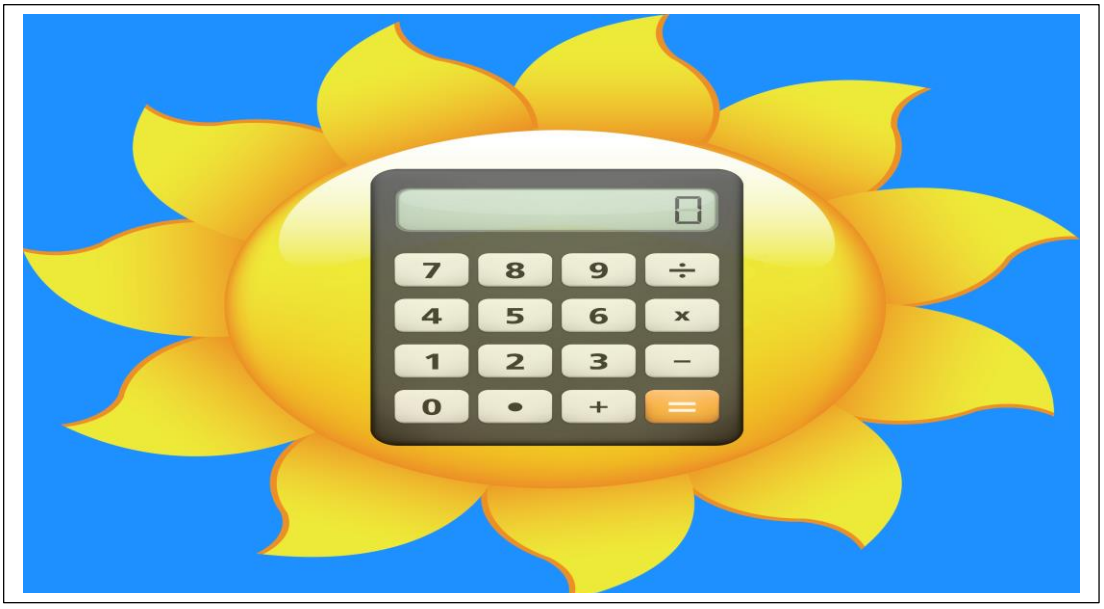
Τεχνολογικό Εκπαιδευτικό Ίδρυμα (ΤΕΙ) Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Μηχανολόγων Μηχανικών

Πτυχιακή Εργασία

Ανάπτυξη Εφαρμογής Android για την Υλοποίηση Μελέτης Αυτόνομου Φωτοβολταϊκού Συστήματος



Του

Αγριόγιαννου Βασίλη

Αρ. Μητρώου: 5424

Επιβλέπων Καθηγητής

κ. Βασιλάκης Κωνσταντίνος

Ηράκλειο 2018

Πρόλογος

Αντικείμενο της πτυχιακής εργασίας είναι η μελέτη και η κατασκευή μιας Android εφαρμογής για την εκπόνηση υπολογισμών που αφορούν σε αυτόνομα φωτοβολταϊκά συστήματα. Βασικός σκοπός αυτής της ανάπτυξης είναι η εφαρμογή που θα προκύψει να ανταποκρίνεται στις ανάγκες κάθε χρήστη που επιθυμεί τη μελέτη ενός αυτόνομου φωτοβολταϊκού συστήματος, προτείνοντας του τα κατάλληλα εξαρτήματα τα οποία θα πληρούν συγκεκριμένα τεχνικά χαρακτηριστικά, διασφαλίζοντας με αυτό τον τρόπο τη σωστή λειτουργία του συστήματος.

Κύριοι στόχοι της συγκεκριμένης πτυχιακής εργασίας είναι μετά την ολοκλήρωση της να παραδοθεί μια πλήρως λειτουργική εφαρμογή σε φορητές συσκευές Android. Βασικά συστατικά της εφαρμογής είναι το λογισμικό κάνει τους απαιτούμενους υπολογισμούς για την εκπόνηση της μελέτης του αυτόνομου φωτοβολταϊκού συστήματος, μια βάση δεδομένων για την αποθήκευση των υπολογισμένων συστημάτων για μελλοντική επισκόπηση καθώς και από ένα μηχανισμό παροχής βοήθειας προς στο χρήστη. Για την υλοποίηση του υπολογισμού της αυτόνομης φωτοβολταϊκής μελέτης χρειάστηκε η ανάπτυξη ενός αλγόριθμου έχοντας ως κύριο αντικείμενο τον υπολογισμό των τεχνικών χαρακτηριστικών του κάθε εξαρτήματος για την σωστή λειτουργία του συστήματος, σε συνδυασμό με προτάσεις για εξαρτήματα και αντίστοιχες ποσότητες, ώστε να πληρούνται τα τεχνικά χαρακτηριστικά της μελέτης.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω την οικογένεια μου, για την βοήθεια και την στήριξη που μου έδειξαν σε όλη την διάρκεια των σπουδών μου. Επίσης τον άνθρωπο που ήταν δίπλα μου από την πρώτη στιγμή στηρίζοντας με σε κάθε δυσκολία.

Επίσης, θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου κ. Βασιλάκη Κωνσταντίνο για την εμπιστοσύνη που μου έδειξε και την βοήθεια που μου παρείχε για την εκπόνηση της πτυχιακής μου εργασίας, καθώς και τον κ. Μαυρομματάκη Φώτη για τις χρήσιμες συμβουλές του σχετικά με τα φωτοβολταϊκά συστήματα.

Περίληψη

Η παρούσα πτυχιακή εργασία αφορά στην δημιουργία μιας πλήρως λειτουργικής Android εφαρμογής για smartphone συσκευές. Ο κύριος σκοπός της είναι η εκπόνηση μιας αυτόνομης φωτοβολταϊκής μελέτης καθώς και προτάσεις για κατάλληλες συσκευές μαζί με τα αντίστοιχα τεχνικά χαρακτηριστικά που θα πρέπει να πληροί η κάθε συσκευή. Τα στοιχεία που απαιτούνται εισάγονται από το χρήστη και για κάθε πεδίο που πρέπει να συμπληρωθεί υπάρχει αναλυτική επεξηγήσεις και βοήθεια. Με την δημιουργία μιας βάσης δεδομένων που θα υπάρχει και θα δουλεύει τοπικά στην συσκευή, θα δίνεται η επιλογή αποθήκευσης της κάθε υπολογισμένης μελέτης για μελλοντική προεπισκόπηση.

Στα πλαίσια της εν λόγω πτυχιακής εργασίας, αναπτύχθηκε μια εφαρμογή που λειτουργεί σε περιβάλλον Android (native Android) χρησιμοποιώντας εργαλεία ανοικτού κώδικα του Android, με την βοήθεια του Android documentation και γράφοντας σε κώδικα XML και Java. Στο κείμενο της εργασίας αναλύεται τόσο ο κώδικας της εφαρμογής που έχει αναπτυχθεί, όσο και οι επιμέρους λειτουργίες που εκτελεί.

Abstract

This diploma thesis aims to create a fully functional Android app for smartphones. Its main purpose is to undertake the calculation of an autonomous photovoltaic system by taking into account the appropriate devices with the technical characteristics that each device must meet. By creating a database that will work locally on the device, to save each calculated system the option for future preview is given. Additionally, detailed explanation and assistance on field completion is provided.

We will analyze both the application code we have developed and the individual functions it performs. To accomplish this work, a fully functional Native Android application was developed, using Android open source tools, provisioning, Android documentation support, and writing XML and Java code for the layout and the application graphics.

Περιεχόμενα

1 Εισαγωγή

Η πτυχιακή εργασία αποσκοπεί στην δημιουργία μιας πλήρως λειτουργικής Android εφαρμογής για smartphone συσκευές. Ο κύριος σκοπός της είναι ο υπολογισμός μιας αυτόνομης φωτοβολταϊκής μελέτης, προτείνοντας τις κατάλληλες συσκευές μαζί με τα τεχνικά χαρακτηριστικά που πρέπει να πληροί η κάθε συσκευή. Με την δημιουργία μιας βάσης δεδομένων που θα υπάρχει στην συσκευή θα δίνεται η επιλογή αποθήκευσης της κάθε υπολογισμένης μελέτης για μελλοντική προεπισκόπηση. Τέλος, θα παρέχεται μια αναλυτική επεξήγηση και βοήθεια του κάθε πεδίου που πρέπει να συμπληρωθεί για τον υπολογισμό της μελέτης.

Στα πλαίσια της εν λόγω πτυχιακής εργασίας, αναπτύχθηκε μια εφαρμογή που λειτουργεί σε περιβάλλον Android (native Android) χρησιμοποιώντας εργαλεία ανοικτού κώδικα του Android, με την βοήθεια του Android documentation και γράφοντας σε κώδικα XML και Java. Στο κείμενο της εργασίας αναλύεται τόσο ο κώδικας της εφαρμογής που έχει αναπτυχθεί, όσο και οι επιμέρους λειτουργίες που εκτελεί.

Τα στάδια που χρειάστηκαν για την ολοκλήρωση της πτυχιακής έχουν ως εξής:

- η αναζήτηση πηγών και εργαλείων στο διαδίκτυο
- ο προσδιορισμός προδιαγραφών
- η συλλογή υλικού
- η επιλογή του κατάλληλου εργαλείου ανάπτυξης
- η αξιολόγηση των στοιχείων της ανάλυσης
- ο προσδιορισμός του τρόπου δομής της εφαρμογής
- ο σχεδιασμός του υπολογισμού μελέτης του φωτοβολταϊκού συστήματος
- η αποθήκευσης των δεδομένων
- η διαγραφή
- ο καθορισμός του τρόπου λειτουργίας και επικοινωνίας με τον χρήστη

Στη πτυχιακή εργασία στο κεφάλαιο 2 θα αναλύσουμε τις απαραίτητες τεχνολογίες τις οποίες θα χρησιμοποιήσουμε για την υλοποίηση της εφαρμογής μας. Επίσης, θα γίνει μια συνοπτική ιστορική αναδρομή σε αυτές. Στη συνέχεια, θα αναλύσουμε τον τρόπο με τον οποίο η εφαρμογή μας θα είναι σε θέση να υπολογίσει την αυτόνομη φωτοβολταϊκή μελέτη και ακολουθεί η βιβλιογραφική ανασκόπηση η οποία εστιάζει στη φωτοβολταϊκή τεχνολογία.

Στο 3ο κεφάλαιο θα αναλυθεί τόσο ο κώδικας της εφαρμογής τον οποίο έχουμε αναπτύξει, όσο και οι επιμέρους λειτουργίες τις οποίες αυτός εκτελεί. Όπως έχει αναφερθεί και σε προηγούμενη ενότητα, για την υλοποίηση της εργασίας σχεδιάστηκε και δημιουργήθηκε μια πλήρως λειτουργική εφαρμογή σε Native Android, που βασίζεται σε εργαλεία ανοικτού κώδικα του Android, με την βοήθεια του Android documentation και εφαρμόζοντας κώδικα XML και Java, για το γραφικό περιβάλλον με τα layout και το λειτουργικό της εφαρμογής.

2 Σχετικά με το Android

Ξεκινώντας την πτυχιακή εργασία στο πρώτο κεφάλαιο θα επικεντρωθούμε στις βασικές έννοιες ανάπτυξης εφαρμογών και στον υπολογισμό φωτοβολταικών συστημάτων. Θα ασχοληθούμε με την ιστορία του λογισμικού κινητών συσκευών, συγκεκριμένα με το Android λογισμικό και τις βασικές εκδόσεις του καθώς και με την υποστηριζόμενη γλώσσα προγραμματισμού για την κατασκευή Android εφαρμογής και τις μεθόδους υπολογισμού αυτόνομου φωτοβολταικού συστήματος.

2.1 Ιστορική Αναδρομή Android Inc

Εικόνα 1 Android Inc.

Στο Palo Alto της Καλιφόρνια ιδρύθηκε η εταιρεία Android Inc το 2003, η οποία αποτελούνταν από τέσσερις ιδρυτές τον ο Rich Miner, ο Nick Sears, ο Chris White και ο Andy Rubin. Τη στιγμή που ιδρύθηκε η εταιρία, ο Andy Rubin δημοσίευσε ότι η Android Inc επρόκειτο να αναπτύξει πιο έξυπνες κινητές συσκευές, με κύριο σκοπό την πιο γρήγορη και πιο ακριβή επίγνωσης της τοποθεσίας του χρήστη μέσω GPS και πιο εξειδικευμένες προτιμήσεις για τους ιδιοκτήτες που τρέχουν το λογισμικό αυτό. Προφανώς όμως, τα αρχικά σχέδια τις εταιρίας Android Inc δεν ήταν η δημιουργία ενός λειτουργικού συστήματος για έξυπνες φορητές συσκευές. Το 2013, 10 έτη μετά την ίδρυση τις Android Inc, ο Andy Rubin πρώην κάτοχος της εταιρίας και πρώην συνεργάτης με την Google Inc πήρε μέρος και μίλησε σε ομιλία στο Τόκιο εξηγώντας ότι το Android λειτουργικό προοριζόταν αρχικά να βελτιώσει τα λειτουργικά συστήματα των ψηφιακών φωτογραφικών μηχανών, όπως αναφέρει η PC World.

2.2 **Ιστορική Αναδρομή Google Inc**

Εικόνα 2 Google Inc.

Η επόμενη μεγάλη αλλαγή στο κεφάλαιο της ιστορία του Android λειτουργικού συστήματος έγινε όταν η αρχική εταιρεία Android Inc εξαγοράστηκε από την Google Inc το 2005. Ο Andy Rubin όπως και τα υπόλοιπα ιδρυτικά μέλη παρέμειναν για να συνεχίσουν το έργο όπου είχαν ξεκινήσει με την ανάπτυξη του λειτουργικού συστήματος υπό τους νέους ιδιοκτήτες τους. Την ίδια χρονική στιγμή πάρθηκε η σημαντική απόφαση το Android λειτουργικό σύστημα να βασιστεί σε Linux πυρήνα και αυτό σήμαινε επίσης ότι θα μπορούσε να προσφερθεί δωρεάν σε τρίτους κατασκευαστές κινητών τηλεφώνων. Προβλέποντας την μεγάλη αγοραστική αξία του λειτουργικού συστήματος δημιουργήθηκαν και άλλες υπηρεσίες που θα χρησιμοποιήσουν το λειτουργικό σύστημα, συμπεριλαμβανομένων των εφαρμογών. Μια νέα εποχή ξεκίνησε το 2007 στον τομέα της κινητής τηλεφωνίας. Την παρούσα εποχή, η Google εξακολουθούσε κρυφά να εργάζεται στο Android, όμως τον Νοέμβριο του ίδιου έτους η εταιρεία ξεκίνησε να αποκαλύπτει τα σχέδιά της. Χρησιμοποιούσε το σχηματισμό του αποκαλούμενου Open Handset Alliance, το οποίο περιλάμβανε κατασκευαστές τηλεφώνων όπως HTC την Motorola, κατασκευαστές chip κύριος κινητών συσκευών όπως η Qualcomm και την Texas Instruments και πάροχους κινητής τηλεφωνίας όπως την T-Mobile. Η ανακοίνωση του πρώτου Android smartphone πραγματοποιήθηκε τον Σεπτέμβριο 2008, το T-Mobile G1, γνωστό και ως HTC Dream σε άλλα μέρη του κόσμου. Την ίδια χρονιά τον Οκτώβριο ξεκίνησε η πώληση της συσκευής στις ΗΠΑ. Σε γενικές γραμμές οι κριτικές για την ίδια την συσκευή δεν ήταν τόσο ικανοποιητικές.

Ωστόσο, όμως το λειτουργικό σύστημα Android 1.0 της συσκευής είχε ήδη τα εμπορικά σήματα του επιχειρηματικού σχεδίου της Google για το λειτουργικό σύστημα. Έχει εν-

σωματώσει διάφορα άλλα προϊόντα και υπηρεσίες της εταιρείας, συμπεριλαμβανομένων των Χαρτών Google, του Youtube και ενός προγράμματος περιήγησης HTML (pre-Chrome) που χρησιμοποιούσε τις υπηρεσίες αναζήτησης της Google. Επίσης, ενσωματωμένη ήταν και η πρώτη έκδοση του Android Market, το παρόν κατάστημα εφαρμογών PlayStore, δηλώνοντας ότι θα περιέχει δεκάδες μοναδικές πρώτες εφαρμογές Android. Όλα αυτά τα χαρακτηριστικά ήταν αρκετά πρωτόγονα την εποχή εκείνη, αλλά αυτό ήταν μόνο η αρχή της εξάπλωσης του Android στην αγορά των κινητών συσκευών. Μια σημαντική κίνηση που έκανε ακόμα η Google Inc ήταν η απόφαση να κάνουν το ίδιο το logo Android ρομπότ, ένα έργο ανοιχτού κώδικα. Σχεδόν κάθε άλλη τεράστια εταιρεία θα προστατεύσει ένα τέτοιο λογότυπο από το να επανασχεδιαστεί και να χρησιμοποιηθεί από άλλους. Ωστόσο, το logo Android έχει τροποποιηθεί και χρησιμοποιηθεί από πολλούς ανθρώπους, διότι η Google επιτρέπει τέτοιες αλλαγές βάσει της Άδειας Παραχώρησης Creative Commons 3.0.

2.2.1 **Android Studio**

Εικόνα 3 Android Studio

Με τον όρο Android Studio, εννοούμε το επίσημο ολοκληρωμένο περιβάλλον ανάπτυξης εφαρμογών (integrated Development environment - IDE) σε λειτουργικό σύστημα Android, το οποίο κατασκευάστηκε με την υποστήριξη της JetBrains και υιοθετεί ένα νέο build system, που ονομάζεται Gradle. Το συγκεκριμένο περιβάλλον έχει βασιστεί σε μεγάλο βαθμό στην «οικογένεια» IDEs της και έχει σχεδιαστεί για Android Development. Το παραπάνω πρόγραμμα διατίθεται δωρεάν για Windows, Mac και Linux. Αξίζει να σημειωθεί ότι έχει αντικαταστήσει το Eclipse Android Development Tools (ADT), το οποίο χρησιμοποιούνταν για κατασκευή λογισμικού Android. Η κυκλοφορία αυτού του νέου προγραμματιστικού περιβάλλοντος ανακοινώθηκε από την Google στις 16 Μαΐου 2013 και η πρώτη σταθερή έκδοσή του ήταν διαθέσιμη για το κοινό το Δεκέμβριο του 2014.

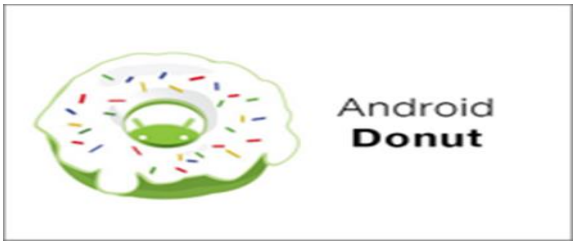
Το σύστημα Gradle, το οποίο είναι ένα αυτοματοποιημένο σύστημα build ανοιχτού κώδικα και βασίζεται στα Apache Ant και Apache Maven, έκανε αρκετά εύκολη όχι μόνο την ανάπτυξη των εφαρμογών με το νέο πρόγραμμα αλλά και την εισαγωγή υπαρχόντων προγραμμάτων μετά από προσαρμογή. Το νέο αυτό σύστημα βασίζεται στην Groovy ως Domain-Specific Language (DSL), ενώ η ανάπτυξη του μπορεί να γίνει με γλώσσες Java και Scala. Οποιαδήποτε αλλαγή πραγματοποιηθεί τόσο στο Project όσο και στα dependencies μέσω του IDE αποθηκεύεται ανάλογα και στο αρχείο build.gradle (GitHub). Τέλος, για τις ανάγκες της εργασίας, χρησιμοποιήθηκε το virtual emulator για διαθέσιμες συσκευές Android και η υλοποίηση έγινε σε επίπεδο API 28.

2.2.2 **CupCake έκδοση 1.5**

Εικόνα 4 Android Cupcake

Αυτή η έκδοση λειτουργικού ήταν η πρώτη και επίσημη με δημόσια ονομασία και κυκλοφόρησε τον Απρίλιο του 2009. Προστέθηκαν αρκετά νέα χαρακτηριστικά και βελτιώσεις σε σύγκριση με τις δύο πρώτες δημόσιες εκδόσεις, συμπεριλαμβανομένων των πραγμάτων που θεωρούμε δεδομένες, όπως η δυνατότητα μεταφόρτωσης βίντεο στο YouTube, ένας τρόπος για την εμφάνιση της οθόνης του τηλεφώνου ώστε να περιστρέφεται αυτόματα στις σωστές θέσεις και υποστήριξη για πληκτρολόγια τρίτου μέρους

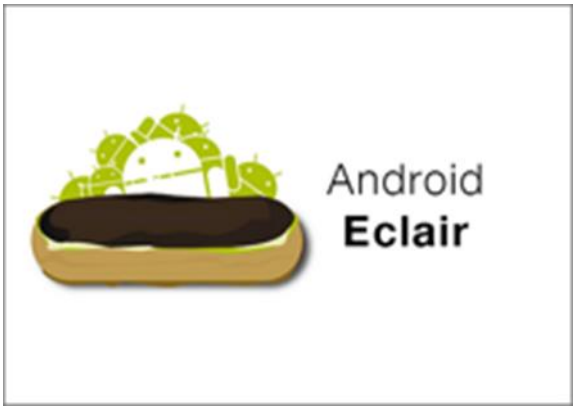
2.2.3 **Donut έκδοση 1.6**



Εικόνα 5 Android Donut

Αμέσως μετά την έκδοση 1.5 το Σεπτέμβριο το ίδιο έτος η Google Inc δρομολόγησε το Android 1.6 με την ονομασία Donut. Προσθέτοντας υποστήριξη για παρόχους κινητής τηλεφωνίας που χρησιμοποιούν δίκτυα που βασίζονται σε CDMA (Code Division Multiple Access), επέτρεψε στα τηλέφωνα Android την πώληση τους από όλους τους παρόχους σε όλο τον κόσμο. Άλλες νέες λειτουργίες εκείνη την εποχή περιλάμβαναν την εισαγωγή του πλαισίου γρήγορης αναζήτησης και την γρήγορη εναλλαγή μεταξύ της κάμερας, της βιντεοκάμερας καθώς και την εφαρμογή «Συλλογή» η οποία προβάλει όλες της φωτογραφίες και βίντεο του χρήστη. Το Donut εισήγαγε επίσης το widget power control για τη διαχείριση wi-fi, bluetooth, GPS, κλπ.

2.2.4 **Eclair έκδοση 2.0**



Εικόνα 6 Android Eclair

Με την έκδοση Eclair προστέθηκε για πρώτη φόρα και η υποστήριξη κειμένου σε ομιλία και επίσης εισήγαγε ζωντανές ταπετσαρίες, πολλαπλή υποστήριξη λογαριασμών και πλοήγηση στο Google Maps, μεταξύ πολλών άλλων νέων λειτουργιών και βελτιώσεων. Το Motorola Droid ήταν το πρώτο τηλέφωνο που συμπεριέλαβε το Android 2.0 εγκαταστημένο. Ήταν επίσης το πρώτο τηλέφωνο με βάση το Android που πωλήθηκε από την Verzion Wireless.

2.2.5 **Froyo έκδοση 2.2**

Εικόνα 7 Android Froyo

Τον Μάιο του 2010 ξεκίνησε επίσημα το Android 2.2 Froyo. Τα smartphone με εγκατεστημένο το Froyo θα μπορούσαν να επωφεληθούν από πολλές νέες λειτουργίες. Μια από αυτές σημαντική λειτουργία είναι το hotspot για κινητά μέσο Wi-Fi, η προώθηση ειδοποιήσεων μέσω της υπηρεσίας Android Cloud to Device Messaging (C2DM), υποστήριξη flash και πολλά άλλα.

2.2.6 **GingerBread έκδοση 2.3**

Εικόνα 8 Android GingerBread

Η έκδοση του λειτουργικού GingerBread κυκλοφόρησε τον Σεπτέμβριο του 2010. Η παρούσα έκδοση είναι αυτή τη στιγμή η παλαιότερη έκδοση του λειτουργικού συστήματος που η Google εξακολουθεί ακόμα να εμφανίζει στη σελίδα ενημερώσεων της μηνιαίας πλατφόρμας. Στο παρόν λειτουργικό σύστημα σημαντικές αλλαγές ανανεώθηκαν στην διεπαφή του χρήστη. Επίσης μια πολύ σημαντική αλλαγή ήταν η υποστήριξη χρήσης λειτουργιών κοντινού πεδίου επικοινωνίας NFC (Near Field Communication) για smartphone που είχαν το απαιτούμενο υλικό. Το πρώτο τηλέφωνο που προστέ-

θηκε τόσο το GingerBread όσο και η λειτουργία NFC ήταν το Nexus S, το οποίο αναπτύχθηκε από την Google με συνεργασία την Samsung. Το GingerBread έθεσε επίσης το υπόβαθρο για τις selfie φωτογραφίες, προσθέτοντας υποστήριξη για πολλαπλές κάμερες και υποστήριξη βίντεο συνομιλίας στο Google Talk.

2.2.7 **HoneyComb έκδοση 3.0**

Εικόνα 9 Android HoneyComb

Αυτή η έκδοση του λειτουργικού συστήματος εισήχθη για πρώτη φορά το Φεβρουάριο του 2011. Η ιδέα ήταν η Honeycomb να προσφέρει συγκεκριμένα χαρακτηριστικά που δεν θα μπορούσαν να αντιμετωπιστούν από τις μικρότερες οθόνες που βρέθηκαν στα smartphone εκείνη περίοδο. Ήταν επίσης μια απάντηση από την Google και τους τρίτους συνεργάτες της στην κυκλοφορία του iPad της Apple το 2010. Παρόλο που η Honeycomb ήταν διαθέσιμη, ορισμένες συσκευές κυκλοφόρησαν ακόμα με τις παλαιότερες εκδόσεις Android 2.x.

Στο τέλος, η Honeycomb κατέληξε να είναι μια έκδοση του Android που δεν ήταν πραγματικά απαραίτητη, καθώς η Google αποφάσισε να ενσωματώσει τα περισσότερα από τα χαρακτηριστικά της στην επόμενη έκδοση 4.0 Ice Cream Sandwich.

2.2.8 **IceCream Sandwich έκδοση 4.0**

Εικόνα 10 Android IceCream Sandwich

Η έκδοση Ice Cream Sandwich του Android κυκλοφόρησε τον Οκτώβριο του 2011, και έφερε πολλά νέα χαρακτηριστικά για τους χρήστες. Ένα από αυτά ήταν η υποστήριξη ξεκλειδώματος οθόνης της συσκευής μέσω της φωτογραφικής μηχανής. Αυτό το είδος επεξεργασίας και έλεγχος των βιομετρικών σημείων σύνδεσης ενός προσώπου έχει εξελιχθεί και βελτιωθεί σημαντικά από τότε. Από τις 6 Ιουλίου της επόμενης χρονιάς, η Google διαπίστωσε ότι το 0,7% όλων των συσκευών Android χρησιμοποιούν αυτήν την περίοδο την συγκεκριμένη έκδοση, το οποίο και είναι οριακά λίγο περισσότερο από την ήδη παλαιότερη έκδοση Gingerbread.

2.2.9 **JellyBean έκδοση 4.1**

Εικόνα 11 Android JellyBean

Η εποχή Jelly Bean του Android ξεκίνησε τον Ιούνιο του 2012 με την κυκλοφορία του Android 4.1. Η Google δημοσίευσε αρκετά γρήγορα τις εκδόσεις 4.2 και 4.3, και οι δύο

υπό την ίδια ονομασία Jelly Bean, τον Οκτώβριο του 2012 και τον Ιούλιο του 2013 αντίστοιχα. Ορισμένες από τις νέες λειτουργίες των εκδόσεων 4.2 και 4.3 περιλάμβαναν νέες λειτουργίες ειδοποιήσεων που πρόσφεραν περισσότερα κουμπιά περιεχομένου ή ενέργειας, καθώς και πλήρη υποστήριξη για την έκδοση Android του προγράμματος περιήγησης Chrome της Google. Το Google Now εμφανίστηκε επίσης ως μέρος της αναζήτησης και το "Project Butter" εισήχθη για να επιταχύνει τα κινούμενα σχέδια και να βελτιώσει την ανταπόκριση των πατημάτων του χρήστη. Οι εξωτερικές οθόνες με την βοήθεια του προγράμματος Miracast άρχισαν να υποστηρίζονται, όπως και η φωτογραφία με τεχνολογία HDR (High Dynamic Range).

2.2.10 KitKat έκδοση 4.4

Εικόνα 12 Android KitKat

Η συγκεκριμένη έκδοση του λειτουργικού συστήματος κυκλοφόρησε τον Σεπτέμβριο του 2010. Η έκδοση KitKat δεν είχε ένα τεράστιο αριθμό νέων δυνατοτήτων, αλλά είχε μια σημαντική βελτίωση που βοήθησε στην επέκταση της συνολικής αγοράς του Android. Ήταν βελτιστοποιημένη για να μπορεί να τρέχει σε smartphones που είχαν μόλις 512 MB μνήμης RAM. Αυτό επέτρεψε στους κατασκευαστές τηλεφώνων να αναβαθμίσουν τις συσκευές τους στην πιο πρόσφατη έκδοση και να το εγκαταστήσουν και σε πολύ φθηνότερα τηλέφωνα.

2.2.11 Lollipop έκδοση 5.0

Εικόνα 13 Android Lollipop

Η Android 5.0 Lollipop έκδοση ξεκίνησε το φθινόπωρο του 2014 και αποτέλεσε σημαντική εξέλιξη στη συνολική εμφάνιση του λειτουργικού συστήματος. Ήταν η πρώτη έκδοση του λειτουργικού συστήματος που χρησιμοποίησε τον νέο τρόπο σχεδίασης της Google, η οποία αξιοποίησέ την χρήση αυτόματης ρύθμισης φωτεινότητας της οθόνης. Το UI (User Interface) έφερε επίσης βασικές αλλαγές, συμπεριλαμβανομένων μιας ανανεωμένης γραμμής πλοήγησης, περισσότερων ειδοποιήσεων για την οθόνης κλειδώματος και πολλά άλλα.

2.2.12 Marshmallow έκδοση 6.0

Εικόνα 14 Android Marshmallow

Κυκλοφόρησε το φθινόπωρο του 2015, το Android 6.0 Marshmallow. Η Google χρησιμοποίησε το "Macadamia Nut Cookie" ως κωδική ονομασία για να περιγράψει την έκδοση Android 6.0 πριν από την επίσημη ανακοίνωση του. Περιλάμβανε χαρακτηριστικά όπως ένα νέο στυλ εμφάνισης την κατακόρυφη κύλιση του μενού, μαζί με το Google Now on Tap, την υποστήριξη ανάγνωσης δακτυλικών αποτυπωμάτων, υποστήριξη στης τεχνολογίας τύπου USB, εισαγωγή του Android Pay και πολλά άλλα.

2.2.13 Nougat έκδοση 7.0

Εικόνα 15 Android Nougat

Η έκδοση 7.0 του λειτουργικού συστήματος Google για κινητά ξεκίνησε το φθινόπωρο του 2016. Η Google πραγματοποίησε επίσης πολλές μεγάλες αλλαγές, όπως τη μετάβαση σε ένα νέο μεταγλωττιστή με την ονομασία JIT (Just In Time) για την επιτάχυνση των εφαρμογών, την υποστήριξη του API (Application programming interface) Vulkan για την ταχύτερη απόδοση 3D γραφικών και την ενεργοποίηση της πλατφόρμας Daydream Virtual Reality.

2.2.14 Oreo έκδοση 8.0

Εικόνα 16 Android Oreo

Τον Μάρτιο του 2017, η Google ανακοίνωσε επίσημα και κυκλοφόρησε την πρώτη προεπισκόπηση για το Android Oreo, επίσης γνωστό ως Android 8.0. Όσον αφορά τις δυνατότητές του, η έκδοση Android Oreo περιείχε πολλές οπτικές αλλαγές, επίσης προσθέσανε νέα API αυτόματης συμπλήρωσης κειμένου, την καλύτερη διαχείριση κωδικών πρόσβασης και πολλά άλλα. Το Android Oreo είναι διαθέσιμο ως λήψη μέσω της πλατφόρμας Android Open Source της Google και είναι επίσης διαθέσιμο ως ενημερωμένη έκδοση για τις παλιότερες (και υποστηριζόμενες) συσκευές Nexus και Pixel της Google.

2.2.15 Pie έκδοση 9.0

Εικόνα 17 Android Pie

Στις 6 Αυγούστου 2018, η εταιρεία δημοσίευσε επίσημα την τελική έκδοση του Android 9.0 και της έδωσε την ονομασία "Pie". Περιλαμβάνει μια σειρά σημαντικών νέων χαρακτηριστικών και αλλαγών. Ένα από τα νέα αυτά χαρακτηριστικά είναι η αφαίρεση των παραδοσιακών κουμπιών πλοήγησης υπέρ ενός επιμήκους κουμπιού στο κάτω μέρος της οθόνης, το οποίο περιέχει το νέο κουμπί επιστροφή στην αρχική σελίδα. Μετακινώντας το νέο κουμπί επιστροφή στην αρχική οθόνη εργασίας προς την άνω μεριά της οθόνης εμφανίζεται η Επισκόπηση, η οποία περιέχει όλες τις πρόσφατα χρησιμοποιημένες εφαρμογές του χρήστη, επιπλέον προβάλετε η γραμμή αναζήτησης.

2.3 **Ιστορική Αναδρομή Oracle Java**

Η Java παράλληλα με άλλες δημοφιλείς γλώσσες όπως η C ++ πλέον χρησιμοποιείται από πολλές online πλατφόρμες όπως την Amazon, την Google, την Netflix, την PayPal και πολλές άλλες. Η Java ξεκίνησε από την Sun Microsystems εταιρία όπου έδρευε στην Silicon Valley, το 1991, σε μια ομάδα με επικεφαλής τον φημισμένο προγραμματιστή James Gosling. Ο Gosling και η ομάδα του βασίστηκαν στη γλώσσα C++, που εξακολουθεί να είναι πολύ δημοφιλής αλλά με σημαντικές αναβαθμίσεις για το μέλλον της.

Η συγκεκριμένη ομάδα παρουσίασε τη νέα της γλώσσα με έναν διαδραστικό, τηλεχειριστήριο για την οικιακή ψυχαγωγία το οποίο είχε αρχικά ως στόχο την βιομηχανία της ψηφιακής καλωδιακής τηλεόρασης. Δυστυχώς, η ιδέα ήταν πολύ προχωρημένη για την ομάδα εκείνη τη στιγμή. Όμως οι μελλοντικές τις ικανότητες για τη χρήση στο διαδίκτυο, το οποίο μόλις άρχισε να απογειώνεται, ήταν πρωτοφανείς. Το 1995, η ομάδα ανακοίνωσε ότι το πρόγραμμα περιήγησης Netscape Navigator Internet θα ενσωμάτωνε την τεχνολογία Java.

Σήμερα, η Java δεν χρησιμοποιείται μόνο στο Διαδίκτυο, αλλά είναι επίσης η αόρατη δύναμη πίσω από πολλές εφαρμογές και συσκευές που τροφοδοτούν την καθημερινή μας ζωή. Από τα κινητά τηλέφωνα μέχρι τις φορητές συσκευές, τα παιχνίδια και τα συστήματα πλοήγησης σε λύσεις ηλεκτρονικού επιχειρείν.

2.4 **Ιστορική Αναδρομή Φ/Β συστημάτων**

Εικόνα 18 History of Photovoltaics

Οι περισσότεροι άνθρωποι εκπλήσσονται από το γεγονός ότι η φωτοβολταϊκή τεχνολογία χρονολογείται πριν από 160 χρόνια. Η βασική επιστήμη ανακαλύφθηκε για πρώτη φορά το 1839, αλλά ο ρυθμός προόδου επιταχύνθηκε πραγματικά με τρεις μεγάλες ωθήσεις τον 20ό αιώνα. Η εμπορική ηλιακή εποχή αρχίζει με την Bell Laboratories, όταν ενώ εργάζονταν σε ημιαγωγούς πυριτίου, ανακάλυψαν ότι το πυρίτιο είχε φωτοηλεκτρικές ιδιότητες και γρήγορα ανέπτυξε ηλιακά κύτταρα Si, επιτυγχάνοντας απόδοση 6%. Οι αρχικοί δορυφόροι ήταν η κύρια χρήση αυτών των πρώτων ηλιακών κυψελών.

Για την ώθησή τους, η Γερμανία και στη συνέχεια η Ιαπωνία ξεκίνησαν σημαντικά προγράμματα επιδοτήσεων και τώρα οι αγορές αυτές υπάρχουν σε μεγάλο βαθμό χωρίς επιδοτήσεις. Το 2007, η Καλιφόρνια οδηγεί τις ΗΠΑ με παρόμοιο 10ετές πρόγραμμα.

2.4.1 **Στήριξη υπολογισμού αυτόνομης Φ/Β Μελέτης**

Για την υλοποίηση μιας αυτόνομης φωτοβολταϊκής μελέτης θα χρειαστεί να υπολογίσουμε την ημερησίως παραγόμενη και καταναλωμένη ηλεκτρική ενέργεια για την πλήρη κάλυψη των συσκευών που θα χρησιμοποιηθούν σε αυτή. Επίσης απαιτείται ο υπολογισμός της απαιτούμενης χωρητικότητας της συστοιχίας συσσωρευτών, για την αποθήκευση και κάλυψη της απαιτούμενης καταναλωμένης ισχύος του συστήματος.

Με την υλοποίηση ενός αλγόριθμου, σε συνεργασία με προϊόντα από τοπικό Ελληνικό προμηθευτή ανανεώσιμων πηγών ενέργειας, καθίσταται δυνατή η πρόταση των απαραίτητων συσκευών για την σωστή λειτουργία της εγκατάστασης.

Στην παρούσα εργασία ο υπολογισμός της ενεργειακής κατανάλωσης θα υπολογιστεί με τη βοήθεια της εξίσωσης του Ενεργειακού Ισοζυγίου : $P_p \cdot PR \cdot \frac{\bar{H}_t}{Gsrc} = m \cdot \left(\frac{E_{La}}{\eta_\alpha} + \frac{E_{L,\varepsilon}}{\eta_\varepsilon} + \left(\frac{n}{N-n}\right) \cdot \frac{E_L}{\eta_\varepsilon}\right)$ (Φραγκιαδάκης, Ι. Ε., 2009), η οποία περιέχει τα εξής χαρακτηριστικά:

- **Ισχύς Αιχμής P_p :** Εκφράζει την μέγιστη ισχύ με την οποία αποδίδει ενέργεια στις πρότυπες συνθήκες (STC, Standart Test Conditions) ο φωτοβολταϊκός συλλέκτης.

- **Πρότυπες Συνθήκες STC :** Ονομάζονται οι συνθήκες εργαστηρίου υπό τις οποίες μετρούνται τα χαρακτηριστικά των φωτοβολταϊκών στοιχείων, οι οποίες είναι Gstc = 1.000 W/m2, θερμοκρασία στοιχείου = 25°C, ηλιακό φάσμα AM = 1,5, κάθετη πρόσπτωση ηλιακής ακτινοβολίας.

- **Πυκνότητα ισχύος σε πρότυπες συνθήκες G_{src} :** Υπό συνθήκες εργαστηρίου, είναι η πυκνότητα ισχύος της ολικής ηλιακής ακτινοβολίας που δέχεται το κάθε τετραγωνικό μέτρο της επιφάνειας του φωτοβολταϊκού στοιχείου.

- **Air Mass AM :** Εκφράζει την απορροφητικότητα της ατμόσφαιρας υπό συνθήκες εργαστηρίου, όταν οι ακτίνες του ήλιου διανύουν απόσταση μέσα στην γήινη ατμόσφαιρα ίση με 1.5 φορές το πάχος της.

- **Λόγος Επίδοσης φωτοβολταϊκού συλλέκτη PR :** Είναι το πηλίκο της ενέργειας που αποδίδει ένας φωτοβολταϊκός συλλέκτης τοποθετημένος σε επιλεγμένη περιοχή προς την ενέργεια που θα απέδιδε αν λειτουργούσε σε πρότυπες συνθήκες STC. Στο πλαίσιο της πτυχιακής εργασίας έχουμε θέσει την τιμή σταθερή και με σχέση τον λόγο επίδοσης των φωτοβολταικών συλλεκτών που διατίθενται στην αγορά θέσαμε τον μέσο όρο της τιμής στο 80%.

- **Μέση ημερήσια ενεργειακή απολαβή $\bar{H}_t$:** Ονομάζεται η ενέργεια που δέχεται το κάθε τετραγωνικό μέτρο επιφάνειας συνολικά κατά τη διάρκεια μίας ημέρας από τον ήλιο.

- **Συντελεστής περιθωρίου φορτίων m :** Ονομάζεται η πρόβλεψη για μελλοντικές επιπλέον έκτακτες ενεργειακές ανάγκες της φωτοβολταϊκής εγκατάστασης.

- **Ημερήσια ηλεκτρική ενέργεια άμεσων φορτίων κατανάλωσης $E_{L,a}$:** Εκφράζει την ενέργεια που χρειάζονται ανά ημέρα οι συσκευές της εγκατάστασης και που μπορούν να τροφοδοτηθούν απευθείας από το φωτοβολταϊκό σύστημα.

- **Συντελεστής μεταφοράς άμεσων φορτίων η_α :** Είναι το ποσοστό της ενέργειας που φτάνει από το φωτοβολταϊκό σύστημα στις συσκευές που τροφοδοτούνται άμεσα. Το υπόλοιπο είναι απώλειες των γραμμών μεταφοράς. Η τιμή του συντελεστή μεταφοράς άμεσων φορτίων στο πλαίσιο της πτυχιακής εργασίας θα είναι σταθερή και το ποσοστό της ενέργειας που θα χάνεται λόγω των απωλειών κατά την μεταφορά θα είναι της τάξης 15%.

- **Ημερήσια ηλεκτρική ενέργεια έμμεσων φορτίων κατανάλωσης $E_{L,\varepsilon}$:** Εκφράζει την ενέργεια που χρειάζονται ανά ημέρα οι συσκευές της εγκατάστασης οι οποίες θα τροφοδοτηθούν μέσω του συσσωρευτή.

- **Συντελεστής μεταφοράς έμμεσων φορτίων η_ε :** Είναι το ποσοστό της ενέργειας που φτάνει από τον συσσωρευτή στις συσκευές που τροφοδοτούνται έμμεσα. Το υπόλοιπο είναι απώλειες του συσσωρευτή. Η τιμή του συντελεστή μεταφοράς άμεσων φορτίων θα είναι σταθερή και το ποσοστό της ενέργειας που θα χάνεται λόγο των απωλειών κατά την μεταφορά θα είναι της τάξης 15%.

- **Συνολική ημερήσια ηλεκτρική ενέργεια φορτίων κατανάλωσης E_L :** Είναι η συνολική ενέργεια που απαιτεί η φωτοβολταϊκή εγκατάσταση ανά ημέρα.

- **Ημέρες του μήνα N :** Είναι το σύνολο των ημερών του μήνα για τον οποίο θα υλοποιήσουμε τον υπολογισμό της φωτοβολταϊκής εγκατάστασης.

- **Διαδοχικές ημέρες αυτονομίας n :** Εκφράζει τις συννεφιασμένες ημέρες που υποθέτουμε για κάθε μήνα.

Επίσης, ο υπολογισμός της χωρητικότητας του συσσωρευτή της φωτοβολταϊκής εγκατάστασης θα υπολογιστεί με τη βοήθεια της εξίσωσης της Χωρητικότητας Συσσωρευτή : $C(n) = \frac{(n+b)\cdot m\cdot E_L}{\eta_{\gamma,\beta}\cdot\eta_{\varepsilon\kappa\varphi}\cdot\beta_{\varepsilon\kappa\varphi}\cdot V_B}$ (Φραγκιαδάκης, Ι. Ε., 2009), για την κάλυψη των διαδοχικών συννεφιασμένων ημερών, η οποία περιέχει τα εξής χαρακτηριστικά:

- **Ποσοστό φορτίων έμμεσης τροφοδοσίας b :** Ονομάζεται το ποσοστό των φορτίων που θα τροφοδοτούνται έμμεσα, μέσω του συσσωρευτή. Η τιμή του

ποσοστού φορτίων έμμεσης τροφοδοσίας στο πλαίσιο της πτυχιακής εργασίας είναι σταθερή και αποτελείται από το 1,0.

- **Συντελεστής περιθωρίου φορτίων m :** Ονομάζεται η πρόβλεψη για μελλοντικές επιπλέον έκτακτες ενεργειακές ανάγκες της φωτοβολταϊκής εγκατάστασης. Ο συντελεστής περιθωρίου φορτίων αποτελείται από σταθερή τιμή την οποία έχουμε ορίσει στο 1,5.

- **Διαδοχικές ημέρες αυτονομίας n :** Εκφράζει τις συννεφιασμένες ημέρες που υποθέτουμε για κάθε μήνα.

- **Συνολική ημερήσια ηλεκτρική ενέργεια φορτίων κατανάλωσης E_L :** Είναι η συνολική ενέργεια που απαιτεί η φωτοβολταϊκή εγκατάσταση ανά ημέρα.

- **Συντελεστής γήρανσης συσσωρευτών $\eta_{\gamma,B}$:** Ονομάζεται το ποσοστό κατά το οποίο μειώνεται η χωρητικότητα του συσσωρευτή με την πάροδο του χρόνου. Στο πλαίσιο της πτυχιακής εργασίας έχουμε θέσει την τιμή σταθερή και με σχέση τον συντελεστή γήρανσης συσσωρευτών που διατίθενται στην αγορά θέσαμε τον μέσο όρο της τιμής στο 80%.

- **Βάθος εκ φόρτισης $\beta_{\varepsilon\kappa\varphi}$:** Εκφράζει το ποσοστό εκ φόρτισης του συσσωρευτή και εξαρτάται από τους κύκλους φόρτισης και εκ φόρτισης που θέλω να αντέξει ο συσσωρευτής. Το βάθος εκ φόρτισης του συσσωρευτή αποτελείται από σταθερή τιμή την οποία έχουμε ορίσει στο 50%, το οποίο είναι ο μέσος όρος του βάθους εκ φόρτισης συσσωρευτών που διατίθενται στην αγορά.

- **Συντελεστής εκ φόρτισης $\eta_{\varepsilon\kappa\varphi}$:** Εκφράζει το πηλίκο της ενέργειας που αποδίδει ο συσσωρευτής κατά την εκ φόρτιση του δια την ενέργεια που απορροφά κατά την φόρτιση. Το υπόλοιπο είναι απώλειες κυρίως θερμότητας που απορροφάτε από τον ηλεκτρολύτη. Ο συντελεστής εκ φόρτισης του συσσωρευτή αποτελείται από την τιμή 80% και είναι σταθερή.

- **Τάση συσσωρευτή V_B :** Ονομάζεται η τάση που απαιτείται για την τροφοδοσία του inverter του συστήματος, εξαρτάται από το μέγεθος της συνολικής ισχύος των συσκευών που θα τροφοδοτούνται.

3 Σχεδιασμός του συστήματος

Στο παρόν κεφάλαιο θα αναλυθεί τόσο ο κώδικας της εφαρμογής τον οποίο έχουμε αναπτύξει, όσο και οι επιμέρους λειτουργίες που εκτελεί. Όπως έχει αναφερθεί και σε προηγούμενη ενότητα, για την ολοκλήρωση εργασίας θα πρέπει ν' αναπτυχθεί μια πλήρως λειτουργική εφαρμογή σε Native Android. Για την υλοποίηση της έγινε η ανάλυση των απαιτήσεων σε συνδυασμό με τις ανάγκες που προκύπτουν, καθώς και να καταγραφούν οι απαραίτητες προδιαγραφές που πρέπει να φέρει η εφαρμογή. Σε δεύτερο στάδιο πραγματοποιήθηκε ο σχεδιασμός της λειτουργικότητας της εφαρμογής και σχεδιάστηκε ο τρόπος απεικόνισης του γραφικού περιβάλλοντος. Στο κείμενο που ακολουθεί επεξηγείται , ο τρόπος της υλοποίησης της εφαρμογής και γίνεται περιγραφή των ελέγχων που πραγματοποιήθηκαν για την ορθή λειτουργία της.

3.1 Θεμελιώδη στοιχεία του εργαλείου ανάπτυξης

Το εργαλείο SDK (Software Development Kit) της Android συνθέτει τον κώδικα της εφαρμογής μαζί με τα αντίστοιχα δεδομένα και τα αρχεία σε ένα πακέτο με την ονομασία APK (Android Application Package). Το APK είναι ένα αρχείο με κατάληξη .apk, περιέχει το σύνολο των περιεχομένων μιας Android εφαρμογής και είναι το βασικό αρχείο για την εγκατάσταση της εφαρμογής σε συσκευές που χρησιμοποιούν λειτουργικό σύστημα Android.

3.1.1 Components

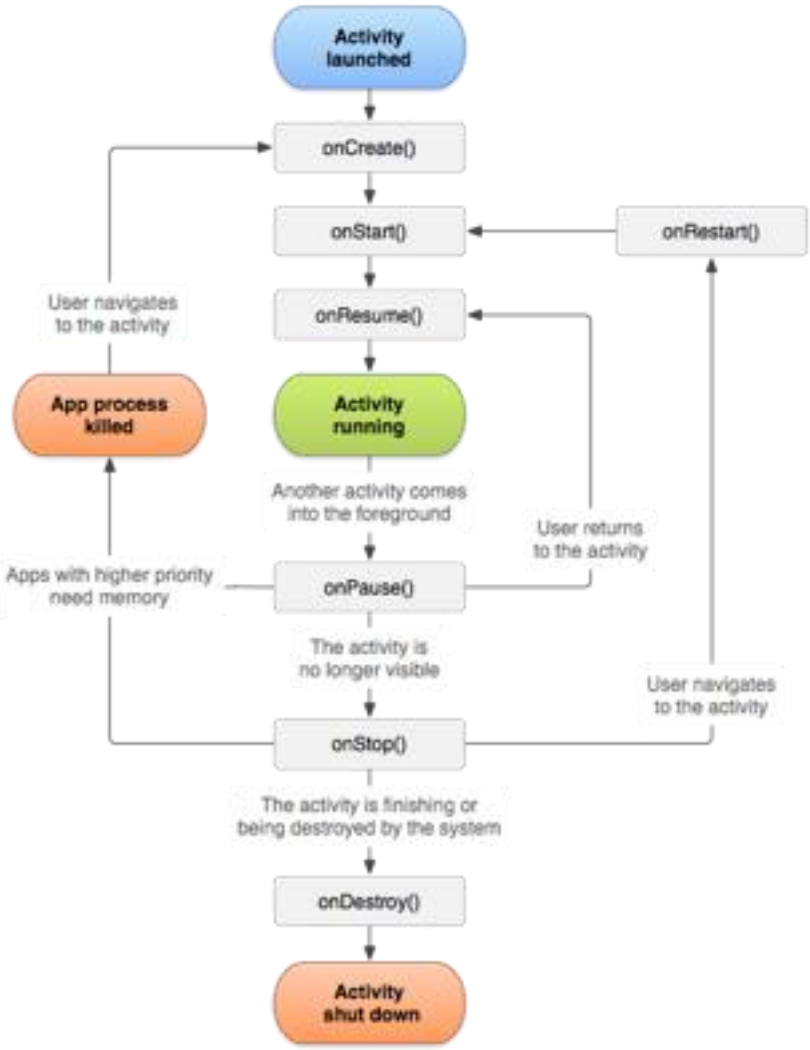
Τα συστατικά-(components) της εφαρμογής, τα οποία είναι τα βασικά δομικά στοιχεία μιας εφαρμογής Android, είναι σημεία εισόδου μέσω των οποίων το σύστημα ή ένας χρήστης μπορεί να εισάγει στην εφαρμογή τις ενέργειες που αυτός επιθυμεί. Κάθε τύπος στοιχείων εξυπηρετεί έναν ξεχωριστό σκοπό και έχει έναν συγκεκριμένο κύκλο ζωής, ο οποίος καθορίζει τον τρόπο με τον οποίο το στοιχείο δημιουργείται και κατα-

στρέφεται. Πολλά από αυτά τα στοιχεία εξαρτώνται από επιπλέον στοιχεία της εφαρμογής. Υπάρχουν τέσσερις διαφορετικοί τύποι στοιχείων από τα οποία μπορεί να αποτελείται μια εφαρμογή:

- Δραστηριότητες (Activities)
- Υπηρεσίες (Services)
- Δέκτες εκπομπής (Broadcast receivers)
- Παροχής περιεχομένου (Content providers)

3.1.2 **Activities**

Η Δραστηριότητα (Activity) είναι ένα ουσιαστικό στοιχείο μιας Android εφαρμογής και ο τρόπος με τον οποίο οι δραστηριότητες ξεκινούν και επικοινωνούν μεταξύ τους είναι ένα βασικό στοιχείο του μοντέλου εφαρμογής του Android λειτουργικού συστήματος. Ενώ στα παραδείγματα προγραμματισμού ξεκινάνε οι εφαρμογές με τη μέθοδο main(), το σύστημα Android εκκινεί τον κώδικα μέσω ενός Activity. Το σύστημα Android για να ενεργοποιήσει το Activity, ξεκινάει με μια κλήση στην μέθοδο onCreate() (callback method) που περιέχει. Υπάρχει μια σειρά μεθόδων callback που θα ξεκινήσουν το Activity και μια σειρά μεθόδων callback που διακόπτουν το Activity, όπως φαίνεται στο παρακάτω διάγραμμα του κύκλου ζωής ενός Activity.



Εικόνα 19 Activity lifecycle

3.1.3 **Services**

Οι υπηρεσίες (Services) αποτελούν ένα από τα βασικά στοιχεία του λειτουργικού συστήματος Android, δεν παρέχουν διεπαφή στον χρήστη και πρόκειται για ένα στοιχείο που εκτελείται στο παρασκήνιο και χρησιμοποιείται για την εκτέλεση εκτεταμένων εργασιών ή και απομακρυσμένων διεργασιών μέσω του διαδικτύου. Ένα από τα στοιχεία (Components) τα οποία περιέχει το Android λειτουργικό σύστημα, όπως η δραστηριότητα (Activity), μπορεί να ξεκινήσει μια υπηρεσία και να την αφήσει να τρέξει στο παρασκήνιο ή ακόμα να αλληλοεπιδράσει με αυτήν.

Οι υπηρεσίες χωρίζονται σε δύο κατηγορίες, οι οποίες αλλεπιδρούν με το λειτουργικό σύστημα με σκοπό την κατάλληλη διαχείριση της εφαρμογής και αυτές είναι:

- Υπηρεσίες εκκίνησης (Started services)
- Δεσμευμένες υπηρεσίες (Bound services)

Οι υπηρεσίες εκκίνησης (Started services) δίνουν εντολή στο σύστημα να τις κρατήσει σε λειτουργία μέχρι να ολοκληρωθεί η εργασία τους και αυτές μπορεί να είναι ο συγχρονισμός ορισμένων δεδομένων στο παρασκήνιο ή η αναπαραγωγή μουσικής ακόμα και μετά την έξοδο από την αντίστοιχη από εφαρμογή.

Οι Δεσμευμένες υπηρεσίες (Bound services) εκτελούνται επειδή κάποια άλλη εφαρμογή (ή και το ίδιο το σύστημα) έχει δηλώσει ότι θέλει να κάνει χρήση της υπηρεσίας και είναι η υπηρεσία που παρέχει ένα API (**Application Programming Interface**) σε μια άλλη διαδικασία. Το σύστημα ξέρει λοιπόν ότι υπάρχει μια εξάρτηση μεταξύ αυτών των διαδικασιών, οπότε αν η πρώτη διαδικασία είναι συνδεδεμένη με μια υπηρεσία στη δεύτερη διαδικασία, γνωρίζει ότι πρέπει να κρατήσει την δεύτερη διαδικασία και την υπηρεσία της ώστε να μπορέσει να τρέξει για την πρώτη διαδικασία. Οι υπηρεσίες έχουν αποδειχθεί ότι είναι ένα πολύ χρήσιμο δομικό στοιχείο, λόγω της ευελιξίας τους, για κάθε είδους έννοιες συστήματος υψηλότερου επιπέδου.

3.1.4 **Intents**

Μια πρόθεση (Intent) είναι ουσιαστικά ένα μήνυμα που μεταδίδεται μεταξύ των στοιχείων (components) του λειτουργικού συστήματος, όπως είναι οι Δραστηριότητες (Activities), οι Υπηρεσίες (Services), οι Δέκτες εκπομπής (Broadcast receivers) και οι Παροχής περιεχομένου (Content providers). Οι προθέσεις είναι ο βασικός τρόπος με τον οποίο μπορούν να επικοινωνήσουν μεταξύ τους οι εγκαταστημένες εφαρμογές που περιέχει ένα Android λειτουργικό σύστημα.

Υπάρχουν δύο τύποι προθέσεων:

- Ρητές προθέσεις (Explicit intents)
- Έμμεσες προθέσεις (Implicit intents)

Οι ρητές προθέσεις (Explicit intents) προσδιορίζουν ποια εφαρμογή είναι εκείνη που θα χρησιμοποιηθεί, παρέχοντας το όνομα του πακέτου της εφαρμογής στόχου ή ένα πλήρες όνομα κλάσης εξειδικευμένου τμήματος. Συνήθως, χρησιμοποιούμε ρητή πρόθεση κατά την εκκίνηση ενός στοιχείου στην εφαρμογή μας, επειδή γνωρίζουμε το όνομα της κλάσης στην οποία ανήκει η δραστηριότητα ή η υπηρεσία που θέλουμε να ξεκινήσουμε. Σε αντίθεση οι έμμεσες προθέσεις δεν εστιάζουν σε ένα συγκεκριμένο στοιχείο, αλλά δηλώνουν γενικά μια ενέργεια που πρέπει να εκτελεστεί.

Οι έμμεσες προθέσεις (Implicit intents) προσδιορίζουν τη δράση που πρέπει να εκτελεστεί και τα δεδομένα που συνιστούν το περιεχόμενο για τη δράση αυτήν. Χρησιμοποιώντας την έμμεση πρόθεση το σύστημα αναζητάει τα στοιχεία εκείνα που έχουν καταχωρηθεί για τη συγκεκριμένη ενέργεια και τον τύπο δεδομένων προσαρμογής τους. Εάν εντοπιστεί μόνο ένα στοιχείο, το Android ξεκινά από αυτό το στοιχείο, όμως εάν εντοπιστούν περισσότερα στοιχεία από το λειτουργικό σύστημα, θα εμφανιστεί στον χρήστη ένας διάλογος επιλογής από τον οποίο μπορεί να αποφασίσει ποιο στοιχείο θα χρησιμοποιηθεί για την πρόθεση αυτή.

Για την δημιουργία μίας πρόθεσης είναι απαραίτητο να του προσθέσουμε πληροφορίες αναλόγως για αυτό που προσδιορίζεται να κάνει:

- Όνομα στοιχείου (Component name).
- Την ενέργεια (Action)
- Πληροφορίες που θα περιέχει (Data)
- Κατηγορία που προσδιορίζεται (Category)
- Επιπρόσθετα (Extras)
- Σημαίες, τα δικαιώματα τα οποία θα έχει (Flags)

3.1.5 **Layout**

Το layout που παρέχει το Android λειτουργικό σύστημα είναι μια κλάση που περιέχει και χειρίζεται την οργάνωση του τρόπου εμφάνισης και την διάταξη των child Views του. Το κυρίως Layout ενός Activity αποτελείται πάντα από ένα Layout το οποίο ονομάζεται γονέας (parent Layout) και αποτελείται από τα στοιχεία γραφικού περιβάλλοντος τα Views, τα widgets και από περισσότερα επιμέρους Layouts.
Το ViewGroup είναι μια από τις υποκατηγορίες που περιέχει το View class και περιέχει ένα σύνολο μη ορατών δομικών πλαισίων που μπορούν να φέρουν μέσα τους περισσότερα από ένα αντικείμενο View ή άλλες ομάδες ViewGroups. Με αυτόν τον τρόπο μπορούμε να ορίσουμε την οπτική δομή των Views για την σωστή προβολή τους στο γραφικό περιβάλλον του χρήστη.
Τα κυρίως Layout ενός Activity είναι :

- **LinearLayout:** Γραμμική στοίχιση των Views είτε σε οριζόντια είτε κάθετη διάταξη.

- **RelativeLayout:** Καθορίζεται η θέση των Views σε σχέση μεταξύ των υπόλοιπων Views.
- **ConstraintLayout:** Το ConstraintLayout είναι ο προκάτοχος του RelativeLayout, τα κυρίως πλεονεκτήματα του σε σχέση με το RelativeLayout είναι η μείωση του αριθμού των ένθετων προβολών (nested Views) και καθώς μπορούμε τώρα να ενώσουμε οποιαδήποτε πλευρά ενός View με οποιαδήποτε πλευρά ενός άλλου View πιο ευκολά.
- **GridLayout:** Πραγματοποιείται η στοίχιση των Views σε πίνακα ο οποίος αποτελείται από τις αντίστοιχες σειρές και στήλες που θα του ορίσουμε.

3.1.6 **Views**

Το κάθε Layout αποτελείται από έναν αριθμό Views, τα οποία προορίζονται για την απεικόνιση του γραφικού περιβάλλοντος του χρήστη. Το αντικείμενο View που παρέχεται από την View κλάση (View class) είναι για αυτόν τον λόγο και το βασικό δομικό στοιχείο για την διεπαφή του χρήστη, αναλόγως τις διαστάσεις που θα του προκαθοριστούν καταλαμβάνει μια περιοχή στην οθόνη και είναι υπεύθυνο για την απεικόνιση τον γραφικών στοιχείων, το χειρισμό συμβάντων και την δημιουργία δια δραστικών στοιχείων στο γραφικό περιβάλλον του χρήστη, όπως τα κουμπιά και τα πεδία κειμένου.

3.2 **Ανάλυση και ο καθορισμός των προδιαγραφών**

Η ανάλυση των λειτουργικών απαιτήσεων ενός λογισμικού όπως η εφαρμογή μας πρέπει να δηλώνουν αυτό που περιμένουμε να κάνει το σύστημα μας. Οι απαιτήσεις λογισμικού κυρίως χωρίζονται σε δυο βασικές κατηγορίες, τις Λειτουργικές Απαιτήσεις (Functional Requirements) και τις Μη-Λειτουργικές Απαιτήσεις (Non-Functional Requirements). Οι Λειτουργικές Απαιτήσεις, περιγράφουν την συμπεριφορά του συστήματος με βάση τα δεδομένα που θα εισαχθούν από τον χρήστη, την επεξεργασία τους και την προβολή των αποτελεσμάτων. Οι Μη-Λειτουργικές Απαιτήσεις περιγράφουν το βαθμό, στον οποίο το σύστημά μας θα είναι σε θέση να υποστηρίξει τις Λειτουργικές απαιτήσεις της μορφής απόδοσης και τις χρηστικότητας που ευθύνονται.

3.2.1 **Ανάλυση λειτουργικών απαιτήσεων**

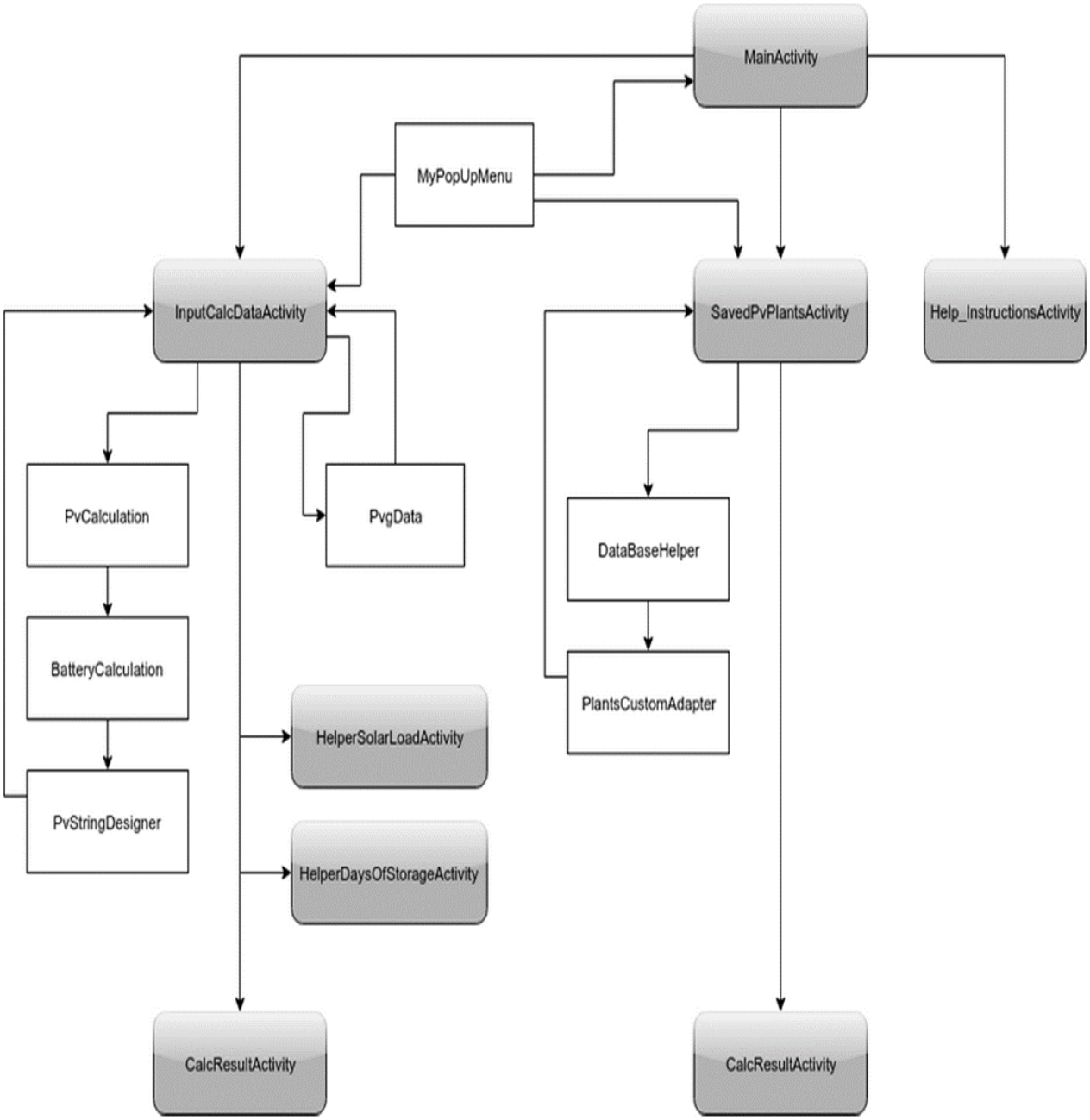
- Το βασικό ζητούμενο της εφαρμογής μας είναι ο υπολογισμός της ζητούμενης από τον χρήστη αυτόνομης φωτοβολταικής μελέτης.
- Η εφαρμογή να είναι σε θέση να μπορεί να προσφέρει την γρήγορη και εύκολη περιήγηση του χρήστη.
- Η εφαρμογή μας να καταφέρει να καλύψει τόσο τις ανάγκες του χρήστη, ο οποίος έχει τις κατάλληλες εξειδικευμένες γνώσεις σχετικά με το αντικείμενο, όσο και του μη εξειδικευμένου.
- Την εύκολη και γρήγορη υλοποίηση της αυτόνομης φωτοβολταικής μελέτης, χωρίς την υποστήριξη επιπλέον εργαλείων.

Την δημιουργία ενός επιπλέον αλγόριθμου ο οποίος θα μπορεί να υπολογίζει και να προτείνει στον χρήστη τις κατάλληλες συσκευές και τον απαιτούμενο αριθμό τους.

3.2.2 **Ανάλυση μη λειτουργικών απαιτήσεων**

- Παροχή χαρτοφυλακίου για την αποθήκευση των είδη υπολογισμένων αυτόνομων φωτοβολταϊκών μελετών.
- Ανάλυση των τεχνολογιών που θα χρησιμοποιηθούν από την εφαρμογή μας για την υλοποίηση της αυτόνομης φωτοβολταϊκής μελέτης.
- Κατάλληλη κατασκευή της τοπικής βάσης δεδομένων για μελλοντική αναβάθμιση και υποβάθμισή της.
- Χρήση δύο γλωσσών, Αγγλικά και Ελληνικά, για την διεπαφή του χρήστη με το σύστημα με σκοπό την ευρεία κάλυψη των αναγκών των χρηστών τόσο σε εθνικό όσο και σε διεθνές επίπεδο.

3.3 **Διάγραμμα διαδρομών Activities**



Εικόνα 20 Διάγραμμα διαδρομών Activities

Στο παραπάνω διάγραμμα απεικονίζουμε τις πιθανές διαδρομές των Activities και τις απαραίτητες κλάσεις με τις οποίες θα συνεργαστούν τα Activities, ώστε να είναι εφικτή η σωστή λειτουργία της εφαρμογής μας.

- Η αρχική Activity που προβάλει η εφαρμογή μας, είναι το MainActivity και περιέχει σε μορφή συνδέσμων τις τρείς κύριες κατηγορίες της εφαρμογής. Επίσης, δημιουργήσαμε ένα μενού, το οποίο θα φέρουν τα απαραίτητα Activities με την ονομασία MyPopUpMenu, που υλοποιήθηκε σε ξεχωριστή κλάση από τον κυρίως κώδικα του κάθε Activity, ώστε να επαναχρησιμοποιηθεί και στα υπόλοιπα επιθυμητά Activities. Για την υλοποίηση μιας αυτόνομης φωτοβολταϊκής μελέτης υλοποιήσαμε την Activity με την ονομασία InputCalcDataActivity, η οποία θα συλλέξει τα απαραίτητα δεδομένα. Για τον υπολογισμό της μελέτης δημιουργήσαμε μια τελική στατική κλάση την PvgData, η οποία φέρει όλες τις απαραίτητες τιμές της ηλιακής ακτινοβολίας που θα είναι διαθέσιμες για τον χρήστη προς επιλογή. Ο υπολογισμός της αυτόνομης φωτοβολταϊκής μελέτης υλοποιήθηκε με δύο κλάσεις την PcCalculation ως βασική κλάση (base class) και την BatteryCalculation ως παράγωγη κλάση (derived class), η οποία κληρονομεί όλες τις ιδιότητες και τις μεθόδους της βασικής κλάσης. Στη συνέχεια δημιουργήσαμε την κλάση PvStringDesigner η οποία είναι υπεύθυνη για την πρόταση των κατάλληλων συσκευών, από τις οποίες πρέπει να αποτελείται η αυτόνομη φωτοβολταϊκή εγκατάσταση. Για την άμεση υποστήριξη του χρήστη περί των δεδομένων που πρέπει να εισάγει προσθέσαμε επιπλέον δύο συνδέσμους στο παρόν Activity, που θα ανακατευθύνουν προσωρινά το χρήστη στο Activity HelperSolarLoadActivity και στο HelperDaysOfStorageActivity. Μετά την ολοκλήρωση των υπολογισμών της μελέτης ανακατευθύνεται ο χρήστης στο Activity CalcResultActivity, το οποίο προβάλει τα απαραίτητα τεχνικά χαρακτηριστικά και τις αντίστοιχες κατάλληλες συσκευές.

- Η δεύτερη βασική κατηγορία είναι η SavedPvPlantsActivity η οποία προβάλει όλες τις αποθηκευμένες μελέτες του χρήστη. Για την προβολή των μελετών υλοποιήσουμε την κλάση DataBaseHelper η οποία περιέχει τα χαρακτηριστικά της βάση δεδομένων και τα δικαιώματα του χρήστη. Επιπλέον, για τη βέλτιστη απόδοση της εξαγωγής των τιμών της κάθε αντίστοιχης μελέτης από την βάση δεδομένων δημιουργήσουμε την κλάση PlantsCustomeAdapter.

- Η τελευταία βασική κατηγορία που περιέχει η εφαρμογή μας είναι η Help_instructionsActivity, η οποία είναι υπεύθυνη για την παροχή βοήθειας και τη σωστή καθοδήγηση του χρήστη για την ολοκλήρωση της αυτόνομης φωτοβολταϊκής μελέτης.

3.4 **Τα resources της εφαρμογής**

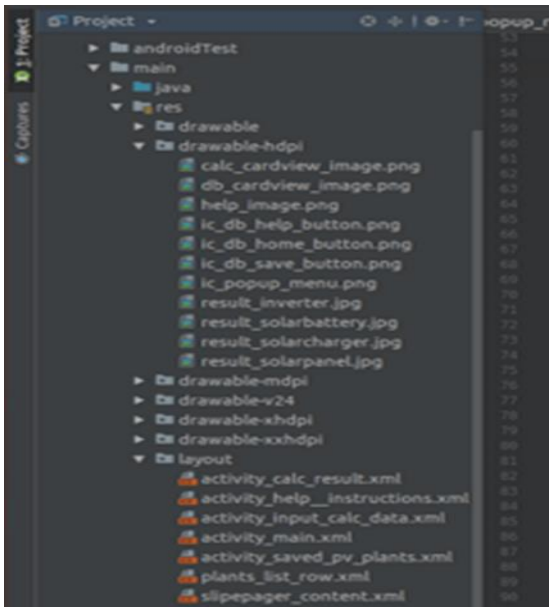
Στη συγκεκριμένη ενότητα θα αναλύσουμε, πως τροποποιήσαμε και δημιουργήσαμε τους απαραίτητους φακέλους, μέσα στα Resources της εφαρμογής μας. Όλες οι εφαρμογές που υποστηρίζονται από το Android λειτουργικό σύστημα αποτελούνται από δύο βασικά αντικείμενα:

- τη λειτουργικότητα της εφαρμογής
- τα Resources που περιέχει

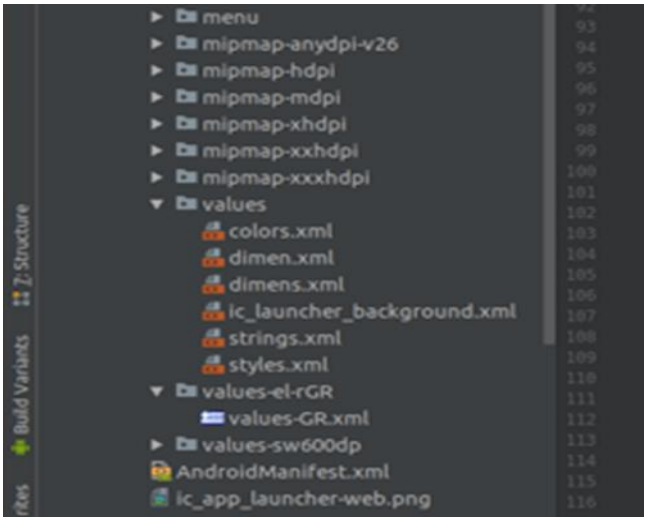
Η λειτουργικότητα της εφαρμογής είναι ο κώδικας που υλοποιήσαμε ο οποίο καθορίζει τον τρόπο λειτουργία της εφαρμογής μας.

Τα Resources της εφαρμογής είναι πρόσθετα αρχεία τα οποία αποτελούνται από στατικό περιεχόμενο. Τα περισσότερα δεδομένα από τα οποία αποτελούνται οι εφαρμογές, ορίζονται σε αυτά τα αρχεία τύπου Resources. Στη συνέχεια έχουμε τη δυνατότητα μέσω αυτών των αρχείων να αντιστοιχήσουμε και να χρησιμοποιήσουμε τα δεδομένα τους στον κώδικα της εφαρμογής μας. Βασικός σκοπός των Resources είναι η κατηγοριοποίηση των δεδομένων για την αυτόματη εύρεσή τους από το λειτουργικό σύστημα, κατά την εκκίνηση ενός Activity. Τα Resources αποτελούνται από εικόνες, αρχεία ήχου, βίντεο, ορίζουν συμβολοσειρές οι οποίες θα προβάλλονται στο γραφικό περιβάλλον του χρήστη, χρώματα και πολλά περισσότερα.

Σκοπός μας είναι η συγκέντρωση των βασικών τιμών στους αντίστοιχους φακέλους που θα χρησιμοποιήσει αυτόματα η εφαρμογή, ώστε να επιτύχουμε στο μέλλον την πιο εποικοδομητική διαχείριση και επέκτασή της.



Εικόνα 22 Android Resources_1



Εικόνα 21 Android Resources_2

3.4.1 Values

Ο φάκελος με την ονομασία Values, χρησιμοποιείται για την αποθήκευση των τιμών. Αυτός περιέχει χαρακτηριστικά χρώματος, στυλ, γραμματοσειρές και διαστάσεις. Οι τιμές αυτές αντιπροσωπεύουν τα Resources της εφαρμογής.

Η εφαρμογή μας θα αποτελείται από δύο κυρίως γλώσσες, στις οποίες θα μπορεί να προβάλλεται το περιεχόμενο του γραφικού περιβάλλοντός της, την Αγγλική και την Ελληνική. Με τη δημιουργία του φακέλου values-el-rGr, ο οποίος περιέχει τον υποφάκελο strings προστέθηκε και η Ελληνική γλώσσα. Στο μέλλον έχει προγραμματιστεί να γίνει επιπλέον επέκταση των ανωτέρω γλωσσών, έχοντας ως αρχική ιδέα τα Γερμανικά.

Η επιλογή της γλώσσας στην οποία θα προβάλλεται η εφαρμογή μας, θα γίνεται αυτόματα και θα εξαρτάται από την προκαθορισμένη γλώσσα που θα χρησιμοποιείται από το λειτουργικού συστήματος της κάθε συσκευής. Πιο συγκεκριμένα όταν το λειτουργικό σύστημα δεν έχει ως προκαθορισμένη γλώσσα τα Ελληνικά, αυτή θα προβάλλεται αυτόματα στα Αγγλικά.

3.4.2 **Drawable**

Για την απεικόνιση των βασικών λειτουργιών από τις οποίες θα αποτελείται η εφαρμογή μας θα προσθέσουμε και τις αντίστοιχές τους εικόνες. Μία από τις βασικές μεθόδους του Android λειτουργικού συστήματος για την προβολή και την κατάλληλη επεξεργασία μίας εικόνας είναι η κλάση με την ονομασία Drawable και αντίστοιχα πρέπει να δημιουργηθεί και ο κατάλληλος φάκελος στα resources της εφαρμογής, για την ενσωμάτωση της σε αυτή. Ο φάκελος θα πρέπει να φέρει ειδική ονομασία, ώστε το Android λειτουργικό σύστημα να μπορέσει να τον αναγνωρίσει και να επεξεργαστεί κατάλληλα το περιεχόμενο του.

Οι διαστάσεις και η ανάλυση της κάθε οθόνης των κινητών συσκευών διαφέρουν ανά μοντέλο και κατασκευαστή. Για αυτόν το λόγο πρέπει να διαμορφώσουμε κατάλληλα τις εικόνες, έτσι ώστε να μπορούν να προβληθούν σε όλα τα μεγέθη οθονών και η ανάλυση τους να μην παραμορφωθεί. Για το παραπάνω λόγο πρέπει να προκαθοριστεί η δημιουργία περισσότερων φακέλων τύπου drawable. Οι επιπλέον φάκελοι θα έχουν στο τέλος τους μια πρόσθετη συμβολοσειρά, η οποία θα αποτελείται από την λέξη κλειδί, έτσι ώστε το λειτουργικό σύστημα να μπορεί να αναγνωρίσει και να χρησιμοποιήσει τον σωστό φάκελο. Με αυτόν το τρόπο έχουμε την δυνατότητα μιας γενικής κατηγοριοποίησης του κάθε τύπου οθόνης όπου και να λειτουργήσει η εφαρμογή μας.

Επομένως, για τη μέγιστη καλύτερη απεικόνιση της κάθε εικόνας από την οποία θα αποτελείται η εφαρμογή μας, θα δημιουργήσουμε τους αντίστοιχους φακέλους με το κυρίως όνομα drawable και με τις εξής συμβολοσειρές στο τέλος τους (-hdpi, -mdpi, -v24, -xhdpi, -xxhdpi).

Για να κάνουμε χρήση των φακέλων που δημιουργήσαμε, οι αρχικές εικόνες που διαλέξαμε για την εφαρμογή μας θα πρέπει να επεξεργαστούν κατάλληλα ως προς την ανάλυση τους σε dpi (dot per inch) και να περαστούν στον κατάλληλο φάκελο που τους αντιστοιχεί.

3.4.3 **Layout**

Λόγω τις περιορισμένης χρονικής διάρκειας για την υλοποίηση της εφαρμογής πήραμε την απόφαση να χρησιμοποιήσουμε μόνο το standard Layout, ώστε το γραφικό περιβάλλον να είναι πλήρως λειτουργικό για smartphone και phablets. Βάσει της απόφασης

αυτής, η δημιουργία των γραφικών περιεχομένων των περισσότερων Activities, έχουν σχεδιαστεί σε δυναμική μορφή προκειμένου να μπορούν να προσαρμοστούν και σε μεγαλύτερες συσκευές.

Σε συσκευές μεγέθους tablet και μεγαλύτερες η εφαρμογή θα είναι πλήρως λειτουργική, όμως όσο αφορά το γραφικό περιβάλλον UI, αυτή θα μειονεκτεί σε θέματα μεγέθους γραμματοσειράς και σε συγκεκριμένα Views στην στοίχιση τους.

3.4.4 **Dimensions**

Τα resources της εφαρμογής μας περιέχουν το αρχείο με την ονομασία dimens.xml, το οποίο μπορούμε να χρησιμοποιήσουμε για να ορίσουμε τις διαστάσεις των γραφικών στοιχείων που θα προβάλλονται στην εφαρμογή μας. Για την σωστή κλιμάκωση των στοιχείων που περιέχει το γραφικό περιβάλλον της εφαρμογής, το λειτουργικό σύστημα κατά την εκκίνηση τους θα ελέγξει αυτόματα τα αντίστοιχα αρχεία στα resources της.

Ένα σημαντικό μέρος το οποίο θα μας βοηθήσει στη σωστή οργάνωση και στη μελλοντική συντήρηση της εφαρμογής μας, είναι η χρήση του υποφακέλου dimen ο οποίος βρίσκεται στον κύριο φάκελο Values. Στο φάκελο αυτό, θα εισάγουμε όλες τις διαστάσεις των γραμματοσειρών που θα περιέχει το γραφικό περιβάλλον της εφαρμογής μας και ορισμένες διαστάσεις αντικειμένων τύπου Views, για τη σωστή λειτουργία τους σε συσκευές μεγέθους smartphone – phablets. Με τη συλλογή των διαστάσεων της εφαρμογής στον φάκελο dimen επιτυγχάνουμε την επαναχρησιμοποίηση των τιμών σε πολλά σημεία του γραφικού περιβάλλοντος της εφαρμογής.

Επιπλέον, με την δημιουργία ενός φακέλου values-sw600dp ο οποίος θα περιέχει μέσα τον υποφάκελο dimen-sw800, η εφαρμογή θα έχει τη δυνατότητα να αλλάζει το μέγεθος κυρίως των γραμματοσειρών σε συσκευές τύπου tablet και μεγαλύτερες, όπως επίσης τα μεγέθη των αντικειμένων τύπου Views.

3.4.5 **Colors**

Κατά την αρχική δημιουργία της εφαρμογής μας, το εργαλείο ανάπτυξης εφαρμογών δηλαδή το Android Studio, στην πρώτη εκκίνησή της εφαρμογής, δημιουργεί αυτόματα φακέλους και υποφακέλους για την διευκόλυνση του χρήστη. Ένας από αυτούς τους

φακέλους μέσα στα resources της εφαρμογής είναι ο υποφάκελος Colors όπου βρίσκεται μέσα στον φάκελο Values.

Με τον φάκελο Colors θα έχουμε την δυνατότητα της συλλογής των βασικών χρωμάτων, από τα οποία θα αποτελείται το γραφικό περιβάλλον της εφαρμογής μας. Με αυτόν τον τρόπο θα έχουμε τη δυνατότητα επαναχρησιμοποίησής τους σε όλες τις Activities καθώς και εύκολη αλλαγή τους στο μέλλον.

Το γραφικό περιβάλλον της εφαρμογή μας, θα αποτελείται από δύο κυρίως χρώματα. Το πρώτο χρώμα είναι αυτό με την ονομασία background και σκοπός του είναι να χρησιμοποιείται από όλες τις root Views των Activities ως υπόβαθρο. Το δεύτερο κυρίως χρώμα είναι αυτό με την ονομασία forground, το οποίο θα αποτελείται σχεδόν από το ίδιο χρώμα με το background χρώμα της εφαρμογής, όμως θα είναι λίγο πιο σκούρο και σκοπός του είναι να χρησιμοποιείται κυρίως από τα child Views των περισσότερων Activities.

Με αυτόν τον τρόπο θα προσπαθήσουμε να επιδιώξουμε τον σωστό συνδυασμό χρωμάτων μεταξύ υποβάθρου και αντικειμένων τύπου Views για τη σωστή προβολή του γραφικού περιβάλλοντος της εφαρμογής μας.

3.4.6 **DataBase**

Η εφαρμογή μας, για την διευκόλυνση και μελλοντική επαναχρησιμοποίηση των ήδη υπολογισμένων συστημάτων μετά τον πέρας του υπολογιστικού κομματιού της εφαρμογής, θα προσφέρει στον χρήστη την δυνατότητα αποθήκευσης της κάθε αυτόνομης φωτοβολταϊκής μελέτης η οποία θα έχει υπολογιστεί από την εφαρμογή.

Για τον σκοπό αυτόν, θα δημιουργήσαμε με την βοήθεια του Συστήματος Διαχείρισης Βάσεων Δεδομένων (ΣΔΒΔ) sqlite που υποστηρίζεται από το λειτουργικό σύστημα μια τοπική βάση δεδομένων. Η βάση δεδομένων αποτελείται λεπτομερώς από όλα τα τεχνικά χαρακτηριστικά των εξαρτημάτων που είναι απαραίτητα για την σωστή λειτουργία του αυτόνομου φωτοβολταϊκού συστήματος και τις βασικές τιμές πριν γίνει η πρόταση των εξαρτημάτων.

Επίσης, για τη μελλοντική συντήρηση και επέκταση της βάσης δεδομένων έχει συμπεριληφθεί επιπλέον κώδικας, ώστε να γίνει εφικτή η αναβάθμιση και η γρήγορη υποβάθμιση - επαναφορά της σε τυχόν προβλήματα που μπορεί να προκύψουν, ώστε αυτό να μην γίνει αντιληπτό από τον χρήστη.

3.5 User Interface

Ο κύριος στόχος που θέσαμε για την σχεδίαση του γραφικού περιβάλλοντος της εφαρμογής μας είναι ο χρήστης, να είναι σε θέση να βρίσκει εύκολα και γρήγορα το αντικείμενο που αναζητάει, ενώ παράλληλα να μην προκύπτουν μειονεκτήματα στη λειτουργική πλευρά και στη σωστή απεικόνιση του γραφικού περιβάλλοντος. Για τον λόγο αυτόν το κάθε Actvity της εφαρμογής μας θα περιέχει μόνο τα απαραίτητα αναγκαία αντικείμενα που χρειάζεται για την σωστή απεικόνιση της και όλα τα Activities θα προβάλλονται σε portrait View ώστε να διασφαλίσουμε την σωστή προβολή τους.

Θα επιλέξουμε δύο βασικά χρώματα που θα περιέχει η εφαρμογή μας, το background και το forground, τα οποία θα βρίσκονται στον φάκελο με την ονομασία Colors και θα τα χρησιμοποιήσουμε σε όλα τα Activities. Με την απόφαση αυτήν θα έχουμε την δυνατότητα στο μέλλον με ελάχιστο κόπο να πραγματοποιήσουμε την εναλλαγή των βασικών χρωμάτων από τα οποία θα αποτελούνται όλα τα Activities.

Επιπλέον για να καταφέρουμε την αρμονική απεικόνιση μεταξύ όλων των Activities θα προσπαθήσουμε να χρησιμοποιήσουμε τις περισσότερες φορές μέσα στα layout των Activities το αντικείμενο με την ονομασία CardView, το οποίο θα περιέχει τα απαραίτητα Views για τον κάθε σκοπό που αντιστοιχεί.

Ένα σημαντικό κομμάτι για την σχεδίαση, ήταν να καταφέρουμε η εφαρμογή μας να μπορεί να καλύψει τόσο τις ανάγκες του χρήστη ο οποίος έχει τις κατάλληλες εξειδικευμένες γνώσεις σχετικά με το αντικείμενο όσο και του μη γνώστη. Για τον λόγο αυτό, στην αντίστοιχη actvity όπου ο χρήστης πρέπει να εισάγει τα απαραίτητα στοιχεία που απαιτούνται για τον υπολογισμό της αυτόνομης φωτοβολταϊκής μελέτης, προστέθηκαν μόνο τα απαραίτητα πεδία εισαγωγής. Παράλληλα, για την καλύτερη υποστήριξη του μη εξειδικευμένου χρήστη κάθε πεδίο εισαγωγής δεδομένων, θα συνοδεύεται από τον αντίστοιχο σύνδεσμο, όπου θα προβάλλεται, μέσω των προσωρινών Activities, η αναλυτική επεξήγηση του παρόντος πεδίου. Επιπλέον, θα προσθέσουμε μια κατηγορία

στο αρχικό Activity της εφαρμογής μας το οποίο θα περιέχει όλες τις επεξηγήσεις συγκεντρωμένες και με επιπλέον οδηγίες.

Σε συνδυασμό με τον αλγόριθμο που υλοποιήσαμε για τον υπολογισμό των χαρακτηριστικών τις φωτοβολταϊκής μελέτης, δημιουργήσαμε ένα επιπλέον αλγόριθμο ο οποίος θα μπορεί να υπολογίζει και να συμβουλεύει τον χρήστη για τις κατάλληλες συσκευές και τον απαιτούμενο αριθμό τους, προκειμένου να μπορέσουν αυτές να πληρούν τα τεχνικά χαρακτηριστικά για την σωστή λειτουργία του αυτόνομου φωτοβολταϊκού συστήματος. Με βάση τα αποτελέσματα του αλγορίθμου που είναι υπεύθυνος για τον υπολογισμό της φωτοβολταϊκής μελέτης, θα γίνεται η σύγκριση των προ- απαιτούμενων τιμών με τις τιμές των πινάκων που περιέχουν τις αντίστοιχες συσκευές και θα προβάλλονται στον χρήστη.

Στον σχεδιασμό μας, η εφαρμογή αποτελείται από το αρχικό Activity με την ονομασία mainActivity.class το οποίο θα περιέχει τον τίτλο, ένα popUp-Menu στην κορυφή δεξιά και τις κύριες κατηγορίες της.

Το popUp-Menu, αποτελείται επίσης από τις κύριες κατηγορίες της εφαρμογής για την γρήγορη περιήγηση του χρήστη, όπου και να βρίσκεται μέσα στην εφαρμογή. Ένα σημαντικό χαρακτηριστικό το οποίο θα έχει το popUp-Menu, είναι ότι δεν δημιουργήθηκε μέσα στην ίδια την κλάση mainActivity από την οποία αποτελείται το αρχικό Activity. Δημιουργήθηκε σε ξεχωριστή κλάση (MyPopUpMenu.class) για την εφικτή επαναχρησιμοποίηση του σε όποιο Activity χρειαστεί.

Η αρχική Activity θα περιέχει τρείς βασικές κατηγορίες εκ των οποίων, η πρώτη υποκατηγορία θα αποτελείται από τον υπολογισμό του αυτόνομου φωτοβοταϊκού συστήματος, με την ονομασία (InputCalcDataActivity.class) και ο κύριος σκοπός της είναι η προβολή μίας φόρμας συμπλήρωσης για την συλλογή των απαραίτητων στοιχείων που θα χρειαστεί ο αλγόριθμος της εφαρμογής μας για να υπολογίσει την αυτόνομη φωτοβολταϊκή μελέτη. Ταυτόχρονα θα περιέχει τον τίτλο, ένα popUp-Menu στην κορυφή δεξιά και την φόρμα συμπλήρωσης, οι οποίες θα προβάλλονται μέσω του αντικειμένου cardView.

Συμπληρώνοντας ο χρήστης τα απαραίτητα στοιχεία του και με την πραγματοποίηση του υπολογισμού της μελέτης θα προβάλλεται αμέσως μετά την επόμενη Activity με

την ονομασία (CalcResultActivity.class), η οποία θα αποτελείται από το εικονίδιο της επιλογής αποθήκευσης του υπολογισμένου συστήματος στο πάνω μέρος αριστερά του Activity. Στο πάνω μέρος δεξιά έχουμε τοποθετήσει ένα εικονίδιο με σκοπό την μετάβαση του χρήση στο αρχικό Activity. Σε μορφή cardViews θα περιέχονται τα αντίστοιχα τεχνικά χαρακτηριστικά του συστήματος μαζί με τις απαραίτητες συσκευές, ώστε να μπορέσει να λειτουργήσει σωστά.

Στη δεύτερη κατηγορία με την ονομασία (SavedPvPlantsActivity.class) που περιέχει το αρχικό μας Activity δίνεται η δυνατότητα μελλοντικής ανασκόπησης των υπολογισμένων φωτοβολταϊκών συστημάτων που έχουν αποθηκευτεί. Αυτή θα αποτελείται από τον τίτλο της και το αντικείμενο τύπου ListView, ώστε να προβληθούν όλα τα αποθηκευμένα συστήματα του χρήστη.

Η τελευταία κατηγορία από την οποία θα αποτελείται η αρχική μας Activity, θα περιέχει τις απαραίτητες επεξηγήσεις και παραδείγματα για την βοήθεια και καθοδήγηση του χρήστη.

3.6 **Ανάλυση του MainActivity**

Κατά την αρχική εκκίνηση της εφαρμογής μας, η οθόνη θα αποτελείται από το Activity με την ονομασία MainActivity.class. Όπως έχουμε αναφέρει και προηγουμένως ο κύριος σκοπός της είναι η γρήγορη καθοδήγηση του χρήστη. Για αυτόν το λόγο προσθέσαμε μόνο τα αναγκαία Views. Το MainActivity περιέχει τρεις βασικές κατηγορίες από τις οποίες αποτελείται η εφαρμογής μας. Η πρώτη κατηγορία είναι υπεύθυνη για την υλοποίηση της αυτόνομης φωτοβολταϊκής μελέτης. Η επόμενη κατηγορία προσφέρει στον χρήστη την αναθεώρηση των ήδη αποθηκευμένων φωτοβολταικών μελετών του και η τελευταία κατηγορία θα περιέχει τις απαραίτητες επεξηγήσεις και παραδείγματα για την βοήθεια και καθοδήγηση του χρήστη.

Εικόνα 23 MainActivity

3.6.1 **Γραφικό περιβάλλον του MainActivity**

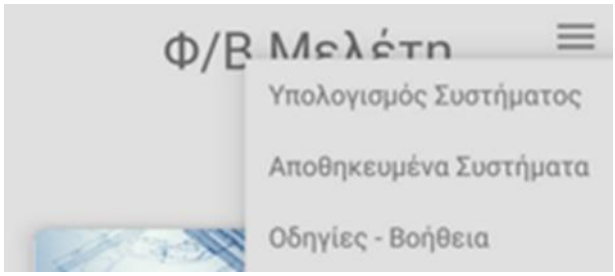
Για το σχεδιαστικό κομμάτι του γραφικού περιβάλλοντος του παρόντος Activity, χρησιμοποιήσαμε το layout με την ονομασία activity_main.xml. Με την δημιουργία ενός καινούργιου Activity αυτόματα δημιουργείται και ο αντίστοιχος xml φάκελός του. Σε αυτόν αποφασίσαμε να χρησιμοποιήσουμε ως Layout το εδώ και λίγο καιρό επίσημα δημοσιοποιημένο από την Google, ConstraitLayout με το εξής xml Tag android.support.constraint.ConstraintLayout. Το constraitLayout θα αποτελείται από τα Views που παρουσιάζονται παρακάτω.

3.6.1.1 Main Title

Εικόνα 24 MainTitle

Ο κύριος τίτλος του Activity υλοποιήθηκε με το αντικείμενο τύπου TextView και στοιχίζεται αυτόματα οριζοντίως στο πάνω μέρος του Activity. Επιπλέον το TextView αποτελείται από το σύνδεσμο με την ονομασία android:text ="@string/main_title" και ο σκοπός του είναι η εύρεση της αντίστοιχης συμβολοσειράς από την οποία θα αποτελείται ο τίτλος και της αντίστοιχης γλώσσας με βάση την προκαθορισμένη γλώσσα του λειτουργικού συστήματος.

3.6.1.2 PopUp Menu



Εικόνα 25 PopupMenu

Για την απεικόνιση του μενού, το οποίο περιέχει το Activity χρησιμοποιήσαμε τη χαρακτηριστική εικόνα των τριών οριζοντίων γραμμών οι οποίες αντιπροσωπεύονται από το αντικείμενο τύπου ImageView. Με το πάτημα της εικόνας, δημιουργείται το αντικείμενο τύπου popUpMenu και εμφανίζεται. Αυτό περιέχει τις κύριες κατηγορίες της εφαρμογής μας.

Επιπλέον, χρειάζεται να δημιουργήσουμε έναν υποφάκελο με την ονομασία menu μέσα στον φάκελο των resources της εφαρμογής. Αυτός περιέχει το xml αρχείο popup_menu.xml. Με το αρχείο αυτό θα μπορέσουμε να ορίσουμε τους τίτλους και τα αντίστοιχα id τους που θα περιέχει το popupMenu μας. Η στοίχιση της εικόνας γίνεται αυτόματα στην πάνω δεξιά περιοχή της οθόνης.

3.6.1.3 Card Views

Εικόνα 26 Main Categories

Το MainActivity περιέχει επιπλέον τρία σύνθετα αντικείμενα τύπου Views που αντιπροσωπεύουν τις κύριες κατηγορίες της εφαρμογής μας.

Οι τρείς κατηγορίες υλοποιήθηκαν με τον ίδιο ακριβώς τρόπο σχεδίασης και διαστασιολόγησης. Το root-View της κάθε κατηγορίας, το οποίο περιέχει τα υπόλοιπα Views είναι το αντικείμενο τύπου CardView. Ο κύριος σκοπός για τον οποίο το επιλέξαμε, είναι να μπορέσουμε έτσι να ομαδοποιήσουμε και να στοιχίσουμε πολλά Views εσωτερικά σε αυτό. Το μήκος της κάθε κατηγορίας επεκτείνεται αυτόματα στην πλήρη επιφάνεια της οθόνης χωρίς προκαθορισμένο μέγεθος, ώστε να διατηρήσουμε την δυναμικότητά του. Για αυτό αποφασίσαμε να προκαθορίσουμε μόνο το κενό το οποίο δημιουργείται στην αρχή και στο τέλος της κάθε κατηγορίας σε σχέση με την οθόνη.

Για την ακριβή τοποθέτηση των δύο child-Views από τα οποία αποτελείται η κάθε κατηγορία, επιλέξαμε το Layout με την ονομασία RelativeLayout. Το Layout αυτό αποτελείται από το αντικείμενο τύπου ImageView το οποίο περιέχει μία χαρακτηριστική εικόνα για την αντίστοιχη κατηγορία που θα αντιπροσωπεύει. Αμέσως κάτω από την εικόνα θα υπάρχει το αντικείμενο TextView όπου περιέχεται ο τίτλος της κατηγορίας.

Οι τρείς κατηγορίες στοιχίζονται και μοιράζουν τις αποστάσεις μεταξύ τους δυναμικά μεταξύ τους, ως προς το σημείο αναφοράς στον κάθετο άξονα της οθόνης. Για την πραγματοποίηση της στοίχισής τους χρησιμοποιήσαμε μια από τις ιδιότητες που πε-

ριέχει το ConstraintLayout με την ονομασία Chains. Για να δημιουργήσουμε την αλυσίδα μεταξύ τους, πρέπει οι τρείς κατηγορίες δηλαδή τα σύνθετα αντικείμενα τύπου Views να συνδεθούν μεταξύ τους. Για να επιτευχθεί αυτό πρέπει η κάθε κατηγορία να γνωρίζει την θέση της προηγούμενης και της επόμενής του, από τις οποίες θα περιβάλλεται. Επιπλέον, η αρχική κατηγορία μας πρέπει να είναι συνδεδεμένη με το τέλος του κυρίου τίτλου ώστε να προσδιορίσουμε το αρχικό σημείο αναφοράς το οποίο δεν πρέπει να υπερβαίνει το ύψος τους, και επιπλέον χρειάζεται να προσδιορίσουμε και το τελικό σημείο αναφοράς το οποίο θα βρίσκεται στο κάτω μέρος της οθόνης.

3.6.2 **Κώδικας του MainActivity**

Στο παρόν Activity στο προγραμματιστικό κομμάτι του δεν χρειάζεται να συμπεριλάβουμε πολλές και ιδιαίτερες γραμμές κώδικα, καθώς σχεδιάστηκε να λειτουργήσει σαν ένα πεδίο που θα συμπεριλαμβάνει όλες τις κύριες λειτουργίες σε μορφή συντομεύσεων για την εφαρμογής μας.

3.6.2.1 Find Views

Για να μετατρέψουμε τις κατηγορίες σε συντομεύσεις, αρχικά πρέπει να βρούμε τα σύνθετα Views, όπως αναφέραμε παραπάνω από το Layout που αντιστοιχεί στο Acitvity και να γίνει αναφορά-reference τους στα αντικείμενα που χρησιμοποιούνται στον κώδικα CardView, calcPvSystem = findViewById(R.id.calcCardView).

3.6.2.2 OnClickListener

Για να γίνει εφικτή από το σύστημα η επιτήρηση των πατημάτων του χρήστη δημιουργήσαμε την private μέθοδο με την ονομασία View OnClickListener onClickListener, η οποία αποτελείται από την υπάρχουσα μέθοδο public void onClick. Χρησιμοποιώντας την μέθοδο αυτή θα γίνει εφικτή η επιτήρηση των πατημάτων. Στην υπάρχουσα μέθοδο θα προσθέσουμε την εντολή ελέγχου switch, ώστε να μπορούμε να εντάξουμε την επιλογή του χρήστη στην σωστή κατηγορία. Στη συνέχεια με την βοήθεια του αντικειμένου Intent πραγματοποιείται άμεσα η μετάβαση στο αντίστοιχο Activity της κατηγορίας που επιλέχτηκε. Για την δημιουργία του αντικειμένου τύπου Intent στην συγκεκριμένη περίπτωση χρειάζεται να προωθήσουμε στον κατασκευαστή-Constructor του αντικειμένου το περιεχόμενο-context του παρόντος Activity και το όνομα του αντίστοιχου Activity που έχει επιλέξει να μεταβεί ο χρήστης. Στο τέλος, για να πραγματοποιηθεί η

μετάβαση είναι απαραίτητο να χρησιμοποιήσουμε την μέθοδο η οποία περιέχει το αντικείμενο Intent και να προσθέσουμε την μεταβλητή που δημιουργήσαμε που περιέχει την αναφορά-reference του, startActivity(intent); .

Για την πραγματοποίηση τις επιτήρησης των πατημάτων από το σύστημα χρειάζεται ακόμα να γίνει η σύνδεση των αντικειμένων τύπου Views με την μέθοδος OnClickListener που έχουμε δημιουργήσει.

3.6.2.3 PopUp Menu

Για την προβολή του popUpMenu μέσω ανίχνευσης πατημάτων στην οθόνη επαναχρησιμοποιήσαμε την μέθοδο που προαναφέραμε με την ονομασία OnClickListener. Με το πάτημα της εικόνας που απεικονίζει το μενού, αυτό θα δημιουργείται και θα εμφανίζεται.

Για την επαναχρησιμοποίηση του μενού σε άλλα Activities στην εφαρμογή μας δεν το συμπεριλάβαμε στον κώδικα στο παρόν Activity. Έτσι δημιουργήσαμε μια καινούργια κλάση-Class με την ονομασία MyPopUpMenu.class. Για την αρχική δημιουργία της Class θα χρειαστεί πρώτα να προωθήσουμε στον κατασκευαστή-Constructor που ενσωματώσαμε, το περιεχόμενο-Context του Activity και το View που θα περιέχει την εικόνα που θα απεικονίζει το μενού.

Η Class θα περιέχει την private μέθοδο void showPopUpMenu(), η οποία στο εσωτερικό σκέλος της θα δημιουργείται το αντικείμενο τύπου popupMenu. Αφού αρχικά δημιουργήσαμε το αντικείμενο μενού θα του προσθέσουμε το αρχείο popup_menu.xml, στο οποίο είχαμε ορίσει τους τίτλους που θα περιέχει και τα αντίστοιχα id τους. Για την επίβλεψη των πατημάτων μέσα στο ίδιο το μενού θα χρησιμοποιήσουμε την αντίστοιχη μέθοδο που μας παρέχει το αντικείμενο popupMenu με την ονομασία setOnMenuItemClickListener.

3.7 **Ανάλυση του InputCalcDataActivity**

Εικόνα 27 InputCalcDataActivity

Κύριος σκοπός του παρόντος Activity είναι η προβολή μίας φόρμας συμπλήρωσης για τη συλλογή των απαραίτητων στοιχείων που χρειάζεται ο αλγόριθμος της εφαρμογής μας, προκειμένου να γίνει εφικτός ο υπολογισμός των απαιτούμενων αναγκών της κάθε αυτόνομης φωτοβολταϊκής μελέτης, την οποία θα χρειαστεί ο χρήστης.

Το Activity αποτελείται από πέντε πεδία συμπλήρωσης και την επιλογή τοποθεσίας της αυτόνομης φωτοβολταϊκής εγκατάστασης. Επίσης, περιλαμβάνεται το πεδίο συμπλήρωσης με την ονομασία αξιολόγηση ηλιακού φορτίου, ώστε να γίνει η συλλογή της απαιτούμενης ενεργειακής κατανάλωσης της εγκατάστασης και το πεδίο συμπλήρωσης των διαδοχικών ημερών χρήσης του συστήματος σε μη επαρκή ηλιοφάνεια. Δίνεται η επιλογή εκτιμώμενης τάσης συστήματος. Για ακόμα πιο ακριβή υπολογισμό της ηλιακής ακτινοβολίας δίνεται και η επιλογή της εποχής στην οποία θα λειτουργήσει η εγκατάσταση. Μετά το πέρας της συμπλήρωσης των απαραίτητων στοιχείων που απαιτούνται και πριν την μετάβαση στο επόμενο Activity, εμφανίζεται στο κάτω μέρος του παρόντος Activity σε οριζόντια μορφή μία ένδειξη επεξεργασίας των παρόντων δεδομένων σε μορφή ProgressBar.

Επιπλέον, το κάθε πεδίο συμπλήρωσης που απαιτείται από το χρήστη να συμπληρώσει, συνοδεύεται από ένα σύνδεσμο ανακατεύθυνσης σε ένα άλλο προσωρινό Activity με σκοπό την επεξήγηση και τη βοήθεια σε αυτόν.

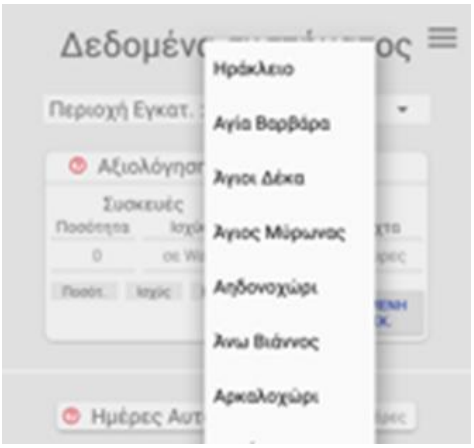
3.7.1 **Γραφικό περιβάλλον του InputCalcDataActivity**

Για το σχεδιαστικό κομμάτι του γραφικού περιβάλλοντος του συγκεκριμένου Activity χρησιμοποιήσαμε το layout με την ονομασία activity_input_calc_data.xml. Σε αυτό χρησιμοποιήθηκε το ConstraitLayout με το εξής xml Tag:

<android.support.constraint.ConstraintLayout>.

Για την πραγματοποίηση της στοίχισης των Views που περιέχει το Layout στον κάθετο άξονα της οθόνης, χρησιμοποιήσαμε μια από τις ιδιότητες που περιέχει το ConstraintLayout τα Chains. Το constraitLayout θα αποτελείται από τα εξής παρακάτω Views.

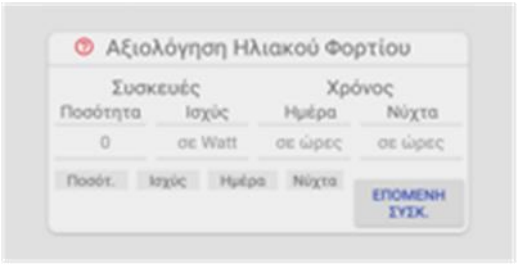
3.7.1.1 Ανάλυση Spinner



Εικόνα 28 Spinner

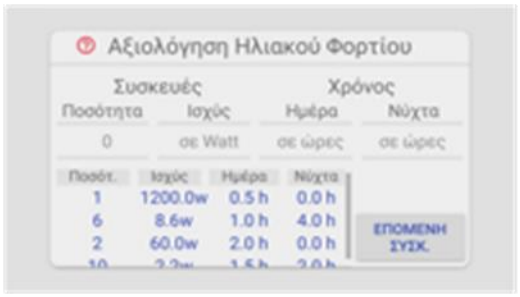
Στο πεδίο όπου ο χρήστης χρειάζεται να επιλέξει την περιοχή στην οποία βρίσκεται η εγκατάσταση, αποφασίσαμε να χρησιμοποιήσουμε το αντικείμενο τύπου View με την ονομασία Spinner. Το αντικείμενο αυτό το επιλέξαμε για την ομαδοποίηση των περιοχών προς επιλογή το οποίο αποτελείται από μια κατακόρυφη λίστα. Για να το καταφέρουμε αυτό προσθέσαμε επιπλέον στο σκέλος του xml κώδικα το android:spinnerMode="dropdown". Το Spinner αναλαμβάνει αυτόματα την επιτήρηση των πατημάτων της οθόνης και αποθηκεύει προσωρινά την επιλογή που διάλεξε ο χρήστης.

3.7.1.2 Αξιολόγηση Ηλιακού Φορτίου View



Εικόνα 29 Solar Evaluation_1

Εικόνα 30 Solar Evaluation_2



Εικόνα 31 Solar Evaluation_3

Για την υλοποίηση του συγκεκριμένου πεδίου εισαγωγής δεδομένων πρέπει να γίνει η χρήση πολλών αντικείμενων Views, ώστε να προκύψει το επιθυμητό αποτέλεσμα που επιδιώκουμε. Το αρχικό View, το οποίο περιέχει τα υπόλοιπα Views είναι το <android.support.v7.widget.CardView>. Το μήκος και το ύψος του εξαρτάται από το περιεχόμενο από το οποίο αποτελείται, για αυτό προσθέσαμε στο εσωτερικό του σκέλος android:layout_width="wrap_content", android:layout_height="wrap_content". Για την αρμονία των σχημάτων των αντικείμενων τύπου CardView σε όλα τα Activities της εφαρμογής χρησιμοποιήσαμε περίπου τις ίδιες xml γραμμές κώδικα, όπως στο αρχικό Activity δηλαδή την ίδια ακτίνα κύκλου για τις γωνίες του και το ίδιο μέγεθος σκιάς.

Για την αυτόματη κατακόρυφη τοποθέτηση των υπόλοιπων child-Views από τα οποία αποτελείται το αντικείμενο CardView θα κάνουμε χρήση του LinearLayout με το εξής χαρακτηριστικό android:orientation="vertical". Τα child-Views που θα περιέχονται θα είναι τα παρακάτω:

- Ο κυρίως τίτλος, ο οποίος με τη βοήθεια ενός <TextView> στοιχίζεται αυτόματα οριζοντίως μέσα στο root-View του.

- Κάτω από τον τίτλο έχουμε προσθέσει το View με την ονομασία <GridLayout>. Ο βασικός λόγος που επιλέξαμε αυτόν τον τύπο View, είναι ότι έχει τα χαρακτηριστικά ενός πίνακα που αποτελείται από στήλες και γραμμές. Με αυτόν τον τρόπο γίνεται η στοίχιση των Views που περιέχει ο πίνακας αυτόματα. Επιπλέον, έχουμε ορίσει η κάθε στήλη του πίνακά μας να περιέχει 4 Views. Τα επιπλέον views στοιχίζονται στην αμέσως επόμενη σειρά. Με αυτές τις ιδιότητες του πίνακα μπορούμε να στοιχίσουμε με ακρίβεια τα Views που περιέχει το GridLayout, δηλαδή τους τίτλους και τα αντίστοιχά τους πεδία εισόδου EditText το ένα κάτω από το άλλο. Επίσης για την καλύτερη κατηγοριοποίηση των πεδίων εισόδων που αναφέραμε παραπάνω προσθέσαμε δύο τίτλους με τις ονομασίες (Συσκευές, Χρόνος), για τη σωστή στοίχιση τους πάνω και στο κέντρο των δύο αντίστοιχων κατηγοριών που εκπροσωπούν. Ο τίτλος αυτόματα επεκτείνεται σε δύο στήλες, android:layout_columnSpan="2". Ο πίνακας αποθηκεύει τις τιμές του χρήστη και μηδενίζει τα πεδία εισαγωγής του αυτόματα, κάθε φόρα που θα πατάει το αντικείμενο τύπου Button με την ονομασία ΕΠΟΜΕΝΗ ΣΥΣΚ.

- Το επόμενο View αποτελείται πάλι από ένα αντικείμενο τύπου GridLayout και ο σκοπός του είναι να απεικονίζει τα δεδομένα που έχει ήδη καταχωρήσει ο χρήστης. Η αρχική του μορφή αποτελείται μόνο από τους τίτλους της κάθε στήλης. Κάθε φορά που ο χρήστης θα καταχωρεί τιμές στα πεδία που έχουμε αναφέρει πιο πάνω προστίθεται μια επιπλέον σειρά στον παρόν πίνακα. Το ύψος του αυξάνεται δυναμικά και δεν υπάρχει περιορισμός στις τιμές που θα δέχεται ο πίνακας. Αυτό όμως μας προτρέπει να περιορίσουμε το ύψος του από ένα σημείο και μετά ώστε να μην καλυφθούν τα υπόλοιπα Views που περιέχει το παρόν Activity. Για αυτόν τον λόγο περιορίσαμε την αύξηση του ύψους του πίνακα με σημείο αναφοράς το μέγεθος της οθόνης. Στη συνέχεια ενεργοποιείται το αντικείμενο τύπου View ScrollBar που περιβάλει τον πίνακα, ώστε να μπορεί να προσθέσει ο χρήστης λεπτομερώς όλα τα δεδομένα που θα χρειαστεί.

- Κάθε φορά που ο χρήστης θα καταχωρεί τιμές στα πεδία που έχουμε αναφέρει πιο πάνω θα προστίθεται μια επιπλέον σειρά στον παρόν πίνακα.

3.7.1.3 Υπόλοιπα Views

Τα υπόλοιπα πεδία εισόδου τα οποία περιέχονται στο συγκεκριμένο Activity θα αποτελούνται από τα αντικείμενα CardViews. Το περιεχόμενό τους θα αποτελείται από ένα TextView που θα εκπροσωπεί τον αντίστοιχο τίτλο του πεδίο εισαγωγής. Για τα πεδία εισαγωγής, που θα μπορεί ο χρήστης να εισάγει τις απαραίτητες τιμές θα κάνουμε χρήση του αντικειμένου EditText. Αυτά τα πεδία δέχονται μόνο δεκαδικούς αριθμούς. Επίσης για την επιλογή της εποχής κατά την οποία γίνεται χρήση του αυτόνομου φωτοβολταϊκού συστήματος χρησιμοποιήσαμε το αντικείμενο τύπου View το CheckBox.

3.7.1.4 Progress Bar

Εικόνα 32 ProgressBar

Το Activity περιέχει μία οριζόντια γραμμή προόδου. Κατά την χρονική στιγμή την οποία ο χρήστης θα συμπληρώνει τα απαραίτητα πεδία εισαγωγής αυτή δεν θα είναι εμφανής. Την στιγμή που πατηθεί το Button View που θα φέρει την ονομασία «Υπολογισμός» και όλα τα απαραίτητα πεδία του Activity θα είναι συμπληρωμένα, εμφανίζεται ανάμεσα στο προτελευταίο και τελευταίο View του. Σκοπός της γραμμής προόδου, είναι να απεικονίσουμε την χρονική διάρκεια που χρειάζεται ο αλγόριθμός μας να επεξεργαστεί τα δεδομένα του χρήστη, ώστε μετά να γίνει η μετάβασή του στο επόμενο Activity με τα αποτελέσματα.

3.7.2 **Κώδικας του InputCalcDataActivity**

Όπως έχουμε αναφέρει και στο προηγούμενο Activity πρέπει πρώτα να βρούμε τα Views από το Layout που αντιστοιχεί στο Activity, για να γίνει η αναφορά-reference τους στα αντικείμενα που θα χρησιμοποιήσουμε στον κώδικα. Πρέπει να δημιουργήσουμε την μέθοδο onClickListener, να γίνεται η επιτήρηση των πατημάτων της οθόνης από το λειτουργικό σύστημα και να συσχετίσουμε τα αντίστοιχα Views με την μέθοδο onClickListener.

3.7.2.1 Λειτουργία του Spinner

Το View που θα αναλυθεί δημιουργείται αυτόματα κατά την εκκίνηση του Activity και για τον λόγο αυτόν συμπεριλάβαμε τον κώδικά του μέσα στην μέθοδο από την οποία αποτελείται το κάθε Activity το onCreate. Μέσα στο σκέλος του, το onCreate, περιέχει τη δημιουργία που αντικείμενο τύπου ScrollView. Για να καταφέρουμε να του εισάγουμε τις επιθυμητές τιμές που περιέχει σε μορφή κατακόρυφης λίστας, πρέπει να χρησιμοποιήσουμε το αντικείμενο ArrayAdapter, μέσω του arrayAdapter. Με τη βοήθειά του πραγματοποιείται η διασύνδεση του αντικειμένου ScrollView και της στατικής λίστας που έχουμε δημιουργήσει σε ξεχωριστή κλάση-class με την ονομασία PvgData. Αυτή θα περιέχει όλες τις απαραίτητες περιοχές. Ταυτόχρονα ορίζουμε με τις μεθόδους onItemSelected και onNothingSelected την επίβλεψη των πατημάτων της οθόνης για την κατακόρυφη λίστα του ScrollView.

3.7.2.2 Έλεγχος δεδομένων εισαγωγής

Ο έλεγχος των δεδομένων που έχει εισάγει ο χρήστης πραγματοποιείται με δύο βασικούς ελέγχους. Ο σκοπός τους είναι να επιβεβαιώσουν τη συμπλήρωση των απαραίτητων πεδίων εισόδου που ζητούνται από τον χρήστη. Αν δεν πληρούνται οι συνθήκες

```
        // Get TextView Quantity User Value
        EditText quantity_editText = findViewById(R.id.editText_quantity);
        if ( quantity_editText.getText().toString().equals("") ) correctValues_Flag = false;
        else quantity_val = Integer.parseInt(quantity_editText.getText().toString() );
        // Get TextView Watt User Value
        EditText watt_editText = findViewById(R.id.editText_watt);
        if ( watt_editText.getText().toString().equals("") ) correctValues_Flag = false;
        else watt_val = Double.parseDouble(watt_editText.getText().toString() );

        // Get TextView Hour-Per-Day User Value
        EditText hourPerDay_editText = findViewById(R.id.editText_hoursPerDay);
        if ( !hourPerDay_editText.getText().toString().equals("") ){
            hoursPerDay_val = Double.parseDouble(hourPerDay_editText.getText().toString() );
        }
        // Get TextView Hour-Per-Night User Value
        EditText hourPerNight_editText = findViewById(R.id.editText_hoursPerNight);
        if ( !hourPerNight_editText.getText().toString().equals("") ){
            hoursPerNight_val = Double.parseDouble(hourPerNight_editText.getText().toString() );
        }
        // Check if Both Hour-Per-(Day, Night) are empty
        if ( hourPerDay_editText.getText().toString().equals("") &&
                hourPerNight_editText.getText().toString().equals("") ) correctValues_Flag = false;

        // Check (Solar Load) all three Fields for values
        if (correctValues_Flag ){
            addSolarLoadViewRow( quantity_val, watt_val, hoursPerDay_val, hoursPerNight_val);

            devicesPower.add(watt_val);
            devicesWattPerHourDay += (quantity_val * watt_val * hoursPerDay_val);
            devicesWattPerHourNight += (quantity_val * watt_val * hoursPerNight_val);

            Log.v( tag: "-----> Day    ", Double.toString(devicesWattPerHourDay) );
            Log.v( tag: "-----> Night  ", Double.toString(devicesWattPerHourNight) );

            /*  Set EditText values to zero TO show hint again  */
            quantity_editText.setText(null);
            watt_editText.setText(null);
            hourPerDay_editText.setText(null);
            hourPerNight_editText.setText(null);
        } else Toast.makeText( context: this, getResources().getString(R.string.toast_solarLoad), Toast
```

Εικόνα 33 Έλεγχος Δεδομένων_1

των εντολών ελέγχου θα εμφανίζεται το αντίστοιχο μήνυμα της κάθε εντολής ελέγχου στην οθόνη σε μορφή κειμένου για την καθοδήγηση του χρήστη. Τα πεδία εισόδων του Activity έχουν τροποποιηθεί για να δέχονται μόνο δεκαδικούς αριθμούς και για τον λόγο αυτόν δεν θα γίνεται περαιτέρω έλεγχος των δεδομένων.

Η ορθή συμπλήρωση των απαραίτητων στοιχείων που απαιτούνται από τον χρήστη για την εισαγωγή δεδομένων από το σύνθετο αντικείμενο View με την ονομασία «Αξιολόγηση Ηλιακού Φορτίου», το οποίο έχουμε αναφέρει σε προηγούμενη ενότητα, πραγματοποιείται με ένα σύνολο εντολών ελέγχου IF, το οποίο και απεικονίζεται στην εικόνα 32 Έλεγχος Δεδομένων_1. Το σύνθετο View αποτελείται από τέσσερα πεδία εισόδου, Ποσότητα (Quantity), Ισχύς (Power), Ημέρα και Νύχτα και ο χρήστης είναι απαραίτητο να συμπληρώσει την ποσότητα, την ισχύ και ένα από τα δύο πεδία ή και τα δύο που αντιστοιχούν στην ημέρα και στην νύχτα. Ο έλεγχος πραγματοποιείται κάθε φορά που ο χρήστης θα κάνει χρήση του button το οποίο περιέχει το σύνθετο View για να εισάγει τα δεδομένα του στον πίνακα, τον οποίο ενσωματώνει το σύνθετο View. Για τη συγγραφή πιο ευανάγνωστου κώδικα ελέγχου τιμών θα ορίσουμε αρχικά μια λογική μεταβλητή με την ονομασία correctValues_Flag. Αυτή θα δηλωθεί με αρχική τιμή True. Στη συνέχεια πραγματοποιείται έλεγχος των πεδίων για την ορθή συμπλήρωση τους και εάν ένα πεδίο δεν τηρεί τις προϋποθέσεις της εντολής ελέγχου θα αλλάξει η τιμή τις

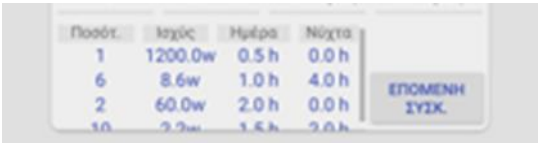
```
295
296     private void getRestOfUserData(){
297         // Set flag for empty input Views (editText, checkBox)
298         boolean fieldsCheck = true;
299
300         // Check (Solar Load) values
301         if (devicesWattPerHourDay == 0 && devicesWattPerHourNight == 0) fieldsCheck = false;
302
303         // Get & Check (Days Of Storage) value
304         EditText daysOfStorage_editText = findViewById(R.id.editText_daysOfStorage);
305         if (daysOfStorage_editText.getText().toString().equals("") ) fieldsCheck = false;
306         else daysOfStorage = Double.parseDouble(daysOfStorage_editText.getText().toString() );
307
308         // Get & Check (Estimate System Voltage) value
309         EditText estimateSystemVolt_editText = findViewById(R.id.editText_estimatedSystemVoltage);
310         if (estimateSystemVolt_editText.getText().toString().equals("") ) fieldsCheck = false;
311         else estimateSystemVoltage = Integer.parseInt(estimateSystemVolt_editText.getText().toString() );
312
313         // Find checked Box
314         if (checkBoxW.isChecked() ) {
315             userSeason = "Winter";
316         } else if (checkBoxS.isChecked() ) {
317             userSeason = "Summer";
318         } else if (checkBoxB.isChecked() ) {
319             userSeason = "Both";
320         } else fieldsCheck = false;
321
```

Εικόνα 34 Έλεγχος Δεδομένων_2

μεταβλητής correctValues_Flag αυτόματα σε False, ώστε η τελική εντολή ελέγχου να μην επιτρέψει την καταχώριση των τιμών στον πίνακα. Αντίστοιχα εμφανίζεται ένα μήνυμα στο γραφικό περιβάλλον του χρήστη επισημαίνοντας του τις κατάλληλες επεξηγήσεις.

Ο σκοπός του επόμενου ελέγχου είναι να ελεγχθούν τα υπόλοιπα πεδία από τα οποία αποτελείται το Activity για την ορθή συμπλήρωσή τους από τον χρήστη, σε συνδυασμό με την ορθή συμπλήρωση των δεδομένων του πίνακα του συνθέτου View Αξιολόγηση Ηλιακού Φορτίου. Ο έλεγχος θα πραγματοποιείται κάθε φορά που ο χρήστης θα κάνει χρήση του button υπολογισμός, που περιέχει το Activtiy στο κάτω μέρος του. Τα πεδία που θα ελεγχθούν αποτελούνται από τις ημέρες αυτονομίας, εκτιμώμενη τάση Συστήματος και την επιλογής της εποχής τα οποία έχουμε αναφέρει σε προηγούμενη ενότητα. Εφόσον πληρούνται τα κριτήρια που έχουμε θέσει μέσα στην εντολή ελέγχου της IF θα επιτραπούν στα δεδομένα να εισαχθούν στον αλγόριθμο για την επεξεργασία τους.

3.7.2.3 Create programmaticaly Views



Εικόνα 35 create views

Στο εσωτερικό του CardView με την ονομασία αξιολόγηση ηλιακού φορτίου, όπως έχουμε αναφέρει στο συγκεκριμένο Activity στην ενότητα 3.4.2.1.2 έχουμε τοποθετήσει το αντικείμενο GridLayout. Ο σκοπός του είναι η συλλογή και η προβολή των δεδομένων που έχει εισάγει και θα εισάγει ο χρήστης εκείνη την χρονική στιγμή. Για το καταφέρουμε αυτό, πρέπει αυτόματα στο GridLayout να προστίθενται αντικείμενα τύπου Views. Για το σκοπό αυτόν, η κάθε σειρά του GridLayout θα αποτελείται από τέσσερα TextViews. Για να προσθέσουμε τα Views είναι απαραίτητο να βρούμε το GridLayout που έχουμε δημιουργήσει στο layout του Activity και να το καταχωρίσουμε σε μια μεταβλητή για την περαιτέρω χρήση του στον κώδικά μας. Τα TextViews μας θα δημιουργούνται με προκαθορισμένο μέγεθος και χρώμα συμβολοσειράς. Το κάθε αντικείμενο θα δέχεται όμως, διαφορετική τιμή σε μορφή γραμματοσειράς, ώστε να είναι συμβατό

με τον τύπο μεταβλητής που δέχεται το αντικείμενο TextView. Για την εισαγωγή τους στο GridLayout θα χρειαστεί να καταχωρίσουμε τις μεταβλητές που περιέχουν την αναφορά στο κάθε αντικείμενο TextView και την ανάλογη θέση τοποθέτησης τους στο GridLayout. Το μέγεθος του πίνακα GridLayout αυξάνεται δυναμικά μέχρι να φτάσει το σύνολο των 5 σειρών δεδομένων με την βοήθεια της εντολής ελέγχου IF. Όταν ξεπεραστεί ο αριθμός των σειρών και δεν πληρούνται πια οι συνθήκες της εντολής ελέγχου, θα ελέγχεται αυτόματα το μέγεθος της οθόνης και θα γίνει η ανάλογη επέκταση του πίνακα σε σταθερό ύψος. Παράλληλα θα ενεργοποιηθεί το αντικείμενο ScrollView, στο οποίο βρίσκεται μέσα ο πίνακας ώστε να μην υπάρχει περιορισμός των δεδομένων προς απεικόνιση.

3.7.2.4 Λειτουργία του Progress Bar

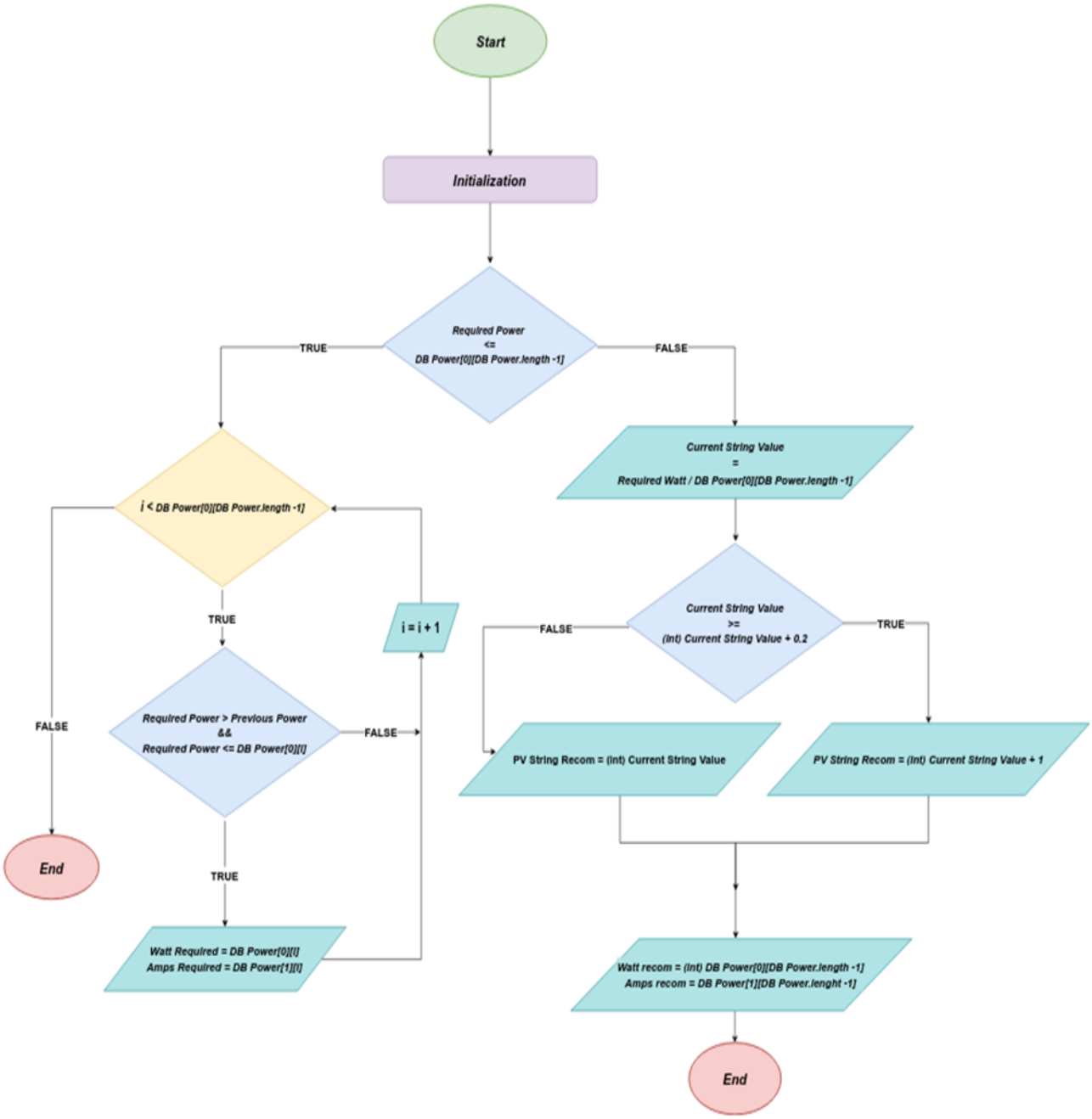
Η γραμμή προόδου αρχικά δεν είναι εμφανής στο γραφικό περιβάλλον του Activity, όμως όταν πληρούνται όλες οι προϋποθέσεις της τελικής εντολής ελέγχου IF που περιέχει ο κώδικάς μας, θα γίνει εντοπισμός του αντικειμένου ProgressBar στο αντίστοιχο Layout όπου βρίσκεται και θα εμφανιστεί μέσα στο γραφικό περιβάλλον της εφαρμογής. Μέσω του αντικειμένου Thread γίνεται εφικτή η αύξηση της προόδου, ωστόσο η ταχύτητα με την οποία θα φορτώνει έχει προκαθοριστεί. Για να γίνει εφικτή η σταδιακή αύξηση της γραμμής προόδου μέσω της μεθόδου Thread που έχουμε δημιουργήσει, χρειάστηκε να προσθέσουμε στο σκέλος της τον βρόχο επανάληψης while και το αντικείμενο τύπου Handler ώστε να πραγματοποιηθεί η πρόσβαση και η συνεχής αύξηση του αντικειμένου ProgressBar κατά την λειτουργία της μεθόδου. Όταν η γραμμή προόδου γεμίσει πλήρως θα διακόπτεται ο βρόχος επανάληψης while και θα πραγματοποιείται η αλλαγή του Activity που περιέχει τα αποτελέσματα του αλγόριθμου. Αυτό πραγματοποιείται μέσω του αντικείμενου Intent που έχουμε αναφέρει.

3.7.2.5 Αλγόριθμος PvStringDesigner

Για την κατάλληλη πρόταση των απαιτούμενων συσκευών από τις οποίες θα αποτελείται η εγκατάσταση, χρειάζεται επιπλέον να υλοποιήσουμε έναν αλγόριθμο. Ο συγκεκριμένος αλγόριθμος ξεκινάει την διαδικασία υπολογισμού, με βάση τα αποτελέσματα που θα προκύψουν από τον υπολογισμό των χαρακτηριστικών της αυτόνομης φωτοβολταϊκής μελέτης. Ο αλγόριθμος είναι σε θέση να προτείνει στον χρήστη την κατάλληλη ισχύ που πρέπει να φέρει ο φωτοβολταϊκός συλλέκτης και αντίστοιχα των

αριθμό τους, ώστε να καλύπτεται η ημερήσια ενεργειακή κατανάλωση της εγκατάστασης. Επίσης, υπολογίζεται από τον αλγόριθμο ο αντίστοιχος ρυθμιστής φόρτισης ο οποίος θα είναι σε θέση να συνεργαστεί με τους φωτοβολταϊκούς συλλέκτες της υπολογισμένης εγκατάστασης. Τα τεχνικά χαρακτηριστικά που φέρουν οι συσκευές και θα χρησιμοποιηθούν από τον αλγόριθμο για τις απαραίτητες προτάσεις στον χρήστη, έχουν συλλεχθεί μέσω του διαδικτύου και έχουν ομαδοποιηθεί κατάλληλα στον αλγόριθμο.

i. Υπολογισμός φωτοβολταϊκού συλλέκτη και αντίστοιχα ο αριθμός τους



Εικόνα 36 PhotovoltaicStringDesign-Flowchart

- Αρχικά τα δεδομένα που αντιστοιχούν στα τεχνικά χαρακτηριστικά των απαραίτητων συσκευών έχουν ομαδοποιηθεί σε δύο δισδιάστατους πίνακες οι οποίοι διαφέρουν στην τάση και στην ισχύς που αποδίδουν οι φωτοβολταϊκοί συλλέκτες. Η πρώτη γραμμή περιέχει την ισχύ των φωτοβολταϊκών συλλεκτών και η δεύτερη γραμμή του πίνακα αποτελείται από το ρεύμα το οποίο αποδίδει η αντίστοιχη στήλη της πρώτης γραμμής. Επίσης, έχουμε καταχωρίσει επιπλέον

μεταβλητή σε μορφή πίνακα, η οποία περιέχει το ρεύμα των διαθέσιμων ρυθμιστών φόρτισης που διατίθενται στην αγορά.

```
public class PvScStringDesigner {

    private double watt_requite;
    private double systemVoltage;
    // 24 volt Market Photovoltaic Panels Specs
    private double[][] pv24voltProduction_Watt_Ipmax = {
            {12, 20, 30, 40, 50, 80, 120, 150, 180, 190, 200, 210, 225, 230, 240, 256, 260, 270, 280, 300, 320, 350, 360},
            {0.67, 1.12, 1.71, 1.2, 1.61, 2.37, 3.3, 3.9, 4.5, 3.44, 5.28, 5.1, 7.5, 7.73, 8.11, 9.3, 8.48, 8.68, 8.89, 9.34, 8.69, 8.12, 9.87}
    };
    // 12 volt Market Photovoltaic Panels Specs
    private double[][] pv12voltProduction_Watt_Ipmax = {
            {5, 10, 20, 25, 40, 55, 80, 85, 105, 125, 130, 145, 150, 155},
            {0.3, 0.59, 1.34, 1.4, 2.28, 3.3, 4.55, 4.87, 5.8, 7.2, 7.5, 8.1, 8.26, 8.03}
    };
    // 12/24 volt Market Solar Charger Specs
    private int[] solarChanger_Amps = {4, 5, 10, 15, 20, 30, 35, 40};
```

Εικόνα 37 PvStringDesigner_Variables

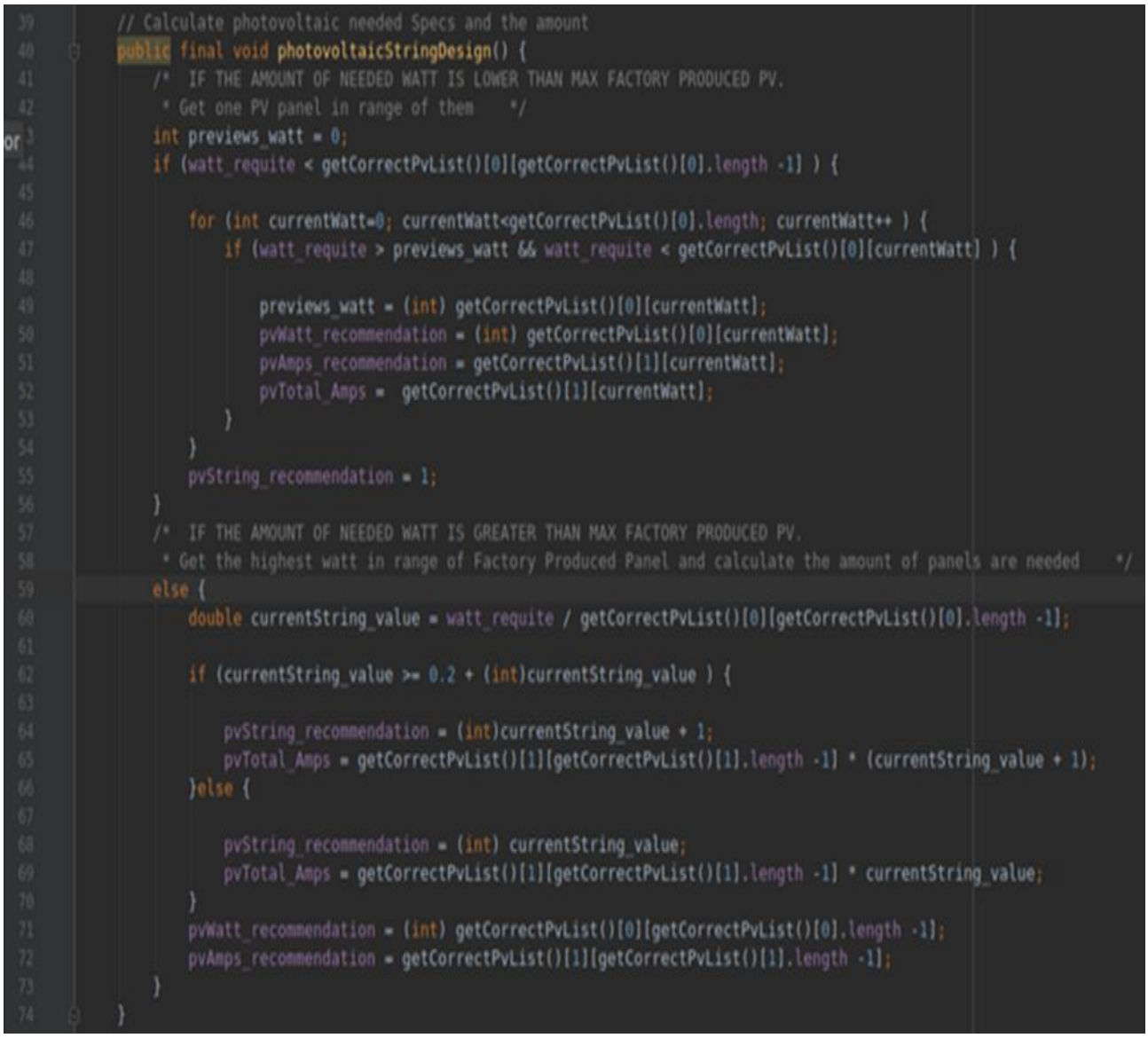
- Επίσης για τον υπολογισμό των απαραίτητων συσκευών θα χρειαστεί να εισάγουμε στον αλγόριθμο την συνολική ισχύ της αυτόνομης φωτοβολταϊκής μελέτης και την τάση στην οποία θα δουλέψει το σύστημα. Τα δεδομένα αυτά εισάγονται αυτόματα κατά την εκκίνηση της κλάσης μέσω του κατασκευαστή (Constructor) που έχουμε προσθέσει. Πριν υπολογιστούν οι συσκευές δημιουργήσαμε την μέθοδο getCorrectPvList(), από την οποία παρέχεται ο κατάλληλος δισδιάστατος πίνακας στον υπόλοιπο κώδικα του αλγορίθμου, ώστε να εξασφαλίσουμε την χρήση του απαραίτητου αντίστοιχου πίνακα και την μείωση της επανάληψης του κώδικα.

```
    public PvScStringDesigner(double watt_requite, double systemVoltage ){
        this.watt_requite = watt_requite;
        this.systemVoltage = systemVoltage;
    }


    private double[][] getCorrectPvList() {
        if ( systemVoltage == 12 ) return pv12voltProduction_Watt_Ipmax;
        else return pv24voltProduction_Watt_Ipmax;
    }
```

Εικόνα 38 PvStringDesigner_Constr_Arrays

- Για τον υπολογισμό της ισχύος του φωτοβολταϊκού συλλέκτη και αντίστοιχα τον αριθμό τους δημιουργήσαμε την μέθοδο με την ονομασία photovoltaicStringDesign().



```
// Calculate photovoltaic needed Specs and the amount
public final void photovoltaicStringDesign() {
    /*  IF THE AMOUNT OF NEEDED WATT IS LOWER THAN MAX FACTORY PRODUCED PV.
     * Get one PV panel in range of them    */
    int previews_watt = 0;
    if (watt_requite < getCorrectPvList()[0][getCorrectPvList()[0].length -1] ) {

        for (int currentWatt=0; currentWatt<getCorrectPvList()[0].length; currentWatt++ ) {
            if (watt_requite > previews_watt && watt_requite < getCorrectPvList()[0][currentWatt] ) {

                previews_watt = (int) getCorrectPvList()[0][currentWatt];
                pvWatt_recommendation = (int) getCorrectPvList()[0][currentWatt];
                pvAmps_recommendation = getCorrectPvList()[1][currentWatt];
                pvTotal_Amps =  getCorrectPvList()[1][currentWatt];
            }
        }
        pvString_recommendation = 1;
    }
    /*  IF THE AMOUNT OF NEEDED WATT IS GREATER THAN MAX FACTORY PRODUCED PV.
     * Get the highest watt in range of Factory Produced Panel and calculate the amount of panels are needed    */
    else {
        double currentString_value = watt_requite / getCorrectPvList()[0][getCorrectPvList()[0].length -1];

        if (currentString_value >= 0.2 + (int)currentString_value ) {

            pvString_recommendation = (int)currentString_value + 1;
            pvTotal_Amps = getCorrectPvList()[1][getCorrectPvList()[1].length -1] * (currentString_value + 1);
        }else {

            pvString_recommendation = (int) currentString_value;
            pvTotal_Amps = getCorrectPvList()[1][getCorrectPvList()[1].length -1] * currentString_value;
        }
        pvWatt_recommendation = (int) getCorrectPvList()[0][getCorrectPvList()[0].length -1];
        pvAmps_recommendation = getCorrectPvList()[1][getCorrectPvList()[1].length -1];
    }
}
```

Εικόνα 39 PvStringDesigner-Method

- Αρχικά περιέχει την εντολή ελέγχου IF-ELSE, η οποία ενσωματώνει και τον υ-πόλοιπο κώδικα που απαιτείται για την επιλογή του συλλέκτη. Η εντολή ελέγχου αντίστοιχα ελέγχει τη φωτοβολταϊκή εγκατάσταση αν χρειάζεται περισσότερους από έναν φωτοβολταϊκό συλλέκτη για την κάλυψη της απαιτούμενης ισχύος της εγκατάστασης. Ο παρών έλεγχος πραγματοποιείται με την σύγκριση της απαι-τούμενης ισχύος της εγκατάστασης σε σχέση με την υψηλότερη ισχύ του φωτο-βολταϊκού συλλέκτη που περιέχει ο αντίστοιχος δισδιάστατος πίνακας.

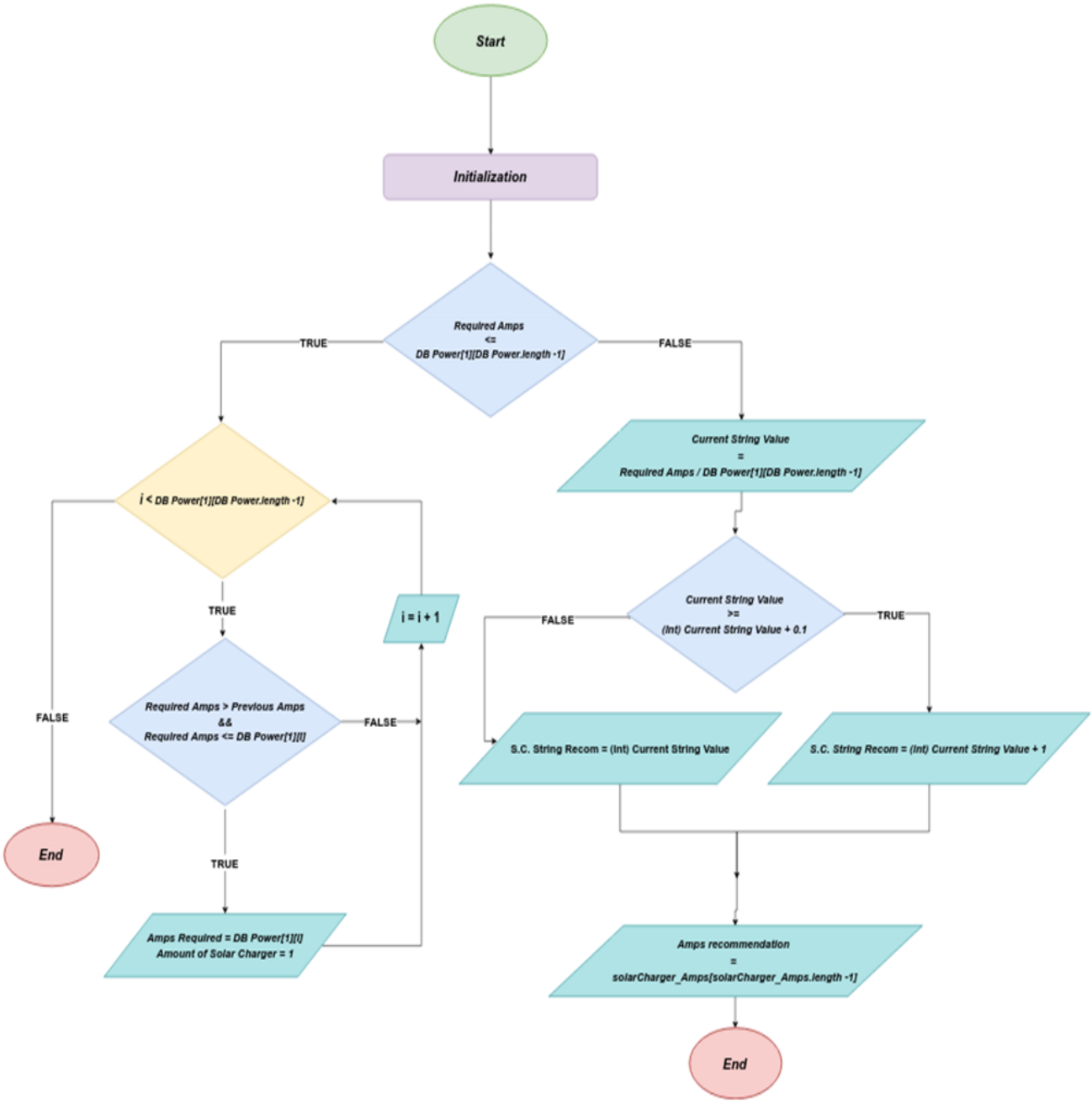
- Εφόσον δεν απαιτούνται περισσότεροι από έναν φωτοβολταϊκό συλλέκτη χρη-σιμοποιείται ο βρόχος επανάληψης FOR, ο οποίος συγκρίνει σε αύξουσα σειρά

τις τιμές του δισδιάστατου πίνακα με την απαιτούμενη ισχύ της εγκατάστασης για την κατάλληλη επιλογή του συλλέκτη.

- Όταν απαιτούνται περισσότεροι φωτοβολταϊκοί συλλέκτες γίνεται αυτόματα χρήση της εντολής ελέγχου ELSE.

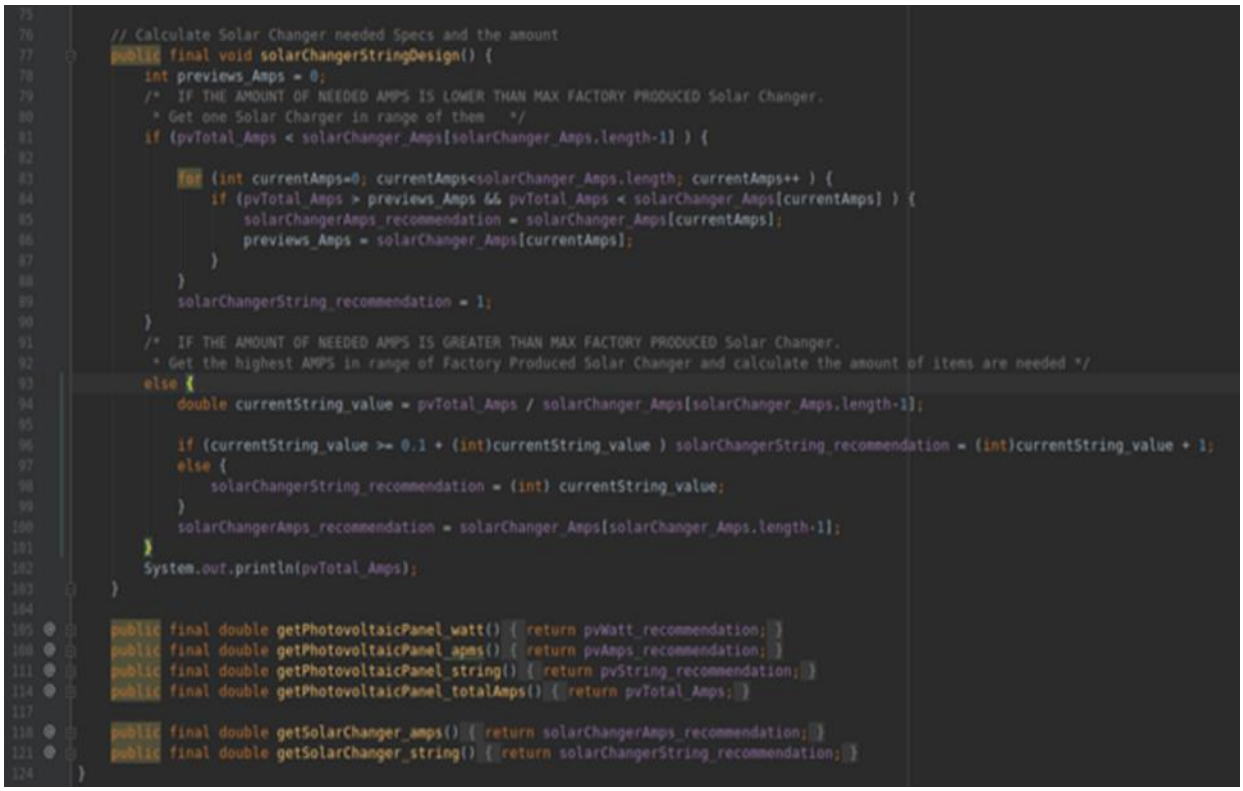
  - Αρχικά θα διαιρέσουμε την απαιτούμενη ισχύ της εγκατάστασης με την υψηλότερη ισχύ του συλλέκτη που παρέχεται από τον δισδιάστατο πίνακα και θα καταχωρίσουμε το πηλίκο στην μεταβλητή currentString_value. Το πηλίκο το οποίο προκύπτει είναι ο αντίστοιχος αριθμός συλλεκτών που απαιτούνται για την κάλυψη της ισχύος της εγκατάστασης. Κατά την διαίρεση τους μπορεί να προκύψει δεκαδικός αριθμός, ο οποίος μας υποδεικνύει με ακρίβεια το σύνολο των συλλεκτών για την κάλυψη της απαιτούμενης ισχύος της εγκατάστασης. Για τον λόγο αυτόν θα πρέπει να ελέγξουμε αν απαιτείται επιπλέον συλλέκτης να προστεθεί στο ακέραιο μέρος του πηλίκου της διαίρεσης με κριτήριο το μέγεθος του δεκαδικού αριθμού. Το κριτήριο ελέγχου που θέσαμε ώστε να μην προσθέσουμε επιπλέον συλλέκτη είναι ο δεκαδικός από τον οποίο αποτελείται το πηλίκο να μην ξεπερνάει την τάξη του 0.2. Για τον παρόν έλεγχο ο αλγόριθμος θα μετατρέψει την απαιτούμενη ισχύ της εγκατάστασης σε ακέραιο αριθμό και θα του προσθέσει το δεκαδικό αριθμό που έχουμε θέσει για τον έλεγχο 0.2 επαναφέροντας τον πάλι σε δεκαδικό όμως με την τιμή ελέγχου. Με αυτόν τον τρόπο θα πραγματοποιηθεί η σύγκριση της αρχικής τιμής (απαιτούμενη ισχύ της εγκατάστασης) με την ισχύ ελέγχου.

ii. Υπολογισμός ρυθμιστή φόρτισης και αντίστοιχα ο αριθμός τους



Εικόνα 40 SolarChargerStringDesign- Flowchart

- Για τον υπολογισμό του ρεύματος του ρυθμιστή φόρτισης και αντίστοιχα τον αριθμό τους δημιουργήσαμε την μέθοδο με την ονομασία solarChargerStringDesign().



```
// Calculate Solar Changer needed Specs and the amount
public final void solarChangerStringDesign() {
    int previews_Amps = 0;
    /*  IF THE AMOUNT OF NEEDED AMPS IS LOWER THAN MAX FACTORY PRODUCED Solar Changer.
     * Get one Solar Charger in range of them   */
    if (pvTotal_Amps < solarChanger_Amps[solarChanger_Amps.length-1] ) {

        for (int currentAmps=0; currentAmps<solarChanger_Amps.length; currentAmps++ ) {
            if (pvTotal_Amps > previews_Amps && pvTotal_Amps < solarChanger_Amps[currentAmps] ) {
                solarChangerAmps_recommendation = solarChanger_Amps[currentAmps];
                previews_Amps = solarChanger_Amps[currentAmps];
            }
        }
        solarChangerString_recommendation = 1;
    }
    /*  IF THE AMOUNT OF NEEDED AMPS IS GREATER THAN MAX FACTORY PRODUCED Solar Changer.
     * Get the highest AMPS in range of Factory Produced Solar Changer and calculate the amount of items are needed */
    else {
        double currentString_value = pvTotal_Amps / solarChanger_Amps[solarChanger_Amps.length-1];

        if (currentString_value >= 0.1 + (int)currentString_value ) solarChangerString_recommendation = (int)currentString_value + 1;
        else {
            solarChangerString_recommendation = (int) currentString_value;
        }
        solarChangerAmps_recommendation = solarChanger_Amps[solarChanger_Amps.length-1];
    }
    System.out.println(pvTotal_Amps);
}

public final double getPhotovoltaicPanel_watt() { return pvWatt_recommendation; }
public final double getPhotovoltaicPanel_amps() { return pvAmps_recommendation; }
public final double getPhotovoltaicPanel_string() { return pvString_recommendation; }
public final double getPhotovoltaicPanel_totalAmps() { return pvTotal_Amps; }

public final double getSolarChanger_amps() { return solarChangerAmps_recommendation; }
public final double getSolarChanger_string() { return solarChangerString_recommendation; }
}
```

Εικόνα 41 PvStringDesigner_SolarCharger-Calculation

- Αρχικά η εντολή ελέγχου IF-ELSE ελέγχει το συνολικό αριθμό απαιτούμενων ρυθμιστών φόρτισης για την ορθή λειτουργία ήδη υπολογισμένων φωτοβολταϊκών συλλεκτών. Ο παρών έλεγχος θα πραγματοποιηθεί με την σύγκριση του ρεύματος του αποδίδει ο συλλέκτης ή το σύνολο τους, το οποίο απαιτείται σε σχέση με το υψηλότερο ρεύμα του ρυθμιστή φόρτισης που περιέχει ο αντίστοιχος πίνακα.

- Εφόσον δεν απαιτούνται περισσότεροι από έναν ρυθμιστή φόρτισης χρησιμοποιείται ο βρόχος επανάληψης FOR, ο οποίος συγκρίνει σε αύξουσα σειρά τις τιμές του ακέραιου πίνακα με το απαιτούμενο ρεύμα του συλλέκτη για την κατάλληλη επιλογή του ρυθμιστή φόρτισης.

- Όταν απαιτούνται περισσότεροι ρυθμιστές φόρτισης γίνεται αυτόματα χρήση της εντολής ελέγχου ELSE, η οποία όπως και έχουμε περιγράψει για την εύρεση

φωτοβολταϊκού συλλέκτη θα διαιρέσει το συνολικό ρεύμα που αποδίδουν οι υπολογισμένοι φωτοβολταϊκοί συλλέκτες με την μεγαλύτερη τιμή ρεύματος που περιέχει ο πίνακας τιμών του ρυθμιστή φόρτισης.

3.8 Ανάλυση του CalcResultActivity

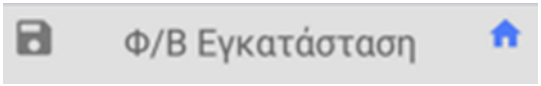
Για την προβολή των αποτελεσμάτων της αυτόνομης φωτολταϊκής μελέτης κάνουμε χρήση του CalcResultActivity Activity. Αυτό αποτελείται από τον τίτλο, περιέχει σε μορφή εικόνας την επιλογή της αποθήκευσης της υπολογισμένης μελέτης και για διευκόλυνση του χρήστη προσθέσαμε επιπλέον την επιλογή μετάβασης του στο αρχικό Activity της εφαρμογής. Η συγκεκριμένη εντολή προβάλλεται με την χαρακτηριστική εικόνα. Για την απεικόνιση και διαχωρισμό των υπολογισμένων δεδομένων της αυτόνομης φωτοβολταϊκής μελέτης δημιουργήσαμε τέσσερις κατηγορίες οι οποίες περιέχουν τα απαραίτητα τεχνικά χαρακτηριστικά της μελέτης. Αυτές εκπροσωπούνται με τις αντίστοιχες εικόνες και τίτλους τους, θα ακολουθούν τα τεχνικά χαρακτηριστικά που πρέπει να πληροί το κάθε εξάρτημα για την σωστή λειτουργία της εγκατάστασης και επιπλέον θα γίνεται και η αντίστοιχη πρόταση εξαρτημάτων τους.

Εικόνα 42 CalcResultActivity

3.8.1 **Γραφικό περιβάλλον του CalcResultActivity**

Για την πραγματοποίηση του σχεδιαστικού κομματιού του γραφικού περιβάλλοντος του παρόντος Activity, χρησιμοποιήσαμε το layout με την ονομασία activity_input_calc_data.xml. Σε αυτό αποφασίσαμε να χρησιμοποιήσουμε το ConstraitLayout με το εξής xml Tag android.support.constraint.ConstraintLayout. Για την πραγματοποίηση της στοίχισης των Views που περιέχει το Layout στον κάθετο άξονα της οθόνης, θα χρησιμοποιήσουμε μια από τις ιδιότητες που περιέχει το ConstraintLayout τα Chains. Το constraitLayout θα αποτελείται από τα εξής Views.

3.8.1.1 DataBase - Home Button



Εικόνα 43 Buttons

Για την πραγματοποίηση της επιλογής αποθήκευσης της αυτόνομής φωτοβολταϊκής μελέτης από τον χρήστη και την επιλογή επιστροφής στο αρχικό Activity χρησιμοποιήσαμε το αντικείμενο τύπου ImageView, το οποίο περιέχει τις αντίστοιχες χαρακτηριστικές εικόνες. Η στοίχισή τους πραγματοποιείται αυτόματα στην πάνω πλευρά του Activity.

3.8.1.2 Προβολή των υπολογισμένων Δεδομένων

Εικόνα 44 ResultView

Τα υπόλοιπα πεδία εισόδου, τα οποία περιέχονται στο Activity αποτελούνται από τα κυρίως αντικείμενα CardViews. Το περιεχόμενό τους αποτελείται από ένα TextView που εκπροσωπεί τον αντίστοιχο τίτλο του κάθε πεδίου δεδομένων μαζί με την αντίστοιχη εικόνα τους σε μορφή ImageView. Για την προβολή των δεδομένων που περιέχει το κάθε αντικείμενο CardView προσθέσαμε επιπλέον το αντικείμενο τύπου GridView. Με την βοήθειά του θα μπορέσουμε να στοιχήσουμε τους τίτλους με τα αντίστοιχα πεδία αποτελεσμάτων τους.

3.8.2 **Κώδικας της CalcResultActivity**

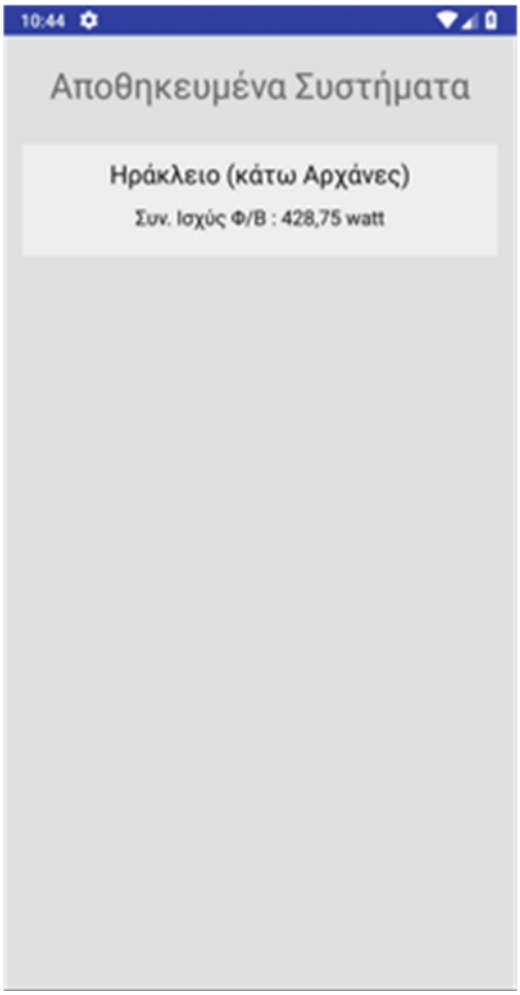
Στο Activity και όπως έχουμε αναφέρει και στα προηγούμενα Activities πρέπει πρώτα να βρούμε τα απαραίτητα Views από το Layout τους για να γίνει η αναφορά-reference τους στα αντικείμενα που χρησιμοποιήσαμε στον κώδικα. Πρέπει να δημιουργήσουμε την μέθοδο onClickListener, να γίνεται η επιτήρηση των πατημάτων της οθόνης από το λειτουργικό σύστημα και να συσχετίσουμε τα αντίστοιχα τους Views με την μέθοδο onClickListener.

3.8.2.1 DataBase Input

Για την πραγματοποίηση της εισαγωγής της υπολογισμένης μελέτης στην βάση δεδομένων χρειαζόμαστε τα δεδομένα που προέκυψαν από τον υπολογισμό της μελέτης και ένα χαρακτηριστικό τίτλο, ο οποίος προσδιορίζεται από τον χρήστη. Για την εισαγωγή του τίτλου χρησιμοποιήσαμε το αντικείμενο τύπου AlertDialogBox το οποίο εμφανίζεται μέσα στο Activity και περιέχει τον χαρακτηριστικό τίτλο του και το πεδίο εισαγωγής του τίτλου της εγκατάστασης. Θα γίνεται έλεγχος για πιθανόν κενή συμβολοσειρά και εφόσον πληρούνται οι προϋποθέσεις της εντολής ελέγχου IF θα εισάγουμε τα απαραίτητα δεδομένα στην τοπική βάση δεδομένων της συσκευής και θα κλείνει αυτόματα το αντικείμενο AlertDialogBox. Εφόσον τα δεδομένα περάστηκαν με επιτυχία στην βάση δεδομένων, ο χρήστης ειδοποιείται κατάλληλα. Το αντικείμενο που χρησιμοποιήσαμε για αυτόν τον σκοπό είναι το Toast, του οποίου η διάρκεια εμφάνισης θα είναι στα τρία δευτερόλεπτα και μετά την παρέλευση του χρόνου θα αποκρύπτεται αυτόματα.

3.9 **Ανάλυση του SavedPvPlantsActivity**

Ο σκοπός του Activity είναι η απεικόνιση όλων των υπολογισμένων αυτόνομων φωτοβολταϊκών συστημάτων που έχει αποθηκεύσει ο χρήστης. Τα δεδομένα τα οποία αποτελούν την κάθε υπολογισμένη μελέτη αποθηκεύονται στην τοπική βάση δεδομένων που έχουμε δημιουργήσει. Για την ομαδοποιημένη προβολή τους μέσα στο Activity χρησιμοποιήσουμε το αντικείμενο ListView. Η σειρά με την οποία θα στοιχίζονται εξαρτάται από την χρονολογική δημιουργία τους, ώστε τα πιο πρόσφατα να εμφανίζονται πρώτα και με την σειρά τους τα επόμενα. Επιλέγοντας προς προεπισκόπηση από τις υπάρχουσες μελέτες, επαναχρησιμοποιήσαμε την κλάση-Class με την ονομασία CalcResultActivity που έχουμε περιγράψει στην ενότητα 3.4.3 ώστε τα δεδομένα τις εγκατάστασης να απεικονιστούν με τον ίδιο τρόπο όπως εμφανίζονται στην ολοκλήρωση του υπολογισμού της αυτόνομης φωτοβολταϊκής μελέτης, όμως αυτήν την φορά παρέχονται τα στοιχεία από την βάση δεδομένων. Για την διαγραφή μίας επιθυμητής μελέτης από την βάση δεδομένων χρειάζεται ένα παρατεταμένο πάτημα πάνω στο υπάρχον αντικείμενο. Το Activity όταν δεν υπάρχουν διαθέσιμες αποθηκευμένες μελέτες θα εμφανίζει αυτόματα ένα προκαθορισμένο κείμενο επισημαίνοντας στον χρήστη ότι δεν υπάρχουν φωτοβολταϊκές μελέτες προς εμφάνιση.



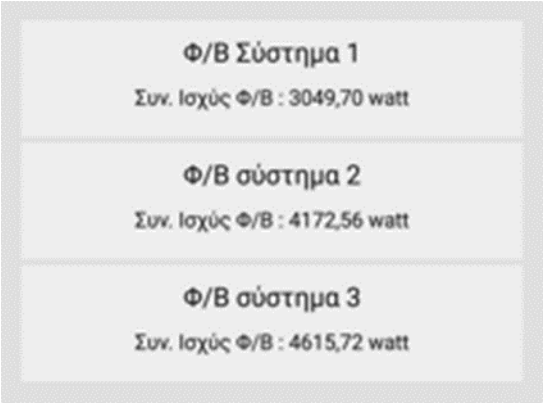
Εικόνα 45 SavedPvPlantsActivity

3.9.1 **Γραφικό περιβάλλον της SavedPvPlantsActivity**

Για το σχεδιαστικό κομμάτι του γραφικού περιβάλλοντος του συγκεκριμένου Activity χρησιμοποιήσαμε το layout με την ονομασία activity_saved_pv_plants.xml. Σε αυτό

αποφασίσαμε να χρησιμοποιήσουμε το RelativeLayout, το οποίο είναι πλήρες συμβατό με το ListView που περιέχει μέσα για την προβολή τον αποθηκευμένων συστημάτων. Το constraitLayout θα αποτελείται από τα εξής παρακάτω Views.

3.9.1.1 ListView



Εικόνα 46 ListView

Για την συλλογή και προβολή πολλών δεδομένων μέσα στο layout μας χρησιμοποιήσαμε το αντικείμενο ListView. Το παρόν αντικείμενο συμπεριλαμβάνει πολλά χαρακτηριστικά τα οποία μας διευκολύναν στην υλοποίηση της προβολής των αποθηκευμένων εγκαταστάσεων του χρήστη. Με τα χαρακτηριστικά αυτά μπορούμε να δημιουργήσουμε μια δυναμική λίστα, η οποία φορτώνει όλα τα δεδομένα που έχει αποθηκεύσει ο χρήστης απευθείας από την βάση δεδομένων και στην οποία πραγματοποιείται η δυναμική αφαίρεση σειρών. Η λίστα γνωρίζει αυτόματα το ύψος της διαθέσιμης οθόνης και σε περίπτωση που ξεπερνάει τα επιθυμητά όρια θα είναι προ βάσιμα τα δεδομένα μέσω μίας κάθετης στήλης επιλογής κατεύθυνσης και επιπλέον συμπεριλαμβάνεται η επίβλεψή πατημάτων για τις κατάλληλες ενέργειες.

3.9.2 **Κώδικας του SavedPvPlantsActivity**

Στο παρόν Activity επικεντρωθήκαμε κυρίως στην δημιουργία της ListView και τη σύναψή της με τα απαραίτητα δεδομένα που θα προβάλει και όπως έχουμε αναφέρει και στα προηγούμενα Activities πρέπει πρώτα να βρούμε τα απαραίτητα Views από το Layout που αντιστοιχεί στο Activity για να γίνει η αναφορά-Reference τους στα αντικείμενα που χρησιμοποιήσαμε στον κώδικα. Δημιουργήσαμε την μέθοδο onClickListener, να γίνεται η επιτήρηση των πατημάτων της οθόνης από το λειτουργικό σύστημα και να συσχετίσουμε τα αντίστοιχα τους Views με την μέθοδο onClickListener.

3.9.2.1 ListView

Το αντικείμενο τύπου ListView που χρησιμοποιήσαμε στο layout του Activity καταλαμβάνει την περισσότερη επιφάνειά της. Αρχικά για την χρήση και την πλήρη διαμόρφωση του ListView για τις απαιτήσεις μας χρειάζεται να δημιουργήσουμε μια ξεχωριστή κλάση-Class με την ονομασία PlantsCustomAdapter. Σε αυτή έχουμε τη δυνατότητα να επιλέξουμε την εμφάνιση και πόσα Views θα περιέχει η κάθε σειρά του αντικειμένου ListView. Για τον σκοπό αυτόν δημιουργήσαμε επιπλέον ένα layout με την ονομασία plants_list_row.xml, το οποίο περιέχει τα Views με τα οποία θα αποτελείται η κάθε σειρά και θα χρησιμοποιείται από το Class μας για την δημιουργία του ListView. Επιπλέον, το συγκεκριμένο Class δέχεται το αντικείμενο Cursor το οποίο θα περιέχει όλες τις πληροφορίες της βάσης δεδομένων μας.

3.9.2.2 Επαναχρησιμοποίηση του CalcResultActivity

Για την πλήρη προβολή της αποθηκευμένης εγκατάστασης την οποία θα επιλέξει ο χρήστης από το αντικείμενο ListView επαναχρησιμοποιήσαμε το Activity που έχουμε αναφερθεί στην ενότητα 3.4.3. Κατά την επιλογή προβολής της αποθηκευμένης εγκατάστασης από το ListView ενεργοποιείται το αντικείμενο τύπου Intent, το οποίο θα περιέχει τα δεδομένα της επιλεγμένης εγκατάστασης και επιπλέον ανακατευθύνει τον χρήστη στο Activity που επαναχρησιμοποιούμε. Για την απόκρυψη της επιλογής αποθήκευσης που περιέχει η CalcResultActivity έχουμε προσθέσει επιπλέον μια εντολή ελέγχου, ώστε να είναι σε θέση να αναγνωρίζει από πιο Activity προέρχονται τα δεδομένα που πρέπει να εμφανίσει.

3.10 **Ανάλυση του Help_InstructionsActivity**

Το παρόν Activity δημιουργήθηκε για την καθοδήγηση και ανάλυση των τεχνολογιών που χρησιμοποιεί η εφαρμογή μας για την υλοποίηση της αυτόνομης φωτοβολταϊκής μελέτης. Αρχικά, γίνεται μια συνοπτική περιγραφή για το θεωρητικό κομμάτι τους. Στην συνέχεια υπάρχει μια λεπτομερής επεξήγηση για το πως ο χρήστης μπορεί να εξακριβώσει και να βρει τις απαραίτητες τιμές που θα χρειαστεί να συμπληρώσει στα πεδία εισαγωγής της εφαρμογής μας

Εικόνα 47 Help_InstructionsActivity

3.10.1 **Γραφικό περιβάλλον του Help_InstructionsActivity**

Για το σχεδιαστικό κομμάτι του γραφικού περιβάλλοντος του συγκεκριμένου Activity χρησιμοποιήσαμε το layout με την ονομασία activity_help_instructions.xml. Αποφασίσαμε να χρησιμοποιήσουμε ως κυρίως layout το linearLayout το οποίο προβάλει το στατικό περιεχόμενο της.

3.10.1.1 Views

Το συνολικό περιεχόμενο το οποίο περιέχει το layout και πρέπει να προβληθεί στον χρήστη συνήθως θα υπερβαίνει τις διαστάσεις των περισσότερων κινητών συσκευών. Για αυτόν το λόγο χρειάστηκε να χρησιμοποιήσουμε το αντικείμενο ScrollView. Αυτό μας εξασφαλίσει την ομαλή προβολή των περιεχομένων του layout σε όλες τις συσκευές μέσω της κάθετης στήλης επιλογής κατεύθυνσης. Για την πραγματοποίηση της στοίχισης των Views τις οποίες περιέχει το ScrollView, χρησιμοποιήσουμε μέσα σε αυτό το LinearLayout, το οποίο στοιχίζει αυτόματα τα views ως προς τον κάθετο άξονα

της οθόνης. Τα views από τα οποία θα αποτελείται το LinearLayout θα είναι TextView και περιέχουν τις απαραίτητες επεξηγήσεις τις οποίες πιθανότατα θα χρειαστεί ο χρήστης.

3.10.2 Κώδικας του Help_InstructionsActivity

Στο προγραμματιστικό κομμάτι του αναφερόμενου Activity δεν χρειάζεται να συμπεριλάβουμε πολλές και ιδιαίτερες γραμμές κώδικα, καθώς σχεδιάστηκε για να λειτουργήσει σαν ένα πεδίο το οποίο θα προβάλει τις απαραίτητες επεξηγήσεις των απαιτούμενων πεδίων συμπλήρωσης, τα οποία ζητούνται από την εφαρμογή μας και τα αντίστοιχα τους παραδείγματα.

3.11 Έλεγχος λειτουργικότητας της εφαρμογής

Στο τελικό στάδιο η εφαρμογή μας μετά την υλοποίηση της, θα πραγματοποιηθούν οι βασικοί έλεγχοι οι οποίοι θα μας επιβεβαιώσουν την ορθή λειτουργία των προμελετημένων λειτουργιών της και αν αυτές οι λειτουργίες είναι συμβατές με τις προδιαγραφές που έχουμε θέσει. Για την διευκόλυνση μας, την πρόληψη δικών μας σφαλμάτων και την άμεση επιδιόρθωση των σφαλμάτων που θα προκύψουν κατά την διαδικασία ελέγχου θα γίνει διαχωρισμός των σταδίων ελέγχου της εφαρμογής σε δύο βασικές κατηγορίες, τις οποίες θα αναφέρουμε ονομαστικά και στη συνέχεια θα αναλύσουμε παρακάτω.

- **Έλεγχος των λειτουργιών της εφαρμογής**
  - Ορθή λειτουργία των δια δραστικών στοιχείων της εφαρμογής.
  - Ορθή λειτουργία των αλγόριθμων.
  - Έλεγχος λειτουργίας της βάσης δεδομένων.
- **Έλεγχος του γραφικού περιβάλλοντος**
  - Την σωστή απεικόνιση των γραφικών στοιχείων.
  - Έλεγχος των διαθέσιμων γλωσσών.

3.11.1 Ορθή λειτουργία των δια δραστικών στοιχείων της εφαρμογής

Ο συγκεκριμένος έλεγχος πραγματοποιείται πρώτα για να εξασφαλίσουμε την σωστή λειτουργία των αντικειμένων που βρίσκονται στο γραφικό περιβάλλον και αλληλοεπιδρούν με τον χρήστη. Με την προσέγγιση αυτή θα έχουμε την δυνατότητα πρόσβασης σε όλα τα Activities που περιέχει η εφαρμογή μας για να πραγματοποιήσουμε τους υπόλοιπους ελέγχους, προκειμένου να μην δημιουργηθούν σφάλματα τα οποία πηγάζουν από τα δια-δραστικά στοιχεία του γραφικού περιβάλλοντος.

Στο κάθε αντίστοιχο Activity πρώτα θα ελεγχθούν οι σύνδεσμοι που περιέχει και είναι υπεύθυνοι για την εναλλαγή μεταξύ των Activities. Τα πεδία εισαγωγής θα ελεγχθούν για την πρόληψη μη σωστά εισαγόμενων τύπων δεδομένων. Οι τιμές τους θα αποστέλλονται στις αντίστοιχες μεταβλητές του κώδικα της εφαρμογής και ταυτόχρονα θα πραγματοποιείται έλεγχος των τιμών. Τα περισσότερα Activities περιέχουν μη ορατές λειτουργίες που δυναμικά προβάλλονται κατά την απόπειρα λάθος εισαγωγής δεδομένων ή για την επιβεβαίωση μιας ενέργειας του χρήστη. Αντίστοιχα θα ελεγχθούν για την ορθή λειτουργία τους με τις απαραίτητες ενέργειες οι οποίες θα αναγκάσουν την εκκίνηση τους. Στο τέλος θα ελέγξουμε την εφαρμογή μας σε περισσότερες συσκευές οι οποίες θα φέρουν διαφορετική έκδοση Android λειτουργικού συστήματος.

3.11.2 Ορθή λειτουργία των αλγόριθμων

Η εφαρμογή μας περιέχει δύο βασικούς αλγόριθμους οι οποίοι και είναι υπεύθυνοι για την υλοποίηση της αυτόνομης φωτοβολταϊκής μελέτης και για την κατάλληλη πρόταση των αντίστοιχων συσκευών από τις οποίες θα αποτελείται το σύστημα. Είναι σημαντικό να τονίσουμε ότι για δική μας ευκολία υλοποιήθηκαν οι παρόντες αλγόριθμοι πρώτα σε ξεχωριστό σχεδιαστικό πρόγραμμα ανάπτυξης λογισμικού, το Eclipse IDE for Java. Με αυτήν την προσέγγιση ήμασταν σε θέση να παρακάμψουμε το γραφικό περιβάλλον της εφαρμογής και να επικεντρωθούμε στην ανάπτυξη και να πραγματοποιήσουμε στους αντιστοίχους ελέγχους. Για τον παρόντα έλεγχο προϋπολογίσαμε διαφορετικά σενάρια απαιτούμενων αναγκών ενός αυτόνομου φωτοβολταϊκού συστήματος. Τα σενάρια αποτελούνται τόσο από μικρές μελέτες όσο και από μεγάλες φωτοβολταϊκές μελέτες. Επίσης, συμπεριλάβαμε και λανθασμένες μελέτες, ώστε να επιβεβαιώσουμε

την ορθή λειτουργία των προμελετημένων λειτουργιών τους και αν αυτές οι λειτουργίες τους είναι συμβατές με τις προδιαγραφές που έχουμε θέσει.

Μετά την ολοκλήρωση των αρχικών ελέγχων ενσωματώσαμε τους αλγόριθμους στον κώδικα της εφαρμογής μας, και επαναλάβαμε την ίδια διαδικασία στην οποία αναφερθήκαμε προηγουμένως, ώστε να διασφαλίσουμε την ορθή λειτουργία τους. Τα δεδομένα αυτήν την φορά όμως θα προέρχονται από την αλληλεπίδραση του χρήστη με τα δια-δραστικά στοιχεία του γραφικού περιβάλλοντος και τα αντίστοιχα δια-δραστικά στοιχεία με τον κώδικα της εφαρμογής. Σε αυτό το στάδιο ελέγχου τα σφάλματα τα οποία μπορεί να προκύψουν δεν προέρχονται απαραίτητα μόνο από τους αλγόριθμους, αλλά και από το σχεδιασμό αλληλεπίδρασης των δια-δραστικών στοιχείων με τον κώδικα της εφαρμογής. Για τον λόγο αυτόν εξετάσαμε πρώτα την ορθή λειτουργία των δια-δραστικών στοιχείων της εφαρμογής, όπως προαναφέραμε για να μειώσουμε τα περισσότερα τυχόν σφάλματα που μπορεί να προκύψουν από αυτά.

3.11.3 Έλεγχος ορθής λειτουργίας της βάσης δεδομένων

Η εφαρμογή μας περιέχει μια τοπική βάση δεδομένων η οποία έχει σχεδιαστεί για την αποθήκευση των υπολογισμένων αυτόνομων φωτοβολταϊκών μελετών του χρήστη. Για την ορθή λειτουργία της δημιουργήσαμε τρία σενάρια αυτόνομων φωτοβολταϊκών μελετών από τις οποίες θα κρατήσουμε τα αντίστοιχα αποτελέσματα για την επιβεβαίωση της ορθής λειτουργίας της.

Ο πρώτος έλεγχος με τον οποίο θα ξεκινήσουμε και χρησιμοποιώντας τα προ-υπολογισμένα σενάρια θα είναι η επιβεβαίωση της ορθή εισαγωγής των δεδομένων σε αυτήν, ταυτόχρονα θα ελεγχθεί και η αποτροπή εισαγωγής μη εξουσιοδοτημένων τύπων δεδομένων. Ο επόμενος έλεγχος, εφόσον αποθηκευτούν τα δεδομένα χωρίς σφάλματα, θα είναι η ορθή εξαγωγή των δεδομένων στην αντίστοιχη Activity η οποία είναι υπεύθυνη για την απεικόνιση των δεδομένων της αυτόνομης φωτοβολταϊκής μελέτης που έχει επιλέξει ο χρήστης.

3.11.4 Την σωστή απεικόνιση των γραφικών στοιχείων

Μετά την ολοκλήρωση των ελέγχων ορθής λειτουργικότητας της εφαρμογής θα επικεντρωθούμε στο γραφικό περιβάλλον του χρήστη. Θα ξεκινήσουμε με τον έλεγχο των αντικειμένων τύπου Views που περιέχει η κάθε αντίστοιχη Activity, τα οποία είναι και

τα κυρίως υπεύθυνα αντικείμενα απεικόνισης των στοιχείων που περιέχει το γραφικό περιβάλλον. Εξετάστηκαν για τη σωστή διαστασιολόγηση που πρέπει να φέρουν και τη σωστή τοποθέτησή τους σε σχέση με τα υπόλοιπα Views. Οι γραμματοσειρές που περιέχουν να μην αλλοιώνονται και το μέγεθός τους να είναι το κατάλληλο σε σχέση με το View που το περιέχει. Αντίστοιχα σε κάθε Activity ελέγχθηκε και ο προσανατολισμός της οθόνης άμα τηρείται. Τέλος ελέγξαμε την εφαρμογή μας σε περισσότερες συσκευές, οι οποίες φέρουν διαφορετικό μέγεθος οθόνης ώστε να επιβεβαιώσουμε την ορθή λειτουργία του γραφικού περιβάλλοντος.

3.11.5 Έλεγχος των διαθέσιμων Γλωσσών

Η εφαρμογή μας περιέχει δύο βασικές γλώσσες, οι οποίες μπορούν να προβληθούν μέσω του γραφικού περιβάλλοντος, οι οποίες είναι η Ελληνική και η Αγγλική. Η γλώσσα που θα επιλέξει η εφαρμογή και θα προβάλλεται εξαρτάται από τη γλώσσα του λειτουργικού συστήματος. Για την ορθή λειτουργία αυτού του χαρακτηριστικού θα αλλάξουμε την γλώσσα του λειτουργικού συστήματος και για την επιτυχή ρύθμισή της θα πραγματοποιηθεί επανεκκίνηση της εφαρμογής μας . Αντίστοιχα θα πρέπει να γίνει έλεγχος σε κάθε Activity για την απεικόνιση της γλώσσας που θα πρέπει να περιέχει. Όπως έχουμε προαναφέρει τα περισσότερα Activities περιέχουν μη ορατές λειτουργίες που δυναμικά προβάλλονται κατά την απόπειρα λάθους εισαγωγής δεδομένων ή για την επιβεβαίωση μιας ενέργειας του χρήστη. Εκτός από τα παραπάνω θα πρέπει να πραγματοποιηθεί και ορθογραφικός έλεγχος όλων των Activities.

4 Εγχειρίδιο χρήσης

Η Εφαρμογή έχει σχεδιαστεί έχοντας ως στόχο την κάλυψη σε έναν βασικό βαθμό των απαραίτητων υπολογισμών για την υλοποίηση μίας φωτοβολταϊκής μελέτης. Στην παρακάτω ενότητα θα περιγράψουμε τη χρήση και τις λειτουργίες της εφαρμογής στην πράξη.

Εικόνα 48 Categories

Στην αρχική οθόνη της εφαρμογής ο χρήστης θα έχει την δυνατότητα επιλογής ανάμεσα σε τρείς κατηγορίες:

- Υπολογισμός συστήματος
- Αποθηκευμένα συστήματα
- Οδηγίες-Βοήθεια

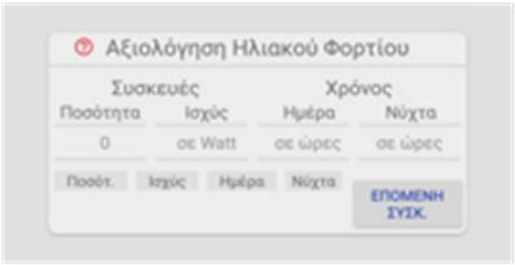
Η πρώτη κατηγορία θα είναι υπεύθυνη για την υλοποίηση της αυτόνομης φωτοβολταϊκής μελέτης. Η επόμενη κατηγορία θα παρέχει στο χρήστη την αναθεώρηση των είδη αποθηκευμένων φωτοβολταϊκών μελετών του. Η τρίτη και τελευταία κατηγορία θα περιέχει τις απαραίτητες επεξηγήσεις και παραδείγματα για την βοήθεια και καθοδήγηση του χρήστη.

4.1 Υπολογισμός συστήματος

Επιλέγοντας ο χρήστης την πρώτη κατηγορία για τον υπολογισμό της επιθυμητής αυτόνομης φωτοβολταϊκής μελέτης, θα μεταβεί στο αντίστοιχο Activity το οποίο θα περιέχει τα απαραίτητα πεδία συμπλήρωσης.

Εικόνα 49 Locations

Στο πρώτο πεδίο ο χρήστης θα μπορεί να επιλέξει την τοποθεσία στην οποία θα βρίσκεται η φωτοβολταϊκή εγκατάσταση. Με την επιλογή αυτή θα είναι σε θέση ο αλγόριθμος για τον υπολογισμό της μελέτης να αντιστοιχεί την ακριβή μέση ημερήσια ενεργειακή απολαβή της κάθε περιοχής, ώστε να πραγματοποιείται πιο αποδοτικά ο υπολογισμός των απαραίτητων φωτοβολταϊκών συλλεκτών από την οποία θα αποτελείται η εγκατάσταση.



Εικόνα 50 Solar Evaluation

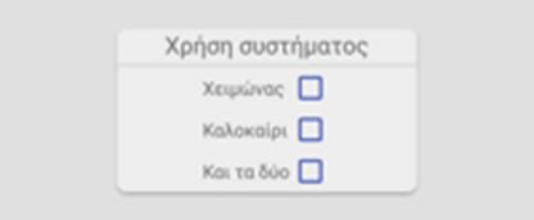
Το αμέσως επόμενο πεδίο στοχεύει στον υπολογισμό της συνολικής απαιτούμενης ισχύος που θα καταναλώνεται ημερησίως από την αυτόνομη φωτοβολταϊκή εγκατάσταση. Για να πραγματοποιηθεί αυτό είναι απαραίτητο ο χρήστης να εισάγει λεπτομερώς την κάθε συσκευή που σκοπεύει να χρησιμοποιήσει, δηλώνοντας τα ζητούμενα τεχνικά χαρακτηριστικά της αντίστοιχης συσκευής και καταχωρώντας την με το κουμπί που φέρει την ονομασία ΕΠΟΜΕΝΗ ΣΥΣΚ. Μετά την καταχώρηση τους θα μηδενίζονται τα πεδία συμπλήρωσης για την επαναχρησιμοποίηση τους. Οι καταχωρημένες συσκευές θα απεικονίζονται στον πίνακα που υπάρχει κάτω από τα πεδία συμπλήρωσης για την διευκόλυνση του χρήστη. Τα πεδία εισαγωγής είναι τέσσερα στα οποία ζητείται ο συνολικός αριθμός των τεμαχίων που θα χρησιμοποιήσει για την κάθε συσκευή. Αντίστοιχα θα ζητείται η ονομαστική ισχύς της συσκευής και τέλος τον χρόνο λειτουργίας την ημέρα και την νύχτα.

Εικόνα 51 Ημέρες Αυτονομίας

Στο επόμενο πεδίο εισαγωγής ο χρήστης καλείται να επιλέξει των αριθμό των διαδοχικών ημερών χρήσης του αυτόνομου φωτοβολταϊκού συστήματος σε μη επαρκή ηλιοφάνεια. Οι ημέρες θα είναι απαραίτητα τουλάχιστον δυο, προκειμένου να επιτευχθεί η ορθή λειτουργία του συστήματος.

Εικόνα 52 System Voltage

Στο επόμενο πεδίο εισαγωγής ο χρήστης θα έχει την δυνατότητα να εκτιμήσει την τάση με την οποία θα δουλεύει το σύστημα που θέλει να υπολογίσει. Κατά τον υπολογισμό της μελέτης θα ελέγχεται η τάση που έχει εισάγει ο χρήστης και εφόσον είναι στα επιθυμητά όρια των προδιαγραφών του συστήματος, αυτό δεν θα τροποποιείται αυτόματα.



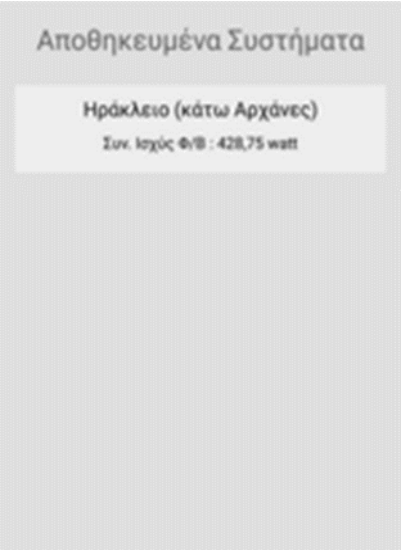
Εικόνα 53 Seasons

Επιπροσθέτως, ο χρήστης απαιτείται να επιλέξει την εποχή που το σύστημα θα είναι ενεργό και θα παράγει ενέργεια, ώστε να εξακριβώνεται η μέση ημερήσια ενεργειακή απολαβή.

Εικόνα 54 Result Activity

4.2 Αποθηκευμένα συστήματα

Ο χρήστης μετά την εισαγωγή των απαραίτητων τιμών που ζητούνται θα μεταβεί στο επόμενο Activity, το οποίο θα είναι υπεύθυνο για την προβολή της υλοποιημένης αυτόνομης φωτοβολταϊκής μελέτης. Αυτή θα αποτελείται από τις τέσσερις κατηγορίες που θα περιέχουν τα αντίστοιχα βασικά τεχνικά χαρακτηριστικά των εξαρτημάτων που πρέπει να πληρούν και την πρόταση της κατάλληλης συσκευής με τον αριθμών τεμαχίων της. Ο χρήστης επιπλέον θα μπορεί να αποθηκεύσει την υπολογισμένη μελέτη εισάγοντας την επιθυμητή ονομασία που θα φέρει η μελέτη ή και να μεταβεί στην αρχική οθόνης της εφαρμογής κάνοντας χρήση των αντίστοιχων εικονιδίων.



Εικόνα 55 Plants

Επιλέγοντας ο χρήστης την δεύτερη κατηγορία από το αρχικό Activity, η οποία θα είναι υπεύθυνη για την απεικόνιση όλων των υπολογισμένων αυτόνομων φωτοβολταϊκών μελετών του, θα μεταβεί στο αντίστοιχο Activity. Αυτό θα περιέχει μια λίστα με όλες τις αποθηκευμένες μελέτης σε μορφή συνδέσμων, οι οποίες θα προβάλλονται με το χαρακτηριστικό τίτλο που έχει αποθηκεύσει ο χρήστης την μελέτη. Πατώντας ο χρήστης την επιθυμητή μελέτη θα μεταβεί στο αντίστοιχο Activity, το οποίο θα προβάλει τα δεδομένα της μελέτης. Επίσης με το παρατεταμένο πάτημα του συνδέσμου που εκπροσωπεί την μελέτη που περιέχει η λίστα, θα γίνεται η διαγραφή της από την βάση δεδομένων και ταυτόχρονα και από την ίδια τη λίστα.

4.3 Οδηγίες-Βοήθεια

Εικόνα 56 Instructions Activity

Ο χρήστης επιλέγοντας την τελευταία κατηγορία που περιέχει το αρχικό Activity, θα μεταβεί στο αντίστοιχο Activity το οποίο θα περιέχει αναλυτικές πληροφορίες για το κάθε πεδίο εισαγωγής που απαιτείται ο χρήστης να συμπληρώσει για την υλοποίηση την αυτόνομης φωτοβολταϊκής μελέτης. Αρχικά θα γίνεται μια συνοπτική περιγραφή για το θεωρητικό κομμάτι τους. Στην συνέχεια θα υπάρχει μια λεπτομερής επεξήγηση σχετικά με το πως ο χρήστης θα μπορεί να εξακριβώσει και να βρει τις απαραίτητες τιμές που θα χρειαστεί να συμπληρώσει στα πεδία εισαγωγής της εφαρμογής μας.

5 Συμπεράσματα

Στόχος της πτυχιακής είναι η σχεδίαση και η υλοποίηση μια πλήρως λειτουργικής εφαρμογής για smartphone συσκευές. Ο κύριος σκοπός της πτυχιακής εργασίας είναι η εκπόνηση μιας αυτόνομης φωτοβολταϊκής μελέτης καθώς και η παράθεση προτάσεων για κατάλληλες συσκευές, σε συνδυασμό με τα αντίστοιχα τεχνικά χαρακτηριστικά που θα πρέπει να πληροί η κάθε συσκευή. Τα στοιχεία που απαιτούνται εισάγονται από το χρήστη και για κάθε πεδίο που πρέπει να συμπληρωθεί υπάρχει αναλυτική επεξήγηση και βοήθεια. Με την δημιουργία μιας βάσης δεδομένων που υπάρχει και λειτουργεί τοπικά στην συσκευή, θα δίνεται η επιλογή αποθήκευσης της κάθε υπολογισμένης μελέτης για μελλοντική προεπισκόπηση.

Μέσω του διαδικτύου η αγορά προϊόντων έχει γίνει ευκολότερη και πιο προσβάσιμη στον μέσο καταναλωτή, ο οποίος έχει τη δυνατότητα επιλογής της τιμής, της ποιότητας και της χώρας προέλευσης της αρεσκείας του, χωρίς να είναι απαραίτητο να εμπλακούν επιπλέον πωλητές στην διαδικασία αγοράς. Σε πολλές περιπτώσεις όμως, υπάρχουν συστήματα τα οποία αποτελούνται από μεγάλο αριθμό προϊόντων και απαιτούνται εξειδικευμένες γνώσεις για τη σωστή επιλογή τους και τη σωστή λειτουργίας τους. Σε αυτές τις περιπτώσεις είναι απαραίτητο για τον μέσο καταναλωτή η ύπαρξη ενός επιπλέον πωλητή. Έχοντας τις απαραίτητες γνώσεις, ο πωλητής είναι σε θέση να προτείνει τα κατάλληλα προϊόντα που θα ικανοποιούν τις ανάγκες των καταναλωτών. Εξίσου για τον υπολογισμό και τη σωστή γνώση των απαραίτητων τεχνικών χαρακτηριστικών των προϊόντων για την σωστή λειτουργία ενός αυτόνομου φωτοβολταϊκού συστήματος τις πιο πολλές φορές χρειάζεται έναν επιπλέον πωλητής. Η εφαρμογή αναλαμβάνει το ρόλο αυτόν και θα προσφέρει τις απαραίτητες πληροφορίες, ώστε ο καταναλωτής να μην εξαρτάται από επιπλέον πωλητές και να διαμορφώνει μόνος του μια σωστή γενική ιδέα σχετικά με τις ανάγκες του. Ταυτόχρονα, η εφαρμογή μπορεί να εξυπηρετήσει και αυτούς που έχουν τις απαραίτητες γνώσεις πάνω στο συγκεκριμένο αντικείμενο που εξετάζουμε, καθώς αναλαμβάνει τους υπολογισμούς και δίνει τη δυνατότητα αποθήκευσης των συστημάτων για μελλοντική επισκόπηση.

Σήμερα ένα από τα μεγάλα θέματα για το μέλλον της ανθρωπότητας είναι και το θέμα του περιβάλλοντος, στο οποίο οι παρεμβάσεις του ανθρώπου, έχουν δημιουργήσει τεράστια προβλήματα στο φυσικό περιβάλλον. Η ενεργοβόρα δομή παραγωγής, ταυτόχρονα η μη συνειδητοποιημένη χρήση της ενέργειας έχουν οδηγήσει σε μείωση των αποθεμάτων ενεργειακών πόρων. Οι ανανεώσιμες πηγές ενέργειας μπορούν να συμβάλλουν στον περιορισμό της ρύπανσης του φυσικού περιβάλλοντος από την παραγωγή ενέργειας και σε μια τέτοια προσπάθεια η Ευρωπαϊκή Ένωση έχει θέσει ως στόχο της για το 2022 το 30% της κατανάλωσης ενέργειας να προέρχεται από ανανεώσιμες πηγές. Ο ήλιος εκπέμπει τεράστια ποσότητα ενέργειας ημερησίως, για αυτό οι τεχνολογίες οι οποίες αξιοποιούν την ηλιακή ενέργεια θεωρούνται από τις πιο διαδεδομένες και ραγδαία αναπτυσσόμενες τεχνολογίες και μπορούν να χρησιμοποιηθούν σε όλα τα φάσματα.

Όπως είναι προφανές, στην εφαρμογή μπορούν να πραγματοποιηθούν ακόμα πολλές προσθήκες, βελτιώσεις ακόμα και η ενσωμάτωση επιπλέον λογισμικού και τεχνολογιών. Ορισμένα τέτοια παραδείγματα σε επόμενο στάδιο της εφαρμογής θα μπορούσαν να είναι :

- Να υποστηρίζεται και από άλλα λειτουργικά συστήματα τόσο κινητών συσκευών, όσο και από λειτουργικά συστήματα σταθερών και φορητών υπολογιστών.
- Προσθήκη περισσότερων γλωσσών.
- Επιλογή τροποποίησης των ήδη υπολογισμένων αυτόνομων φωτοβολταϊκών μελετών, οι οποίες βρίσκονται στο χαρτοφυλάκιο της εφαρμογής.
- Ενσωμάτωση backup επιλογής για τη διατήρηση των υπολογισμένων αυτόνομων φωτοβολταϊκών μελετών, σε περίπτωση αλλαγής συσκευής.

Αναφορές

William L. Hosch, Mark Hall (03.10.2016) BRITANNICA. Google Inc. https://www.britannica.com/topic/Google-Inc (τελευταία πρόσβαση 15 Αυγ 2018).

J.mercer (05.02.2018). KETTLEMAG. The history of the Java programming language. https://www.kettlemag.co.uk/the-history-of-the-java-programming-language/ (τελευταία πρόσβαση 17 Αυγ 2018).

Android Developer (09.12.2018). DEVELOPER ANDROID. Meet Android Studio. https://developer.android.com/studio/intro/ (τελευταία πρόσβαση 20 Αυγ 2018).

Christian de Looper (23.10.2018). DIGITALTRENDS. From Android 1.0 to Android 9.0, here's how Google's OS evolved over a decade. https://www.digitaltrends.com/mobile/android-version-history/ (τελευταία πρόσβαση 25 Αυγ 2018).

Android Developer (29.02.2018). DEVELOPER ANDROID. Android Studio. https://developer.android.com/studio/ (τελευταία πρόσβαση 30 Αυγ 2018).

Alex Mullis (11.11.2017). ANDROIDAUTHORITY. Android Studio. tutorial http://www.androidauthority.com/android-studio-tutorial-beginners-637572 (τελευταία πρόσβαση 04 Σεπτ 2018).

Vaibhav Bajpai (08.03.2017). GEEKSFOGEEKS. Components of an Android Application. https://www.geeksforgeeks.org/components-android-application/ (τελευταία πρόσβαση 15 Σεπτ 2018).

Android Developer (06.06.2018). DEVELOPER ANDROID. Handling Lifecycles with Lifecycle-Aware Components. https://developer.android.com/topic/libraries/architecture/lifecycle/ (τελευταία πρόσβαση 19 Σεπτ 2018).

Alyssa Baker (05.04.2017). SOLARPOWER. A History of Solar Cells: How Technology Has Evolved. https://www.solarpowerauthority.com/a-history-of-solar-cells (τελευταία πρόσβαση 30 Σεπτ 2018).

Android Developer (06.06.2018). DEVELOPER ANDROID. Layouts. https://developer.android.com/guide/topics/ui/declaring-layout (τελευταία πρόσβαση 04 Ιαν 2019).

Android Developer (06.06.2018). DEVELOPER ANDROID. View. https://developer.android.com/reference/android/view/View (τελευταία πρόσβαση 05 Ιαν 2019).

Βιβλιογραφία

Φραγκιαδάκης, Ι. Ε. (2009) Ενεργειακό ισοζύγιο ημερησίως παραγόμενης-καταναλισκόμενης ηλεκτρικής ενέργειας, 281.

Φραγκιαδάκης, Ι. Ε. (2009) Υπολογισμός της απαιτούμενης χωρητικότητας της συστοιχίας συσσωρευτών ενός αυτόνομου ΦΒ συστήματος, 305.