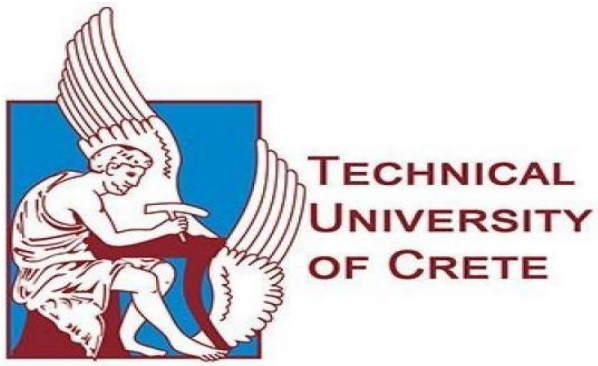

Πολυτεχνείο Κρήτης

Τμήμα Ηλεκτρολόγων Μηχανικών Και

Μηχανικών Υπολογιστών

AQUACOUNT :

Ανάπτυξη Κινητού Πληροφοριακού Συστήματος με Υποστήριξη Υπηρεσιών Θέσης για το Βέλτιστο Έλεγχο και Διαχείριση Κατανάλωσης Υπηρεσιών Κοινής Ωφέλειας

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Του

Τσίρτση Σπυρίδωνος

Επιβλέπων Καθηγητής

Γαροφαλάκης Μίνως

Εξεταστική Επιτροπή

Γαροφαλάκης Μίνως ,

Χαλκιαδάκης Γεώργιος ,

Δεληγιαννάκης Αντώνιος

Abstract

Η παρούσα εργασία εξετάζει τον σχεδιασμό και την υλοποίηση ενός προηγμένου συστήματος υδρομετρήσεων με στόχο την ακριβή και αξιόπιστη καταγραφή των μετρήσεων κατανάλωσης νερού. Η ανάλυση απαιτήσεων καλύπτει τόσο τις λειτουργικές όσο και τις μη λειτουργικές πτυχές του συστήματος.

Στις λειτουργικές απαιτήσεις, περιλαμβάνονται η ανάγκη για ακριβή ανίχνευση των μετρήσεων, ασφαλείς συνδέσεις χρηστών και η δυνατότητα παρακολούθησης σε πραγματικό χρόνο. Επιπλέον, εξετάζεται η δυνατότητα εκτέλεσης μετρήσεων σε offline καταστάσεις με αυτόματη μεταφορά δεδομένων κατά την επανασύνδεση.

Στον τομέα των μη λειτουργικών απαιτήσεων, δίνεται έμφαση στην ασφάλεια των δεδομένων με κρυπτογράφηση επικοινωνιών και εξουσιοδότηση μέσω μοναδικών κλειδιών. Το περιβάλλον του χρήστη θα πρέπει να είναι ευανάγνωστο και φιλικό, ενώ το σύστημα θα πρέπει να είναι συμβατό με διάφορες πλατφόρμες κινητών τηλεφώνων.

Αυτή η έρευνα ανοίγει τον δρόμο για την ανάπτυξη προηγμένων συστημάτων υδρομετρήσεων που θα πληρούν υψηλές προδιαγραφές απόδοσης και ασφάλειας.

Ευχαριστίες

Θα ήθελα να εκφράσω τις ειλικρινείς μου ευχαριστίες προς όλους όσους συνέβαλαν στην υλοποίηση και επιτυχή ολοκλήρωση αυτής της διατριβής.

Αρχικά θέλω να ευχαριστήσω τον επιβλέποντα μου κ. Γαροφαλάκη Μίνω για την ευκαιρία που μου έδωσε να αναλάβει την πτυχιακή μου , όπως επίσης και τον κ. Νικόλαο Γιατράκο για τις πολύτιμες υποδείξεις και τη καθοδήγηση κατά τη διάρκεια της εκπόνησης της διπλωματικής μου εργασίας. Θερμές ευχαριστίες στην επιτροπή αξιολόγησης αυτής της εργασίας και στους κ. Χαλκιαδάκη και Δεληγιαννάκη για τον πολύτιμο χρόνο που αφιέρωσαν και τα χρήσιμα σχόλιά τους.

Ακόμη οφείλω ένα μεγάλο ευχαριστώ στην οικογένεια μου για την αμέριστη υποστήριξη τους καθ’ όλη την διάρκεια των σπουδών μου.

Περιεχόμενα

Εισαγωγή

Η ταχεία εξέλιξη της τεχνολογίας και η διαρκής ψηφιακή εξέλιξη έχουν ανοίξει νέους ορίζοντες και έχουν αναδιαμορφώσει τον τρόπο που αλληλεπιδρούμε με τον κόσμο γύρω μας. Στο επίκεντρο της ραγδαίας αυτής μετασχηματικής διαδικασίας παρουσιάζεται η ανάγκη για νέες, έξυπνες λύσεις που να λύνουν προβλήματα της καθημερινότητας μας.

Η παρούσα πτυχιακή εργασία επικεντρώνεται στην τεχνολογική εξέλιξη των δημόσιων δομών και υπηρεσιών και συγκεκριμένα στο κομμάτι της υδρομέτρησης προσφέροντας μια προηγμένη και ολοκληρωμένη λύση για τη διαχείριση της καταμέτρησης και της κοστολόγησης του ύδατος.

Στόχος είναι να δημιουργηθεί ένα ευφυές σύστημα παρακολούθησης που όχι μόνο θα παρέχει ακριβείς μετρήσεις, αλλά επίσης θα εξασφαλίζει την εύκολη πρόσβαση, την ενημέρωση και τον αποτελεσματικό έλεγχο της κατανάλωσης των υδάτινων πόρων.

Είναι ανάγκη πλέον, να συμπορεύσουμε με την τεχνολογία και μέσω εκείνης να καταφέρουμε να απλοποιήσουμε τις δημόσιες δομές μας , να τις ελαφρύνουμε και να τις απλοποιήσουμε με γνώμονα την ακρίβεια και τις αυτοματοποιημένες λειτουργίες που μπορεί να μας προσφέρει η τεχνολογία.

Στο πλαίσιο αυτής της εργασίας συνδέονται ουσιαστικά ο κόσμος της τεχνολογίας και των δημόσιων δομών προωθώντας μια προσεκτική και αποδοτική χρήση των υδάτινων δεδομένων.

Στο κεφάλαιο που ακολουθεί θα εξετάσουμε λεπτομερώς τις απαιτήσεις του συστήματος και τον τρόπο με τον οποίο η εφαρμογή ανταποκρίνεται σε αυτές, παρέχοντας μια συνολική εικόνα της λειτουργικότητας και της σχεδίασης της εφαρμογής.

1.2 Επιχειρησιακές Διαδικασίες

Οι υδρομετρήσεις αποτελούν κρίσιμο κομμάτι της διαχείρισης των υδάτινων πόρων. Η παρούσα ενότητα εισάγει τον τρόπο λειτουργίας του συστήματος υδρομετρήσεων μέχρι στιγμής και επίσης τον τρόπο που προτείνεται από αυτή την εργασία, ενσωματώνοντας σύγχρονες τεχνολογίες για την πολυδιάστατη παρακολούθηση και τον έλεγχο της κατανάλωσης, προσδίδοντας έμφαση στα βασικά στοιχεία και τις λειτουργίες που καθιστούν αυτό το σύστημα κρίσιμο και χρήσιμο για τη διαχείριση των υδάτινων πόρων.

Συγκεκριμένα, το σύστημα υδρομέτρησης αποτελείται από κάποιους βασικούς πυλώνες. Αυτοί είναι οι **διαδρομές** οι οποίες αποτελούνται από **ρολόγια** μέτρησης κατανάλωσης**,** τα οποία με τη σειρά τους χαρακτηρίζονται από **μετρήσεις.** Ο τέταρτος πυλώνας είναι οι **καταμετρητές.** Οι καταμετρητές είναι ανθρώπινοι πράκτορες που αναλαμβάνουν μία διαδρομή και επιφορτίζονται με την καταχώρηση των μετρήσεων που χαρακτηρίζει κάθε ρολόι της διαδρομής αυτής.

Η διαδικασία της καταμέτρησης έχει ως εξής :

1. Στον κάθε καταμετρητή ανατίθεται μια διαδρομή.
2. Ο καταμετρητής έχει υποχρέωση να δει τα ρολόγια που έχει η συγκεκριμένη διαδρομή και να αρχίσει την καταμέτρηση τους με βάση την τοποθεσία τους.
3. Στην καταμέτρηση, ο καταμετρητής αφού έχει φτάσει στο ρολόι, βλέπει το χαρακτηριστικό του ρολογιού(id) και καταγράφει την μέτρηση του.
4. Καταγράφεται η μέτρηση, η ημερομηνία και η ώρα που καταχωρήθηκε.
5. Έπειτα, στο τέλος της διαδρομής που του έχει ανατεθεί, όλες οι μετρήσεις του έχουν καταγραφεί και τότε πρέπει να παραδώσει την συσκευή καταχώρησης δεδομένων υδρομέτρησης στο αρμόδιο τμήμα , ώστε να πάρουν τις μετρήσεις χειροκίνητα από την συσκευή και να τις περάσουν στον κεντρικό server.

1.3 Εισαγωγή στο νέο τρόπο λειτουργίας του Συστήματος Υδρομετρήσεων

Στο υπάρχον σύστημα η διαδικασία ταυτοποίησης των χρηστών δεν είναι επαρκής. Με μία πιο ολοκληρωμένη διαδικασία ταυτοποίησης θα μπορούσε να βελτιωθεί αλλά και να αυτοματοποιηθεί η διαδικασία μοιράσματος των διαδρομών σε καταμετρητές και επίσης να εξασφαλιστεί η ακεραιότητα της.

Επίσης από τη στιγμή που ο κάθε χρήστης θα έχει ένα μοναδικό προσδιοριστή θα είναι πιο εύκολος ο έλεγχος και η παρακολούθηση των ενεργειών του.

Στο υπάρχον σύστημα, επίσης, η διαδικασία μεταφοράς των δεδομένων (μετρήσεων) γίνεται χειροκίνητα, δηλαδή ο κάθε καταμετρητής πρέπει να παραδώσει το tablet, στο οποίο έχει κάνει τις μετρήσεις του στο αρμόδιο τμήμα. Εκείνο με τη σειρά του πρέπει να κάνει εξαγωγή (export) ένα αρχείο και να το περάσει στον κεντρικό server.

Αυτό είναι αρκετά χρονοβόρο, αλλά και επιρρεπές σε λάθη καθώς είναι μια μη αυτοματοποιημένη διαδικασία στην οποία πρέπει να εμπλακούν πολλά τμήματα μεταξύ τους έτσι ώστε να καταχωρηθούν οι μετρήσεις στο σύστημα. Επομένως πρέπει να δημιουργηθεί ένα αυτοματοποιημένο σύστημα μεταφοράς δεδομένων.

Κατά την διάρκεια των μετρήσεων, ο καταμετρητής ξέρει τις διευθύνσεις των ρολογιών αλλά δεν τις έχει όλες εντυπωμένες στον χάρτη της περιοχής που βρίσκεται κάθε φορά. Έτσι, πρέπει ξεχωριστά να ψάχνει την κάθε μια.

Τέλος αυτή τη στιγμή δεν υπάρχει ένα admin dashboard , δηλαδή ένα εύκολο σύστημα διαχείρισης στο οποίο να μπορεί να εισέλθει ο διαχειριστής και να δει χρήσιμες και συγκεντρωτικές πληροφορίες για τις μετρήσεις , τα ρολόγια , τις διαδρομές και τους καταμετρητές. Μάλιστα μέσω αυτού , όχι μόνο να βλέπει αναφορές, αλλά να μπορεί να επεξεργαστεί και τα δεδομένα.

Με βάση τις παραπάνω πρακτικές ανάγκες που προκύπτουν δημιουργήθηκαν τα εξής σενάρια χρήσης και λειτουργικότητας:

Σύνδεση Χρήστη και Ασφάλεια :

Η διασύνδεση του συστήματος με τα στοιχεία σύνδεσης χρηστών είναι ουσιώδης για την ασφάλεια και τον έλεγχο πρόσβασης. Μέσω της διαδικασίας σύνδεσης, οι χρήστες (διαχειριστές ή καταμετρητές) αποκτούν πρόσβαση σε εξειδικευμένες λειτουργίες με βάση το ρόλο τους, ενώ εξασφαλίζεται η προστασία των ευαίσθητων δεδομένων.

Αναγνώριση χρήστη και έλεγχος εγκυρότητας :

Το σύστημα υλοποιεί ασφαλείς μεθόδους αναγνώρισης χρήστη, με χρήση κλειδιού(token). Αυτό εξασφαλίζει πως μόνο εξουσιοδοτημένοι χρήστες έχουν πρόσβαση σε ευαίσθητες λειτουργίες και πληροφορίες.

Με τον συγκεκριμένο τρόπο, η εφαρμογή επιτυγχάνει όχι μόνο την οπτική παρακολούθηση και διαχείριση των υδρομέτρων, αλλά και την ασφαλή και ευέλικτη σύνδεση των χρηστών, ενισχύοντας τη διαφάνεια και τον έλεγχο του συστήματος.

Ανίχνευση και Συλλογή Δεδομένων :

Η ανίχνευση και συλλογή δεδομένων αποτελεί τον πυρήνα του συστήματος υδρομετρήσεων, διασφαλίζοντας τη συνεχή και ακριβή μέτρηση της κατανάλωσης νερού. Τα υδρόμετρα εξοπλίζονται με προηγμένα αισθητήρια για την ακριβή μέτρηση του όγκου νερού που διακινείται μέσω τους. Τα δεδομένα αυτά συλλέγονται σε πραγματικό χρόνο, διευκολύνοντας την άμεση παρακολούθηση της κατανάλωσης και τον εντοπισμό πιθανών προβλημάτων.

Μεταφορά Δεδομένων :

Τα δεδομένα που συλλέγονται από τα υδρόμετρα μεταφέρονται σε πραγματικό χρόνο σε ένα κεντρικό σύστημα. Η μεταφορά μπορεί να πραγματοποιηθεί μέσω ασύρματων δικτύων επικοινωνίας, εξασφαλίζοντας άμεση πρόσβαση στα δεδομένα.

Αποθήκευση και Διαχείριση Δεδομένων :

Τα δεδομένα που συλλέγονται από τα υδρόμετρα αποθηκεύονται σε ασφαλείς βάσεις δεδομένων. Η διαδικασία αποθήκευσης εξασφαλίζει την ακεραιότητα και την προσβασιμότητα των δεδομένων. Ταυτόχρονα, η διαχείριση των δεδομένων

περιλαμβάνει την οργάνωση τους για να επιτρέψει εύκολη ανάκτηση και ανάλυση.

Διαδικασία Ενημέρωσης Μετρήσεων :

Η διαδικασία ενημέρωσης μετρήσεων είναι κρίσιμη για τη διατήρηση της ακρίβειας των δεδομένων. Το σύστημα αναλαμβάνει την χειροκίνητη ενημέρωση των μετρήσεων από τους καταμετρητές, εξασφαλίζοντας την αποφυγή λαθών και πως οι πληροφορίες που παρέχονται στο σύστημα είναι πάντα ενημερωμένες και αξιόπιστες.

Χαρτογράφηση Ρολογιών :

Το σύστημα επιτρέπει τη χαρτογράφηση κάθε υδρόμετρου σε γεωγραφικά δεδομένα, επιτρέποντας την εύκολη αντίληψη της κατανομής τους στο χώρο. Μέσω της γεωγραφικής απεικόνισης, οι διαχειριστές μπορούν να παρατηρούν την τοποθεσία και την συγκέντρωση των υδρομέτρων επιτρέποντας πιο αποτελεσματική ανάθεση διαδρομών με ρολόγια στους καταμετρητές. Ταυτόχρονα, κατά τον υπολογισμό των βέλτιστων διαδρομών για να πάει ο καταμετρητής σε κάθε ρολόι, όλες αυτές οι διαδρομές θα ξεκινούν από ένα συγκεκριμένο σημείο, που θα έχει ορίσει ο χρήστης σαν την τοποθεσία του.

Dashboard Διαχειριστή :

Το Dashboard του διαχειριστή αποτελεί τον εγγενή ελεγκτικό πύργο του συστήματος. Εδώ, ο διαχειριστής έχει πρόσβαση σε όλες τις πληροφορίες του συστήματος. Έχει πρόσβαση σε όλους τους χρήστες και τα δεδομένα τους , σε όλες τις διαδρομές και τα ρολόγια τους , όπως επίσης και στις μετρήσεις των ρολογιών. Το Dashboard επιτρέπει εύκολη παρακολούθηση και λήψη αποφάσεων, καθιστώντας τον διαχειριστή ενεργητικό στη βέλτιστη λειτουργία του συστήματος.

Ανάλυση Απαιτήσεων

Η ανάλυση απαιτήσεων αποτελεί το θεμέλιο για τον σχεδιασμό και την υλοποίηση ενός συστήματος υδρομετρήσεων.Στο κεφάλαιο αυτό, περιγράφονται λεπτομερώς οι λειτουργικές και μη λειτουργικές απαιτήσεις του συστήματος, προκειμένου να διασφαλιστεί η συνολική αποτελεσματικότητα και ικανοποίηση των αναγκών των χρηστών.

2.1 Λειτουργικές Απαιτήσεις

Σύστημα Καταγραφής Μετρήσεων :

Το σύστημα πρέπει να είναι ικανό να καταγράφει ακριβώς τις μετρήσεις που πραγματοποιούνται από κάθε υδρόμετρο, εξασφαλίζοντας υψηλή ακρίβεια και αξιοπιστία στα δεδομένα.

Ασφαλής Σύνδεση Χρηστών :

Η εφαρμογή πρέπει να παρέχει ασφαλή μέθοδο σύνδεσης για τους χρήστες, χρησιμοποιώντας μοναδικά κλειδιά (tokens) για την αναγνώριση και εξουσιοδότηση.

Ενημέρωση Δεδομένων σε Πραγματικό Χρόνο :

Το σύστημα πρέπει να ενημερώνει τα δεδομένα σε πραγματικό χρόνο, επιτρέποντας στους χρήστες να παρακολουθούν άμεσα την κατανάλωση και την απόδοση των υδρομέτρων, σύμφωνα με το ρόλο τους.

Offline χρήση για καταγραφή μετρήσεων :

Οι χρήστες πρέπει να έχουν τη δυνατότητα να πραγματοποιούν μετρήσεις ακόμη και όταν δεν υπάρχει σύνδεση στο διαδίκτυο, με τα δεδομένα να αποθηκεύονται τοπικά και να στέλνονται αυτόματα όταν η σύνδεση αποκατασταθεί.

2.2 Μη Λειτουργικές Απαιτήσεις

Ασφάλεια Δεδομένων :

Το σύστημα πρέπει να παρέχει υψηλά επίπεδα ασφάλειας για τα δεδομένα, με κρυπτογράφηση των επικοινωνιών και πρόσβαση μόνο με εξουσιοδοτημένα κλειδιά.

Συμβατότητα Συστημάτων :

Το σύστημα πρέπει να είναι συμβατό με διάφορες πλατφόρμες, όπως Android για την εφαρμογή κινητού και web για τον διαχειριστικό πίνακα.

Δικαιώματα Πρόσβασης :

1. **Users (Καταμετρητές)**
    - Προβολή και Επεξεργασία Μετρήσεων:
      Οι χρήστες έχουν το δικαίωμα να προβάλλουν τις τελευταίες μετρήσεις που έχουν πραγματοποιήσει και να τις επεξεργαστούν.
    - Καταχώρηση Νέων Μετρήσεων:
      Οι χρήστες μπορούν να καταχωρούν νέες μετρήσεις στο σύστημα στα ρολόγια που βρίσκονται στην διαδρομή που τους έχει ανατεθεί.
    - Προβολή ρολογιών στον χάρτη :
      Οι καταμετρητές έχουν την δυνατότητα να προβάλουν στον χάρτη μέσω Google Maps, τα ρολόγια που ανήκουν στην διαδρομή που τους έχει ανατεθεί.

Παρακάτω παρουσιάζεται(Figure 1) το use case diagram του User.Σε αυτό παρατηρούμε τι δυνατότητες έχει ο χρήστης χρησιμοποιώντας την εφαρμογή**.**Συγκεκριμένα, ο χρήστης αφού κάνει log in, μπορεί να κάνει καταχώρηση μια νέας μέτρησης, να κάνει εμφάνιση των μετρήσεων του έχοντας την δυνατότητα να επεξεργαστεί οποια θέλει. Τέλος μπορεί να εμφανίσει τις συντεταγμένες των ρολογιών της διαδρομής που έχει αναλάβει έχοντας την δυνατότητα να κάνει προβολή αυτών στον χάρτη.

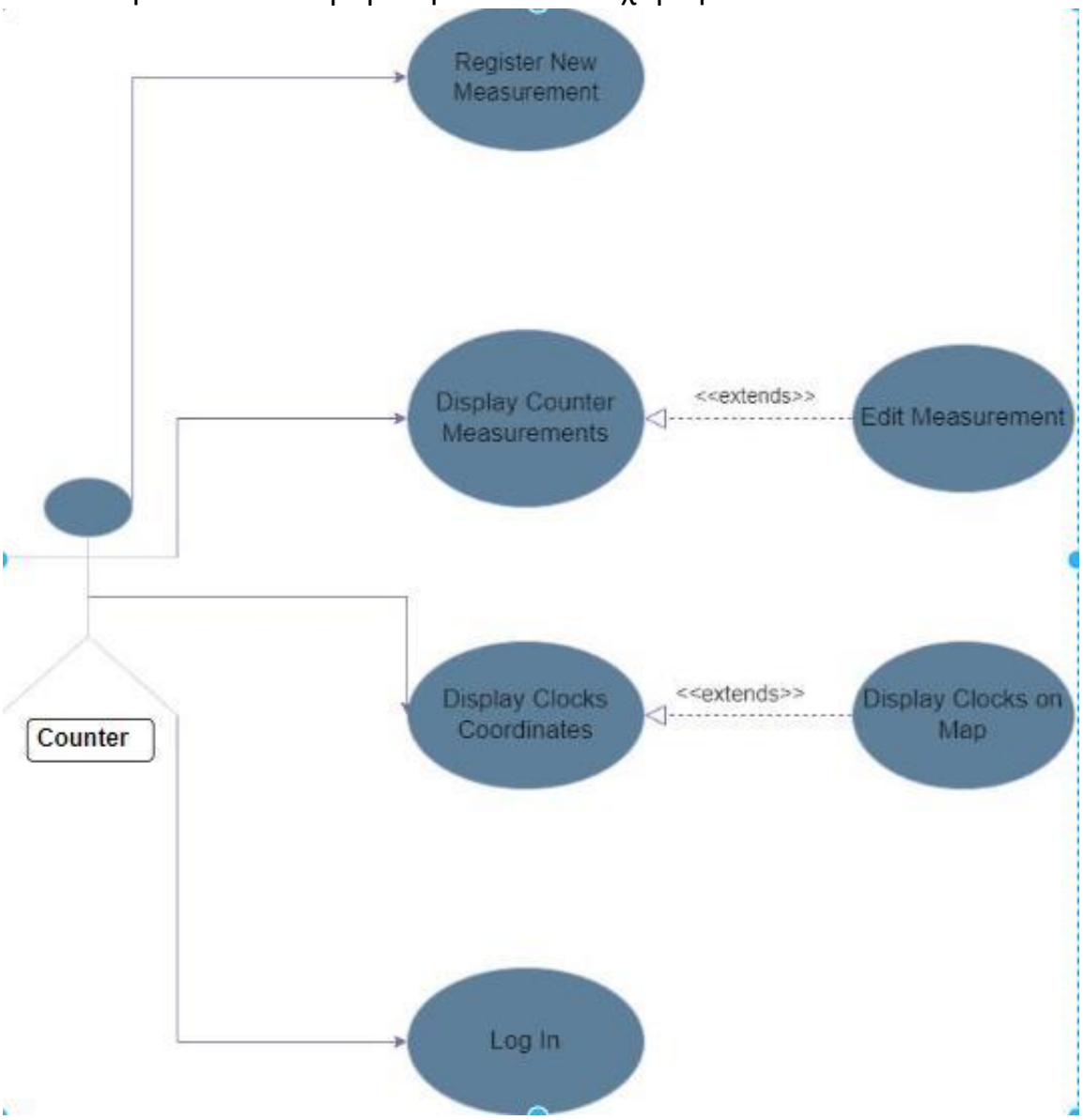

Figure 1 - Use case Diagram of a User

Το παρακάτω διάγραμμα(Figure 2) απεικονίζει τις δραστηριότητες που μπορεί να εκτελέσει ένας χρήστης(User) μετά την επιτυχημένη σύνδεση (login) στην android εφαρμογή. Οι βασικές ενέργειες περιλαμβάνουν:

- Register Measurements (Εγγραφή Μετρήσεων):
  Ο χρήστης επιλέγει να καταχωρήσει νέες μετρήσεις.
  Καταχωρεί τα δεδομένα των μετρήσεων.
  Υποβάλλει τις μετρήσεις.
  Λαμβάνει επιβεβαίωση για την επιτυχή καταχώρηση.
- Update Value (Ενημέρωση Τιμής):
  Ο χρήστης επιλέγει να ενημερώσει την τιμή μιας υπάρχουσας μετρητικής μονάδας.
  Επιλέγει τη μετρητική μονάδα.
  Καθορίζει τη νέα τιμή.
  Υποβάλλει την ενημέρωση.
  Λαμβάνει επιβεβαίωση για την επιτυχή ενημέρωση.
- Get Coordinates (Λήψη Συντεταγμένων):
  Ο χρήστης επιλέγει να δει τις συντεταγμένες ενός συγκεκριμένου δρομολογίου (route).
  Επιλέγει το συγκεκριμένο δρομολόγιο.
  Λαμβάνει τις συντεταγμένες.
  Εμφανίζονται οι συντεταγμένες για προβολή.
  Προβολή συντεταγμένων στον χάρτη.
- Logout (Αποσύνδεση):
  Ο χρήστης αποσυνδέεται από το σύστημα.

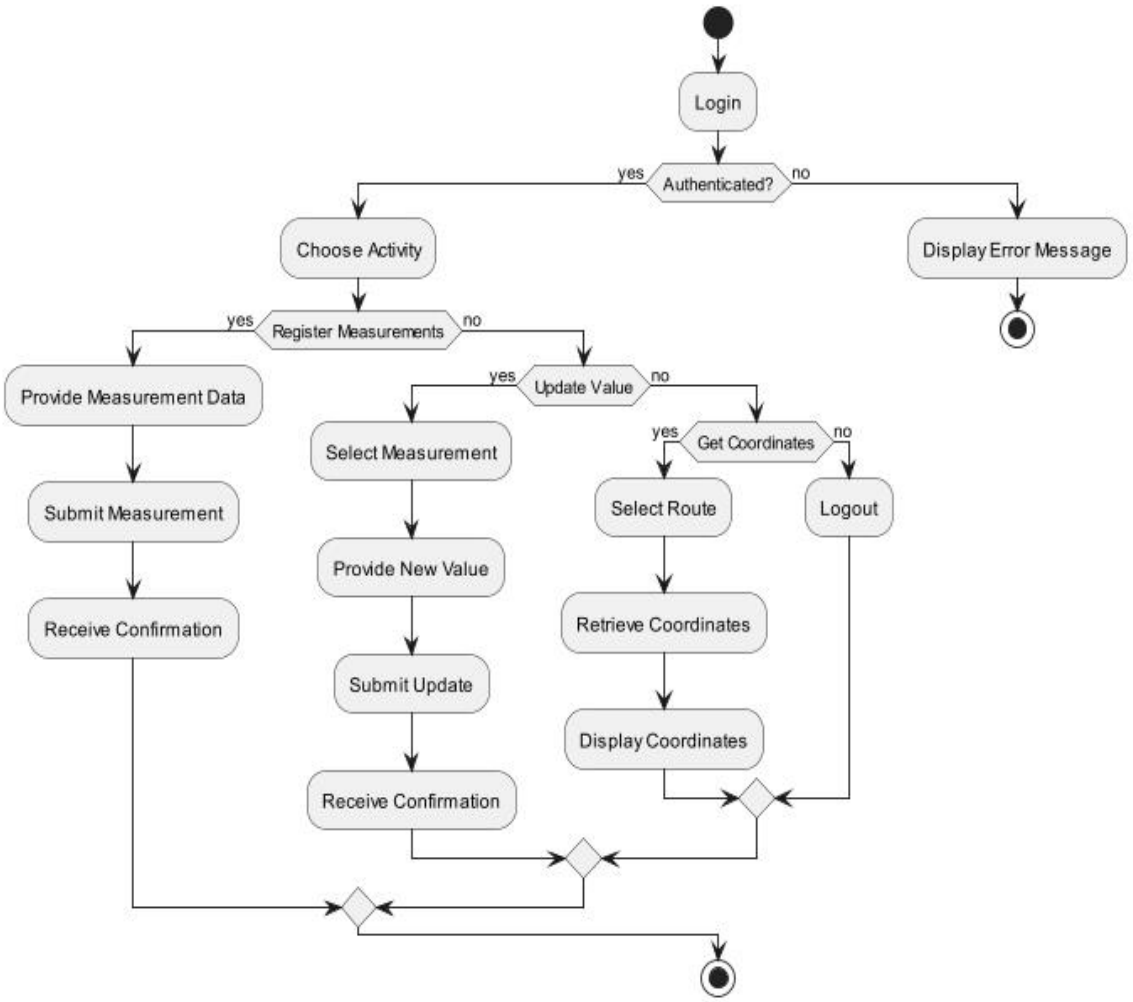

Figure 2 - Activity Diagram of a User

2. Admins (Διαχειριστές)

- Πλήρης Πρόσβαση σε Μετρήσεις:
  Οι διαχειριστές έχουν πλήρη πρόσβαση και δικαιώματα επεξεργασίας για όλες τις μετρήσεις.
- Διαχείριση Χρηστών:
  Δικαίωμα δημιουργίας, επεξεργασίας και διαγραφής χρηστών.
- Διαχείριση Ρολογιών :
  Δικαίωμα δημιουργίας, επεξεργασίας και διαγραφής ρολογιών.
- Διαχείριση Διαδρομών :
  Δικαίωμα δημιουργίας, επεξεργασίας και διαγραφής διαδρομών.

Παρακάτω παρουσιάζεται(Figure 3) το use case diagram του Admin. Σε αυτό παρατηρούμε τι δυνατότητες έχει ο χρήστης χρησιμοποιώντας το admin dashboard**.**Συγκεκριμένα ο admin αφου κάνει login , έχει τρείς επιλογες : να

κάνει εμφάνιση των διαδρομών , εμφάνιση μετρητών ή εμφανιση ρολογιών.Κατά την εμφάνιση διαδρομών μπορεί να κάνει επεξεργασία , προσθήκη ή διαγραφή διαδρομής και επίσης μπορεί να εμφανίσει τα ρολόγια που ανήκουν στην εκάστοτε διαδρομή.

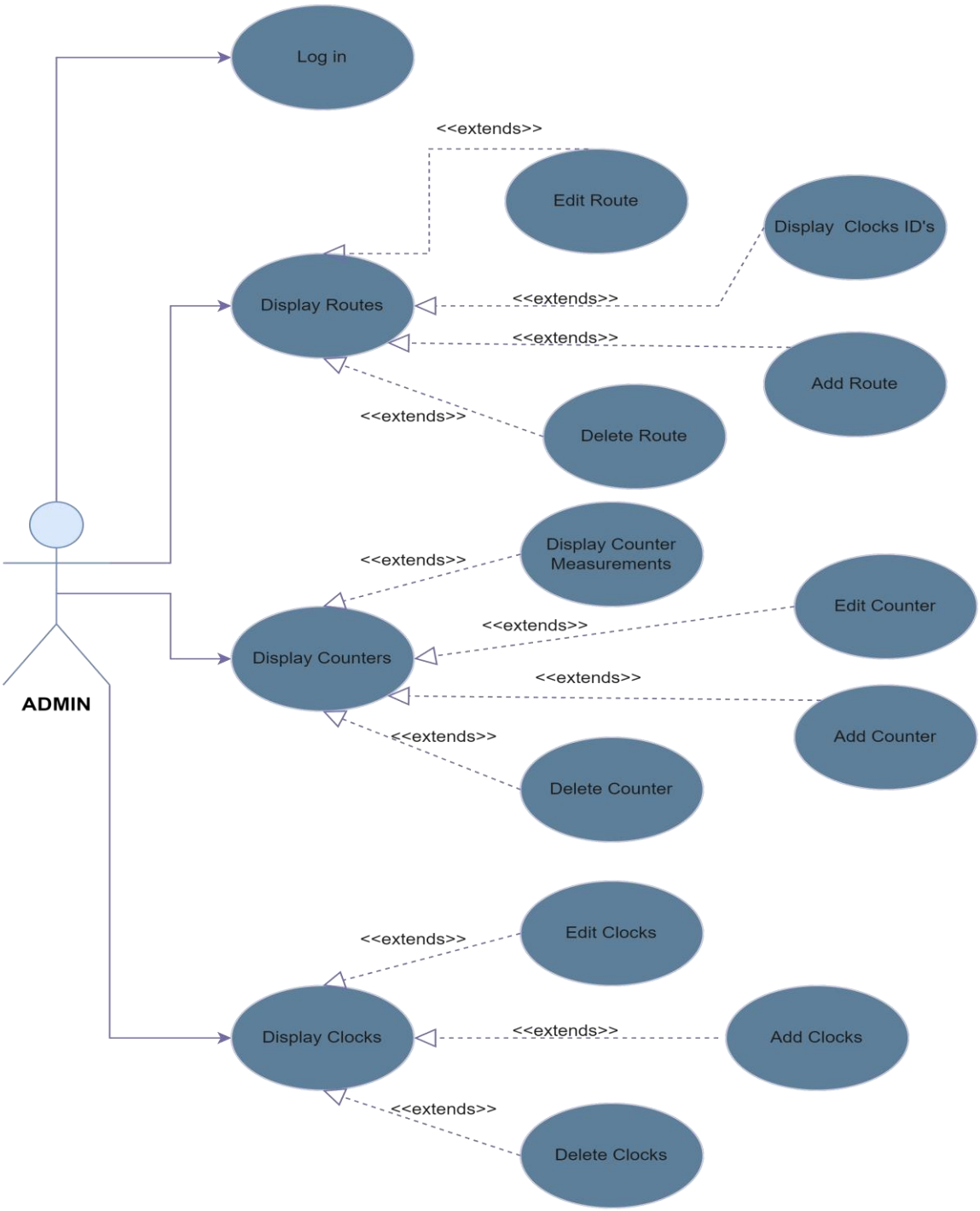

Figure 3- Use case Diagram of an Admin

Το παρακάτω διάγραμμα δραστηριότητας (Figure 4) απεικονίζει τις δραστηριότητες που μπορεί να εκτελέσει ένας χρήστης στο admin dashboard.
Τα βασικά βήματα είναι τα εξής :

Login: Ο χρήστης ξεκινά με την ενέργεια της σύνδεσης (Login).
Αυθεντικοποίηση: Ελέγχεται αν ο χρήστης είναι αυθεντικοποιημένος. Αν δεν είναι, εμφανίζεται ένα μήνυμα σφάλματος.
Επιλογή Δραστηριότητας: Αφού συνδεθεί επιλέγει μια από τις διαθέσιμες δραστηριότητες.
Προβολή Δρομολογίων (View Routes): Αν επιλέξει να δει τα δρομολόγια, πραγματοποιούνται οι εξής υποενέργειες:
Ανάκτηση δεδομένων δρομολογίων.
Εμφάνιση δρομολογίων.
Επεξεργασία, διαγραφή, προσθήκη νέου δρομολογίου ή εμφάνιση των id’s των ρολογιών της συγκεκριμένης διαδρομής.
Προβολή Μετρητών (View Counters): Αν θέλει να δει τους μετρητές, πραγματοποιούνται οι εξής υποενέργειες :
Ανάκτηση δεδομένων μετρητών.
Εμφάνιση Μετρητών.
Επεξεργασία, διαγραφή, προσθήκη νέου μετρητή ή εμφάνιση των μετρήσεων του μετρητή που θα επιλέξει.
Προβολή Ρολογιών (View Clocks): Αν επιλέξει να δει τα ρολόγια, πραγματοποιούνται οι υποενέργειες :
Ανάκτηση δεδομένων ρολογιών.
Εμφάνιση Ρολογιών.
Επεξεργασία, διαγραφή, προσθήκη νέου ρολογιού.
Αποσύνδεση (Logout): Ο χρήστης μπορεί να αποσυνδεθεί ανά πάσα στιγμή.

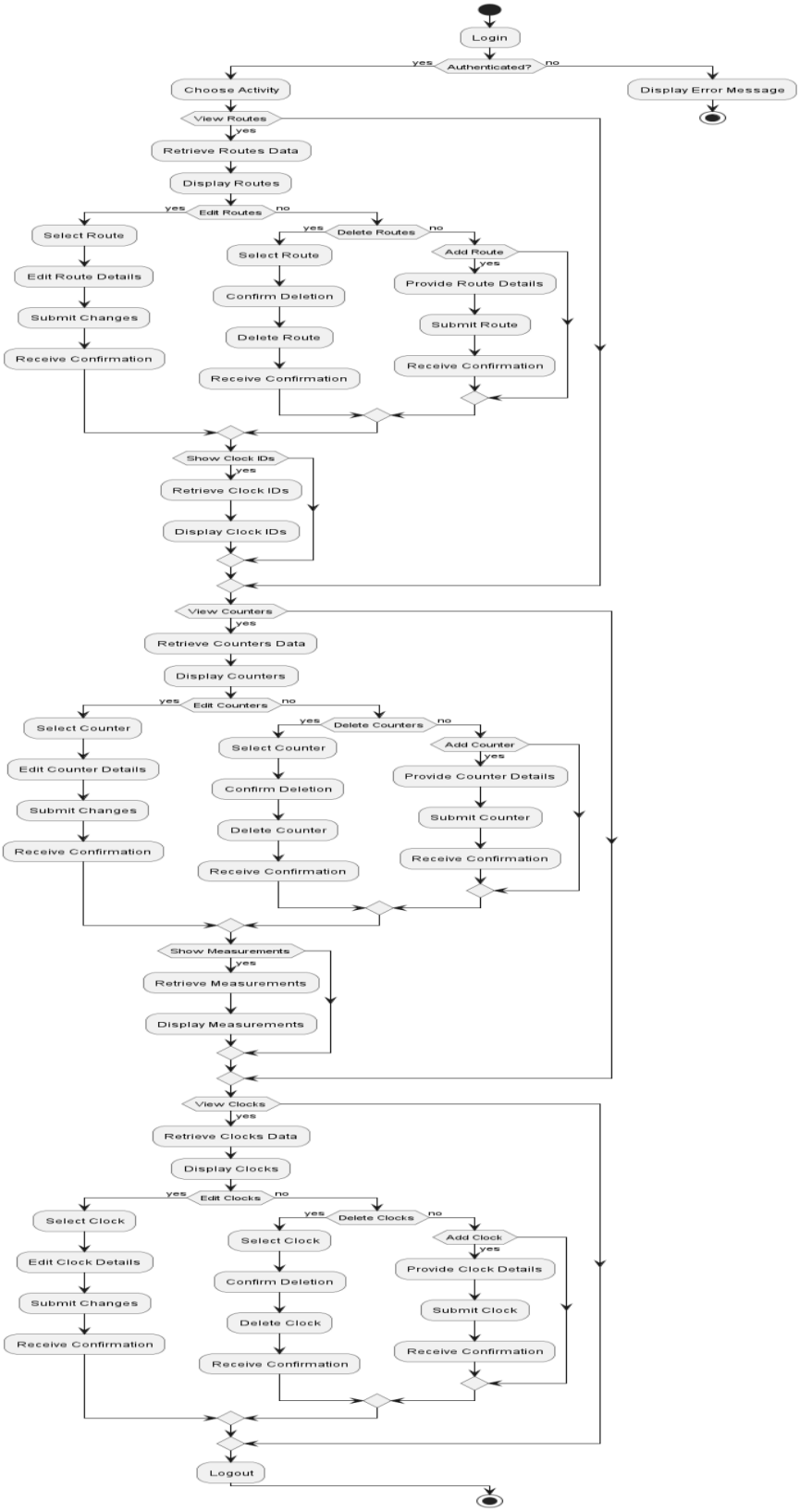

Figure 4 - Activity Diagram of an Admin

Σχεδιασμός Οντοτήτων

3.1 Σχεδιασμός Βάσης Δεδομένων

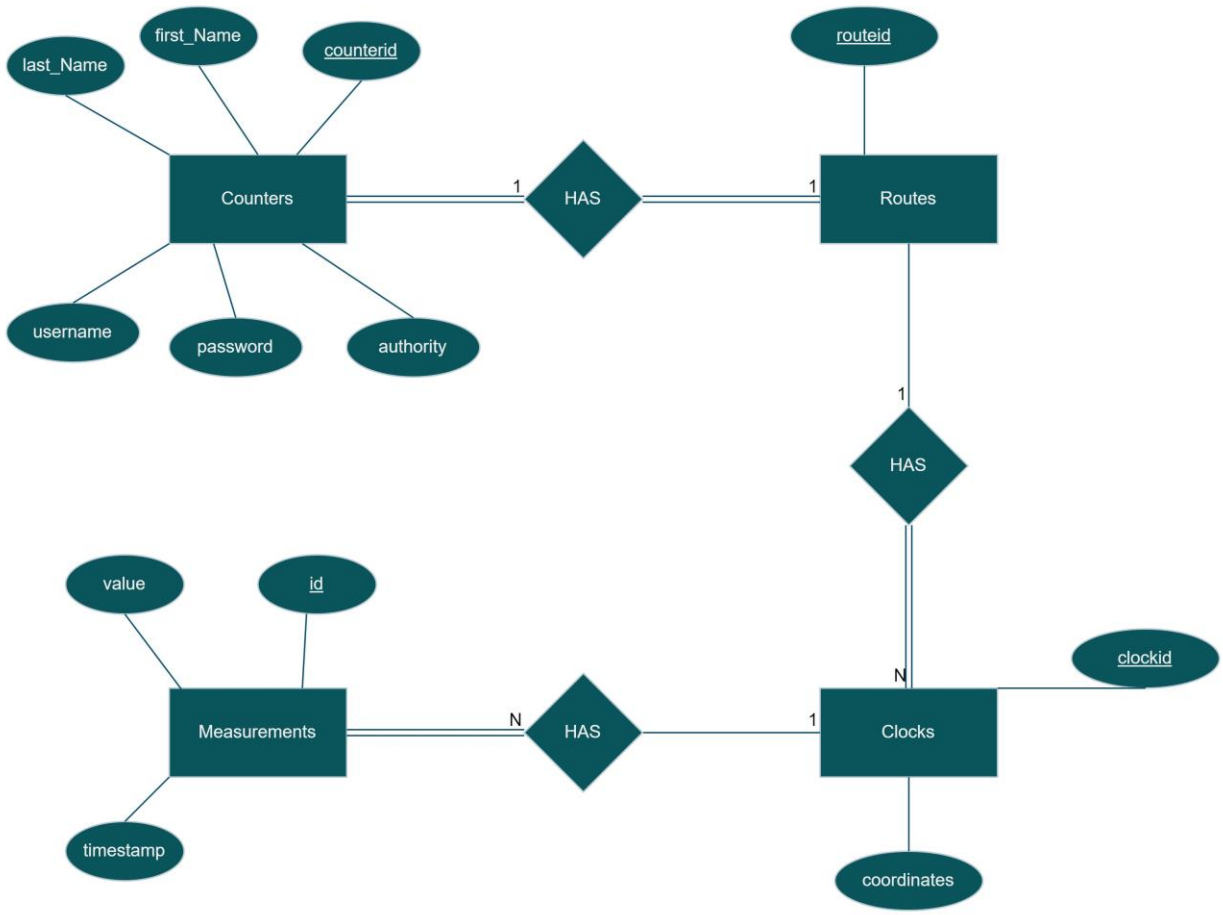

Figure 5- E/R Diagram for the database

Όπως φαίνεται και στην εικόνα (Figure 5) έχουμε τα εξής :

Οντότητες :

1.Routes:

routeid (Πρωτεύον Κλειδί)

2.Counters:

counterid (Πρωτεύον Κλειδί)

authority

first_name

last_name

password

username

3.Clocks:

clockid (Πρωτεύον Κλειδί)

coordinates

4.Measurements:

id (Πρωτεύον Κλειδί)

value

timestamp

Συσχετίσεις:

Ένας Counter έχει ένα συγκεκριμένο Route (σχέση ένα προς ένα).

Ένα Route ανήκει σε έναν συγκεκριμένο Counter (σχέση ένα προς ένα).

Ένα Clock ανήκει σε ένα συγκεκριμένο Route (σχέση πολλά προς ένα).

Ένα Measurement ανήκει σε ένα συγκεκριμένο Clock (σχέση πολλά προς ένα).

Παρακάτω παρουσιάζονται οι κλάσεις που θα υπάρχουν στην κεντρική βάση με τα χαρακτηριστικά τους(Figure 6).

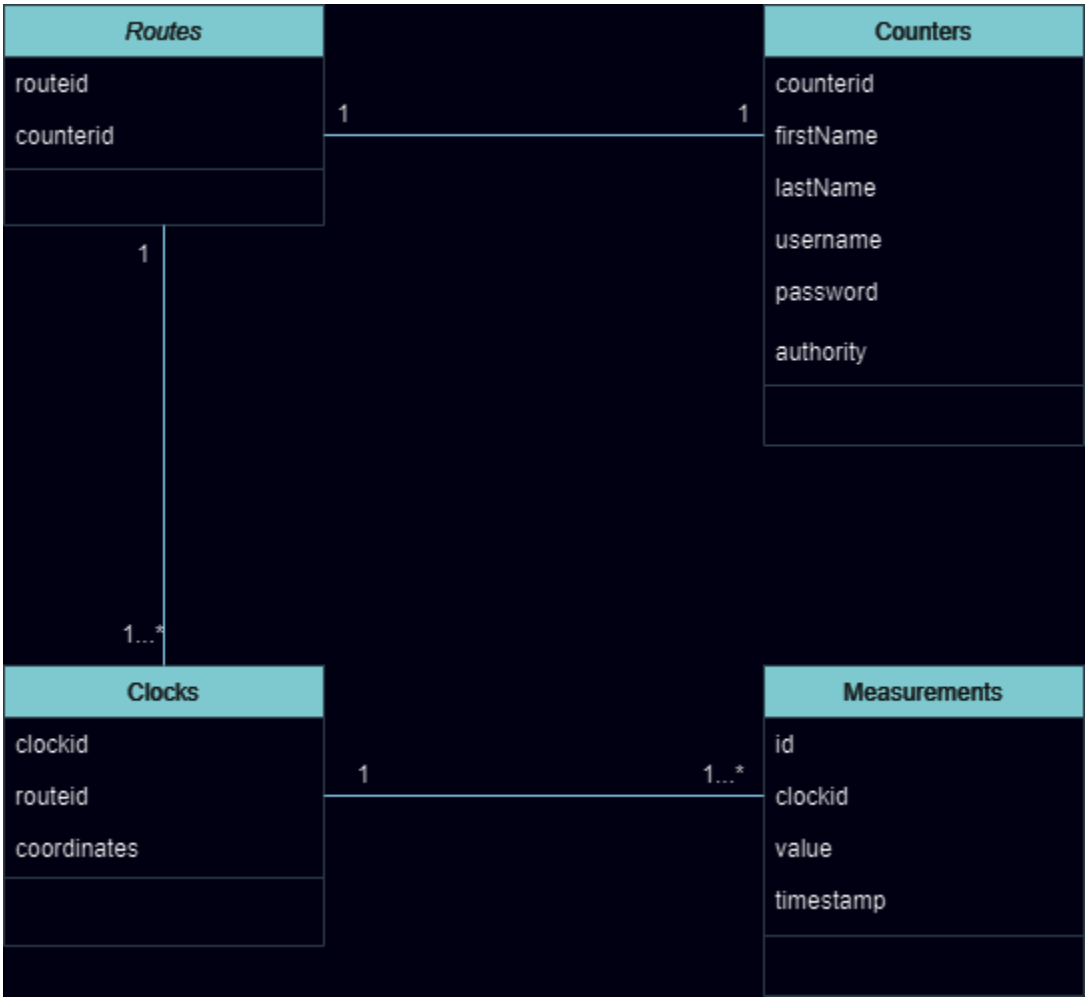

Figure 6 - A class diagram of database tables

3.2 Σχεδιασμός Συστήματος

Αρχιτεκτονική της Εφαρμογής:

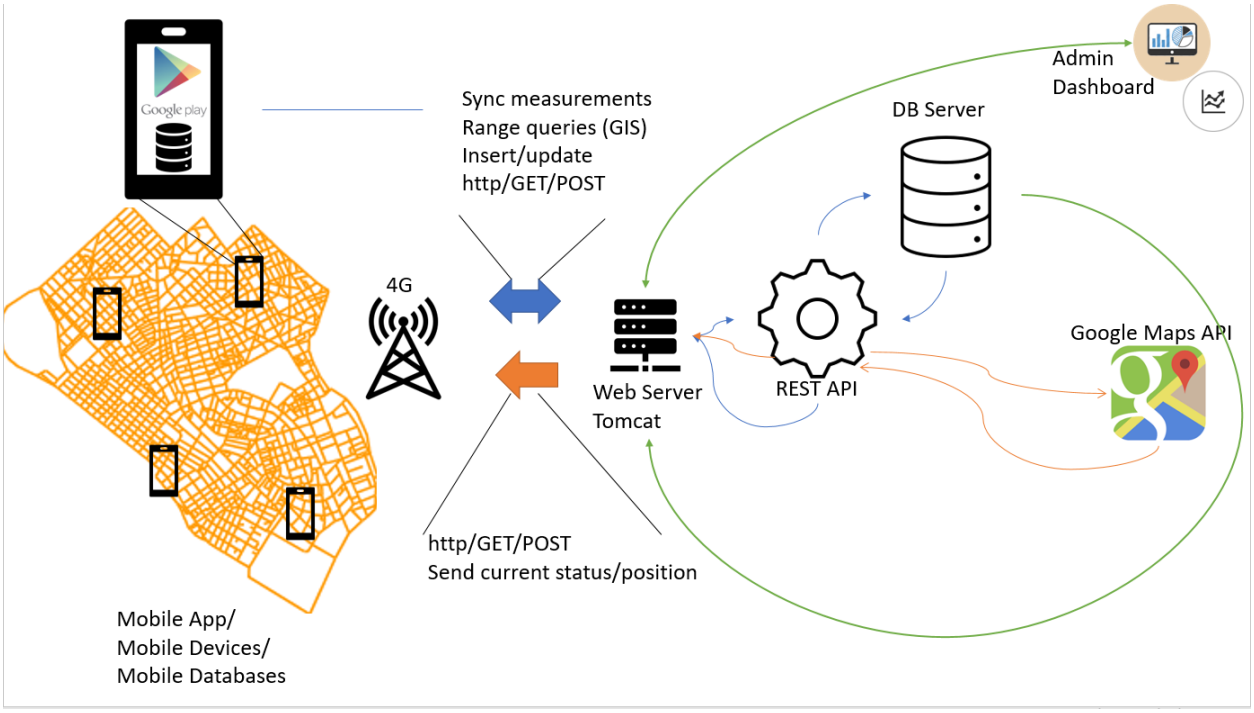

Figure 7 -Application Architecture and Fundamental Components

Όπως βλέπουμε στην παραπάνω φωτογραφία (Figure 7) το σύστημα αποτελείται από πολλά δομικά στοιχεία που συνεργάζονται για την ομαλή λειτουργία του. Η Android εφαρμογή εξυπηρετεί τους καταμετρητές παρέχοντας μια διεπαφή για τη διαχείριση δεδομένων όπως μετρήσεις, ρολόγια και διαδρομές. Οι αιτήσεις από την Android εφαρμογή αλληλεπιδρούν με το σύστημα μέσω ενός RESTful API που υλοποιείται με τη χρήση του Spring Boot. Το API επικοινωνεί με μια βάση δεδομένων, η οποία αποθηκεύει πληροφορίες σχετικά με μετρήσεις, ρολόγια, διαδρομές και καταμετρητές.

Το σύστημα διαθέτει επίσης έναν διαχειριστικό πίνακα (admin dashboard) που επιτρέπει στους διαχειριστές να διαχειρίζονται τα δεδομένα και τους καταμετρητές. Οι διαχειριστές μπορούν να πραγματοποιούν ενέργειες όπως προσθήκη, επεξεργασία και διαγραφή ρολογιών, μετρήσεων και διαδρομών. Το Android σύστημα, το RESTful API, το Spring Boot και το admin dashboard

συνθέτουν ένα ολοκληρωμένο περιβάλλον που παρέχει ασφάλεια και αποδοτικότητα για τους τελικούς χρήστες και τους διαχειριστές.

Back-end:

Γλώσσα Προγραμματισμού:

Ο κώδικας του back-end είναι υλοποιημένος σε Java.

Πλαίσιο Εργασίας (Framework):

Το Spring Boot [1] είναι ένα ευέλικτο και ισχυρό πλαίσιο ανάπτυξης για τη δημιουργία εφαρμογών Java. Σχεδιάστηκε για να είναι εύκολο στη χρήση, με ενσωματωμένες προεπιλεγμένες ρυθμίσεις και συμβάσεις που επιτρέπουν την γρήγορη ανάπτυξη εφαρμογών χωρίς την ανάγκη για πολύπλοκη διαμόρφωση.

Μερικά χαρακτηριστικά του Spring Boot περιλαμβάνουν:

Ενσωματωμένος Web Server:

Το Spring Boot περιλαμβάνει ενσωματωμένους web servers όπως το Tomcat, το Jetty ή το Undertow, επιτρέποντας την εκκίνηση ενός web server χωρίς την ανάγκη εγκατάστασης ή περαιτέρω ρύθμιση.

Διαχείριση Εξαρτήσεων (Dependency Management):

Το Spring Boot χρησιμοποιεί το Maven ή το Gradle για τη διαχείριση των εξαρτήσεων, κάνοντας εύκολο τον έλεγχο των εκδόσεων και των συμβατοτήτων.

Εύκολη Ρύθμιση (Convention over Configuration):

Προσφέρει προεπιλεγμένες ρυθμίσεις για την ανάπτυξη, με τη δυνατότητα προσαρμογής σε περιβάλλοντα όπου απαιτείται.

Ενσωματωμένη Σχεδίαση RESTful Υπηρεσιών:

Υποστηρίζει τη σχεδίαση RESTful υπηρεσιών, καθιστώντας εύκολο τον σχεδιασμό και την υλοποίηση API.

Επικοινωνία με Βάση Δεδομένων:

Χρησιμοποιήθηκε το PostgreSQL [2] για την αποθήκευση και ανάκτηση δεδομένων.

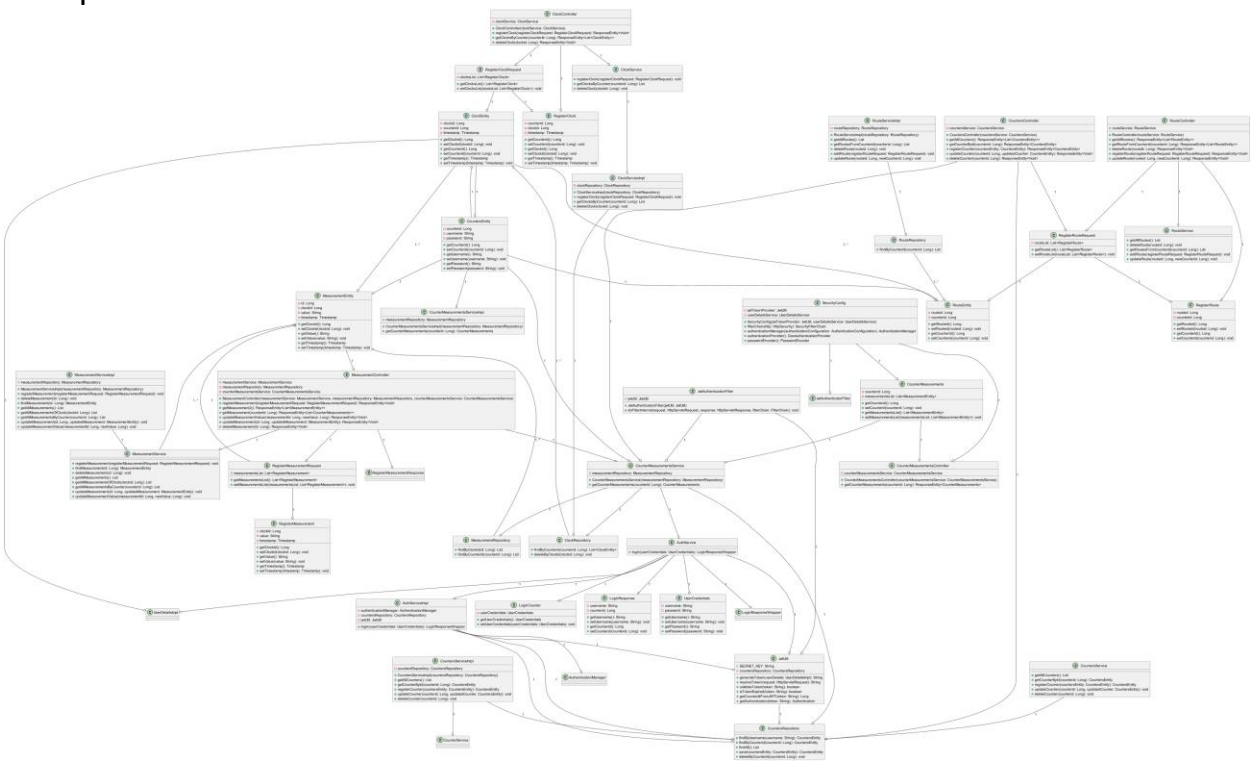

Figure 8 - A class diagram showing the classes used for back-end

Παραπάνω(Figure 8) βλέπουμε το class diagram του back-end, στο οποίο φαίνονται οι κλάσεις και οι σχέσεις μεταξύ τους. Αυτά αναλύονται περαιτέρω στο επόμενο κεφάλαιο (4.2 Framework - Springboot).

Front-end:

Το Android Studio χρησιμοποιείται για τον σχεδιασμό του front-end της εφαρμογής. Η Android Studio είναι μια ενσωματωμένη ανάπτυξη περιβάλλοντος για τη δημιουργία εφαρμογών Android.

Το Android Studio χρησιμοποιείται για τον σχεδιασμό του front-end της εφαρμογής. Το Android Studio είναι μια ενσωματωμένη ανάπτυξη περιβάλλοντος για τη δημιουργία εφαρμογών Android.

Το Android Studio αποτελεί το επίσημο περιβάλλον ανάπτυξης (IDE) για το λειτουργικό σύστημα Android, προσφέροντας ένα ολοκληρωμένο σετ εργαλείων για τη δημιουργία εφαρμογών για κινητές συσκευές. Αναπτύχθηκε από τη Google και βασίζεται στο IntelliJ IDEA της JetBrains.

Κύρια χαρακτηριστικά:

- Σχεδίαση Διεπαφής:
  Το Android Studio παρέχει εύχρηστα εργαλεία για τον σχεδιασμό της γραφικής διεπαφής μιας εφαρμογής. Μπορείς να διαμορφώσεις τον τρόπο προβολής των οθονών για διάφορες συσκευές και αναλύσεις.
- Ανάπτυξη Κώδικα:
  Ο υπερσύγχρονος επεξεργαστής κώδικα βοηθά στο γράψιμο και στην επεξεργασία του κώδικα αποτελεσματικά. Υποστηρίζει γλώσσες όπως το Java, το Kotlin, και το C++.
- Εξομοίωση Συσκευής:
  Δυνατότητα ελέγχου της εφαρμογής σε εικονικές συσκευές Android ή φυσικές συσκευές που συνδέονται στον υπολογιστή. Η εξομοίωση συσκευής επιτρέπει την προεπισκόπηση της συμπεριφοράς της εφαρμογής σε διάφορες συνθήκες.
- Διαχείριση Εξαρτημάτων:
  Το Android Studio διαθέτει ενσωματωμένο σύστημα διαχείρισης εξαρτημάτων που επιτρέπει την εγκατάσταση, ενημέρωση και διαχείριση βιβλιοθηκών και SDKs.
- Ανάλυση Απόδοσης:
  Το Android Studio παρέχει εργαλεία για τον έλεγχο και τη βελτιστοποίηση της απόδοσης της εφαρμογής, βοηθώντας στην αναγνώριση πιθανών προβλημάτων.

Front-end - Android Studio:

Ο σχεδιασμός του front-end γίνεται σε Java με τη χρήση του Android SDK [3]. Χρήση XML για τον ορισμό της διάταξης του UI.

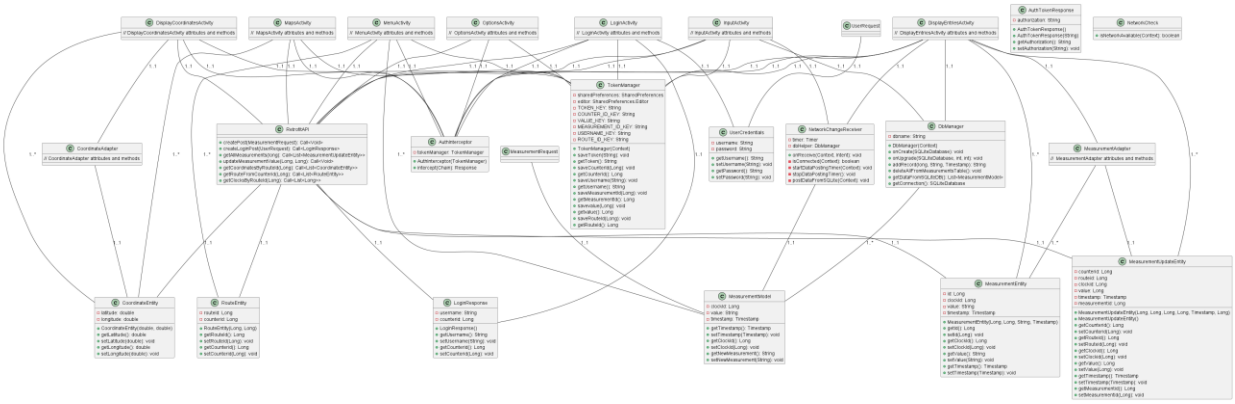

Figure 9 - A class diagram showing the classes used for front-end

Παραπάνω (Figure 9) βλέπουμε το class diagram του front-end(android), στο οποίο φαίνονται οι κλάσεις και οι σχέσεις μεταξύ τους. Αυτά αναλύονται περαιτέρω σε επόμενο κεφάλαιο (4.3 Application – Android Studio).

<u>Σύστημα Login και Authorization</u>:

Σύστημα Login:

Ο χρήστης εισέρχεται στην εφαρμογή μέσω ενός συστήματος σύνδεσης (login) που απαιτεί όνομα χρήστη και κωδικό πρόσβασης.

Σύστημα Authorization με χρήση JWT:

Μετά την επιτυχή σύνδεση, ο χρήστης λαμβάνει ένα JSON Web Token (JWT) [4] ως απόδειξη ταυτότητας. Το JWT περιέχει πληροφορίες για τον χρήστη και τα δικαιώματά του.

Η επικοινωνία με το back-end γίνεται με το JWT, το οποίο προσαρτάται στα αιτήματα για επικύρωση και αυθεντικοποίηση. Αυτό εξασφαλίζει ότι μόνο εξουσιοδοτημένοι χρήστες έχουν πρόσβαση σε συγκεκριμένους πόρους του back-end.

Αυτή η αρχιτεκτονική δίνει έμφαση στην ασφάλεια, την αποδοτικότητα και τη συνοχή της εφαρμογής υδρομετρήσεων και πραγματοποιείται με τον τρόπο που φαίνεται στο παρακάτω διάγραμμα δραστηριότητας (Figure 10) :

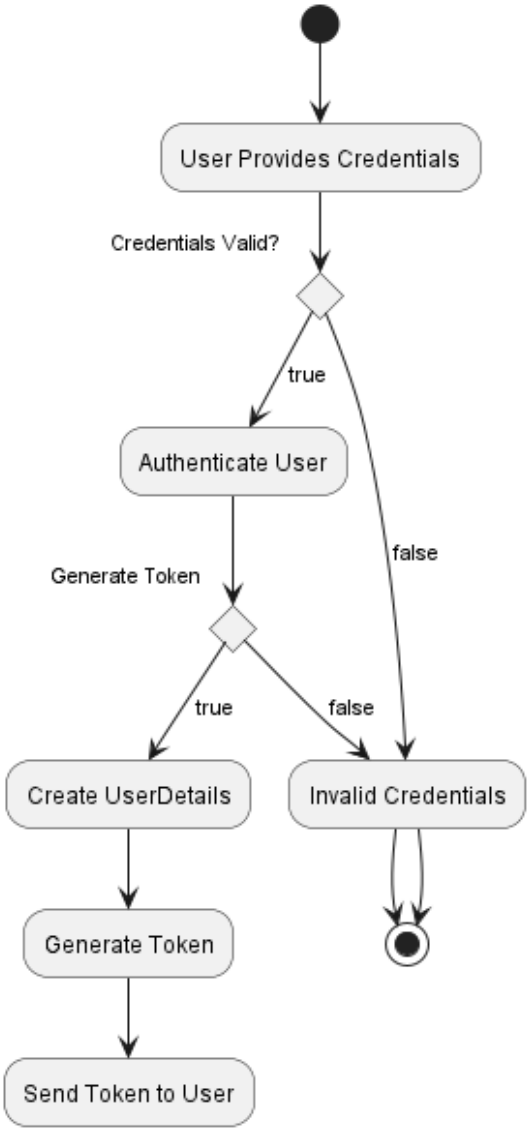

Figure 10 - Activity Diagram showing the process of User Authedication

Admin Dashboard :

Javascript [5] :

Η JavaScript είναι μια αντικειμενοστραφής γλώσσα προγραμματισμού και υποστηρίζει τον προγραμματισμό που βασίζεται σε συμβάντα, λειτουργικά και επιτακτικά. Χρησιμοποιείται κυρίως για την ανάπτυξη εφαρμογών στην πλευρά του πελάτη, που σημαίνει ότι ο κώδικας εκτελείται από το πρόγραμμα περιήγησης ιστού του χρήστη και όχι από τον διακομιστή. Αυτό επιτρέπει στη JavaScript να δημιουργεί διαδραστικά εφέ σε μια ιστοσελίδα, όπως ενημέρωση του περιεχομένου της σελίδας χωρίς επαναφόρτωση της σελίδας, επικύρωση φορμών, δημιουργία κινούμενων εικόνων και πολλά άλλα.

Είναι μια ευέλικτη γλώσσα και μπορεί να χρησιμοποιηθεί με πολλούς διαφορετικούς τρόπους, όπως δημιουργία εφαρμογών διακομιστή με χρήση Node.js, ανάπτυξη παιχνιδιών και δημιουργία εφαρμογών για desktop και mobile συσκευές χρησιμοποιώντας εργαλεία, όπως το Electron και το React Native.

Είναι μία από τις τρεις βασικές τεχνολογίες ανάπτυξης ιστοσελίδων, μαζί με την HTML (Hypertext Markup Language) και την CSS (Cascading Style Sheets) και αποτελεί βασικό συστατικό της σύγχρονης ανάπτυξης εφαρμογών. Η JavaScript είναι μια δημοφιλής γλώσσα και πολλές βιβλιοθήκες και πλαίσια έχουν αναπτυχθεί γι' αυτήν, διευκολύνοντας τους προγραμματιστές να δημιουργούν πολύπλοκες εφαρμογές και να αλληλεπιδρούν με άλλες τεχνολογίες.

Typescript [6]:

Η TypeScript είναι ένα υπερσύνολο της JavaScript, που σημαίνει ότι όλος ο έγκυρος κώδικας JavaScript είναι επίσης έγκυρος κώδικας TypeScript. Προσθέτει προαιρετικούς, στατικού τύπου σχολιασμούς στη JavaScript, οι οποίοι μπορούν να βοηθήσουν να γίνουν οι μεγάλες, πολύπλοκες βάσεις κώδικα πιο διαχειρίσιμες και πιο κατανοητές. Περιλαμβάνει επίσης δυνατότητες από τις πιο πρόσφατες εκδόσεις JavaScript και μπορεί να χρησιμοποιηθεί για τη σύνταξη σύγχρονου κώδικα JavaScript, όπως async/wait και destructuring.

Παρέχει έλεγχο τύπων κατά τη στιγμή της μεταγλώττισης, ο οποίος μπορεί να βοηθήσει στη σύλληψη σφαλμάτων και στη βελτίωση της συνολικής ποιότητας του κώδικα. Αυτό μπορεί να είναι ιδιαίτερα χρήσιμο σε μεγαλύτερα έργα με πολλούς προγραμματιστές, καθώς μπορεί να βοηθήσει να διασφαλιστεί ότι όλοι

χρησιμοποιούν τους σωστούς τύπους και να αποτρέψουν ορισμένους τύπους σφαλμάτων χρόνου εκτέλεσης.

Η TypeScript γίνεται όλο και πιο δημοφιλής και πολλά δημοφιλή πλαίσια front-end, όπως το Angular και το React, χρησιμοποιούν πλέον την TypeScript ως προεπιλεγμένη γλώσσα.

Συνοπτικά, η TypeScript είναι μια γλώσσα στατικών τύπων, εξαιρετικά ευανάγνωστη, που μπορεί να χρησιμοποιηθεί για την ανάπτυξη μεγάλων, πολύπλοκων εφαρμογών. Προσθέτοντας σχολιασμούς στατικού τύπου, μπορεί να βοηθήσει στον εντοπισμό σφαλμάτων και στη βελτίωση της συνολικής ποιότητας του κώδικα, διευκολύνοντας τη διατήρηση και την κλιμάκωση.

HTML [7]:

Η HTML (Hypertext Markup Language) είναι η τυπική γλώσσα σήμανσης που χρησιμοποιείται για τη δημιουργία ιστοσελίδων. Είναι μια μορφή που βασίζεται σε κείμενο που παρέχει τη δομή και το περιεχόμενο των ιστοσελίδων, συμπεριλαμβανομένων επικεφαλίδων, παραγράφων, συνδέσμων, εικόνων και άλλων. Δεν είναι γλώσσα προγραμματισμού, αλλά χρησιμοποιείται για τον καθορισμό της δομής και του περιεχομένου των ιστοσελίδων και αποτελεί βασικό συστατικό της ανάπτυξης διαδικτυακών εφαρμογών.

Είναι μια γλώσσα σήμανσης, που σημαίνει ότι αποτελείται από ένα σύνολο ετικετών και χαρακτηριστικών, τα οποία χρησιμοποιούνται για να περιγράψουν το περιεχόμενο και τη δομή μιας ιστοσελίδας. Τα χαρακτηριστικά της μπορούν να χρησιμοποιηθούν για την παροχή πρόσθετων πληροφοριών σχετικά με μια ετικέτα, όπως η πηγή μιας εικόνας ή ο προορισμός ενός συνδέσμου.

CSS [8]:

Η CSS (Cascading Style Sheets) είναι μια γλώσσα που χρησιμοποιείται για την περιγραφή της εμφάνισης και της μορφοποίησης ενός εγγράφου γραμμένου σε HTML. Χρησιμοποιείται για να διαχωρίσει το περιεχόμενο μιας ιστοσελίδας, η οποία ορίζεται με χρήση HTML, από την παρουσίασή της, η οποία ορίζεται με χρήση CSS.

Επιτρέπει στους προγραμματιστές να ελέγχουν τη διάταξη, το χρώμα, τη γραμματοσειρά και άλλα οπτικά στοιχεία μιας ιστοσελίδας, καθιστώντας δυνατή

τη δημιουργία σελίδων με στυλ και οπτικά ελκυστικές. Οι κανόνες CSS ορίζονται χρησιμοποιώντας επιλογείς, οι οποίοι επιλέγουν τα στοιχεία HTML, στα οποία πρέπει να εφαρμοστούν τα στυλ, και ένα σύνολο δηλώσεων, που ορίζουν τα ίδια τα στυλ. Για παράδειγμα, ένας κανόνας CSS μπορεί να ορίζει ότι όλες οι επικεφαλίδες πρέπει να είναι μπλε και να έχουν συγκεκριμένο μέγεθος γραμματοσειράς.

JSON [9]:

JSON (JavaScript Object Notation) ονομάζεται μια απλή μορφή ανταλλαγής δεδομένων που είναι εύκολο να διαβάσουν και να γράψουν οι προγραμματιστές και εύκολα να αναλύσουν και να δημιουργήσουν οι μηχανές. Είναι μια ανοιχτή τυπική μορφή που χρησιμοποιείται ευρέως για ανταλλαγή δεδομένων μεταξύ εφαρμογών, ειδικά για ανταλλαγή δεδομένων στο διαδίκτυο.

Βασίζεται σε ένα υποσύνολο της γλώσσας προγραμματισμού JavaScript και χρησιμοποιεί ζεύγη κλειδιών-τιμών για την αναπαράσταση δεδομένων. Τα κλειδιά είναι συμβολοσειρές και οι τιμές μπορεί να είναι συμβολοσειρές, αριθμοί, αντικείμενα (συλλογές ζευγών κλειδιών-τιμών), πίνακες (διατεταγμένες συλλογές τιμών), booleans (true ή false) ή null.

Ένα από τα πλεονεκτήματα είναι ότι η αναπαράσταση είναι ανεξάρτητη από τη γλώσσα, πράγμα που σημαίνει ότι μπορεί να χρησιμοποιηθεί με μια μεγάλη ποικιλία γλωσσών προγραμματισμού, συμπεριλαμβανομένων των JavaScript, Python, Ruby, PHP και πολλών άλλων. Είναι επίσης ελαφριά και συμπαγή, γεγονός που την καθιστά κατάλληλη για χρήση σε δίκτυα, ειδικά στο διαδίκτυο.

Η JSON μορφή έχει γίνει μια δημοφιλής εναλλακτική της XML (Extensible Markup Language), μια άλλη μορφή ανταλλαγής δεδομένων, λόγω της απλότητας, της συμπαγούς και ευκολίας χρήσης της. Χρησιμοποιείται συχνά για REST (Representational State Transfer) API, τα οποία χρησιμοποιούνται για την ανταλλαγή δεδομένων μεταξύ υπηρεσιών web, και χρησιμοποιείται επίσης ως μορφή δεδομένων για αποθήκευση και ανταλλαγή δεδομένων σε πολλές σύγχρονες εφαρμογές.

Angular [10]:

Η Angular είναι μια πλατφόρμα και ένα πλαίσιο για τη δημιουργία εφαρμογών πελάτη μιας σελίδας χρησιμοποιώντας HTML και TypeScript. Η Angular είναι γραμμένη σε TypeScript. Εφαρμόζει βασική και προαιρετική λειτουργικότητα ως ένα σύνολο βιβλιοθηκών TypeScript που εισάγονται στις εφαρμογές. Η αρχιτεκτονική μιας εφαρμογής Angular βασίζεται σε ορισμένες θεμελιώδεις έννοιες, όπως τα components και τα services. Τα πρώτα αποτελούν βασικά δομικά στοιχεία της Angular και είναι οργανωμένα σε NgModules. Τα NgModules συλλέγουν σχετικό κώδικα σε λειτουργικά σύνολα και προσφέρουν μια μορφή ομαδοποίησης. Μια Angular εφαρμογή ορίζεται από ένα σύνολο NgModules, εκ των οποίων ένα παίζει πηγαίο ρόλο και επιτρέπει την εκκίνηση, καθώς συνήθως υπάρχουν διάφορες λειτουργικές μονάδες.

Visual Studio Code [11]:

Το VSCode είναι ένα IDE περιβάλλον κατασκευασμένο και υποστηριζόμενο από την Microsoft. Παρέχει υποστήριξη για λειτουργίες ανάπτυξης, όπως εντοπισμός σφαλμάτων, εκτέλεση εργασιών και version control μέσω git. Μπορεί να χρησιμοποιηθεί για οποιαδήποτε γλώσσα προγραμματισμού, αλλά έχει ειδικά ενσωματωμένη υποστήριξη για την σύνταξη Typescript. Επίσης, υπάρχει η δυνατότητα εγκατάστασης επιπρόσθετων εφαρμογών για πιο εξειδικευμένες λειτουργίες, όπως π.χ. Angular snippets.

Υλοποίηση Λειτουργιών

4.1 Βάση Δεδομένων – Postgresql

Η βάση δεδομένων υλοποιήθηκε σε μία PostgreSQL βάση.

Στην βάση δημιουργήθηκαν τα βασικά tables δηλαδή τα **Routes**, οι **Counters** ,τα **Clocks** , τα **Measurements** και οι διασυνδέσεις μεταξύ τους (βλ.Figure 6). Επίσης δημιουργήθηκε αυτόματα από την PostgreSQL ο πίνακας με όνομα 'spatial_ref_sys'.

Το spatial_ref_sys είναι ένας πίνακας που χρησιμοποιείται στη PostgreSQL με την επέκταση PostGIS για την αποθήκευση και διαχείριση συστημάτων αναφοράς χωρικών δεδομένων (Spatial Reference Systems - SRS). Στο πλαίσιο των γεωχωρικών εφαρμογών, τα SRS καθορίζουν πώς τα γεωχωρικά δεδομένα αντιπροσωπεύονται χωρικά και πώς γίνεται ο προσδιορισμός της τοποθεσίας σε έναν χωρικό χώρο.

Ο πίνακας spatial_ref_sys περιλαμβάνει πληροφορίες για τα διάφορα συστήματα αναφοράς που υποστηρίζονται. Κάποιες από τις κύριες στήλες που συνήθως περιλαμβάνονται είναι οι εξής:

srid: Ο μοναδικός αριθμός αναγνώρισης του συστήματος αναφοράς.

auth_name και auth_srid: Τα ονόματα και οι αριθμοί αναγνώρισης του οργανισμού που καθορίζει το συγκεκριμένο SRS.

srtext και proj4text: Ο κείμενος περιγραφής του SRS σε μορφή WKT (Well-Known Text) και Proj4.

Ο πίνακας αυτός δημιουργήθηκε έτσι ώστε να ελέγχονται και να αποθηκεύονται σωστά οι συντεταγμένες (coordinates) των ρολογιών (Clocks).

4.2 Framework-Springboot

Οργάνωση και Αρχιτεκτονική Springboot:

Η αρχιτεκτονική της εφαρμογής βασίζεται στο πλαίσιο Spring Boot [1] και ακολουθεί τη δομή MVC (Model-View-Controller), που είναι μια διαδεδομένη διακριτική αρχιτεκτονική για εφαρμογές διαδικτύου αλλά και εφαρμογές για κινητά και tablets.

- **Controllers(Ελεγκτές)**:
  Οι κλάσεις στο πακέτο com.aquacount.aquacount.controller (Figure 7,10) παρέχουν τα σημεία εισόδου (endpoints) για την αλληλεπίδραση με το σύστημα. Αυτά τα σημεία εισόδου ανταποκρίνονται σε αιτήσεις HTTP και καθορίζουν πώς η εφαρμογή θα ανταποκριθεί.
  Οι controllers είναι οι εξής: CountersController , ClocksController , MeasurementController και RouteController.

  ClockController: Παρέχει τις κατάλληλες μεθόδους για την ανταλλαγή δεδομένων με το frontend, καλώντας τις αντίστοιχες μεθόδους του ClockService.

  CountersController: Παρέχει τις κατάλληλες μεθόδους για την ανταλλαγή δεδομένων με το frontend, καλώντας τις αντίστοιχες μεθόδους του CountersService.

  RouteController: Παρέχει τις κατάλληλες μεθόδους για την ανταλλαγή δεδομένων με το frontend, καλώντας τις αντίστοιχες μεθόδους του RouteService.

  MeasurementController: Παρέχει τις κατάλληλες μεθόδους για την ανταλλαγή δεδομένων με το frontend, καλώντας τις αντίστοιχες μεθόδους του MeasurementService.

- **Entities (Οντότητες)** : Οι κλάσεις RouteEntity, MeasurementEntity, CountersEntity, ClockEntity, που βρίσκονται στο πακέτο com.aquacount.aquacount.model (Figure 7), αντιπροσωπεύουν τα αντικείμενα που αποθηκεύονται στη βάση δεδομένων.

- **Data Transfer Objects (DTOs)** : Τα DTOs(Figure 7) χρησιμοποιούνται για τη μεταφορά δεδομένων μεταξύ του ελεγκτή (controller) και των υπηρεσιών (services) του backend, καθώς και μεταξύ του backend και του frontend.

  RegisterClock:Αντιπροσωπεύει μια εγγραφή ρολογιού στο σύστημα.
  Περιέχει τα πεδία counterid, clockid, και timestamp για να αποθηκεύσει πληροφορίες σχετικά με την καταμέτρηση και τον χρόνο καταγραφής.

  RegisterClockRequest:Χρησιμοποιείται για την αποστολή αιτημάτων εγγραφής ρολογίων.Περιέχει μια λίστα από αντικείμενα τύπου RegisterClock.

  RegisterMeasurement:Αντιπροσωπεύει μια εγγραφή μέτρησης στο σύστημα, περιέχοντας τα πεδία clockid, value και timestamp.

  RegisterMeasurementRequest:Χρησιμοποιείται για την αποστολή αιτημάτων εγγραφής μετρήσεων.Περιέχει μια λίστα από αντικείμενα τύπου RegisterMeasurement.

  RegisterRoute:Αντιπροσωπεύει μια εγγραφή δρομολογίου μεταξύ μετρητή και διαδρομής.

  RegisterRouteRequest:Χρησιμοποιείται για την αποστολή αιτημάτων εγγραφής δρομολογίων.Περιέχει μια λίστα από αντικείμενα τύπου RegisterRoute.

  LoginCounter:Αντιπροσωπεύει τα στοιχεία σύνδεσης ενός μετρητή.

  LoginResponse:Αντιπροσωπεύει την απάντηση επιτυχούς σύνδεσης, περιέχοντας το όνομα χρήστη και το JWT.

RegisterRequest:Χρησιμοποιείται για την αποστολή αιτημάτων εγγραφής νέου μετρητή.

RegisterResponse:Αντιπροσωπεύει την απάντηση επιτυχούς εγγραφής, περιέχοντας ένα μήνυμα επιτυχίας.

- **Repositories** :
  Οι κλάσεις στο πακέτο com.aquacount.aquacount.repository(Figure 7,10) παρέχουν πρόσβαση στη βάση δεδομένων για τις οντότητες, επιτρέποντας την αποθήκευση, την ανάγνωση και τη διαγραφή δεδομένων.

  ClockRepository:Παρέχει μέθοδο για την ανάκτηση ρολογιών βάσει του counterid και διαγραφή ρολογιών βάσει του clockid.

  CountersRepository:Παρέχει μέθοδο για την ανάκτηση μετρητών βάσει του ονόματος χρήστη (username) και διαγραφή μετρητών βάσει του counterid.

  MeasurementRepository:Παρέχει μέθοδο για την ανάκτηση μετρήσεων βάσει του clockid και διαγραφή μετρήσεων βάσει του id.

  RouteRepository:Παρέχει μέθοδο για την ανάκτηση δρομολογίων βάσει του counterid.

- **Services(Υπηρεσίες)** :
  Οι κλάσεις στο πακέτο com.aquacount.aquacount.service(Figure 7,10) περιέχουν την επιχειρησιακή λογική της εφαρμογής. Αυτές οι κλάσεις διαχειρίζονται την αλληλεπίδραση μεταξύ των ελεγκτών και των repositories, επιτυγχάνοντας την αντιμετώπιση των επιχειρησιακών απαιτήσεων της εφαρμογής.

  ClockService:Περιλαμβάνει λειτουργίες για τον χειρισμό των ρολογιών, όπως η εγγραφή, η ανάκτηση και η διαγραφή.

  CountersService:Περιλαμβάνει λειτουργίες για τη διαχείριση των μετρητών, όπως η ανάκτηση όλων των μετρητών, η ανάκτηση μετρητή βάσει του counterid, η εγγραφή νέου μετρητή, η ενημέρωση μετρητή και η διαγραφή μετρητή.

MeasurementService:Περιλαμβάνει λειτουργίες για τον χειρισμό των μετρήσεων, όπως η εγγραφή, η ανάκτηση, η ενημέρωση και η διαγραφή.

RouteService:Περιλαμβάνει λειτουργίες για τον χειρισμό των δρομολογίων, όπως η ανάκτηση όλων των δρομολογίων, η διαγραφή δρομολογίων και η προσθήκη δρομολογίων. Στο πακέτο αυτό συμπεριλαμβάνεται και το implementation των παραπάνω κλάσεων.

- **Security (Ασφάλεια)** : περιέχει κλάσεις που σχετίζονται με την ασφάλεια, όπως η κλάση JwtAuthenticationFilter, JwtUtil και SecurityConfig. Αυτές οι κλάσεις διαχειρίζονται την αυθεντικοποίηση και την εξουσιοδότηση των χρηστών μέσω του μηχανισμού JWT (JSON Web Token) όπως επίσης και τον διαχωρισμό τους σε απλούς χρήστες(users) και σε διαχειριστές(admins).

Όσο αναφορά τις σχέσεις των παραπάνω κλάσεων αυτές είναι οι εξής (Figure 8) :

Η κλάση MeasurementService διαχειρίζεται λειτουργίες σχετικές με τις μετρήσεις.

Υπάρχει σχέση ένα προς πολλά (1..*) με την κλάση MeasurementEntity, δεδομένου ότι μια υπηρεσία μέτρησης μπορεί να διαχειριστεί πολλές μετρήσεις.

Η κλάση MeasurementServiceImpl υλοποιεί το interface MeasurementService.Υπάρχει σχέση ένα προς ένα (1) με την κλάση MeasurementService.

Το MeasurementRepository παρέχει μεθόδους πρόσβασης στη βάση δεδομένων για τις μετρήσεις.

Υπάρχει σχέση ένα προς πολλά (1..*) με την κλάση MeasurementEntity, δεδομένου ότι μπορεί να ανακτηθούν πολλές μετρήσεις από τη βάση δεδομένων.

Η κλάση MeasurementEntity αντιπροσωπεύει τα δεδομένα μιας μετρητικής μονάδας.

Υπάρχει σχέση ένα προς ένα (1) με την κλάση MeasurementController, καθώς η μετρητική μονάδα χρησιμοποιείται από τον ελεγκτή.

Ο ελεγκτής MeasurementController επικοινωνεί με την υπηρεσία μετρήσεων MeasurementService για τη διαχείριση των μετρήσεων.

Υπάρχει σχέση ένα προς ένα (1) με την κλάση MeasurementService.

Ο ελεγκτής MeasurementController επικοινωνεί με την υπηρεσία CounterMeasurementsService για τη διαχείριση μετρητικών δεδομένων.

Υπάρχει σχέση ένα προς ένα (1) με την κλάση CounterMeasurementsService.

Η κλάση MeasurementController έχει μια σχέση ένα προς ένα (1) με την κλάση RegisterMeasurementRequest.

Αυτό υποδεικνύει ότι ο MeasurementController να χρησιμοποιεί ή να αναμένει ένα αντικείμενο τύπου RegisterMeasurementRequest.

Η κλάση MeasurementController έχει μια σχέση ένα προς ένα (1) με την κλάση RegisterMeasurementResponse.

Αυτό υποδεικνύει ότι ο MeasurementController παράγει ή να αναμένει ένα αντικείμενο τύπου RegisterMeasurementResponse.

Η κλάση RouteServiceImpl έχει μια σχέση ένα προς ένα (1) με την κλάση RouteRepository.

Αυτό υποδηλώνει ότι ο RouteServiceImpl χρησιμοποιεί την κλάση RouteRepository για την υλοποίηση των λειτουργιών του.

Η κλάση RouteRepository έχει μια σχέση ένα προς πολλά (1..*) με την κλάση RouteEntity.

Αυτό υποδηλώνει ότι ένα RouteRepository μπορεί να περιλαμβάνει πολλές εγγραφές τύπου RouteEntity.

Η κλάση RouteController έχει μια σχέση ένα προς ένα (1) με την κλάση RouteService.

Αυτό υποδηλώνει ότι ο RouteController χρησιμοποιεί την κλάση RouteService για την υλοποίηση των λειτουργιών του.

Η κλάση RouteController έχει μια σχέση ένα προς ένα (1) με την κλάση RegisterRouteRequest.

Αυτό υποδηλώνει ότι ο RouteController χρησιμοποιεί ή να αναμένει ένα αντικείμενο τύπου RegisterRouteRequest.

Η κλάση RouteController έχει μια σχέση ένα προς ένα (1) με την κλάση RegisterRoute.

Αυτό υποδηλώνει ότι ο RouteController χρησιμοποιεί ή να αναμένει ένα αντικείμενο τύπου RegisterRoute.

Η κλάση ClockController έχει μια σχέση ένα προς ένα (1) με την κλάση ClockService.

Αυτό υποδηλώνει ότι ο ClockController χρησιμοποιεί την κλάση ClockService για την υλοποίηση των λειτουργιών του.

Η κλάση ClockService έχει μια σχέση ένα προς ένα (1) με την κλάση ClockServiceImpl.

Αυτό υποδηλώνει ότι ο ClockService χρησιμοποιεί την κλάση ClockServiceImpl για την υλοποίηση των λειτουργιών του.

Η κλάση ClockServiceImpl έχει μια σχέση ένα προς ένα (1) με την κλάση ClockRepository.

Αυτό υποδηλώνει ότι ο ClockServiceImpl χρησιμοποιεί την κλάση ClockRepository για την υλοποίηση των λειτουργιών του.

Η κλάση ClockRepository έχει μια σχέση ένα προς πολλά (1..*) με την κλάση ClockEntity.

Αυτό υποδηλώνει ότι ένα ClockRepository μπορεί να περιλαμβάνει πολλές εγγραφές τύπου ClockEntity.

Η κλάση ClockController έχει μια σχέση ένα προς ένα (1) με την κλάση RegisterClockRequest.

Αυτό υποδηλώνει ότι ο ClockController χρησιμοποιεί ή να αναμένει ένα αντικείμενο τύπου RegisterClockRequest.

Η κλάση ClockController έχει μια σχέση ένα προς ένα (1) με την κλάση RegisterClock.

Αυτό υποδηλώνει ότι ο ClockController χρησιμοποιεί ή να αναμένει ένα αντικείμενο τύπου RegisterClock.

Η κλάση ClockEntity έχει μια σχέση ένα προς πολλά (1..*) με την κλάση MeasurementEntity.

Αυτό υποδηλώνει ότι μια εγγραφή τύπου ClockEntity μπορεί να συσχετίζεται με πολλές εγγραφές τύπου MeasurementEntity.

Η κλάση ClockEntity έχει μια σχέση ένα προς πολλά (1..*) με την κλάση RouteEntity.

Αυτό υποδηλώνει ότι μια εγγραφή τύπου ClockEntity μπορεί να συσχετίζεται με πολλές εγγραφές τύπου RouteEntity.

Η κλάση ClockEntity έχει μια σχέση ένα προς ένα (1) με την κλάση UserDetailsImpl.

Αυτό υποδηλώνει ότι μια εγγραφή τύπου ClockEntity μπορεί να συσχετίζεται με ένα αντικείμενο τύπου UserDetailsImpl.

Η κλάση ClockEntity έχει μια σχέση ένα προς ένα (1) με την κλάση CountersEntity.

Αυτό υποδηλώνει ότι μια εγγραφή τύπου ClockEntity μπορεί να συσχετίζεται με ένα αντικείμενο τύπου CountersEntity.

Σχέση μεταξύ CounterMeasurementsService και διαφόρων κλάσεων (MeasurementRepository, CounterRepository, ClockRepository, AuthService):

Η κλάση CounterMeasurementsService έχει μια σχέση ένα προς ένα (1) με κάθε μια από τις κλάσεις MeasurementRepository, CounterRepository, ClockRepository, και AuthService.

Αυτό υποδηλώνει ότι η κλάση CounterMeasurementsService χρησιμοποιεί όλες αυτές τις κλάσεις για την υλοποίηση των λειτουργιών της, συνδυάζοντας τις υπηρεσίες που παρέχουν αυτές οι κλάσεις.

Σχέση μεταξύ AuthService και των κλάσεων χρησιμότητας (Utility Classes):

Η κλάση AuthService έχει μια σχέση ένα προς ένα (1) με τις κλάσεις:

JwtUtil: Χρησιμοποιείται για τη δημιουργία, επικύρωση και ανάκτηση πληροφοριών από το JWT token.

UserDetailsImpl: Παρέχει πληροφορίες για τον χρήστη (π.χ., όπως το όνομα χρήστη και οι ρόλοι) που χρησιμοποιούνται κατά τη διάρκεια της αυθεντικοποίησης.

LoginCounter: Αντιπροσωπεύει τα στοιχεία του χρήστη που χρησιμοποιούνται κατά τη διάρκεια της είσοδου (login).

LoginResponse: Αντιπροσωπεύει την απάντηση της υπηρεσίας σύνδεσης, περιλαμβάνοντας πληροφορίες όπως το όνομα χρήστη και τον αναγνωριστικό του χρήστη.

LoginResponseWrapper: Είναι ένα περιτυλιγμένο για την απάντηση σύνδεσης που περιέχει επιπλέον πληροφορίες, αν χρειάζεται.

Η κλάση AuthService έχει μια σχέση ένα προς ένα (1) με την υλοποίηση της υπηρεσίας, AuthServiceImpl. Αυτό υποδηλώνει ότι η AuthService αξιοποιεί τις υπηρεσίες που παρέχονται από την AuthServiceImpl για την υλοποίηση των λειτουργιών της.

Σχέση μεταξύ AuthServiceImpl και διάφορων υπηρεσιών και αποθηκευτικών μέσων:

Η κλάση AuthServiceImpl έχει μια σχέση ένα προς ένα (1) με τις κλάσεις:

AuthenticationManager: Χρησιμοποιείται για τη διαχείριση των αιτημάτων πιστοποίησης.

CountersRepository: Πιθανόν χρησιμοποιείται για την ανάκτηση πληροφοριών σχετικά με τον χρήστη κατά τη διάρκεια της αυθεντικοποίησης.

JwtUtil: Χρησιμοποιείται για τη διαχείριση των JWT tokens κατά τη διάρκεια της αυθεντικοποίησης.

CountersRepository: χρησιμοποιείται για την ανάκτηση πληροφοριών σχετικά με τον χρήστη κατά τη διάρκεια της αυθεντικοποίησης.

Η κλάση JwtUtil έχει μια σχέση ένα προς ένα (1) με το CountersRepository. Αυτό υποδηλώνει ότι η JwtUtil χρησιμοποιεί το CountersRepository για την

επικύρωση πληροφοριών χρήστη κατά την ανάκτηση πληροφοριών από το JWT token.

Η κλάση CountersController έχει μια σχέση ένα προς ένα (1) με το CountersRepository. Αυτό υποδηλώνει ότι ο ελεγκτής χρησιμοποιεί το CountersRepository για την αλληλεπίδραση με τη βάση δεδομένων για θέματα που αφορούν τα στοιχεία των χρηστών.

Η κλάση CountersController έχει μια σχέση ένα προς ένα (1) με την υπηρεσία CounterMeasurementsService. Αυτό υποδηλώνει ότι ο ελεγκτής χρησιμοποιεί την υπηρεσία CounterMeasurementsService για τη διαχείριση στατιστικών στοιχείων που σχετίζονται με τους χρήστες.

Η κλάση CountersController έχει μια σχέση ένα προς ένα (1) με την κλάση RegisterRouteRequest. Αυτό υποδηλώνει ότι ο ελεγκτής χρησιμοποιεί το RegisterRouteRequest για την παραλαβή και διαχείριση αιτημάτων σχετικά με την εγγραφή νέων δρομολογίων.

Η κλάση RegisterClockRequest έχει μια σχέση ένα προς ένα (1) με την κλάση RegisterClock. Αυτό υποδηλώνει ότι η κλάση RegisterClockRequest περιλαμβάνει ή αλληλεπιδρά με ένα αντικείμενο τύπου RegisterClock.

Η κλάση RegisterMeasurementRequest έχει μια σχέση ένα προς ένα (1) με την κλάση RegisterMeasurement. Αυτό υποδηλώνει ότι η κλάση RegisterMeasurementRequest περιλαμβάνει ή αλληλεπιδρά με ένα αντικείμενο τύπου RegisterMeasurement.

Η κλάση RegisterRouteRequest έχει μια σχέση ένα προς ένα (1) με την κλάση RegisterRoute. Αυτό υποδηλώνει ότι η κλάση RegisterRouteRequest περιλαμβάνει ή αλληλεπιδρά με ένα αντικείμενο τύπου RegisterRoute.

Η κλάση CountersService έχει μια σχέση ένα προς ένα (1) με το CountersRepository. Αυτό υποδηλώνει ότι η υπηρεσία CountersService χρησιμοποιεί το CountersRepository για την αλληλεπίδραση με τη βάση δεδομένων κατά τη διαχείριση των χρηστών.

Η κλάση CountersServiceImpl έχει μια σχέση ένα προς ένα (1) με το CountersRepository. Αυτό υποδηλώνει ότι η υπηρεσία υλοποιείται από την υπηρεσία CountersRepository.

Η κλάση CountersServiceImpl έχει μια σχέση ένα προς ένα (1) με την υπηρεσία CounterService. Αυτό υποδηλώνει ότι η υπηρεσία CountersServiceImpl χρησιμοποιεί την υπηρεσία CounterService.

Η κλάση SecurityConfig έχει μια σχέση ένα προς ένα (1) με το JwtAuthenticatorFilter. Αυτό υποδηλώνει ότι η κατασκευή των φίλτρων ασφαλείας συμπεριλαμβάνει τον JwtAuthenticatorFilter.

Η κλάση SecurityConfig έχει μια σχέση ένα προς ένα (1) με την κλάση CounterMeasurements. Αυτό υποδηλώνει ότι η κλάση ασφαλείας εμπεριέχει κανόνες ασφαλείας σχετικά με τα στατιστικά στοιχεία.

Η κλάση SecurityConfig έχει μια σχέση ένα προς ένα (1) με το CounterMeasurementsController. Αυτό υποδηλώνει ότι οι ρυθμίσεις ασφαλείας επηρεάζουν τον τρόπο λειτουργίας του CounterMeasurementsController.

Η κλάση JwtAuthenticationFilter έχει μια σχέση ένα προς ένα (1) με το JwtUtil. Αυτό υποδηλώνει ότι το φίλτρο ελέγχου ταυτότητας χρησιμοποιεί το JwtUtil για τη διαχείριση των JWT.

Η κλάση JwtAuthenticationFilter έχει μια σχέση ένα προς ένα (1) με το CounterMeasurementsService. Αυτό υποδηλώνει ότι το φίλτρο ελέγχου ταυτότητας μπορεί να επικοινωνεί με την υπηρεσία μετρήσεων.

Η κλάση CounterMeasurements έχει μια σχέση ένα προς ένα (1) με το CounterMeasurementsController. Αυτό υποδηλώνει ότι οι μετρήσεις που συγκεντρώνονται μπορούν να επιστραφούν ή να ενημερωθούν μέσω του CounterMeasurementsController.

Η κλάση CounterMeasurements έχει μια σχέση ένα προς ένα (1) με το CounterMeasurementsService. Αυτό υποδηλώνει ότι οι μετρήσεις μπορούν να διαχειριστούν από την υπηρεσία μετρήσεων.

Η κλάση CountersEntity έχει μια σχέση ένα προς ένα (1) με το CounterMeasurementsService. Αυτό υποδηλώνει ότι η οντότητα του μετρητή συσχετίζεται με τις υπηρεσίες μέτρησης που αφορούν αυτόν τον μετρητή.

Η κλάση CountersEntity έχει μια σχέση ένα προς ένα (1) με το CounterMeasurementsServiceImpl. Αυτό υποδηλώνει ότι η οντότητα του μετρητή συσχετίζεται με την υπηρεσία υλοποίησης των μετρήσεων που αφορούν αυτόν τον μετρητή.

Η κλάση CountersEntity έχει μια σχέση ένα προς ένα (1) με το MeasurementEntity. Αυτό υποδηλώνει ότι η οντότητα του μετρητή συσχετίζεται με τα μετρήσιμα δεδομένα.

Η κλάση CountersEntity έχει μια σχέση ένα προς ένα (1) με το RouteEntity. Αυτό υποδηλώνει ότι η οντότητα του μετρητή συσχετίζεται με τα δεδομένα διαδρομής που αφορούν αυτόν τον μετρητή.

Η κλάση CountersEntity έχει μια σχέση ένα προς ένα (1) με το ClockEntity. Αυτό υποδηλώνει ότι η οντότητα του μετρητή συσχετίζεται με τις ώρες που αφορούν αυτόν τον μετρητή.

Τα παραπάνω απεικονίζονται δομικά στο διάγραμμα κλάσης (Figure 8) όπως επίσης και στο παρακάτω component διάγραμμα (Figure 11).

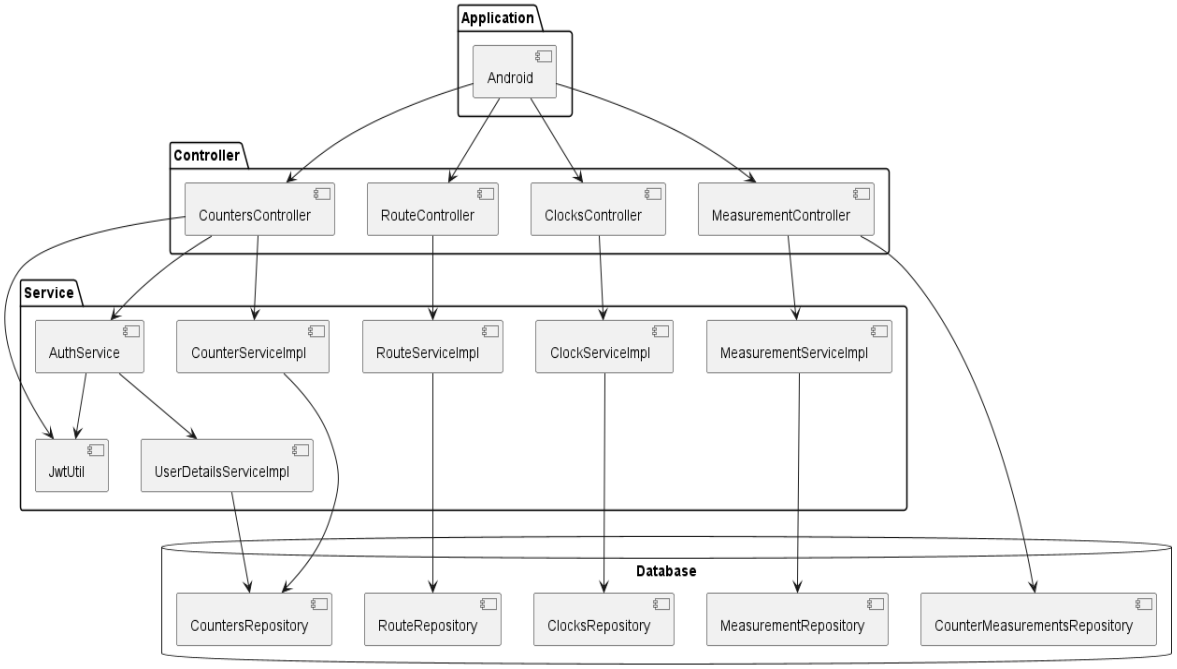

Figure 11- Component Diagram showing the components of backend (Springboot)

Στο παρακάτω διάγραμμα (Figure 12) ακολουθεί η διαδικασία που ακολουθεί ένας χρήστης για να συνδεθεί (login) σε ένα σύστημα, χρησιμοποιώντας μια υπηρεσία πιστοποίησης (AuthServiceImpl).

User Provides Credentials: Ο χρήστης παρέχει τα στοιχεία σύνδεσής του (όνομα χρήστη και κωδικό πρόσβασης).

Authenticate: Η υπηρεσία πιστοποίησης (AuthServiceImpl) ξεκινά τη διαδικασία πιστοποίησης καλώντας τον AuthenticationManager.

Get Username and Password: Ο AuthenticationManager ανακτά το όνομα χρήστη και τον κωδικό πρόσβασης από το UserCredentials.

Find By Username: Ο AuthenticationManager χρησιμοποιεί το CountersRepository για να βρεί τα στοιχεία του χρήστη βάσει του ονόματος χρήστη.

Generate Token: Αν ο χρήστης βρεθεί και τα στοιχεία είναι σωστά, το JwtUtil δημιουργεί ένα token.

Generate UserDetails: Το JwtUtil παράγει τα στοιχεία του χρήστη (user details) χρησιμοποιώντας το UserDetailsImpl.

Create UserDetails: Το UserDetailsImpl χρησιμοποιεί τα στοιχεία του χρήστη και δημιουργεί ένα αντικείμενο UserDetails.

Generate Token: Το JwtUtil παράγει το τελικό token με τα στοιχεία του χρήστη.

Token: Το AuthenticationManager επιστρέφει το token στην υπηρεσία πιστοποίησης (AuthServiceImpl).

Create LoginResponseWrapper: Το AuthServiceImpl δημιουργεί ένα αντικείμενο LoginResponseWrapper που περιλαμβάνει το token.

Response: Το AuthServiceImpl επιστρέφει το αντικείμενο LoginResponseWrapper στον χρήστη, παρέχοντας ένα επιτυχημένο μήνυμα σύνδεσης.

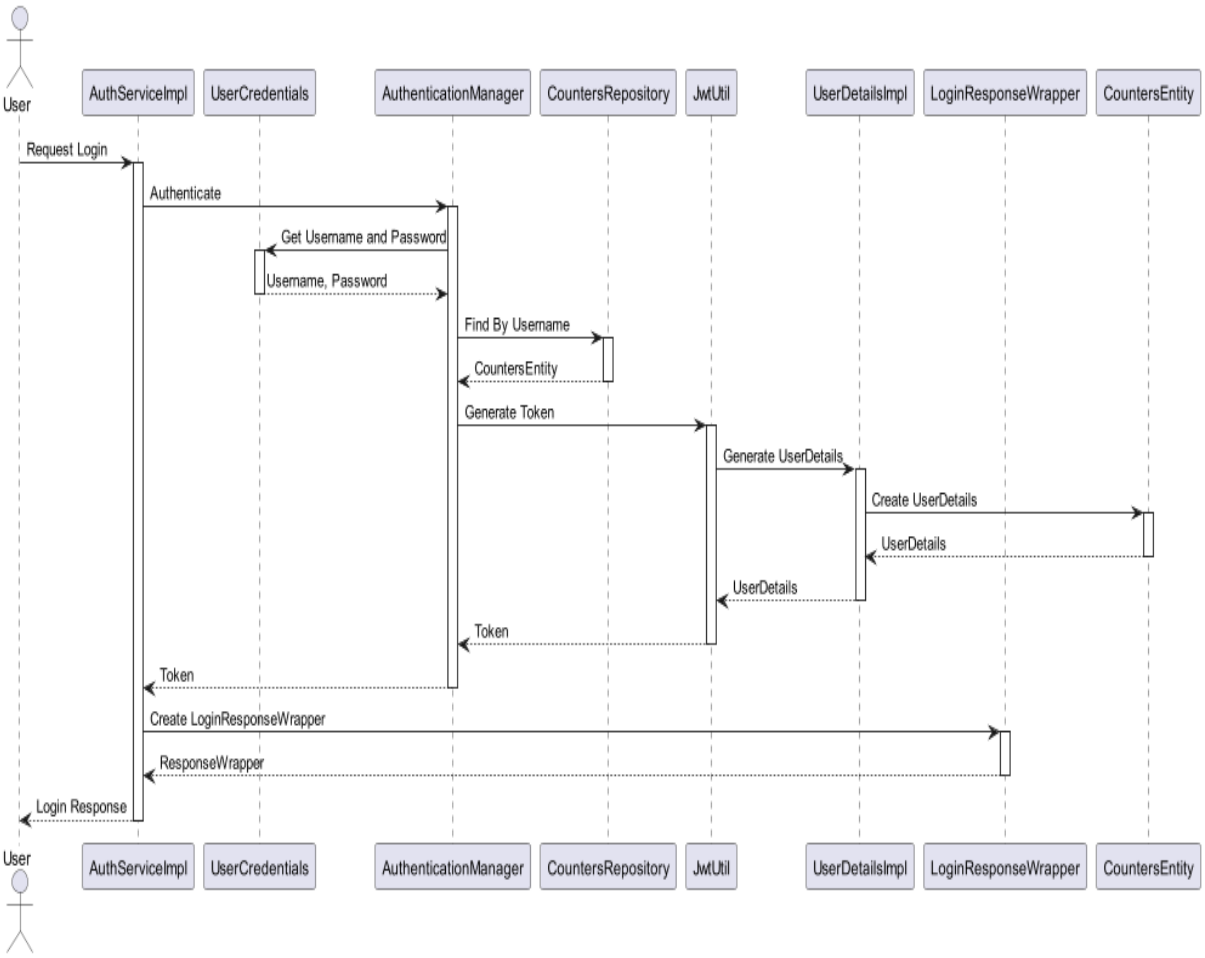

Figure 12 - Sequence Diagram with the process of User Authedication

Παρακάτω παρουσιάζονται κάποια sequence διαγράμματα που δείχνουν , αφού ο χρήστης συνδεθεί κάποιες από τις ενέργειες που μπορεί να ακολουθήσει:

Το παρακάτω διάγραμμα (Figure 13) απεικονίζει δύο δραστηριότητες που συμβαίνουν όταν ο χρήστης αλληλεπιδρά με το σύστημα προκειμένου να λάβει μετρήσεις (measurements) και να ενημερώσει την τιμή μιας μέτρησης.

Λήψη Μετρήσεων (Get Measurements):

Ο χρήστης αιτείται με τον MeasurementController να λάβει μετρήσεις.

Ο MeasurementController ενεργοποιείται και καλεί την υπηρεσία CounterMeasurementsService.

Η υπηρεσία CounterMeasurementsService ενεργοποιείται και αλληλεπιδρά με το CounterMeasurementsRepository για να βρει όλες τις μετρήσεις που σχετίζονται με έναν μετρητή.

Ο CounterMeasurementsRepository ανακτά τις μετρήσεις από τη βάση δεδομένων και τις επιστρέφει στην υπηρεσία CounterMeasurementsService.

Η υπηρεσία επιστρέφει τη λίστα με τις μετρήσεις στον MeasurementController.

Ο MeasurementController στέλνει τη λίστα με τις μετρήσεις στον χρήστη.

Ενημέρωση Τιμής Μέτρησης (Update Measurement Value):

Ο χρήστης αιτείται να ενημερώσει την τιμή μιας συγκεκριμένης μέτρησης.

Ο MeasurementController ενεργοποιείται και καλεί την υπηρεσία MeasurementService.

Η υπηρεσία MeasurementService ενεργοποιείται και αλληλεπιδρά με το MeasurementRepository για να βρει την μέτρηση με βάση το ID της.

Το MeasurementRepository επιστρέφει την μέτρηση.

Η υπηρεσία MeasurementService αλληλεπιδρά με το MeasurementRepository για να ενημερώσει την τιμή της μέτρησης.

Το MeasurementRepository επιστρέφει την ενημερωμένη μέτρηση.

Η υπηρεσία επιστρέφει την απόκριση στον MeasurementController.

Ο MeasurementController στέλνει την απόκριση στον χρήστη.

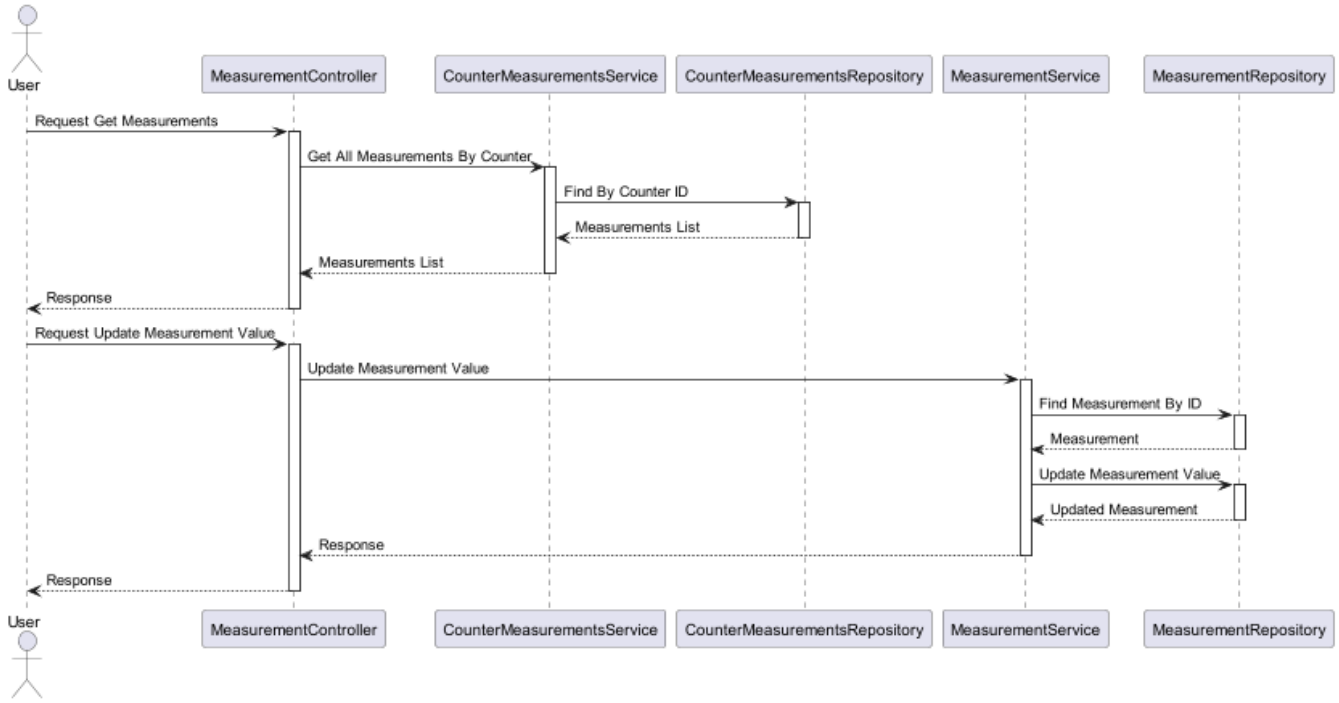

Figure 13 - Sequence Diagram showing the process a User make to update a Measurement value

Το παρακάτω διάγραμμα(Figure 14) απεικονίζει τη διαδικασία για τη λήψη συντεταγμένων (coordinates) για ένα συγκεκριμένο route. Εδώ είναι μια περιγραφή του διαγράμματος:

Λήψη Συντεταγμένων (Get Coordinates):

Ο χρήστης αιτείται να λάβει τις συντεταγμένες για ένα συγκεκριμένο route.

Ο ClockController ενεργοποιείται και καλεί την υπηρεσία ClockService για τη λήψη των συντεταγμένων.

Η υπηρεσία ClockService ενεργοποιείται και αλληλεπιδρά με το ClocksRepository για να βρει τις συντεταγμένες με βάση το Route ID.

Το ClocksRepository ανακτά τις συντεταγμένες ως συμβολοσειρές (Coordinate Strings) από τη βάση δεδομένων.

Η υπηρεσία ClockService καλεί τη μέθοδο Parse Coordinate Strings στην υπηρεσία ClockServiceImpl για να μετατρέψει τις συμβολοσειρές σε λίστα συντεταγμένων.

Η υπηρεσία επιστρέφει τη λίστα με τις συντεταγμένες στον ClockController.

Ο ClockController στέλνει την απόκριση με τις συντεταγμένες στον χρήστη.

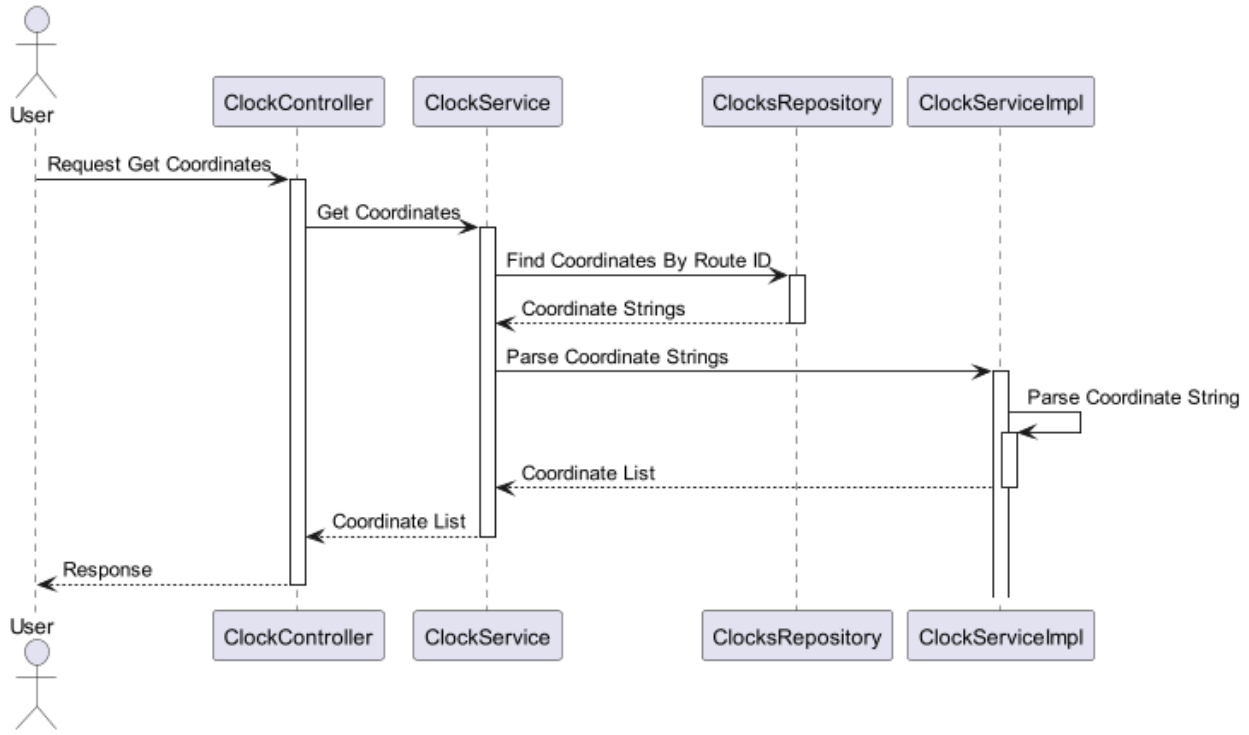

Figure 14 - Sequence Diagram with the process a User makes to get Coordinates

Το παρακάτω διάγραμμα (Figure 15) αναπαριστά τη διαδικασία εγγραφής (register) μετρήσεων (measurements). Εδώ είναι μια περιγραφή του διαγράμματος:
Αίτηση Εγγραφής Μέτρησης (Request Register Measurement):
Ο χρήστης υποβάλλει αίτηση για την εγγραφή νέας μέτρησης.
Ο MeasurementController ενεργοποιείται και καλεί την υπηρεσία MeasurementServiceImpl για την εγγραφή των μετρήσεων.
Εγγραφή Μετρήσεων (Register Measurement):
Η υπηρεσία MeasurementServiceImpl ενεργοποιείται και εκκινεί ένα loop για κάθε μέτρηση που υποβάλλεται.
Κάθε μέτρηση αποθηκεύεται στο MeasurementRepository.
Αφού η μέτρηση αποθηκευτεί με επιτυχία, η αποθηκευμένη μέτρηση επιστρέφεται στην υπηρεσία MeasurementServiceImpl.

Η υπηρεσία MeasurementServiceImpl συνεχίζει το loop για τις υπόλοιπες μετρήσεις.
Απόκριση στον Χρήστη (Response to User):
Όταν ολοκληρωθεί η διαδικασία εγγραφής όλων των μετρήσεων, η υπηρεσία MeasurementServiceImpl στέλνει μια απόκριση στον MeasurementController.
Ο MeasurementController στέλνει την τελική απόκριση στον χρήστη.

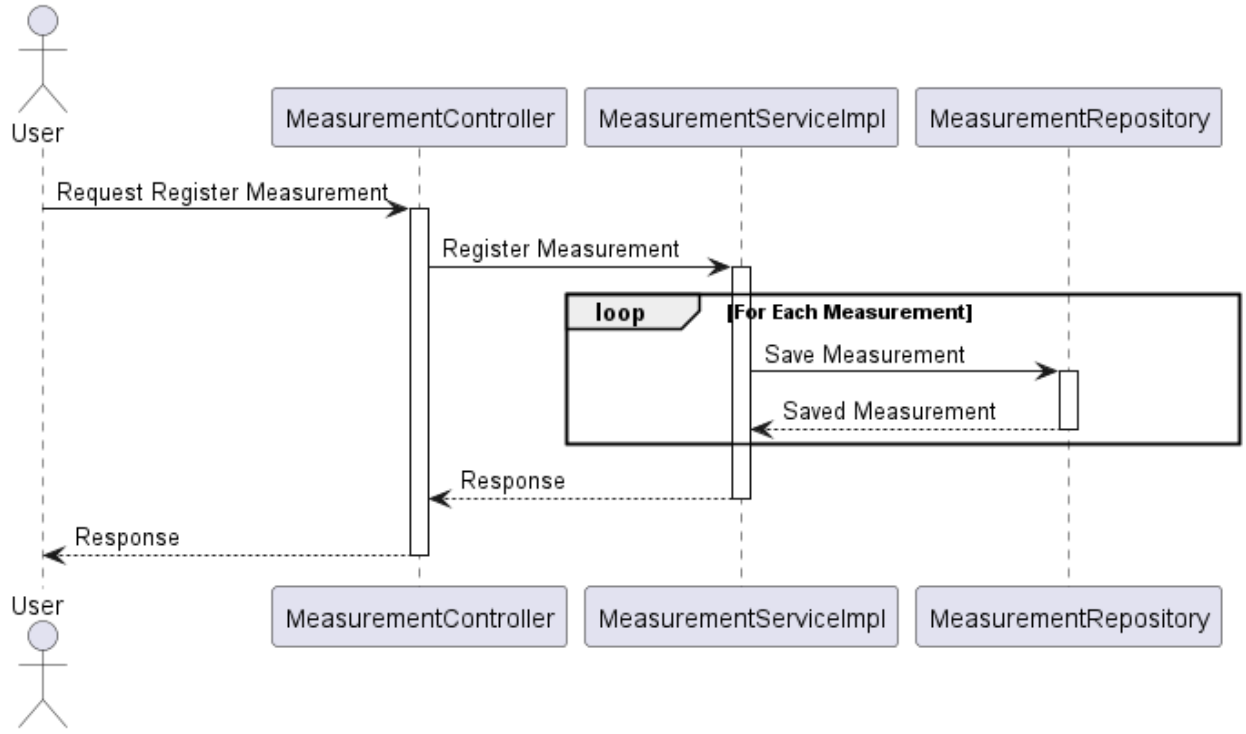

Figure 15 - Sequence Diagram with the process a User makes to register a new Measurement value

4.3 Application-Android Studio

Υλοποίηση Android Studio(front-end):

Στην προκειμένη περίπτωση χρησιμοποιήθηκε η γλώσσα προγραμματισμού Java.

1. Η εφαρμογή αποτελείται από ένα σύνολο δραστηριοτήτων (activities) που συνεργάζονται για τη διευκόλυνση της παρακολούθησης και διαχείρισης δεδομένων. Κάθε δραστηριότητα έχει τον δικό της σκοπό και συνέβαλε στη δημιουργία μιας ολοκληρωμένης εμπειρίας χρήστη. Οι δραστηριότητες είναι οι εξής:

   - **LoginActivity**
     Το LoginActivity χρησιμοποιείται για τη σύνδεση του χρήστη. Στο onCreate, αρχικοποιεί τα πεδία κειμένου και το κουμπί, και χειρίζεται το κουμπί κατά το πάτημά του. Όταν η σύνδεση είναι επιτυχής, αποθηκεύεται το JWT token και άλλες πληροφορίες χρήστη χρησιμοποιώντας τον TokenManager.

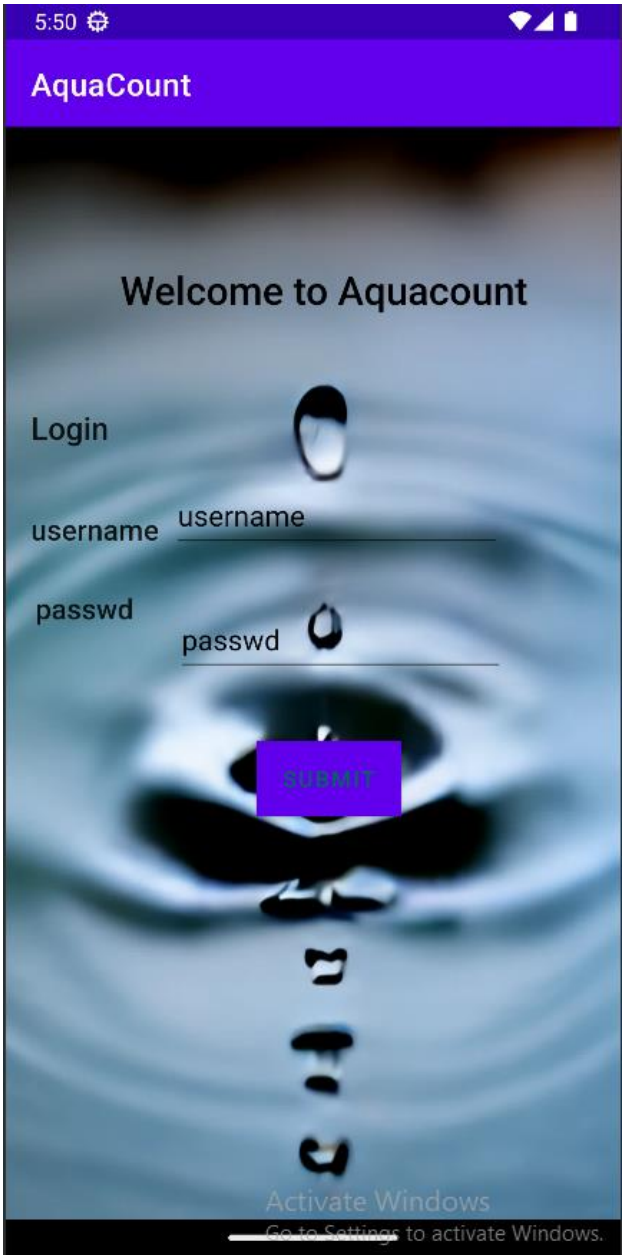

Figure 16 - AquaCount App Login

- **OptionsActivity**
  Το OptionsActivity αντιπροσωπεύει την κύρια οθόνη επιλογών της εφαρμογής. Στο onCreate, τα κουμπιά και το κείμενο εμφανίζονται στην οθόνη. Το κείμενο εμφανίζει το όνομα του χρήστη από τα δεδομένα αυθεντικοποίησης. Έχει κουμπιά που παραπέμπουν σε άλλες δραστηριότητες, όπως το MenuActivity, το DisplayEntriesActivity και το InputActivity.

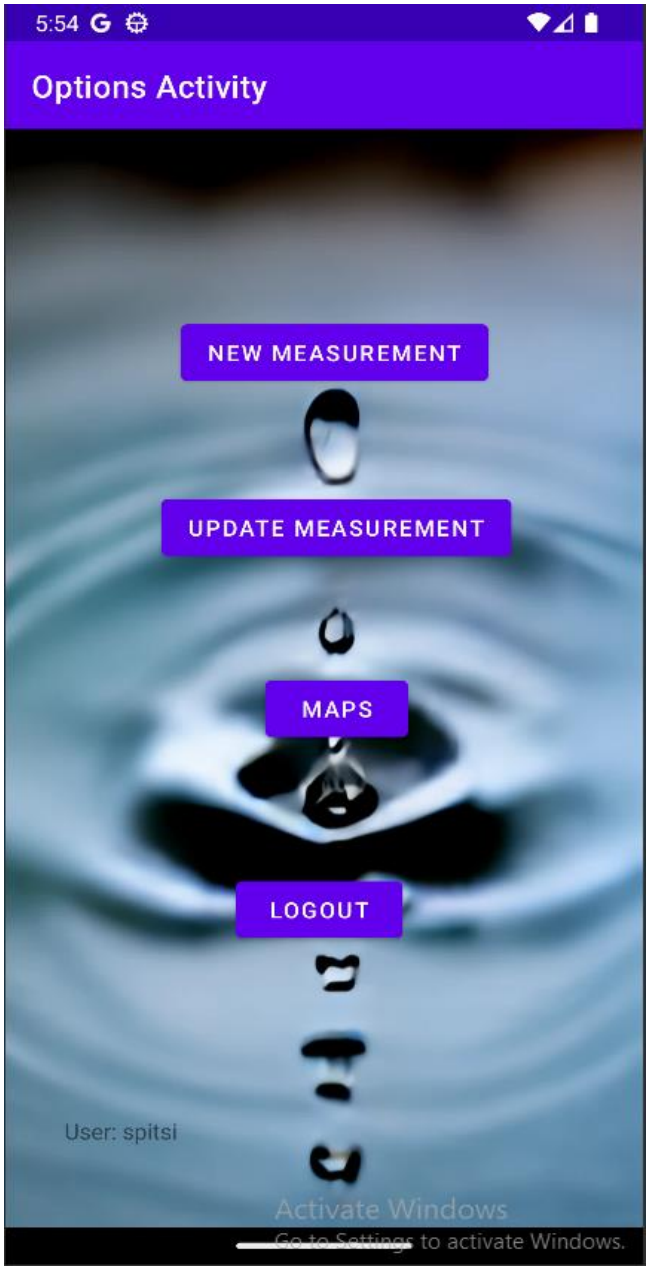

Figure 17 - List of options a User has in app

- **MapsActivity**
  Το MapsActivity διαχειρίζεται τον χάρτη Google Maps και την εμφάνιση των συντεταγμένων στο χάρτη. Στο onCreate, αρχικοποιεί τον χάρτη και τις παραμέτρους του, ενώ στο onMapReady, προσθέτει δείκτες στον χάρτη. Χρησιμοποιεί τις συντεταγμένες για να υπολογίσει και να εμφανίσει τη βέλτιστη διαδρομή με πολλαπλά waypoints.
  Επίσης στην κλάση αυτή χρησιμοποιείται η συνάρτηση “calculateAndDisplayOptimalRoute”, η οποία χρησιμοποιώντας το Google Maps Directions Api, υπολογίζει την βέλτιστη διαδρομή που πρέπει να ακολουθήσει ο χρήστης έτσι ώστε να περάσει από όλα τα ρολόγια του route και να κάνει τις μετρήσεις του.

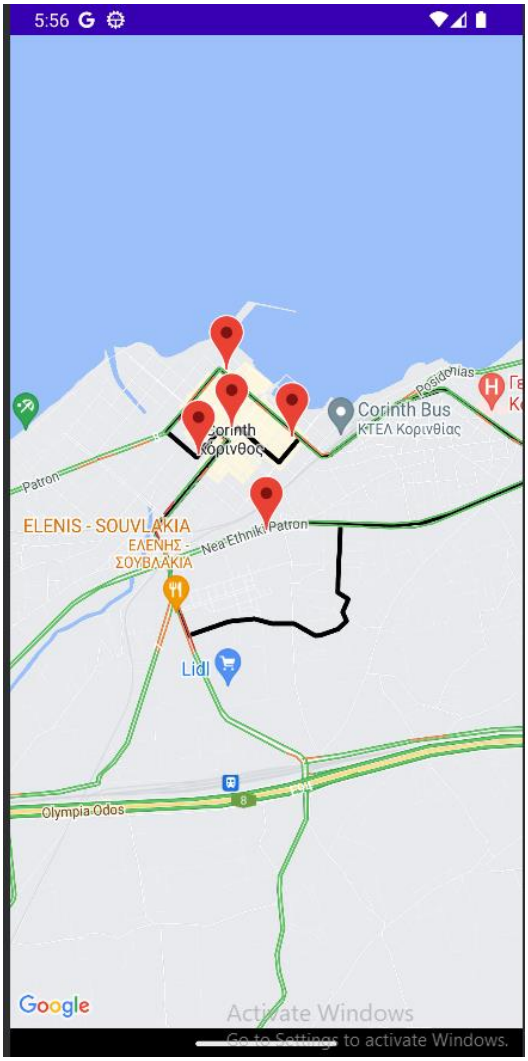

Figure 18 - The points of Clocks in map

- **MenuActivity**
  Το MenuActivity δίνει στον χρήστη τη δυνατότητα να καταχωρήσει μετρήσεις. Χρησιμοποιεί το Retrofit για την αποστολή δεδομένων προς τον διακομιστή. Επίσης, ανακτά και προβάλλει Clock IDs, προκειμένου ο χρήστης να επιλέξει προς ποιό ρολόι θα καταχωρήσει τα δεδομένα και συγκεκριμένα το σύστημα λαμβάνει μόνο τα ρολόγια που ανήκουν στην διαδρομή που του έχει ανατεθεί και έχει την δυνατότητα να επιλέξει ένα από αυτά όπως φαίνεται στο (Figure 18,19).Χρησιμοποιεί τον NetworkChangeReceiver για να ανιχνεύει αλλαγές στη σύνδεση και να αποστέλλει τα δεδομένα από τη βάση δεδομένων SQLite όταν υπάρχει σύνδεση.

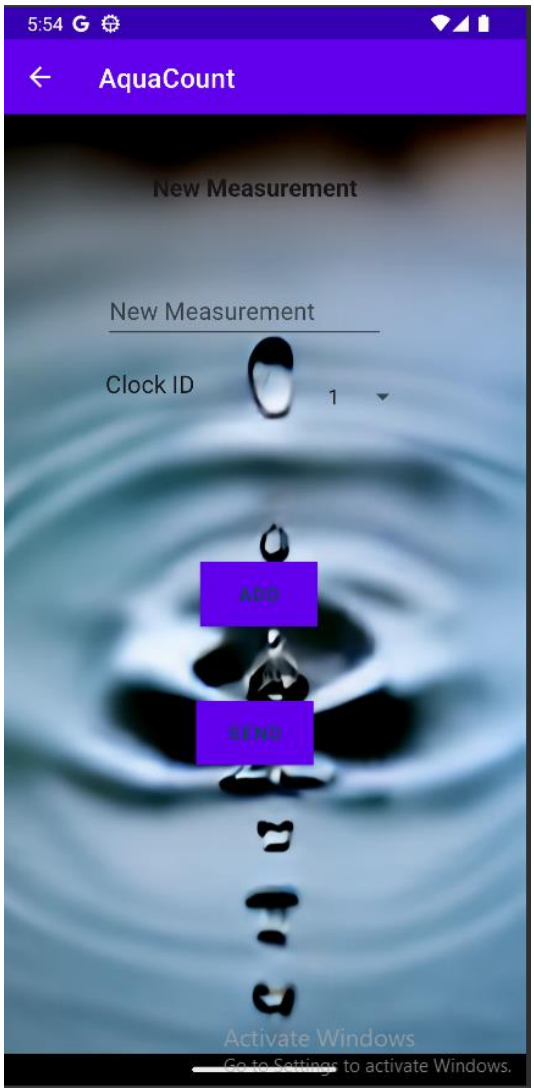

Figure 19 - Add a new Measurement

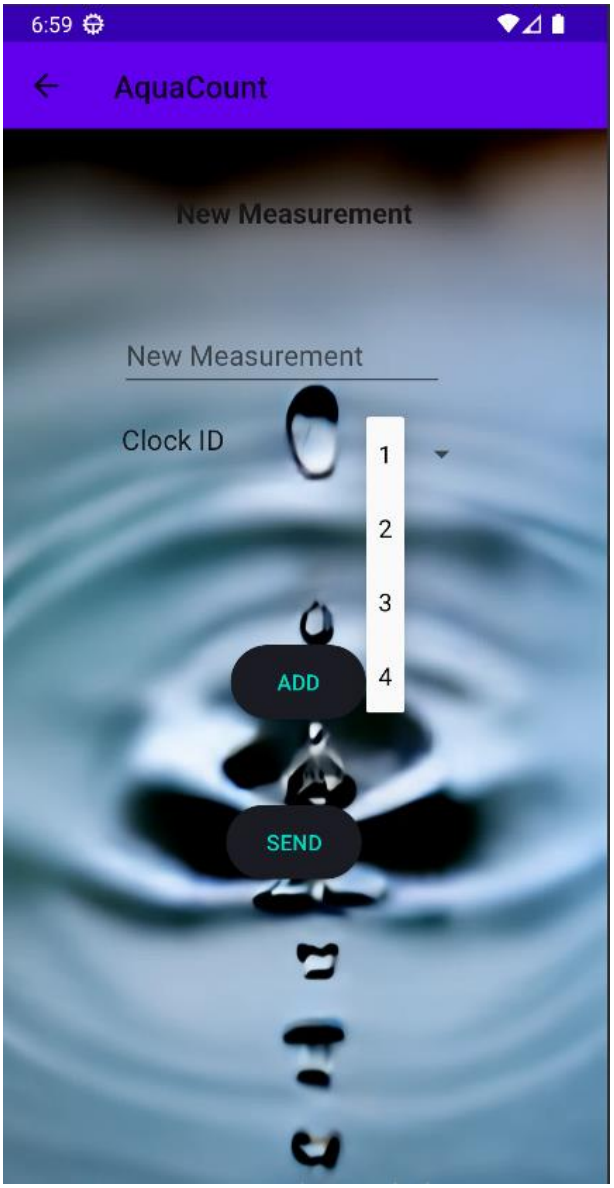

Figure 20- Counter can only take measurements of clocks that belong to his route

- **InputActivity**
  Το InputActivity είναι υπεύθυνο για την εισαγωγή του αριθμού routeId από τον χρήστη.Εάν ο χρήστης εισάγει έγκυρο αριθμό, δημιουργείται ένα Intent και ο χρήστης προωθείται στο DisplayCoordinatesActivity.
- **DisplayEntriesActivity**
  Το DisplayEntriesActivity εμφανίζει τις μετρήσεις για ένα συγκεκριμένο counterId.Κάνει χρήση του Retrofit για να αλληλεπιδρά με έναν διακομιστή REST API και εμφανίζει τα δεδομένα στο UI.Υπάρχει η δυνατότητα επεξεργασίας των μετρήσεων.

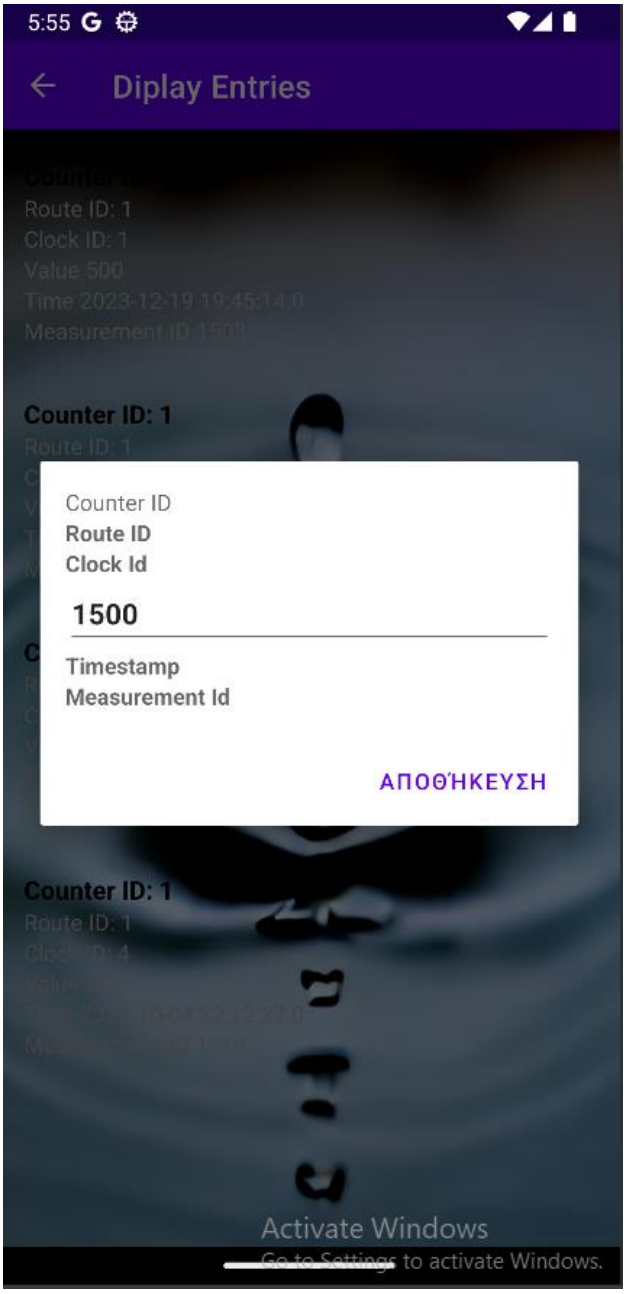

Figure 21 - Update the value of existent Measurement

- **DisplayCoordinatesActivity**
  Το DisplayCoordinatesActivity εμφανίζει τις συντεταγμένες για ένα συγκεκριμένο routeId.

Figure 22 - Projection of the clocks coordinates

Κάνει χρήση του Retrofit για να ανακτήσει τις συντεταγμένες από έναν διακομιστή REST API [12].Προσφέρει τη δυνατότητα προβολής αυτών των συντεταγμένων σε έναν χάρτη.

2. Η εφαρμογή αποτελείται επίσης και από μοντέλα(models) τα οποία εξυπηρετούν στην αποδοτική ανταλλαγή δεδομένων ανάμεσα στο Android Studio [3] και στο Springboot [1].

Τα μοντέλα είναι τα εξής :

- **UserCredentials**
  Χρησιμοποιείται για την αποστολή συνταγών χρήστη (όπως όνομα χρήστη και κωδικό πρόσβασης) προκειμένου να γίνει αυθεντικοποίηση μέσω των requests προς το Spring Boot.
- **RouteEntity**
  Αντιπροσωπεύει τα δεδομένα ενός δρομολογίου. Χρησιμοποιείται για τη μεταφορά του routeId και του counterId μεταξύ του Android Studio και του Spring Boot.
- **MeasurementUpdateEntity**
  Χρησιμοποιείται για τη μεταφορά δεδομένων μετρήσεων. Περιλαμβάνει τιμές όπως counterid, routeid, clockid, value, timestamp, και measurementid. Χρησιμοποιείται στο Android Studio για την αποστολή και λήψη μετρήσεων από το Spring Boot.
- **MeasurementModel**
  Παρέχει ένα μοντέλο για τις μετρήσεις. Χρησιμοποιείται για την απεικόνιση δεδομένων μετρήσεων στο Android Studio.
- **MeasurementEntity**
  Παρέχει ένα μοντέλο για τις μετρήσεις με επιπλέον πεδία όπως το id. Αυτό το μοντέλο μπορεί να χρησιμοποιηθεί στην αλληλεπίδραση με το Spring Boot.
- **LoginResponse**
  Χρησιμοποιείται για την αντιπροσώπευση των δεδομένων απάντησης κατά την είσοδο του χρήστη. Περιέχει πληροφορίες όπως το username και το counterid.
- **CoordinateEntity**
  Χρησιμοποιείται για τη μεταφορά δεδομένων συντεταγμένων. Τα δεδομένα αυτά χρησιμοποιούνται για την απεικόνιση διαδρομών στον χάρτη στο Android Studio.

3. Για την σωστή και ομαλή λειτουργία της εφαρμογής χρησιμοποιούνται και κάποιες επιπλέον κλάσεις οι οποίες εξυπηρετούν τη λειτουργικότητα της.
Συγκεκριμένα:

- **RetrofitAPI**
  Αυτή η διεπαφή περιγράφει τα endpoints που χρησιμοποιούνται για την ανταλλαγή δεδομένων μεταξύ του Android Studio και του Spring Boot. Κάθε μέθοδος αντιπροσωπεύει ένα συγκεκριμένο τύπο αίτησης (POST, GET, PUT) και τα αντίστοιχα δεδομένα που στέλνονται ή λαμβάνονται.
- **AuthInterceptor**
  Αυτή η κλάση είναι ένας interceptor για το OkHttpClient και προσθέτει το JWT token στην κεφαλίδα "Authorization" για κάθε αίτηση που στέλνεται προς το Spring Boot. Εξασφαλίζει ότι ο χρήστης είναι σωστά αυθεντικοποιημένος.
- **CoordinateAdapter**
  Αυτός ο Adapter χρησιμοποιείται για την εμφάνιση των συντεταγμένων σε ένα RecyclerView. Αναλαμβάνει τον χειρισμό της λογικής που αφορά τον τρόπο που παρουσιάζονται τα δεδομένα.
- **MeasurementAdapter**
  Αυτός ο Adapter χρησιμοποιείται για την εμφάνιση των μετρήσεων σε ένα RecyclerView. Επίσης, αναλαμβάνει τον χειρισμό της λογικής για τον τρόπο παρουσίασης των δεδομένων και για τη δυνατότητα αλληλεπίδρασης του χρήστη.
- **NetworkChangeReceiver**
  Αυτός ο δέκτης (Receiver) είναι υπεύθυνος για την παρακολούθηση της κατάστασης της σύνδεσης δικτύου και εκτελεί ενέργειες ανάλογα με την κατάσταση του δικτύου. Επίσης, περιλαμβάνει μια μέθοδο για την αποστολή δεδομένων που δεν απεστάλησαν λόγω έλλειψης σύνδεσης.
- **NetworkCheck**
  Αυτή η κλάση περιέχει έναν απλό έλεγχο για τη διαθεσιμότητα της σύνδεσης δικτύου. Χρησιμοποιείται για να ελέγξει αν η συσκευή έχει σύνδεση στο διαδίκτυο.

4. Για την λειτουργία του **UI** (User Interface) δημιουργήθηκαν .xml [13] αρχεία.Κάθε δραστηριότητα(activity) έχει το δικό της .xml αρχείο , όπως επίσης και τα items ή κάποια dialogs που χρειάστηκαν έχουν και αυτά το απαραίτητο .xml αρχείο.

5. Για τη σωστή λειτουργία της ανταλλαγής δεδομένων από το android studio στο springboot και για την λειτουργικότητα της εφαρμογής και offline δημιουργήθηκε στο κινητό μια **SQLite** [14] βάση δεδομένων στην οποία αποθηκεύονται αρχικά οι μετρήσεις που κάνει ο χρήστης και όταν βρεί δίκτυο wifi ή δεδομένα τότε αποστέλλονται στον κεντρικό server μέσω της επικοινωνίας springboot και android studio.Στην συνέχεια οι καταχωρήσεις που έχουν σταλεί στον server διαγράφονται.

Για την υλοποίηση αυτής της λειτουργίας χρειάστηκε η δημιουργία μίας κλάσης με όνομα 'DbManager' η οποία είναι υπεύθυνη για την δημιουργία ενός table , το οποίο θα μπορεί να πάρει από τον χρήστη τις απαραίτητες εισαγωγές που θα κάνει και να τις βάλει στο συγκεκριμένο table , όπως επίσης περιέχει και την μέθοδο για να ληφθούν αυτές οι τιμές από την βάση και τέλος αφού έχουν μεταφερθεί στον κεντρικό server υπάρχει και η μέθοδος που διαγράφει όλες τις προηγούμενες καταχωρήσεις και έτσι διασφαλίζεται ο κίνδυνος της πανωγραφίας.

Όσον αφορά , τις σχέσεις των παραπάνω κλάσεων όπως μπορούμε να δούμε και στην φωτογραφία (Figure 8) είναι οι εξής :

Η DisplayCoordinatesActivity έχει 1 προς 1 σχέση με το TokenManager. Αυτό σημαίνει ότι κάθε ενέργεια εμφάνισης συντεταγμένων συνδέεται με ένα και μόνο TokenManager.

Η DisplayCoordinatesActivity έχει 1 προς 1 σχέση με το RetrofitAPI. Κάθε ενέργεια εμφάνισης συντεταγμένων συνδέεται με ένα συγκεκριμένο αντικείμενο RetrofitAPI.

Η DisplayCoordinatesActivity έχει 1 προς 1 σχέση με το AuthInterceptor. Κάθε ενέργεια εμφάνισης συντεταγμένων συνδέεται με ένα συγκεκριμένο αντικείμενο AuthInterceptor.

Η DisplayCoordinatesActivity έχει 1 προς 1 σχέση με το CoordinateAdapter. Κάθε ενέργεια εμφάνισης συντεταγμένων χρησιμοποιεί ένα συγκεκριμένο αντικείμενο CoordinateAdapter.

Η DisplayCoordinatesActivity έχει 1 προς πολλά σχέση με το CoordinateEntity. Κάθε ενέργεια εμφάνισης συντεταγμένων συσχετίζεται με πολλά αντικείμενα CoordinateEntity.

Το CoordinateAdapter χρησιμοποιεί το CoordinateEntity. Κάθε αντικείμενο CoordinateAdapter αντιστοιχεί σε ένα και μόνο αντικείμενο CoordinateEntity.

Η MapsActivity έχει 1 προς πολλά σχέση με το CoordinateEntity. Κάθε δραστηριότητα χαρτών συσχετίζεται με πολλά αντικείμενα CoordinateEntity.

Το RetrofitAPI χρησιμοποιεί το CoordinateEntity. Κάθε αντικείμενο RetrofitAPI συσχετίζεται με ένα και μόνο αντικείμενο CoordinateEntity.

Το RetrofitAPI έχει 1 προς 1 σχέση με το LoginResponse. Κάθε αίτημα προς το RetrofitAPI από το LoginActivity αποκτά ένα και μόνο αντικείμενο LoginResponse.

Το LoginActivity έχει 1 προς 1 σχέση με το LoginResponse. Κάθε δραστηριότητα σύνδεσης αντιστοιχεί με ένα συγκεκριμένο αντικείμενο LoginResponse.

Το RetrofitAPI έχει 1 προς 1 σχέση με το MeasurementEntity. Κάθε αίτημα προς το RetrofitAPI από το DisplayEntriesActivity αποκτά ένα και μόνο αντικείμενο MeasurementEntity.

Η DisplayEntriesActivity έχει 1 προς πολλά σχέση με το MeasurementEntity. Κάθε ενέργεια εμφάνισης καταγραφών συσχετίζεται με πολλά αντικείμενα MeasurementEntity.

Η MenuActivity έχει 1 προς πολλά σχέση με το MeasurementModel. Κάθε ενέργεια στο μενού συσχετίζεται με πολλά αντικείμενα MeasurementModel.

Ο DbManager έχει 1 προς πολλά σχέση με το MeasurementModel. Κάθε διαχείριση βάσης δεδομένων συσχετίζεται με πολλά αντικείμενα MeasurementModel.

Το αίτημα μέτρησης έχει 1 προς 1 σχέση με το MeasurementModel. Κάθε αίτημα μέτρησης συνδέεται με ένα και μόνο αντικείμενο MeasurementModel.

Το MeasurementAdapter έχει 1 προς 1 σχέση με το MeasurementUpdateEntity. Κάθε προσαρμογέας μέτρησης αντιστοιχεί με ένα συγκεκριμένο αντικείμενο MeasurementUpdateEntity.

Το RetrofitAPI έχει 1 προς 1 σχέση με το MeasurementUpdateEntity. Κάθε αίτημα προς το RetrofitAPI για ενημέρωση μέτρησης αποκτά ένα και μόνο αντικείμενο MeasurementUpdateEntity.

Η DisplayEntriesActivity διατηρεί πολλές σχετικές εγγραφές MeasurementUpdateEntity (ένα προς πολλά), όπου ένα MeasurementUpdateEntity μπορεί να σχετίζεται με πολλές εγγραφές της DisplayEntriesActivity.

Η RetrofitAPI συνδέεται με μία RouteEntity (ένα προς ένα), υποδεικνύοντας ότι μία RetrofitAPI είναι συνδεδεμένη με μια μόνο RouteEntity.

Η MenuActivity συνδέεται με πολλές RouteEntity (ένα προς πολλά), υποδεικνύοντας ότι μια MenuActivity μπορεί να έχει σχέση με πολλές RouteEntity.

Η LoginActivity συνδέεται με μία UserCredentials (ένα προς ένα), υποδεικνύοντας ότι κάθε LoginActivity συσχετίζεται με ένα μόνο σύνολο διαπιστευτηρίων χρήστη.

Η UserRequest έχει μια σχέση 1 προς 1 με τα UserCredentials, υποδεικνύοντας ότι ένα αίτημα χρήστη αντιστοιχεί σε ένα σύνολο διαπιστευτηρίων.

Τα παραπάνω απεικονίζονται δομικά στον διάγραμμα κλάσεων (Figure 9) όπως επίσης και στο component diagram (Figure 23).

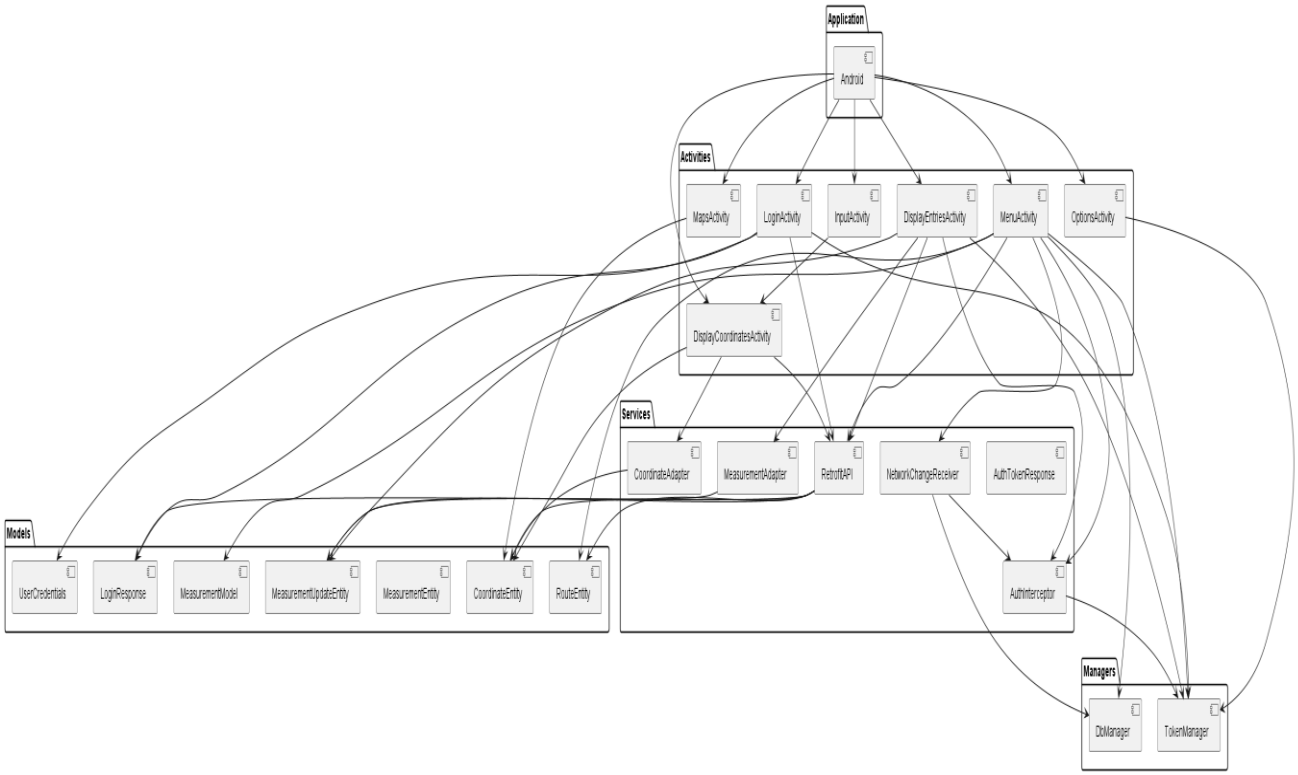

Figure 23 - Component Diagram of Android Application

4.4 Admin Dashboard

Βασιζόμενοι στην Typescript επιλέχθηκε το framework Angular, το οποίο έχει πλήρη υποστήριξη για Typescript προσφέροντας μοντέρνο και εύχρηστο UI με την χρήση της Angular Material βιβλιοθήκης. Επιπροσθέτως, η Angular μπορεί να χρησιμοποιήσει άριστα και την βιβλιοθήκη rxjs, η οποία περιλαμβάνει αντιδραστικά στοιχεία (reactive observables), υπεύθυνα για την ασύγχρονη ανανέωση των παρατηρήσιμων στοιχείων (asynchronous observables) που χρειαστήκαμε για την υλοποίησή μας.

Με την επιλογή της Angular, ακολουθήσαμε πιστά και την λογική της, δηλαδή τη δημιουργία συστατικών (components), ένα για κάθε λογική οντότητα. Με τον όρο Angular Components αναφερόμαστε στα δομικά στοιχεία μιας εφαρμογής που καθορίζουν διαφορετικές πτυχές της διεπαφής χρήστη. Είναι ένα directive με ένα πρότυπο που επιτρέπει την δημιουργία στοιχείων μιας διεπαφής χρήστη σε μια εφαρμογή Angular. Τα στοιχεία θα έχουν πάντα ένα πρότυπο, έναν επιλογέα και μπορεί να έχουν ή να μην έχουν ξεχωριστό ύφος και εμφάνιση. Κάθε Angular Component αποτελείται από:

- Ένα πρότυπο HTML που δηλώνει τι αποδίδεται στη σελίδα.
- Μια κλάση TypeScript που ορίζει τη συμπεριφορά.
- Έναν επιλογέα CSS που ορίζει πώς χρησιμοποιείται το στοιχείο σε ένα πρότυπο.
- Ένα αρχείο spec.ts για testing.

4.4.1 Προσέγγιση Σχεδιασμού

Με βάση τα παραπάνω, δημιουργήσαμε τα Angular Components που ακολουθούν:

routes.component : υλοποιεί τον έλεγχο και την προβολή των δρομολογίων (routes) στην εφαρμογή AquaCounter. Παρακάτω περιγράφονται η λειτουργία του.

Εμφάνιση Δρομολογίων: Το component εμφανίζει έναν πίνακα με τα δρομολόγια που ανακτούνται από το back-end.

Εμφάνιση ρολογιών: Ο κώδικας επιτρέπει την εμφάνιση των id των ρολογιών για κάθε δρομολόγιο όταν ο χρήστης πατάει το κουμπί "Show Clocks" (Figure 24). Αυτό γίνεται με την βοήθεια ενός άλλου component που ονομάζεται routes-dialog.component.

Προσθήκη Δρομολογίου: Ο χρήστης μπορεί να προσθέσει ένα νέο δρομολόγιο εισάγοντας τις τιμές του routeId και counterId και πατώντας το κουμπί "Save".

Διαγραφή Δρομολογίου: Ο χρήστης μπορεί να διαγράψει ένα δρομολόγιο πατώντας το κουμπί "Delete".

Επεξεργασία Δρομολογίου: Ο χρήστης πατώντας το κουμπί για επεξεργασία , μπορεί να διαχειριστεί ήδη αποθηκευμένα δρομολόγια , αλλάζοντας τους το counterid και πατώντας save.Στην ουσία έτσι μπορεί να αναθέσει κάποιο δρομολόγιο σε άλλο μετρητή.

Εμφάνιση ρολογιών στον χάρτη : Ο διαχειριστής έχει την δυνατότητα πατώντας το κουμπι «Show clocks on map» να δει τα ρολόγια της κάθε διαδρομής στον χάρτη(Figure 24)(routes-dialog-map.component).

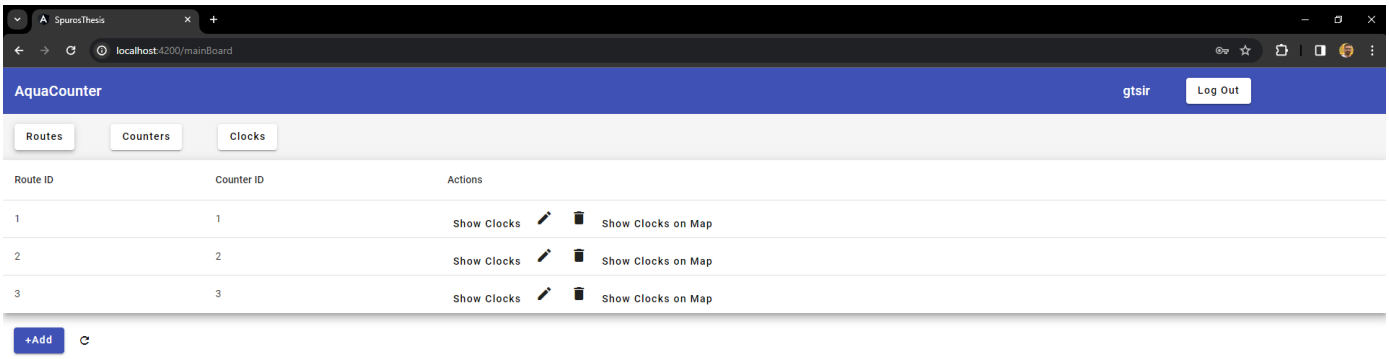

Figure 24 - Routes display in the Admin Dashboard

routes-dialog-map.component : η κλάση αυτή δημιουργεί και διαχειρίζεται έναν χάρτη με τη χρήση της βιβλιοθήκης Leaflet.Ο χάρτης αυτός χρησιμοποιείται για την εμφάνιση των ρολογιών συγκεκριμένων διαδρομών(Figure 25).

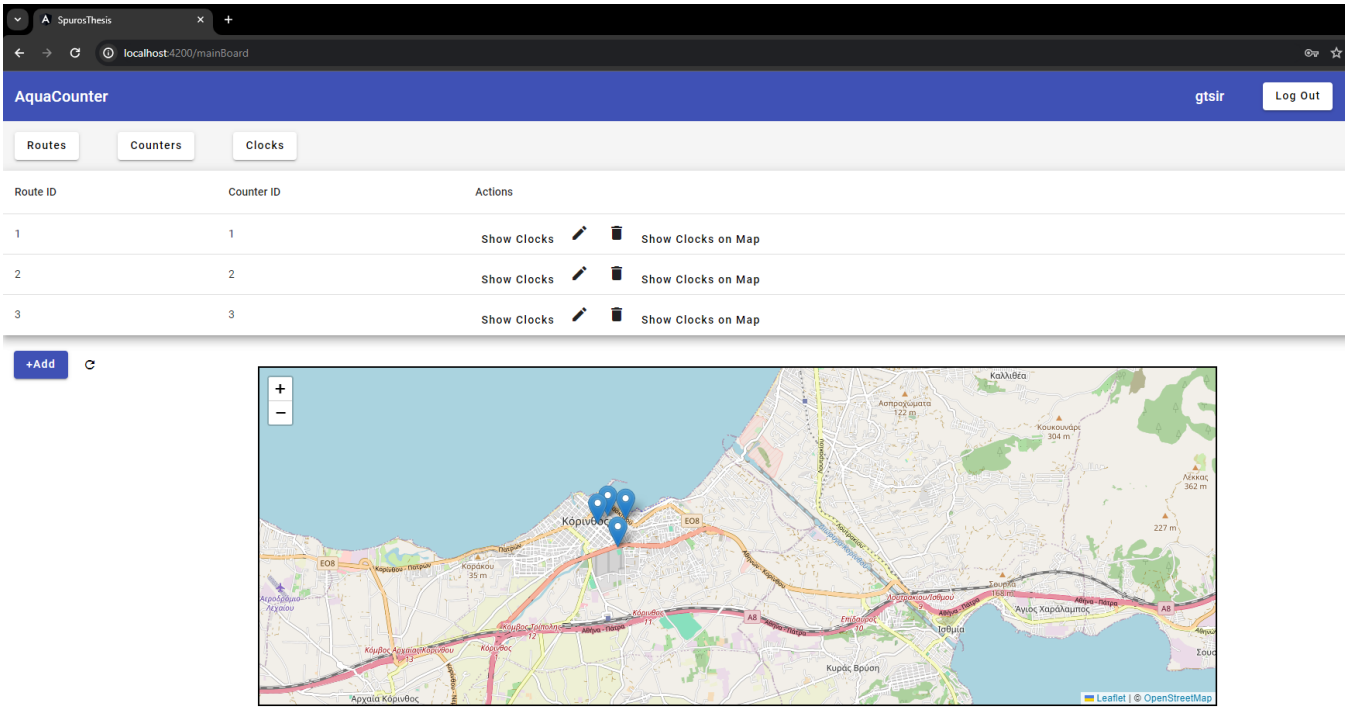

Figure 25-Clocks on map

routes-dialog.component : χρησιμοποιείται για να εμφανίσει τα ρολόγια που σχετίζονται με έναν συγκεκριμένο δρομολόγιο.

Προβολή Ρολογιών: Το component εμφανίζει τα ρολόγια που σχετίζονται με έναν συγκεκριμένο δρομολόγιο, παίρνοντας τα δεδομένα από το API και εμφανίζοντας τα IDs των ρολογιών.

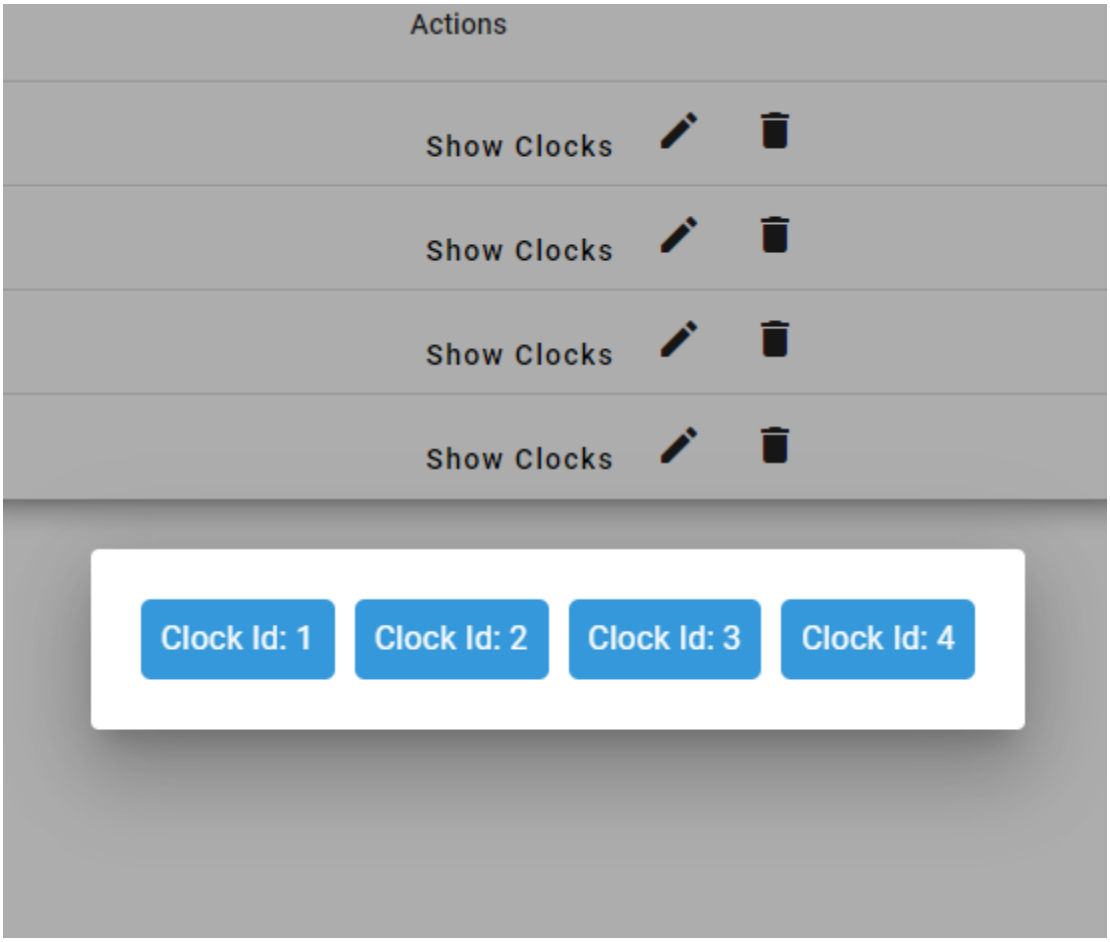

Figure 26 - Display the clock id's of a specific route

clocks.component : χρησιμοποιείται για να εμφανίζει τα ρολόγια που υπάρχουν στο σύστημα.

Εμφάνιση Ρολογιών: Το component εμφανίζει τα ρολόγια που υπάρχουν στο σύστημα, με πληροφορίες όπως το ID του ρολογιού, το ID του δρομολογίου και τις συντεταγμένες του.

Διαγραφή Ρολογιού: Ο χρήστης μπορεί να διαγράψει ένα ρολόι πατώντας το κουμπί διαγραφής.

Προσθήκη Ρολογιού: Ο χρήστης μπορεί να προσθέσει ένα νέο ρολόι εισάγοντας τις τιμές του clockid ,routeid και coordinates πατώντας το κουμπί "Save".

Επεξεργασία Ρολογιού:Ο χρήστης πατώντας το κουμπί για επεξεργασία , μπορεί να διαχειριστεί ήδη αποθηκευμένα ρολόγια, αλλάζοντας τους το routeid και πατώντας save.Στην ουσία έτσι μπορεί να αναθέσει κάποιο ρολόι σε άλλο δρομολόγιο.

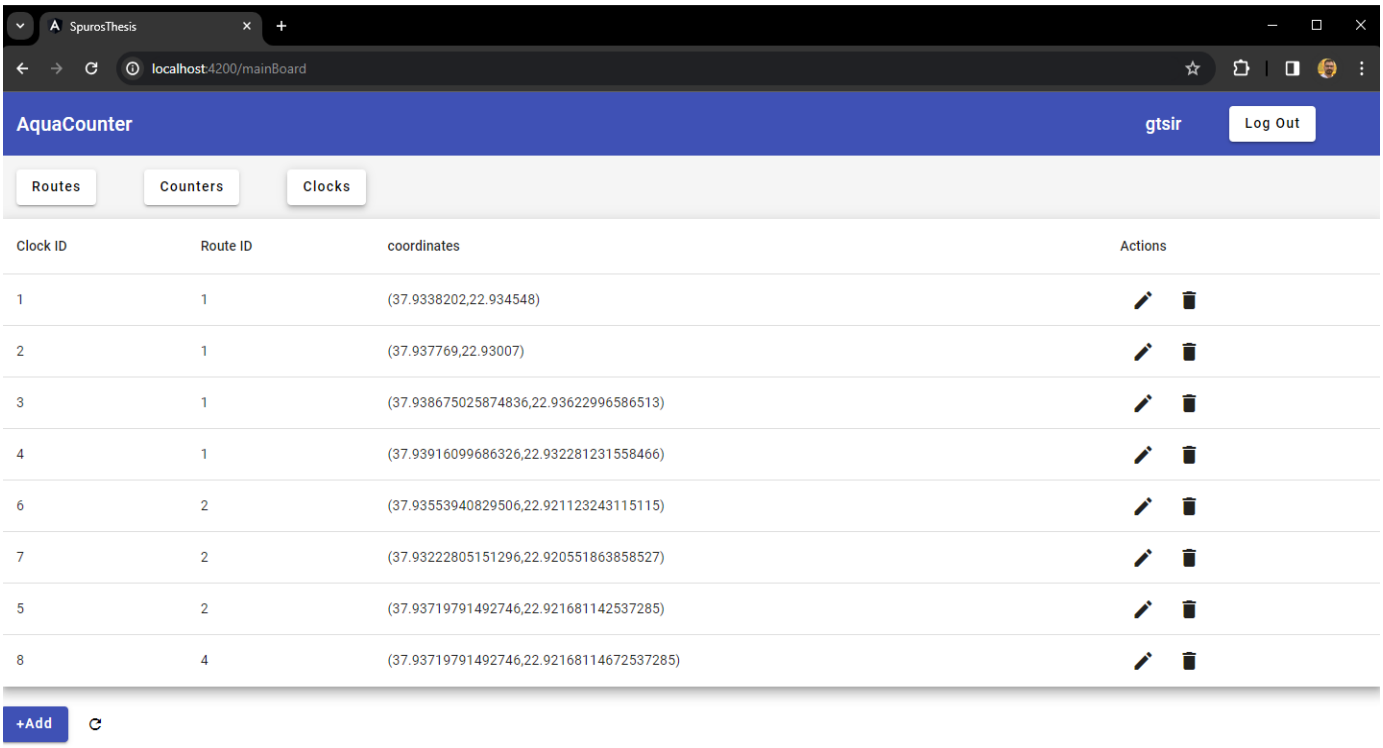

Figure 27 - Display the list of Clocks

counters.component : χρησιμοποιείται για να εμφανίζει τους μετρητές που υπάρχουν στο σύστημα.

Εμφάνιση Μετρητών: Το component εμφανίζει τους μετρητές που υπάρχουν στο σύστημα, με πληροφορίες όπως το ID του μετρητή, το όνομα, το επώνυμο, το όνομα χρήστη και την εξουσία του.

Προσθήκη Μετρητή: Ο χρήστης μπορεί να προσθέσει έναν νέο μετρητή εισάγοντας τις τιμές του counterid,first_Name,last_Name,username,password και authority πατώντας το κουμπί "Save".

Επεξεργασία Μετρητή:Ο χρήστης πατώντας το κουμπί για επεξεργασία , μπορεί να διαχειριστεί ήδη αποθηκευμένους μετρητές, αλλάζοντας τους το first_Name , last_Name , username και authority πατώντας save.

Διαγραφή Μετρητή: Ο χρήστης μπορεί να διαγράψει ένα μετρητή πατώντας το κουμπί διαγραφής.

Προβολή Μετρήσεων: Υπάρχει ένα κουμπί που επιτρέπει στον χρήστη να δει τις μετρήσεις που σχετίζονται με τον επιλεγμένο καταμετρητή. Αυτό υλοποιείται με τη δημιουργία ενός άλλου component που ονομάζεται measurements-dialog.component.

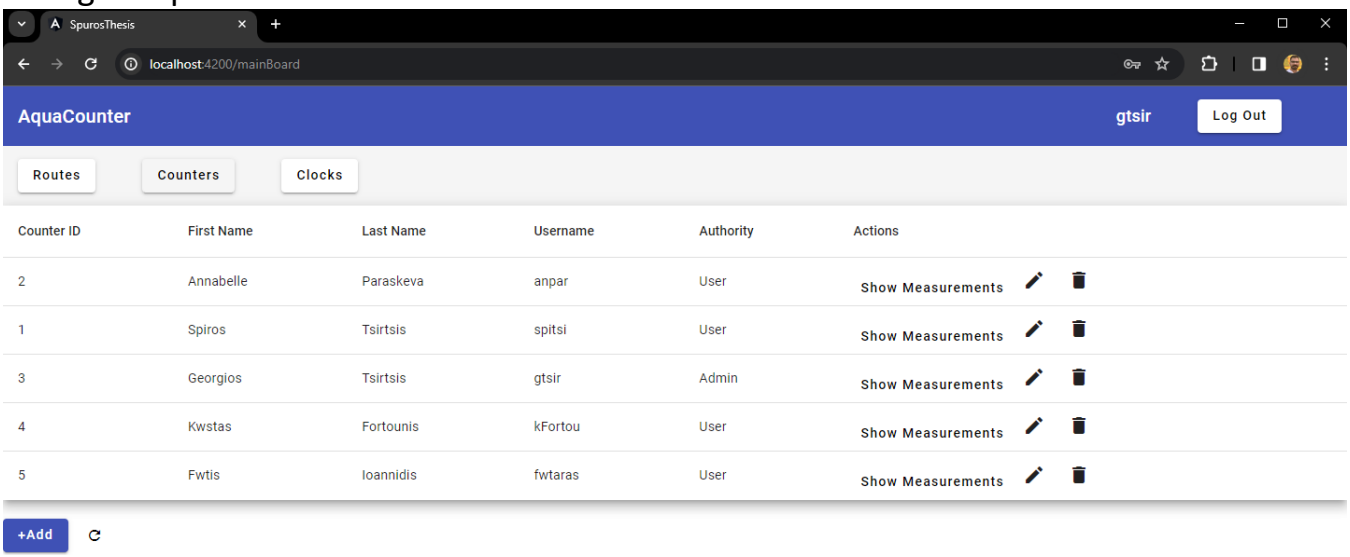

Figure 28 - Display the list of Counters

measurements-dialog.component : χρησιμοποιείται για την εμφάνιση των μετρήσεων που σχετίζονται με έναν συγκεκριμένο μετρητή.

Προβολή Μετρήσεων: Το component εμφανίζει τις μετρήσεις που σχετίζονται με έναν συγκεκριμένο μετρητή, παίρνοντας τα δεδομένα από το API και εμφανίζοντας τις τιμές, το ID του ρολογιού και τη χρονική σήμανση.

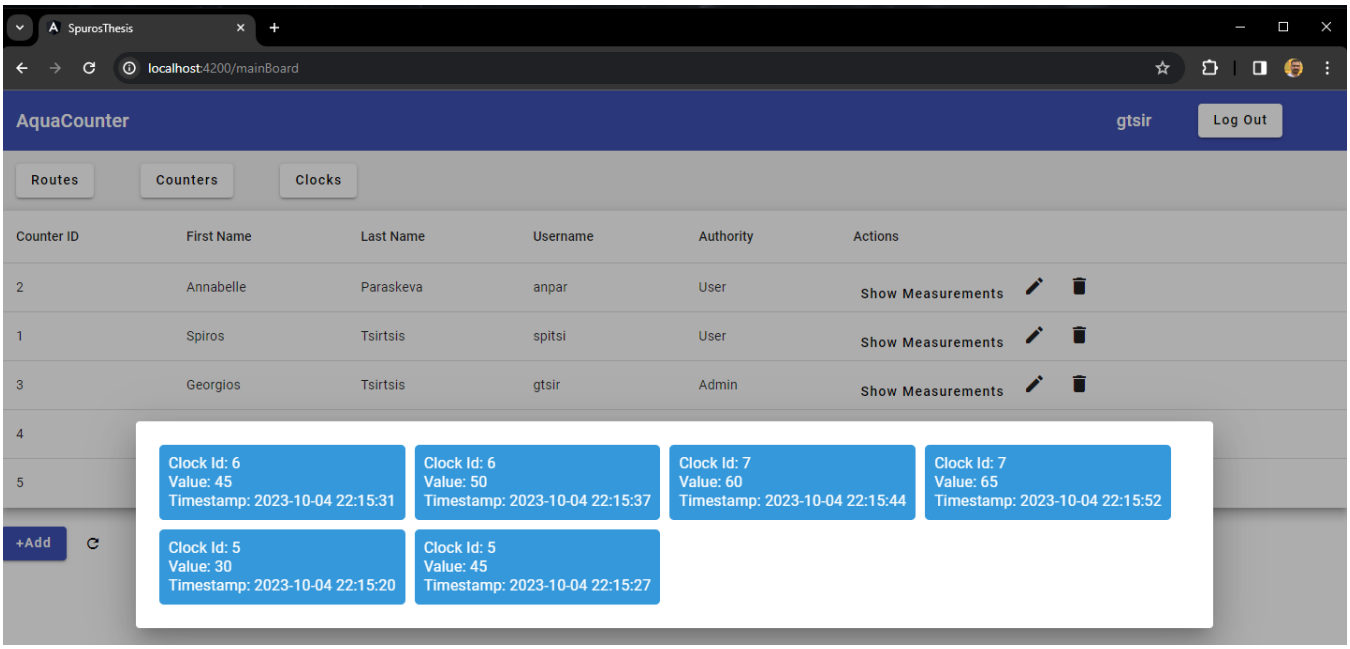

Figure 29 - Display the measurements of a specific Counter

login.component : αντιπροσωπεύει τη σελίδα σύνδεσης της εφαρμογής AquaCounter. Είναι υπεύθυνο για τον έλεγχο των διαπιστευτηρίων του χρήστη και την επικοινωνία με το back-end της εφαρμογής για την εξουσιοδότηση.

Το LoginComponent επιτρέπει στους χρήστες να εισέρχονται στο σύστημα πληκτρολογώντας το όνομα χρήστη και τον κωδικό τους. Όταν ο χρήστης πατήσει το κουμπί "Login", ο κώδικας εκτελεί τη login διαδικασία με το back-end της εφαρμογής.

Ο κώδικας του LoginComponent χρησιμοποιεί την υπηρεσία HttpService για την επικοινωνία με το back-end. Όταν ο χρήστης πατήσει το κουμπί "Login", δημιουργείται ένα αντικείμενο UserCredentials που περιέχει το όνομα χρήστη και τον κωδικό που πληκτρολογεί ο χρήστης. Αυτά τα διαπιστευτήρια αποστέλλονται στο back-end για έλεγχο.

Εάν η εξουσιοδότηση είναι επιτυχής, ο server επιστρέφει ένα token εξουσιοδότησης, το οποίο αποθηκεύεται στην υπηρεσία HttpService. Στη συνέχεια, ο χρήστης κατευθύνεται στη σελίδα mainBoard.

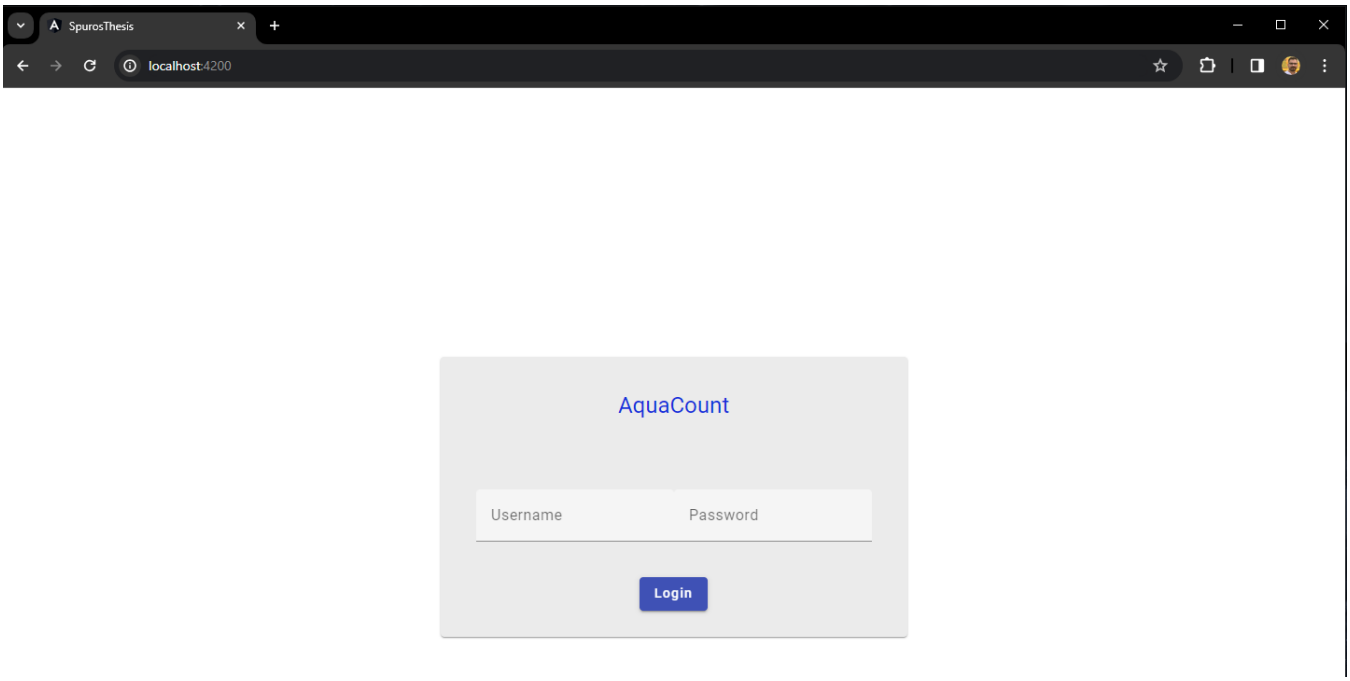

Figure 30 - Login screen of the Admin Dashboard

main-board.component : αντιπροσωπεύει τον κυρίως πίνακα ελέγχου της εφαρμογής AquaCounter.Διαχειρίζεται την κεντρική διακίνηση στην εφαρμογή και παρέχει πρόσβαση σε κυρίως λειτουργίες όπως οι Διαδρομές, οι Καταμετρητές, και τα Ρολόγια.

Κατά την εκκίνηση, το `MainBoardComponent` φορτώνει το token του χρήστη για εξουσιοδότηση και εμφανίζει την κύρια γραμμή εργαλείων της εφαρμογής. Ο χρήστης μπορεί να επιλέξει μεταξύ των κυρίων κατηγοριών όπως Διαδρομές, Καταμετρητές, και Ρολόγια.

Το κεντρικό περιεχόμενο αλλάζει δυναμικά ανάλογα με την επιλογή του χρήστη. Για παράδειγμα, εμφανίζονται τα στοιχεία `app-routes`, `app-counters`, ή `app-clocks` ανάλογα με την επιλογή "Διαδρομές", "Καταμετρητές", ή "Ρολόγια".

Επιπροσθέτως, ο χρήστης έχει τη δυνατότητα να αποσυνδεθεί από το σύστημα με το πάτημα του κουμπιού "Log Out" στην κύρια γραμμή εργαλείων.

Το `MainBoardComponent` επικοινωνεί με το back-end της εφαρμογής μέσω της υπηρεσίας `HttpService`. Η υπηρεσία αυτή χρησιμοποιείται για την επικοινωνία με τον server και τη διαχείριση του χρήστη, συμπεριλαμβανομένου του login καιτου logout.

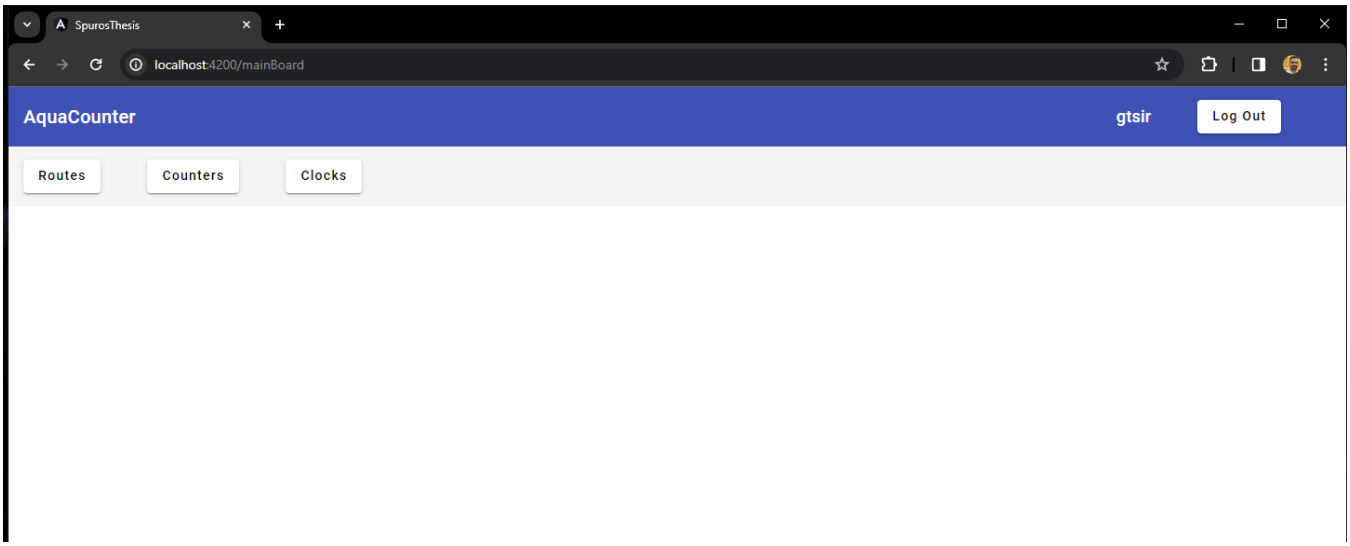

Figure 31 - Main page of Admin Dashboard when you login

http.service : Το HttpService παρέχει μια συλλογή από μεθόδους για την αλληλεπίδραση με έναν διακομιστή που χρησιμοποιείται στην εφαρμογή. Ακολουθεί μια περιγραφή των βασικών λειτουργιών του:

<u>Αλληλεπίδραση με τα Ρολόγια (Clocks):</u>

getAllClocks(): Λαμβάνει όλα τα ρολόγια από τον διακομιστή.

addNewClock (clockObj): Προσθέτει ένα νέο ρολόι στον διακομιστή.

editClock(id, newRoute): Επεξεργάζεται το ρολόι με βάση τη δοθείσα νέα διαδρομή.

deleteClock(id): Διαγράφει το ρολόι με το δοθέν ID.

<u>Αλληλεπίδραση με τους Μετρητές (Counters):</u>

getAllCounters(): Λαμβάνει όλους τους μετρητές από τον διακομιστή.

addNewCounter(counterObj): Προσθέτει ένα νέο μετρητή στον διακομιστή.

editCounter(id, counterObj): Επεξεργάζεται τον μετρητή με βάση τα δοθέντα στοιχεία.

deleteCounter(id): Διαγράφει τον μετρητή με το δοθέν ID.

getCounterMeasurements(id): Λαμβάνει τις μετρήσεις ενός μετρητή με βάση το δοθέν ID.

Αλληλεπίδραση με τις Διαδρομές (Routes):

getAllRoutes(): Λαμβάνει όλες τις διαδρομές από τον διακομιστή.

addNewRoute(routeObj): Προσθέτει μια νέα διαδρομή στον διακομιστή.

editRoute(id, newCounterId): Επεξεργάζεται τη διαδρομή με το δοθέν ID με βάση τον νέο μετρητή.

deleteRoute(id): Διαγράφει τη διαδρομή με το δοθέν ID.

getRouteClocks(id): Λαμβάνει τα ρολόγια που σχετίζονται με μια διαδρομή με βάση το δοθέν ID.

Αλληλεπίδραση με τις Μετρήσεις (Measurements):

getClockMeasurements(id): Λαμβάνει τις μετρήσεις ενός ρολογιού με βάση το δοθέν ID.

Εξουσιοδότηση και Σύνδεση (Authorization & Login):

login(credentials): Αποστέλλει τα διαπιστευτήρια σύνδεσης στον διακομιστή.

saveToken(token): Αποθηκεύει το token στο localStorage.

getToken(): Λαμβάνει το token από το localStorage.

removeToken(): Διαγράφει το token από το localStorage.

saveUser(username): Αποθηκεύει το όνομα χρήστη στο localStorage.

getUser(): Λαμβάνει το όνομα χρήστη από το localStorage.

removeUser(): Διαγράφει το όνομα χρήστη από το localStorage.

Το HttpService είναι υπεύθυνο για τις HTTP αιτήσεις στον διακομιστή και τη διαχείριση των δεδομένων που λαμβάνονται από αυτόν. Επιπλέον, χειρίζεται τον έλεγχο εξουσιοδότησης μέσω tokens και διαχειρίζεται τις αποθηκευμένες πληροφορίες χρήστη.

Τα παραπάνω components φαίνονται στην επόμενη φωτογραφία(Figure 32).

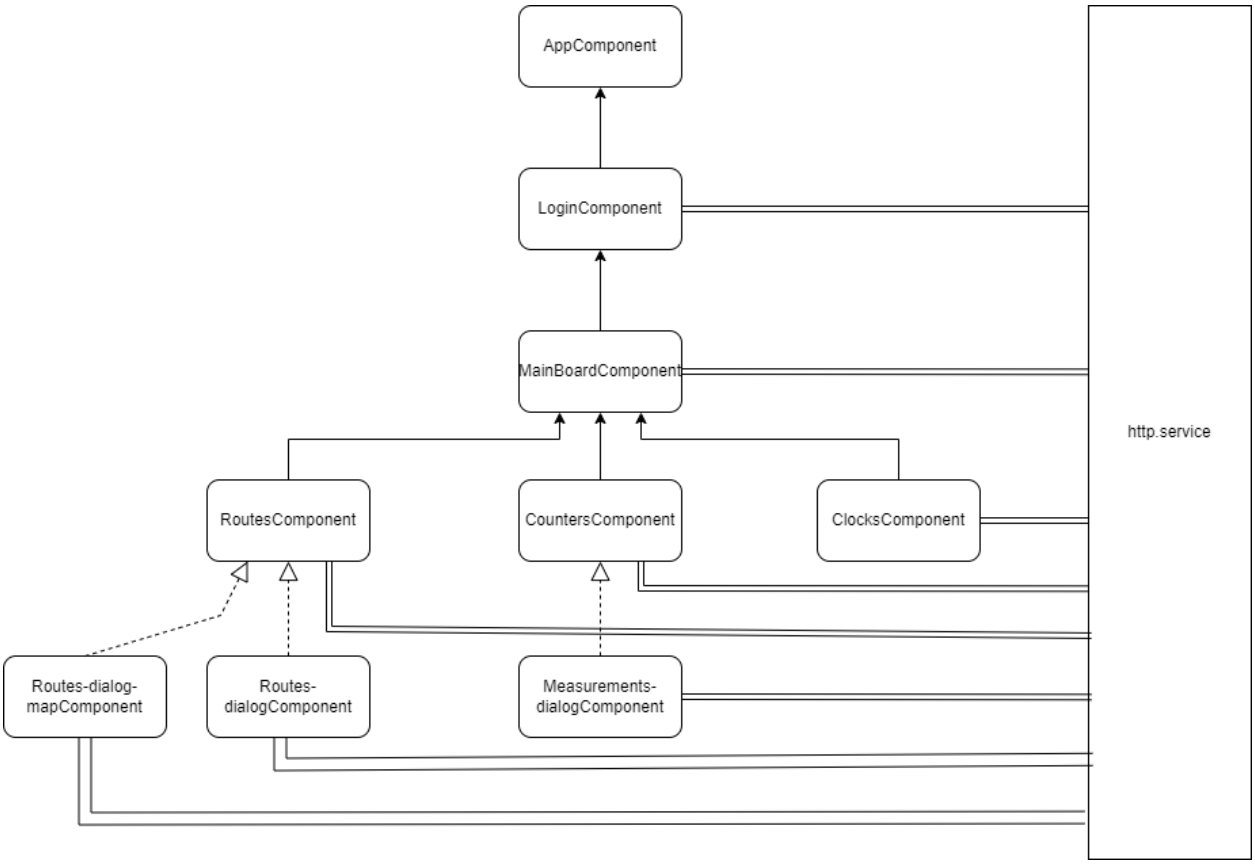

Figure 32 - Component diagram of Admin Dashboard

Βελτιώσεις – Προτάσεις

Η ανάπτυξη ενός συστήματος υδρομετρήσεων αποτελεί ένα διαρκές εγχείρημα, με την ανάγκη για συνεχείς εκσυγχρονίσεις και βελτιώσεις που να ανταποκρίνονται στις αναγκαίες εξελίξεις και στις απαιτήσεις των χρηστών.

Στο πλαίσιο αυτό, παρουσιάζω μια σειρά από προτάσεις και βελτιώσεις που στοχεύουν στην ενίσχυση της λειτουργικότητας, την ευκολία χρήσης και τη βέλτιστη απόδοση του συστήματος υδρομετρήσεων.

- Αρχικά, όσον αφορά την ασφάλεια του συστήματος , μία καλή λύση θα ήταν αντί οι κωδικοί των χρηστών να αποθηκεύονται στην βάση κρυπτογραφημένοι, να αποθηκεύονται πάλι κρυπτογραφημένοι αλλά σε ένα cloud.
- Όσο αναφορά την android εφαρμογή, μία βασική επέκταση που θα μπορούσε να γίνει , θα ήταν ο χρήστης να μπορεί να βλέπει δυναμικά την τοποθεσία του σαν σημείο στον χάρτη και ταυτόχρονα να αλλάζει δυναμικά η διαδρομή με βάση την κίνηση του.
  Επίσης, θα ήταν καλό ο καταμετρητής όταν παρατηρήσει μία βλάβη σε κάποιο ρολόι, να μπορεί να την καταχωρήσει και να γράψει και κάποια σχόλια σχετικά με αυτή.
- Όσο αναφορά το Admin Dashboard, θα ήταν χρήσιμο να προστεθούν πίνακες και ‘πίτες’ που να παρουσιάζουν συνολικά την κατανάλωση που έχει γίνει σε οποιοδήποτε ρολόι.

Αναφορές

[1] «Spring Boot Reference Documentation,» [Ηλεκτρονικό]. Available: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/.

[2] «PostgreSQL Reference Documentation,» [Ηλεκτρονικό]. Available: https://www.postgresql.org/docs/16/index.html.

[3] «Android SDK Reference Documentation,» [Ηλεκτρονικό]. Available: https://developer.android.com/reference/.

[4] «Jwt Reference Documentation,» [Ηλεκτρονικό]. Available: https://auth0.com/docs/secure/tokens/json-web-tokens.

[5] «What is Javascript? -Learn web development MDN Web Docs,» [Ηλεκτρονικό]. Available: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript.

[6] H.Agrawal, «What is typescript and why should you use it?,» [Ηλεκτρονικό]. Available: https://www.contentful.com/blog/what-is-typescript-and-why-should-you-use-it/.

[7] «What is HTML? Hypertext Markup Language Basics Explained,» [Ηλεκτρονικό]. Available: https://www.hostinger.com/tutorials/what-is-html.

[8] «CSS Introduction.W3Schools,» [Ηλεκτρονικό]. Available: https://www.w3schools.com/css/css_intro.asp.

[9] «What is JSON? W3Schools,» [Ηλεκτρονικό]. Available: https://www.w3schools.com/whatis/whatis_json.asp.

[10] «What is Angular? Angular,» [Ηλεκτρονικό]. Available: https://angular.io/guide/what-is-angular.

[11] «Visual Studio Code Frequently Asked Questions.,» [Ηλεκτρονικό]. Available: https://code.visualstudio.com/docs/supporting/FAQ.

[12] «REST API Reference Documentation,» [Ηλεκτρονικό]. Available: https://restfulapi.net/.

[13] X. R. Documentation. [Ηλεκτρονικό]. Available: https://developer.android.com/reference/android/util/Xml.

[14] «SQLite Reference Documentation,» [Ηλεκτρονικό]. Available: https://www.sqlite.org/docs.html.