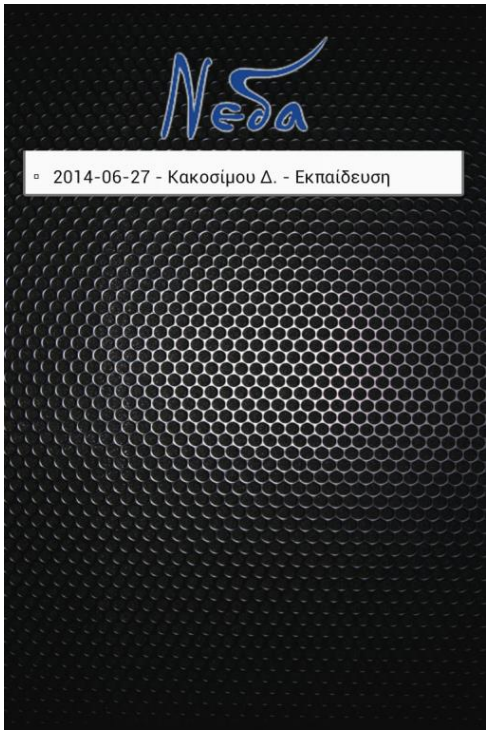
Εικόνα 6.11: Οθόνη εμφάνισης λεπτομερειών ταξιδιού

6.5 Έγκριση Ταξιδιού

Όταν κάνει login ο διαχειριστής που θα εγκρίνει ή θα απορρίψει μία αίτηση για νέο ταξίδι, εμφανίζεται στην αρχική οθόνη μια ειδοποίηση με τις νέες αιτήσεις, που απαιτούν άμεση έγκριση ή μη. Πατώντας στην ειδοποίηση ανοίγει νέο παράθυρο με μία λίστα των αιτήσεων. Αν δεν υπάρχει κάμια νέα αίτηση εμφανίζεται και πάλι η λίστα με τις αιτήσεις, τόσο με αυτές που δεν χρειάζονται άμεση απάντηση όσο και με αυτές που έχουν απορριφθεί. Αν περάσει ο χρόνος λήξης ενός ταξιδιού, χωρίς να το έχει εγκρίνει ο διαχειριστής το σύστημα το απορρίφθει αυτόματα.

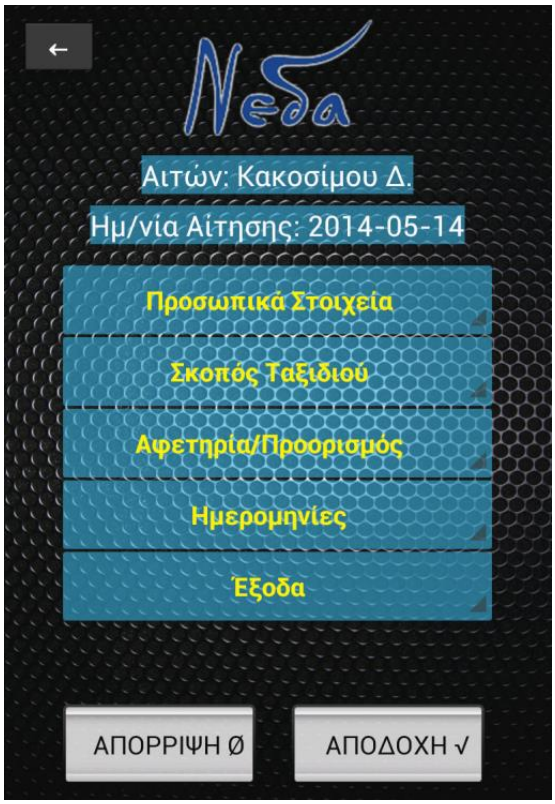
Παρακάτω φαίνεται η οθόνη που βλέπει ο διαχειριστής κατά την είσοδό του στο σύστημα καθώς και η οθόνη προβολής της λίστας των αιτήσεων.

Εικόνα 6.12: Οθόνη προβολής νέων αιτήσεων



Εικόνα 6.13: Οθόνη εμφάνισης λίστας αιτήσεων

Ο διαχειριστής επιλέγοντας μία αίτηση από την λίστα βλέπει λεπτομερώς τα προσωπικά στοιχεία του χρήστη που έκανε την αίτηση καθώς και τα στοιχεία του ταξιδιού. Οι οθόνες με τα στοιχεία που βλέπει ο διαχειριστής φαίνονται παρακάτω.



Εικόνα 6.14: Οθόνη εμφάνισης λεπτομερειών αίτησης

Εικόνα 6.15: Οθόνη εμφάνισης προσωπικών στοιχείων

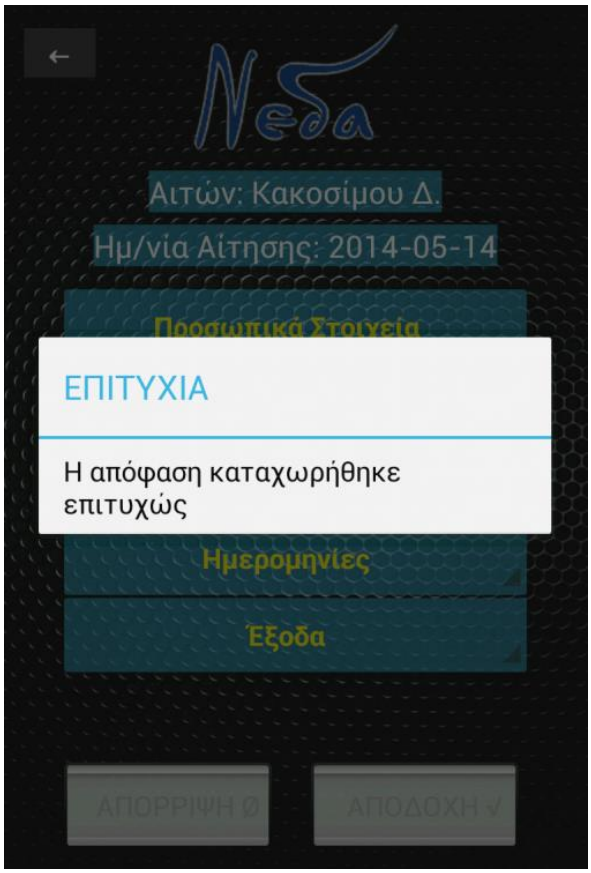
Εικόνα 6.16: Σκοπός ταξιδιού

Εικόνα 6.17: Στοιχεία αναχώρησης και προορισμού

Εικόνα 6.18: Χρονική περίοδος ταξιδιού

Εικόνα 6.19: Προβλεπόμενα έξοδα ταξιδιού

Αφού δει όλα τα στοιχεία του ταξιδιού ο διαχειριστής μπορεί να εγκρίνει ή να απορρίψει την αίτηση. Με αυτήν την επιλογή εμφανίζεται ένα popup παράθυρο με μήνυμα επιτυχούς καταχώρησης. Η εικόνα φαίνεται παρακάτω.



Εικόνα 6.20: Επιτυχία καταχώρησης απόφασης του διαχειριστή

VII. Σύνοψη και Συμπεράσματα

Στην συγκεκριμένη διπλωματική ασχοληθήκαμε με την ανάπτυξη μίας εφαρμογής, η οποία απευθύνεται στους χρήστες του συστήματος Νεδα του Ινστιτούτου Τεχνολογίας Υπολογιστών στην Πάτρα και θα εκτελείται σε κινητές συσκευές που χρησιμοποιούν λειτουργικό σύστημα Android. Οι λόγοι που επιλέχθηκε η ανάπτυξη της εφαρμογής να γίνει σε Android είναι οι εξής:

- Το λειτουργικό σύστημα Android χρησιμοποιείται στο μεγαλύτερο ποσοστό smartphones της αγοράς.
- Οι μεγαλύτερες εταιρίες κινητών τηλεφώνων όπως οι Sony, Samsung, LG, HTC χρησιμοποιούν το λειτουργικό σύστημα σε όλο και περισσότερες συσκευές τους.
- Η εκμάθηση του λειτουργικού συστήματος καθώς και ανάπτυξη εφαρμογών δεν έχουν μεγάλο βαθμό δυσκολίας.
- Δεν απαιτούνται ιδιαίτερες γνώσεις προγραμματισμού για την ανάπτυξη απλών εφαρμογών. Αρκεί η εξοικείωση με την γλώσσα προγραμματισμού Java για την ανάπτυξη απλών εφαρμογών καθημερινής χρήσης.
- Τα προγραμματιστικά εργαλεία που απαιτούνται για την ανάπτυξη εφαρμογών υπάρχουν ελεύθερα στο διαδίκτυο και καθένας μπορεί να τα κατεβάσει και να τα χρησιμοποιήσει.

- Η επίσημη ιστοσελίδα http://developer.android.com/index.html σχετικά με την κατασκευή εφαρμογών σε Android, περιλαμβάνει αναλυτικά βήματα και οδηγούς που βοηθάνε τόσο τον προγραμματιστή όσο και αρχάριους χρήστες στο ξεκίνημά τους.

Η έρευνα, πάνω στις τεχνολογίες του Spring και του Android, που πραγματοποιήθηκε για την εκπόνηση της διπλωματικής εργασίας ήταν ενδιαφέρουσα και με βοήθησε στην εξοικείωση νέων τεχνολογιών που χρησιμοποιούνται όλο και περισσότερο καθημερινά. Κατανοήθηκε η έννοια της προσωποποίησης περιεχομένου και εφαρμόστηκαν στην πράξη βασικές έννοιες σχετικά με την αλληλεπίδραση ανθρώπου-μηχανής. Αποκτήθηκαν πολύ σημαντικές γνώσεις για τις τεχνολογίες των smartphones και τον τρόπο ανάπτυξης εφαρμογών σε αυτά, καθώς αποτελεί ένα καινούργιο και συνεχώς αναπτυσσόμενο κλάδο της επιστήμης υπολογιστών, ο οποίος δημιουργεί συνεχώς νέες θέσεις εργασίας.

Βιβλιογραφία

- Περιβάλλον Spring.io: http://spring.io/
- Consuming a RESTful Web Service with Spring for Android: http://spring.io/guides/gs/consuming-rest-android/
- Json: http://spring.io/understanding/JSON
- Spring for Android: http://projects.spring.io/spring-android/
- Github Social Coding: https://github.com/
- MySQL Connector: http://dev.mysql.com/doc/connector-j/en/
- Android Development: http://developer.android.com/training/basics/firstapp/index.html
- Android Development Community: http://www.anddev.org/
- Wikipedia Linux: http://el.wikipedia.org/wiki/Linux
- Wikipedia MySQL: http://el.wikipedia.org/wiki/MySQL
- Wikipedia Android: http://el.wikipedia.org/wiki/Android