ΤΕΧΝΟΛΟΓΙΚΟ
ΕΚΠΑΙΔΕΥΤΙΚΟ
ΙΔΡΥΜΑ
ΤΕΙ ΗΠΕΙΡΟΥ

Εικόνα 1: Λογότυπο Android & Εικονίδιο εφαρμογής

ΤΙΤΛΟΣ

«ΜΕΛΕΤΗ, ΑΝΑΛΥΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΕΦΑΡΜΟΓΗΣ ΣΕ ΠΛΑΤΦΟΡΜΑ ΓΙΑ ΚΙΝΗΤΑ ΤΗΛΕΦΩΝΑ ΜΕ ΣΤΟΧΟ ΤΗΝ ΔΙΑΧΕΙΡΙΣΗ ΚΑΙ ΕΠΕΞΕΡΓΑΣΙΑ ΥΠΗΡΕΣΙΩΝ ΠΟΥ ΣΧΕΤΙΖΟΝΤΑΙ ΜΕ ΤΗΝ ΣΥΝΤΗΡΗΣΗ ΑΥΤΟΚΙΝΗΤΩΝ»

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΑΠΟ ΤΗΝ ΦΟΙΤΗΤΡΙΑ:
ΔΗΜΗΤΡΑ ΦΑΣΟΥΛΗ

ΑΜ:11175

Email: dimitra_fa@hotmail.com

Επιβλέπουσα καθηγήτρια: ΛΙΑΓΚΟΥ ΒΑΣΙΛΙΚΗ

ΑΡΤΑ 2018

TITLE
"STUDY, ANALYSIS AND IMPLEMENTATION OF A PLATFORM APPLICATION FOR MOBILE PHONES WITH THE OBJECTIVES OF THE MANAGEMENT AND PROCESS OF SERVICES CONNECTED WITH MAINTENANCE OF MOTOR VEHICLES"

Εγκρίθηκε από τριμελή εξεταστική επιτροπή :

1. Επιβλέπων καθηγητής
2. Μέλος επιτροπής
3. Μέλος επιτροπής

Ο/Η Προϊστάμενος/η του τμήματος

ΔΗΛΩΣΗ ΠΝΕΥΜΑΤΙΚΗΣ ΙΔΙΟΚΤΗΣΙΑΣ

Η παρούσα εργασία αποτελεί προϊόν αποκλειστικά δικής μου προσπάθειας. Όλες οι πηγές που χρησιμοποιήθηκαν περιλαμβάνονται στην βιβλιογραφία και γίνεται ρητή αναφορά σε αυτές μέσα στο κείμενο όπου έχουν χρησιμοποιηθεί .

ΠΕΡΙΛΗΨΗ

Στη σημερινή εποχή οι εφαρμογές κινητών τηλεφώνων σε συνδυασμό με τις βάσεις δεδομένων συμβάλλουν σημαντικά στην εξέλιξη του τομέα παροχής υπηρεσιών των επιχειρήσεων. Στην εργασία αυτή αρχικά αναφέρονται οι λόγοι κυριαρχίας του android στην παγκόσμια αγορά, οι οποίοι συνέβαλαν στην επιλογή αυτής της πλατφόρμας για την ανάπτυξη της εφαρμογής που πραγματεύεται η πτυχιακή και αφορά την γραμματειακή υποστήριξη σε φανοποιεία αυτοκινήτων. Η εφαρμογή θα παρέχει απλοποιημένες υπηρεσίες, όπως την καταγραφή αλλά και ανάκτηση εύκολα και γρήγορα όλων των απαραίτητων στοιχείων των πελατών, των αυτοκινήτων και των επισκευών τους μέσω κινητού τηλεφώνου, εφόσον αυτό έχει λειτουργικό σύστημα android. Επιπροσθέτως μέσω της εφαρμογής δίνεται η δυνατότητα προβολής συγκεντρωτικών αποτελεσμάτων για το σύνολο των επισκευών που πραγματοποιούνται ανά μήνα και έτος. Σχετικά με το android παραθέτονται επιπλέον τα βασικά χαρακτηριστικά του. Εν' συνεχεία γίνεται παρουσίαση της δομής και του γραφικού περιβάλλοντος της εφαρμογής που αναπτύχθηκε καθώς και ανάλυση της λειτουργίας της μέσα από την περιγραφή των κλάσεων της εφαρμογής. Τέλος γίνεται αναφορά στη μελλοντική εξέλιξη της εφαρμογής, ενώ λαμβάνοντας υπόψη τον βαθμό επιρροής των επιχειρήσεων από την αξιοποίηση εφαρμογών, επισημαίνονται τα οφέλη τους από αυτές, τα πιθανά προβλήματα ως προς την αποδοχή και ανάπτυξη τους με την σχετική αναφορά στην επίλυση αυτών.

Λέξεις κλειδιά :εφαρμογή, android, SQLite, επιχείρηση.

Abstract

Nowadays, mobile phone applications combined with databases contribute significantly to the development of the business services sector .This paper initially mentions the reasons for android's dominance in the global marketplace, which contributed to the choice of this platform for application development covered by the bachelor's degree in secretarial assistance in car body shops. The application will provide simplified services such as recording and recovering all the necessary items of their customers, cars and repairs by mobile phone if it has an Android operating system. Additionally, through the application, it is possible to display aggregate results for all repairs carried out per month and year. About android are also referring to its basic features. Next, the structure and graphical environment of the application developed and an analysis of its function are described through the description of the application classes. Finally, reference is made to the future development of the application, while taking into account the degree of influence of the enterprises on the exploitation of applications, their benefits are highlighted, the possible problems regarding their acceptance and development with the relevant reference to their solution.

Keywords: application, android, SQLite, business

Πίνακας περιεχομένων

Ευρετήριο Εικόνων:

1. ΚΕΦΑΛΑΙΟ:ΕΙΣΑΓΩΓΗ

Στο κεφάλαιο αυτό αναφέρεται το αντικείμενο της πτυχιακής ενώ περιγράφεται η οργάνωση του κειμένου της.

1.1 Αντικείμενο πτυχιακής

Η είσοδος των νέων τεχνολογιών, όπως η ένταξη μιας εφαρμογής android ή περισσοτέρων στην λειτουργία μιας επιχείρησης, δίνει νέες δυνατότητες και προοπτικές σε αυτή και ιδιαίτερα στον τομέα της παροχής υπηρεσιών, γεγονός που την καθιστά πιο ανταγωνιστική. Βάσεις Δεδομένων και εφαρμογές κινητών τηλεφώνων κάνουν πιο διαλειτουργικές τις παρεχόμενες υπηρεσίες. Στο πλαίσιο αυτό, **σκοπός της παρούσας πτυχιακής** εργασίας είναι η μελέτη, ανάπτυξη και υλοποίηση μιας εφαρμογής για κινητά τηλέφωνα με λειτουργικό σύστημα android που προορίζεται για γραμματειακή υποστήριξη σε φανοποιεία-βαφεία αυτοκινήτων, στοχεύοντας στην απλοποίηση των διαδικασιών της επιχείρησης. Με την εφαρμογή αυτή ο χρήστης θα μπορεί εύκολα να καταγράφει στοιχεία που αφορούν την επιχείρηση όπως τα στοιχεία των πελατών και των αυτοκινήτων τους, καθώς επίσης μέσω μιας γρήγορης αναζήτησης να έχει άμεση πρόσβαση σε αυτά μέσω της σχεσιακής βάσης δεδομένων SQLite. Παράλληλα ο χρήστης έχει την δυνατότητα μέσα από την εφαρμογή να λαμβάνει εύκολα το πλήθος των επισκευών που πραγματοποιούνται κάθε μήνα ανά έτος ώστε να παρακολουθεί και να προβλέπει την πορεία της επιχείρησης σε βάθος χρόνου.

1.2 Δομή κειμένου

Η πτυχιακή αυτή εργασία αναλύεται συνολικά σε επτά κεφάλαια. Στο **πρώτο κεφάλαιο** περιγράφεται το αντικείμενο της πτυχιακής και η οργάνωση του κειμένου αυτής. Στο **δεύτερο κεφάλαιο** αναφέρονται οι λόγοι επιλογής του περιβάλλοντος android για την ανάπτυξη της εφαρμογής ενώ γίνεται και περιγραφή αυτού. Στο **τρίτο κεφάλαιο** παρουσιάζονται τα βασικά βήματα ανάπτυξης μιας εφαρμογής (πχ η διαδικασία εγκατάστασης των απαραίτητων προγραμματιστικών εργαλείων για την υλοποίησή της). Το **τέταρτο κεφάλαιο** περιλαμβάνει την οργάνωση της εφαρμογής, με κύριο θέμα την δομή της. Στο **πέμπτο κεφάλαιο** περιγράφεται το γραφικό περιβάλλον της εφαρμογής, δηλαδή παρουσιάζονται και αναλύονται όλες οι διαφορετικές οθόνες της εφαρμογής. Στο **έκτο** κεφάλαιο παρουσιάζονται οι λειτουργίες της εφαρμογής με την αντίστοιχη αναφορά των πιο σημαντικών τμημάτων του κώδικα καθορισμού αυτών. Στο **έβδομο** και τελευταίο κεφάλαιο παρουσιάζονται οι μελλοντικές εξελίξεις της εφαρμογής και τα συμπεράσματα που προκύπτουν μετά την υλοποίηση της. Τέλος,

παρουσιάζονται η **Βιβλιογραφία** δηλαδή όλες οι πηγές (πτυχιακές , ιστοσελίδες , βιβλία κ.α.**)** που χρησιμοποιήθηκαν για την συγγραφή του παρόντος κειμένου, καθώς για την ανάπτυξη της εφαρμογής.

2. ΚΕΦΑΛΑΙΟ :ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΣΕ ΠΕΡΙΒΑΛΛΟΝ ANDROID

Στο κεφάλαιο που ακολουθεί γίνεται αναφορά στους λόγους που η εφαρμογή αναπτύχθηκε στο περιβάλλον Android και στη συνέχεια γίνεται εισαγωγή στο λειτουργικό σύστημα Android.

2.1 Γιατί android ;

Πριν την περιγραφή του λειτουργικού συστήματος Android σημειώνονται μερικοί καθοριστικοί παράγοντες που λαμβάνονται υπόψη ως προς την επιλογή αυτής της πλατφόρμας για την ανάπτυξη μιας εφαρμογής [1].

- ✓ Αποτελεί ανοιχτή , ελεύθερη πλατφόρμα ανάπτυξης , βασισμένη στο Linux.
- ✓ Η αρχιτεκτονική της είναι βασισμένη σε δομικά στοιχεία τα οποία τροποποιούνται και προσαρμόζονται στις ανάγκες κάθε κατασκευαστή και κατά συνέπεια χρήστη.
- ✓ Διαθέτει πολλές ενσωματωμένες υπηρεσίες που προσφέρουν κάθε φορά ανεπανάληπτη εμπειρία στον χρήστη, όπως υπηρεσίες βασισμένες στην τοποθεσία, πανίσχυρη SQL βάση δεδομένων, μηχανή αναζήτησης και χάρτες.
- ✓ Αυτόματη διαχείριση του κύκλου ζωής μιας εφαρμογής, με πολλαπλές δικλίδες ασφαλείας ανάμεσα στα προγράμματα, βελτιστοποιήσεις στον τομέα διαχείρισης μνήμης και χαμηλής κατανάλωσης .
- ✓ Παροχή υψηλής ποιότητας γραφικών και ήχου.
- ✓ Φορητότητα ανάμεσα σε ένα ευρύ φάσμα ήδη υπάρχον υλικού αλλά και μελλοντικού ,που οφείλεται στο γεγονός ότι όλα τα προγράμματα γράφονται σε Java και εκτελούνται από την εικονική μηχανή Dalvik. Επίσης καθίσταται εφικτή η τροποποίηση των οθονών κατάλληλα ώστε να υποστηρίζουν οποιαδήποτε ανάλυση, μέγεθος και προσανατολισμό οθόνης.

Η επιλογή του λειτουργικού συστήματος android βασίζεται στο γεγονός ότι σε αντίθεση με άλλες πλατφόρμες κινητών συσκευών δεν υπάρχει επιπρόσθετο κόστος για την ανάπτυξη εφαρμογών android. Επιπλέον οι προγραμματιστές σε android λογισμικό έχουν πλήρη πρόσβαση στα χαρακτηριστικά των συσκευών, δυνατότητα που παρέχεται και στους χρήστες. Τέλος υπάρχει πληθώρα εργαλείων για τα λειτουργικά συστήματα Android μιας και ο προγραμματισμός και η ανάπτυξη τους είναι εύκολη και στηρίζονται στη γλώσσα java, μια από τις δημοφιλέστερες γλώσσες προγραμματισμού στον κόσμο.

2.1.1 Παγκόσμιες πωλήσεις

Σύμφωνα με το άρθρο της ιστοσελίδας tecky.eu [3], που αναφέρεται στην έρευνα της IDC σχετικά με τις 5 κορυφαίες εταιρείες κινητών τηλεφώνων που δημοσίευσε 31/7/2018 και αφορά τις παγκόσμιες πωλήσεις την περίοδο Απρ-Ιουν18 προέκυψαν τα εξής συμπεράσματα :

- Η **Samsung**, αν και είχε απώλειες (3,4%) παρέμεινε στην **πρώτη θέση** με μερίδιο αγοράς **20,9%** και αποστολές 71,5 εκατ. συσκευών.
- Στη **δεύτερη θέση** πέρασε η **Huawei** αποστέλλοντας 54,2 εκατ. συσκευές έναντι 38,5 εκατ. την αντίστοιχη περίοδο του 2017. Το **μερίδιο αγορά**ς της εταιρείας εκτοξεύτηκε από το 11% στο **15,8%**. Η τάση αυτή είχε καταγραφεί και από την πρόσφατη έρευνα της Kantar, τα ευρήματα της οποίας έδειχναν άνοδο [για τη Huawei] στο Ηνωμένο Βασίλειο κατά 11% σε διάστημα ενός έτους.
- Η **Apple έπεσε** στην **τρίτη θέση**. Αν και η εταιρεία αύξησε (ελάχιστα) τις αποστολές της αλλά και το μερίδιο αγοράς της κατά 0,7%, δεν κατάφερε να κρατήσει τη δεύτερη θέση, καθώς η άνοδος της Huawei ήταν εντυπωσιακή (περίπου 40%).

- Ακολουθεί η **Xiaomi**, η οποία σημείωσε το μεγαλύτερο ποσοστό ανόδου (48,8%) εκτινάσσοντας το **μερίδιο αγοράς** της στο **9,3%**, έχοντας αποστείλει σχεδόν 32 εκατ. συσκευές. Η Xiaomi έριξε από την τέταρτη θέση την OPPO.
- Η **OPPO**, αν και αύξησε τις αποστολές της κατά 1,4 εκατ. συσκευές, δεν κατάφερε να κρατήσει την τέταρτη θέση. Το **μερίδιο αγοράς** της εταιρείας σημείωσε άνοδο (5,1%) και έφτασε το **8,6%**.

  Άρα η Android είναι η πρώτη σε πωλήσεις παγκοσμίως πλατφόρμα για έξυπνα κινητά τηλέφωνα.

2.2 Ορισμός android

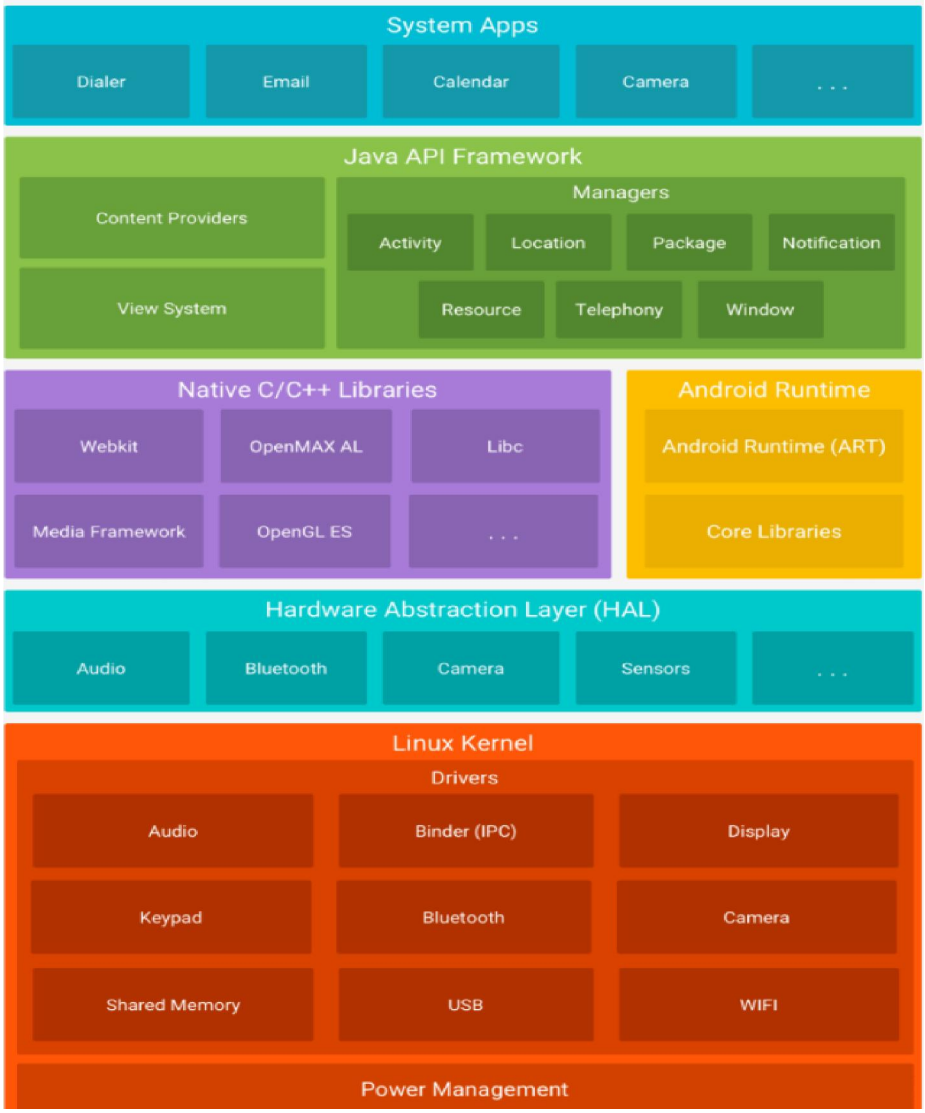
Το Android είναι λειτουργικό σύστημα ανοιχτού κώδικα, για έξυπνες φορητές συσκευές όπως κινητά τηλέφωνα αφής και τάμπλετ , το οποίο τρέχει τον πυρήνα του λειτουργικού Linux [2]. Το Android έχει μια μεγάλη κοινότητα προγραμματιστών που γράφουν εφαρμογές, οι οποίες επεκτείνουν τη λειτουργικότητα των συσκευών.

2.2.1 Ιστορική αναδρομή.

Το 2007, μια ομάδα κατασκευαστών τηλεφωνικών συσκευών, φορέων ασύρματων επικοινωνιών και προγραμματιστών λογισμικού (κυρίως Google), σχημάτισαν τη συμμαχία Open Handset Alliance [4][5] και ανάπτυξαν την επόμενη γενιά πλατφόρμας ασύρματων δικτύων. Συγκεκριμένα στις 5 Νοεμβρίου 2007 έγινε η πρώτη παρουσίαση της πλατφόρμας android που αποτελεί την πρώτη πλήρης, ανοιχτή και δωρεάν διαθέσιμη πλατφόρμα κινητών επικοινωνιών. Το μεγαλύτερο μέρος του κώδικα του android δημοσιεύτηκε από την Google με τις δωρεάν άδειες χρήσης της Apache [6]. Έπειτα, η πλατφόρμα ξεκίνησε να υποστηρίζει το μοντέλο cloud computing και σχεσιακή βάση δεδομένων (SQLite) με αποτέλεσμα την ραγδαία διάδοση και εξέλιξη των συσκευών με λειτουργικό android [2].

2.2.2 Σχεδιασμός-Αρχιτεκτονική

Το android αποτελείται από έναν πυρήνα βασισμένο σε αυτόν του Linux, ο οποίος είναι υπεύθυνος για τη διαχείριση των οδηγών συσκευών, τον έλεγχο πρόσβασης στους πόρους του συστήματος, τη διαχείριση μνήμης και τις λοιπές υπηρεσίες που παρέχει ένα λειτουργικό σύστημα [7]. Στους οδηγούς συσκευών συγκαταλέγονται αυτοί της οθόνης, του ασύρματου δικτύου, της κάμερας, του ήχου κ.α. Ένα επίπεδο επάνω βρίσκονται οι native βιβλιοθήκες του συστήματος που είναι γραμμένες σε C++ και περιλαμβάνουν το OpenGL, την SQLite, την Media library κ.α. Οι εφαρμογές που τρέχουν στο κινητό μπορούν να έχουν πρόσβαση στις βιβλιοθήκες αυτές μέσω της εικονικής μηχανής Dalvik JVM. Οι εφαρμογές android είναι γραμμένες σε Java και άρα για να τρέξουν χρειάζονται το αντίστοιχο περιβάλλον. Όπως λοιπόν για να εκτελέσουμε μία εφαρμογή σε ένα PC είναι απαραίτητο να είναι εγκατεστημένο το κατάλληλο JRE (Java Runtime Environment), για τις εφαρμογές Android τον ρόλο του JRE παίζει η εικονική μηχανή Dalvik VM.



Εικόνα 2: Η Αρχιτεκτονική Android[24]

2.2.3 Χαρακτηριστικά του Android

Μερικά από τα χαρακτηριστικά του Android είναι τα ακόλουθα[8]:

Συνδεσιμότητα: Το Android υποστηρίζει διάφορες τεχνολογίες όπως GMS/EDGE, Bluetooth, Wi-Fi, WiMAX και LTE.

Μηνύματα: Υποστηρίζει τις κλασσικές μορφές SMS και MMS καθώς επίσης και το Android Cloud to Device Messaging (C2DM), που είναι μια υπηρεσία προώθησης ειδοποιήσεων που βοηθά τους προγραμματιστές να στέλνουν δεδομένα από τον εξυπηρετητή στις εφαρμογές τους.

Φυλλομετρητής (browser): Είναι διαθέσιμος στο Android και βασίζεται στην διάταξη ανοικτού κώδικα WebKit, που είναι μια μηχανή που ενδυναμώνει διάφορους browsers.

Υποστήριξη Java: Οι περισσότερες εφαρμογές είναι γραμμένες σε Java και οι κλάσεις της τρέχουν με μια εικονική μηχανή ειδικά για το Android που είναι βελτιστοποιημένο για συσκευές που τροφοδοτούνται από μπαταρία με περιορισμένη μνήμη και υπολογιστική ισχύ.

Αποθήκευση: Η αποθήκευση επιτυγχάνεται με την χρήση της SQLite που αποτελεί μια μικρή σε απαιτήσεις σχεσιακή βάση δεδομένων.

Επιπλέον υπάρχει ένα ευρύ φάσμα από χαρακτηριστικά όπως η πολυδιεργασία , κλήσεις μέσω IP, εξωτερική κάρτα μνήμης κ.α.

2.3 Εκδόσεις λογισμικού

Οι συνεχείς ενημερώσεις στην λειτουργία του λειτουργικού συστήματος android επιτρέπουν στον χρήστη μετά την ενημέρωση της έκδοσης να αξιοποιεί τις νέες δυνατότητες που παρέχονται [2]. Υπάρχει όμως το μειονέκτημα ότι δεν μπορούν όλες οι συσκευές να ενημερώνονται με τις τελευταίες εκδόσεις, καθώς υπάρχουν περιορισμοί ανάλογα με τις δυνατότητες της εκάστοτε συσκευής.

Έκδοση	Κωδική Ονομασία	Ημερομηνία	API level
9.0	*Pie*	6 Αυγούστου 2018	28
8.1	*Oreo*	5 Δεκεμβρίου 2017	27
8.0		21 Αυγούστου 2017	26
7.1	*Nougat*	4 Οκτωβρίου 2016	25
7.0		22 Αυγούστου 2016	24
6.0	*Marshmallow*	5 Οκτωβρίου 2015	23
5.1	*Lollipop*	9 Μαρτίου 2015	22
5.0		3 Νοεμβρίου 2014	21
4.4	*KitKat*	31 Οκτωβρίου 2013	19
4.3	*Jelly Bean*	24 Ιουλίου 2013	18
4.2		13 Νοεμβρίου 2012	17
4.1		9 Ιουλίου 2012	16
4.0	*Ice Cream Sandwich*	16 Δεκεμβρίου 2011	15
3.2	*Honeycomb*	15 Ιουλίου 2011	13
2.3	*Gingerbread*	9 Φεβρουαρίου 2011	10
2.2	*Froyo*	20 Μαΐου 2010	8
2.0	*Eclair*	26 Οκτωβρίου 2009	7
1.6	*Donut*	15 Σεπτεμβρίου 2009	4

Εικόνα 3: Χρονοδιάγραμμα Εκδόσεων Android[2]

2.3.1 Android 1.0

Το Android 1.0, η πρώτη εμπορική έκδοση του λογισμικού, κυκλοφόρησε στις 23 Σεπτεμβρίου 2008 [9]. Η πρώτη διαθέσιμη στο εμπόριο συσκευή Android ήταν το HTC Dream . Το Android 1.0 ενσωμάτωσε τα ακόλουθα χαρακτηριστικά:

- Το Android Market επέτρεψε τη λήψη και την ενημέρωση εφαρμογών μέσω της εφαρμογής Market
- Πρόγραμμα περιήγησης στο Web για προβολή, μεγέθυνση και περιστροφή πλήρων σελίδων HTML και XHTML - Πολλές σελίδες εμφανίζονται ως παράθυρα ("κάρτες")
- Υποστήριξη κάμερας - ωστόσο, αυτή η έκδοση δεν είχε την επιλογή αλλαγής της ανάλυσης της κάμερας, της ισορροπίας λευκού, της ποιότητας κ.λπ.
- Φάκελοι που επιτρέπουν την ομαδοποίηση ενός αριθμού εικονιδίων εφαρμογής σε ένα εικονίδιο ενός φακέλου στην Αρχική οθόνη
- Πρόσβαση σε διακομιστές ηλεκτρονικού ταχυδρομείου ιστού, με υποστήριξη POP3, IMAP4 και SMTP
- Συγχρονισμός Gmail με την εφαρμογή Gmail
- Συγχρονισμός επαφών Google με την εφαρμογή Άτομα
- Συγχρονισμό Ημερολογίου Google με την εφαρμογή Ημερολόγιο
- Χάρτες Google με Street View για προβολή χαρτών και δορυφορικών εικόνων, καθώς και για τοπικές επιχειρήσεις και οδηγίες οδήγησης χρησιμοποιώντας GPS
- Google Sync, επιτρέποντας τη διαχείριση του συγχρονισμού over-the-air του Gmail, του People και του Ημερολογίου

- Αναζήτηση Google, επιτρέποντας στους χρήστες να πραγματοποιούν αναζήτηση στο Internet και σε εφαρμογές τηλεφώνου, επαφές, ημερολόγιο κ.λπ.
- Google Talk άμεσων μηνυμάτων
- Στιγμιαία μηνύματα, μηνύματα κειμένου και MMS
- Media Player, επιτρέποντας τη διαχείριση, την εισαγωγή και την αναπαραγωγή αρχείων πολυμέσων - ωστόσο, αυτή η έκδοση δεν είχε υποστήριξη βίντεο και στερεοφωνικό Bluetooth
- Οι ειδοποιήσεις εμφανίζονται στη γραμμή κατάστασης, με επιλογές για τον ορισμό του ήχου κλήσης, των ειδοποιήσεων LED ή των δονήσεων
- Το Voice Dialer επιτρέπει την κλήση και την τοποθέτηση τηλεφωνικών κλήσεων χωρίς την πληκτρολόγηση ενός ονόματος ή αριθμού
- Ταπετσαρία επιτρέπει στο χρήστη να ρυθμίσει την εικόνα φόντου ή τη φωτογραφία πίσω από τα εικονίδια και τα γραφικά στοιχεία της Αρχικής οθόνης
- Αναπαραγωγή βίντεο YouTube
- Άλλες εφαρμογές περιλαμβάνουν: Ξυπνητήρι, Αριθμομηχανή, Κλήση (τηλέφωνο), Αρχική οθόνη (Εκκίνηση), Εικόνες (Gallery) και Ρυθμίσεις
- Υποστήριξη Wi-Fi και Bluetooth

2.3.2 Android 1.1

Στις 9 Φεβρουαρίου 2009 κυκλοφόρησε η ενημερωμένη έκδοση Android 1.1, αρχικά μόνο για το HTC Dream[9]. Το Android 1.1 ήταν γνωστό ως "Petit Four" εσωτερικά, αν και αυτό το όνομα δεν χρησιμοποιήθηκε επισήμως. Η ενημέρωση επιλύει τα σφάλματα, αλλάζει το Android API (Application Programming Interface) και προσθέτει μια σειρά από λειτουργίες:

- Λεπτομέρειες και κριτικές διαθέσιμα όταν ένας χρήστης αναζητά επιχειρήσεις σε Χάρτες
- Μεγαλύτερη προεπιλογή οθόνης κατά τη διάρκεια της κλήσης όταν χρησιμοποιείτε το μεγάφωνο, καθώς και δυνατότητα εμφάνισης / απόκρυψης του πληκτρολογίου
- Δυνατότητα αποθήκευσης συνημμένων σε μηνύματα
- Υποστήριξη προστιθέμενη για το marquee σε διατάξεις του συστήματος

2.3.3 Android 1.5 Cupcake

Εικόνα 4: Android 1.5 Cupcake

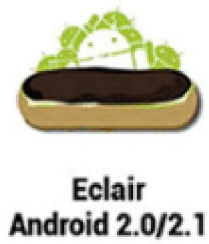
Η έκδοση “Cupcake” [10], βασισμένη στο LinuxKernel 2.6. 27, υποστηρίζει νέες λειτουργίες για την κάμερα της συσκευής, όπως η καταγραφή και παρακολούθηση βίντεο και η άμεση μεταφόρτωση αυτών στο Youtube και το Picasa απευθείας από το τηλέφωνο. Έχει νέο έξυπνο πληκτρολόγιο με πρόβλεψη κειμένου. Επιπλέον υποστηρίζει πρότυπο Bluetooth A2DP και AVRCP ενώ έχει και την ικανότητα να συνδέεται αυτόματα σε μικροσυσκευές Bluetooth από μια συγκεκριμένη απόσταση. Ακόμα στην έκδοση αυτή έχει νέο γραφικό περιβάλλον με κινούμενες μεταβάσεις οθόνης.

2.3.4 Android 1.6 Donut

Εικόνα 5: Android 1.6 Donut

Η έκδοση "Donut", βασισμένη στο LinuxKernel 2.6.29, παρουσιάστηκε στις 15 Σεπτεμβρίου του 2009 [10]. Έχει ταχύτερη απόκριση σε σχέση με την προηγούμενη έκδοση. Υποστηρίζεται πλέον η επιλογή πολλαπλών αρχείων ταυτόχρονα, έχει ανανεωμένη έκθεση φωτογραφιών και φωτογραφική μηχανή, καθώς και βελτιωμένο Android Market. Επιπροσθέτως έχει ανανεωμένη φωνητική αναζήτηση, με ταχύτερη απόκριση και βαθύτερη ολοκλήρωση με εγγενείς (native)εφαρμογές, συμπεριλαμβανομένης της δυνατότητας κλήσης επαφών. Ακόμα παρέχει την δυνατότητα αναζήτησης σελιδοδεικτών, ιστορικού, επαφών αλλά και στο διαδίκτυο από την αρχική οθόνη .

2.3.5 Android 2.0/2.1 Eclair



Εικόνα 6: Android 2.0/2.1 Eclair

Η Google κυκλοφόρησε το Gingerbread επίσημα τον Δεκέμβριο του 2010 [11], με την επανέκδοση του σε Android 2.3.3 τον Φεβρουάριο του 2011. Έτρεχε στην πρώτη συσκευή Nexus που παράχθηκε από τη Samsung. Αυτό ήταν ένα σημαντικό βήμα, καθώς η Samsung έχει γίνει ο μεγαλύτερος κατασκευαστής smartphone. Το Gingerbread ήταν ένα πολύ επιτυχημένο λειτουργικό σύστημα, που με τον καιρό έγινε η πιο δημοφιλής έκδοση του Android. Οι αλλαγές που έχουν γίνει είναι οι ακόλουθες:

- Πιο γρήγορη, πιο διαισθητική εισαγωγή κειμένου
- Επιλογή λέξεων και αντιγραφή/επικόλληση με ένα άγγιγμα
- Βελτιωμένη ενεργειακή διαχείριση
- Υποστήριξη NFC (Near Field Communication)
- Υποστήριξη video κλήσης
- Υποστήριξη του πρωτόκολλου WebM για αναπαραγωγή video
- Υποστήριξη κοινωνικών δικτύων
- Υποστήριξη Google Wallet
- Υποστήριξη βαρόμετρου, γυροσκοπίου και βαρυτόμετρου
- Περιέχει πλούσια πολυμέσα
- Μετάβαση στην μπροστινή κάμερα από πρόγραμμα κάμερας
- Διαχείριση εφαρμογών

2.3.6 Android 2.2 Froyo

Εικόνα 7: Android 2.2 Froyo

Το Android 2.2 Froyo που εκδόθηκε τον Μάιο του 2010 επιτρέπει πλέον την αποθήκευση εφαρμογών όχι μόνο στην εσωτερική μνήμη της συσκευής αλλά και σε μια εξωτερική συσκευή μνήμης [12]. Επίσης εισήγαγε την υπηρεσία **Android Cloud to Device Messaging(C2DM).** Η υπολογιστική νέφους επιτρέπει την χρήση λογισμικού και δεδομένων , που είναι αποθηκευμένα στο "νέφος" – δηλαδή, να προσπελαύνονται από απομακρυσμένους υπολογιστές μέσω του Internet και να διατίθενται κατ ΄ απαίτηση. Ακόμα η υπολογιστική νέφους παρέχει την ευελιξία αυξομείωσης των υπολογιστικών πόρων. Μερικές από τις νέες αλλαγές που περιλαμβάνει είναι[11]:

- Νέο Android Market με αυτόματη ανανέωση των εφαρμογών
- Ενσωμάτωση στην μηχανή αναζήτησης, της μηχανής JavaScript V8
- Αυξημένη υποστήριξη Microsoft Exchange (σε πολιτικές ασφαλείας. ημερολογίου, auto-discovery, GAL lookup, remote wipe)
- Σύνδεση USB και λειτουργία δυναμικής ζώνης (hotspot) WiFi
- Επιλογή για απαγόρευση πρόσβασης δεδομένων πάνω από ένα δίκτυο κινητής τηλεφωνίας Γρήγορη εναλλαγή ανάμεσα σε πολλαπλές γλώσσες του πληκτρολογίου και των λεξικών τους Φωνητική κλήση και διαμοιρασμός επαφών με Bluetooth
- Υποστήριξη για αριθμητικούς και αλφαριθμητικούς κωδικούς
- Η μηχανή αναζήτησης μπορεί να αποτυπώσει κινούμενα GIFs
- Υποστήριξη για πεδία μεταφόρτωσης αρχείων στον browser
- Υποστήριξη για εγκατάσταση εφαρμογών στην επεκτάσιμη μνήμη
- Υποστήριξη Adobe Flash 10.1

2.3.7 Android 2.3 Gingerbread

Εικόνα 8: Android 2.3 Gingerbread

Η έκδοση "Gingerbread", βασισμένη στο Linux Kernel 2.6.35.7, παρουσιάστηκε στις 6 Δεκεμβρίου του203 [12], ενώ τον Φεβρουάριο του 2011 επανεκδόθηκε σε Android 2.3.3. Το Android 2.3, προσέθεσε περισσότερες βελτιώσεις χρήστη, όπως ένα ανασχεδιασμένο

πληκτρολόγιο, βελτιωμένες δυνατότητες πλοήγησης και πολλές άλλες. Επίσης προσέθεσε αρκετά χαρακτηριστικά προγραμματιστή, για επικοινωνίες, για πολυμέσα και για παιχνίδια. Το σημαντικότερο νέο χαρακτηριστικό αυτής της έκδοσης ήταν η υποστήριξη για επικοινωνίες κοντινού επιπέδου (NFC). Το NFC αποτελεί ένα πρότυπο ασύρματων επικοινωνιών μικρής απόστασης, που επιτρέπει την επικοινωνία ανάμεσα σε δύο συσκευές, οι οποίες απέχουν μεταξύ τους λίγα εκατοστά του μέτρου.

2.3.8 Android 3.0 Honeycomb

Εικόνα 9: Android 3.0 Honeycomb

Η έκδοση "Honeycomb", βασισμένη στο Linux Kernel 2.6.36, παρουσιάστηκε στις 9 /5/2011 [10], με την ιδιαιτερότητα ότι προοριζόταν αποκλειστικά για tablets. Υπάρχει ένα νέο, εντελώς διαφορετικό, User Interface και υποστηρίζονται διπύρηνοι και τετραπύρηνοι επεξεργαστές. Ακόμα, έχει απλοποιηθεί το multitasking έτσι ώστε ο χρήστης να μπορεί με τη χρήση ενός πλήκτρου (recentapps) να περνάει από μια εφαρμογή σε άλλη. Υπάρχει η δυνατότητα για Video Chat μέσω της εφαρμογής Google Talk καθώς η ανάγνωση βιβλίων μέσω του Google eBooks. Επιπλέον, μπορούν να κρυπτογραφηθούν όλα τα δεδομένα χρήστη.

2.3.9 Android 4.0 Ice Cream Sandwich

Εικόνα 10: Android 4.0 Ice Cream Sandwich

Η έκδοση "Ice Cream Sandwich", βασισμένη στο Linux Kernel 3.0.1, παρουσιάστηκε στις 19 Οκτωβρίου του 2011 [11] και ήταν σχεδιασμένη τόσο για τα κινητά όσο και για τα tablet. Μερικά από τα σημαντικότερα νέα χαρακτηριστικά που παρέχει η έκδοση αυτή είναι τα εξής [11] [12]:

- Η Ανίχνευση Προσώπου για να ξεκλειδώνει η συσκευή.
- Ο Εικονικός χειριστής κάμερας
- Το Android Beam που αξιοποιεί πλέον το NFC αφού επιτρέπει την αποστολή δεδομένων σε κοντινής εμβέλειας συσκευές.
- API κοινωνικής δικτύωσης με το οποίο με την άδεια του χρήστη πραγματοποιείται διαμοιρασμός πληροφοριών ανάμεσα σε κοινωνικά δίκτυα.
- API Διευκόλυνσης προσπέλασης για άτομα με ειδικές ανάγκες
- Πλαίσιο εργασίας Android@Home, όπου μέσω εφαρμογών ελέγχονται οικιακές συσκευές.
- Το Wi-Fi Direct, με το οποίο οι συσκευές μπορούν να συνδεθούν μεταξύ τους ασύρματα, χωρίς την μεσολάβηση κάποιου σημείου πρόσβασης.

2.3.10 Android 4.1-4.3.1 Jelly Bean

Εικόνα 11: Android 4.1-4.3.1 Jelly Bean

Τον Ιούλιο του 2012 παρουσιάζεται η έκδοση Jelly Bean [11]. Έκανε μεγάλη επιτυχία καθώς εισήγαγε πάρα πολλές νέες λειτουργίες, συνδυάζοντας χαρακτηριστικά του 'Honeycomb' και του 'Gingerbread' .
Τα νέα χαρακτηριστικά που προσέθεσε η έκδοση 4.1 είναι τα εξής:

- Εισαγωγή του Google Now
- Φωνητική αναζήτηση
- File manager
- Εσωτερικούς χάρτες
- Βελτιώσεις στην εφαρμογή της κάμερας
- Εισαγωγή της λειτουργίας χειρονομίας
- Φωτογραφίες 360 μοιρών με το Photo Sphere
- Widgets στο κλείδωμα οθόνης
- Πρόσθεση υποστήριξης για Bluetooth gamepads - συσκευές και χειριστήρια HID
- Βελτιώσεις στην ασφάλεια
- Δυνατότητα δημιουργίας περιορισμένων προφίλ στα tablet

2.3.11 Android 4.4-4.4.4 KitKat

Εικόνα 12: Android 4.4-4.4.4 KitKat

Η έκδοση Android 'KitKat' που παρουσιάστηκε στις 31 Οκτωβρίου του 2013 [11] και έχει σχεδιαστεί για να τρέχει γρήγορα, ομαλά, και με απόκριση, σε ένα πολύ ευρύτερο φάσμα των συσκευών από ποτέ. Τα νέα χαρακτηριστικά είναι τα εξής:

- Βελτίωση απόδοσης και ασφάλειας
- Βελτιωμένη πρόσβαση στις κοινοποιήσεις
- Επιλογή καταγραφής οθόνης
- Βελτιωμένη χρήση της μνήμης
- Νέο διαφανές σύστημα UI

2.3.12 Android 5.0-5.1.1 Lollipop

Εικόνα 13: Android 5.0-5.1.1 Lollipop

Το Android 5.0-5.1.1 "Lollipop" είναι μια έκδοση λειτουργικού συστήματος για κινητά Android [13] που αναπτύχθηκε από την Google και παρουσιάστηκε στις 25 Ιουνίου 2014 στο συνέδριο του Google I/O και έγινε διαθέσιμο μέσω της επίσημης over-the-air (OTA) ενημέρωσης στις 12 Νοεμβρίου 2014, για επιλεγμένες συσκευές που τρέχουν διανομές του Android και εξυπηρετούνται από την Google (όπως το Nexus και συσκευές έκδοσης Google Play). Τα βασικά χαρακτηριστικά που προστέθηκαν :

- Βελτίωση ταχύτητας
- Διορθώσεις σφαλμάτων
- Υποστηρίζονται πολλαπλές κάρτες SIM
- Συντομεύσεις γρήγορων ρυθμίσεων για σύνδεση σε δίκτυα Wi-Fi ή για έλεγχο συσκευών Bluetooth
- Κλείστε την προστασία εάν χάσετε ή κλαπεί
- Φωνητική κλήση υψηλής ευκρίνειας
- Βελτιώσεις σταθερότητας και επιδόσεων
- Διορθώσεις σφαλμάτων, επιδιόρθωση ζητημάτων με αναπαραγωγή βίντεο και αποτυχίες κωδικού πρόσβασης
- Νέο σχέδιο (σχεδιασμός υλικών)
- Βελτίωση ταχύτητας
- Βελτίωση της κατανάλωσης μπαταρίας

2.3.13 Android 6.0-6.0.1 Marshmallow

Εικόνα 14: Android 6.0-6.0.1 Marshmallow

Το Android 6.0-6.0.1 "Marshmallow" (με την κωδική ονομασία Android M κατά τη διάρκεια της ανάπτυξης του) είναι η έκτη κύρια έκδοση του λειτουργικού συστήματος Android από την Google[13]. Παρουσιάστηκε τον Μάιο του 2015 στο συνέδριο του Google I/O, και κυκλοφόρησε επίσημα τον Οκτώβριο του 2015. Το Marshmallow

επικεντρώνεται κυρίως στη βελτίωση της συνολικής εμπειρίας του χρήστη σε σύγκριση με τον προκάτοχο του, Lollipop. Τα βασικά χαρακτηριστικά που προστέθηκαν :

- Υποστήριξη USB Type-C
- Υποστήριξη ελέγχου ταυτότητας δακτυλικών αποτυπωμάτων
- Καλύτερη διάρκεια ζωής της μπαταρίας με "βαθιά ύπνο"
- Πίνακας ελέγχου δικαιωμάτων
- Android Pay
- Υποστήριξη MIDI
- Βελτιώσεις του Google Now

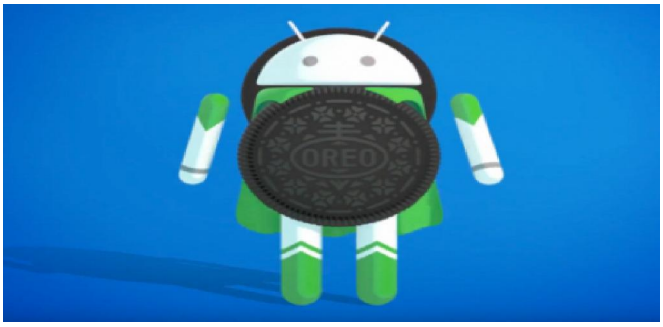
2.3.14 Android 7.0-7.1.1 Nougat

Εικόνα 15: Android 7.0-7.1.1 Nougat

Το Android 7.0-7.1.1 "Nougat" (με την κωδική ονομασία Android N κατά τη διάρκεια της ανάπτυξης του) είναι η έβδομη σημαντική έκδοση του λειτουργικού συστήματος Android από την Google. Κυκλοφόρησε για πρώτη φορά ως beta build στις 9/3/2016 και επίσημα στις 22 /8/ 2016 [13]. Βασικά χαρακτηριστικά της έκδοσης αυτής είναι:

- Πατήστε παρατεταμένα το εικονίδιο της εφαρμογής για να ενεργοποιήσετε νέες ενέργειες εκκίνησης
- Το προεπιλεγμένο πληκτρολόγιο επιτρέπει τώρα την άμεση αποστολή GIF
- Νέο σύνολο emojis
- Εικονική πραγματικότητα λειτουργία Daydream
- Φωτάκι νυκτός
- Βελτιώσεις διαχειριστή αποθήκευσης
- Βελτιώσεις απόδοσης για τις διευθύνσεις αφής και προβολής
- Επιλογή για ενεργοποίηση του δακτυλικού αποτυπώματος σύρετε προς τα κάτω τη χειρονομία
- Αδιάλειπτες ενημερώσεις συστήματος
- Λειτουργία πολλαπλών παραθύρων (PIP, παράθυρο Freeform)
- Αδιάλειπτες ενημερώσεις συστήματος (με διπλή κατάτμηση συστήματος)
- Καλύτερη απόδοση και μέγεθος κώδικα χάρη στον νέο συντάκτη JIT

2.3.15 Android 8.0 Oreo



Εικόνα 16: Android 8.0 Oreo

Κυκλοφόρησε 21/8/2017 με τα εξής νέα χαρακτηριστικά [14],[15]:

- Ταχύτητα: Το Android Oreo θα έχει διπλάσια ταχύτητα, όπως μετρήθηκε με το Google Pixel. Επίσης θα περιορίζει τη δραστηριότητα των εφαρμογών στο παρασκήνιο, κάτι που επίσης θα συντελεί στη μεγαλύτερη ταχύτητά του.
- Αυτόματη συμπλήρωση: Με την άδεια του χρήστη, το Autofill (αυτόματη συμπλήρωση) θυμάται τους κωδικούς του κι έτσι εξασφαλίζει γρήγορη πρόσβαση στις αγαπημένες του εφαρμογές.
- Picture In Picture: Χάρη στη δυνατότητα αυτή, πλέον ο χρήστης θα μπορεί να χρησιμοποιεί δύο εφαρμογές ταυτόχρονα.
- Ευκολότερη πρόσβαση σε εφαρμογές: Στο εξής ο χρήστης θα μπορεί να μεταβαίνει άμεσα σε εφαρμογές από τον browser χωρίς να χρειάζεται αποθήκευση.

Επιπλέον, τα λεγόμενα «notification dots» προσφέρουν άμεση πρόσβαση στις ενημερώσεις και θα μπορούν εξίσου εύκολα να αποκρύπτονται. Επίσης, το Android Oreo παρέχει αυξημένη ασφάλεια και βελτιωμένη διαχείριση της μπαταρίας και 60 νέα Emojis.

2.3.16 Android 9.0 Pie

Εικόνα 17: Android 9.0 Pie

Στις 7 Αυγούστου 2018, κυκλοφόρησε η επίσημη ενημέρωση έκδοσης 9.0 [2] για το Android.

Η συγκεκριμένη έκδοση περιλαμβάνει τα παρακάτω χαρακτηριστικά βελτιώνοντας και το παρεχόμενο γραφικό περιβάλλον:

- Τη λειτουργία Adaptive Battery, η οποία δίνει προτεραιότητα στην κατανάλωση της μπαταρίας μόνο για τις εφαρμογές και τις υπηρεσίες που χρησιμοποιεί

περισσότερο ο χρήστης, κλείνοντας ταυτόχρονα όλες τις υπόλοιπες εφαρμογές που "τρέχουν" παράλληλα.

- Τη λειτουργία Adaptive Brightness, η οποία μαθαίνει πως να προσαρμόζει την φωτεινότητα της οθόνης ανάλογα με το περιβάλλον.
- Τα App Actions που θα βοηθούν στην πραγματοποίηση της επόμενης ενέργειας του χρήστη πιο γρήγορα επειδή θα αντιλαμβάνονται τι ακριβώς θέλει να κάνει.
- Η εμφάνιση και η εμπειρία χρήσης γίνεται πιο εύκολη με το νέο σύστημα πλοήγησης.
- Το Android Dashboard, το οποίο θα δείχνει στον χρήστη αναλυτικά πόσο χρόνο ξοδεύει σε συγκεκριμένες εφαρμογές, πόσες φορές, ξεκλείδωσε το κινητό τηλέφωνο και πόσες ειδοποιήσεις έλαβε κατά τη διάρκεια της ημέρας.
- Το App Timer είναι ένα εργαλείο που θα του επιτρέπει να θέτει χρονικό περιορισμό στην χρήση μιας εφαρμογής, θα τον ειδοποιεί όταν φτάνει στο όριο που έχει θέσει και θα μετατρέπει το εικονίδιο της εφαρμογής σε ασπρόμαυρο για να του υπενθυμίσει
- Η λειτουργία Wind Down θα ενεργοποιεί το Night Light όταν σκοτεινιάζει, θα βάζει την συσκευή σε Do Not Disturb mode και θα μετατρέπει την οθόνη σε ασπρόμαυρη την ώρα που έχεις δηλώσει ότι βρίσκεσαι στο κρεβάτι.

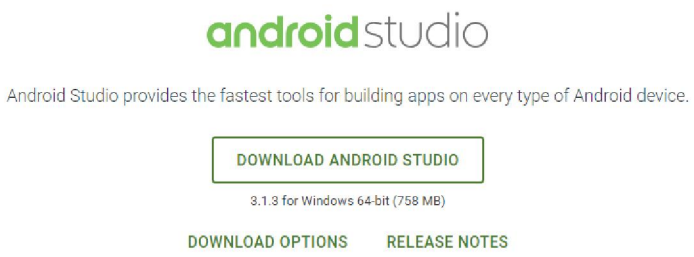
3. ΚΕΦΑΛΑΙΟ :ΒΗΜΑΤΑ ΑΝΑΠΤΥΞΗΣ ΕΦΑΡΜΟΓΗΣ

Το κεφάλαιο αυτό περιλαμβάνει τα βήματα ανάπτυξης εφαρμογής [18] ξεκινώντας από την εγκατάσταση του προγραμματιστικού περιβάλλοντος ανάπτυξης [17] ενώ στην συνέχεια περιγράφεται η διαδικασία δημιουργίας εικονικής συσκευής για την δοκιμαστική φάση της [19].

3.1 Εγκατάσταση Android Studio

Το Android Studio είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (integrated development environment ή IDE) που παρέχει ένα σύνολο βιβλιοθηκών για τον προγραμματισμό και την ανάπτυξη εφαρμογών για κινητά τηλέφωνα με λειτουργικό σύστημα android [16]. Το περιβάλλον ανάπτυξης android studio είναι χτισμένο σε IntelliJIDEA Έκδοση της Κοινότητας, το δημοφιλές JavaIDE από τον JetBrains.
Η επιλογή του android studio ως περιβάλλον ανάπτυξης εφαρμογής στηρίζεται στις δυνατότητες που παρέχει. Συγκεκριμένα δίνει την δυνατότητα για:

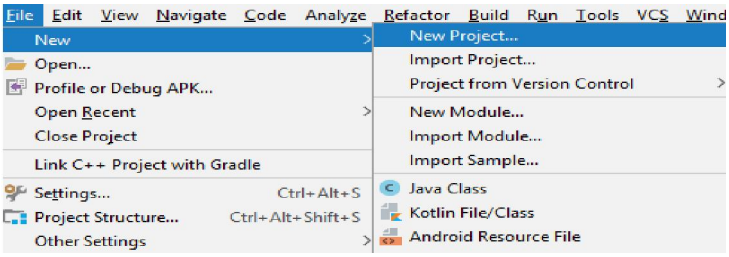
- Ευέλικτο Gradle-based σύστημα κατασκευής.
- Διευρυμένο πρότυπο υποστήριξης για τις υπηρεσίες της Google και τα διάφορα είδη της συσκευής.
- Πλούσιο επεξεργαστή διάταξης με υποστήριξη για το θέμα μοντάζ.
- Lint εργαλεία για να πιάσει την απόδοση, τη χρηστικότητα, τη συμβατότητα έκδοση, και άλλα προβλήματα.
- Proguard και app-υπογραφή δυνατότητες.
- Παρέχει ενσωματωμένη υποστήριξη για την πλατφόρμα της Google Cloud, καθιστώντας εύκολη την ενσωμάτωση του Google Cloud Messaging και App Engine.



Εικόνα 18: Μονοπάτι για κατέβασμα του android studio.

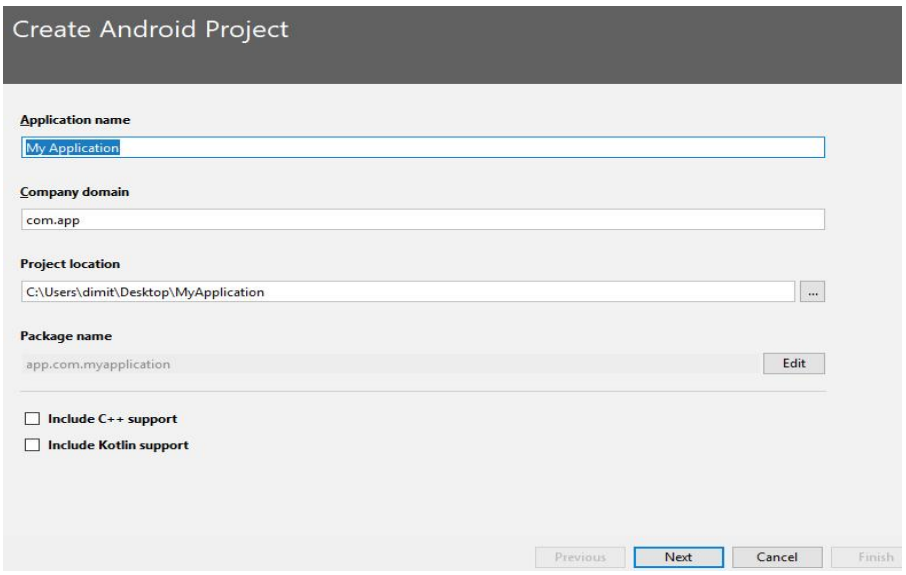
3.2 Ανάπτυξη-Δημιουργία του Project

Παρακάτω δίνεται μια αναλυτική περιγραφή των βασικών ενεργειών που πρέπει να εκτελεστούν για την δημιουργία εφαρμογής, μέσω της χρήσης του android studio. Αρχικά για την δημιουργία του project επιλέγεται από την μπάρα επιλογών File>NewProject, όπως φαίνεται στην **Σφάλμα! Το αρχείο προέλευσης της αναφοράς δεν βρέθηκε.**.



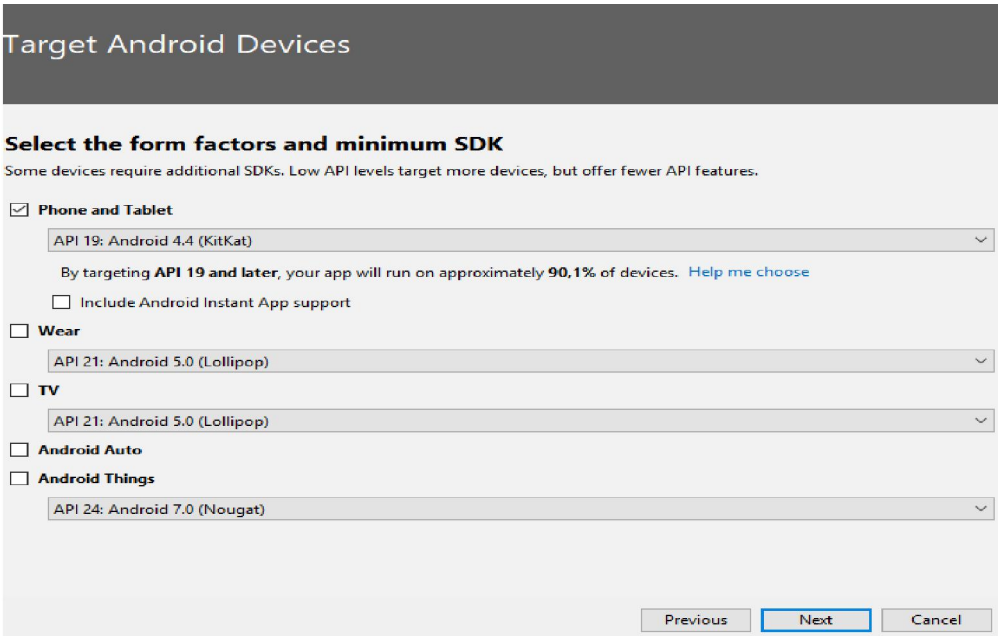
Εικόνα 19: Έναρξη δημιουργίας νέου project

- Έπειτα ορίζεται η ονομασία της εφαρμογής και η τοποθεσία αυτής δηλαδή που θα αποθηκευθεί, όπως φαίνεται στην **Σφάλμα! Το αρχείο προέλευσης της αναφοράς δεν βρέθηκε.** και ορίζονται τα παρακάτω πεδία:
  - Στο πεδίο Application Name γίνεται ορισμός του ονόματος της εφαρμογής θα εμφανιστεί στον χρήστη.
  - Στο πεδίο Company Domain ορίζεται το όνομα του πακέτου του έργου .
  - Στο πεδίο Package Name ορίζεται το πλήρες όνομα πακέτου έργου.
  - Στο πεδίο Project Location πρέπει να οριστεί ο κατάλογος όπου θα αποθηκευτεί το έργο .



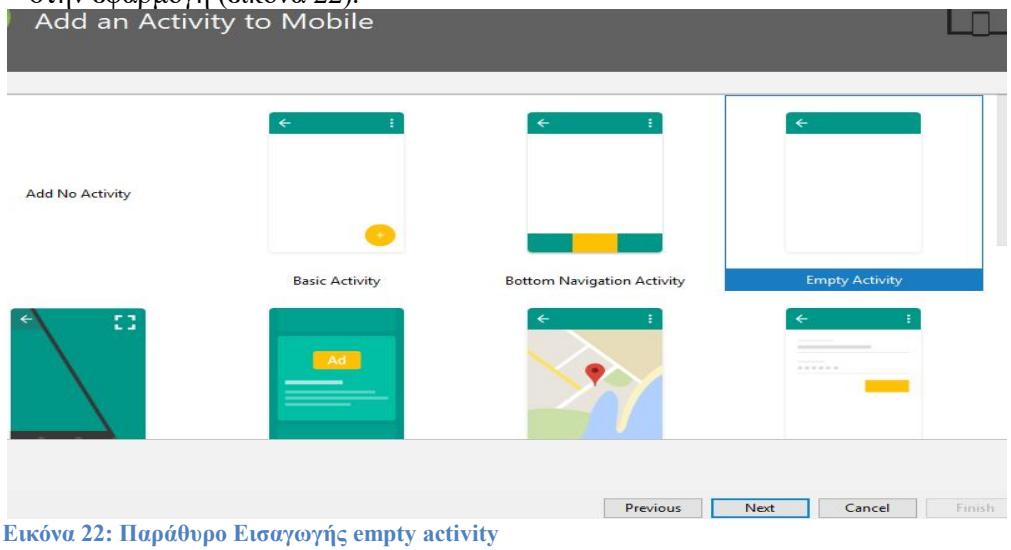
Εικόνα 20: Παράθυρο ορισμού Ονομασίας & τοποθεσίας της εφαρμογής

- Το επόμενο παράθυρο (εικόνα 21) επιτρέπει την επιλογή δημιουργίας των συντελεστών φόρμας της συσκευής και την ελάχιστη έκδοση υποστήριξης για καθένα από αυτά. Για κάθε επιλεγμένη συσκευή, ο οδηγός προσθέτει μια αντίστοιχη ενότητα στο έργο.
  Κάθε ενότητα περιέχει όλο τον κώδικα και τους πόρους που θα ενσωματωθούν σε ένα πακέτο εφαρμογών Android (APK) για την αντίστοιχη συσκευή. Αν αργότερα χρειαστεί να προστεθεί υποστήριξη για μια νέα συσκευή, προστίθεται μια ενότητα εκείνη τη στιγμή. Επίσης κώδικες και πόροι μπορούν να μοιραστούν μεταξύ των ενοτήτων χρησιμοποιώντας μια βιβλιοθήκη Android.



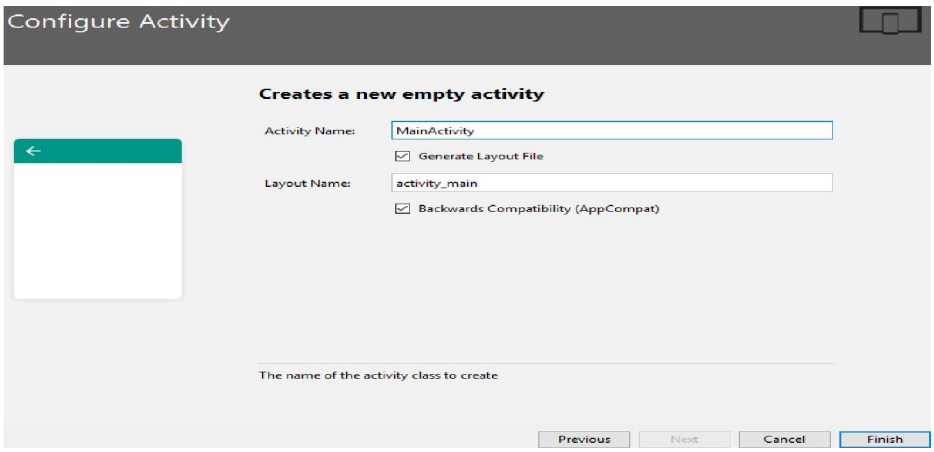
Εικόνα 21: Target android devices

- Η επόμενη οθόνη επιτρέπει την επιλογή ενός τύπου δραστηριότητας που θα προστεθεί στην εφαρμογή (εικόνα 22).



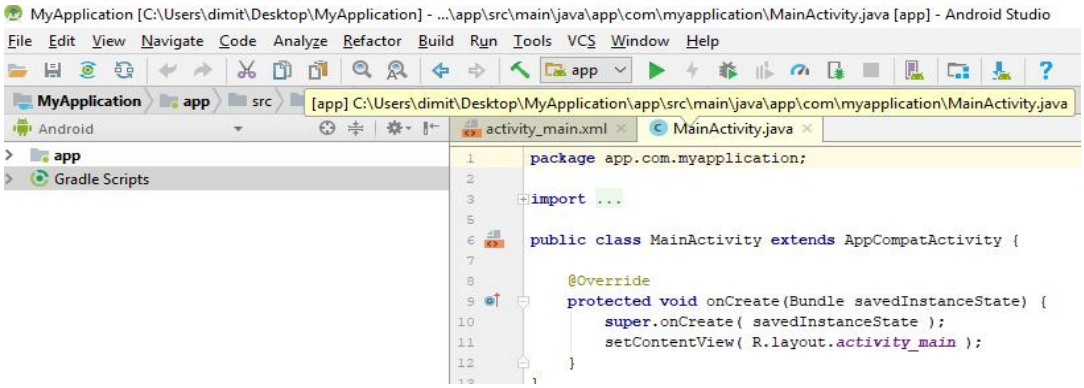
Εικόνα 22: Παράθυρο Εισαγωγής empty activity

- Η επόμενη οθόνη επιτρέπει την ρύθμιση της πρώτης δραστηριότητας που προστίθεται στην εφαρμογή. Μπορεί να εισαχθεί το όνομα δραστηριότητας, το όνομα της διάταξης και ο τίτλος, όπως φαίνεται από την εικόνα 23. Η διαδικασία δημιουργίας του έργου ολοκληρώνετε κάνοντας κλικ στο Τέλος.



Εικόνα 23: Activity Name

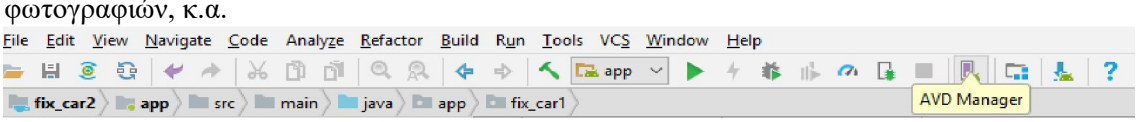
- Τέλος το Android Studio ρυθμίζει το έργο και ανοίγει το IDE, όπως απεικονίζεται στην εικόνα 24.



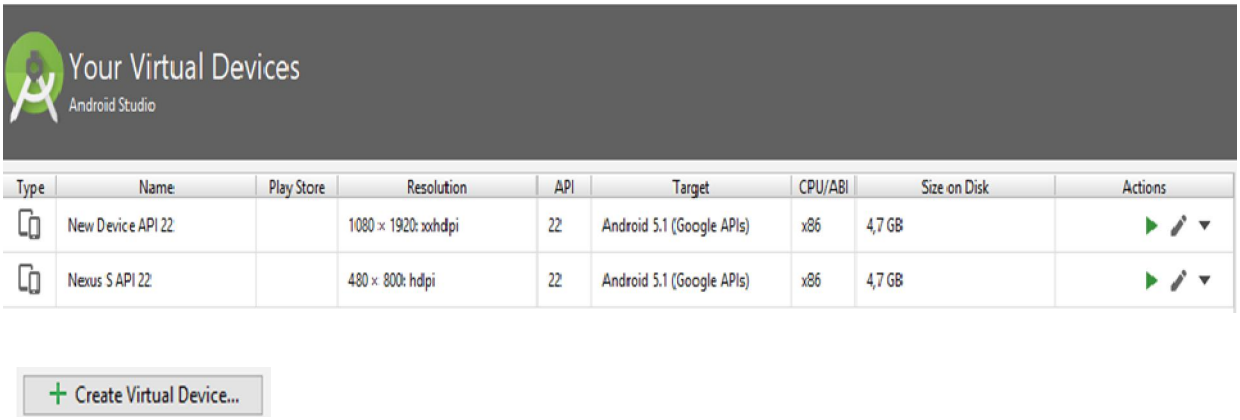
Εικόνα 24: Window Project

3.3 Δημιουργία Εικονικής Συσκευής για δοκιμαστική φάση

Για την δοκιμή της εφαρμογής και την εμφάνιση των αποτελεσμάτων της χρειάζεται κάποιος προσομοιωτής κινητού τηλεφώνου. Τη λύση δίνει μια εικονική συσκευή Android Virtual Device ή AVD η οποία ουσιαστικά αποτελεί προσομοιωτή ενός κινητού τηλεφώνου με android [19], από όπου ρυθμίζονται οι παράμετροι 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.



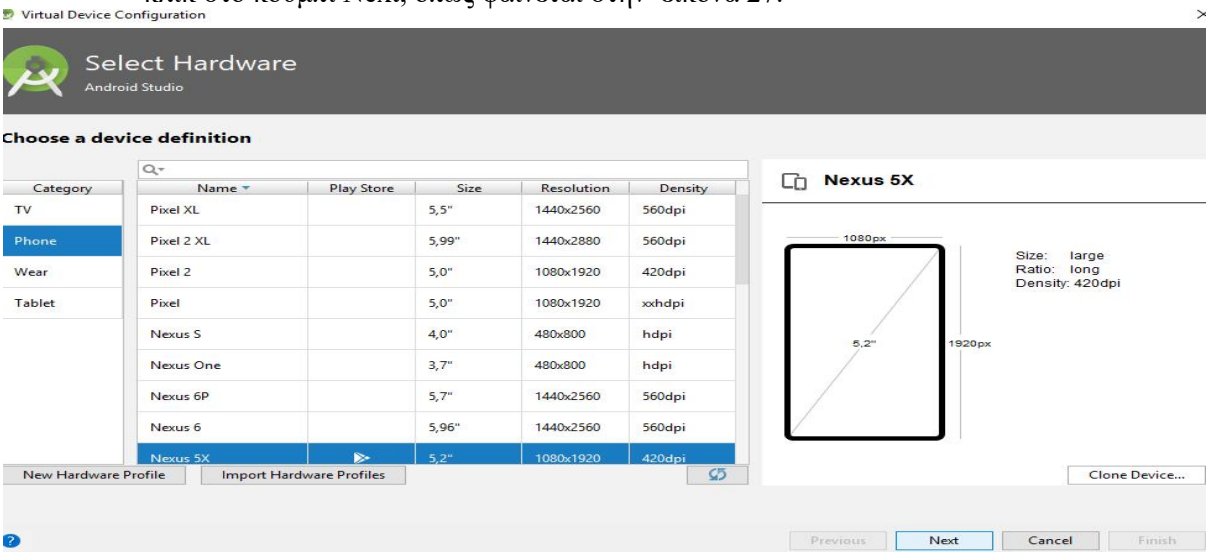
Εικόνα 25: Android Virtual Device Manager



Εικόνα 26: Shows ready devices on AVD

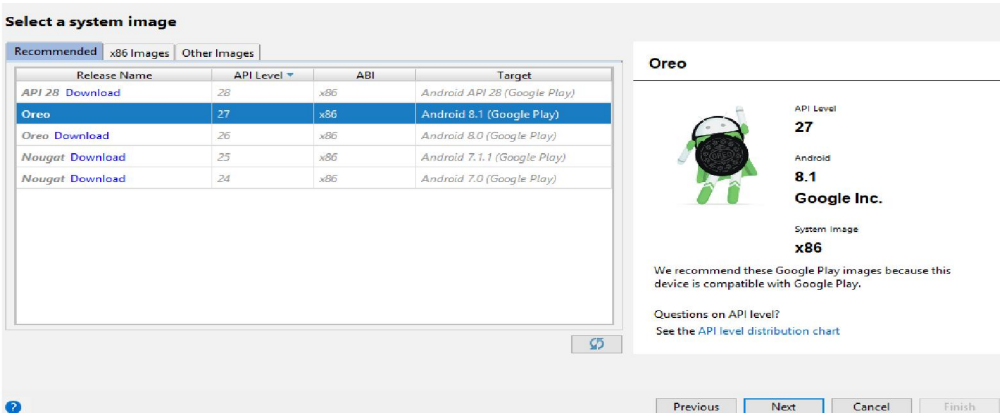
Ακολουθούν τα Παράθυρα παραμετροποίησης της AVD:

1) Πρώτο βήμα είναι η επιλογή συσκευής και μεγέθους οθόνης αυτής και στη συνέχεια κλικ στο κουμπί Next, όπως φαίνεται στην εικόνα 27.



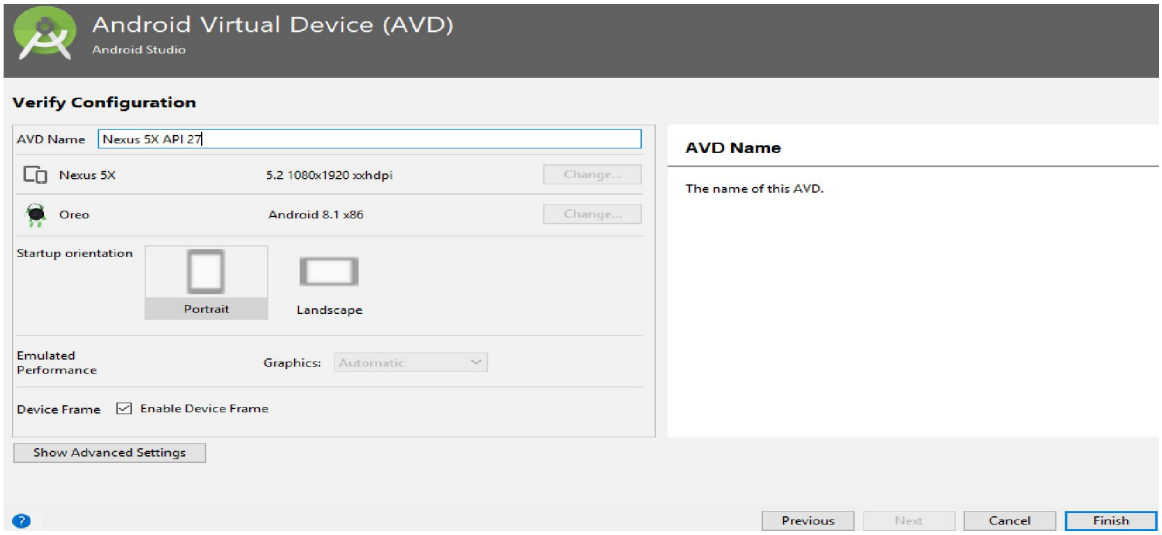
Εικόνα 27: Επιλογή Hardware

2) Επιλογή εικόνας έναρξης του συστήματος και επιλογή της έκδοση και του είδους των APIs που θα χρησιμοποιηθούν για να την ανάπτυξη της εφαρμογή, όπως φαίνεται στην εικόνα 28 .



Εικόνα 28: Επιλογή εικόνας συστήματος

3) Τέλος ορίζεται το όνομα για τον εξομοιωτή , τον αρχικό προσανατολισμό και ένα πλαίσιο γύρω από αυτό, όπως φαίνεται στην εικόνα 29.



Εικόνα 29: AVD Name

3.4 Δημοσίευση-Play Store

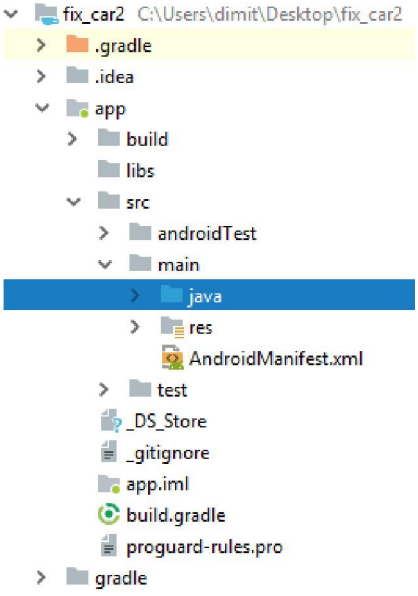
Το **Google Play**, παλαιότερα γνωστό ως Android Market, είναι μια ψηφιακή εφαρμογή διανομής για το Android [20], το οποίο αναπτύσσεται και συντηρείται από τη Google . Η υπηρεσία επιτρέπει στους χρήστες να περιηγηθούν και να κατεβάσουν μουσική, περιοδικά, βιβλία, ταινίες, παιχνίδια, τηλεοπτικά προγράμματα και **εφαρμογές** που δημοσιεύθηκαν μέσω του Google. Άρα μετά την δημιουργία της, μια εφαρμογή μπορεί να ενταχθεί στο Play Store έτσι ώστε να είναι διαθέσιμη προς εγκατάσταση και σε άλλους χρήστες.

4. ΚΕΦΑΛΑΙΟ : ΟΡΓΑΝΩΣΗ ΕΦΑΡΜΟΓΗΣ

Στο κεφάλαιο που ακολουθεί γίνεται περιγραφή της οργάνωσης της εφαρμογής που πραγματεύεται η παρούσα πτυχιακή, η οποία δημιουργήθηκε σύμφωνα με τα βήματα ανάπτυξης που διατυπώθηκαν στο προηγούμενο κεφάλαιο.

4.1 Δομή

Στην ενότητα Project Explorer εμφανίζεται το project που δημιουργήθηκε καθώς και τα περιεχόμενα-αρχεία αυτού. Το όνομα της εφαρμογής της παρούσας πτυχιακής είναι Fix_car2 (εικόνα 30), το όνομα του πακέτου είναι app.fix_car1, ενώ στην εικόνα 30 φαίνεται και η τοποθεσία που είναι αποθηκευμένη.

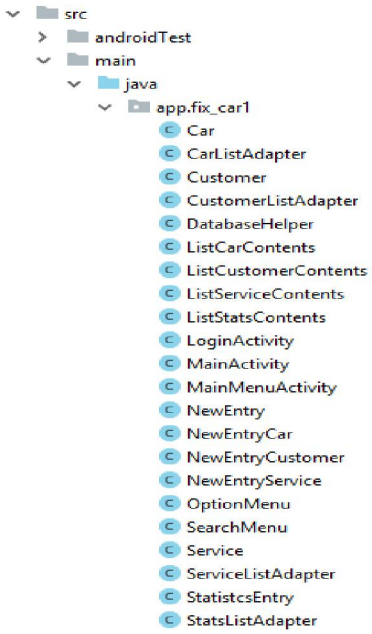


Εικόνα 30: Περιεχόμενα του Project

Ακολουθεί η εξήγηση των παρακάτω φακέλων και αρχείων καθώς και του περιεχόμενού τους [21]:

4.2 Φάκελος src

Το όνομα του φακέλου προέρχεται από την αγγλική λέξη source που σημαίνει πηγή. Εδώ περιέχεται όλος ο πηγαίος κώδικας και συγκεκριμένα όλα τα αρχεία Java της εφαρμογής. Όλες οι κλάσεις βρίσκονται κάτω από το όνομα του πακέτου (app.fix_car1)[21]. Έπειτα στο πακέτο αυτό περιέχονται όλες οι κλάσεις της εφαρμογής μας, όπως φαίνεται στην εικόνα 31. Οι κλάσεις χωρίζονται σε activities, δηλαδή στις διαφορετικές οθόνες και στις υπόλοιπες που εκτελούν κάποια ειδική λειτουργία.



Εικόνα 31: Απεικόνιση του φακέλου των java κλάσεων

4.3 Φάκελος gen

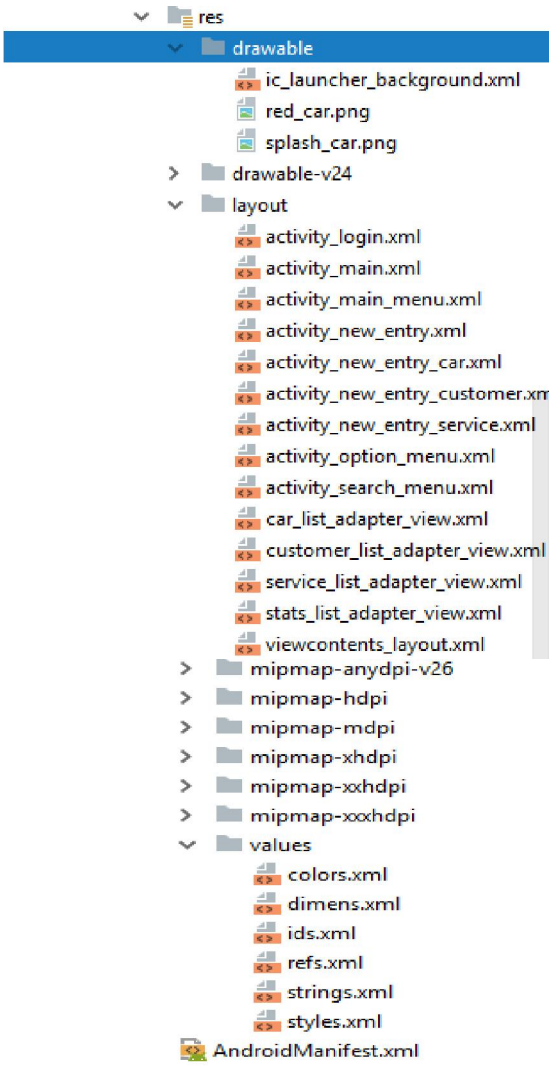
Στον φάκελο gen περιέχονται αρχεία Java εξίσου, όμως η διαφορά με τον προηγούμενο φάκελο είναι ότι αυτά παράγονται αυτόματα από το Android Development Tools (ADT). Αναφέρονται σε πολλές πηγές που υπάρχουν μέσα στην εφαρμογή, όπως η ειδική κλάση R[21].

4.4 Φάκελος bin

Εδώ βρίσκονται τα αρχεία που παράγονται μετά την μεταγλώττιση του πηγαίου κώδικα. Συγκεκριμένα, το τελικό αρχείο .apk το οποίο χρησιμοποιείται για την εγκατάσταση της εφαρμογής στις συσκευές [21]. Αξίζει να σημειωθεί ότι το αρχείο αυτό από μόνο του δεν μπορεί να δημοσιευτεί στο Google Play, γιατί απαιτείται να φέρει ψηφιακή υπογραφή.

4.5 Φάκελος res

Ο φάκελος res (resources) που απεικονίζεται στην εικόνα 32 περιέχει τους πόρους της εφαρμογής [21], οι οποίοι μπορεί να είναι αρχεία XML, εικόνες, ήχοι, animation και string τιμές. Ο μεταγλωττιστής πόρων (resource compiler) κατηγοριοποιεί τους πόρους βάσει του τύπου τους, για παράδειγμα οι εικόνες βρίσκονται στον υποφάκελο drawable.



Εικόνα 32: Απεικόνιση του φακέλου res και των υποφακέλων της

4.5.1 Υποφάκελος res/drawable

Στον υποφάκελο drawable βρίσκονται κυρίως τα αρχεία που είναι στατικά γραφικά [21], όπως για παράδειγμα εικόνες, σχήματα και animation. Το Android υποστηρίζει αρχεία εικόνων τύπου PNG, JPG και GIF, και αρχεία XML τα οποία ορίζουν σχήματα, χρώματα, διαγράμματα, σχέδια που αλλάζουν το μέγεθός τους ανάλογα με την οθόνη και εικόνες που εναλλάσσουν την κατάστασή τους σε καθορισμένα γεγονότα.

4.5.2 Υποφάκελος res/layout

Η χρήση των αρχείων XML είναι ο ευκολότερος τρόπος για την απεικόνιση γραφικών στοιχείων στην εφαρμογή [21]. Μεταγλωττίζονται σε διατάξεις οθόνης ή μέρους της οθόνης. Αυτός ο τρόπος, υπερτερεί της κλασσικής δημιουργίας αντικειμένων προγραμματιστικά επειδή διαχωρίζει τον πηγαίο κώδικα που αφορά τις λειτουργίες της εφαρμογής με τον σχεδιασμό της και γίνεται εύκολη η αλλαγή της εμφάνισης της εφαρμογής χωρίς να ασχοληθούμε με τον πηγαίο κώδικα.

Ο σχεδιασμός υλοποιείται εύκολα και γρήγορα καθώς η διάταξη των αντικειμένων ορίζεται παρόμοια με την φιλοσοφία δημιουργίας ιστοσελίδων με τη γλώσσα HTML. Τα αρχεία αυτά αποθηκεύονται στον υποφάκελο res/layout και αποτελούν την γενική μορφοποίηση της Εφαρμογής.

4.5.3 Υποφάκελος res/values

Ο υποφάκελος values περιλαμβάνει τους πόρους της εφαρμογής [21]. Μέσα στο φάκελο res/values υπάρχει το αρχείο string.xml όπου εμφανίζονται τα ονόματα των πόρων String αυτής της εφαρμογής με τις αντίστοιχες τιμές τους, και το αρχείο styles.xml όπου ορίζονται τα στυλ πχ για τα πεδία κειμένου.

4.6 Android Manifest

Το αρχείο AndroidManifest αποτελεί το σημαντικότερο αρχείο της εφαρμογής [21]. Σε αυτό ο προγραμματιστής ορίζει τις απαιτήσεις της, δηλώνει τα δικαιώματα που απαιτούνται και καθορίζει ποιες εκδόσεις θα μπορούν να την τρέχουν. Κάθε εφαρμογή πρέπει να έχει αυτό το αρχείο για να μπορεί να γίνει εκτελέσιμη. Εδώ, δηλώνονται επίσης οι δραστηριότητες (activities) μαζί με τις ενέργειές τους. Με την δημιουργία κάθε νέου project, δηλώνεται αυτόματα η activity της κεντρικής οθόνης (main screen).
Παρακάτω απεικονίζεται ένα παράδειγμα αρχείου AndroidManifest:

```
<?xmlversion="1.0"encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="app.fix_car1">
      <uses-permission-sdk-23 android:name="android.permission.ACCESS_CHECKIN_PROPERTIES" />
  <application
      android:allowBackup="true"
      android:icon="@drawable/red_car"
      android:label="@string/app_name"
      android:supportsRtl="true"
      android:theme="@android:style/Theme.Holo.Light.DarkActionBar">
      <activity android:name=".MainActivity"
         android:label="@string/app_name"
         android:noHistory="true">
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
      <activity  android:name=".MainMenuActivity"
         android:label="@string/title_activity_main_menu"></activity>
//ομοίως περιλαμβάνονται  και οι υπόλοιπες activities//
</application>
</manifest>
```

5. ΚΕΦΑΛΑΙΟ:ΥΛΟΠΟΙΗΣΗ ΚΑΙ ΠΑΡΟΥΣΙΑΣΗ ΤΟΥ ΓΡΑΦΙΚΟΥ ΠΕΡΙΒΑΛΛΟΝΤΟΣ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

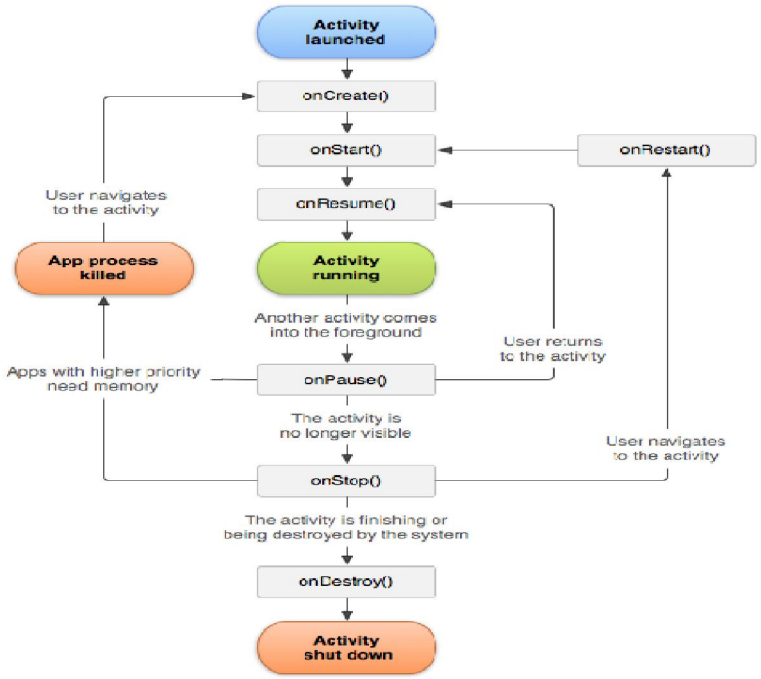
Σε αυτό το κεφάλαιο αρχικά γίνεται αναφορά στον σκοπό της εφαρμογής και των δυνατοτήτων που αυτή προσφέρει. Έπειτα ακολουθεί περιγραφή όλων των διαφορετικών οθονών της εφαρμογής, ο τρόπος σύνδεσης μεταξύ τους, οι κλάσεις στις οποίες υλοποιούνται και όλες οι δυνατές επιλογές που έχει ο χρήστης όταν βρίσκεται σε συγκεκριμένη οθόνη. Τα διάφορα στιγμιότυπα της εφαρμογής παρουσιάζονται σε εικόνες και έπειτα εξηγούνται οι λειτουργίες ενώ εμπεριέχονται κομμάτια κώδικα από τα σημαντικότερα αρχεία της εφαρμογής για πιο εύκολη κατανόηση αυτής.

5.1 Σκοπός της εφαρμογής.

Ο σκοπός της εργασίας αυτής είναι η ανάπτυξη μιας εύχρηστης εφαρμογής που θα επιτρέπει σε υπαλλήλους γραμματειακής υποστήριξης σε φανοποιεία-βαφεία να εισάγουν και να αποθηκεύουν στοιχεία πελατών και των αυτοκινήτων τους. Επιπροσθέτως η εφαρμογή επιτρέπει στους χρήστες της να έχουν άμεση πρόσβαση στα στοιχεία που έχουν καταγραφεί σε αυτή μέσω μιας γρήγορης μηχανής αναζήτησης. Τέλος μια ακόμα επιλογή που παρέχει η εφαρμογή είναι η εμφάνιση στατιστικών στοιχείων με το πάτημα ενός κουμπιού, επιτρέποντας στον χρήστη να παρακολουθεί την πορεία της επιχείρησης σύμφωνα με τα στατιστικά στοιχεία που προκύπτουν από τα service που πραγματοποιήθηκαν.

5.2 Βασικές διαδικασίες

Το βασικότερο στοιχείο της εφαρμογής είναι η δημιουργία των activities. Η κάθε μια δραστηριότητα αποτελεί μια διαφορετική οθόνη της εφαρμογής με την οποία ο χρήστης αλληλεπιδρά [24] [25]. Σε αυτή φορτώνεται το γραφικό περιβάλλον το οποίο βλέπει ο χρήστης. Κάθε activity μπορεί να καλέσει κάποια άλλη. Η πρώτη οθόνη που θα δει ο χρήστης όταν τρέξει την εφαρμογή αποτελεί την κεντρική activity[26], από την οποία προέρχονται όλες οι υπόλοιπες. Όταν η εφαρμογή καλεί νέα activity ο χρήστης βλέπει την νέα οθόνη και με το πλήκτρο cancel μπορεί να επιστρέψει στην προηγούμενη(LIFO).



Εικόνα 33: Κύκλος ζωής των activities και οι αντίστοιχοι μέθοδοι[25]

Η υλοποίηση της εφαρμογής πραγματοποιήθηκε με την δημιουργία των παρακάτω βασικών activities:

- Main Activity η οποία εισάγει τον χρήστη στην εφαρμογή.
- Login Activity όπου ο χρήστης ορίζει τα στοιχεία εισόδου του.
- Main Menu Activity η οποία αποτελεί το βασικό μενού επιλογών της εφαρμογής (New, search, statistics, options, exit).
- New Entry που παρέχει τις επιλογές ως προς το είδος των στοιχείων για εισαγωγή (New Car, New Customer, Manage, Cancel).
- New Entry Customer όπου ο χρήστης εισάγει τα στοιχεία των πελατών.
- New Entry Car όπου ο χρήστης εισάγει τα στοιχεία του αυτοκινήτου.
- New Entry Service όπου ο χρήστης εισάγει τα στοιχεία των service.
- Search Menu όπου πραγματοποιείται γρήγορη αναζήτηση στοιχείων.
- Statistics που εμφανίζει στατιστικά στοιχεία χρήσιμα για να παρακολουθείται η πορεία της επιχείρησης.
- Option Menu όπου περιέχει τις ρυθμίσεις του χρήστη.

Κάθε activity υλοποιεί την μέθοδο onCreate() η οποία καλείται την στιγμή που ξεκινάει activity[26]. Επίσης μέσα στην μέθοδο αυτή πρέπει να δημιουργηθεί η διεπαφή χρήστη, η οποία δηλώνεται με την εντολή setContentView() και η findViewByld() για να ανακτήσει τα διάφορα widgets προγραμματιστικά. Για να ξεκινήσουμε μια activity B μέσα από μια activity A χρησιμοποιούμε τις παρακάτω εντολές: intent intent = new Intent (this, MyActivityB class); startActivity(intent); . Ενώ για να τερματίσουμε μια activity αρκεί η εντολή finish.

Σε κάθε οθόνη της εφαρμογής πρωταρχικό στοιχείο του γραφικού περιβάλλοντος αποτελεί η διάταξη των γραφικών στοιχείων ή layout [27]. Το layout περιλαμβάνει όλα τα γραφικά στοιχεία της οθόνης τα οποία μπορεί να είναι διατεταγμένα σε επιμέρους layouts. Το παράθυρο του διαλόγου είναι το μοναδικό με το οποίο μπορεί να αλληλεπιδράσει ο χρήστης. Χρησιμοποιείται είτε για να ορίσει ο χρήστης κάποια επιλογή του είτε για να ενημέρωση του χρήστη για κάποιο γεγονός. Τα δυο πιο σημαντικά είδη είναι ο διάλογος ειδοποίησης και ο διάλογος προόδου**.**

5.3 Γραφικά Περιβάλλοντα Εφαρμογής

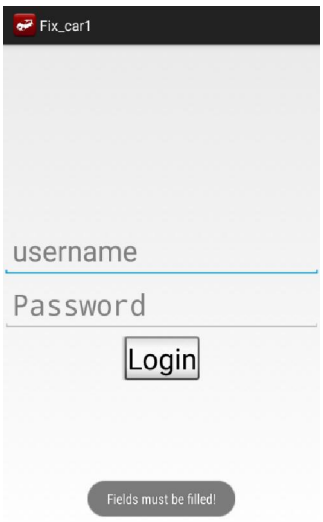
5.3.1 Κεντρική οθόνη

Εικόνα 34:Κεντρική οθόνη

Μετά την ολοκλήρωση εγκατάστασης της εφαρμογής και αφού την επιλέξουμε για εκκίνηση εμφανίζεται η κεντρική οθόνη της (εικόνα 34). Όσον αφορά την **λειτουργία** της που έχει καθοριστεί από την MainActivity.java κλάση , το κουμπί (btenter) που έχει τοποθετηθεί σε αυτή ενεργοποιείται με την μέθοδο **setOnClickListener(new** View.OnClickListener()..) και μόλις το πατήσουμε αν η εφαρμογή τρέχει για πρώτη φορά μεταβαίνει στην οθόνη Login , click = **new** Intent(MainActivity.**this**, LoginActivity.**class**); αλλιώς μεταβαίνουμε στο βασικό μενού της εφαρμογής.

Σχετικά με την **διάταξη** αυτής όπως φαίνεται στο αρχείο activity_main.xml επιλέχθηκε η LinearLayout . Επιπροσθέτως πραγματοποιήθηκε τοποθέτηση ενός imageView (εικόνας) που νωρίτερα έχουμε εισάγει στο φάκελο drawable και ενός button όπου ιδιότητες αυτού ορίσαμε id:button_enter, TextView:Enter, **style="@android:style/Widget.Button.Inset"** και textAppearance="@style/TextAppearance.AppCompat.Display1" όπου οι δυο τελευταίες ιδιότητες θα είναι κοινές και στα button των υπόλοιπων οθονών. Γενικά σε κάθε εργαλείο που επιλέγουμε από την παλέτα για να το Design της οθόνης πρέπει να ορίζουμε και τις ιδιότητες αυτού. Τέλος σημαντική εντολή για τον καθορισμό της στοίχισης των imageview &button είναι η **android:layout_gravity="center_horizontal".**

5.3.2 Οθόνη Login



Εικόνα 35:Είσοδος χρήστη

Η λειτουργία της οθόνης που απεικονίζεται στην εικόνα 35, όπως τέθηκε στην LoginActivity.java κλάση είναι ο ορισμός των στοιχείων της διεπαφής του χρήστη και εν' συνεχεία η μετάβαση στο μενού της εφαρμογής. Συγκεκριμένα ο χρήστης θέτει το όνομα και τον κωδικό που θέλει για την είσοδο του στην εφαρμογή ,στοιχεία τα οποία κάνοντας κλικ στο κουμπί login αποθηκεύονται και ο χρήστης μεταβαίνει απευθείας στο μενού εφόσον έχει προηγηθεί έλεγχος ότι τα πεδία δεν είναι κενά . Στη περίπτωση που είναι κενά τα πεδία εμφανίζεται το μήνυμα WARN_EMPTY_FIELDS όπου ο χρήστης πρέπει να εισάγει username και password ,στοιχεία τα οποία αξίζει να επισημάνουμε ότι αποτελούν το κλειδί των κοινών προτιμήσεων δηλαδή αποθηκεύονται και με την επόμενη έναρξη του app ο χρήστης μεταβαίνει απευθείας από την αρχική οθόνη, στην οθόνη του μενού μη έχοντας ξεχάσει για να γίνει αυτό να προσθέσουμε την εντολή editor.putBoolean(**"firstRun"**, **false**); στην LoginActivity.java κλάση στην οποία ορίζονται οι λειτουργίες της οθόνης.

Για την σχεδίαση της οθόνης ορίσαμε στο login.xml την διάταξη RelativeLayout, προσθέσαμε ένα PlainText για το πεδίο username και ένα τύπου TextPassword για το password καθώς και ένα button με όνομα login. Για να εμφανίζεται στο κάθε πεδίο τι πρέπει να εισάγουμε αρκεί να το σημειώσουμε στο πεδίο hint στις ιδιότητεςhint:username.

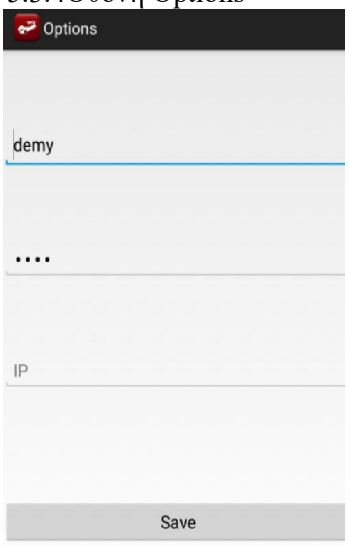
5.3.3 Οθόνη Βασικού Menu

Εικόνα 36:Βασικό Menu

Στην οθόνη που απεικονίζεται στην εικόνα 36 ο χρήστης διαλέγει από το βασικό μενού της εφαρμογής την λειτουργία που θέλει να εκκινήσει. Συγκεκριμένα πατώντας το κουμπί New μεταβαίνει στο μενού NewEntry όπου επιλέγει τι πληροφορίες θα εισάγει. Δεύτερη επιλογή του Main Menu είναι το search ,όπου κάνοντας κλικ πάνω σ' αυτό το κουμπί μεταβαίνουμε στην οθόνη αναζήτησης καταχωρημένων πληροφοριών. Τρίτη επιλογή είναι το κουμπί Statistics όπου με ένα κλικ πάνω του εμφανίζεται απευθείας μια οθόνη με στατιστικά στοιχεία. Τέταρτο στην ιεραρχία του μενού είναι η επιλογή Options, όπου επιλέγοντας την ο χρήστης μεταβαίνει στις ρυθμίσεις του χρήστη. Τέλος κάνοντας κλικ στο κουμπί EXIT πραγματοποιείται έξοδος από την εφαρμογή. Στο σημείο αυτό αξίζει να σημειωθεί ότι χρήστης μπορεί να χρησιμοποιεί και το κουμπί της επιστροφής της εκάστοτε συσκευής προκειμένου να μεταβαίνει σε προηγούμενη οθόνη .

Σχετικά με την διάταξη (activity_main_menu.xml), η οθόνη Main Menu έχει πέντε κουμπιά που δείχνουν τις επιλογές που έχει ο χρήστης με βάση την εφαρμογή. Αφού ο χρήστης πατήσει την επιλογή που θέλει, μεταβαίνει αυτόματα στην αντίστοιχη οθόνη της δραστηριότητας που διάλεξε να πραγματοποιήσει. Ταυτόχρονα έχει την δυνατότητα εναλλαγής οθονών. Η διάταξη είναι LinearLayout με orientation=vertical(κάθετη) και η διαδικασία για την εισαγωγή και ρύθμιση των properties των Button είναι ιδία με τις προηγούμενες οθόνες με την μόνη διαφορά στην στοίχιση τους.

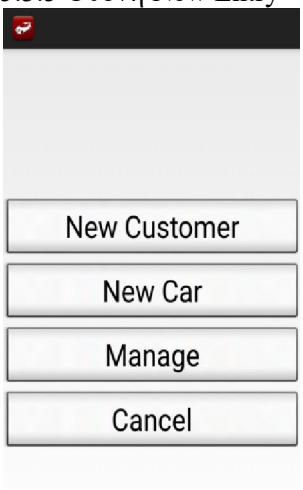
5.3.4Οθόνη Options



Εικόνα 37:Ρυθμίσεις

Ο χρήστης στην οθόνη της εικόνας 37 έχει την δυνατότητα να αλλάξει τις υπάρχουσες ρυθμίσεις διεπαφή χρήστη δηλαδή ρυθμίζει, Όνομα χρήστη ,Κωδικό χρήστη, IP χρήστη (προαιρετικά)και να τις αποθηκεύσει. Όσον αφορά την διάταξη της που διατυπώνεται στο activity_option_menu.xml αρχείο, τοποθετήθηκαν τρία πλαίσια συμπλήρωσης κειμένου και ένα κουμπί για την εκτέλεση της αποθήκευσης.

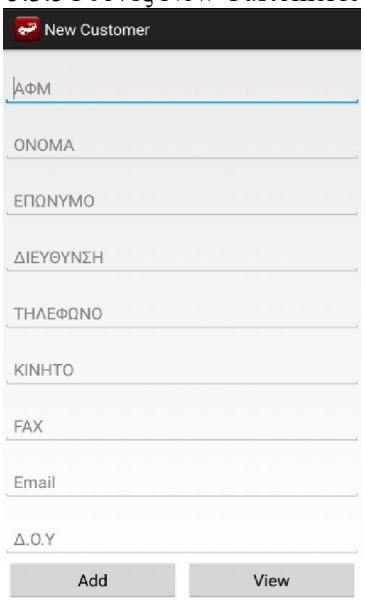
5.3.5 Οθόνη New Entry-> New Customer /New Car/Manage/Cancel



Εικόνα 38:Οθόνη επιλογής είδους στοιχείων προς εγγραφή

Η οθόνη new entry είναι το μενού επιλογών σχετικά με το είδος των στοιχείων που επιθυμεί να εισάγει ο χρήστης (στοιχεία των πελατών, αυτοκινήτων ή των service), όπως φαίνεται στην εικόνα 38. Η οθόνη έχει τέσσερα κουμπιά (activity_new_entry.xml). Πιο αναλυτικά κάνοντας κλικ στο new customer ανοίγει η οθόνη συμπλήρωσης των στοιχείων των πελατών, πατώντας το new car ο χρήστης μεταβαίνει στην οθόνη εγγραφής στοιχείων των αυτοκινήτων ενώ πατώντας το κουμπί manage ανοίγει η οθόνη για συμπλήρωση πληροφοριών σχετικά με τις επισκευές αυτών. Τέλος με το κουμπί cancel επιστρέφουμε στην προηγούμενη οθόνη.

5.3.5Οθόνες New Customer&Customerse



Εικόνα 39:Οθόνη εισαγωγής στοιχείων νέων πελατών

Στην εικόνα 39 απεικονίζεται η οθόνη όπου ο χρήστης έχει την δυνατότητα να εισάγει και να αποθηκεύει πατώντας το κουμπί add τα στοιχεία των πελατών, τα οποία εμφανίζονται αυτόματα σε μορφή λίστας πατώντας το κουμπί view .Η διαμόρφωση της διάταξης της οθόνης εισαγωγής των στοιχείων των πελατών ορίζεται από το activity_new_entry_customer.xml αρχείο το οποίο περιλαμβάνει εννέα πεδία προς συμπλήρωση και δύο κουμπιά. Όταν ο χρήστης πατήσει το κουμπί add αν τα πεδία δεν είναι συμπληρωμένα εμφανίζεται ένα μήνυμα που λέει ότι πρέπει να συμπληρωθούν αλλιώς εμφανίζεται μήνυμα ότι η αποθήκευση ήταν επιτυχής. Αν ο χρήστης πατήσει το

κουμπί view εμφανίζονται όλα τα αποθηκευμένα στοιχεία των πελατών(Πίνακας Customer).

Εικόνα 40:Οθόνη απεικόνισης των καταχωρημένων πελατών

Η διάταξη της οθόνης εμφάνισης των εισαχθέντων πληροφοριών, που απεικονίζεται στην εικόνα 40, ορίζεται απο τον συνδυασμό των αρχείων viewcontents_layout.xml όπου εισάγεται μια λίστα εμφάνισης για τα στοιχεία (**ListView**) και customer_list_adapter_view.xml, όπου ορίζεται η διάταξη της λίστας των στοιχείων να είναι οριζόντια και εισάγονται εννέα πεδία εμφάνισης κειμένου. Άρα τα στοιχεία κάθε πελάτη εμφανίζονται ανά σειρά ενώ τα στοιχεία είναι χωρισμένα ανά είδος σε εννέα στήλες.

5.3.6Οθόνη New Car

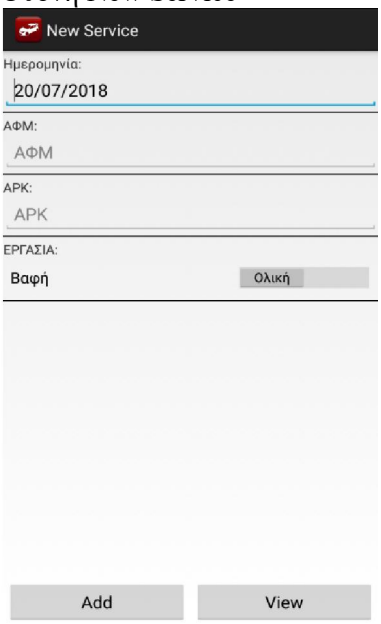
Εικόνα 41:Οθόνη εισαγωγής στοιχείων νέων αυτοκινήτων

Στην οθόνη New Car ο χρήστης εισάγει τα στοιχεία του αυτοκινήτου(εικόνα 41), τα οποία παραμένουν αποθηκευμένα και προβάλλονται κατά αίτηση του χρήστη, όπως απεικονίζονται στην εικόνα 42. Δίνεται δηλαδή στον χρήστη η δυνατότητα να εισάγει και να αποθηκεύει πατώντας το κουμπί add τα στοιχεία των αυτοκινήτων, τα οποία εμφανίζονται αυτόματα σε μορφή λίστας πατώντας το κουμπί view. Η διαμόρφωση της διάταξης της οθόνης εισαγωγής των στοιχείων των πελατών ορίζεται μέσα από το activity_new_entry_car.xml αρχείο. Η οθόνη έχει έξι πλαίσια συμπλήρωσης κειμένου και δύο κουμπιά. Όταν ο χρήστης πατήσει το κουμπί add αν τα πεδία δεν είναι συμπληρωμένα εμφανίζεται ένα μήνυμα που λέει ότι πρέπει να συμπληρωθούν αλλιώς εμφανίζεται μήνυμα ότι η αποθήκευση ήταν επιτυχής. Αν ο χρήστης πατήσει το κουμπί view εμφανίζονται όλα τα αποθηκευμένα στοιχεία των πελατών(πίνακας αυτοκινήτων).

Εικόνα 42:Οθόνη απεικόνισης των καταχωρημένων αυτοκινήτων

Η διάταξη της παραπάνω οθόνης δηλαδή της οθόνης εμφάνισης των εισαχθέντων πληροφοριών ορίζεται απο τον συνδυασμό των αρχείων viewcontents_layout.xml όπου εισάγεται μια λίστα για τα στοιχεία (**ListView**) και car_list_adapter_view.xml όπου τοποθετούνται έξι πλαίσια εμφάνισης κειμένου και η διάταξη της λίστας των στοιχείων είναι οριζόντια .

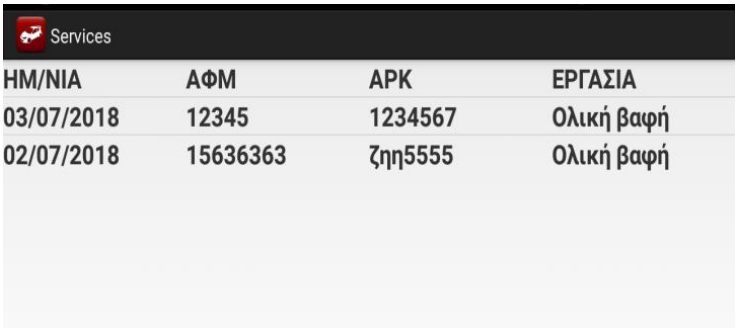
5.3.7Οθόνη New Service



Εικόνα 43:Οθόνη εισαγωγής επισκευών

Η NewService είναι η οθόνη εισαγωγής των στοιχείων των επισκευών, όπως απεικονίζεται στην εικόνα 43. Είναι εύχρηστη διότι η ημερομηνία ανανεώνεται αυτόματα και μέσω ενός διακόπτη επιλέγουμε εύκολα για εισαγωγή την εργασία που πραγματοποιήθηκε. Σύμφωνα με το αρχείο activity_new_entry_service.xml η οθόνη περιλαμβάνει τα εξής:

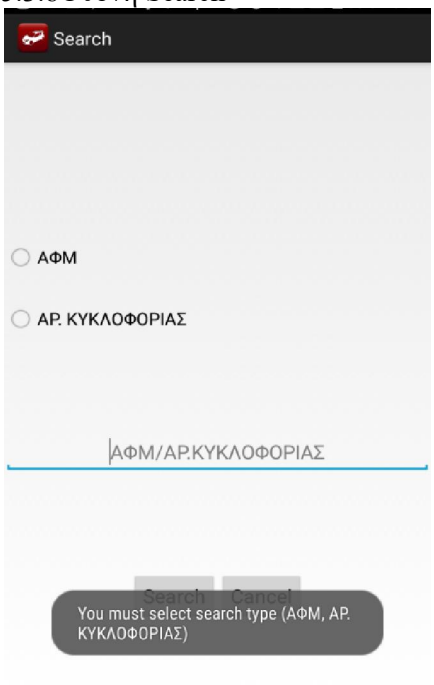
- Ένα πεδίο εμφάνισης κειμένου που είναι ενημερωμένο αυτόματα από την ημερομηνία,
- Δύο πεδία συμπλήρωσης κειμένου , του ΑΦΜ και του Αριθμού κυκλοφορίας
- Ένα πεδίο εμφάνισης κειμένου με το όνομα εργασία που περιλαμβάνει ένα διακόπτη(Switch) με το οποίο ο χρήστης μπορεί εύκολα να επιλέξει το είδος της εργασίας που πραγματοποιήθηκε(ολική ή τοπική βαφή)
- Δύο κουμπιά, ένα για την εισαγωγή και ένα για την εμφάνιση των στοιχείων.



Εικόνα 44:Οθόνη εμφάνισης επισκευών

Η διάταξη της οθόνης εμφάνισης των εισαχθέντων πληροφοριών (βλέπε εικόνα 44) σχετικά με τα service ορίζεται απο τον συνδυασμό των αρχείων viewcontents_layout.xml όπου εισάγεται μια λίστα για τα στοιχεία (**ListView**) και service_list_adapter_view.xml όπου ορίζεται η διάταξη της λίστας των στοιχείων να είναι οριζόντια και εισάγονται τέσσερα πεδία εμφάνισης κειμένου. Έτσι τα στοιχεία εμφανίζονται ανά σειρά ενώ τα στοιχεία είναι χωρισμένα ανά είδος σε τέσσερα columns.

5.3.8Οθόνη Search

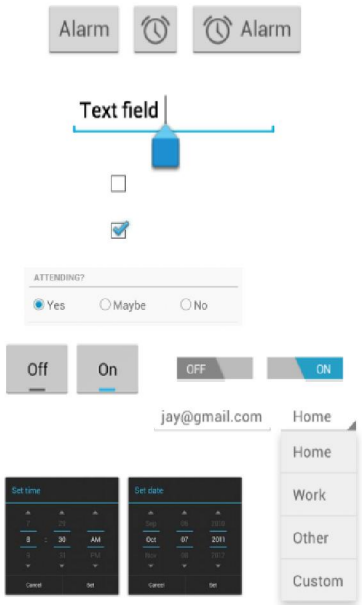


Εικόνα 45:Οθόνη αναζήτησης

Μια ακόμη επιλογή που παρέχεται στον χρήστη μέσα από την εφαρμογή είναι αυτή της γρήγορης αναζήτησης και εμφάνισης καταχωρημένων πληροφοριών. Ο χρήστης όπως φαίνεται στην εικόνα 45 έχει την δυνατότητα να επιλέξει μεταξύ των δύο radiobutton, ΑΦΜ ή του ΑΡ.ΚΥΚΛΟΦΟΡΙΑΣ που θα αποτελέσει το είδος του κλειδιού αναζήτησης που θα εισάγει ο χρήστης στο πεδίο συμπλήρωσης κειμένου με βάση τι πληροφορίες θέλει να λάβει αφού κάνει κλικ στο κουμπί Search. Άρα με την δραστηριότητα search γίνεται αναζήτηση στοιχείων είτε με βάσει το ΑΦΜ, είτε με τον αρ.κυκλοφορίας που αποτελούν τα πρωτεύοντα κλειδιά πινάκων Customer και Cars αντίστοιχα, απ' τους οποίους θα λαμβάνονται τα ανάλογα στοιχεία. Με κλικ στο κουμπί cancel μεταβαίνουμε στην προηγούμενη οθόνη. Πιο συγκεκριμένα με το πάτημα του κουμπιού Search, ο χρήστης αφού έχει επιλέξει το είδος της αναζήτησης του λαμβάνει τα ανάλογα αποτελέσματα

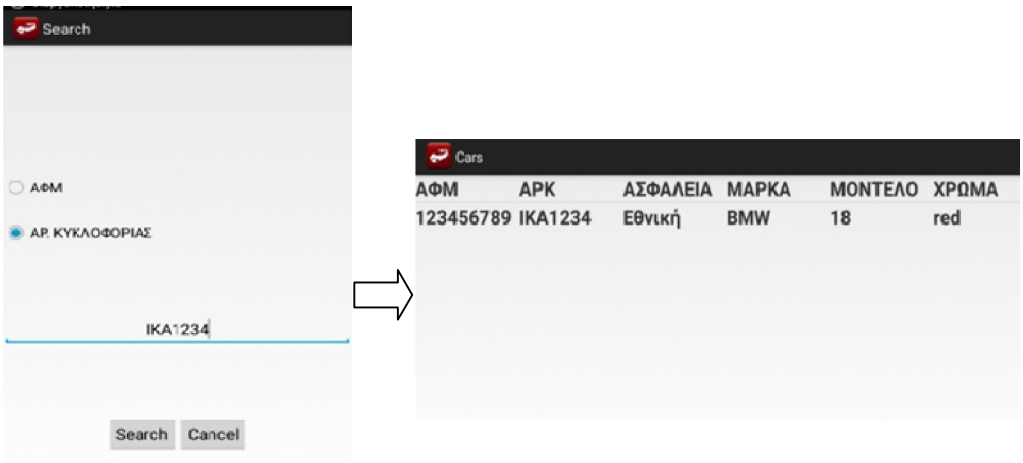
αυτής μέσα από την οθόνη απεικόνισης του πίνακα Customer (εικόνα 47) ή του Car (εικόνα 48).

Control Type	Description	Related Classes
Button	A push-button that can be pressed, or clicked, by the user to perform an action.	Button
Text field	An editable text field. You can use the AutoCompleteTextViewwidget to create a text entry widget that provides auto-complete suggestions	EditText, AutoCompleteTextView
Checkbox	An on/off switch that can be toggled by the user. You should use checkboxes when presenting users with a group of selectable options that are not mutually exclusive.	CheckBox
Radio button	Similar to checkboxes, except that only one option can be selected in the group.	RadioGroup RadioButton
Toggle button	An on/off button with a light indicator.	ToggleButton
Spinner	A drop-down list that allows users to select one value from a set.	Spinner
Pickers	A dialog for users to select a single value for a set by using up/down buttons or via a swipe gesture. Use a DatePickercode> widget to enter the values for the date (month, day, year) or a TimePicker widget to enter the values for a time (hour, minute, AM/PM), which will be formatted automatically for the user's locale.	DatePicker, TimePicker



Εικόνα 46: Input controls

Εικόνα 47: Παράδειγμα αναζήτησης και εμφάνισης στοιχείων ενός πελάτη



Εικόνα 48: Παράδειγμα αναζήτησης και εμφάνισης στοιχείων ενός αυτοκινήτου

5.3.9 Οθόνη Statistics

Εικόνα 49:Οθόνη εμφάνισης στατιστικών

Στην οθόνη Statistics απεικονίζονται στατιστικά στοιχεία, χρήσιμα για να παρακολουθείται η πορεία της επιχείρησης, συγκεκριμένα το πλήθος το service ανά μήνα και έτος, όπως φαίνεται στην εικόνα 49. Η διαμόρφωση της οθόνης πραγματοποιείται μέσω της σύνδεσης των αρχείων Viewcontents_layout.xml όπου εισάγεται μια λίστα εμφάνισης πληροφοριών και του stats_list_adapter_view.xml που περιλαμβάνει τρία πεδία εμφάνισης κειμένου (έτος, μήνας, πλήθος).

6. ΚΕΦΑΛΑΙΟ: ΠΕΡΙΓΡΑΦΗ ΣΥΝΑΡΤΗΣΕΩΝ ΚΑΙ ΛΕΙΤΟΥΡΓΙΩΝ ΤΗΣ ΕΦΑΡΜΟΓΗΣ.

Στο κεφάλαιο αυτό αναλύονται οι λειτουργίες της εφαρμογής με την βοήθεια του διαγράμματος ροής και των περιγραφή σημαντικών κομματιών κώδικα.

6.1 Λειτουργικότητα

Σύμφωνα με τις απαιτήσεις που προέκυψαν από την μελέτη των αναγκών της θέσης της γραμματέως στο φανοποιείο, ο χρήστης θα έχει στην διάθεσή του μια δέσμη λειτουργιών που θα διευκολύνει την καταγραφή και διαχείριση των στοιχείων της επιχείρησης . Οι σημαντικότερες από αυτές τις λειτουργίες περιλαμβάνουν:

- ✓ Καταχώρηση και εμφάνιση στοιχείων πελάτη.
- ✓ Εισαγωγή ,προβολή και διαχείριση στοιχείων αυτοκινήτων και service αυτών.
- ✓ Αναζήτηση και εμφάνιση κατοχυρωμένων στοιχείων.
- ✓ Προβολή Στατιστικών.

6.2 Συστατικά στοιχεία εφαρμογής

Με τον όρο αυτό εννοούμε τα απαραίτητα δομικά στοιχεία μιας εφαρμογής Android [21]. Το κάθε στοιχείο στην ουσία είναι ένας τρόπος πρόσβασης του λειτουργικού συστήματος στην εφαρμογή μας.

Τα βασικά συστατικά στοιχεία εφαρμογής είναι τα εξής:

- **Δραστηριότητα** είναι μια οθόνη διεπαφής χρήστη (GUI) και προβολής πληροφοριών. Κάθε εφαρμογή έχει τόσες Activities όσες και οι διαφορετικές οθόνες οι οποίες εμφανίζονται στον χρήστη. Όλες οι δραστηριότητες συνεργάζονται μεταξύ τους για να δώσουν στον χρήστη μια συνολική εμπειρία χρήσης της εφαρμογής.
- **Προθέσεις** (Intents) που έχουν τον ρόλο του μεσολαβητή στην επικοινωνία και εναλλαγή της λειτουργίας των δραστηριοτήτων. Εξασφαλίζουν την μετάβαση από την μία δραστηριότητα σε μια άλλη και επίσης χρησιμοποιούνται για ανταλλαγή δεδομένων. Η ανταλλαγή δεδομένων, μπορεί να γίνει είτε μεταξύ των Activities μιας εφαρμογής, είτε από τη μία εφαρμογή στην άλλη.

- **Υπηρεσίες** (Services) είναι οι λειτουργίες της εφαρμογής οι οποίες είναι σχεδιασμένες να τρέχουν στο παρασκήνιο και να επιστρέφουν αποτελέσματά ακόμη και όταν η εφαρμογή δεν είναι στο προσκήνιο.
- **Πάροχος Περιεχόμενου (**Content Providers).Οι content providers μιας εφαρμογής διαχειρίζονται συγκεκριμένα δεδομένα της εφαρμογής τα οποία έχει ορίσει ο προγραμματιστής κατά την κατασκευή του. Συνηθισμένα δεδομένα τα οποία μοιράζονται μέσω Content Providers, είναι οι βάσεις δεδομένων SQLite μιας εφαρμογής, και οι επαφές του χρήστη.
- **Δέκτες Μετάδοσης** (Broadcast Receivers). Πρόκειται για ένα είδους υπηρεσία η οποία αντιλαμβάνεται κάποια γεγονότα του συστήματος και αναλαμβάνει να ενημερώσει το σύστημα η τις υπόλοιπες εφαρμογές. Ο σκοπός τους είναι διπλός καθότι μπορούν και να ενημερωθούν για κάποιο συμβάν από άλλες εφαρμογές, αλλά και να ειδοποιήσουν τις υπόλοιπες εφαρμογές και το σύστημα για κάποιο συμβάν που τις ενεργοποίησε. Δεν έχουν γραφικό περιβάλλον αλλά μπορούν να προβάλουν ειδοποίηση στον χρήστη μέσω της μπάρας ειδοποιήσεων. Συνήθως χρησιμοποιούνται ως διαμεσολαβητές μεταξύ των Activities και των Services μιας εφαρμογής.

6.3 SQLite

Η εφαρμογή Fix_car1 παρέχει εύκολη πρόσβαση σε πληροφορίες πελατών και των αυτοκινήτων τους, που αποθηκεύονται μέσα σε μια βάση δεδομένων SQLite, που βρίσκεται μέσα στη συσκευή. Οι πληροφορίες που εισάγονται, αποθηκεύονται μέσα σε μια βάση δεδομένων SQLite. Η SQLite είναι μια απ' τις ευρύτερα διαδεδομένες μηχανές βάσεων δεδομένων στον κόσμο [22], συμβατή με SQL που δεν χρειάζεται ξεχωριστό διακομιστή για εκτέλεση [23]. Επίσης δεν απαιτεί ξεχωριστή εγκατάσταση και είναι δωρεάν.

6.4 Σενάριο χρήσης

Για να γίνει όμως πιο κατανοητή η φιλοσοφία της εφαρμογής παρακάτω γίνεται λεπτομερής ανάλυση των περιπτώσεων της εφαρμογής με την σχετική περιγραφή κώδικα ενώ παρατίθεται διάγραμμα ροής των επιλογών της εφαρμογής. Στις παρακάτω παραγράφους γίνεται αναλυτική περιγραφή των διαδικασιών που περιλαμβάνει το διάγραμμα ροής.

Οι υποενότητες που ακολουθούν σχετικά με τα στοιχεία που έπειτα περιγράφονται είναι:

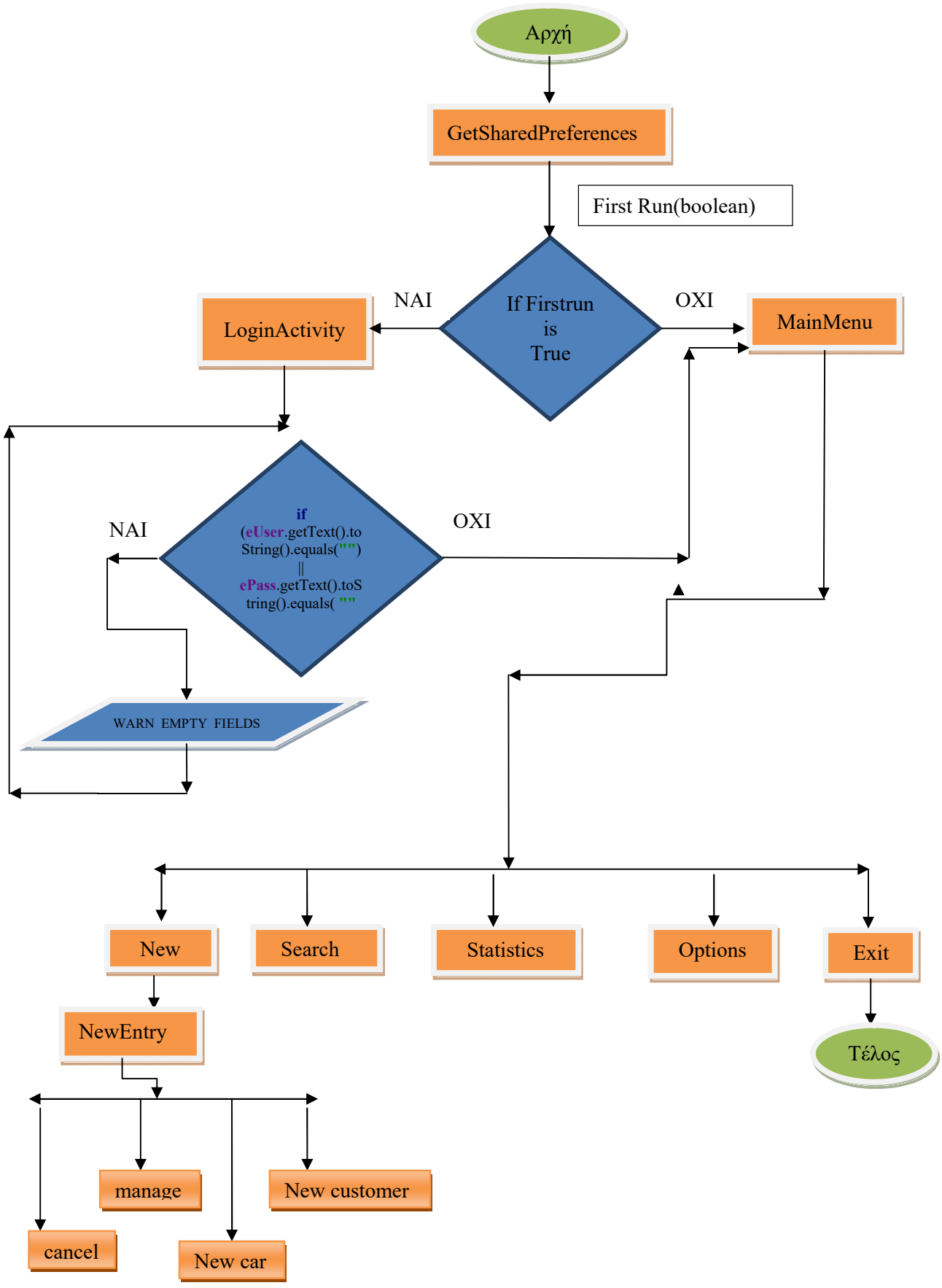
Main Activity-Είσοδος στην εφαρμογή,
Login Activity-Ορισμός στοιχείων εισόδου του χρήστη.
Main Menu -Βασικό μενού επιλογών της εφαρμογής.

Οι υποενότητες της Main Menu είναι :

- New – Μεταβαίνει στην New Entry που περιέχει τις επιλογές ως προς το είδος των στοιχείων για εισαγωγή.
- Search Menu - Πραγματοποιείται γρήγορη αναζήτηση και εμφάνιση στοιχείων ενός συγκεκριμένου πελάτη ή αυτοκινήτου.
- Statistics - Εμφάνιση στατιστικών στοιχείων.
- Option Menu - Ρυθμίσεις του χρήστη.
- Exit - Έξοδος από την εφαρμογή.

Υποενότητες της New Entry είναι:

- New Entry Customer - Εισαγωγή στοιχείων των πελατών.
- New Entry Car - Εισαγωγή στοιχείων του αυτοκινήτου.
- New Entry Service - Εισαγωγή στοιχείων των επισκευών.
- Cancel - Επιστροφή στο βασικό μενού.



"ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ ΕΠΙΛΟΓΩΝ ΤΗΣ ΕΦΑΡΜΟΓΗΣ "

6.4.1MainActivity

Με την εκκίνηση της εφαρμογής ανοίγει αυτόματα η κεντρική οθόνη (εικόνα(34)), οι λειτουργίες της οποίας ορίζονται μέσα στην MainActivity.java κλάση. Αρχικά διατυπώνεται ο ορισμός του γραφικού περιβάλλοντος αυτής, αυτό γίνεται σε κάθε κλάση, πχ δήλωση του button (Button btEnter = findViewById(R.id.***button_enter***);).
Σημαντικό σημείο του κώδικα αποτελεί η παρακάτω γραμμή **final boolean** firstRun = getSharedPreferences(LoginActivity.***MY_PREF_NAME***,***MODE_PRIVATE***).getBoolean(**"firstRun"**, **true**); πριν τον έλεγχο. Η firstrun είναι η έξοδος της getSharedPreferences που επιστρέφει μια μεταβλητή τύπου Boolean η οποία είναι αληθής μόνο για την πρώτη εκτέλεση του προγράμματος. Αν η firstrun είναι αληθής πραγματοποιείται μετάβαση στην LoginActivity αλλιώς γίνεται μετάβαση στο MainMenuActivity.
Εφόσον ο χρήστης έχει θέσει τις ρυθμίσεις του , την επόμενη φορά που θα εκκινήσει την εφαρμογή το σύστημα διαβάζοντας την τιμή false στην MainActivity, θα μεταβεί απευθείας στο βασικό μενού.

6.4.2LoginActivity

Όταν ο χρήστης επιλέξει να αποκτήσει πρόσβαση στην εφαρμογή, από την οθόνη του κινητού μπορεί να εισάγει τα στοιχεία εισόδου του χρήστη (όνομα και κωδικό) όπως φαίνεται στην εικόνα(35). Τότε με το πάτημα του κουμπιού login καλείται η συνάρτηση LoginActivity η οποία παίρνει σαν είσοδο το κείμενο από τα δύο πεδία κειμένου, κάνει τον έλεγχο αν τα πεδία είναι συμπληρωμένα και δίνει την έξοδο της (username&password) στην getSharedPreferences, η οποία επιστρέφει αληθές ή ψευδές αν είναι ορθός ή όχι ο έλεγχος ταυτότητας.

Πιο αναλυτικά αρχικά γίνεται δήλωση μεταβλητής στιγμιότυπου (prefs), η οποία θα χρησιμοποιηθεί για να χειρίζεται τα ζεύγη-κλειδιά των κοινών προτιμήσεων που παριστούν τις αποθηκευμένες ρυθμίσεις του χρήστη. Στην συνέχεια γίνεται ανάθεση τιμών των κλειδιών .Έπειτα γίνεται έλεγχος αν τα πεδία είναι μη συμπληρωμένα η εφαρμογή επιστρέφει το μήνυμα "WARN_EMPTY_FIELDS" τύπου Toast, που εμφανίζεται μέσω της μεθόδου show() για ένα χρονικό διάστημα και ρόλος του είναι η ειδοποίηση του χρήστη ότι τα πεδία είναι κενά. Αντιθέτως αν τα πεδία κειμένου δεν είναι κενά δημιουργείται ένας νέος συντάκτης-επεξεργαστής για τις κοινές προτιμήσεις, μέσω του οποίου δίνεται η δυνατότητα ορισμού των δεδομένων των προτιμήσεων (ορισμός του ονόματος και του κωδικού μέσω του Editor) και μεταβίβασης αυτών στο αντικείμενο SharedPreferences. Στον επεξεργαστή(editor) προτιμήσεων ορίζεται μια τιμή false στο κλειδί FirstRun .

6.4.3MainMenuActivity

Στην MainMenuActivity.java κλάση ορίζεται το κύριο μενού των λειτουργιών της εφαρμογής, όπως φαίνεται από την εικόνα(36). Πιο αναλυτικά ο χρήστης έχει την δυνατότητα να επιλέξει μεταξύ των ακόλουθων πέντε λειτουργιών:
1) Εισαγωγή νέων στοιχείων
2) Αναζήτηση και εύρεση αποθηκευμένων πληροφοριών
3) Εμφάνιση στατιστικών στοιχείων
4) Ορισμός ρυθμίσεων του χρήστη
5) Έξοδος από την εφαρμογή
Όταν ο χρήστης πατήσει το κουμπί με την εκάστοτε λειτουργία που αποφάσισε να πραγματοποιήσει τίθεται μια **Πρόθεση** που ορίζει σε ποια κλάση μεταβαίνει για να την εκτέλεση της ενέργειας αυτής. Με την επιλογή EXIT μέσω της μεθόδου Finish() επιτυγχάνεται έξοδος από την εφαρμογή.

Συγκεκριμένα πατώντας το κουμπί New ο χρήστης μεταβαίνει στο μενού επιλογής του είδους των πληροφοριών που θέλει να εισάγει (NewEntry.class). Δεύτερη επιλογή είναι το search , όπου κάνοντας κλικ πάνω σ' αυτό το κουμπί μεταβαίνουμε στην αναζήτηση

καταχωρημένων πληροφοριών (SearchMenu.class). Τρίτη επιλογή είναι το κουμπί Statistics όπου πατώντας το εμφανίζεται απευθείας μια οθόνη με στατιστικά στοιχεία (ListStatsContents.class). Τέταρτο στην ιεραρχία του μενού είναι η επιλογή Options, όπου επιλέγοντας την ο χρήστης μεταβαίνει στις ρυθμίσεις του χρήστη(OptionMenu.class). Τέλος με το κουμπί EXIT πραγματοποιείται έξοδος από την εφαρμογή .

6.4.4OptionMenu

Όταν ο χρήστης επιλέξει τις ρυθμίσεις από το βασικό μενού της εφαρμογής, μπορεί να αλλάξει τις υπάρχουσες ρυθμίσεις διεπαφή χρήστη και στην συνέχεια να τις αποθηκεύσει, όπως φαίνεται στην εικόνα(37).

Πιο αναλυτικά αρχικά γίνεται δήλωση μεταβλητής στιγμιότυπου (prefs) η οποία θα χρησιμοποιηθεί για να χειρίζεται τα ζεύγη-κλειδιά των κοινών προτιμήσεων που παριστούν τις αποθηκευμένες ρυθμίσεις του χρήστη. Στην συνέχεια γίνεται ανάθεση τιμών των κλειδιών .Έπειτα γίνεται έλεγχος αν τα πεδία είναι μη συμπληρωμένα τότε η εφαρμογή επιστρέφει το μήνυμα “WARN_EMPTY_FIELDS” . Αντιθέτως αν τα πεδία κειμένου δεν είναι κενά δημιουργείται ένας συντάκτης-επεξεργαστής για τις προτιμήσεις, μέσω του οποίου δίνεται η δυνατότητα πραγματοποίησης αλλαγών στα δεδομένα των προτιμήσεων και μεταβίβασης αυτών στο αντικείμενο SharedPreferences. Αφού πραγματοποιηθούν οι αλλαγές, αποθηκεύονται μόλις ο χρήστης πατήσει το κουμπί αποθήκευσης και πραγματοποιείται μετάβαση στην Login, της οποίας τα πεδία είναι αυτόματα συμπληρωμένα σύμφωνα με τις νέες ρυθμίσεις .

6.4.5NewEntry

Η NewEntry.java κλάση περιλαμβάνει τις επιλογές του χρήστη όσον αφορά το είδος των στοιχείων που θέλει να εισάγει, όπως φαίνεται από την εικόνα(38). Οι επιλογές είναι:

1. Εισαγωγή στοιχείων νέου πελάτη (NewEntryCustomer.class).
2. Εισαγωγή στοιχείων νέου αυτοκινήτου (NewEntryCar.class).
3. Εισαγωγή πληροφοριών για τα service (NewEntryService.class).
4. Επιστροφή στην οθόνη του βασικού μενού (cancel).

Επιλέγοντας λοιπόν ο χρήστης τι θέλει να κάνει, μεταβαίνει αυτόματα στην οθόνη της διαδικασίας που επέλεξε να πραγματοποιήσει με την χρήση των προθέσεων όπως στην MainMenuActivity.

6.4.6DatabaseHelper

Πολλά κομμάτια μέσα σ’ αυτή την εφαρμογή αλληλεπιδρούν με μια βάση δεδομένων SQLite μέσω της βοηθητικής κλάσης DatabaseHelper. Αυτή η κλάση χρησιμοποιεί μια ένθετη υποκλάση της SQLiteOpenHelper (πακέτο android.database.sqlite), η οποία απλοποιεί την δημιουργία βάσης δεδομένων και επιτρέπει την απόκτηση αντικειμένου τύπου SQLiteDatabase για την διαχείριση των περιεχόμενων μιας βάσης δεδομένων.
Τα ερωτήματα στην βάση δεδομένων γίνονται στην γλώσσα SQL(Structured Query Language) και ο χειρισμός των αποτελεσμάτων των ερωτημάτων γίνεται μέσω μιας μεταβλητής τύπου Cursor[22].

Η σύνδεση με την βάση δεδομένων SQLite στο Android επιτυγχάνεται με την δημιουργία ενός αρχείου Java (DatabaseHelper.java) που μετατρέπεται σε ένα αρχείο βάσης δεδομένων επεκτείνοντας το από την κλάση SQLiteOpenHelper (public class DatabaseHelper extends SQLiteOpenHelper).

Μετά την επέκταση αυτής της τάξης, εκτελούνται απαραιτήτως οι παρακάτω ενέργειες [23]:

- Πρώτα ορίζεται το όνομα της βάσης (fix_cars.db). Έπειτα η **getInstance(Context context**) μέθοδος που ελέγχει εάν το μέλος sInstance είναι μηδενικό και αν είναι null τότε θα δημιουργήσει ένα νέο. Με κάθε τρόπο, μια εμφάνιση της κλάσης DatabaseHelper επιστρέφεται πάντα κάθε φορά που καλείται αυτή η μέθοδος. Αυτό το μοτίβο διασφαλίζει ότι υπάρχει μόνο μία παρουσία της κλάσης DatabaseHelper.

```
public static final String DATABASE_NAME = "fix_cars.db";
private static DatabaseHelper sInstance;
public static synchronized DatabaseHelper getInstance(Context context)
{ if (sInstance == null)
        // λαμβάνει μια παρουσία της δημιουργηθείσας ΒΔ με την
        getApplicationContext())
  {  sInstance = new DatabaseHelper(context.getApplicationContext());  }
     return sInstance;
```

- **Κατασκευαστής:** Δεδομένου ότι το αρχείο κλάσης DatabaseHelper.java εκτείνεται από το SQLiteOpenHelper, εκτελώντας τον κατασκευαστή της μητρικής κλάσης. Ο κατασκευαστής δέχεται ένα αντικείμενο Context, το όνομα της βάσης δεδομένων, ένα εργοστάσιο δρομολόγησης και την έκδοση της βάσης δεδομένων. Η έκδοση μπορεί να είναι ένας ακέραιος αριθμός, ο οποίος συνήθως ξεκινά από 1.

```
private DatabaseHelper(Context context) {
        super(context, DATABASE_NAME, null, 1);  }
```

Τα δύο βασικά στοιχεία της μεθόδου είναι το όνομα της ΒΔ και η έκδοσή της (παράμετροι name και version).

- **onCreate () -** αυτή η μέθοδος καλείται όταν δημιουργείται η βάση δεδομένων για πρώτη φορά. Η παράμετρος για αυτήν τη μέθοδο είναι η πραγματική βάση δεδομένων που δημιουργείται. Αυτή η βάση δεδομένων έχει μια μέθοδο που ονομάζεται execSQL () που χρησιμοποιείται για την δημιουργία των πινάκων της βάσης δεδομένων. Η μέθοδος execSQL χρησιμοποιείται για την εκτέλεση οποιοδήποτε ερωτήματος σε Sqlite ΒΔ.

```
public void onCreate(SQLiteDatabase db) {
  db.execSQL(Car.getCreateTable());
  db.execSQL(Customer.getCreateTable());
  db.execSQL(Service.getCreateTable());  }
```

Αν η ΒΔ δεν υπάρχει, τότε με την γραμμή database=this.getWritableDatabase(); καλείται η μέθοδος onCreate() η οποία και θα αναλάβει την δημιουργία των πινάκων.

```
// ----- Services -----
  public boolean save(Service service) {
  SQLiteDatabase db = this.getWritableDatabase();
  return service.save(db);  }
//ανοίγει την σύνδεση με την βάση δεδομένων
  public List<Service> getServices(String ak) throws IllegalStateException {
  SQLiteDatabase db = this.getReadableDatabase();
  return Service.list(db, ak); }
```

- **onUpgrade () -** αυτή η μέθοδος καλείται όταν η βάση δεδομένων χρειάζεται αναβάθμιση. Η βάση δεδομένων θα επισημανθεί για αναβάθμιση κατά την αύξηση του αριθμού έκδοσης της βάσης δεδομένων. Μέσα σε αυτήν τη μέθοδο διατυπώνονται οι μέθοδοι που θα εφαρμόσουν τις αλλαγές που επιθυμεί να κάνει ο χρήστης στο σχήμα βάσης δεδομένων του.

```
  public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  }
```

6.4.7 Car, Customer , Service

Για την διαχείριση των τριών πινάκων της εφαρμογής χρησιμοποιείται μια κατηγορία για κάθε πίνακα που αποτελεί μια συλλογή από πίνακες και στήλες. Επιπροσθέτως σε κάθε κατηγορία δημιουργούνται τα ερωτήματα που θέτονται στην βάση δεδομένων.
Παρακάτω παρουσιάζονται τα περιεχόμενα της Car:

- Δήλωση τοπικών μεταβλητών που αντιστοιχούν στα στοιχεία του αυτοκινήτου που πρέπει να εισάγει ο χρήστης .
- Παραμετροποιημένο κατασκευαστή που θα χρησιμοποιηθεί για να προστεθούν τιμές στον πίνακα. Car (String afm, String ark, String insurance, String brand, String model, String color) έχει παραμέτρους τις αλφαριθμητικές μεταβλητές των στοιχείων του αυτοκινήτου και ορίζει δείκτη για κάθε μεταβλητή π.χ. afm = afm;
- Κενό κατασκευαστή **private** Car() { }
- Μεθόδους δημιουργίας και επιστροφής (set()&get()) για τις μεταβλητές όλων των στιγμιότυπων δηλαδή για κάθε στοιχείο του αυτοκινήτου.
- Δήλωση του ονόματος του πίνακα που θα είναι τύπου στατικό αλφαριθμητικό.
  public static String *TABLE_NAME* = **"car"**;
- Δήλωση δημόσιων στατικών αλφαριθμητικών μεταβλητών που αντιστοιχούν στα ονόματα κάθε στήλης του πίνακα.
  public static String *COL_ARK* = **"ARK"**;
- Μέθοδος getCreateTable() Με την οποία καθορίζεται ο πίνακας δεδομένων για την καταχώρηση των εγγραφών των αυτοκινήτων.

```
public static String getCreateTable() { String tmp = "CREATE TABLE " +
TABLE_NAME;
        tmp += " (ID INTEGER PRIMARY KEY AUTOINCREMENT, ";
        final String[] COLUMNS = {COL_AFM, COL_ARK, COL_INSURANCE,
        COL_BRAND, COL_MODEL, COL_COLOR};
```

  Με την χρήση της παραπάνω δήλωσης SQL θα δημιουργήσει έναν πίνακα βάσης δεδομένων, των οποίων οι στήλες έχουν σχεδόν ένα προς ένα χαρτογράφηση στην κλάση Java που αντιπροσωπεύει.
- Η μέθοδος parse(Cursor cursor) δημιουργεί ένα αντικείμενο που ανήκει στην κατηγορία του αυτοκινήτου και αφορά κάθε νέο αυτοκίνητο, διαβάζει κάθε τρέχουσα γραμμή του πίνακα και χρησιμοποιώντας την μέθοδος getColumnIndex παίρνει τους δείκτες στηλών του πίνακα car της ΒΔ, και επιστρέφει το αντικείμενο car με όλα τα δεδομένα για την βάση δεδομένων .
- Μέθοδος save(SQLiteDatabase db) αποθήκευσης αντικειμένου της βάσης
  Ορίζει μια μεταβλητή με όνομα content values της κλάσης ContentValues που χρησιμοποιείται ώστε οριστούν οι τιμές για κάθε κολώνα μιας εγγραφής που προορίζονται για αποθήκευση. Επιπλέον χρησιμοποιείται μια μεταβλητή με όνομα result που με την βοήθεια της μεθόδου replace σώζει την εγγραφή, όπως έχει περιγραφεί στο contentValues ενώ αν υπάρχει ήδη εγγραφή στην βάση με ίδιο πρωτεύον κλειδί, την αντικαθιστά.Η επιστρεφόμενη τιμή από αυτή τη μέθοδο είναι η ταυτότητα της γραμμής που μόλις εισήχθη στον πίνακα, αν η τιμή είναι -1 η εισαγωγή απέτυχε αλλιώς αποθηκεύει ένα αντικείμενο αυτοκινήτου στη βάση δεδομένων.

```
long result = db.replace(TABLE_NAME, null, contentValues);
if (result == -1) {return false; }
else { return true; }
    }
```

- Για την αποθήκευση αντικειμένων της κατηγορίας Car χρησιμοποιείται ArrayList.Το car χρησιμοποίει ένα ArrayList που περιλαμβάνει τα αντικείμενα της βάσης δεδομένων. Για την επιστροφή των χαρακτηριστικών του αντικειμένου car χρησιμοποιείται μια μεταβλητή τύπου cursor. Ενώ για την αποθήκευση των αυτοκινήτων στην βάση χρησιμοποιείται ένα μοναδικό κλειδί (ο αριθμός κυκλοφορίας), με το οποίο γίνεται και η αναζήτηση και λήψη της εγγραφής του αυτοκινήτου από τη βάση δεδομένων, που έχει ως μοναδικό κλειδί αυτό που έθεσε ο χρήστης.

```
public static List<Car> list(SQLiteDatabase db, String ak) {
      final List<Car> toReturn = new ArrayList<>();
      if (ak == null || ak.length() == 0) {
         ak = null;
      } else {  ak = ak. ();
      }
      try (Cursor cursor = db.rawQuery("SELECT * FROM " + TABLE_NAME
+ (ak == null ? "" : " WHERE " + COL_ARK + " like ?"), (ak == null ? null :
new String[]{ak + "%"}))) {

         while (cursor.moveToNext()) {
            Car car = Car.parse(cursor);
            toReturn.add(car);
         }
      }  return toReturn;
```

Έπειτα από την πρόσθεση όλων των εντολών SQL για την δημιουργία των πινάκων της βάσης δεδομένων, πρέπει να ενημερώνεται η μέθοδος onCreate () της κλάσης DatabaseHelper.

Η κατηγορία πελάτη (customer.java) έχει όμοια περιεχόμενα με την Car με την διαφορά όμως ότι αφορά τον πίνακα πελατών, επιστρέφει δηλαδή χαρακτηριστικά του αντικειμένου πελάτη.

Η κατηγορία επισκευής (service.java)εισάγει και επιστρέφει χαρακτηριστικά του αντικειμένου επισκευής. Σ' αυτή την κατηγορία τα περιεχόμενα έχουν μερικές διαφοροποιήσεις σε σχέση με τις δύο προηγούμενες κατηγορίες, οι οποίες παρατίθενται παρακάτω :

- ✓ Η μέθοδος **public static** Service parse(Cursor cursor) χρησιμοποιεί την μέθοδο setDate για θέσει την ημερομηνία εισαγωγής του αυτοκινήτου, η οποία επιστρέφεται αυτόματα ενημερωμένη με την χρήση της υποκλάσης SimpleDateFormat(***DATE_FORMAT***).

- ✓ Η μέθοδος save(SQLiteDatabase db) χρησιμοποιεί την κλάση IllegalStateException για να κατασκευάσει νέες εξαιρέσεις με την προκαθορισμένη αιτία και ένα μήνυμα λεπτομέρειας τους, για την αποφυγή εκτέλεσης αιτούμενης λειτουργίας όταν αυτή δεν είναι σε κατάλληλη κατάσταση.

Για την αποφυγή σφαλμάτων χρησιμοποιείται η IllegalStateException()

```
public boolean save(SQLiteDatabase db) throws IllegalStateException {
      // Validate Date
      java.util.Date tmpDate;
      try {
         tmpDate = new SimpleDateFormat(DATE_FORMAT).parse(getDate());
         if (tmpDate == null) {
            throw new IllegalStateException("Η ημερομηνία δεν έχει σωστή μορφή.
Πρέπει να την ορίσετε οπως 30/12/2018");
         }
      } catch (Exception e) {
         throw new IllegalStateException("Η ημερομηνία δεν έχει σωστή μορφή.
Πρέπει να την ορίσετε οπως 30/12/2018");
      }
      if (Customer.list(db, getAfm()).size() < 1) {
         throw new IllegalStateException("Δεν υπάρχει πελάτης με ΑΦΜ " +
```

```
getAfm());
        }
        if (Car.list(db, getArk()).size() < 1) {
            throw new IllegalStateException("Δεν υπάρχει όχημα με Αριθμό
Κυκλοφορίας " + getArk());
        }
```

- ✓ Εκτός από το ερώτημα στην βάση για επιστροφή των χαρακτηριστικών του αντικειμένου Service, τίθεται ένα επιπλέον ερώτημα για την εμφάνιση του πλήθους των επισκευών ανά χρονολογία .

```
    public static List<String[]> stats(SQLiteDatabase db) {
        final List<String[]> toReturn = new ArrayList<>();

        final String SQL = "SELECT " + COL_DATE_YEAR + ", " +
COL_DATE_MONTH + ", count(*) as count FROM " + TABLE_NAME + " GROUP
BY " + COL_DATE_YEAR + ", " + COL_DATE_MONTH;
        toReturn.add(new String[]{
                "YEAR", "MONTH", "COUNT"
        });
        try (Cursor cursor = db.rawQuery(SQL, new String[0])) {
            while (cursor.moveToNext()) {
                int year = cursor.getInt(cursor.getColumnIndex(COL_DATE_YEAR)) + 1900;
                int month = cursor.getInt(cursor.getColumnIndex(COL_DATE_MONTH)) +
1;
                int count = cursor.getInt(cursor.getColumnIndex("count"));
                toReturn.add(new String[]{String.valueOf(year), String.valueOf(month),
String.valueOf(count)});
            }
        }
        return toReturn;
    }
}
```

6.4.8 CarListAdapter, CustomerListAdapter, ServiceListAdapter, StatsListAdapter

Οι κλάσεις CarListAdapter, CustomerListAdapter και ServiceListAdapter ορίζουν πώς θα προβάλλονται τα στοιχεία του πίνακα αυτοκινήτων, πελατών και επισκευών αντίστοιχα. Η StatsListAdapter είναι υπεύθυνη για την παραμετροποίηση της εμφάνισης των στατιστικών στοιχείων.

Για τον ορισμό μιας προσαρμοσμένης προβολής για κάθε στοιχείο του συνόλου δεδομένων των αυτοκινήτων στην εφαρμογή, γίνεται εφαρμογή ενός ListAdapter [28]. Η κλάση CarListAdapter επεκτείνει την ArrayAdapter και ο ρόλος της είναι η δημιουργία αλλά και η ρύθμιση των παραμέτρων της προβολής για κάθε στοιχείο δεδομένων μέσω της μεθόδου getView().

Η δημιουργία του CarListAdapter πραγματοποιείται με την εκτέλεση των παρακάτω ενεργειών:

- Δημιουργία μιας κλάσης επεκτείνοντας το ArrayAdapter
- Πρόσθεση ενός κατασκευαστή στην τάξη
- Αντικατάσταση της μεθόδου getView

Η CarListAdapter επιστρέφει μια προβολή για κάθε αντικείμενο σε μια συλλογή αντικειμένων δεδομένων που παρέχετε και μπορεί να χρησιμοποιηθεί με widgets για τη διεπαφή χρήστη που βασίζονται σε λίστες. Επίσης υλοποιείται μια μέθοδος getCount() που επιστρέφει πόσα στοιχεία θα εμφανιστούν. Εμφανίζονται όσα είναι τα αυτοκίνητα κι ένα επιπλέον για τις

επικεφαλίδες. Μετά υλοποιείται μια μέθοδος getView που επιστρέφει το τι θα εμφανιστεί για κάθε γραμμή της λίστας. Στην πρώτη γραμμή πρέπει να εμφανίζονται οι τίτλοι και στις υπόλοιπες να εμφανιστούν τα αυτοκίνητα. Χρησιμοποιείται ο mInflater για την δημιουργία του Object convertView που βασίζεται στην δομή που έχει περιγραφεί στο layout xml με το id που έχει οριστεί το mViewResourceId.

Ομοίως δημιουργούνται οι κλάσεις CustomerListAdapter, ServiceListAdapter και StatsListAdapter.

Ακολουθεί ο κώδικας ρύθμισης των παραμέτρων της προβολής των στατιστικών δηλαδή η κλάση StatsListAdapter, ως παράδειγμα για καλύτερη κατανόηση της παραμετροποίησης.

```
public class StatsListAdapter extends ArrayAdapter<String[]> {
   private LayoutInflater mInflater;
   private List<String[]> data;
   private int mViewResourceId;
   StatsListAdapter(Context context, List<String[]> data) {
      super(context, R.layout.stats_list_adapter_view, data);
      this.data = data;
      mInflater = (LayoutInflater)
context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
      mViewResourceId = R.layout.stats_list_adapter_view; }
   @SuppressLint("ViewHolder")
   @NonNull
   public View getView(int position, View convertView, @NonNull ViewGroup parent) {
      convertView = mInflater.inflate(mViewResourceId, null);
      TextView tYear = convertView.findViewById(R.id.text_year);
      TextView tMonth = convertView.findViewById(R.id.text_month);
      TextView tCount = convertView.findViewById(R.id.text_count);
      String[] rowData = data.get(position);
      tYear.setText(rowData[0]);
      tMonth.setText(rowData[1]);
      tCount.setText(rowData[2]);
      return convertView; }
}
```

6.4.9 ListCarContents, ListCustomerContents, ListServiceContents, ListStatsContents

Οι κλάσεις ListCarContents, ListCustomerContents, ListServiceContents και ListStatsContents καλούνται για παραμετροποίηση των ήδη αποθηκευμένων αντικειμένων των πινάκων της βάσης δεδομένων.
Οι κλάσεις αυτές περιέχουν μια μέθοδο OnCreate με παράμετρο την savedInstanceState που καλείται απ' το σύστημα όταν αλλάζει η παραμετροποίηση της εφαρμογής (πχ περιστροφή οθόνης) και χρησιμοποιείται για να αποθηκεύει πληροφορίες που θέλει να επαναφέρει ο χρήστης.

```
      protected void onCreate(@Nullable Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
```

Μέσα στην OnCreate καλείται η μέθοδος setContentView που θέτει τα περιεχόμενα εμφάνισης. Π.Χ.

```
      setContentView(R.layout.viewcontents_layout);
```

```
// Η αναζήτηση περνάει μια "παράμετρο" με κλειδί "q". Το q είναι μια προσωρινή
μεταβλητή που κρατάει τι έχει πληκτρολογήσει ο χρήστης και με βάσει αυτήν κάνει
αναζήτηση.
 final String ak = getIntent().getStringExtra("q");
// συμπλήρωση πίνακα από τη βάση δεδομένων
 final List<Car> carList = DatabaseHelper.getInstance(this).getCars(ak);
```

Επίσης γίνεται έλεγχος αν το μήκος της λίστας είναι 0 εμφανίζει ένα μήνυμα που ειδοποιεί τον χρήστη ότι η βάση είναι κενή. Διαφορετικά εμφανίζει τα στοιχεία της λίστας των αντικειμένων των αυτοκινήτων, των πελατών ή των επισκευών ανάλογα με την κλήση της εκάστοτε κλάσης με την χρήση της ListView σύμφωνα με τις ορισμένες ρυθμίσεις προβολής των κλάσεων CarListAdapter, CustomerListAdapter και ServiceListAdapter.

```
        if (carList == null || carList.size() == 0)
      Toast.makeText(ListCarContents.this, "The Database is empty  :(.",
Toast.LENGTH_LONG).show();
    else { CarListAdapter adapter = new CarListAdapter(this, carList);
      listView = findViewById(R.id.listView);
      listView.setAdapter(adapter); }
```

6.4.10 NewEntryCar, NewEntryCustomer, NewEntryService

Οι κλάσεις NewEntryCar , NewEntryCustomer και NewEntryService χρησιμοποιούνται για την εισαγωγή και αποθήκευση εγγραφών στην βάση, οι οποίες παραμένουν αποθηκευμένες και προβάλλονται κατά αίτηση του χρήστη, βλέπε εικόνες(41) , (39) και (43).

Ο χρήστης έχει την δυνατότητα να εισάγει και να αποθηκεύει πατώντας το κουμπί add τις εγγραφές, οι οποίες εμφανίζονται αυτόματα σε μορφή λίστας πατώντας το κουμπί view. Πιο αναλυτικά πατώντας το κουμπί εισαγωγής καλείται η μέθοδος setOnClickListener( ) για επεξεργασία συμβάντων του button, η οποία χρησιμοποιεί την μέθοδο onClick που ελέγχει αν το μήκος κάθε στοιχείου ότι είναι μεγαλύτερο των 0 χαρακτήρων. Αν δεν ισχύει εμφανίζεται το μήνυμα "You must put something in the text field!" το οποίο ενημερώνει τον χρήστη ότι πρέπει να συμπληρωθούν τα πεδία ενώ αν ισχύει καλεί την AddData( ). Με την AddData δημιουργείται ένα instance της κλάσης Car με τις παραμέτρους της και καλείται η DatabaseHelper να το σώσει. Αν το σώσει, τότε επιστρέφει true, αλλιώς false. Αν επιστρέψει true, τότε δείχνει μήνυμα "Successfully Entered Data!" που ενημερώνει τον χρήστη ότι τα δεδομένα εισήχθησαν επιτυχώς, διαφορετικά "Something went wrong :(." που ειδοποιεί τον χρήστη ότι κάτι πήγε στραβά. Αν ο χρήστης πατήσει το κουμπί view εμφανίζονται όλες οι αποθηκευμένες εγγραφές όπως φαίνεται από τις εικόνες (40) , (42), (44).

Η NewEntryCustomer περιέχει τις ίδιες μεθόδους με την NewEntryCar για αποθήκευση εγγραφών πελάτη.
Η NewEntryService αφορά την αποθήκευση εγγραφών που αφορούν τις επισκευές. Είναι εύχρηστη διότι η ημερομηνία ανανεώνεται αυτόματα και μέσω ενός διακόπτη καταχωρείται η εργασία που πραγματοποιήθηκε. Επίσης χρησιμοποιεί χειρισμό λαθών try { } catch(IllegalStateException e) {} ώστε να ενημερώνει ότι "Απέτυχε η αποθήκευση σε περίπτωση σφάλματος.
/ " + e.getMessage() όπου το e.getMessage() είναι μια επεξήγηση του τι πρόβλημα προέκυψε.

```
        void AddData(String afm, String ark, String date, String ergasia) {
          Service service = new Service(afm, ark, date, ergasia);
          try {
            boolean insertData =
DatabaseHelper.getInstance(getApplicationContext()).save(service);
```

```
                if (insertData) {
                    Toast.makeText(NewEntryService.this, "Successfully Entered Data!",
Toast.LENGTH_LONG).show();
                    finish();
                } else {
                    Toast.makeText(NewEntryService.this, "Απέτυχε η αποθήκευση. Δείτε ότι ΑΦΜ,
ΑΡΚ είναι σωστά και ότι η ΗΜ/ΝΙΑ έχει σωστή μορφή (dd/MM/yyyy)",
Toast.LENGTH_LONG).show();
                }
            } catch (IllegalStateException e) {
                Toast.makeText(NewEntryService.this, "Απέτυχε η αποθήκευση. " + e.getMessage(),
Toast.LENGTH_LONG).show();
            }
        }
    });
```

6.4.11SearchMenu

Ο χρήστης μέσα από την εφαρμογή μπορεί να κάνει γρήγορη αναζήτηση και εμφάνιση καταχωρημένων πληροφοριών όπως ορίζεται στην SearchMenu κλάση . Ο χρήστης όπως φαίνεται στην εικόνα(45) έχει την δυνατότητα να επιλέξει μεταξύ ή το ΑΦΜ ή τον ΑΡ.ΚΥΚΛΟΦΟΡΙΑΣ που θα αποτελέσει το είδος του κλειδιού αναζήτησης που θα εισάγει ο χρήστης. Πατώντας το κουμπί search μπορεί να πραγματοποιηθεί αναζήτηση είτε με βάσει το ΑΦΜ, είτε με τον αρ.κυκλοφορίας, που αποτελούν τα πρωτεύοντα κλειδιά πινάκων Customer και Cars αντίστοιχα, απ' τους οποίους θα λαμβάνονται τα ανάλογα στοιχεία. Πιο συγκεκριμένα με το πάτημα του κουμπιού Search, ο χρήστης αφού έχει επιλέξει το είδος της αναζήτησης του, λαμβάνει τα ανάλογα αποτελέσματα αυτής μέσα από την οθόνη απεικόνισης του πίνακα Customer ή του Car , όπως φαίνονται από τις εικόνες (47)και(48).
Εάν ο χρήστης δεν επιλέξει στοιχείο αναζήτησης ενημερώνεται με ένα μήνυμα ότι πρέπει να επιλέξει κάποιο τύπο αναζήτησης. Επίσης αν το μέγεθος του κλειδιού αναζήτησης είναι ίσο με το μηδέν εμφανίζεται ένα μήνυμα που ειδοποιεί τον χρήστη ότι πρέπει να ορίσει το term αναζήτησης διαφορετικά εκτελείται κανονικά η αναζήτηση. Το κουμπί cancel μεταβαίνει το χρήστη στην προηγούμενη οθόνη, δηλαδή στο βασικό μενού.

7. ΚΕΦΑΛΑΙΟ :ΣΥΜΠΕΡΑΣΜΑΤΑ

7.1 Μελλοντικές εξελίξεις

Η εφαρμογή που πραγματεύεται η παρούσα πτυχιακή έχει απλή δομή για βέλτιστη απόδοση χρόνου και προσφέρει τις βασικές λειτουργίες για γραμματειακή υποστήριξη ενός φανοποιείου αυτοκινήτων. Η συγκεκριμένη εφαρμογή μπορεί να εξελιχθεί εισάγοντας στις λειτουργίες της την επεξεργασία των εγγραφών που αποθηκεύονται. Επιπροσθέτως εξέλιξη της εφαρμογής θα αποτελεί η επέκταση των δυνατοτήτων της δηλαδή το σύστημα θα πρέπει να είναι σε θέση να εξάγει τις καταχωρημένες πληροφορίες και να τις μεταφέρει σε αρχείο στην κάρτα μνήμης, που μπορεί να μεταφερθεί και προσπελαστεί δια μέσου ηλεκτρονικού υπολογιστή ή άλλης συσκευής. Ακόμα βελτιστοποίηση της εφαρμογής θα είναι η παροχή της μέσω διαδικτύου με χρήση διακομιστή (xamp, phpmysql). Τέλος αξίζει να σημειωθεί ότι η συνεχόμενη αύξηση των υπηρεσιών που προσφέρει το android στους χρήστες δεν επιτρέπει σε καμία εφαρμογή να παραμείνει στάσιμη στο χρόνο.

7.2 Σύνοψη και συμπεράσματα

Το τεχνολογικό περιβάλλον περιλαμβάνει την πλέον "ορατή" επίδραση του εξωτερικού περιβάλλοντος στην επιχείρηση. Επηρεάζει ποιοτικά και ποσοτικά τη φύση και την απόδοση του εξοπλισμού της και των εργαζομένων, το είδος των προϊόντων και των υπηρεσιών.

Αξιοποιώντας όλα όσα προσφέρει η τεχνολογία android δηλαδή εντάσσοντας μια ή περισσότερες εφαρμογές στην επιχείρηση προσαρμοσμένες να καλύπτουν τις ανάγκες της ,εκτελούνται πιο γρήγορα και εύκολα οι διαδικασίες λειτουργία της. Με την ένταξη λοιπόν καινοτόμων εφαρμογών στην επιχείρηση επιτυγχάνεται βέλτιστη λειτουργία αυτής κάνοντας την πιο ανταγωνιστική ,εξασφαλίζοντας έτσι την επιβίωση αλλά και την ανάπτυξη της(καινοτομία και επιχειρηματικότητα). Η πλατφόρμα android παρόλο που είναι δωρεάν και εύκολη στην χρήση παρέχοντας πληθώρα δυνατοτήτων, απαιτούνται κάποιες βασικές γνώσεις προγραμματισμού όπως γνώση της JAVA που είναι μια βασική γλώσσα προγραμματισμού. Πιθανά προβλήματα που μπορεί να προκύψουν στην ανάπτυξη της εφαρμογής είναι η μη ορθή εγκατάσταση ορισμένων προγραμματιστικών εργαλείων (πχ παράληψη βιβλιοθηκών) ή μη ενημέρωση της νεότερης έκδοσης. Επομένως απαιτείται προσοχή κατά την εκτέλεση των βημάτων εγκατάστασης και πρέπει να γίνεται ενημέρωση του προγράμματος που χρησιμοποιείται σύμφωνα με τις νεότερες εκδόσεις.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1]Διπλωματική εργασία:Διαμαντής.Β(Σεπτέμβριος 2013).*Σχεδίαση και αξιολόγηση χώρο-ευαίσθητης εφαρμογής σε συσκευή Android*
[2]https://el.wikipedia.org/wiki/Android
[3]https://tecky.eu/idc-i-huawei-xeperase-tin-apple-kai-egine-o-deyteros-se-poliseis-kataskeyastis-syskeyon-smartphone/
[4]Lauren Darcey & Shane Conder (2012- *2η έκδοση*)Μάθετε την ανάπτυξη εφαρμογών για το Android σε 24 ώρες ,Εκδόσεις Μ.Γκιούρδας.
[5]www.openhandsetalliance.com/press_110507.html
[6] https://www.apache.org/foundation/
[7] http://www.dga.gr/web/publications/files/android.pdf
[8] Διπλωματική εργασία : Πραμαγκιούλης Ευστάθιος(Ιούνιος 2012) *Κατασκευή δικτυακής εφαρμογής στην αρχιτεκτονική android που υλοποιεί ένα παιχνίδι ερωτοαπαντήσεων*
[9] https://en.wikipedia.org/wiki/Android_version_history
[10]Πτυχιακή εργασία: Νίκος Παρδάλης Τ-2179-Θωμάς Καραδήμος Τ-2044(2012)
" Ανάπτυξη mobile εφαρμογής διεπαφής χρήστη, για τις υπηρεσίες του Τμήματος Πληροφορικής & Τηλεπικοινωνιών, χρησιμοποιώντας την πλατφόρμα ανοιχτού κώδικα Android SDK. "
[11]Σπυρούκλας Χριστόφορος(Ιούνιος 2014) ΑΝΑΠΤΥΞΗ ΕΦΑΡΜΟΓΗΣ ANDROID ΓΙΑ ΤΗΝ ΔΙΑΧΕΙΡΗΣΗ ΑΥΤΟΚΙΝΗΤΟΥ
[12] *DEITEL (2014 -2η έκδοση)Android Προγραμματισμός*, εκδόσεις Μ.ΓΚΙΟΥΡΔΑΣ.
[13] http://socialcompare.com/en/comparison/android-versions-comparison
[14] https://www.android.com/versions/oreo-8-0/
[15] https://techblog.gr/software/android-oreo-specs-0954/
[16] https://en.wikipedia.org/wiki/Android_Studio
[17]http://developer.android.com/sdk/index.html
[18] https://developer.android.com/studio/projects/create-project
[19] https://developer.android.com/studio/run/managing-avds
[20] https://en.wikipedia.org/wiki/Google_Play
[21]Πτυχιακή : Τσιαφίτσας Χρυσοβαλάντης *(2014)Εφαρμογή Android για το Γραφείο Φυσικής Αγωγής ΑΤΕΙ Θεσσαλίας*()
[22] www.sqlite.org
[23] http://valokafor.com/android-sqlite-database/
[24] http://developer.android.com/guide/index.html
[25] http://developer.android.com/reference/android/app/Activity.html
[26] http://www.learncomputer.com/tutorials/android/activities-and-intents/activity/
[27] http://developer.android.com/guide/topics/ui/declaring-layout.html

[28]https://medium.com/@Sudhagar/android-array-adapters-what-most-of-the-tutorials-don-t-tell-you-90f898fb54a2

Κώδικες :
http://github.com/mitchtabian/BasicLoginAndRegister -sharedpreferences
http://github.com/mitchtabian/ListAdapter -
http://myweb.teleinfom.teiep.gr/gtsoulos/public_html/mobile.php
http://developer.android.com

ΠΑΡΑΡΤΗΜΑ Α: Κώδικες των Κλάσεων της εφαρμογής.

Main Activity

```
package app.fix_car1;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

import java.util.Objects;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate( savedInstanceState );
        setContentView( R.layout.activity_main );
        //απόκρυψης της γραμμής κατάστασης
        Objects.requireNonNull( getActionBar() ).hide();

        final boolean firstRun =
getSharedPreferences(LoginActivity.MY_PREF_NAME,MODE_PRIVATE).getBoolean
("firstRun", true);
        Button btEnter = findViewById( R.id.button_enter );
        btEnter.setOnClickListener( new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent click;
                if (firstRun){
                    click = new Intent(MainActivity.this,
LoginActivity.class);
                }else {
                    click = new Intent(MainActivity.this,
MainMenuActivity.class);
                }
                startActivity(click);

            }
        });
    }
}
```

Login Activity

```
package app.fix_car1;

import android.annotation.SuppressLint;
import android.app.Activity;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

@SuppressLint("Registered")
public class
LoginActivity extends Activity {
        public final static String MY_PREF_NAME = "login_settings";
        private static final String WARN_EMPTY_FIELDS = "Fields must be
filled!";
        private EditText eUser;
        private EditText ePass;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_login);

            @SuppressLint("WrongViewCast") Button btLogin =
findViewById( R.id.button );

            eUser =  findViewById(R.id.username);
            ePass =  findViewById(R.id.password);

            SharedPreferences prefs =
getSharedPreferences(MY_PREF_NAME,MODE_PRIVATE);
            eUser.setText(prefs.getString("username",""));
            ePass.setText(prefs.getString("password",""));

            btLogin.setOnClickListener( new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    Intent click = new Intent( LoginActivity.this,
MainMenuActivity.class );
                    //αν τα πεδία =κενά
                    if (eUser.getText().toString().equals( "" ) ||
ePass.getText().toString().equals( "" )) {
                        //makeText () η μέθοδος λαμβάνει 3
παραμέτρους:πλαίσιο εφαρμογής,μήνυμα κειμένου & τη διάρκεια του toast.
                        // Εμφάνιση της ειδοποίησης toast με show ()
                        Toast.makeText( getBaseContext(),
WARN_EMPTY_FIELDS, Toast.LENGTH_LONG ).show();
                    } else {
                        SharedPreferences.Editor editor =
getSharedPreferences( MY_PREF_NAME, MODE_PRIVATE ).edit();
                        editor.putBoolean( "firstRun", false );
                        editor.putString( "username",
eUser.getText().toString() );
                        editor.putString( "password",
ePass.getText().toString() );
                        editor.apply();
                        startActivity( click );
```

```
                    }

                }
            } );
        }
}
```

MainMenu

```
package app.fix_car1;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainMenuActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main_menu);

        Button btNew =  findViewById(R.id.button_new_entry);
        Button btSearch =  findViewById(R.id.button_search);
        Button btnStatistics = findViewById(R.id.button_statistics);
        Button btOptions =  findViewById(R.id.button_options);
        Button btExit =  findViewById(R.id.button_exit);

        btNew.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent click_new = new Intent(MainMenuActivity.this,
NewEntry.class);
                startActivity(click_new);
            }
        });

        btSearch.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent click_search = new Intent(MainMenuActivity.this,
SearchMenu.class);
                startActivity(click_search);
            }
        });

        btnStatistics.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent click_stats = new Intent(MainMenuActivity.this,
ListStatsContents.class);
                startActivity(click_stats);
            }
        });

        btOptions.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent click_options = new Intent(MainMenuActivity.this,
OptionMenu.class);
```

```
                startActivity(click_options);
            }
        });

        btExit.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
    }
}
```

NewEntry

```
package app.fix_car1;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class NewEntry extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_new_entry);

        Button btNewCustomer = findViewById(R.id.button_customer);
        Button btNew_Car = findViewById(R.id.button_car);
        Button btnManager = findViewById(R.id.button_manage);
        Button btCancel = findViewById(R.id.button_cancel);

        btNewCustomer.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent click_customer = new Intent(NewEntry.this,
NewEntryCustomer.class);
                startActivity(click_customer);
            }
        });

        btNew_Car.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent click_car = new Intent(NewEntry.this,
NewEntryCar.class);
                startActivity(click_car);
            }
        });

        btnManager.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent click_car = new Intent(NewEntry.this,
NewEntryService.class);
                startActivity(click_car);
            }
        });
```

```
        btCancel.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();

            }
        });
    }
}
```

NewEntryCar

```
package app.fix_car1;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class NewEntryCar extends Activity {

    Button btnView, btnAdd;
    EditText etAfm, etArk, etInsurance, etBrand, etModel, etColor;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_new_entry_car);
        btnView = findViewById(R.id.btnView);
        btnAdd = findViewById(R.id.btnAdd);
        etAfm = findViewById(R.id.etAfm);
        etArk = findViewById(R.id.etArk);
        etInsurance = findViewById(R.id.etInsurance);
        etBrand = findViewById(R.id.etBrand);
        etModel = findViewById(R.id.etModel);
        etColor = findViewById(R.id.etColor);

        btnView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(NewEntryCar.this,
ListCarContents.class);
                startActivity(intent);
            }

        });

        btnAdd.setOnClickListener(new View.OnClickListener() {
            @Override
            //η μέθοδος onClick ελέγχει οτι το μήκος κάθε στοιχείου
είναι μεγαλύτερο των 0 χαρακτήρων
            public void onClick(View v) {
                String fAfm = etAfm.getText().toString();
                String rArk = etArk.getText().toString();
                String iInsurance = etInsurance.getText().toString();
                String bBrand = etBrand.getText().toString();
                String mModel = etModel.getText().toString();
                String cColor = etColor.getText().toString();
```

```
                if (fAfm.length() != 0 && rArk.length() != 0 &&
iInsurance.length() != 0 && bBrand.length() != 0 && mModel.length() != 0
&& cColor.length() != 0) {
                    AddData(fAfm, rArk, iInsurance, bBrand, mModel,
cColor);
                    etAfm.setText("");
                    etArk.setText("");
                    etInsurance.setText("");
                    etBrand.setText("");
                    etModel.setText("");
                    etColor.setText("");
                } else {
                    Toast.makeText(NewEntryCar.this, "You must put
something in the text field!", Toast.LENGTH_LONG).show();
                }

            }

            void AddData(String afm, String ark, String insurance,
String brand, String model, String color) {
                Car car = new Car(afm, ark, insurance, brand, model,
color);
                boolean insertData =
DatabaseHelper.getInstance(getApplicationContext()).save(car);
                if (insertData) {
                    Toast.makeText(NewEntryCar.this, "Successfully
Entered Data!", Toast.LENGTH_LONG).show();
                } else {
                    Toast.makeText(NewEntryCar.this, "Something went
wrong :(.", Toast.LENGTH_LONG).show();
                }

            }

        });
    }
}
```

Car

```
package app.fix_car1;

import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import java.util.ArrayList;
import java.util.List;

/**
 * ΟΧΗΜΑ
 * =====
 * ΑΦΜ
 * ΑΡ. ΚΥΚΛΟΦΟΡΙΑΣ
 * ΑΣΦΑΛΙΣΤΙΚΗ
 * ΜΑΡΚΑ
 * ΜΟΝΤΕΛΟ
 * ΧΡΩΜΑ
 */
public class Car {

    private String afm;
    private String ark;
```

```
    private String insurance;
    private String brand;
    private String model;
    private String color;
//create parameterized constructor will be used to add value to the
table
    public Car(String afm, String ark, String insurance, String brand,
String model, String color) {
        this.afm = afm;
        this.ark = ark;
        this.insurance = insurance;
        this.brand = brand;
        this.model = model;
        this.color = color;
    }

    //create empty constructor
    private Car() {

    }

    //create the getter and setters for all instances variables
    public String getAfm() {
        return afm;
    }

    public void setAfm(String afm) {
        this.afm = afm;
    }

    public String getArk() {
        return ark;
    }

    public void setArk(String ark) {
        this.ark = ark;
    }

    public String getInsurance() {
        return insurance;
    }

    public void setInsurance(String insurance) {
        this.insurance = insurance;
    }

    public String getBrand() {
        return brand;
    }

    public void setBrand(String brand) {
        this.brand = brand;
    }

    public String getModel() {
        return model;
    }

    public void setModel(String model) {
        this.model = model;
    }
```

```
    public String getColor() {
        return color;
    }

    public void setColor(String color) {
        this.color = color;
    }

    /**
     * Database Code
     */
    public static String TABLE_NAME = "car";
    public static String COL_AFM = "AFM";
    public static String COL_ARK = "ARK";
    public static String COL_INSURANCE = "INSURANCE";
    public static String COL_BRAND = "BRAND";
    public static String COL_MODEL = "MODEL";
    public static String COL_COLOR = "COLOR";


    public static String getCreateTable() {

        String tmp = "CREATE TABLE " + TABLE_NAME;
        tmp += " (ID INTEGER PRIMARY KEY AUTOINCREMENT, ";
                final String[] COLUMNS = {COL_AFM, COL_ARK,
COL_INSURANCE, COL_BRAND, COL_MODEL, COL_COLOR};
                for (int i = 0; i < COLUMNS.length; i++) {
            if (i > 0) {
                tmp += ",";
            }
            tmp += COLUMNS[i] + " TEXT";
        }

        tmp += ")";

        return tmp;
    }

       public static Car parse(Cursor cursor) {
        //create an object car from car class
        Car car = new Car();
        //get all data for the database
        car.setAfm(cursor.getString(cursor.getColumnIndex(COL_AFM)));
        car.setArk(cursor.getString(cursor.getColumnIndex(COL_ARK)));

car.setInsurance(cursor.getString(cursor.getColumnIndex(COL_INSURANCE)))
;

car.setBrand(cursor.getString(cursor.getColumnIndex(COL_BRAND)));

car.setModel(cursor.getString(cursor.getColumnIndex(COL_MODEL)));

car.setColor(cursor.getString(cursor.getColumnIndex(COL_COLOR)));
        return car;
    }

    public boolean save(SQLiteDatabase db) {
        ContentValues contentValues = new ContentValues();
        contentValues.put(COL_AFM, getAfm());
        contentValues.put(COL_ARK, getArk());
```

```
        contentValues.put(COL_INSURANCE, getInsurance());
        contentValues.put(COL_BRAND, getBrand());
        contentValues.put(COL_MODEL, getModel());
        contentValues.put(COL_COLOR, getColor());
       // Η εντολή replace σώζει την εγγραφή, όπως έχει περιγραφεί στο
contentValues.
        // Η διαφορά της από την insert ..αν υπάρχει ηδη εγγραφή στην
βάση με ιδιο πρωτεύον κλειδί, τότε την αντικαθιστά
        long result = db.replace(TABLE_NAME, null, contentValues);

        //if date as inserted incorrectly it will return -1
        if (result == -1) {
            return false;
        } else {
            return true;
        }
    }

    public static List<Car> list(SQLiteDatabase db, String ak) {
        final List<Car> toReturn = new ArrayList<>();
        if (ak == null || ak.length() == 0) {
            ak = null;
        } else {
                        ak = ak.trim();
        }

        try (Cursor cursor = db.rawQuery("SELECT * FROM " + TABLE_NAME +
(ak == null ? "" : " WHERE " + COL_ARK + " like ?"), (ak == null ? null
: new String[]{ak + "%"}))) {

            while (cursor.moveToNext()) {
                Car car = Car.parse(cursor);
                toReturn.add(car);
            }
        }
        return toReturn;
    }
}
```

CarListAdapter

```
package app.fix_car1;

import android.annotation.SuppressLint;
import android.content.Context;
import android.support.annotation.NonNull;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;

import java.util.List;

public class CarListAdapter extends ArrayAdapter<Car> {
    //(α) mInflater ειναι ένα εργαλείο που μετατρέπει ένα layout xml στο
αντιστοιχο View object,
    // (β) Κάθε layout xml καθως και γενικότερα οτιδήποτε υπάρχει στον
φάκελο res, δημιουργεί έναν μοναδικό int id,
    // το id για το layout ..για κάθε γραμμή.
    //Επιστρέφει μια προβολή για κάθε αντικείμενο σε μια συλλογή
αντικειμένων δεδομένων που παρέχετε
```

```
    // και μπορεί να χρησιμοποιηθεί με widgets για τη διεπαφή χρήστη που
βασίζονται σε λίστες
    private LayoutInflater mInflater;
    private List<Car> cars;
    private int mViewResourceId;

    CarListAdapter(Context context, List<Car> cars) {
        super(context, R.layout.car_list_adapter_view, cars);
        this.cars = cars;
        mInflater = (LayoutInflater)
context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        mViewResourceId = R.layout.car_list_adapter_view;
    }

   // επιστρέφει πόσα στοιχεία δείχνεις, δείχνω όσα ειναι τα αυτοκίνητα
κι ενα επιπλέον για τις επικεφαλίδες.
    @Override
    public int getCount() {
        return cars == null ? 1 : cars.size() + 1;
    }
    //Ο ArrayAdapter σε υποχρεώνει να υλοποιήσεις μια μέθοδο getView που
επιστρέφει το τι θα εμφανιστεί για κάθε γραμμή της λίστας.
    // Στην πρώτη γραμμή εμφανίζονται οι τίτλοι και στις υπόλοιπες τα
αυτοκίνητα.
    @SuppressLint("ViewHolder")
    @NonNull
    public View getView(int position, View convertView, @NonNull
ViewGroup parent) {
        convertView = mInflater.inflate(mViewResourceId, null);
        //έλεγχος ότι αυτή η προβολή δεν είναι μηδενική και κατάλληλου
τύπου πριν τη χρήση.
        // που βασίζεται στην δομή που έχει περιγραφή στο layout xml με
το id που εχει οριστεί το mViewResourceId.

        TextView afm = convertView.findViewById(R.id.text_Afm);
        TextView ark = convertView.findViewById(R.id.text_Ark);
        TextView insurance =
convertView.findViewById(R.id.text_Insurance);
        TextView brand = convertView.findViewById(R.id.text_Brand);
        TextView model = convertView.findViewById(R.id.text_Model);
        TextView color = convertView.findViewById(R.id.text_Color);
//Επειδή έχουμε εμφανίσει τους τίτλους στην πρώτη, τα αυτοκινητα ->σε
επόμενο position,
// το όχημα που είναι στην θέση 0 θα εμφανιστεί στην θέση 1 της λίστας,
// άρα η θέση στην ArrayList cars είναι η θέση στην ListView - 1 =>
cars.get(position - 1)
        Car car = position == 0 ? null : cars.get(position - 1);

        if (car == null) {
            afm.setText("ΑΦΜ");
            ark.setText(("ΑΡΚ"));
            insurance.setText("ΑΣΦΑΛΕΙΑ");
            brand.setText("ΜΑΡΚΑ");
            model.setText("ΜΟΝΤΕΛΟ");
            color.setText("ΧΡΩΜΑ");
        } else {
            afm.setText(car.getAfm());
            ark.setText((car.getArk()));
            insurance.setText((car.getInsurance()));
            brand.setText((car.getBrand()));
            model.setText((car.getModel()));
```

```
            color.setText((car.getColor()));
        }
//Επιστρέφει το view, αφού έχει κανει και set τις τιμές που πρέπει να
φανούν σε κάθε "κολώνα"
        return convertView;
    }
}
```

ListCarContent

```
package app.fix_car1;

import android.app.Activity;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.widget.ListView;
import android.widget.Toast;

import java.util.List;

public class ListCarContents extends Activity {

    ListView listView;
   //savedInstanceState καλείται απ'το σύστημα όταν αλλάζει η
παραμετροποίηση της εφαρμογής (πχ περιστροφή οθόνης)
   // και χρησιμοποιείται για να αποθηκεύει πληροφορίες που θέλουμε να
επαναφέρουμε ,όταν καλείται η OnCreate .keyword παρασκήνιο
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.viewcontents_layout);
        //Η αναζήτηση περνάει μια "παράμετρο" με κλείδι "q". Αυτο ειναι
το keyword βαση του οποιου ψαχνεις.
        final String ak = getIntent().getStringExtra("q");

        final List<Car> carList =
DatabaseHelper.getInstance(this).getCars(ak);
        if (carList == null || carList.size() == 0)
            Toast.makeText(ListCarContents.this, "The Database is empty
:(.", Toast.LENGTH_LONG).show();
        else {
            CarListAdapter adapter = new CarListAdapter(this, carList);
            listView = findViewById(R.id.listView);
            listView.setAdapter(adapter);
        }
    }
}
```

DatabaseHelper

```
package app.fix_car1;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import java.util.List;

public class DatabaseHelper extends SQLiteOpenHelper {
      //όνομα βάσης δεδομένων
    public static final String DATABASE_NAME = "fix_cars.db";

    private static DatabaseHelper sInstance;
```

```
    public static synchronized DatabaseHelper getInstance(Context
context) {

        if (sInstance == null) {
            sInstance = new
DatabaseHelper(context.getApplicationContext());
        }
        return sInstance;
    }
      //δημιουργός για την DatabaseHelper-δημιουργεί ένα νέο
    private DatabaseHelper(Context context) {
        super(context, DATABASE_NAME, null, 1);
    }

    //δημιουργείτους πίνακες όταν δημιουργείται η βάση
    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(Car.getCreateTable());//εκτελεί ερώτημα για να
δημιουργήσει
        db.execSQL(Customer.getCreateTable());
        db.execSQL(Service.getCreateTable());
    }
    //αναβάθμιση της βάσης..αλλαγή στο σχήμα της βάσης δεδομένων
συνεπάγεται αύξηση της έκδοσης της ΒΔ.
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {

    }

    // ----- Customers -----
    //δημιουργεί ή ανοίγει μια βάση δεδομένων για εγγραφή/ανάγνωση
    //getWritableDatabase() κληρονομείται απ'την
SQLiteOpenHelper,επιστρέφει αντικείμενα SQLiteDatabase
    public boolean save(Customer customer) {
        //get permission to write to the database
        SQLiteDatabase db = this.getWritableDatabase();
        return customer.save(db);
    }

    public List<Customer> getCustomers(String afm) {
        //get the context to read the database
        SQLiteDatabase db = this.getReadableDatabase();
        return Customer.list(db, afm);
    }

    // ----- Cars -----
    public boolean save(Car car) {
        SQLiteDatabase db = this.getWritableDatabase();
        return car.save(db);
    }

    public List<Car> getCars(String ak) {
        SQLiteDatabase db = this.getReadableDatabase();
        return Car.list(db, ak);
    }

    // ----- Services -----
    public boolean save(Service service) {
        SQLiteDatabase db = this.getWritableDatabase();
```

```
        return service.save(db);
    }

    public List<Service> getServices(String ak) throws
IllegalStateException {
        SQLiteDatabase db = this.getReadableDatabase();
        return Service.list(db, ak);
    }

    // ----- Statistics -----
    public List<String[]> getStatistics() {
        SQLiteDatabase db = this.getReadableDatabase();
        return Service.stats(db);
    }
}
```

NewEntryCustomer

```
package app.fix_car1;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class NewEntryCustomer extends Activity {

    //private static final String WARN_EMPTY_FIELDS = "Afm and Ark must
be filled";

    Button btnView, btnAdd;
    EditText etAfm, etFirstName, etLastName, etAddress, etTelephone,
etMobile, etFax, etEmail, etDoy;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_new_entry_customer);
        btnView = findViewById(R.id.btnView);
        btnAdd = findViewById(R.id.btnAdd);

        etAfm = findViewById(R.id.text_afm);
        etFirstName = findViewById(R.id.text_name);
        etLastName = findViewById(R.id.text_lastname);
        etAddress = findViewById(R.id.text_address);
        etTelephone = findViewById(R.id.text_phone);
        etMobile = findViewById(R.id.text_mobile);
        etFax = findViewById(R.id.text_fax);
        etEmail = findViewById(R.id.text_email);
        etDoy = findViewById(R.id.text_doy);
// extra οθόνης
        //Button btCancel = (Button) findViewById( R.id.button_cancel );
        // Button btClear = (Button) findViewById( R.id.button_clear );

        //  btCancel.setOnClickListener( new View.OnClickListener() {
        //@Override
        //public void onClick(View v) {
        //   finish();
        // }
```

```
        // } );

        //   btClear.setOnClickListener( new View.OnClickListener() {
        // @Override
        //  public void onClick(View v) {
        //   etAfm.setText( "" );
        //  etArk.setText( "" );
        //  etInsurance.setText( "" );
        // etBrand.setText( "" );
        //  etModel.setText( "" );
        //  etColor.setText( "" );
        // }
        // } );

        btnView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(NewEntryCustomer.this,
ListCustomerContents.class);
                startActivity(intent);
            }

        });

        btnAdd.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String afm = etAfm.getText().toString();
                String firstName = etFirstName.getText().toString();
                String lastName = etLastName.getText().toString();
                String address = etAddress.getText().toString();
                String telephone = etTelephone.getText().toString();
                String mobile = etMobile.getText().toString();
                String fax = etFax.getText().toString();
                String email = etEmail.getText().toString();
                String doy = etDoy.getText().toString();

                if (afm.length() != 0 && firstName.length() != 0 &&
lastName.length() != 0 && address.length() != 0 && telephone.length() !=
0 && mobile.length() != 0 && fax.length() != 0 && email.length() != 0 &&
doy.length() != 0) {
                    AddData(afm, firstName, lastName, address,
telephone, mobile, fax, email, doy);


                } else {
                    Toast.makeText(NewEntryCustomer.this, "You must put
something in the text field!", Toast.LENGTH_LONG).show();
                }

            }

            void AddData(String afm, String firstName, String lastName,
String address, String telephone, String mobile, String fax, String
email, String doy) {
                Customer customer = new Customer(afm, firstName,
lastName, address, telephone, mobile, fax, email, doy);
                boolean insertData =
DatabaseHelper.getInstance(getApplicationContext()).save(customer);
                if (insertData) {
                    Toast.makeText(NewEntryCustomer.this, "Successfully
```

```
Entered Data!", Toast.LENGTH_LONG).show();
                } else {
                    Toast.makeText(NewEntryCustomer.this, "Something
went wrong :(.", Toast.LENGTH_LONG).show();
                }

            }

        });
    }
}
```

NewEntryService

```
package app.fix_car1;

import android.annotation.SuppressLint;
import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Switch;
import android.widget.Toast;

import java.text.SimpleDateFormat;
import java.util.Date;

public class NewEntryService extends Activity {

    Button btnView, btnAdd;
    EditText etAfm, etArk, etDate;
    Switch etIsLocalPaint;

    @SuppressLint("SimpleDateFormat")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_new_entry_service);
        btnView = findViewById(R.id.btnView);
        btnAdd = findViewById(R.id.btnAdd);

        etAfm = findViewById(R.id.text_afm);
        etArk = findViewById(R.id.text_ark);
        etDate = findViewById(R.id.text_date);
        etIsLocalPaint = findViewById(R.id.is_local_paint);

        etDate.setText(new SimpleDateFormat("dd/MM/yyyy").format(new
Date()));

        btnView.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(NewEntryService.this,
ListServiceContents.class);
                startActivity(intent);
            }

        });

        btnAdd.setOnClickListener(new View.OnClickListener() {
```

```
            @Override
            public void onClick(View v) {
                String afm = etAfm.getText().toString();
                String ark = etArk.getText().toString();
                String date = etDate.getText().toString();
                String ergasia = etIsLocalPaint.isChecked() ? "Τοπική
βαφή" : "Ολική βαφή";

                if (afm.length() != 0 && ark.length() != 0 &&
date.length() != 0 && ergasia.length() != 0) {
                    AddData(afm, ark, date, ergasia);
                } else {
                    Toast.makeText(NewEntryService.this, "You must put
something in the text field!", Toast.LENGTH_LONG).show();
                }

            }

            void AddData(String afm, String ark, String date, String
ergasia) {
                Service service = new Service(afm, ark, date, ergasia);
                try {
                    boolean insertData =
DatabaseHelper.getInstance(getApplicationContext()).save(service);
                    if (insertData) {
                        Toast.makeText(NewEntryService.this,
"Successfully Entered Data!", Toast.LENGTH_LONG).show();
                        finish();
                    } else {
                        Toast.makeText(NewEntryService.this, "Απέτυχε η
αποθήκευση. Δείτε οτι ΑΦΜ, ΑΡΚ ειναι σωστά και οτι η ΗΜ/ΝΙΑ έχει σωστή
μορφή (dd/MM/yyyy)", Toast.LENGTH_LONG).show();
                    }
                } catch (IllegalStateException e) {
                    Toast.makeText(NewEntryService.this, "Απέτυχε η
αποθήκευση. " + e.getMessage(), Toast.LENGTH_LONG).show();
                }
            }

        });
    }
}
```

OptionMenu

```
package app.fix_car1;

import android.app.Activity;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class OptionMenu extends Activity {

    private static final String WARN_EMPTY_FIELDS = "Fields must be
filled!";
    private EditText eUser;
    private EditText ePass;
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_option_menu);

        Button btSave =  findViewById( R.id.button );

        eUser =  findViewById(R.id.login);
        ePass =  findViewById(R.id.password);

        SharedPreferences prefs =
getSharedPreferences(LoginActivity.MY_PREF_NAME, MODE_PRIVATE);
        eUser.setText(prefs.getString("username", ""));
        ePass.setText(prefs.getString("password", ""));
        //key:θέτουμε ποιά στοιχεία θα αποτελούν το login key of shared
preferences
        //we can also change the key to open the app
        btSave.setOnClickListener( new View.OnClickListener() {
    Intent click = new Intent( OptionMenu.this, LoginActivity.class );
            if (eUser.getText().toString().equals("") ||
ePass.getText().toString().equals("") ){
                Toast.makeText(getBaseContext(), WARN_EMPTY_FIELDS,
Toast.LENGTH_LONG).show();
            } else {//edit on key of shared preferences
                SharedPreferences.Editor editor =
getSharedPreferences(LoginActivity.MY_PREF_NAME, MODE_PRIVATE).edit();
                editor.putString("username",
eUser.getText().toString());
                //shared preferences save the information
(username,password)
                editor.putString("password",
ePass.getText().toString());
                //εφαρμογή των κλειδιών of shared preferences
                editor.apply();
                //Toast.makeText(OptionMenu.this, "Successfull save!",
Toast.LENGTH_LONG).show();
                startActivity( click );
                //   finish();
            }
        }
    });
}
```

SearchMenu

```
package app.fix_car1;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.Toast;

public class SearchMenu extends Activity {

    RadioButton radioButton_afm, radioButton_ak;
    EditText keyword;
    Button button_search, button_cancel;
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_search_menu);

        radioButton_afm = findViewById(R.id.radioButton_afm);
        radioButton_ak = findViewById(R.id.radioButton_ak);
        keyword = findViewById(R.id.keyword);

        button_search = findViewById(R.id.button_search);
        button_cancel = findViewById(R.id.button_cancel);

        button_search.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                boolean hasSelectedType = radioButton_afm.isChecked() ||
radioButton_ak.isChecked();
                boolean searchCustomer = radioButton_afm.isChecked();
                String q = keyword.getText().toString();

                if (!hasSelectedType) {
                    Toast.makeText(SearchMenu.this, "You must select
search type (ΑΦΜ, ΑΡ. ΚΥΚΛΟΦΟΡΙΑΣ)", Toast.LENGTH_LONG).show();
                } else if (q.length() == 0) {
                    Toast.makeText(SearchMenu.this, "You must define
search term (ΑΦΜ/ΑΡ. ΚΥΚΛΟΦΟΡΙΑΣ)", Toast.LENGTH_LONG).show();
                } else if (searchCustomer) {
                    Intent intent = new Intent(SearchMenu.this,
ListCustomerContents.class);
                    intent.putExtra("q", q);
                    startActivity(intent);
                } else {
                    Intent intent = new Intent(SearchMenu.this,
ListCarContents.class);
                    intent.putExtra("q", q);
                    startActivity(intent);
                }
            }
        });
        button_cancel.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
    }
}
```

StatisticsEntry

```
package app.fix_car1;

public class StatisticsEntry {}
```

StatsListAdapter

```
package app.fix_car1;

import android.annotation.SuppressLint;
import android.content.Context;
import android.support.annotation.NonNull;
```

```
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;

import java.util.List;

public class StatsListAdapter extends ArrayAdapter<String[]> {

    private LayoutInflater mInflater;
    private List<String[]> data;
    private int mViewResourceId;

    StatsListAdapter(Context context, List<String[]> data) {
        super(context, R.layout.stats_list_adapter_view, data);
        this.data = data;
        mInflater = (LayoutInflater)
context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        mViewResourceId = R.layout.stats_list_adapter_view;
    }

    @SuppressLint("ViewHolder")
    @NonNull
    public View getView(int position, View convertView, @NonNull
ViewGroup parent) {
        convertView = mInflater.inflate(mViewResourceId, null);
        TextView tYear = convertView.findViewById(R.id.text_year);
        TextView tMonth = convertView.findViewById(R.id.text_month);
        TextView tCount = convertView.findViewById(R.id.text_count);

        String[] rowData = data.get(position);
        tYear.setText(rowData[0]);
        tMonth.setText(rowData[1]);
        tCount.setText(rowData[2]);

        return convertView;
    }
}
```

ListStatsContents

```
package app.fix_car1;

import android.app.Activity;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.widget.ListView;
import android.widget.Toast;

import java.util.List;

public class ListStatsContents extends Activity {

    ListView listView;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.viewcontents_layout);

        final List<String[]> statistics =
```

```
DatabaseHelper.getInstance(this).getStatistics();
        if (statistics == null || statistics.size() == 0)
            Toast.makeText(ListStatsContents.this, "The Database is
empty  :(.", Toast.LENGTH_LONG).show();
        else {
            StatsListAdapter adapter = new StatsListAdapter(this,
statistics);
            listView = findViewById(R.id.listView);
            listView.setAdapter(adapter);
        }
    }
}
```

Customer

```
package app.fix_car1;

import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;

import java.util.ArrayList;
import java.util.List;

/**
 * ΠΕΛΑΤΗΣ
 * =======
 * ΑΦΜ
 * ΟΝΟΜΑ
 * ΕΠΩΝΥΜΟ
 * ΔΙΕΥΘΥΝΣΗ
 * ΤΗΛΕΦΩΝΟ
 * ΚΙΝΗΤΟ
 * FAX
 * Email
 * Δ.Ο.Υ.
 */

public class Customer {

    private String afm;
    private String firstName;
    private String lastName;
    private String address;
    private String telephone;
    private String mobile;
    private String fax;
    private String email;
    private String doy;

    public Customer(String afm, String firstName, String lastName,
String address, String telephone, String mobile, String fax, String
email, String doy) {
        this.afm = afm;
        this.firstName = firstName;
        this.lastName = lastName;
        this.address = address;
        this.telephone = telephone;
        this.mobile = mobile;
        this.fax = fax;
        this.email = email;
        this.doy = doy;
```

```
}

private Customer() {

}

public String getAfm() {
    return afm;
}

public void setAfm(String afm) {
    this.afm = afm;
}

public String getFirstName() {
    return firstName;
}

public void setFirstName(String firstName) {
    this.firstName = firstName;
}

public String getLastName() {
    return lastName;
}

public void setLastName(String lastName) {
    this.lastName = lastName;
}

public String getAddress() {
    return address;
}

public void setAddress(String address) {
    this.address = address;
}

public String getTelephone() {
    return telephone;
}

public void setTelephone(String telephone) {
    this.telephone = telephone;
}

public String getMobile() {
    return mobile;
}

public void setMobile(String mobile) {
    this.mobile = mobile;
}

public String getFax() {
    return fax;
}

public void setFax(String fax) {
    this.fax = fax;
}
```

```
    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getDoy() {
        return doy;
    }

    public void setDoy(String doy) {
        this.doy = doy;
    }

    /**
     * Database Code
     */
    public static String TABLE_NAME = "customer";
    public static String COL_AFM = "AFM";
    public static String COL_ONOMA = "FNAME";
    public static String COL_EPONYMO = "LNAME";
    public static String COL_ADDRESS = "ADDRESS";
    public static String COL_TELEPHONE = "TEL";
    public static String COL_MOBILE = "MOB";
    public static String COL_FAX = "FAX";
    public static String COL_EMAIL = "EMAIL";
    public static String COL_DOY = "DOY";

    public static String getCreateTable() {

        String tmp = "CREATE TABLE " + TABLE_NAME;
        tmp += " (" + COL_AFM + " TEXT PRIMARY KEY, ";

        final String[] COLUMNS = {COL_ONOMA, COL_EPONYMO, COL_ADDRESS,
COL_TELEPHONE, COL_MOBILE, COL_FAX, COL_EMAIL, COL_DOY};
        for (int i = 0; i < COLUMNS.length; i++) {
            if (i > 0) {
                tmp += ",";
            }
            tmp += COLUMNS[i] + " TEXT";
        }

        tmp += ")";

        return tmp;
    }
    //μέθοδος που παίρνει έναν Δρομέα και επιστρέφει ένα αντικείμενο
Πελάτη .
    public static Customer parse(Cursor cursor) {
        Customer customer = new Customer();

customer.setAddress(cursor.getString(cursor.getColumnIndex(COL_ADDRESS))
);

customer.setAfm(cursor.getString(cursor.getColumnIndex(COL_AFM)));

customer.setFirstName(cursor.getString(cursor.getColumnIndex(COL_ONOMA))
);
```

```
customer.setLastName(cursor.getString(cursor.getColumnIndex(COL_EPONYMO)
));

customer.setAddress(cursor.getString(cursor.getColumnIndex(COL_ADDRESS))
);

customer.setTelephone(cursor.getString(cursor.getColumnIndex(COL_TELEPHO
NE)));

customer.setMobile(cursor.getString(cursor.getColumnIndex(COL_MOBILE)));

customer.setFax(cursor.getString(cursor.getColumnIndex(COL_FAX)));

customer.setEmail(cursor.getString(cursor.getColumnIndex(COL_EMAIL)));

customer.setDoy(cursor.getString(cursor.getColumnIndex(COL_DOY)));
        return customer;
    }
    //μέθοδος που αποθηκεύει τα στοιχεία των πελατών στην βάση
δεδομένων-εισαγωγή εγγραφής πελατών
    public boolean save(SQLiteDatabase db) {
        ContentValues contentValues = new ContentValues();
        contentValues.put(COL_AFM, getAfm());
        contentValues.put(COL_ONOMA, getFirstName());
        contentValues.put(COL_EPONYMO, getLastName());
        contentValues.put(COL_ADDRESS, getAddress());
        contentValues.put(COL_TELEPHONE, getTelephone());
        contentValues.put(COL_MOBILE, getMobile());
        contentValues.put(COL_FAX, getFax());
        contentValues.put(COL_EMAIL, getEmail());
        contentValues.put(COL_DOY, getDoy());

        long result = db.replace(TABLE_NAME, null, contentValues);

        if (result == -1) {
            return false;
        } else {
            return true;
        }
    }

    public static List<Customer> list(SQLiteDatabase db, String afm) {
        final List<Customer> toReturn = new ArrayList<>();
        if (afm == null || afm.length() == 0) {
            afm = null;
        } else {
            afm = afm.trim();
        }

        try (Cursor cursor = db.rawQuery("SELECT * FROM " + TABLE_NAME +
(afm == null ? "" : " WHERE " + COL_AFM + " like ?"), (afm == null ?
null : new String[]{afm + "%"}))) {
            while (cursor.moveToNext()) {
                Customer customer = Customer.parse(cursor);
                toReturn.add(customer);
            }
        }
        return toReturn;
    }
}
```

CustomerListAdapter

```
package app.fix_car1;

import android.annotation.SuppressLint;
import android.content.Context;
import android.support.annotation.NonNull;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;

import java.util.List;

public class CustomerListAdapter extends ArrayAdapter<Customer> {

    private LayoutInflater mInflater;
    private List<Customer> customers;
    private int mViewResourceId;

    CustomerListAdapter(Context context, List<Customer> customers) {
        super(context, R.layout.customer_list_adapter_view, customers);
        this.customers = customers;
        mInflater = (LayoutInflater)
context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        mViewResourceId = R.layout.customer_list_adapter_view;
    }

    @Override
    public int getCount() {
        return customers == null ? 1 : customers.size() + 1;
    }

    @SuppressLint("ViewHolder")
    @NonNull
    public View getView(int position, View convertView, @NonNull
ViewGroup parent) {
        convertView = mInflater.inflate(mViewResourceId, null);
        TextView tAfm = convertView.findViewById(R.id.text_afm);
        TextView tFirstName =
convertView.findViewById(R.id.text_firstName);
        TextView tLastName =
convertView.findViewById(R.id.text_lastName);
        TextView tAddress = convertView.findViewById(R.id.text_address);
        TextView tTelephone =
convertView.findViewById(R.id.text_telephone);
        TextView tMobile = convertView.findViewById(R.id.text_mobile);
        TextView tFax = convertView.findViewById(R.id.text_fax);
        TextView tEmail = convertView.findViewById(R.id.text_email);
        TextView tDoy = convertView.findViewById(R.id.text_doy);

        Customer customer = position == 0 ? null :
customers.get(position - 1);
        if (customer == null) {
            tAfm.setText("ΑΦΜ");
            tFirstName.setText("ΟΝΟΜΑ");
            tLastName.setText("ΕΠΩΝΥΜΟ");
            tAddress.setText("ΔΙΕΥΘΥΝΣΗ");
            tTelephone.setText("ΤΗΛΕΦΩΝΟ");
            tMobile.setText("ΚΙΝΗΤΟ");
            tFax.setText("FAX");
```

```
            tEmail.setText("EMAIL");
            tDoy.setText("ΔΟΥ");
        } else {
            tAfm.setText(customer.getAfm());
            tFirstName.setText(customer.getFirstName());
            tLastName.setText(customer.getLastName());
            tAddress.setText(customer.getAddress());
            tTelephone.setText(customer.getTelephone());
            tMobile.setText(customer.getMobile());
            tFax.setText(customer.getFax());
            tEmail.setText(customer.getEmail());
            tDoy.setText(customer.getDoy());
        }
        return convertView;
    }
}
```

ListCustomerContents

```
package app.fix_car1;
import android.app.Activity;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.widget.ListView;
import android.widget.Toast;

import java.util.List;
public class ListCustomerContents extends Activity {
    ListView listView;
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.viewcontents_layout);
        final String afm = getIntent().getStringExtra("q");
        final List<Customer> customerList =
DatabaseHelper.getInstance(this).getCustomers(afm);
        if (customerList == null || customerList.size() == 0)
            Toast.makeText(ListCustomerContents.this, "The Database is
empty  :(.", Toast.LENGTH_LONG).show();
        else {
            CustomerListAdapter adapter = new CustomerListAdapter(this,
customerList);
            listView = findViewById(R.id.listView);
            listView.setAdapter(adapter);
        }
    }
}
```

Service

```
package app.fix_car1;

import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;

import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;

/**
 * ΑΦΜ
 * APK
```

```
 * ΗΜ/ΝΙΑ
 * ΕΡΓΑΣΙΑ
 */
public class Service {

    public static final String DATE_FORMAT = "dd/MM/yyyy";

    private String afm;
    private String ark;
    private String date;
    private String ergasia;

    private Service() {

    }

    public Service(String afm, String ark, String date, String ergasia)
{
        this.afm = afm;
        this.ark = ark;
        this.date = date;
        this.ergasia = ergasia;
    }

    public String getAfm() {
        return afm;
    }

    public void setAfm(String afm) {
        this.afm = afm;
    }

    public String getArk() {
        return ark;
    }

    public void setArk(String ark) {
        this.ark = ark;
    }

    public String getDate() {
        return date;
    }

    public void setDate(String date) {
        this.date = date;
    }

    public String getErgasia() {
        return ergasia;
    }

    public void setErgasia(String ergasia) {
        this.ergasia = ergasia;
    }

    /**
     * Database Code
     */
    public static String TABLE_NAME = "task";
    public static String COL_AFM = "afm";
```

```
    public static String COL_ARK = "ark";
    public static String COL_DATE_YEAR = "date_year";
    public static String COL_DATE_MONTH = "date_month";
    public static String COL_DATE_DAY = "date_day";
    public static String COL_ERGASIA = "ergasia";

    public static String getCreateTable() {

        String tmp = "CREATE TABLE " + TABLE_NAME;
        tmp += " (ID INTEGER PRIMARY KEY AUTOINCREMENT, ";

        final String[] COLUMNS = {COL_AFM, COL_ARK, COL_DATE_YEAR,
COL_DATE_MONTH, COL_DATE_DAY, COL_ERGASIA};
        for (int i = 0; i < COLUMNS.length; i++) {
            if (i > 0) {
                tmp += ",";
            }
            tmp += COLUMNS[i] + " TEXT";
        }

        tmp += ")";

        return tmp;
    }

    public static Service parse(Cursor cursor) {
        Service car = new Service();
        car.setAfm(cursor.getString(cursor.getColumnIndex(COL_AFM)));
        car.setArk(cursor.getString(cursor.getColumnIndex(COL_ARK)));
        int year = cursor.getInt(cursor.getColumnIndex(COL_DATE_YEAR));
        int month =
cursor.getInt(cursor.getColumnIndex(COL_DATE_MONTH));
        int day = cursor.getInt(cursor.getColumnIndex(COL_DATE_DAY));

        car.setDate(new SimpleDateFormat(DATE_FORMAT).format(new
Date(year, month, day)));

car.setErgasia(cursor.getString(cursor.getColumnIndex(COL_ERGASIA)));
        return car;
    }

    public boolean save(SQLiteDatabase db) throws IllegalStateException
{
        // Validate Date
        java.util.Date tmpDate;
        try {
            tmpDate = new
SimpleDateFormat(DATE_FORMAT).parse(getDate());
            if (tmpDate == null) {
                throw new IllegalStateException("Η ημερομηνία δεν έχει
σωστή μορφή. Πρέπει να την ορίσετε οπως 30/12/2018");
            }
        } catch (Exception e) {
            throw new IllegalStateException("Η ημερομηνία δεν έχει σωστή
μορφή. Πρέπει να την ορίσετε οπως 30/12/2018");
        }
        // Validate AFM
        if (Customer.list(db, getAfm()).size() < 1) {
            throw new IllegalStateException("Δεν υπάρχει πελάτης με ΑΦΜ
" + getAfm());
        }
```

```
        // Validate ARK
        if (Car.list(db, getArk()).size() < 1) {
            throw new IllegalStateException("Δεν υπάρχει όχημα με Αριθμο
Κυκλοφορίας " + getArk());
        }

        ContentValues contentValues = new ContentValues();
        contentValues.put(COL_AFM, getAfm());
        contentValues.put(COL_ARK, getArk());
        contentValues.put(COL_DATE_DAY, tmpDate.getDay());
        contentValues.put(COL_DATE_MONTH, tmpDate.getMonth());
        contentValues.put(COL_DATE_YEAR, tmpDate.getYear());
        contentValues.put(COL_ERGASIA, getErgasia());

        long result = db.insert(TABLE_NAME, null, contentValues);

        //if date as inserted incorrectly it will return -1
        if (result == -1) {
            return false;
        } else {
            return true;
        }
    }

    public static List<Service> list(SQLiteDatabase db, String ak) {
        final List<Service> toReturn = new ArrayList<>();
        if (ak == null || ak.length() == 0) {
            ak = null;
        } else {
            ak = ak.trim();
        }

        try (Cursor cursor = db.rawQuery("SELECT * FROM " + TABLE_NAME +
(ak == null ? "" : " WHERE " + COL_ARK + " like ?"), (ak == null ? null
: new String[]{ak + "%"}))) {
            while (cursor.moveToNext()) {
                Service service = Service.parse(cursor);
                toReturn.add(service);
            }
        }
        return toReturn;
    }

    public static List<String[]> stats(SQLiteDatabase db) {
        final List<String[]> toReturn = new ArrayList<>();

        final String SQL = "SELECT " + COL_DATE_YEAR + ", " +
COL_DATE_MONTH + ", count(*) as count FROM " + TABLE_NAME + " GROUP BY "
+ COL_DATE_YEAR + ", " + COL_DATE_MONTH;
        toReturn.add(new String[]{
                "YEAR", "MONTH", "COUNT"
        });
        try (Cursor cursor = db.rawQuery(SQL, new String[0])) {
            while (cursor.moveToNext()) {
                int year =
cursor.getInt(cursor.getColumnIndex(COL_DATE_YEAR)) + 1900;
                int month =
cursor.getInt(cursor.getColumnIndex(COL_DATE_MONTH)) + 1;
                int count =
cursor.getInt(cursor.getColumnIndex("count"));
                toReturn.add(new String[]{String.valueOf(year),
```

```
String.valueOf(month), String.valueOf(count)});
            }
        }
        return toReturn;
    }
}
```

ServiceListAdapter

```
package app.fix_car1;

import android.annotation.SuppressLint;
import android.content.Context;
import android.support.annotation.NonNull;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;

import java.util.List;

public class ServiceListAdapter extends ArrayAdapter<Service> {

    private LayoutInflater mInflater;
    private List<Service> services;
    private int mViewResourceId;

    ServiceListAdapter(Context context, List<Service> services) {
        super(context, R.layout.service_list_adapter_view, services);
        this.services = services;
        mInflater = (LayoutInflater)
context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        mViewResourceId = R.layout.service_list_adapter_view;
    }

    @Override
    public int getCount() {
        return services == null ? 1 : services.size() + 1;
    }

    @SuppressLint("ViewHolder")
    @NonNull
    public View getView(int position, View convertView, @NonNull
ViewGroup parent) {
        convertView = mInflater.inflate(mViewResourceId, null);
        TextView tDate = convertView.findViewById(R.id.text_date);
        TextView tAfm = convertView.findViewById(R.id.text_afm);
        TextView tArk = convertView.findViewById(R.id.text_ark);
        TextView tErgasia = convertView.findViewById(R.id.text_ergasia);
        Service service = position == 0 ? null : services.get(position -
1);
        if (service == null) {
            tDate.setText("ΗΜ/ΝΙΑ");
            tAfm.setText("ΑΦΜ");
            tArk.setText("ΑΡΚ");
            tErgasia.setText("ΕΡΓΑΣΙΑ");
        } else {
            tDate.setText(service.getDate());
            tAfm.setText(service.getAfm());
            tArk.setText(service.getArk());
            tErgasia.setText(service.getErgasia());
```

```
        }
        return convertView;
    }
}
```

ListServiceContents

```
package app.fix_car1;

import android.app.Activity;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.widget.ListView;
import android.widget.Toast;

import java.util.List;

public class ListServiceContents extends Activity {
    ListView listView;
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.viewcontents_layout);
        final String ak = getIntent().getStringExtra("q");

        final List<Service> serviceList =
DatabaseHelper.getInstance(this).getServices(ak);
        if (serviceList == null || serviceList.size() == 0)
            Toast.makeText(ListServiceContents.this, "The Database is
empty  :(.", Toast.LENGTH_LONG).show();
        else {
            ServiceListAdapter adapter = new ServiceListAdapter(this,
serviceList);
            listView = findViewById(R.id.listView);
            listView.setAdapter(adapter);
        }
    }
}
```

ΠΑΡΑΡΤΗΜΑ Β: Κώδικες του Γραφικού Περιβάλλοντος.

LoginActivity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="LoginActivity">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true">

        <EditText
            android:id="@+id/login"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
```

```
            android:layout_weight="1"
            android:ems="10"
            android:gravity="bottom"
            android:hint="username"
            android:inputType="textPersonName"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            tools:ignore="HardcodedText" />

        <EditText
            android:id="@+id/password"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_below="@+id/login"
            android:layout_gravity="center_horizontal"
            android:layout_marginTop="4dp"
            android:layout_weight="1"
            android:ems="10"
            android:gravity="bottom"
            android:hint="Password"
            android:inputType="textPassword"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            tools:ignore="HardcodedText" />

        <Button
            android:id="@+id/button"
            style="@android:style/Widget.Button.Inset"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@+id/password"
            android:layout_centerHorizontal="true"
            android:layout_marginTop="4dp"
            android:text="Login"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            tools:ignore="HardcodedText,RtlHardcoded" />

    </RelativeLayout>
</RelativeLayout>
```

MainActivity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".MainActivity">

    <ImageView
        android:id="@+id/imageView"
```

```
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:adjustViewBounds="true"
        android:scaleType="fitCenter"
        android:src="@drawable/splash_car"
        tools:ignore="ContentDescription" />

    <Button
        android:id="@+id/button_enter"
        style="@android:style/Widget.Button.Inset"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:text="Enter"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
        tools:ignore="HardcodedText" />
</LinearLayout>
```

MainMenuActivity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="MainMenuActivity">
    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="0dp"
        android:layout_weight="6"
        android:orientation="vertical"
        tools:ignore="UselessParent">
        <Button
            android:id="@+id/button_new_entry"
            style="@android:style/Widget.Button.Inset"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_weight="1"
            android:text="New"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            tools:ignore="HardcodedText,NestedWeights" />
        <Button
            android:id="@+id/button_search"
            style="@android:style/Widget.Button.Inset"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_weight="1"
            android:text="Search"
```

```
android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            tools:ignore="HardcodedText" />
        <Button
            android:id="@+id/button_statistics"
            style="@android:style/Widget.Button.Inset"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_weight="1"
            android:text="Statistics"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            tools:ignore="HardcodedText" />

        <Button
            android:id="@+id/button_options"
            style="@android:style/Widget.Button.Inset"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_weight="1"
            android:text="Options"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            tools:ignore="HardcodedText" />

        <Button
            android:id="@+id/button_exit"
            style="@android:style/Widget.Button.Inset"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:layout_weight="1"
            android:text="Exit"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            tools:ignore="HardcodedText" />

    </LinearLayout>
</LinearLayout>
```

NewEntry.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="NewEntry">

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="0dp"
        android:layout_weight="2"
```

```
        android:orientation="vertical"
        tools:ignore="UselessLeaf" />

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="0dp"
        android:layout_weight="3.5"
        android:orientation="vertical">

        <Button
            android:id="@+id/button_customer"
            style="@android:style/Widget.Button.Inset"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:layout_weight="1"
            android:text="New Customer"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            tools:ignore="HardcodedText,NestedWeights" />

        <Button
            android:id="@+id/button_car"
            style="@android:style/Widget.Button.Inset"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:layout_weight="1"
            android:text="New Car"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            tools:ignore="HardcodedText,NestedWeights" />

        <Button
            android:id="@+id/button_manage"
            style="@android:style/Widget.Button.Inset"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:layout_weight="1"
            android:text="Manage"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            tools:ignore="HardcodedText,NestedWeights" />

        <Button
            android:id="@+id/button_cancel"
            style="@android:style/Widget.Button.Inset"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:layout_weight="1"
            android:text="Cancel"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            tools:ignore="HardcodedText" />

    </LinearLayout>

    <LinearLayout
        android:layout_width="fill_parent"
```

```
        android:layout_height="0dp"
        android:layout_weight="2"
        android:orientation="vertical"
        tools:ignore="UselessLeaf" />

</LinearLayout>
```

NewEntryCar.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".NewEntryCar">

    <EditText
        android:id="@+id/etAfm"
        android:layout_width="305dp"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="30dp"
        android:hint="ΑΦΜ"
        android:inputType="number"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
        tools:ignore="HardcodedText" />


    <EditText
        android:id="@+id/etArk"
        style="@android:style/Widget.DeviceDefault.EditText"
        android:layout_width="304dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/etAfm"
        android:layout_centerHorizontal="true"
        android:layout_gravity="center_horizontal"
        android:hint="ΑΡ_ΚΥΚΛΟΦΟΡΙΑΣ"
        android:inputType="text"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
        tools:ignore="HardcodedText" />

    <EditText
        android:id="@+id/etInsurance"
        style="@android:style/Widget.DeviceDefault.EditText"
        android:layout_width="305dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/etArk"
        android:layout_centerHorizontal="true"
        android:layout_gravity="center_horizontal"
        android:hint="ΑΣΦΑΛΙΣΤΙΚΗ"
        android:inputType="text"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
        tools:ignore="HardcodedText" />
```

```
    <EditText
        android:id="@+id/etBrand"
        style="@android:style/Widget.DeviceDefault.EditText"
        android:layout_width="304dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/etInsurance"
        android:layout_centerHorizontal="true"
        android:layout_gravity="center_horizontal"
        android:hint="ΜΑΡΚΑ"
        android:inputType="textCapWords"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
        tools:ignore="HardcodedText" />

    <EditText
        android:id="@+id/etModel"
        style="@android:style/Widget.DeviceDefault.EditText"
        android:layout_width="307dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/etBrand"
        android:layout_centerHorizontal="true"
        android:layout_gravity="center_horizontal"
        android:hint="ΜΟΝΤΕΛΟ"
        android:inputType="text"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
        tools:ignore="HardcodedText" />

    <EditText
        android:id="@+id/etColor"
        android:layout_width="305dp"
        android:layout_height="wrap_content"
        android:layout_below="@+id/etModel"
        android:layout_centerHorizontal="true"
        android:layout_gravity="center_horizontal"
        android:hint="ΧΡΩΜΑ"
        android:inputType="text"

android:textAppearance="@style/TextAppearance.AppCompat.Display1"
        tools:ignore="HardcodedText" />

    <Button
        android:id="@+id/btnAdd"
        style="@android:style/Widget.Button.Inset"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:text="Add"

android:textAppearance="@style/TextAppearance.AppCompat.Display2"
        tools:ignore="HardcodedText,RtlCompat,RtlHardcoded" />

    <Button
        android:id="@+id/btnView"
        style="@android:style/Widget.Button.Inset"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignEnd="@+id/etModel"
        android:layout_alignTop="@+id/btnAdd"
```

```
        android:text="View"

android:textAppearance="@style/TextAppearance.AppCompat.Display2"
        tools:ignore="HardcodedText" />

</RelativeLayout>
```

NewEntryCustomer.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".NewEntryCustomer">

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:orientation="vertical"
        tools:ignore="InefficientWeight">

        <EditText
            android:id="@+id/text_afm"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:layout_weight="1"
            android:ems="10"
            android:gravity="left|bottom"
            android:hint="ΑΦΜ"
            android:inputType="number"
            tools:ignore="HardcodedText,NestedWeights,RtlHardcoded" />

        <EditText
            android:id="@+id/text_name"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:layout_weight="1"
            android:ems="10"
            android:gravity="left|bottom"
            android:hint="ONOMA"
            android:inputType="textPersonName"
            tools:ignore="HardcodedText,RtlHardcoded" />

        <EditText
            android:id="@+id/text_lastname"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:layout_weight="1"
            android:ems="10"
            android:gravity="left|bottom"
            android:hint="ΕΠΩΝΥΜΟ"
            android:inputType="textPersonName"
            tools:ignore="HardcodedText,RtlHardcoded" />

        <EditText
            android:id="@+id/text_address"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:ems="10"
        android:gravity="left|bottom"
        android:hint="ΔΙΕΥΘΥΝΣΗ"
        android:inputType="textPostalAddress" />

    <EditText
        android:id="@+id/text_phone"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:ems="10"
        android:gravity="left|bottom"
        android:hint="ΤΗΛΕΦΩΝΟ"
        android:inputType="phone" />

    <EditText
        android:id="@+id/text_mobile"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:ems="10"
        android:gravity="left|bottom"
        android:hint="ΚΙΝΗΤΟ"
        android:inputType="phone" />

    <EditText
        android:id="@+id/text_fax"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:ems="10"
        android:gravity="left|bottom"
        android:hint="FAX"
        android:inputType="phone" />

    <EditText
        android:id="@+id/text_email"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:ems="10"
        android:gravity="left|bottom"
        android:hint="Email"
        android:inputType="textEmailAddress" />

    <EditText
        android:id="@+id/text_doy"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:ems="10"
        android:gravity="left|bottom"
```

```
            android:hint="Δ.Ο.Υ"
            android:inputType="text" />

    </LinearLayout>

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal">

        <Button
            android:id="@+id/btnAdd"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_margin="4dp"
            android:layout_weight="1"
            android:text="Add"
            tools:ignore="HardcodedText" />

        <Button
            android:id="@+id/btnView"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_margin="4dp"
            android:layout_weight="1"
            android:text="View"
            tools:ignore="HardcodedText" />
    </LinearLayout>

</LinearLayout>
```

NewEntryService.xml

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".NewEntryService">

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:orientation="vertical">

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="4dp"
            android:text="Ημερομηνία:"
            tools:ignore="HardcodedText" />

        <EditText
            android:id="@+id/text_date"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:ems="10"
```

```
        android:editable="false"
        android:gravity="left|bottom"
        android:hint="Ημερομηνία"
        android:inputType="date"
        tools:ignore="Deprecated,HardcodedText,RtlHardcoded" />

    <View
        android:layout_width="match_parent"
        android:layout_height="1dp"
        android:layout_marginBottom="4dp"
        android:layout_marginTop="4dp"
        android:background="@android:color/black" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="ΑΦΜ:"
        tools:ignore="HardcodedText" />

    <EditText
        android:id="@+id/text_afm"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:ems="10"
        android:gravity="left|bottom"
        android:hint="ΑΦΜ"
        android:inputType="number"
        tools:ignore="HardcodedText,RtlHardcoded" />

    <View
        android:layout_width="match_parent"
        android:layout_height="1dp"
        android:layout_marginBottom="4dp"
        android:layout_marginTop="4dp"
        android:background="@android:color/black" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="ΑΡΚ:"
        tools:ignore="HardcodedText" />

    <EditText
        android:id="@+id/text_ark"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:ems="10"
        android:gravity="left|bottom"
        android:hint="ΑΡΚ"
        android:inputType="text"
        tools:ignore="HardcodedText,RtlHardcoded" />

    <View
        android:layout_width="match_parent"
        android:layout_height="1dp"
        android:layout_marginBottom="4dp"
        android:layout_marginTop="4dp"
        android:background="@android:color/black" />
```

```
        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="ΕΡΓΑΣΙΑ:"
            tools:ignore="HardcodedText" />

        <Switch
            android:id="@+id/is_local_paint"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_margin="8dp"

android:switchTextAppearance="@style/TextAppearance.AppCompat"
            android:text="Βαφή"
            android:textOff="Ολική"
            android:textOn="Τοπική"
            tools:ignore="HardcodedText" />

        <View
            android:layout_width="match_parent"
            android:layout_height="1dp"
            android:layout_marginBottom="4dp"
            android:layout_marginTop="4dp"
            android:background="@android:color/black" />
    </LinearLayout>

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:orientation="horizontal">

        <Button
            android:id="@+id/btnAdd"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_margin="4dp"
            android:layout_weight="1"
            android:text="Add"
            tools:ignore="ButtonStyle,HardcodedText" />

        <Button
            android:id="@+id/btnView"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_margin="4dp"
            android:layout_weight="1"
            android:text="View"
            tools:ignore="ButtonStyle,HardcodedText" />
    </LinearLayout>

</RelativeLayout>
```

SearchMenu.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

```
android:orientation="vertical"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="SearchMenu">

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_weight="3"
    android:orientation="vertical"
    tools:ignore="UselessLeaf"> </LinearLayout>

<LinearLayout
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_weight="1"
    android:orientation="vertical">

    <RadioGroup
        android:id="@+id/search_type"
        android:layout_width="fill_parent"
        android:layout_height="0dp"
        android:layout_weight="2"
        tools:ignore="NestedWeights">

        <RadioButton
            android:id="@+id/radioButton_afm"
            android:layout_width="wrap_content"
            android:layout_height="0dp"
            android:layout_weight="1"
            android:text="ΑΦΜ"
            tools:ignore="HardcodedText" />

        <RadioButton
            android:id="@+id/radioButton_ak"
            android:layout_width="wrap_content"
            android:layout_height="0dp"
            android:layout_weight="1"
            android:text="ΑΡ. ΚΥΚΛΟΦΟΡΙΑΣ"
            tools:ignore="HardcodedText" />

    </RadioGroup>

    <EditText
        android:id="@+id/keyword"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="2"
        android:gravity="center|bottom"
        android:hint="ΑΦΜ/ΑΡ.ΚΥΚΛΟΦΟΡΙΑΣ"
        tools:ignore="HardcodedText,TextFields" />

    <LinearLayout
        android:gravity="center_horizontal"
        android:layout_gravity="center_horizontal"
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:layout_weight="4"
        android:orientation="horizontal">
```

```
            <Button
                android:id="@+id/button_search"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_gravity="center_vertical"
                android:text="Search"
                tools:ignore="ButtonStyle,HardcodedText" />

            <Button
                android:id="@+id/button_cancel"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_gravity="center_vertical"
                android:text="Cancel"
                tools:ignore="HardcodedText" />
        </LinearLayout>
    </LinearLayout>
</LinearLayout>
```

car _list_adapter_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/text_Afm"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold"
        android:layout_width="match_parent" />

    <TextView
        android:id="@+id/text_Ark"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_Insurance"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_Brand"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
```

```
        android:id="@+id/text_Model"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_Color"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

</LinearLayout>
```

customer _list_adapter_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/text_afm"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_firstName"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_lastName"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_address"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_telephone"
        android:layout_width="match_parent"
```

```
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_mobile"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_fax"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_email"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_doy"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />
</LinearLayout>
```

servise_list_adapter_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/text_date"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_afm"
        android:layout_width="match_parent"
```

```
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_ark"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_ergasia"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textSize="20sp"
        android:textStyle="bold" />
</LinearLayout>
```

stats _list_adapter_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">
    <TextView
        android:id="@+id/text_year"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textAlignment="center"
        android:textSize="20sp"
        android:textStyle="bold" />
    <TextView
        android:id="@+id/text_month"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textAlignment="center"
        android:textSize="20sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/text_count"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="33"
        android:textAlignment="center"
        android:textSize="20sp"
        android:textStyle="bold" />

</LinearLayout>
```

Viewcontents_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent"
 >
    <ListView
        android:id="@+id/listView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />
</LinearLayout>
```

activity_option_menu.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
    android:layout_height="match_parent"
android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".OptionMenu"
    android:baselineAligned="false">

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="textPersonName"
        android:ems="10"
        android:id="@+id/login"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:hint="Login"
        android:elegantTextHeight="true"
        android:gravity="bottom"
        tools:ignore="HardcodedText,UnusedAttribute" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="textPassword"
        android:ems="10"
        android:id="@+id/password"
        android:hint="Password"
        android:layout_weight="1"
        android:gravity="bottom"
        tools:ignore="HardcodedText" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="phone"
        android:hint="IP"
        android:ems="10"
        android:id="@+id/ip"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:gravity="bottom"
        tools:ignore="HardcodedText" />
```

```
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="3">

        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Save"
            android:id="@+id/button"
            android:layout_gravity="center"
            android:layout_centerVertical="true"
            android:layout_centerHorizontal="true"
            tools:ignore="HardcodedText" />

    </RelativeLayout>
</LinearLayout>
```