Εθνικο Μετσοβιο Πολυτεχνειο
Τμημα Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Συστηματων Μεταδοσης και Τεχνολογιας Υλικων

Σχεδιασμός και ανάπτυξη συστήματος ασφαλούς συλλογής και διαχείρισης δεικτών ευημερίας του ατόμου

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Παύλου Κ. Ζάκκα

Επιβλέπων: Ιάκωβος Σ. Βενιέρης
Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2017

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΜΕΤΑΔΟΣΗΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΛΙΚΩΝ

Σχεδιασμός και ανάπτυξη συστήματος ασφαλούς συλλογής και διαχείρισης δεικτών ευημερίας του ατόμου

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Παύλου Κ. Ζάκκα

Επιβλέπων: Ιάκωβος Σ. Βενιέρης
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 23η Οκτωβρίου 2017.

.....	.....	.....
Ιάκωβος Σ. Βενιέρης	Δήμητρα-Θεοδώρα Κακλαμάνη	Γεώργιος Ματσόπουλος
Καθηγητής Ε.Μ.Π.	Καθηγήτρια Ε.Μ.Π.	Αναπληρωτής Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2017

.....

Παύλος Κ. Ζάκκας
(Διπλωματούχος Ηλεκτρολόγος Μηχανικός & Μηχανικός Υπολογιστών Ε.Μ.Π.)

Περίληψη

Ένα από τα σημαντικότερα θέματα που απασχολούν σε καθημερινή βάση τους ανθρώπους είναι η παρακολούθηση της φυσικής τους κατάστασης και γενικότερα της υγείας τους, με απώτερο στόχο τη βελτίωση της ποιότητας ζωής τους. Φυσικά, εύκολα καταλαβαίνει κάποιος ότι το παραπάνω είναι δύσκολο να επιτευχθεί, αν αναλογιστούμε τους γρήγορους ρυθμούς ζωής ενός μέσου ανθρώπου της εποχής μας. Προκειμένου να ξεπεραστεί η δυσκολία αυτή, η τεχνολογία έχει παίξει καθοριστικό ρόλο, καθώς, μέσω της ευρείας χρήσης προσωπικών ενδυτών συσκευών και έξυπνων εφαρμογών, έχει καταστεί δυνατή η παρακολούθηση των καθημερινών δραστηριοτήτων του ατόμου και η μέτρηση σημαντικών δεικτών ευημερίας του, προκειμένου να σχηματιστεί μια συνολική εικόνα για την υγεία του και τις καθημερινές του συνήθειες. Ωστόσο, οι μετρήσεις που πρέπει να ληφθούν υπόψη για την εξαγωγή συμπερασμάτων με απώτερο σκοπό τη βελτίωση της ποιότητας ζωής του προέρχονται από πολλές και ποικίλες πηγές, γεγονός που οδηγεί σε έναν τεράστιο όγκο δεδομένων, ποικίλων μορφών και δομών. Έτσι, καθίσταται εξαιρετικά επίπονη η επεξεργασία των δεδομένων αυτών και η εξαγωγή ορθών συμπερασμάτων.

Σκοπός αυτής της διπλωματικής εργασίας είναι, αφενός, η προτυποποίηση της μορφής των δεδομένων που συλλέγονται από ενδυτές συσκευές του χρήστη και εφαρμογές που παρακολουθούν τη φυσική του κατάσταση και, αφετέρου, η ανάπτυξη ενός συστήματος που αποτελείται από έναν εξυπηρετητή και μια εφαρμογή κινητού τηλεφώνου. Τελικός στόχος της παρούσας εργασίας είναι η ασφαλής συλλογή προσομοιωτικών δεδομένων υγείας απο διάφορες πηγές, η μετέπειτα επεξεργασία τους, η εξαγωγή ασφαλών συμπερασμάτων και η δημιουργία προτάσεων προς αυτόν με απώτερο σκοπό τη βελτίωση της ευημερίας του ατόμου μέσω σωστής καθοδήγησης. Συγκεκριμένα, προτείνεται, αρχικά, μία οντολογία δεδομένων υγείας και δεικτών ευημερίας, η οποία αποτελεί ουσιαστικά ένα κοινό λεξιλόγιο, η χρήση του οποίου διευκολύνει σημαντικά τη διαχείριση πληθώρας προσωπικών δεδομένων συλλεγμένων από ετερογενείς πηγές. Στη συνέχεια, παρουσιάζεται ένα σύστημα το οποίο μετατρέπει με βάση την προαναφερθείσα οντολογία τα συλλεχθέντα δεδομένα σε σημασιολογικά και, κατόπιν, τα επεξεργάζεται με σκοπό την εξαγωγή ορθών παρατηρήσεων. Απώτερη επιδίωξη του συστήματος που αναπτύχθηκε στα πλαίσια της διπλωματικής αυτής εργασίας είναι η συνεχής συμβολή στη βελτίωση του ευ ζην του ατόμου μέσω παροχής καίριων συμβουλών που εχουν προέλθει από την επεξεργασία και τον περαιτέρω εμπλουτισμό δεδομένων που συλλέχθηκαν από προσωπικές του συσκευές και αισθητήρες στο περιβάλλον αυτού.

Λέξεις κλειδιά

Σύστημα διαχείρισης δεδομένων, Βελτίωση ευημερίας, Οντολογία, Μη-σχεσιακή βάση δεδομένων, Βάση δεδομένων γραφημάτων, JSON-LD, Εφαρμογή, Android.

Abstract

Nowadays it is well-known that one of the biggest issues of human beings has to do with their need to monitor their health in order to improve their everyday life. Generally, achieving this goal may be difficult, if we consider, also, the fast pace of life. In order, though, to improve our wellbeing, we have resorted to applications of technology, which can monitor our fitness and way of life. As a result, there is a wide range of data sources, including wearable devices, mobile applications and sensors, that have been developed to record body metrics and fitness habits. This monitoring process is essential, if we want to provide the right guidance to a person towards the improvement of their quality of life. On the other hand, the big number of data sources has led to a huge volume of data. Moreover, there is a wide variety of health-related data models and representations being used by health professionals. As a consequence, it is really hard to process the total amount of data collected by heterogeneous data sources and develop a useful recommendation system for the users.

The main goals of this thesis are, firstly, the presentation of a semantically rich information model representing health data aggregated by wearable devices and applications, and, secondly, the development of a system, containing a server and a mobile application, for improving the quality of personal health. Once the simulated health data are securely collected and stored, reasoning techniques are applied and their further processing leads to meaningful information that is leveraged for making proper conclusions about a person's daily health and fitness routine, as well as suggestions that may be useful to the user. To be more specific, an ontology of health data and wellness metrics is introduced as a common vocabulary among data sources so that we can manipulate the heterogeneity of health-related data. Furthermore, we present a system that was developed in order to aggregate the aforementioned data and transform them to semantic data on the purpose of easy handling and processing. Finally, the improvement of wellbeing will be fulfilled through the provision of advice and motivation to any user that possesses wearable devices, mobile applications and smart home sensors via a unified Graphical User Interface provided by the "MyWellnessUp" Android application that was developed in the context of this thesis.

Keywords

Data management system, Wellbeing improvement, Ontology, NoSQL Database, Graph Database, JSON-LD, Mobile Application, Android.

Ευχαριστίες

Αρχικά, θα ήθελα να ευχαριστήσω τον επιβλέποντα της διπλωματικής μου εργασίας, καθηγητή Ιάκωβο Βενιέρη, ο οποίος μου έδωσε τη δυνατότητα να ασχοληθώ με το ιδιαίτερα ενδιαφέρον αντικείμενο της παρούσας εργασίας, μέσα από την οποία ήρθα για πρώτη φορά σε επαφή με θέματα που βρίσκονται στο επίκεντρο των τεχνολογικών εξελίξεων.

Στη συνέχεια, θα ήθελα να ευχαριστήσω τα μέλη της τριμελούς επιτροπής, την καθηγήτρια Δήμητρα-Θεοδώρα Κακλαμάνη και τον αναπληρωτή καθηγητή Γεώργιο Ματσόπουλο για την υποστήριξή τους καθόλη τη διάρκεια εκπόνησης της παρούσας διπλωματικής εργασίας.

Ιδιαίτερες ευχαριστίες θα ήθελα να απευθύνω, επίσης, στην υποψήφια διδάκτορα του Ε.Μ.Π. Μαρία-Ελευθερία Παπαδοπούλου για την άριστη καθοδήγησή της και την πολύτιμη βοήθειά της σε οποιοδήποτε πρόβλημα αντιμετώπισα κατά τη διάρκεια εκπόνησης της εργασίας.

Τέλος, οφείλω ένα μεγάλο ευχαριστώ στην οικογένειά μου και στα αγαπημένα μου πρόσωπα για όλη τη βοήθεια, υποστήριξη, εμπιστοσύνη και αντοχή που έδειξαν κατά τη διάρκεια όλων των σπουδών μου.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

Είναι γεγονός ότι μια από τις βασικότερες έγνοιες του ανθρώπου κάθε εποχής είναι να έχει μία ποιοτική ζωή. Για αυτό το λόγο, αναζητά τρόπους με τους οποίους μπορεί να βελτιώσει τις καθημερινές του συνήθειες, παρόλο που οι σύγχρονοι ρυθμοί ζωής στέκονται εμπόδιο πολλές φορές. Είναι αναμφίβολα επιθυμητό, λοιπόν, να έχει την δυνατότητα να συλλέγει σε πρώτο στάδιο όσο το δυνατόν περισσότερες και ακριβείς πληροφορίες που σχετίζονται με την φυσικη του κατάσταση και κατ΄ επέκταση με το τρόπο ζωής του. Στη συνέχεια, είναι απαραίτητο αυτά τα δεδομένα να μελετηθούν και να επεξεργαστούν συνδυαστικά μεταξύ τους, προκειμένου να παραχθούν συμπεράσματα που αντικατοπτρίζουν την κατάσταση της υγείας του. Σε τελικό στάδιο, η ευημέρια του σύγχρονου ανθρώπου, θα προέλθει από την καθοδήγηση που υποδεικνύεται απο τα παραπάνω συμπεράσματα καθώς μέσω αυτών μπορεί να αλλάξει τυχόν καθημερινές συνήθειες που είναι επιβλαβείς και να βελτιώσει την ποιότητα ζωής του.

Απόρροια της προαναφερθείσας ανάγκης είναι το γεγονός ότι ο άνθρωπος έχει καταφύγει σε τεχνολογίες της σύγχρονης εποχής με σκοπό να παρακολουθεί τον τρόπο ζωής του. Ειδικότερα, στις μέρες μας, είναι αναρίθμητες οι πηγές που έχουν αναπτυχθεί για την καταγραφή μετρήσεων που αφορούν την υγεία και τους σχετικούς δείκτες ευημερίας του ατόμου. Αυτές οι πηγές μπορεί να είναι συσκευές που φέρει ο χρήστης πάνω του, εφαρμογές που έχουν σχεδιαστεί για κινητά τηλέφωνα, καθώς επίσης και αισθητήρες που υπάρχουν στο περιβάλλον του χρήστη. Η ποικιλία και η ποσότητα αυτών των εφαρμογών είναι εξαιρετικά μεγάλη, καθώς χρησιμοποιούνται ευρέως από τον σύγχρονο άνθρωπο και έχουν εισέλθει στην καθημερινότητα του. Ο ρόλος τους είναι συμβουλευτικός και αποσκοπεί στην βελτίωση της ποιότητας ζωής του χρήστη μέσω της παρακολούθησης μετρήσεων που ο ίδιος επιθυμεί και που τον απασχολούν περισσότερο.

Φυσικά είναι εύκολο να σκεφτεί κάποιος ότι όλη αυτή η πληθώρα εφαρμογών της τεχνολογίας οδηγεί ραγδαία στην παραγωγή ενός τεράστιου όγκου δεδομένων που σχετίζονται με την υγεία του ατόμου. Σαν αποτέλεσμα είναι επίπονη η διαδικασία της επεξεργασίας και η εξαγωγή ορθών συμπερασμάτων. Γενικότερα, είναι γνωστό ότι οι συσκευές που χρησιμοποιούνται καταγράφουν πολλές φορές ίδιου τύπου δεδομένα, αλλά εξετάζοντάς τα με διαφορετικούς τρόπους με αποτέλεσμα αν μπορούσε κάποιος να συλλέξει αυτές τις μετρήσεις, να έπαιρνε πολλές διαφορετικές τιμές για τον ίδιο δείκτη. Ακόμα, κάθε τέτοια εφαρμογή αναπαριστά με τον δικό της τρόπο τα δεδομένα που συλλέγει προκειμένου στην συνέχεια να

τα αξιοποιήσει όπως η ίδια κρίνει σωστό. Επομένως, εύκολα διακρίνει κανείς τους ποικίλους τρόπους για την αναπαράσταση ίδιου τύπου πληροφορίας.

Ολοκληρώνοντας, είναι προφανές, πως με αυτήν την λογική, δυσχεραίνεται ο συσχετισμός των δεδομένων που συλλέγονται από διαφορετικές πηγές, ενώ επίσης καθίσταται δύσκολη μια πιθανή επικοινωνία μεταξύ παρόμοιων εφαρμογών. Μια τέτοια επικοινωνία θα ήταν επιθυμητή, διότι θα μπορούσε να ενισχύσει την προσπάθεια για βελτίωση της ποιότητας ζωής του σύγχρονου ανθρώπου, μέσω της σύνθεσης των παρατηρήσεων που καταγράφηκαν συνολικά.

1.1 Αντικείμενο διπλωματικής εργασίας

Σκοπός της διπλωματικής εργασίας είναι η **μελέτη ενός τρόπου διασύνδεσης όλων αυτών των δεδομένων υγείας και δεικτών ευημερίας** που προέρχονται από ενδυτές συσκευές και άλλες εφαρμογές. Απώτερος στόχος αυτου του συσχετισμού των δεδομένων είναι η συμβολή στο ευ ζην του ατόμου, καθώς η προτυποποίηση τους και η δημιουργία μιας συνολικής εικόνας της υγείας ενός ατόμου αποτελεί αναπόσπαστο κομμάτι της σωστής καθοδήγησης των χρηστών της σύγχρονης τεχνολογίας.

Ειδικότερα, με την ανάπτυξη ενός κοινού κώδικα επικοινωνίας μεταξύ των πηγών που συλλέγουν τέτοιου είδους πληροφορία, γίνεται ευκολότερη η εφαρμογή μιας διαδικασίας που θα δώσει νόημα σε αυτά τα δεδομένα. Σαν αποτέλεσμα, θα οδηγήσει στην παραγωγή νέας πληροφορίας ικανής να καθοδηγήσει κάθε άτομο ξεχωριστά στην καθημερινότητά του. Η **δημιουργία, λοιπόν, μιας καθολικής γλώσσας οντολογικού ιστού**, αποτελεί ένα κύριο μέλημα αυτής της εργασίας. Για το σκοπό αυτό, θα γίνει εμφανής ο τρόπος με τον οποίο τα δεδομένα υγείας και οι δείκτες ευημερίας μπορούν να εκφραστούν με βάση αυτή την γλώσσα, που θα αναπτυχθεί με χρήση τεχνολογιών σημασιολογικού ιστού. Έτσι, όσο διαφορετικά κι αν είναι τα δεδομένα που συλλέγονται, θα μπορουν να συσχετιστούν μεταξύ τους ακολουθώντας ενα καθολικό **κοινό λεξιλόγιο**.

Επιπροσθέτως, η **ανάπτυξη συστήματος ενός εξυπηρετητή και μιας εφαρμογής σε κινητά τηλέφωνα** αποτελεί το άλλο μεγάλο κομμάτι αυτής της εργασίας. Με αυτό το τρόπο θα γίνει εφικτή, αφενός η ασφαλής συνάθροιση και προτυποποίηση μετρήσεων και δεικτών ευημερίας απο ήδη υπάρχουσες συσκευές και εφαρμογές και αφετέρου η λογική επεξεργασία τους. Μέσω αυτής, μάλιστα, θα επιτευχθεί η **παροχή συμβουλών και παρατηρήσεων στους χρήστες** της εφαρμογής. Βεβαίως, είναι απαραίτητο τα συμπεράσματα που θα προέλθουν απο την συσσωρευμένη πληροφορία να είναι ασφαλή, καθώς η υγεία του ατόμου αποτελεί το σημαντικότερο αγαθό του. Τέλος, είναι εξίσου ίδιας σημασίας ο ρόλος της εφαρμογής να είναι συμβουλευτικός και όχι παρεμβατικός στην ζωή του ατόμου.

Αξίζει να σημειωθεί σε αυτό το σημείο ότι καθόλη τη διάρκεια ανάπτυξης του συστήματος, πρέπει να υπάρχει **επίγνωση της ανάγκης για ασφάλεια και προστασία της ιδιωτικότητας του χρήστη**, διότι πρόκειται να γίνει διαχείριση ευαίσθητων προσωπικών δεδομένων του. Είναι εμφανές ότι μια πιθανή διαρροή πληροφορίας σχετικής με την υγεία του ατόμου, θα καθιστούσε αυτόματα το σύστημά μας αναξιόπιστο.

1.2 Διάρθρωση της εργασίας

Η δομή της παρούσας διπλωματικής εργασίας είναι η εξής:
Στο *Κεφάλαιο 2* παρουσιάζεται ο τρόπος με τον οποίο γίνεται η συλλογή και η διαχείριση δεδομένων υγείας και δεικτών ευημερίας από ενδυτές συσκευές και εφαρμογές. Γίνεται, ακόμη, περιγραφή του προβλήματος που προκύπτει ως προς την επεξεργασία του μεγάλου όγκου δεδομένων που συγκεντρώνονται, καθώς και των αναγκών που δημιουργούνται.
Το *Κεφάλαιο 3* αφιερώνεται στην ανάλυση του συστήματος που θα αναπτυχθεί. Γίνεται αναφορά στις απαιτήσεις τις εφαρμογής και στις λειτουργίες του εξυπηρετητή και της εφαρμογής. Τέλος, περιγράφονται οι τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση.
Στο *Κεφάλαιο 4* παρουσιάζεται η σχεδίαση του συστήματος τόσο σε επίπεδο δεδομένων όσο και σε επίπεδο εφαρμογής. Σε επίπεδο δεδομένων, αναπτύσσεται η γλώσσα οντολογικού ιστού που θα χρησιμοποιηθεί, ενώ σε επίπεδο εξυπηρετητή και εφαρμογής, γίνεται περιγραφή του τρόπου με τον οποίο συλλέγονται, αποθηκεύονται και επεξεργάζονται τα δεδομένα και του τρόπου που παρουσιάζονται τα εξαγόμενα συμπεράσματα.
Το *Κεφάλαιο 5* αφιερώνεται στην ανάπτυξη των επιμέρους στοιχείων του συστήματος, δηλαδή στη δημιουργία προσομοιωτικών δεδομένων, στην υλοποίηση του εξυπηρετητή της εφαρμογής που συλλέγει τα δεδομένα και εν τέλει στην υλοποίηση της ίδιας της εφαρμογής σε κινητά τηλέφωνα.
Τέλος, στο *Κεφάλαιο 6*, εξετάζονται τα συμπεράσματα της παρούσας εργασίας, ενώ, ακόμη, προτείνονται μελλοντικές επεκτάσεις της.

Κεφάλαιο 2

Συλλογή και διαχείριση δεικτών ευημερίας

Σε αυτό το κεφάλαιο γίνεται μια πιο εκτενής αναφορά στις ένδυτες συσκευές (wearables), στις εφαρμογές κινητών τηλεφώνων και στους σχετικούς αισθητήρες που υπάρχουν ήδη στην αγορά. Αναλύεται επίσης ο τρόπος με τον οποίο συλλέγουν και διαχειρίζονται τα δεδομένα τους. Επισημαίνονται οι δυσκολίες διαχείρισης του όγκου της πληροφορίας, ενώ, τέλος, παρουσιάζεται η σύγχρονη ιδέα του 'Διαδικτύου των Πραγμάτων' (Internet of Things), μέσω της οποίας θα πραγματοποιηθεί η ικανοποίηση της ανάγκης για διασύνδεση και συσχετισμό των δεδομένων που καταγράφονται.

2.1 Υφιστάμενη κατάσταση ενδυτών συσκευών και εφαρμογών

Σε πρώτο στάδιο, αξίζει να αναφερθεί κανείς στην κατάσταση που επικρατεί αυτή τη στιγμή στο τομέα των τεχνολογιών (State of the Art), που ασχολούνται με την συλλογή δεδομένων υγείας και μετρικών ευημερίας. Είναι ξεκάθαρο ότι οι τεχνολογίες τέτοιου τύπου που έχουν αναπτυχθεί είναι αναρίθμητες, καθότι είναι επιτακτική η ανάγκη του ανθρώπου για μια ολοκληρωμένη παρακολούθηση της υγείας του και των καθημερινών συνηθειών του που σχετίζονται με αυτήν. Επομένως, η χρήση τους είναι εκτεταμένη στη ζωή του μέσου ανθρώπου, ενώ επίσης είναι αξιοσημείωτο το γεγονός ότι κάθε χρήστης μπορεί να επιλέγει διαφορετικές εφαρμογές και συσκευές ανάλογα με το τί στόχο επιθυμεί να πετύχει και ποιούς δείκτες φυσικής κατάστασης και ευημερίας θέλει να μετρήσει.

Αρχικά, είναι ιδιαίτερα διαδεδομένες οι **ένδυτες συσκευές** που καταγράφουν τέτοια δεδομένα υγείας. Είναι σημαντικό, λοιπόν να αναφερθούν κάποιες από αυτες παρακάτω:

- **Fitbit**[1]: Πρόκειται για μία εταιρία παραγωγής μιας σειράς από ενδυτές συσκευές και όχι μόνο. Τα πιο πρόσφατα προϊόντα της, που σχετίζονται με την καταγραφή της φυσικής δραστηριότητας του χρήστη (fitness trackers, smart watches), και που μπορεί να φοράει κάποιος στο χέρι του σαν ρολόι κατά τη διάρκεια της μέρας, ενδεικτικά είναι

[1] https://www.fitbit.com/eu/home

τα ακόλουθα: Fitbit Alta, AltaHR, Fitbit Charge 2, Fitbit Flex 2, Fitbit Zip, Fitbit Blaze, Fitbit Ionic. Οι δυνατότητες που παρέχει καθένα από αυτά στον χρήστη της είναι η μέτρηση και παρακολούθηση του καρδιακού του παλμού, η αυτόματη αναγνώριση της εκάστοτε σωματικής κατάστασης του ή ακόμη και της αθλητικής δραστηριότητας που κάνει. Στα πλαίσια της καθημερινής δραστηριότητας του ατόμου γίνεται ακόμη καταγραφή των βημάτων που κάνει, της απόστασης που διανύει είτε με τα πόδια είτε με άλλο μέσο(ποδήλατο) καθώς και των θερμίδων που καίει κατά τη διάρκεια της μέρας. Επιπλέον, παρέχει την δυνατότητα αυτόματης καταγραφής του ύπνου του χρήστη, καθώς και των σταδίων του ύπνου που βρίσκεται την κάθε χρονική στιγμή. Τέλος, επεξεργάζοντας τα παραπάνω δεδομένα που συλλέγονται, η συσκευή υπενθυμίζει στον χρήστη να ασκείται σωματικά όταν πρέπει, ενώ επίσης τον συμβουλεύει για το επίπεδο της φυσικής του κατάστασης. Είναι άξιο αναφοράς, επίσης, ότι το τελευταίο προϊόν στη προηγούμενη σειρά (Fitbit Ionic-smartwatch) είναι ανθεκτικό στο νερό με αποτέλεσμα να παρέχει την δυνατότητα καταγραφής δεδομένων κατά την κολύμβηση, κάτι που είναι μείζονος σημασίας για μια συγκεκριμένη ομάδα χρηστών(π.χ. αθλητές κολύμβησης).

- **Garmin**[2]: Είναι ακόμη μια μεγάλη εταιρία που παρέχει εκτός των άλλων και ενδυτές συσκευές παρόμοιες με τις παραπάνω. Συγκεκριμένα, η σειρά vivo (vivo-series activity trackers) έχει προϊόντα που καταγράφουν ακριβώς τα ίδια δεδομένα που περιγράφηκαν προηγουμένως. Αξίζει να σημειωθεί όμως ότι παρέχει, ακόμα, και την δυνατότητα παρακολούθησης των επιπέδων άγχους που βρίσκεται κάθε στιγμή της μέρας ο χρήστης (all-day stress and recovery monitoring). Ο τρόπος που το πετυχαίνει αυτό είναι παρατηρώντας την εκάστοτε κατάσταση του χρήστη και τον καρδιακό του παλμό, οι διακυμάνσεις του οποίου συνδέονται στενά με τα επίπεδα άγχους. Σαν απόρροια αυτών των δεδομένων υγείας είναι η επιστημονική καθοδήγηση της ζωής του χρήστη από την εφαρμογή μιας τέτοιας συσκευής με σκοπό μια πιο ποιοτική ζωή. Τέλος, σε αυτό συνεισφέρει και μια σειρά από ενσωματωμένες στην συσκευή εφαρμογές (built-in apps) μέσω τον οποίων συμβουλεύεται ο χρήστης για προγράμματα αθλητικής δραστηριότητας που μπορεί να ακολουθήσει.

- **LumoBodyTech**[3]: Ομοίως και με συσκευές αυτής της εταιρίας (Lumo Lift, Lumo Run) γίνεται παρακολούθηση της φυσικής δραστηριότητας του χρήστη. Καταγράφονται παρόμοια δεδομένα με τις παραπάνω συσκευές, ενώ υποστηρίζει και φωνητική καθοδήγηση για την αθλητική άσκηση του χρήστη. Ο λόγος που αναφέρεται σε αυτό το σημείο είναι η διαφορετική μορφή που έχει σαν ενδυτή συσκευή, καθώς επισυνάπτεται (clip-on) εύκολα σε ρούχα, κάτι που μπορεί να είναι πιο επιθυμητό από κάποιους χρήστες.

- **Sensoria Fitness Sock**[4]: Πρόκειται για μια διαφορετική ενδυτή τεχνολογία που έχει την μορφή κάλτσας. Πράγματι, ο χρήστης καλείται να φορέσει μια ειδική κάλτσα, μέσω της οποίας είναι δυνατή η καταγραφή δεδομένων σχετικών με το περπάτημα ή το τρέξιμο(θερμίδες, βήματα, απόσταση, ταχύτητα). Αξιοσημείωτη είναι και η δυνατότητα ελέγχου της ποιότητας της κίνησης του ατόμου. Πιο συγκεκριμένα, μελετάει τον τρόπο

[2]https://www.garmin.com/en-US
[3]https://www.lumobodytech.com
[4]http://www.sensoriafitness.com

με τον οποίο πατάει το πόδι στο έδαφος και την κατανομή του βάρους του σώματος στα διάφορα σημεία του ποδιού. Με αυτές τις μετρήσεις, δίνει πληροφορίες στον χρήστη για τον τρόπο που περπατάει ή τρέχει και τον συμβουλεύει για να αποκτήσει μια σωστή τεχνική.

- **Beddit Smart 3**[5]: Μέσω αυτής της συσκευής γίνεται συλλογή εξειδικευμένων δεδομένων για τον ύπνο του χρήστη. Τοποθετείται κάτω από τα σεντόνια και έχει την δυνατότητα να μετράει αυτόματα την χρονική περίοδο του ύπνου και των φάσεων του. Επικεντρώνεται στη μελέτη της ποιότητας του ύπνου του ατόμου, καταγράφοντας, μεταξύ άλλων, την διάρκεια του, του παλμούς του χρήστη και τον ρυθμό που αναπνέει. Υπάρχουν, προφανώς αρκέτες παρόμοιες τεχνολογίες, μερικές από τις οποίες (Withings Aura) επιδιώκουν με την αναπαραγωγή κατάλληλης μουσικής να επιδράσουν θετικά στη ποιότητα του ύπνου.

- **Omron**[6](Omron 10, Omron M7 και άλλα): Πρόκειται για μία σειρά από συσκευές που μπορεί να φορέσει κάποιος στο μπράτσο του χεριού του για να μετρήσει την πίεση του αίματος στο σώμα του την δεδομένη χρονική στιγμή. Εξασφαλίζουν μεγάλη αξιοπιστία στις μετρήσεις που κάνουν και είναι ιδιαίτερα χρήσιμη τεχνολογία που χρησιμοποιείται είτε από ειδικούς στους ασθενείς τους, είτε και για προσωπική χρήση ειδικότερα από άτομα που πρέπει να μετράνε τακτικά την αρτηριακή τους πίεση(υπέρβαροι, υποτασικοί, υπερτασικοί κλπ). Σαφώς, υπάρχει πληθώρα τέτοιων προϊόντων που μετράνε την πίεση στο αίμα και μερικά από αυτά (Nokia BPM+[7]) παρέχουν και την δυνατότητα γραφικής παρουσίασης των δεδομένων μέσω εφαρμογών στα κινητά τηλέφωνα.

- **June Bracelet**[8]: Άλλη μια φορετή συσκευή για ειδικούς σκοπούς είναι το June Bracelet. Είναι ένα βραχιόλι που μετράει την έκθεση στον ήλιο, και γενικότερα σε επιβλαβείς ακτίνες για τον οργανισμό, ενώ δείχνει επίσης την εκάστοτε τιμή του δείκτη UV(Ultraviolet) που αποτελεί τον κύριο παράγοντα για τον καρκίνο του δέρματος.

- **SirenCare**[9]: Είναι το όνομα μιας εταιρίας που έχει εισάγει στην αγορά ειδικές κάλτσες που μετράνε και ρυθμίζουν την θερμοκρασία των ποδιών με σκοπό την αποφυγή ανάπτυξης έλκους απο διαβητικούς. Για το ίδιο κοινό χρηστών, πρόκειται να βγουν στην αγορά και άλλες συσκευές που μετράνε τα επίπεδα γλυκόζης του οργανισμού.

- **Ava**[10]: Επίσης για εξειδικευμένο κοινό, υπάρχει το βραχιόλι Ava που καταγράφει τα επίπεδα γονιμότητας του ατόμου, μέσω μετρήσεων, όπως η θερμοκρασία του σώματος, ο ρυθμός αναπνοής και η διακυμάνσεις της θερμοκρασίας του σώματος.

Φυσικά υπάρχουν κι άλλες συσκευές τέτοιου είδους, που είτε καταγράφουν γενικές πληροφορίες όπως οι πρώτες που αναφέρθηκαν είτε πιο εξειδικευμένες όπως οι τελευταίες που απευθύνονται και σε πιο συγκεκριμένο κοινό.

[5]https://www.beddit.com/
[6]https://omronhealthcare.com/
[7]https://www.apple.com/shop/product/HLBA2ZM/A/nokia-bpm-blood-pressure-monitor
[8]https://www.wareable.com/wearable-tech/netatmo-june-review-1176
[9]https://siren.care/
[10]https://www.avawomen.com/

Παρομοίως, υπάρχουν και πολλές **εφαρμογές κινητών τηλεφώνων** που μετράνε τέτοιου είδους στοιχεία. Ενδεικτικά, αναφέρονται κάποιες από αυτές στη συνέχεια.

- **MapMyFitness**[11]: Υπάρχει μια σειρά απο εφαρμογές (MapMyWalk, MapMyRun, MapMyRide, MapMyHike) που μέσω της χρήσης του Παγκόσμιου Συστήματος Στιγματοθέτησης (GPS), καταγράφει τις λεπτομέρειες της εκάστοτε αθλητικής δραστηριότητας του χρήστη(Περπάτημα, τρέξιμο, ποδηλασία, πεζοπορία). Μετράει πληροφορίες όπως την διάρκεια, την απόσταση, τον βηματισμό και την ταχύτητα του χρήστη. Επίσης, καταγράφει τις θερμίδες που έκαψε. Γενικά υπάρχουν αναρίθμητες εφαρμογές (Runkeeper, Runtastic) που μετράνε μέ αυτό το τρόπο (GPS) τα παραπάνω δεδομένα καθώς ο χρήστης βρίσκεται σε κίνηση.

- **Sworkit**[12]: Πρόκειται για μια εφαρμογή που λειτουργεί σαν τον προσωπικό σύμβουλο για την προπόνηση του χρήστη. Με άλλα λόγια, ο χρήστης διαλέγει τον τύπο αθλητικής άσκησης που θέλει και πόσο χρόνο διαθέτει. Στη συνέχεια, η εφαρμογή φτιάχνει ένα πρόγραμμα για το τί μπορεί να κάνει για να πετύχει το στόχο του. Μάλιστα, απευθύνεται κυρίως σε χρήστες που δεν καταφεύγουν στα γυμναστήρια για να αθληθούν, δίνοντας έτσι πιο εναλλακτικούς και απλούς τρόπους γυμναστικής. Και εδώ οι εφαρμογές που έχουν τον ρόλο προπονητή για τον χρήστη είναι πολλές και ποικίλες. Παρόμοιες με την παραπάνω είναι οι εφαρμογές Strava, You are your own Gym και άλλες, ενώ εφαρμογές που απευθύνονται σε κοινό που πάει στο γυμναστήριο και βοηθάνε το χρήστη καθοδηγώντας τον για τις επαναλήψεις και τα κιλά των ασκήσεων για παράδειγμα είναι οι εξής: Strong, Stacked, Jefit, Workit και πολλές ακόμη.

- **MyFitnessPal**[13]: Διαθέτει μία βάση με πάνω απο 4000 φαγητά, με αποτέλεσμα να δίνει την δυνατότητα στον χρήστη να κρατήσει επαφή με την δίαιτά του και με τις θερμίδες που καταναλώνει κάθε μέρα. Για να γίνει πιο εύκολη η καταχώρηση των τροφίμων, διαθέτει και μια λειτουργία όπου μέσω σκαναρίσματος του κωδικού που φέρει η συσκευασία του τροφίμου (barcode), μπορεί κάποιος να το καταχωρήσει αν δεν επιθυμεί να το ψάξει με το όνομά του.

Πέρα από τις παραπάνω εφαρμογές, αξίζει να αναφέρουμε και τον ρόλο κάποιων εφαρμογών διαδικτύου ή γενικότερα συστημάτων, που είναι να συναθροίζουν δεδομένα από διάφορες ενδυτές συσκευές. Σύμφωνα με την δημοσίευση *Improving Quality of Life with the Internet of Everything*[1], έχουμε τις εξής περιπτώσεις:

- **SuperTracker**[14]: πρόκειται για μια εφαρμογή διαδικτύου, που σχετίζεται με την διατροφή του χρήστη και την φυσική του δραστηριότητα. Μέσω αυτής ο χρήστης μπορεί να θέσεις τους στόχους του και να λάβει συμβουλές για τις διατροφικές του συνήθειες. Έχει επιπλέον την δυνατότητα να παρακολουθήσει ακριβώς την ποσότητα του φαγητού που καταναλώνει ανά κατηγορία τροφίμων, καθώς επίσης και την σωματική του δραστηριότητα μέσα στη μέρα.

- **eat this much**[15]: Μια εφαρμογή με παρόμοιες λειτουργίες είναι η eat this much,

[11] http://www.mapmyfitness.com/app
[12] https://sworkit.com
[13] https://www.myfitnesspal.com
[14] https://www.supertracker.usda.gov
[15] https://www.eatthismuch.com/

η οποία παρέχει πλήρες διαιτητικό πρόγραμμα στον χρήστη με βάση τους στόχους του.

- **Glooko**[16]: είναι μια πλατφόρμα που συναθροίζει δεδομένα από συσκευές καταγραφής δραστηριοτήτων και βιομετρικών δεδομένων στοχεύοντας στην παροχή χρήσιμης πληροφορίας για διαβητικούς ασθενείς.

- **Sen.se**[17]: πρόκειται για μια ανοικτή πλατφόρμα, όπου επιτρέπει τον συνδυασμό δεδομένων από διαφορετικές πηγές. Έτσι παρέχει την δυνατότητα επεξεργασίας των δεδομένων και λήψης αποφάσεων με βάση αυτά.

Τέλος, εξίσου σημαντικό ρόλο παίζουν και άλλου είδους αισθητήρες που συνήθως βρίσκονται στον περιβάλλοντα χώρο του χρήστη και μπορεί εκ πρώτης όψεως να μην συνδέονται άμεσα με δεδομένα για την φυσική κατάσταση του ατόμου. Ωστόσο, συνδυαστικά και με άλλα δεδομένα μπορούν να παρέχουν πληροφορίες χρήσιμες για την κατάσταση του χρήστη και να συμβάλλουν στη εξαγωγή αξιόπιστων και πιο συγκεκριμένων συμπερασμάτων.

Ειδικότερα, τεχνολογίες που χρησιμοποιούνται κυρίως για την ανάπτυξη της ιδέας του έξυπνου σπιτιού (smart home) μπορούν να έχουν καθοριστικό ρόλο στη συλλογή δεδομένων υγείας του χρήστη. Για παράδειγμα, **αισθητήρες πίεσης** (pressure sensors) **σε καναπέδες, κρεβατια** και άλλα, δίνουν την δυνατότητα ανίχνευσης της θέσης του ατόμου και της δραστηριότητας του, δηλαδή αν κάθεται, αν πιθανώς κοιμάται και λοιπά. Στην ίδια λογική, με τη χρήση **αισθητήρων σε τηλεοράσεις, ραδιόφωνα**, μπορεί κάποιος να συλλέξει πληροφορίες για το τί πιθανώς κάνει ο χρήστης. Αυτή η πληροφορία, συνδυαστικά με την θέση του μέσα στο σπίτι, δημιουργεί μια συνολική εικόνα της κατάστασης του, που είναι χρήσιμη για την παρακολούθηση του καθημερινού τρόπου ζωής του ατόμου μέσα στο σπίτι του. Βεβαίως, όσον αφορά συσκευές για 'έξυπνο σπίτι', πρέπει να σημειωθεί ότι πρόκειται για μια πρόσφατη ιδέα, που τώρα εξελίσσεται. Οι ρυθμοί, ωστόσο, ανάπτυξης αυτής της ιδέας είναι ιδιαίτερα γρήγοροι με αποτέλεσμα, στο άμεσο μέλλον, αυτές οι συσκευές να εισέλθουν στα σύγχρονα σπίτια και να αξιοποιηθούν, πέρα των άλλων, και για τους σκοπούς που αναλύθηκαν παραπάνω.

Συνοψίζοντας, θα μπορούσε κανείς να αφιερώσει πολύ χρόνο στο παραπάνω κομμάτι για την υφιστάμενη κατάσταση των ενδυτών συσκευών, των εφαρμογών και των αισθητήρων που καταλαμβάνουν τον πρωταρχικό ρόλο στη συλλογή δεδομένων υγείας και δεικτών ευημερίας. Ωστόσο και με την παραπάνω ανάλυση, είναι προφανής η ποικιλία που υπάρχει σε κάθε είδος πηγών δεδομενων καθώς και η πληθώρα των συσκευών που καταγράφουν ίδιου τύπου δεδομένα για έναν χρήστη. Αυτό οδηγεί σε μια σειρά απο προβλήματα που σχετίζονται με την διαχείριση της πληροφορίας, τα οποία περιγράφονται πιο αναλυτικά στην ακόλουθη ενότητα.

2.2 Περιγραφή του προβλήματος

Η περιγραφή της κατάστασης που επικρατεί αυτή τη στιγμή στο τομέα της τεχνολογίας που μας αφορά, καθώς και οι ταχείς ρυθμοί εξέλιξης του, οδηγεί στο σαφές συμπέρασμα ότι οι μετρήσεις που μπορεί να μαζέψει κάποιος για δείκτες προέρχονται από εξαιρετικά πολλές πηγές δεδομένων.

[16] https://www.glooko.com/
[17] https://sen.se/store/c/peanuts

Όπως είδαμε και παραπάνω, για κάθε είδους δεδομένο υπάρχει μια μεγάλη ποικιλία συσκευών και εφαρμογών που έχουν την δυνατότητα μέτρησής του. Αυτό οδηγεί τον χρήστη στην επιλογή των ελάχιστων συσκευών ή εφαρμογών για τη μέτρηση της πληροφορίας που επιθυμεί να παρατηρήσει. Πρόκειται για ένα γεγονός που είναι φυσικό να συμβαίνει, γιατί ο χρήστης μπορεί και με μία συσκευή να καταγράψει την μέτρηση που τον αφορά χωρίς να χρειάζεται να καταναλώνει χρόνο σε παραπάνω απο μία εφαρμογές. Επίσης, δεν θα συγχέεται από πιθανές διαφοροποιήσεις που θα είχαν παραπάνω από μια συσκευές-εφαρμογές για την μέτρηση ενός ίδιου δείκτη.

Επιπροσθέτως, είναι λογικό κάθε συσκευή μέτρησης να μετράει διαφορετικά τα δεδομένα που προέρχονται κατά τη φυσική δραστηριότητα του χρήστη, είτε αυτά αφορούν μετρικές του οργανισμού του ατόμου(π.χ καρδιακοι παλμοί ή αρτηριακή πίεση του αίματος), είτε αφορούν γενικότερες πληροφορίες για την εκάστοτε δραστηριότητα του(π.χ. χιλιομετρική απόσταση που διήνυσε, ένταση αθλητικής δραστηριότητας κλπ). Υπάρχει, λοιπόν, αναμφισβήτητα μια **ποικιλία στις τιμές που μπορούν να συλλεχθούν απο τη μέτρηση του ίδιου δείκτη από διαφορετικές πηγές**. Αυτό αποτελεί ένα κύριο χαρακτηριστικό του προβλήματος, αφού εύλογα αναρωτιέται κανείς ποιά τιμή απο τις συλλεχθείσες είναι πιο ακριβής, ποιά συσκευή μέτρησης είναι πιο αξιόπιστη ή και ποιά μέθοδος καταγραφής εν τέλει είναι κατάλληλη για την παρακολούθηση ενός συγκεκριμένου δεδομένου υγείας.

Σε δεύτερο επίπεδο, είναι προφανές ότι οι τρόποι αναπαράστασης ίδιου τύπου πληροφορίας είναι αναρίθμητοι. Με άλλα λόγια, αν πάρει κάποιος διάφορες μετρήσεις για τον ίδιο δείκτη θα παρατηρήσει την μεγάλη ποικιλία στον τρόπο που αποθηκεύονται τα δεδομένα από κάθε εφαρμογή. Αυτή η ποικιλία στην μορφή αναπαράστασης της πληροφορίας οφείλεται στον διαφορετικό τρόπο που επιλέγει κάθε εφαρμογή να αναπαριστά τα δεδομένα της για να τα επεξεργαστεί κατά πώς επιθυμεί και να βγάλει τα αποτελέσματά της.

Ανακεφαλαιώνοντας λοιπόν, τα προβλήματα που προκύπτουν είναι τόσο ο **τεράστιος όγκος δεδομένων υγείας και δεικτών ευημερίας** που συλλέγεται απο ενδυτές συσκευές, εφαρμογές κινητών τηλεφώνων και άλλους αισθητήρες, όσο και οι **διαφορετικοί τρόποι αναπαράστασής** τους. Σαν αποτέλεσμα, καθίσταται δυσχερής η προσπάθεια διασύνδεσης και επεξεργασίας της πληροφορίας και κατ΄ επέκταση η εξαγωγή ορθών συμπερασμάτων για τη ζωή του χρήστη.

2.3 Εφαρμογές του Διαδικτύου των Πραγμάτων στον τομέα της Υγείας

Σε αυτό το σημείο, και αφού έχουν αναλυθεί τα ζητήματα που προκύπτουν λόγω του μεγάλου όγκου των δεδομένων, είναι σημαντικό να αναφερθούμε γενικότερα στον όρο 'Διαδίκτυο των Πραγμάτων' ("Internet of Things"). Ουσιαστικά πρόκειται για την ανάπτυξη τεχνολογιών που στοχεύουν στην διασύνδεση πληροφορίας η οποία συλλέγεται από οποιαδήποτε ηλεκτρονική συσκευή μπορεί να σκεφτεί κάποιος. Στις μέρες μας, ειδικά, **το 'Διαδίκτυο των Πραγμάτων', δημιουργεί απεριόριστες δυνατότητες και προοπτικές για την εξέλιξη κάθε τομέα** στον οποίο μπορεί να αναπτυχθεί ένα τέτοιο δίκτυο.

Αυτή η διπλωματική εργασία επικεντρώνεται στην εφαρμογή της παραπάνω ιδέας σε τομείς που αφορούν την φυσική κατάσταση και υγεία του ατόμου με σκοπό την συμβολή στο ευ ζην του[2]. Συγκεκριμένα, οι δυνατότητες που μπορεί να παρέχει η υλοποίηση του 'Διαδικτύου των Πραγμάτων' στην αξιοποίηση δεδομένων υγείας είναι εξαιρετικά καθοριστικές για τη δημιουργία μιας συνολικής εικόνας του καθημερινού τρόπου ζωής του ατόμου. Η καθολική αυτή εικόνα της υγείας είναι αναντίρρητα απαραίτητη για να προταθούν στοχευμένες λύσεις για τη βελτίωση της ζωής του ατόμου μέσα από σωστή καθοδήγηση μέσω μιας εφαρμογής[3]. Με πιο απλά λόγια, φανταστείτε την προσφορά ενός ασφαλούς συστήματος που θα μπορούσε να συσχετίσει και να αξιοποιήσει καθολικά όλα τα συλλεχθέντα δεδομένα υγείας από όποια πηγή κι αν προέρχονται.

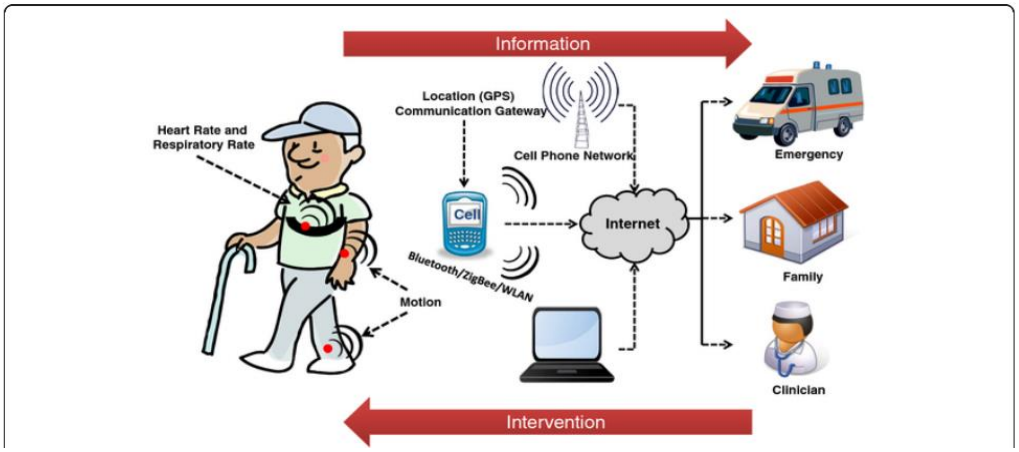
Προκειμένου να επιτευχθεί η διασύνδεση των δεδομένων αυτών, έχουν αναπτυχθεί κάποια πρότυπα για τη μορφή της αναπαράστασης και το τρόπο επεξεργασίας της πληροφορίας που σχετίζεται με την υγεία. Για παράδειγμα, υπάρχουν πλατφόρμες που επιτρέπουν την συλλογή δεδομένων υγείας και την μετατροπή τους σε προκαθορισμένες μορφές[1]. Σε αυτές τις πλατφόρμες γίνονται χρήση και συγκεκριμένων προτύπων όπως το HL7, το LOINC(Logical Observation Identifiers Names and Codes) και άλλα.

Επιπροσθέτως, στα πλαίσια του Διαδικτύου των Πραγμάτων, εχουν αρχίσει να αναπτύσσονται πολλές εφαργογές τόσο **στον χώρο του αθλητισμού**, όσο και **στον χώρο της υγείας** γενικότερα[4]. Με άλλα λόγια, παρατηρείται πως με την συνάθροιση τέτοιων δεδομένων, μπορεί ένα άτομο να βελτιώσει την φυσική του κατάσταση και να παρακολουθήσει διάφορους δείκτες σχετικούς με την υγεία του. Έτσι, έχει την δυνατότητα να προλαμβάνει δυσχερείς καταστάσεις, αφού μπορεί να παρατηρήσει μέσω μετρικών ζωτικής σημασίας την εκάστοτε κατάσταση του οργανισμού του. Στη συνεχεια, αν παρατηρήσει κάποια ανωμαλία στις τιμές αυτών των δεικτών ευημερίας μπορεί να απευθυνθεί στον γιατρό του.

Εμβαθύνοντας λίγο παραπάνω στις εφαρμογές στον χώρο της υγείας, αξίζει να αναφέρουμε ότι υπάρχουν πολλοί ενδυτοί αισθητήρες, οι οποίοι μετράνε δείκτες ζωτικής σημασίας όπως το οξυγόνο στο αίμα του ατόμου, τη πίεση του, τη θερμοκρασία του σώματος και διάφορα άλλα. Μάλιστα, μέσω αυτών μπορεί να επιτευχθεί παρακολούθηση ατόμων που έχουν κάποια ασθένεια ή σωματική δυσλειτουργία[5].

Σε επόμενο στάδιο, μπορεί να σκεφτεί κανείς την σύνδεση χρηστών διαφορετικών ρόλων μεσα σε ένα σύστημα, όπου από την μια μεριά θα βρίσκονται οι απλοί χρήστες και από την άλλη χρήστες διαφόρων ειδικοτήτων. Οι τελευταίοι, θα μπορούσαν να συλλέξουν δεδομένα υγείας από τους απλούς χρήστες και να μελετήσουν διάφορες ασθένειες και ιατρικές μεθόδους. Επιπλέον, θα μπορούσαν να βοηθήσουν στην πρόληψη και διάγνωση ασθενειών και στην ίαση τους, συμβουλεύοντας κατάλληλα τους χρήστες. Με άλλα λόγια, η ανάπτυξη ενός συστήματος απομακρυσμένης παρακολούθησης της υγείας των χρηστών, με χρήση ενδυτών συσκευών, θα ήταν μια καθοριστική λύση στην βελτίωση της ποιότητας ζωής.

Με βάση ένα τέτοιο σύστημα, μπορεί εύκολα να σκεφτεί κανείς τις εφαρμογές του στην σύγχρονη πραγματικότητα. Ενδεικτικά, μια πολύ πετυχημένη εφαρμογή σχετίζεται με την παρακολούθηση της υγείας ενός ηλικιωμένου ατόμου. Με την χρήση αισθητήρων στο σπίτι του, ή μέσω μικρών συσκευών που φοράει, υπάρχει η δυνατότητα να παρατηρούμε την κατάσταση υγείας του από μακρία και να του παρέχουμε βοήθεια όποτε κρίνεται απαραίτητο (Σχήμα 2.1[6]).



Σχήμα 2.1: Απομακρυσμένη παρακολούθηση υγείας ατόμου με χρήση ενδυτών συσκευών[6]

Κλείνοντας, είναι χρήσιμο να αναφερθούμε και σε κάποιες ολοκληρωμένες εφαρμογες του Διαδικτύου των Πραγμάτων στον χώρο της υγείας, που έχουν αναπτυχθεί ήδη τα τελευταία χρόνια. Ενδιαφέρον παρουσιάζει για παράδειγμα, η ανάπτυξη ενός 'Ψηφιακού Νοσοκομείου'[7], στον Καναδά που παρέχει ραντεβού με γιατρούς μέσω διαδικτύου καθώς και παρακολούθηση υγείας των ασθενών σε πραγματικό χρόνο.

2.4 Ανάγκες στη διαδικασία διασύνδεσης της πληροφορίας

Έχοντας, λοιπόν, αναφερθεί και στις πρώτες προσπάθειες που έχουν γίνει για την συλλογή των δεδομένων υγείας σύμφωνα με κάποια πρότυπα, θα επικεντρωθούμε σε αυτό το κομμάτι στις ανάγκες που υπάρχουν για να υλοποιηθεί ο σωστός συσχετισμός της πληροφορίας που συλλέγεται από συσκευές, αισθητήρες και εφαρμογές για έναν χρήστη.

Αρχικά, είναι αναγκαία η **προτυποποίηση των δεδομένων υγείας και των δεικτών ευημερίας**, όποια και αν είναι η μορφή τους. Με άλλα λόγια, πρέπει να υπάρχει η δυνατότητα, η μορφή όλων των δεδομένων που συλλέγονται να ακολουθούν ένα συγκεκριμένο μοντέλο, ή αλλιώς να μπορούν να εκφραστούν με ένα κοινό και καθολικό λεξιλόγιο. Αν λοιπόν, κάθε πηγή τέτοιων δεδομένων ακολουθεί το ίδιο λεξιλόγιο, τότε τα οφέλη θα ήταν πολύ σημαντικά, τόσο στην ευκολία συλλογής και αποθήκευσης της πληροφορίας, όσο και στις δυνατότητες άμεσης επεξεργασίας της που θα οδηγούσε σε καίρια συμπεράσματα.

Εμβαθύνοντας λίγο στον όρο κοινό λεξιλόγιο, αξίζει να γίνει αναφορά στις **τεχνολογίες σημασιολογικού ιστού** (semantic web) που παρέχουν, μεταξύ άλλων, την δυνατότητα ανάπτυξης μιας γλώσσας (semantic web language[18]) ικανής να αναπαραστήσει πολύπλοκη γνώση για αντικείμενα(οντότητες) καθώς επίσης και τις σχέσεις που προκύπτουν μεταξύ τους. Επομένως, πρόκειται για κάτι απαραίτητο για την εφαρμογή του 'Διαδικτύου των Πραγμάτων' και σε αυτόν το τομέα.

[18]https://www.w3.org/standards/semanticweb/

Επιπλέον, πέρα από την μοντελοποίηση της πληροφορίας που συλλέγεται, επιτακτική είναι και η **ανάγκη ορθής διαχείρισής** της σε επόμενο στάδιο. Όπως, προαναφέρθηκε, ένα ζήτημα που προκύπτει από τον όγκο των δεδομένων είναι ο μεγάλος αριθμός, πιθανώς διαφορετικών τιμών και παρατηρήσεων για την ίδια μέτρηση. Αυτό το χαρακτηριστικό του προβλήματος, κάνει εμφανή την ανάγκη για **ανάπτυξη ενός συστήματος**, που θα έχει την δυνατότητα αρχικά να ξεχωρίσει ποιές τιμές είναι λογικές και ποιές όχι και εν συνεχεία να παρουσιάζει στον χρήστη του συστήματος όσο το δυνατόν πιο ακριβή αποτελέσματα. Με αυτόν το τρόπο, κάθε χρήστης θα μπορούσε να τροφοδοτήσει ένα τέτοιο σύστημα με πληθώρα δεδομένων από πολλές και διαφορετικές συσκευές και να πάρει σαν αποτέλεσμα μια συνολική εικόνα της υγείας και της φυσικής του κατάστασης. Βεβαίως, απαραίτητη προϋπόθεση είναι η **ασφαλής διαχείριση των δεδομένων** εισόδου και η επιστημονική τους επεξεργασία, καθώς πρόκειται για πληροφορίες με ευαίσθητο περιεχόμενο και με σημαντικό αντίκτυπο στη ζωή του ατόμου.

Κεφάλαιο 3

Ανάλυση απαιτήσεων και γενική αρχιτεκτονική συστήματος

Στο παρόν κεφάλαιο, αναλύονται οι απαιτήσεις του συστήματος και της εφαρμογής προτού περάσουμε στον σχεδιασμό. Ειδικότερα, περιγράφονται οι τρόποι συλλογής δεδομένων, οι λειτουργίες της εφαρμογής καθώς και η συνολική αρχιτεκτονική του συστήματός μας. Ακόμα, γίνεται αναφορά στις τεχνολογίες που χρησιμοποιήθηκαν συνολικά για την ανάπτυξη του συστήματος.

3.1 Απαιτήσεις συστήματος

Κυρίαρχο ρόλο στην ανάπτυξη του συστήματος έχουν οι δείκτες ευημερίας που συλλέγονται και ο τρόπος που γίνεται η επεξεργασία τους στη πορεία. Επομένως, οι προϋποθέσεις για να παρέχει η εφαρμογή τα επιθυμητά αποτελέσματα βασίζονται στο τρόπο αλληλεπίδρασης με τον χρήστη ώστε μέσω σωστής καθοδήγησης να βελτιωθεί ο καθημερινός τρόπος ζωής του, καθώς επίσης και στη διαδικασία διαχείρισης των δεδομένων σε όλα τα επίπεδα.

3.1.1 Αλληλεπίδραση με τον χρήστη

Οι απαιτήσεις του συστήματος και ειδικότερα της εφαρμογής σχετίζονται, σε πρώτο πλάνο, με τους τρόπους που θα αλληλεπιδρά με τον χρήστη.

Άξιο αναφοράς σε αυτό το σημείο είναι ότι η κύρια αλληλεπίδραση, που επιθυμούμε να αναπτυχθεί, βασίζεται από τη μια μεριά στην προσεγμένη παρουσίαση των δεικτών ευημερίας του ατόμου και από την άλλη στη προσπάθεια συμβούλευσης με σκοπό το ευ ζην του. Κύριο μέλημα λοιπόν, είναι το πώς η εφαρμογή θα παρέχει τα τελικά αποτελέσματα στον χρήστη.

Αρχικά, είναι εξέχουσας σημασίας να υπάρχει ένα **φιλικό προς το χρήστη γραφικό περιβάλλον** (Graphical User Interface - GUI). Πιο συγκεκριμένα, θέλουμε η πλοήγηση στις οθόνες της εφαρμογής να γίνεται με εύκολο και άμεσα κατανοητό τρόπο, ώστε ο χρήστης να μάθει γρήγορα τις δυνατότητες που του παρέχονται. Έχει σημασία, λοιπόν να χωριστούν με λογικό τρόπο οι οθόνες, ώστε να είναι σαφές το τί μπορεί να βρει σε καθεμία ο χρήστης. Ακόμη, χρειάζεται μεθοδικότητα ώστε να επιλεγούν για προβολή στις οθόνες της εφαρμογής τα δεδομένα που έχουν πραγματική ουσία για τον μέσο χρήστη.

Ακόμη, ο τρόπος που θα δίνονται οι συμβουλές στο χρήστη θέλει ιδιαίτερη προσοχή, διότι πρέπει να τον παροτρύνουν με αποτελεσματικότητα αλλά ταυτόχρονα και διακριτικότητα. Δηλαδή, είναι επιθυμητό να διατυπώνονται προτάσεις προς τον χρήστη που θα του δίνουν κίνητρο να κάνει αυτό που θεωρείται σωστό από την εφαρμογή για να βελτιώσει την ποιότητα ζωής του. Πρέπει, λοιπόν, κάθε συμβουλή να δίνεται με εμφανή και συνάμα παρακινητικό τρόπο. Παράλληλα, θυμίζουμε ότι πρόκειται απλά για μία εφαρμογή κινητών τηλεφώνων οπότε ο ρόλος της δεν πρέπει να φτάσει σε σημείο που να είναι παρεμβατικός στη ζωή του ατόμου. Απαιτείται, επομένως, διακριτικότητα στον τρόπο αλληλεπίδρασης με το χρήστη, ώστε να είναι εμφανής ο **καθοδηγητικός σκοπός** της εφαρμογής.

3.1.2 Ασφάλεια δεδομένων

Σε δεύτερο επίπεδο, είναι σημαντικό να αναφερθούμε στη διαχείριση των δεδομένων και των χρηστών καθώς και στις απαιτήσεις ασφαλείας του συστήματος σχετικά με αυτό το κομμάτι.

Αρχικά, όπως έχει προαναφερθεί, η **ιδιωτικότητα των δεδομένων** παίζει κυρίαρχο ρόλο, λόγω του ευαίσθητου περιεχομένου της πληροφορίας. Μπορεί αυτό το κομμάτι να μην αποτελει το επίκεντρο αυτής της διπλωματικής, ωστόσο η ασφαλής συλλογή και αποθήκευση των δεδομένων υγείας του ατόμου αποτελεί μείζον ζήτημα. Απαιτείται, λοιπόν, **κρυπτογραφημένη επικοινωνία** μεταξύ των συστατικών του συστήματος που ανταλλάσσουν τέτοια πληροφορία και για αυτό το λόγο χρησιμοποιήθηκαν βασικές τεχνικές κρυπτογράφησης. Σε δεύτερη φάση, θεωρήθηκε ότι η επικοινωνία με τις βάσεις δεδομένων γίνεται με συγκεκριμένους τρόπους προκειμένου να επιτευχθεί η ασφαλής αποθήκευση της πληροφορίας.

Τέλος, η διαχείριση των χρηστών πρέπει να γίνεται με προσοχή, πάλι για λόγους ασφαλείας. Καθίσταται με άλλα λόγια αναγκαία η **αυθεντικοποίηση** και **πιστοποίησή** τους προκειμένου να αλληλεπιδρούν με το σύστημα μόνο όσοι έχουν το δικαίωμα, με αποτέλεσμα να εξαλειφθεί ο κίνδυνος για διαρροή της πληροφορίας. Όπως και προηγουμένως, η διαχείριση και αποθήκευση των προσωπικών δεδομένων κάθε χρήστη οφείλει να γίνεται από το σύστημά μας με κατάλληλες τεχνικές κρυπτογραφίας.

3.1.3 Επιλογή δεδομένων

Τέλος, είναι απαραίτητο να περιγραφεί η συλλογή των δεδομένων υγείας και των δεικτών ευημερίας με τα οποία θα τροφοδοτηθεί το σύστημά μας. Αρχικά, είναι γνωστό και από τα προηγούμενα κεφάλαια, ότι αυτά τα δεδομένα δημιουργούνται από **εφαρμογές ενδυτών συσκευών, κινητών τηλεφώνων και από διάφορους αισθητήρες** κυρίως από τον περιβάλλοντα χώρο του χρήστη. Επομένως, αυτά είναι και τα ακατέργαστα δεδομένα που χρειάζεται και το σύστημά μας σαν είσοδο.

Ιδανικά, η συγκέντρωση όλων αυτών των δεδομένων στην εφαρμογή μας θα γινόταν μέσω διεπαφών προγραμματισμού των εφαρμογών (Application Programming Interface - API). Ειδικότερα, οι πηγές δεδομένων παρέχουν στο κοινό συγκεκριμένες διευθύνσεις των πόρων τους. Σαν αποτέλεσμα, με τη χρήση του Εννιαίου Εντοπιστή Πόρων (Uniform Resource Locator - URL) η εφαρμογή μας θα είχε την δυνατότητα να ζητήσει τα συλλεχθέντα δεδομένα για έναν συγκεκριμένο χρήστη. Φυσικά, για να γίνει αυτό πρέπει ο ίδιος ο χρήστης να δώσει

την συγκατάθεση του, καθώς πρόκειται για προσωπικά δεδομένα, οπότε πρέπει να επιτρέψει την πρόσβαση σε αυτά από την εφαρμογή. Απόρροια της παραπάνω διαδικασίας θα ήταν η συνάθροιση των δεδομένων υγείας όλων των χρηστών από οποιαδήποτε συσκευή ή εφαρμογή έχει συνδέθεί στη δική μας εφαρμογή.

Σε πρακτικό επίπεδο, ωστόσο, και για τους σκοπούς της διπλωματικής εργασίας τα δεδομένα που θα χρησιμοποιηθούν σαν 'είσοδος' στο σύστημα μας θα είναι **προσομοιωτικά** και δημιουργημένα από μια **γεννήτρια**, καθότι δεν βρέθηκε κάποιο έτοιμο και πλήρες σύνολο δεδομένων (dataset) που να ικανοποιεί τις ανάγκες του συστήματός μας. Ακόμα, οι **συσκευές και εφαρμογές που επιλέχθηκαν** σαν βάση για να δημιουργήσουμε τα δεδομένα μας, **είναι πεπερασμένες και συγκεκριμένες**, αφού στη τρέχουσα πραγματικότητα δεν είναι δυνατή η σύνδεση κάθε είδους συσκευής ή εφαρμογής λόγω του διαφορετικού τρόπου αναπαράστασης της πληροφορίας και της έλλειψης ενός κοινού λεξιλογίου. Φυσικά, δώσαμε ιδιαίτερη προσοχή στην απαίτηση να καλυφθεί το μεγαλύτερο εύρος τέτοιων πηγών δεδομένων, δηλαδή με άλλα λόγια επιλέχθηκαν πηγές διαφορετικών ειδών για να έχουμε καλύτερη προσέγγιση του πραγματικού σεναρίου. Τέλος, αυτά τα δεδομένα δεν θα τα ζητάει το σύστημά μας μέσω κάποιας διεπαφής, αλλά αντίθετα θα στέλνονται κατευθείαν σε αυτό από τη γεννήτρια δεδομένων που θα αναλάβει την παραγωγή τους.

Επιπλέον, χωρίς βλάβη της γενικότητας θα θεωρήσουμε για τους σκοπούς της εργασίας ότι **τα προσωμοιωτικά δεδομένα** που εξετάζουμε **αφορούν έναν χρήστη**. Παρόλο δηλαδή που θα υποστηρίζεται η εγγραφή και σύνδεση παραπάνω χρηστών, τα δεδομένα θα αφορούν πάντα τον ίδιο 'εικονικό' χρήστη, διότι δεν μας ενδιαφέρει να ελέγξουμε την συμπεριφορά του συστήματος στη περίπτωση πολλών χρηστών με πολλά δεδομένα. Τα ζητήματα που μπορεί να προκύψουν από την προσομοίωση πολλών χρηστών δεν σχετίζονται με τους σκοπούς της διπλωματικής. Άλλωστε, υπάρχουν πολλά εργαλεία της τεχνολογίας(όπως τεχνολογίες υπολογιστικού νέφους), που χρησιμοποιούνται από σύγχρονα συστήματα για να εξυπηρετήσουν μεγάλο αριθμό χρηστών, τα οποία όμως δεν υπάρχει λόγος να ενσωματωθουν σε αυτήν την εργασία.

Οι λόγοι που μπορεί κάποιος να θεωρήσει τις παραπάνω παραδοχές με ασφάλεια, είναι ότι στη πραγματικότητα δεν επηρεάζουν τα κομμάτια στα οποία στοχεύει να επικεντρωθεί η διπλωματική. Πιο συγκεκριμένα, αρκεί να φροντίσουμε η πληροφορία που προσομοιώνεται να ακολουθεί τη δομή που θα είχε αν τη συλλέγαμε κανονικά μέσω της διεπαφής κάθε εφαρμογής-συσκευής. Έτσι, θα δημιουργηθούν αληθοφανή δεδομένα ενός μέσου χρήστη της εφαρμογής μας και βασισμένοι σε αυτά θα μελετηθεί η ιδέα της προτυποποίησης τους και η μετέπειτα επεξεργασία και διασύνδεσή τους για την εξαγωγή χρήσιμων συμπερασμάτων.

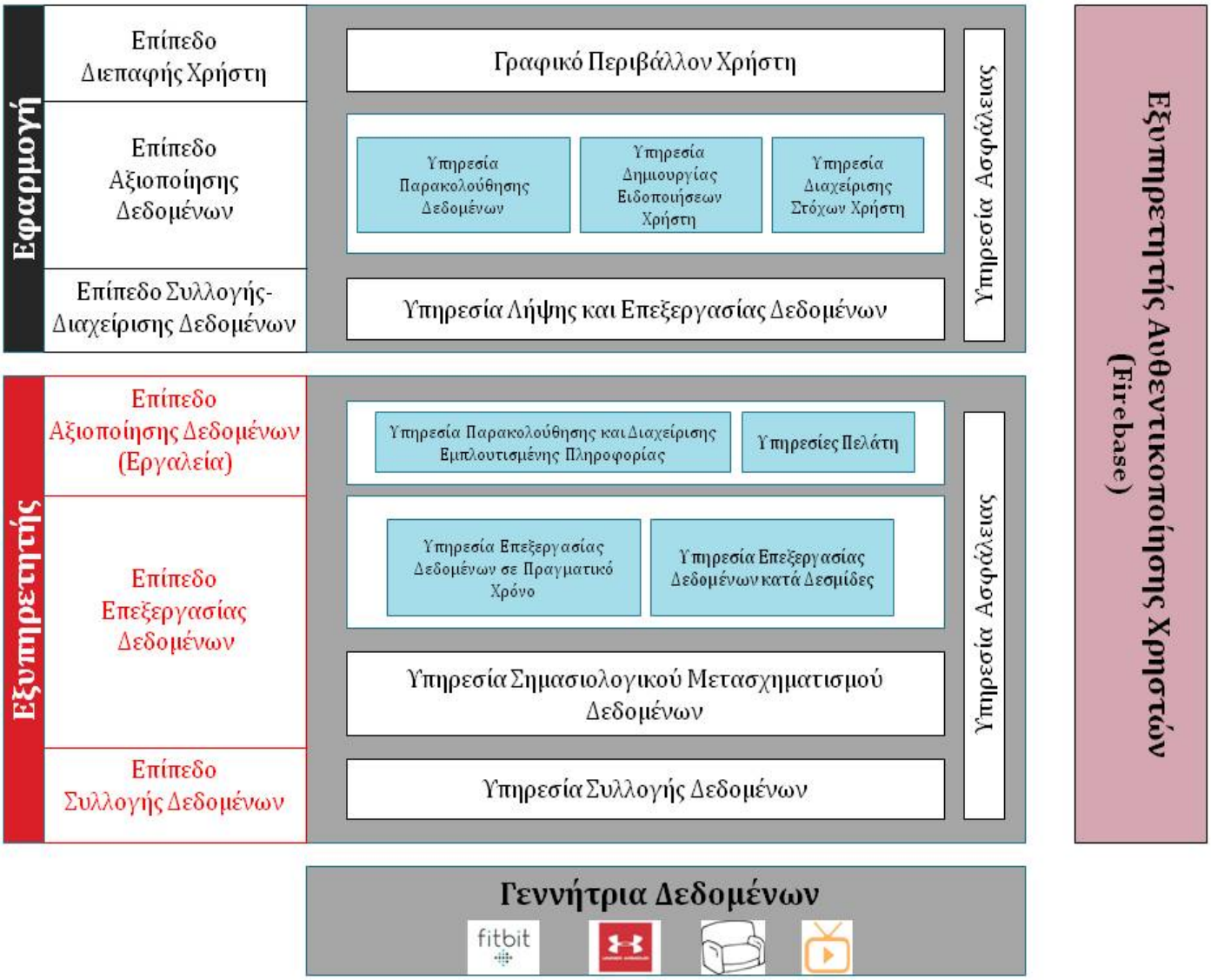
3.2 Συνολική αρχιτεκτονική του συστήματος

Σε αυτό το σημείο, γίνεται ανάλυση της αρχιτεκτονικής του συστήματος(εξυπηρετητής, εφαρμογή) που αναπτύχθηκε και των βοηθητικών εξυπηρετητών(γεννήτρια δεδομένων, εξυπηρετητής αυθεντικοποίησης). Για το σκοπό αυτό, περιγράφονται συνοπτικά και οι λειτουργίες τους.

3.2.1 Γενική αρχιτεκτονική

Παρατίθεται παρακάτω ένα γενικό σχήμα (Σχήμα 3.1), όπου φαίνονται η γεννήτρια δεδομένων, τα επίπεδα αρχιτεκτονικής του εξυπηρετητή και της εφαρμογής καθώς και ο ρόλος του εξυπηρετητή αυθεντικοποίησης χρηστών (Firebase[9]) που χρησιμοποιήθηκε.

Με στόχο μια πρώτη ανάλυση στην αρχιτεκτονική του συστήματος και στις επιλογές που έγιναν, παρουσιάζονται στη συνέχεια περιγραφικά τα επίπεδα λειτουργιών κάθε συστατικού στοιχείου του σχήματος. Ολοκληρώνοντας αυτό το κομμάτι, θα έχει κανείς μια γενική εικόνα της συνολικής αρχιτεκτονικής ώστε να περάσει στη συνέχεια ομαλά στη σχεδίαση και την υλοποίηση.



Σχήμα 3.1: Γενική αρχιτεκτονική συστήματος

3.2.2 Αρχιτεκτονική εξυπηρετητή

Έχοντας υπόψιν το κομμάτι του εξυπηρετητή στο γενικό σχήμα της αρχιτεκτονικής, παρατηρούμε ότι οι υπηρεσίες του χωρίζονται σε τρία επίπεδα και συγκεκριμένα στο **επίπεδο συλλογής δεδομένων**, στο **επίπεδο επεξεργασίας** τους και στο **επίπεδο αξιοποίησης** τους. Στη συνέχεια, αναλύεται το κάθε επίπεδο ξεχωριστά και οι λειτουργίες που παρέχει η κάθε υπηρεσία.

Στο κατώτερο, λοιπόν στρώμα του εξυπηρετητή που αποτελεί το επίπεδο συλλογής δεδομένων υπάρχει μια υπηρεσία (**Υπηρεσία Συλλογής Δεδομένων**), που αναλαμβάνει την λήψη όλων των ακατέργαστων δεδομένων (raw data). Πρόκειται ουσιαστικά για τα προσομοιωτικά δεδομένα που έχει δημιουργήσει η γεννήτρια. Αυτά αποστέλλονται κρυπτογραφημένα σε συγκεκριμένες διευθύνσεις (Uniform Resource Locator - URL) της υπηρεσίας του εξυπηρετητή, η οποία κάνει τις απαραίτητες ενέργειες για αποκρυπτογράφηση και αποθήκευση τους σε κατάλληλες βάσεις δεδομένων.

Το επόμενο επίπεδο είναι αυτό της επεξεργασίας των δεδομένων. Σε αυτό, υπάρχει αρχικά μια υπηρεσία που μετασχηματίζει τα ακατέργαστα δεδομένα σε σημασιολογικά (**Υπηρεσία Σημασιολογικού Μετασχηματισμού Δεδομένων**). Η μετατροπή γίνεται με βάση την οντολογία που αναπτύχθηκε για τους δείκτες ευημερίας του ατόμου. Μετά λοιπόν από αυτή τη πρώτη επεξεργασία, αναλαμβάνουν δράση υπηρεσίες που διαχειρίζονται σε δεύτερη φάση τα σημασιολογικά πλέον δεδομένα. Εκεί, μπορούμε να βρούμε την **Υπηρεσία Επεξεργασίας Δεδομένων κατά Δεσμίδες** (batch processing) και την **Υπηρεσία Επεξεργασίας Δεδομένων σε Πραγματικό Χρόνο** για πληροφορία που χρήζει άμεσης διαχείρισης.

Στο ανώτερο επίπεδο, γίνεται η αξιοποίηση των επεξεργασμένων δεδομένων μέσω των εργαλείων που παρέχει ο εξυπηρετητής. Χρειάζεται επομένως, η **Υπηρεσία Παρακολούθησης και Διαχείρισης Εμπλουτισμένης Πληροφορίας** καθώς και ένα σύνολο από **Υπηρεσίες Πελάτη** για την επικοινωνία της εφαρμογής με τον εξυπηρετητή.

Κάθετα σε όλα τα παραπάνω επίπεδα, δρα η **Υπηρεσία Ασφάλειας** του εξυπηρετητή. Οι λειτουργίες της, σχετίζονται με όλες τις απαραίτητες ενέργειες που πρέπει να γίνουν ώστε να διασφαλιστεί η ιδιωτικότητα των δεδομένων των χρηστών. Αναλαμβάνει, ειδικότερα, την κρυπτογράφηση των δεδομένων καθώς και την πιστοποίηση των χρηστών της εφαρμογής μέσω χρήσης κατάλληλων λειτουργιών που παρέχονται από τον **Εξυπηρετητή Αυθεντικοποίησης Χρηστών (Firebase)** που χρησιμοποιήθηκε.

3.2.3 Αρχιτεκτονική εφαρμογής

Όπως φαίνεται στο σχήμα, τα επίπεδα, στα οποία κατανέμονται οι υπηρεσίες της εφαρμογής είναι το **επίπεδο συλλογής και διαχείρισης των δεδομένων**, το **επίπεδο αξιοποίησης** τους και τέλος το **επίπεδο διεπαφής** με τον τελικό χρήστη.

Στο κατώτερο επίπεδο, υπάρχει η **Υπηρεσία Λήψης και Επεξεργασίας Δεδομένων**, που συλλέγει τα δεδομένα στην μορφή που έχουν καταλήξει μετά την επεξεργασία τους από τον εξυπηρετητή. Σε αυτή την υπηρεσία γίνονται οι κατάλληλες ενέργειες για να λαμβάνεται σε πραγματικό χρόνο η πληροφορία. Πραγματοποιείται επίσης και η επεξεργασία της πληροφορίας, αφού υπάρχουν περιπτώσεις που λαμβάνονται διαφορετικές

τιμές για τον ίδιο δείκτη ευημερίας. Αυτό συμβαίνει, γιατί διαφορετικές πηγές καταγράφουν ίδιου είδους δεδομένα, τα οποία ναι μεν έχουν μετατραπεί στην ίδια οντολογική μορφή από τον εξυπηρετητή αλλά από την άλλη ενδέχεται να περιέχουν πολλαπλές τιμές για την ίδια μετρική. Καθίσταται λοιπόν απαραίτητο να παραχθεί από την υπηρεσία μια τελική και ολοκληρωμένη πληροφορία για τους δείκτες ευημερίας στον χρήστη.

Έχοντας, λοιπόν, δημιουργήσει την τελική πληροφορία, πραγματοποιείται η αξιοποίηση της από τις υπηρεσίες που βρίσκονται στο επίπεδο αξιοποίησης των δεδομένων. Εκεί υπάρχουν τρεις υπηρεσίες. Η πρώτη είναι η **Υπηρεσία Παρακολούθησης Δεδομένων**, που συλλέγει τα τελικά δεδομένα που δημιουργήθηκαν από το προηγούμενο στρώμα και αναλαμβάνει να ενημερώνει το επίπεδο διεπαφής χρήστη όταν πρέπει να ανανεωθούν οι οθόνες της εφαρμογής. Η δεύτερη είναι η **Υπηρεσία Δημιουργίας Ειδοποιήσεων Χρήστη**, που αναλαμβάνει να ενημερώσει το στρώμα διεπαφής, για τα σημαντικά συμπεράσματα που προέκυψαν φτιάχνοντας κατάλληλες ειδοποιήσεις για την γραμμή εργασιών στο κινητό του χρήστη. Τέλος, η τρίτη υπηρεσία είναι η **Υπηρεσία Διαχείρισης Στόχων Χρήστη**. Όπως υποδεικνύει και το όνομα, διαχειρίζεται τους στόχους για κάθε δείκτη ευημερίας του ατόμου. Πρόκειται για τιμές-στόχους που έχει θέσει ο ίδιος ο χρήστης σε σχετική οθόνη και που του παρουσιάζονται μαζί με τις καταγεγραμμένες τιμές για να έχει την δυνατότητα σύγκρισης.

Στο ανώτερο επίπεδο βρίσκεται το **Γραφικό Περιβάλλον Χρήστη**, δηλαδή κάθε υπηρεσία που αναλαμβάνει την υλοποίηση της διεπαφής χρήστης. Πρόκειται ουσιαστικά, για όλες τις **οθόνες** που αλληλεπιδρούν με τον χρήστη και είτε του παρουσιάζουν κάποια αποτελέσματα σαν έξοδο, είτε δέχονται από αυτόν αιτήματα για να γίνει κάποια συγκεκριμένη λειτουργία.

Φυσικά, κάθετα σε όλα τα επίπεδα βρίσκεται η **Υπηρεσία Ασφάλειας** της εφαρμογής, που αναλαμβάνει και εδώ την προστασία του ευαίσθητου περιεχομένου της πληροφορίας. Υλοποίει βασικές τεχνικές κρυπτογραφίας για την επικοινωνία με τον εξυπηρετητή, ενώ επιπλέον υλοποιεί και την αυθεντικοποίηση των χρηστών, δηλαδή την ασφαλή εγγραφή/σύνδεσή τους, μέσω της χρήσης συγκεκριμένων λειτουργιών που παρέχει ο **Εξυπηρετητής Αυθεντικοποίησης** που χρησιμοποιήθηκε (Firebase).

3.2.4 Γεννήτρια δεδομένων

Όπως αναφέρθηκε και προηγουμένως, ο ρόλος της γεννήτριας δεδομένων είναι να δημιουργήσει τα **προσομοιωτικά δεδομένα** που θα τροφοδοτήσουν τον εξυπηρετητή του συστήματος μας. Η υλοποίηση αυτού του συστατικού στοιχείου έγινε μέσω **ανάπτυξης ενός εξυπηρετητή**, ο οποίος αναλαμβάνει τον ρόλο όλων των πηγών δεδομένων που καταγράφουν δείκτες ευημερίας του ατόμου.

Για να προσομοιωθεί πιστά η πληροφορία, επιλέξαμε πηγές δεδομένων που παρέχουν μέσω ανοικτών διεπαφών προγραμματισμού εφαρμογών (Application Programming Interface - API) την δυνατότητα πρόσβασης στην δομή που αναπαριστούν τα δεδομένα τους. Δίνεται κατ΄ επέκταση και η δυνατότητα να ακολουθήσουμε αυτή την δομή κατά την δημιουργία των προσομοιωτικών δεδομένων.

Ακόμη, απαίτηση μας είναι να καλύψουμε όλα τα βασικά είδη πηγών δεδομένων που καταγράφουν δείκτες ευημερίας. Για αυτό το λόγο, οι πηγές που επιλέχθηκαν είναι οι εξής

ανά κατηγορία:

- **Κατηγορία ενδυτών συσκευών**

  Fitbit[1]: πρόκειται για μια ενδυτή συσκευή που μπορεί να φέρει κανείς στον καρπό του χεριού του. Μπορεί να καταγράψει δείκτες σχετικούς με την κατάσταση και την δραστηριότητα του ατόμου μέσα στη μέρα.

- **Κατηγορία εφαρμογών κινητών τηλεφώνων**

  Under Armour[2]: πρόκειται για μια εφαρμογή που είναι βασισμένη σε μια ομάδα εφαρμογών (MapMyFitness, MapMyRun, MapMyWalk κλπ), που καταγράφουν μετρικές σχετικές και πάλι με την καθημερινή δραστηριότητα του ατόμου.

- **Κατηγορία αισθητήρων περιβάλλοντα χώρου**

  Αισθητήτας πίεσης στον καναπέ (couch sensor): πρόκειται για έναν αισθητήρα που χρησιμοποιείται σε ένα έξυπνο σπίτι και καταγράφει αν ο χρήστης κάθεται στον καναπέ του σπιτιού του.

  Αισθητήτας λειτουργίας της τηλεόρασης (tv sensor): πρόκειται για έναν αισθητήρα που χρησιμοποιείται και πάλι σε ένα έξυπνο σπίτι και καταγράφει αν η τηλεόραση είναι ανοικτή ή όχι.

Σε αυτό το σημείο, καλό είναι να αναφερθούμε λίγο πιο πολύ στις παραπάνω πηγές προκειμένου να περιγραφούν συνοπτικά ποιά από τα δεδομένα που καταγράφει η κάθε μία, επιλέξαμε να προσομοιώσουμε.

Πρώτα, λοιπόν, η επιλογή του **Fitbit** έγινε γιατί είναι μια ιδιαίτερα διαδεδομένη ενδυτή συσκευή, η οποία καταγράφει πολλών ειδών μετρήσεις καλύπτοντας έτσι ένα σημαντικό εύρος των δεικτών ευημερίας. Συγκεκριμένα, έχει την δυνατότητα να καταγράφει την εκάστοτε κατάσταση του χρήστη (ύπνος, ξεκούραση, αθλητική δραστηριότητα) και ένα πλήθος μετρικών που σχετίζονται με αυτήν. Συγκεντρώνει, επίσης μετρήσεις για άλλους δείκτες υγείας του ατόμου, και ειδικότερα μετρήσεις για το βάρος, τον καρδιακό παλμό του, το φαγητό και το νερό που καταναλώνει.

Δεύτερον, η εφαρμογή **Under Armour** συγκεντρώνει από μια ομάδα άλλων εφαρμογών, πληροφορίες και πάλι για την κατάσταση του ατόμου(ύπνος, αθλητική δραστηριότητα) και τις σχετικές με αυτήν μετρικές . Ωστόσο, από το API της εφαρμογής δεν είδαμε να καταγράφει την κατάσταση που ο χρήστης κάθεται. Επιπλέον, από πιο γενικούς δείκτες του σώματος, μετράει τον καρδιακό παλμό.

Τέλος, οι **αισθητήρες** στο περιβάλλοντα χώρο του χρήστη καταγράφουν, όπως προαναφέρθηκε, το αν ο χρήστης κάθεται στον καναπέ και αν η τηλεόραση του είναι ανοικτή. Πρόκειται, ουσιαστικά για πληροφορία που δεν έχει άμεση σχέση με δεδομένα υγείας του ατόμου, αλλά συνδυαζόμενη με την παραπάνω πληροφορία μπορεί να δώσει, στα πλαίσια ανάπτυξης ενός Διαδικτύου των Πραγμάτων, μια πιο ακριβή εικόνα της κατάστασης του.

[1]https://www.fitbit.com/eu/home

[2]https://developer.underarmour.com/

3.3 Τεχνολογίες

Ολοκληρώνοντας το κομμάτι της ανάλυσης, είναι σημαντικό να γίνει αναφορά στις τεχνολογίες που χρησιμοποιήθηκαν για να υλοποιηθεί τόσο το κομμάτι του εξυπηρετητή μας όσο και της εφαρμογής. Στη συνέχεια, λοιπόν, αφιερώνεται σε κάθε τεχνολογία μια υποενότητα, προκειμένου να δούμε πώς αξιοποιήθηκε η καθεμία στα επόμενα κεφάλαια.

3.3.1 Android

Ξεκινάμε, επομένως από το λειτουργικό σύστημα Android[10], πάνω στο οποίο αναπτύχθηκε η εφαρμογή μας. Είναι ουσιαστικά λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας το οποίο τρέχει τον πυρήνα του λειτουργικού Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την Open Handset Alliance. Πρόκειται με άλλα λόγια για μια πλατφόρμα ανοικτού κώδικα με εξαίρεση κάποια ιδιόκτητα μέρη, μέσω της οποίας δίνεται η δυνατότητα στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας την συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων απο την Google.

Η πλατφόρμα Android αποτελείται από διάφορα βασικά συστατικά και γενικότερα από συγκεκριμένα επίπεδα αρχιτεκτονικής, τα οποία πρώτον φαίνονται στο Σχήμα 3.2[10] και δεύτερον περιγράφονται συνοπτικά παρακάτω.

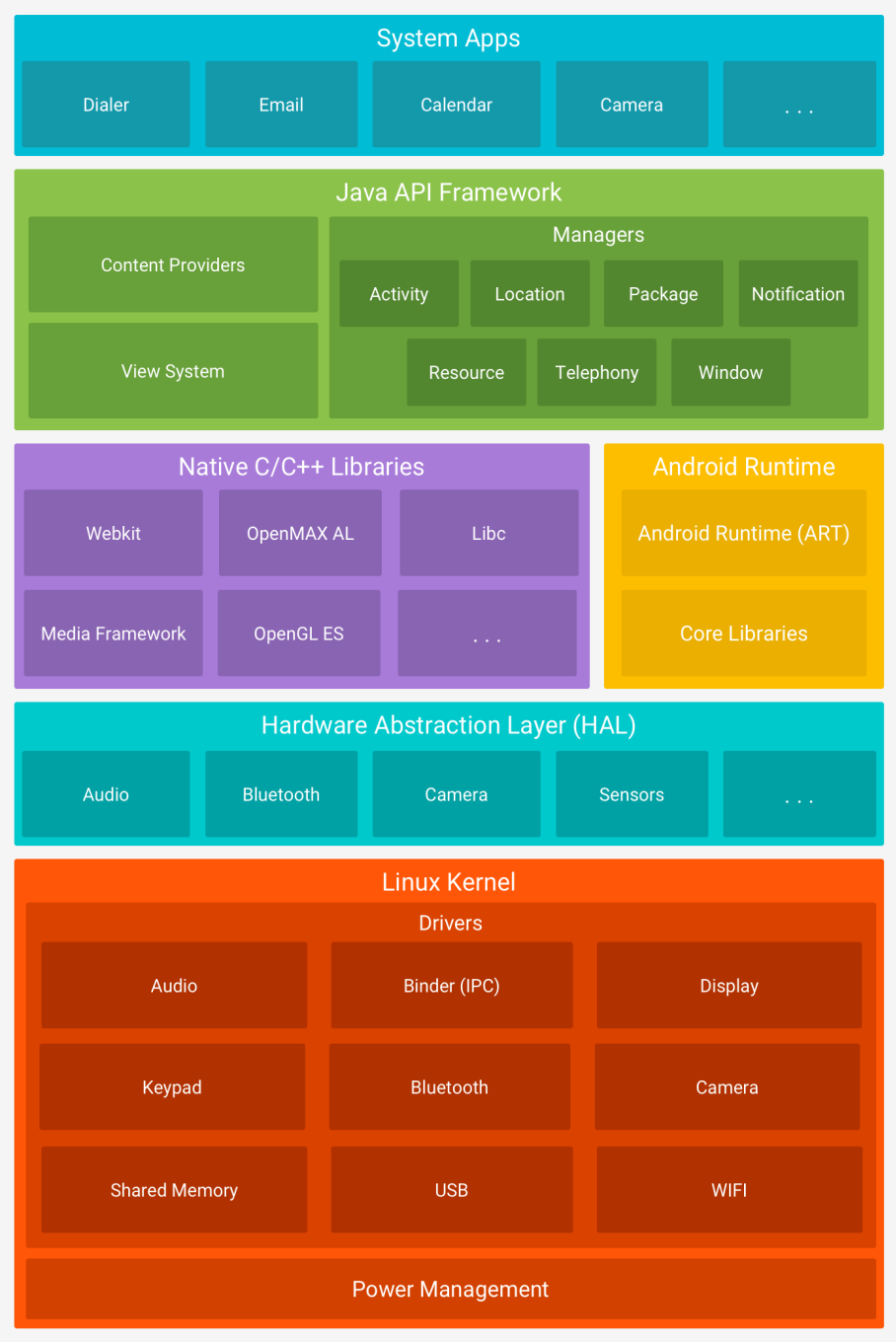
- Ο πυρήνας Linux:

  Τα θεμέλια της πλατφόρμας Android είναι ο πυρήνας των Linux(Linux Kernel). Για παράδειγμα, το συστατικό Android Runtime(ART) της πλατφόρμας χρησιμοποιεί βασικές λειτουργίες των Linux όπως είναι οι σχετικές λειτουργίες των νημάτων (threading) ή η διαχείριση μνήμης χαμηλού επιπέδου (low-level memory management).

  Επιπλέον, μέσω αυτού του πυρήνα (Linux) του λειτουργικού γίνεται χρήση βασικών χαρακτηριστικών ασφαλείας (key security features) , ενώ επίσης δίνεται η δυνατότητα στους κατασκευαστές συσκευών να αναπτύξουν οδηγούς υλικού (hardware drivers) χρησιμοποιώντας έναν ήδη γνωστό πυρήνα.

- Hardware Abstraction Layer(HAL)

  Πρόκειται για ένα συνδετικό επίπεδο στην αρχιτεκτονική της πλατφόρμας, το οποίο παρέχει συγκεκριμένες διεπαφές ώστε να επιτευχθεί η χρήση του υλικού των συσκευών (device hardware capabilities) από το ανώτερο στρώμα της αρχιτεκτονικής Java API Framework. Με άλλα λόγια, αποτελείται από διάφορες βιβλιοθήκες (library modules) όπου η καθεμία υλοποιεί τη διεπαφή για κάποιο συγκεκριμένο στοιχείο του υλικού του συστήματος (hardware component), όπως είναι η φωτογραφική μηχανή (camera) ή τα bluetooth. Έτσι, όταν γίνεται κλήση για πρόσβαση σε μια συσκευή του υλικού, η πλατφόρμα φορτώνει την κατάλληλη βιβλιοθήκη για να εξυπηρετήσει το αντίστοιχο αίτημα που προήλθε από ανώτερο στρώμα της αρχιτεκτονικής.



Σχήμα 3.2: Αρχιτεκτονική πλατφόρμας Android

- Android Runtime

  Κάθε εφαρμογή εκτελείται μέσω της αντίστοιχης διεργασίας της (process) και με το αντίστοιχο 'στιγμιότυπο' (instance) του Android Runtime(ART). Ειδικότερα, αυτό το στοιχείο της πλατφόρμας χρησιμοποιείται για την εκτέλεση εικονικών μηχανημάτων σε συσκευές χαμηλών δυνατοτήτων σε μνήμη.

  Χαρακτηριστικά του συστατικού ART είναι πρώτον η μεταγλώττιση κώδικα πριν την ώρα του ή ακριβώς τη στιγμή που πρέπει (ahead-of-time and just-in-time compilation).

Δεύτερον, η βελτιστοποιημένη ΄συλλογή σκουπιδιών΄ δηλαδή μνήμης που δεσμεύεται χωρίς να χρησιμοποιείται (garbage collection) και τρίτον μια βελτιωμένη υποστήριξη αποσφαλμάτωσης (debugging support).

Τέλος, η πλατφόρμα περιλαμβάνει και μια σειρά από βιβλιοθήκες που παρέχουν το μεγαλύτερο κομμάτι των λειτουργιών της γλώσσας προγραμματισμού Java.

- Γνήσιες C/C++ βιβλιοθήκες (Native C/C++ Libraries)

  Πολλά συστατικά του συστήματος Android, καθώς και αρκετές υπηρεσίες του, αναπτύχθηκαν από κώδικα που απαιτεί την χρήση τέτοιων βιβλιοθηκών γραμμένων σε C/C++. Η πλατφορμα, επομένως, παρέχει μια σειρά διεπαφών (Java Framework API) σε Java για να δώσει τη δυνατότητα χρήσης αυτών των βιβλιοθηκών από τις εφαρμογές.

- Java API Framework (Σκελετός διεπαφών σε Java)

  Όλη η λειτουργικότητα του συστήματος Android παρέχεται μέσω διεπαφών (APIs) γραμμένων σε γλώσσα Java. Μέσω αυτών των διεπαφών, μπορεί κανείς να έχει στη διάθεση του ό,τι χρειάζεται για να αναπτύξει την εφαρμογή που επιθυμεί. Για παράδειγμα, υπηρεσίες που παρέχει η πλατφόρμα για τη διευκόλυνση του χρήστη είναι οι εξής με την αγγλική τους ονομασία: View System, Resource Manager, Notification Manager, Activity Manager, Content Providers.

- Εφαρμογές συστήματος (System Apps)

  Η πλατφόρμα Android διαθέτει επιπλεόν κάποιες ενσωματωμένες εφαρμογές που μπορεί να χρησιμοποιήσει ο χρήστης εφόσον το επιθυμεί. Ακόμη, μπορεί να χρησιμοποιηθούν και από τους κατασκευαστές νέων εφαρμογών που θέλουν να αξιοποιήσουν κάποια συγκεκριμένη λειτουργία μιας ενσωματωμένης εφαρμογής. Τέτοια παραδείγματα είναι οι εφαρμογές για λήψη μηνυμάτων, κλήσεων, διαχείρισης ηλεκτρονικού ταχυδρομείου, επαφών καθώς και εφαρμογές πλοήγησης στο διαδίκτυο.

Τέλος, έχοντας περιγράψει την αρχιτεκτονική της πλατφόρμας, αξίζει να αναφερθεί και το εργαλείο **Android Studio**, το οποίο χρησιμοποιήθηκε **για να υλοποιηθεί η εφαρμογή**. Πρόκειται για ένα ολοκληρωμένο προγραμματιστικό περιβάλλον για ανάπτυξη εφαρμογών στην πλατφόρμα Android.

3.3.2 Node.js

Πρόκειται για μια ανοικτού κώδικα πλατφόρμα ανάπτυξης λογισμικού χτισμένη σε περιβάλλον Javascript. Στόχος του Node.js[11] είναι να παρέχει έναν εύκολο τρόπο δημιουργίας κλιμακωτών διαδικτυακών εφαρμογών. Σε αντίθεση, μάλιστα, με τα περισσότερα σύγχρονα περιβάλλοντα ανάπτυξης εφαρμογών δικτύων μία διεργασία node δεν στηρίζεται στην πολυνηματικότητα αλλά σε ένα μοντέλο ασύγχρονης επικοινωνίας εισόδου/εξόδου. Με άλλα λόγια, χαρακτηριστικό της πλατφόρμας είναι η ασύγχρονη επικοινωνία μεταξύ των υπολογιστικών πόρων.

Η πλατφόρμα αυτή χρησιμοποιήθηκε για την **ανάπτυξη** τόσο **του εξυπηρετητή που παράγει τα δεδομένα υγείας**, όσο **και του εξυπηρετητή της εφαρμογής** που λαμβάνει και διαχειρίζεται τα δεδομένα.

3.3.3 Java

Όσον αφορά την Java[12], πρόκειται για μια αντικειμενοστρεφή γλώσσα προγραμματισμού που σχεδιάστηκε από την εταιρεία πληροφορικής Sun Microsystems.

Το πιο βασικό πλεονέκτημα της γλώσσας αυτής έναντι άλλων, είναι η ανεξαρτησία του λειτουργικού συστήματος και πλατφόρμας. Με άλλα λόγια, τα προγράμματα που είναι γραμμένα σε Java τρέχουν ακριβώς το ίδιο σε κάθε λειτουργικό σύστημα (Windows, Linux, Unix, Macitosh κλπ), χωρίς να απαιτείται εκ νέου μεταγλώττιση ή αλλαγή του πηγαίου κώδικα. Ο τρόπος, με τον οποίο επιτεύχθηκε αυτό είναι με την χρήση της εικονικής μηχανής της java(Virtual Machine). Ειδικότερα, όταν γραφεί κάποιο πρόγραμμα σε Java μεταγλωττίζεται μέσω του μεταγλωττιστή javac, ο οποίος παράγει έναν αριθμό από αρχεία .class. Στη συνέχεια, η εικονική μηχανή της Java μεταφράζει αυτά τα αρχεία στη γλώσσα μηχανής που υποστηρίζεται από το εκάστοτε λειτουργικό σύστημα και επεξεργαστή.

Χρησιμοποιήθηκε στο σύστημα μας **προκειμένου να υλοποιηθεί ένα κομμάτι του εξυπηρετητή της εφαρμογής**. Συγκεκριμένα, μέσω προγράμματος αυτής της γλώσσας, επιτυγχάνεται η ενημέρωση της γραφικής βάσης δεδομένων και η υλοποίηση της οντολογίας σε αυτή. Επιπλέον, υπάρχει κι ακόμη ένα Java πρόγραμμα που επικοινωνεί με μηνύματα με το πρώτο και αναλαμβάνει την εξαγωγή προειδοποιήσεων και περιλήψεων της δραστηριότητας του χρήστη.

3.3.4 Neo4j

Μια επιπλέον χρήσιμη τεχνολογία είναι το Neo4j[13] που είναι υπεύθυνο για την διαχείριση της γραφικής βάσης δεδομένων. Αρχικά, όταν αναφερόμαστε σε γραφική βάση δεδομένων, πρόκειται για μια βάση που χρησιμοποιεί την δομή ενός γράφου για να αναπαραστήσει τα δεδομένα που επιθυμεί να αποθηκεύσει ο χρήστης της. Για την αποθήκευση, κάνει χρήση κόμβων (nodes), ακμών (edges) και ιδιοτήτων (properties,attributes) για να εκφράσει τα δεδομένα και τη μεταξύ τους συσχέτιση.

Εμβαθύνοντας τώρα στο σύστημα διαχείρισης Neo4j, πρόκειται για ένα εργαλείο γραμμένο σε γλώσσα προγραμματισμού Java που είναι προσβάσιμο από λογισμικό γραμμένο σε άλλες γλώσσες μέσω της χρήσης της γλώσσας ερωτημάτων (Query Language) Cypher. Μέσω της Cypher[14] μπορούν να εκφραστούν πολύπλοκα ερωτήματα προς την βάση με έναν εύκολο τρόπο. Αυτό συμβαίνει, διότι είναι μια δηλωτική γλώσσα ερωτημάτων γράφου (declarative graph query language) που υποστηρίζει εκφραστικότητα, αποτελεσματική αναζήτηση και ενημέρωση ενός γράφου.

Επιπροσθέτως, αξίζει να αναφερθεί κανείς λίγο παραπάνω και στη δομή που έχουν τα δεδομένα σε μια τέτοια βάση. Όπως είπαμε, κάθετι αποθηκεύεται είτε σαν κόμβος (node) , είτε σαν ακμή (edge) , είτε σαν ιδιότητα (attribute) . Κάθε κόμβος ή ακμή μπορεί να έχει πολλές ιδιότητες. Επιπλέον, κόμβοι και ακμές μπορούν να φέρουν και ετικέτες (labels), οι οποίες μπορούν να χρησιμοποιηθούν και κατά τις αναζητήσεις στη βάση.

Στην βάση αυτή αποθηκεύτηκε όλη η **τελική πληροφορία** που σχετίζεται με τον χρήστη του συστήματός μας, εκφρασμένη στην **οντολογική μορφή** που αναπτύχθηκε.

3.3.5 MongoDB

Πρόκειται για μια ελεύθερη και ανοιχτού κώδικα πλατφόρμα διαχείρισης μη σχεσιακών βάσεων δεδομένων. Σημειώνεται επίσης ότι είναι ανεξάρτητη λειτουργικού συστήματος.

Όπως αναφέρθηκε, η MongoDB[15] ασχολείται με τη διαχείριση μη σχεσιακών βάσεων το οποίο σημαίνει ότι ξεφεύγει από την κλασσική δομή των σχεσιακών βάσεων με αποτέλεσμα να προσφέρει ευελιξία. Αποθηκεύει τα δεδομένα σε μορφή εγγράφων με μορφή JSON ή για να είμαστε πιο ακριβείς BSON. Αυτό το χαρακτηριστικό δίνει την δυνατότητα στα πεδία να διαφέρουν από έγγραφο σε έγγραφο, ενώ είναι εφικτή η εύκολη αλλαγή της δομής των δεδομένων.

Για την **υλοποίηση**, επομένως, **των μη σχεσιακών βάσεων** του συστήματός μας, χρησιμοποιήθηκε η υπηρεσία **mLab**[16] που φιλοξενεί βάσεις δεδομένων MongoDB. Συγκεκριμένα, πρόκειται για μια υπηρεσία υπολογιστικού νέφους (cloud) , η οποία παρέχεται σαν υπηρεσία στους πελάτες που επιθυμούν να υλοποιήσουν την δική τους μη σχεσιακή βάση στο διαδίκτυο και όχι τοπικά στο μηχάνημά τους.

3.3.6 Apache Kafka

Πρόκειται για μια ακόμη πλατφόρμα ανοικτού κώδικα που ασχολείται με τη διαχείριση και επεξεργασία μια ροής μηνυμάτων ή συμβάντων.

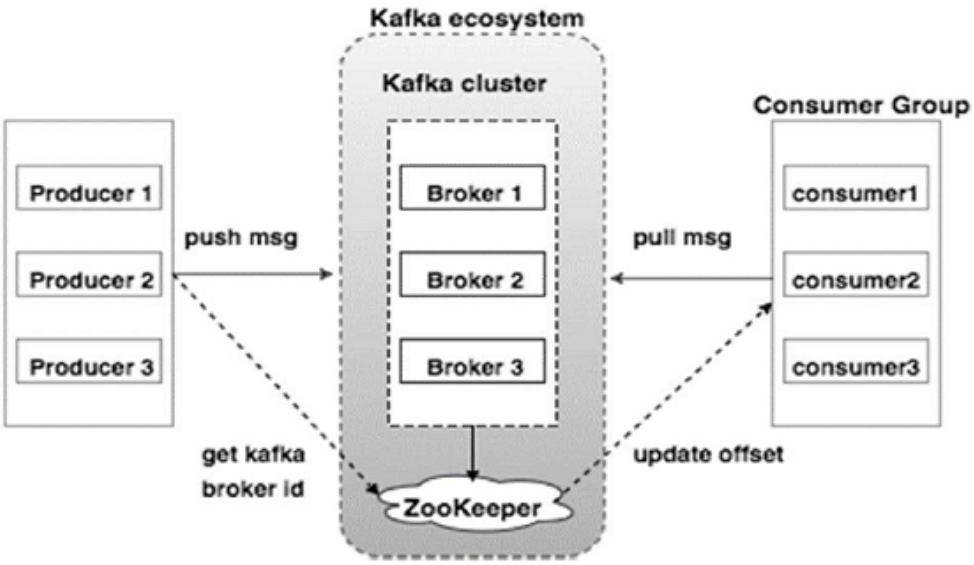
Ειδικότερα, η πλατφόρμα Apache Kafka[17] παρέχει ένα ενιαίο σύστημα διαχείρισης της τροφοδοσίας δεδομένων σε πραγματικό χρόνο και είναι γραμμένη σε Scala και Java. Πετυχαίνει υψηλή απόδοση και χαμηλή καθυστέρηση στη διαβίβαση της πληροφορίας, κάτι που την κάθιστά ως ένα από τα πιο χρήσιμα εργαλεία για συστήματα που διαχειρίζονται μεγάλο όγκο μηνυμάτων και συμβάντων ή που κλιμακώνονται γρήγορα.

Σε αυτό το σημείο, είναι σημαντικό να περιγραφεί γενικά και η αρχιτεκτονική της πλατφόρμας για να γίνει εμφανής και ο τρόπος με τον οποίο παράγεται και διαχειρίζεται η πληροφορία. Αρχικά η πλατφόρμα αποθηκεύει τα μυνήματα που δέχεται από συγκεκριμένες διεργασίες που καλούνται 'παραγωγοί' (producers). Αυτά κρατώνται μέσα στη συστάδα (cluster) του συστήματος Kafka σε συγκεκριμένες θέσεις ανάλογα με το θέμα (topic) και την διαμέριση (partition), που μπορεί να επιλέξει ο παραγωγός κατά την αποστολή του μυνήματος. Σε επόμενο στάδιο, διεργασίες που ονομάζονται 'καταναλωτές' (consumers) αναζητούν αυτά τα μηνύματα από τις διαμερίσεις του συστήματος. Συγκεκριμένα, κάθε 'καταναλωτής' μπορεί να εγγραφεί σε συγκεκριμένα θέματα (topics) και να λάβει μηνύματα μόνο από αυτά. Τέλος, το σύστημα Kafka τρέχει σε μια συστάδα (Kafka cluster) με έναν ή περισσότερους εξυπηρετητές με στόχο την κατανομή των διαμερίσεων (partitions) μεταξύ των στοιχείων-κόμβων της συστάδας (cluster nodes).

Εμβαθύνοντας στην αρχιτεκτονική οι κύριες έννοιες του συστήματος είναι τα θέματα (topics) που αναφέρθηκαν παραπάνω, τα έγγραφα-καταγραφές (records) και οι 'διαμεσολαβητές' (brokers). Τα θέματα αποτελούνται από ροές εγγράφων που φέρουν

διαφορετική πληροφορία για το θέμα, ενώ οι 'διαμεσολαβητές' (brokers) είναι υπεύθυνη για τη σωστή διαχείριση των μηνυμάτων και για την αντιγραφή τους.

Επιπλέον, η πρόσβαση και διαχείριση της πλατφόρμας από τον χρήστη γίνεται μέσω συγκεκριμένων διεπαφών (API). Τέτοιες είναι η διεπαφή του 'παραγωγού' (Producer API), του καταναλωτή (Consumer API), της ροής δεδομένων (Stream API) για τη μετατροπή τους και η διεπαφή για σύνδεση των θεμάτων σε υπάρχουσες εφαρμογές (Connector API). Στο Σχήμα 3.3, φαίνεται μια αφαιρετική αναπαράσταση της αρχιτεκτονικής της πλατφόρμας Kafka.



Σχήμα 3.3: Συνοπτική αρχιτεκτονική πλατφόρμας Apache Kafka

Στο σύστημα μας, χρησιμοποιείται αρχικά για την **παραγωγή μηνυμάτων που σχετίζονται με τα δεδομένα υγείας** που συλλέγει ο εξυπηρετητής της εφαρμογής και εν συνεχεία με την κατανάλωσή τους από το κομμάτι του εξυπηρετητή που είναι γραμμένο σε Java και αποθηκεύει αυτά τα δεδομένα στο γράφο μας με βάση την οντολογίκη μορφή τους. Φυσικά, η πληροφορία που διαθέτουμε για έναν χρήστη δεν είναι και τόσο ογκώδης, ωστόσο πρόκειται αναμφισβήτητα για ένα σύστημα που μπορεί να επεκταθεί πολύ γρήγορα. Γι' αυτό το λόγο επιλέχθηκε η παρούσα τεχνολογία.

3.3.7 RabbitMQ

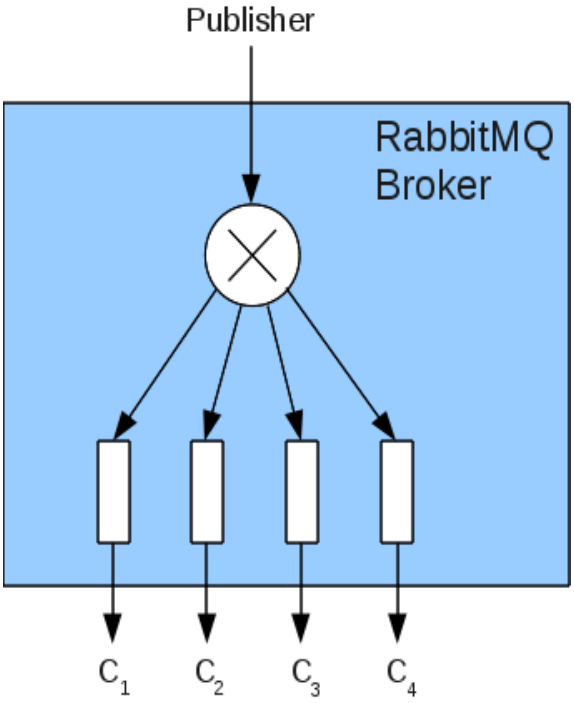
Το RabbitMQ[18] είναι ακόμη ένα λογισμικό που χρησιμοποιείται για την διαχείριση μηνυμάτων μεταξύ διαφορετικών συστατικών στοιχείων ενός συστήματος.

Αρχικά, πρέπει να αναφερθεί ότι πρόκειται για ένα ανοικτού κώδικα σύστημα που έχει το ρόλο του 'διαμεσολαβητή' (broker) μηνυμάτων. Υλοποιεί το πρωτόκολλο προηγμένων ουρών μηνυμάτων (Advanced Message Queueing Protocol - AMQP) ενώ ο εξυπηρετητής RabbitMQ είναι γραμμένος σε γλώσσα προγραμματισμού Erlang και υλοποιημένος πάνω στη πλατφόρμα Open Telecom Platform. Φυσικά, υπάρχουν βιβλιοθήκες για τους χρήστες του

λογισμικού που είναι διαθέσιμες σε όλες τις βασικές και γνωστές γλώσσες προγραμματισμού του σήμερα.

Εμβαθύνοντας λίγο στο πρωτόκολλο (AMQP) που υλοποιείται μέσω του RabbitMQ αξίζει να σημειωθεί ότι πρόκειται για ένα πρωτόκολλο σε επίπεδο εφαρμογής για την υλοποίηση του στρώματος λήψης και αποστολής μηνυμάτων μεταξύ στοιχείων ενός κατανεμημένου συστήματος (message-oriented middleware). Κύρια χαρακτηριστικά του πρωτοκόλλου είναι ο προσανατολισμός των μηνυμάτων (message orientation), οι ουρές μηνυμάτων (queueing), η δρομολόγηση τους και τέλος η αξιοπιστία και η ασφάλεια στη διαχείρισή τους.

Η μέθοδος που χρησιμοποιείται απο το σύστημά μας για τη δρομολόγηση των μηνυμάτων είναι αυτή της 'δημοσιέυσης/εγγραφής' (Publish/Subscribe). Ειδικότερα, γίνεται 'δημοσίευση' μηνυμάτων σε δίαφορα θέματα του διαμεσολαβητή του RabbitMQ. Στη συνέχεια, υπάρχει μια σειρά από καταναλωτές (Consumers), οι οποίοι είναι εγγεγραμμένοι σε αντίστοιχα θέματα και καταναλώνουν τα μηνύματα που τους ενδιαφέρουν προκειμένου να τα επεξεργαστούν με το τρόπο που πρέπει κάθε φορά. Μια πολύ συνοπτική αρχιτεκτονική ενός τέτοιου συστήματος φαίνεται στο Σχήμα 3.4.



Σχήμα 3.4: Συνοπτική αρχιτεκτονική λογισμικού RabbitMQ

Συγκεκριμένα, στο σύστημά μας η τεχνολογία αξιοποιήθηκε για την **επίτευξη της επικοινωνίας** μεταξύ των δύο προγραμμάτων του εξυπηρετητή σε Java. Το ένα παράγει μηνύματα σε συγκεκριμένα θέματα (topics) για να δηλώσει την εμφάνιση κάποιων **συμβάντων-γεγονότων** σχετικών με τη δραστηριότητα του χρήστη, ενώ το άλλο πρόγραμμα τα λαμβάνει για την εξαγωγή συμπερασμάτων.

3.3.8 JSON / JSON-LD

Το JSON (Javascript Object Notation[19]) είναι ένας τρόπος ανταλλαγής δεδομένων που στηρίζεται στην εύκολη ανάγνωση από τον άνθρωπο (human-readable text). Αποτελείται από ζεύγη ονόματος-τιμής (attribute-value pairs) και από πίνακες δεδομένων(ή αλλιώς ταξινομημένη λίστα).

Επιπλέον, πρόκειται για μια αναπράσταση δεδομένων, ανεξάρτητη της γλώσσας που χρησιμοποιείται. Σημειώνεται ότι αρχικά προήλθε από τη γλώσσα Javascript, αλλά πλέον πολλές γλώσσες προγραμματισμού υποστηρίζουν έτοιμο κώδικα για την παραγωγή και την ανάγνωση δεδομένων σε αυτή τη μορφή. Μάλιστα, είναι σημαντικό ότι μέσω JSON τα δεδομένα μπορούν να πάρουν μια μορφή που είναι αναγνώσιμη εύκολα από πολλές διαφορετικές γλώσσες και αυτό είναι ένα χαρακτηριστικό που μπορεί να λύσει πολλά ζητήματα σχετικά με την μεταφορά των δεδομένων μεταξύ υπηρεσιών και γενικότερα συστημάτων.

Όσον αφορά τώρα το JSON-LD[20] επισημαίνεται ότι είναι μια μέθοδος αναπαράστασης-κωδικοποίησης (encoding) διασυνδεδεμένων δεδομένων (linked data) σε μορφή JSON.

Πρέπει ακόμη να σημειωθεί ότι με τον όρο **διασυνδεδεμένα δεδομένα** (linked data) αναφερόμαστε σε μια μέθοδο δημοσιοποίησης δομημένων δεδομένων ώστε να είναι αλληλένδετα και να γίνουν πιο χρήσιμα. Πιο συγκεκριμένα, στηριζόμενοι σε γνωστές τεχνολογίες του Ιστού, ανταλλάσσονται πληροφορίες με τρόπο που μπορούν να διαβαστούν αυτόματα από υπολογιστές. Αυτό επιτρέπει δεδομένα από διαφορετικές πηγές να συνδέονται και να μπορούν να αναζητηθούν με ευκολία.

Εμβαθύνοντας, σε αυτό το σημείο, στο τρόπο με τον οποίο επιτυγχάνονται τα παραπάνω μέσω JSON αξίζει να αναφερθεί ότι υπάρχει η έννοια του context. Με τη χρήση της στη δομή του JSON παρέχεται η δυνατότητα σύνδεσης του εκάστοτε JSON με το αντίστοιχο μοντέλο περιγραφής πόρων (RDF). Με άλλα λόγια, ο όρος context συνδέει τις ιδιότητες των αντικειμένων σε ένα αρχείο JSON με τις αντίστοιχες έννοιες σε μια οντολογία. Επιπλέον, υπάρχουν οι έννοιες type και id για να καθορίζονται αντίστοιχα ο τύπος του αντικειμένου που αναφερόμστε στη μορφή JSON, καθώς επίσης και το αναγνωριστικό (Internationalized Resource Identifier - IRI) του συγκεκριμένου αντικειμένου.

Στο σύστημά μας χρησιμοποιείται εκτενώς η μορφή JSON για την **αναπαράσταση κάθε είδους συλλεχθέντων δεδομένων**, ενώ συγκεκριμένα η μορφή JSON-LD χρησιμοποιείται για την **αναπαράσταση των σημασιολογικών δεδομένων** που προκύπτουν.

3.3.9 OWL

Πρόκειται για γλώσσα οντολογιών ιστού (Web Ontology Language - OWL[21]). Είναι μια οικογένεια γλωσσών αναπαράστασης γνώσης για τη συγγραφή οντολογιών.

Ειδικότερα, με τον όρο **οντολογία** αναφέρεται κανείς σε έναν επίσημο τρόπο περιγραφής ταξινομημένων δικτύων ταξινόμησης (classification network), δηλαδή με άλλα λόγια η οντολογία ορίζει μια δομή για την γνώση που σχετίζεται με ένα συγκεκριμένο τομέα. Επιπλέον, μια τέτοια γλώσσα είναι ιδιαίτερα ευέλικτη αφού σκοπεύει στην αναπαράσταση δεδομένων που προέρχονται από πολλές και διαφορετικές πηγές δεδομένων του διαδικτύου.

Χαρακτηριστικό των γλωσσών οντολογικού ιστού είναι ακόμη και η σημασιολογία με επίσημη μορφή (formal semantics). Οι γλώσσες αυτές αναπτύσσονται πάνω στη βάση της γλώσσας σήμανσης XML(eXtensible Markup Language) του οργανισμού W3C(World Wide Web Consortium) για αντικείμενα, που καλείται RDF(Resource Description Framework).

Η χρήση της γλώσσας OWL στο σύστημά μας είναι προφανής προκειμένου να αναπτυχθεί το κοινό λεξιλόγιο που θα ακολουθεί κάθε είδους δεδομένο υγείας του ατόμου. Τέλος, για την ανάπτυξη της οντολογίας, έγινε χρήση του λογισμικού **Protege**[3]. Είναι λογισμικό ανοικτού κώδικα για την σύνταξη οντολογιών ενώ επίσης αποτελεί και σύστημα διαχείρισης της γνώσης (knowledge management). Μέσω αυτού του εργαλείου επομένως, **υλοποιήθηκε η οντολογία** μας και η εξαγωγή της σαν αρχείο τύπου rdf για μετέπειτα χρήση.

[3]https://protege.stanford.edu

Κεφάλαιο 4

Σχεδίαση συστήματος

Το Κεφάλαιο 4 αφιερώνεται σε πρώτη φάση στη σχεδίαση των δεδομένων και στη συνέχεια στη σχεδίαση του εξυπηρετητή και της εφαρμογής. Αναλύεται, αντίστοιχα, η οντολογία που αναπτύχθηκε και που αναφέραμε και στα προηγούμενα κεφάλαια, ενώ επίσης περιγράφονται πιο συγκεκριμένα οι λειτουργίες του συστήματος, μέσω ειδικών σχημάτων της αρχιτεκτονικής του εξυπηρετητή και της εφαρμογής.

4.1 Δεδομένα

Από την αρχή αυτής της διπλωματικής, αναφέρθηκε η ανάγκη για προτυποποίηση των δεικτών ευημερίας του ατόμου μέσω ενός κοινού λεξιλογίου που θα μπορούσε να ακολουθήθει από κάθε είδους πληροφορία, σχετική με την υγεία και την φυσική κατάσταση του ατόμου. Θα γίνει εμφανής, λοιπόν, σε αυτό το σημείο ο τρόπος με τον οποίο αναπτύχθηκε η οντολογία. Ακόμη, θα περιγραφεί η χρησιμότητα της μορφής JSON-LD(linked data) και του γράφου σε Neo4j για την αναπαράσταση των σημασιολογικών δεδομένων.

4.1.1 Ανάπτυξη οντολογίας με χρήση τεχνολογιών σημασιολογικού ιστού

Ο σχεδιασμός ενός **κοινού λεξιλογίου** ικανού να εκφράσει κάθε είδους δεδομένο υγείας αποτελεί κύριο μέλημα για την επίτευξη των στόχων της διπλωματικής. Με την χρήση ενός τέτοιου λεξιλογίου, **ο μεγάλος όγκος της πληροφορίας θα μοντελοποιηθεί** με τον ίδιο τρόπο, καθιστώντας πολύ πιο εύκολη την διαχείρισή της. Με άλλα λόγια, θα γίνει μετασχηματισμός των ποικίλων τρόπων αναπαράστασης των δεδομένων σε μια νέα αναπαράσταση που θα εκφράζει σημασιολογικά πλέον τους δείκτες ευημερίας του ατόμου.

Για την υλοποίηση του κοινού λεξιλογίου, γίνεται χρήση της **γλώσσας οντολογιών ιστού** (Ontology Web Language - OWL). Ο λόγος που χρησιμοποιούμε την οντολογική μορφή για να εκφράσουμε το κοινό αυτό λεξιλόγιο, είναι διότι πρόκειται να αναπαραστήσουμε έννοιες που σχετίζονται με την σύγχρονη πραγματικότητα. Γενικά, η χρήση οντολογιών ενθαρρύνεται, όταν σκοπός είναι η **αναπαράσταση γνώσης που εκφράζει έναν τομέα ενδιαφέροντος στον πραγματικό κόσμο**. Στην περίπτωσή μας, λοιπόν,

ο **τομέας** που μας αφορά είναι αυτός **της υγείας και των δεικτών ευημερίας** που συλλέγονται για ένα άτομο.

Αναπτύχθηκε, επομένως μέσω της γλώσσας οντολογιών ιστού, ένα σύνολο από κλάσεις που περιγράφουν τα δεδομένα υγείας που καταγράφονται για ένα άτομο από τις διάφορες πηγές δεδομένων. Καθεμία από αυτές τις κλάσεις έχει τις δικές της ιδιότητες οι οποίες φέρουν πληροφορία που είναι αναγκαία για τον ακριβή προσδιορισμό των αντικειμένων της. Η λογική για να επιλεχθούν οι κλάσεις και οι ιδιότητες, βασίστηκε στον τρόπο καταγραφής δεικτών ευημερίας από τα περισσότερα συστήματα. Μελετήσαμε, με άλλα λόγια, τις διεπαφές προγραμματισμού εφαρμογών (APIs) που παρέχουν διαδεδομένες συσκευές και εφαρμογές και εξετάσαμε τον τρόπο με τον οποίο συλλέγουν τα δεδομένα. Έτσι, καταφέραμε να αντλήσουμε σημαντική πληροφορία για τα δεδομένα υγείας ενός ατόμου και εν τέλει να την αναπαραστήσουμε με κλάσεις και ιδιότητες μιας οντολογίας.

Συνοπτικά, λοιπόν, παρατηρήσαμε ότι **κάθε άτομο** (Person) **κάνει χρήση διαφόρων πηγών δεδομένων** (DataSource) ,**οι οποίες καταγράφουν με τη σειρά τους παρατηρήσεις** (Observation) για το άτομο αυτό. Επίσης, **κάθε παρατήρηση** (Observation) **μπορεί να σχετίζεται με μια συγκεκριμένη κατάσταση** (State) του ατόμου.

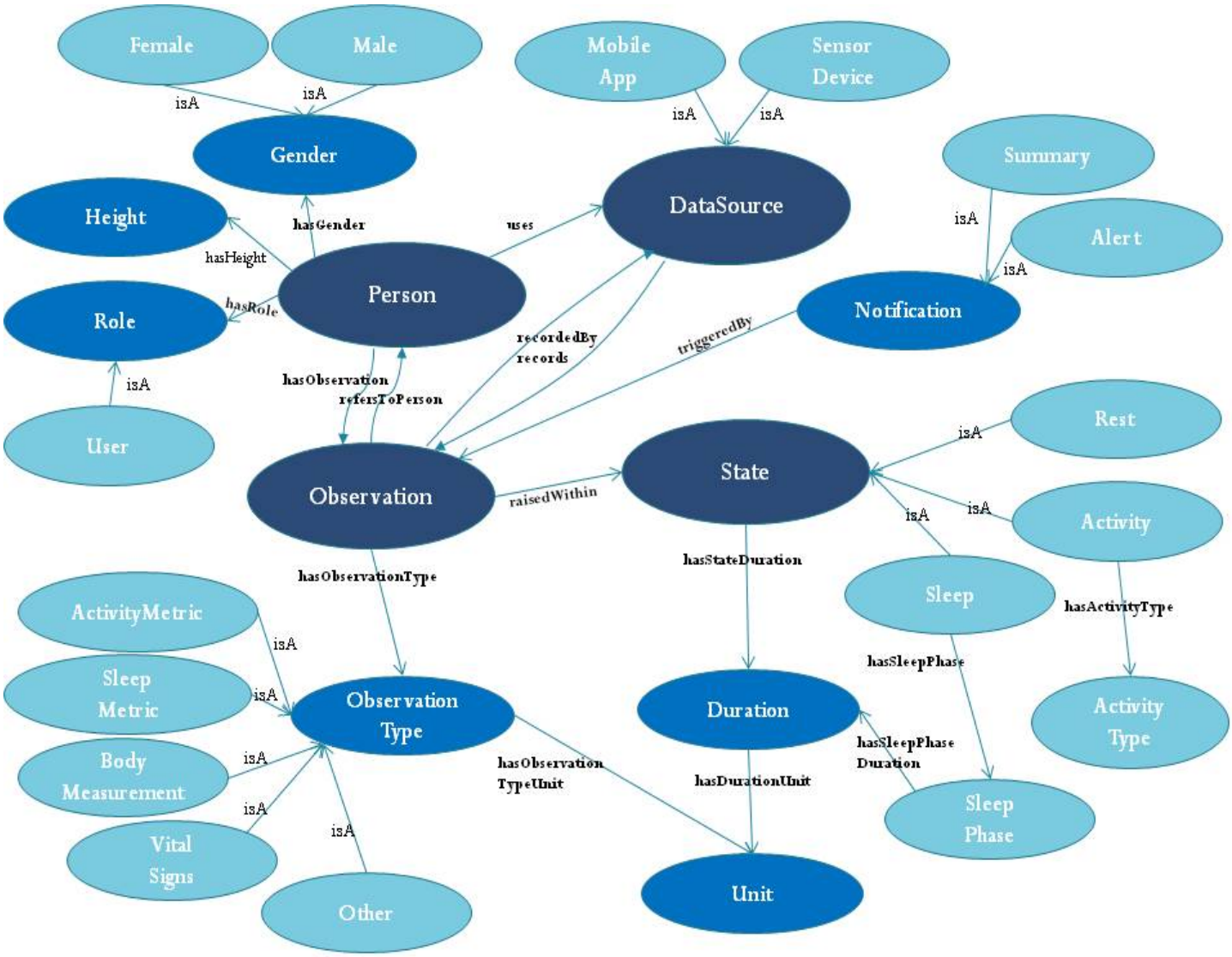
Με βάση αυτά καταλήξαμε στο τελικό σχήμα της οντολογίας (Σχήμα 4.1). Συγκεκριμένα, φαίνονται οι κλάσεις της οντολογίας και οι διασυνδέσεις μεταξύ τους. Οι ιδιότητες τύπου δεδομένων των κλάσεων δεν παρατίθενται λόγω περιορισμένου χώρου, ωστόσο θα αναφερθούμε σε αυτές στην συνέχεια που θα γίνει πιο εκτενής περιγραφή. Τέλος, σημειώνεται ότι με πιο σκούρο χρώμα φαίνονται οι κυρίαρχες κλάσεις της οντολογίας.

Έχοντας με βάση τα παραπάνω, μια γενική ιδέα του τρόπου με τον οποίο σχεδιάστηκε η οντολογία μπορούμε να δουμε μία προς μία τις κλάσεις, καθώς επίσης να αναφερθούμε και στις ιδιότητες τύπου δεδομένων (data properties) και τύπου αντικειμένων (object properties), που διαθέτουν. Σημειώνεται ότι για τον σχεδιασμό έγινε χρήση του εργαλείου Protege(σχήμα 4.2).

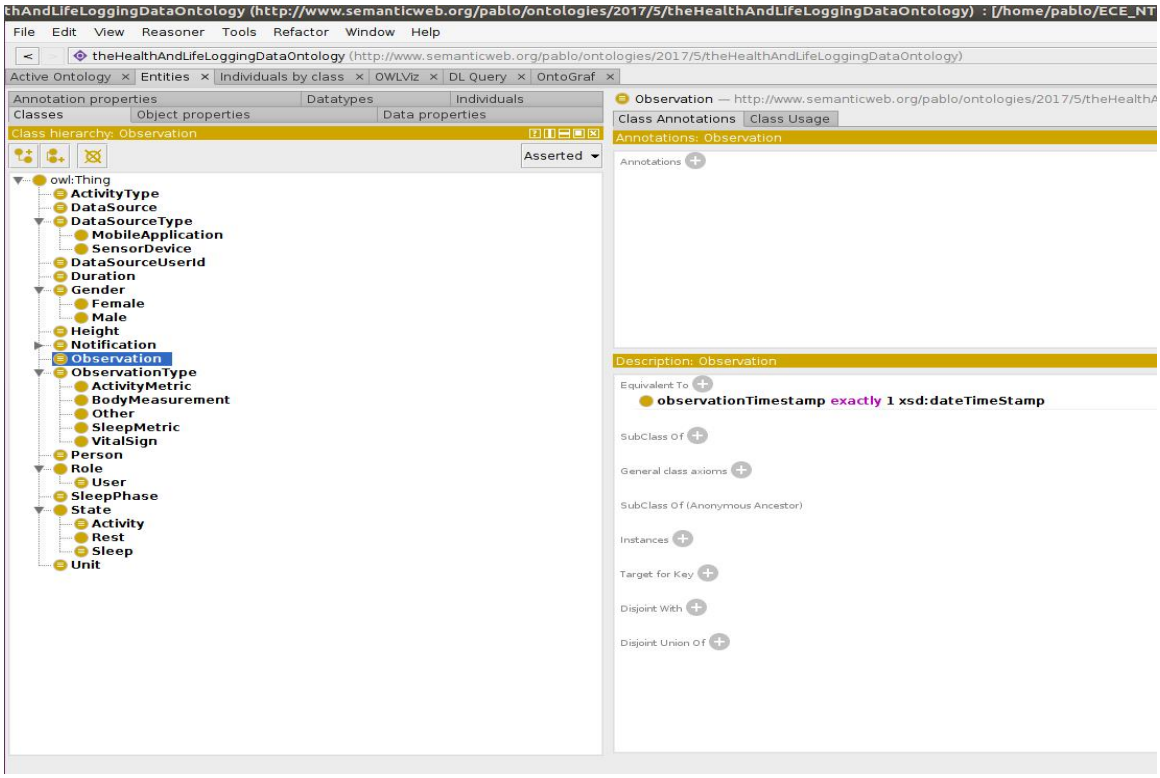
- **Person**: Η κλάση Person περιγράφει κάθε άτομο που είναι συνδεδεμένο στο σύστημα που εκφράζει η οντολογία.

  Διαθέτει, σαν ιδιότητα τύπου δεδομένων, τον αριθμό ταυτοποίησης του (personId) για να προσδιορίζεται μοναδικά στο σύστημα. Πρόκειται προφανώς για ιδιότητα με τύπο ακεραίου. Υπάρχει ακόμη ιδιότητα με τύπο ημερομηνίας (date) για την ημερομηνία γέννησης (birthDate) του ατόμου. Άλλες ιδιότητες τύπου δεδομένων της κλάσης είναι το όνομα (firstName) και το επώνυμο (lastName) του ατόμου, η ηλικία του (age), η διεύθυνση ηλεκτρονικού ταχυδρομείου (email) που διαθέτει, η διεύθυνση κατοικίας του (address), καθώς και η πόλη (addressCity) που διαμένει. Κάθε μία από αυτές τις ιδιότητες εκφράζεται με συμβολοσειρά.

  Σαν ιδιότητες τύπου αντικειμένων έχει την ιδιότητα hasGender με πολλαπλότητα 1 προς 1 που δείχνει ότι κάθε άτομο ανήκει σε ένα φύλο(κλάση Gender), την ιδιότητα hasHeight (πολλαπλότητας 1 προς 1) για το ύψος(κλάση Height) του ατόμου καθώς και την ιδιότητα hasRole με πολλαπλότητα 1 προς ν, που δείχνει ότι κάθε άτομο έχει τουλάχιστον έναν ρόλο(κλάση Role) στο σύστημα που περιγράφει η οντολογία.



Σχήμα 4.1: Κλάσεις οντολογίας



Σχήμα 4.2: Στιγμιότυπο από Protege

Πιο σημαντικές ιδιότητες τύπου αντικειμένων είναι η ιδιότητα uses που συνδέεται με την κλάση DataSource και περιγράφει ότι το άτομο κάνει χρήση διαφόρων πηγών δεδεδομένων, η ιδιότητα hasObservation για τις παρατηρήσεις που έχουν συνδεθεί με το άτομο(αντίστροφη της ιδιότητας refersToPerson) και η ιδιότητα isAt για τη κατάσταση στην οποία βρίσκεται μια δεδομένη χρονική στιγμή.

- **Gender**: Η κλάση Gender περιγράφει το φύλο του ατόμου. Δεν χρειάζεται κάποια ιδιότητα, παρά μόνο τις υποκλάσεις της, **Male** και **Female**, για τον προσδιορισμό του φύλου.

- **Height**: Η κλάση Height περιγράφει το ύψος του ατόμου μέσω αφενός της ιδιότητας δεδομένων heightValue που είναι η τιμή του ύψους και αφετέρου της ιδιότητας αντικειμένων hasHeightUnit για τον απαιτούμενο προσδιορισμό της μονάδας μέτρησης του ύψους. Η τελευταία συνδέεται με την κλάση Unit, κάτι που δεν φαίνεται στο σχήμα λόγω περιορισμένου χώρου.

- **Role**: Η κλάση Role απαιτείται για τον προσδιορισμό του ρόλου ή των ρόλων του ατόμου. Αποτελεί γενίκευση ενός συνόλου ρόλων όπως είναι ο χρήστης (**User**), ο γιατρός, ο φυσικοθεραπευτής και λοιπά.

  Σημειώνεται πως σε αυτή την εργασία, όλα τα άτομα που συνδέονται στο σύστημα που εκφράζει η οντολογία, έχουν τον ρόλο του χρήστη (User). Δεν θα μελετηθούν με άλλα λόγια άλλοι τύποι ρόλων και γιάυτό δεν παρατίθενται στο σχήμα.

- **DataSource**: Η κλάση DataSource περιγράφει κάθε πηγή δεδομένων που καταγράφει μετρήσεις για δείκτες ευημερίας.

  Οι ιδιότητες τύπου δεδομένων της κλάσης είναι ο αριθμός ταυτοποίησης (dataSourceId) της πηγής που την αντιστοιχεί μοναδικά με εναν ακέραιο αριθμό, το όνομα (dataSourceName) της πηγής που είναι μια συμβολοσείρα χαρακτήρων, καθώς και ο αριθμός ταυτοποιήσης που χρησιμοποιεί η πηγή δεδομένων για τον συγκεκριμένο χρήστη της (dataSourceUserIdValue) .

  Ιδιότητα τύπου αντικειμένων είναι η records που συνδέεται με πολλαπλότητα 1 προς ν με την κλάση Observation για την περιγραφή όλων των παρατηρήσεων που καταγράφει μια πηγή δεδομένων. Είναι, όπως θα δούμε αντίστροφη της ιδιότητας recordedBy.

- **MobileApp, SensorDevice**: Σημειώνεται σε αυτό το σημείο, ότι η κλάση DataSource αποτελεί γενίκευση των κλάσεων MobileApp και SensorDevice της οντολογίας. Αυτό σημαίνει ότι οι εφαρμογές κινητών τηλεφώνων και οι συσκευές-αισθητήρες είναι δύο είδη των πηγών δεδομένων υγείας. Γι΄ αυτό το λόγο υπάρχουν οι σχέσεις isA που δηλώνουν ότι οι συγκεκριμένες κλάσεις αποτελούν υποκλάσεις των πηγών δεδομένων (DataSource) .

- **Observation**: Η κλάση Observation περιγράφει κάθε παρατήρηση που σχετίζεται με τους δείκτες ευημερίας.

  Ιδιότητες δεδομένων της κλάσης είναι ένας ακέραιος αριθμός για την παρατήρηση (observationLogId) που την προσδιορίζει μοναδικά στο σύστημα της πηγής δεδομένων που την κατέγραψε, καθώς επίσης και ένα χρονόσημο (observationTimestamp) με τύπο ημερομηνίας και ώρας (dateTimeStamp) για τον προσδιορισμό της χρονικής στιγμής που καταγράφηκε η παρατήρηση. Υπάρχει, τέλος και μια ιδιότητα που αν χρησιμοποιηθεί εκφράζει με συμβολοσειρά την γενική περιγραφή της παρατήρησης (observationDescription).

  Όσον αφορά τις ιδιότητες αντικειμένων της κλάσης Observation, έχουμε πρώτον την refersToPerson που δείχνει σε ποιό αντικείμενο τύπου Person αντιστοιχεί η παρατήρηση(σχέση 1 προς 1). Αυτή η ιδιότητα είναι η αντίστροφη της hasObservation της κλάσης Person, διότι όταν ένα άτομο συνδέεται (hasObservation) με μια συγκεκριμένη παρατήρηση τότε αυτή η παρατήρηση θα αναφέρεται (refersToPerson) αντίστοιχα στο συγκεκριμένο άτομο. Δεύτερον, η ιδιότητα recordedBy, που είναι αντίστροφη της records, μας δείχνει την πηγή δεδομένων (DataSource) από την οποία μετρήθηκε η παρατήρηση. Τρίτον, έχουμε την ιδιότητα hasObservationType η οποία δηλώνει ότι κάθε παρατήρηση έχει ένα σύνολο από τύπους παρατηρήσεων(σχέση 1 προς ν με τη κλάση ObservationType). Τέταρτον, υπάρχει και η ιδιότητα raisedWithin που αν χρησιμοποιηθεί υποδηλώνει ότι η παρατήρηση σχετίζεται με μια συγκεκριμένη κατάσταση (State) του ατόμου.

- **ObservationType**: Η κλάση ObservationType αναφέρεται στους τύπους παρατηρήσεων, δηλαδή με άλλα λόγια σε όλες τις μετρήσεις που σχετίζονται με δείκτες ευημερίας, που μπορούν να καταγραφούν για ένα άτομο.

  Οι ιδιότητες τύπου δεδομένων που απαιτούνται εδώ για να χαρακτηρίσουν τα αντικείμενα της κλάσης είναι η observationTypeValue που είναι η τιμή της μετρικής που σχετίζεται

με τον εκάστοτε τύπο παρατήρησης, καθώς και η ιδιότητα observationTypeLoincNum. Η τελευταία συμπληρώνεται στις περιπτώσεις που υπάρχει μια μοναδική συμβολοσειρά από αριθμούς που προσδιορίζει μοναδικά την μετρική σύμφωνα το σύστημα LOINC (Logical Observation Identifiers Names and Codes[1]).

Τέλος, η μοναδική ιδιότητα τύπου αντικειμένων, που έχει αυτή η κλάση είναι η hasObservationTypeUnit που αντιστοιχίζει τον τύπο παρατήρησης δηλαδή την μέτρηση με ένα αντικείμενο της κλάσης Unit για να είναι γνωστή η μονάδα μέτρησης που χρησιμοποιήθηκε για την τιμή της μέτρησης (observationTypeValue).

- **VitalSign, BodyMeasurement, ActivityMetric, SleepMetric, Other**: Σημειώνεται σε αυτό το σημείο, ότι οι κλάσεις VitalSign, BodyMeasurement, ActivityMetric, SleepMetric και Other αποτελούν υποκλάσεις της ObservationType για να εκφράσουν ειδικότερα τα διάφορα είδη μετρικών που υπάρχουν. Έτσι, η VitalSign αναφέρεται σε δείκτες ζωτικής σημασίας, η κλάση BodyMeasurement σε μετρικές του σώματος, οι κλάσεις ActivityMetric και SleepMetric σε μετρικές δραστηριότητας και ύπνου αντίστοιχα, ενώ η κλάση Other σε άλλους τύπους παρατηρήσεων.

- **State**: Η κλάση State χρησιμοποιείται για να περιγράψει την κατάσταση του ατόμου.

  Απαραίτητες ιδιότητες τύπου δεδομένων είναι οι ιδιότητες που μέσω χρονόσημου (dateTimeStamp) κρατάνε πληροφορία για την χρονική στιγμή έναρξης (startTime) και λήξης (endTime) της κατάστασης. Επιπλέον, υπάρχει και μια ιδιότητα (stateLogId), που προσδιορίζει μοναδικά(με έναν αριθμό) την κατάσταση του χρήστη στο σύστημα της πηγής δεδομένων που την κατέγραψε. Υπάρχει, τέλος και άλλη μια ιδιότητα για την συνοπτική περιγραφή της κατάστασης (stateDescription), αν είναι επιθυμητό να συμπληρωθεί.

  Σχετικά με τις ιδιότητες τύπου αντικειμένων έχουμε μόνο την hasStateDuration που συνδέεται με σχέση 1 προς 1 με την κλάση Duration για να ενσωματώσει πληροφορία σχετική με τη διάρκεια της κατάστασης.

- **Activity, Rest, Sleep**: Υποκλάσεις της κλάσης State είναι οι Activity, Rest και Sleep, υποδηλώνοντας ότι πιθανές ειδικεύεσεις της κατάστασης του ατόμου είναι η φυσική δραστηριότητα, η κατάσταση ξεκούρασης(δηλαδή η κατάσταση που το άτομο κάθεται) και η κατάσταση ύπνου.

  Η κλάση Activity έχει ιδιότητα δεδομένων μια συμβολοσειρά χαρακτήρων για την ένταση της φυσικής δραστηριότητας (intensity) και ιδιότητα αντικειμένων την hasActivityType για τον προσδιορισμό του είδους της δραστηριότητας μέσω της σχέσης 1 προς 1 με την κλάση ActivityType.

  Επίσης, η κλάση Sleep έχει την ιδιότητα αντικειμένων hasSleepPhase με πολλαπλότητα 1 προς ν για τον προσδιορισμό των φάσεων του ύπνου.

- **ActivityType**: Η κλάση ActivityType περιγράφει τα είδη των δραστηριοτήτων που υπάρχουν όπως περπάτημα, τρέξιμο, ποδηλασία και πολλά άλλα.

[1]https://loinc.org

Η μόνη ιδιότητα δεδομένων που διαθέτει είναι ένας αριθμός (activityTypeId), που περιγράφει μοναδικά την εκάστοτε δραστηριότητα στο σύστημα που χρησιμοποιεί η πηγή δεδομένων που την κατέγραψε.

- **SleepPhase**: Η κλάση SleepPhase έχει αντικείμενα που περιγράφουν κάθε φάση ύπνου του ατόμου.

  Έχει σαν ιδιότητες δεδομένων μια συμβολοσειρά για να ορίζεται υποχρεωτικά το όνομα (sleepPhaseName) της φάσης ύπνου καθώς και έναν αριθμό που δηλώνει τις φορές (sleepPhaseCount), που ο ύπνος πέρασε από μια τη συγκεκριμένη φάση ύπνου.

  Διαθέτει, ακόμη, την ιδιότητα αντικειμένων hasSleepPhaseDuration που συνδέεται με την κλάση Duration με σχέση 1 προς 1, προκειμένου να υπάρχει πληροφορία σχετική με τη διάρκεια της φάσης του ύπνου.

- **Duration**: Η κλάση Duration περιγράφει την χρονική διάρκεια. Χρησιμοποιεί την ιδιότητα δεδομένων durationValue που είναι η τιμή της διάρκειας και την ιδιότητα αντικειμένων hasDurationUnit για τον απαιτούμενο προσδιορισμό της μονάδας μέτρησης (Unit).

- **Unit**: Η κλάση Unit συνδέεται με πολλές άλλες κλάσεις για να περιγράψει τις μονάδες οποιασδήποτε μέτρησης. Διαθέτει την ιδιότητα δεδομένων unitDescription για τον προσδιορισμό μέσω συμβολοσειράς χαρακτήρων της μονάδας μέτρησης του εκάστοτε αντικειμένου.

- **Notification** Η κλάση Notification αφορά όλες τις ειδοποιήσεις που δημιουργούνται για το άτομο.

  Σαν ιδιότητες δεδομένων έχει μια συμβολοσειρά που περιγράφει την ειδοποίηση (notificationDescription) που δημιουργήθηκε, καθώς επίσης και δύο χρονόσημα (dateTimeStamp) για την χρονική στιγμή που φτιάχτηκε η ειδοποίηση (notificationCreatedTimestamp) και για τη χρονική στιγμή που καλείται να ενεργοποιηθεί (notificationTriggerTimestamp). Διαθέτει, επιπλέον την ιδιότητα triggered, που εκφράζει μέσω μεταβλητής αληθείας αν η ειδοποίηση έχει ενεργοποιηθεί ή όχι.

  Η μοναδική ιδιότητα ατικειμένων είναι η triggeredBy που συνδέεται με την κλάση Observation υποδεικνύοντας από ποιά παρατήρηση προήλθε κάθε φορά η κάθε ειδοποίηση.

- **Summary, Alert**: Οι κλάσεις αυτές αποτελούν υποκλάσεις της Notification καθώς οι ειδοποιήσεις προς το άτομο θα αφορούν είτε προειδοποιήσεις (Alert) για κάποιο σημαντικό ζήτημα είτε περιλήψεις-συνόψεις (Summary) κατάστασης η δραστηριότητας του ατόμου.

  Σημειώνεται ότι η κλάση Alert ενδέχεται να έχει δύο ακόμη ιδιότητες δεδομένων, πέρα από αυτές που κληρονόμησε από την Notification. Η πρώτη χρειάζεται προκειμένου να είναι γνωστό αν η εκάστοτε προειδοποιήση έχει αντιμετωπιστεί ή αλλιώς ληφθεί υπόψιν από το άτομο. Η ιδιότητα αυτή καλείται executed και χρησιμοποιεί μεταβλητή τύπου αληθείας. Η δεύτερη έχει όνομα alertValue καθώς ένδεχεται να είναι χρήσιμη για προειδοποιήσεις που σχετίζονται με κάποια τιμή μετρικής.

Σημειώνεται ότι για την δημιουργία της παραπάνω οντολογίας, καλούμαστε να ορίσουμε ένα **διεθνές αναγνωριστίκο πόρου** (**IRI** - Internationalized Resource Identifier), όπου θα προσδιορίζει μοναδικά την οντολογία μας στο σύνολο των πόρων που υπάρχουν στο διαδίκτυο. Το αναγνωριστικό της οντολογίας μας επομένως είναι το εξής: "http://www.semanticweb.org/pablo/ontologies/2017/5/theHealthAndLife LoggingDataOntology"

Συνοψίζοντας, βασισμένοι στο παραπάνω σχήμα, έχουμε την δυνατότητα να προτυποποιήσουμε τα δεδομένα που συλλέγονται συνολικά από πηγές δεδεμένων, οι οποίες καταγράφουν δείκτες ευημερίας για τους χρήστες τους. Στη συνέχεια, θα γίνει εμφανής ο τρόπος με τον οποίο μπορούμε να αξιοποιήσουμε εν τέλει αυτό το κοινό λεξιλόγιο για να μετασχηματίσουμε τα δεδομένα υγείας σε σημασιολογικά.

4.1.2 Χρήση JSON-LD για την αναπαράσταση δεδομένων

Όπως αναφέρθηκε και σε προηγούμενο κεφάλαιο, πρόκειται ουσιαστικά για μια μέθοδο αναπαράστασης-κωδικοποίησης (encoding) διασυνδεδεμένων δεδομένων (linked data) σε μορφή JSON. Η χρησιμότητα του JSON-LD σε αυτό το σημείο έγκειται στην **ανάγκη αναπαράστασης δεδομένων σε οντολογική μορφή**. Θα περιγραφεί, λοιπόν, ο τρόπος με τον οποίο αξιοποιήθηκε αυτή η μορφή αναπαράστασης για να διευκολυνθεί ο σημασιολογικός μετασχηματισμός των δεδομένων και η διαχείριση τους από το σύστημα.

Ειδικότερα, είναι σημαντικό να αναφερθούν οι λόγοι για τους οποίους επιλέχθηκε η μορφή JSON-LD για την αναπαράσταση. Αρχικά, είναι γνωστό ότι η διαχείριση οντολογικών δεδομένων έχει πολλές απαιτήσεις οι οποίες την καθιστούν δύσκολη. Είναι αναγκάιο, λοιπόν, να βρεθεί ένας πιο εύκολος και ευέλικτος τρόπος αναπαράστασης της οντολογίας. Η μορφή JSON-LD συνδυάζει την **ευκολία αναπαράστασης δεδομένων** σε JSON και την ανάγκη της οντολογίας για **διασύνδεση δεδομένων** μέσω κλάσεων και ιδιοτήτων.

Επιπλέον, είναι ευρέως διαδεδομένη η χρήση JSON για την αναπαράσταση δεδομένων. Πρόκειται ουσιαστικά για μια μορφή που χρησιμοποιείται σχεδόν από όλα τα συστήματα που αναπτύσσουν σύγχρονες εταιρείες. Μια **πιθανή μετάβαση**, λοιπόν, **σε JSON-LD θα ήταν άμεσα εφικτή** και θα είχε σαν αποτέλεσμα την καθιέρωση ενός κοινού κώδικα επικοινωνίας μεταξύ διαφορετικών συστημάτων. Με πολύ έυκολο τρόπο, δηλαδή, θα υλοποιούνταν η επιθυμητή διασύνδεση των δεδομένων.

Τέλος με βάση και τα παραπάνω, εξασφαλίζεται **ευκολία** και **στην επικοινωνία μεταξύ υπηρεσιών**, αφού τόσο η κωδικοποίηση δεδομένων σε JSON (και κατέπέκταση σε JSON-LD) όσο και η αποκωδικοποίηση τους υποστηρίζεται από όλες τις διαδεδομένες γλώσσες προγραμματισμού και μπορεί να γίνει άμεσα χωρίς κινδύνους αλλοίωσης του περιεχομένου της πληροφορίας.

Εμβαθύνοντας, τώρα, στο περιεχόμενο των δεδομένων σε JSON-LD αξίζει να αναφερθούμε στα πεδία που χρησιμοποιούνται για να επιτευχθεί η διασύνδεση της πληροφορίας. Τα πεδία, λοιπόν που έχουν εξέχουσα σημασία είναι τα εξής:

- **@context**

Πρόκειται για ένα πεδίο που έχει σαν τιμή το διεθνές αναγνωριστικό πόρου (IRI) της οντολογίας, όπως ορίστηκε στη προηγούμενη ενότητα. Αυτό το πεδίο υποδηλώνει ότι η δομή του εκάστοτε JSON-LD ακολουθεί το σχήμα της οντολογίας, δηλαδή τις κλάσεις και τις ιδιότητες της. Πραγματοποιείται, λοιπόν, το πρώτο στάδιο της προτυποποίησης των δεδομένων με βάση το κοινό λεξιλόγιο που προτάσσει η οντολογία.

- **@type**

  Το πεδίο αυτό φέρει σαν τιμή το όνομα μιας κλάσης της οντολογίας για να υποδηλώσει ότι το εκάστοτε αντικείμενο του JSON είναι αντικείμενο της συγκεκριμένης κλάσης της οντολογίας. Κατ΄ επέκταση, πρέπει να πληροί τις προϋποθέσεις της κλάσης, δηλαδή να περιέχει πεδία για τις απαραίτητες ιδιότητες της κλάσης με τιμές που ακολουθούν τον τύπο της κάθε ιδιότητας.

- **@id**

  Με τη χρήση αυτού του πεδίου είναι δυνατή η διασύνδεση μεταξύ ίδιων αντικειμένων μιας κλάσης. Με άλλα λόγια, όσα αντικείμενα έχουν την ίδια τιμή στο πεδίο @id , δηλαδή το ίδιο αναγνωριστικό πόρου, μας υποδηλώνουν ότι αναφέρονται στο ίδιο αντικείμενο. Όταν, λοιπόν, μια ομάδα δεδομένων χρησιμοποιεί το ίδιο αντικείμενο σε κάποιο από τα πεδία της, τότε μπορεί να γίνει χρήση του @id για να είναι εμφανές ότι αναφέρονται στον ίδιο πόρο.

Με τη χρήση, λοιπόν, των παραπάνω πεδίων είναι δυνατή η οντολογική αναπαράσταση πληροφορίας για τους δείκτες ευημερίας που καταγράφονται για ένα άτομο από μια πηγή δεδομένων. Με σωστή, δηλαδή, δημιουργία αντικειμένων, ώστε να διατηρείται η σύνδεση των κλάσεων και οι ιδιότητες αυτών, μπορεί κανείς να πετύχει την επιθυμητή διασύνδεση των δεδομένων υγείας και την προτυποποίηση τους βάσει ενός κοινού λεξιλογίου.

Στο κεφάλαιο της υλοποίησης, θα παρουσιαστούν και παραδείγματα μετατροπής δεδομένων από JSON σε JSON-LD βασισμένα στην οντολογία που αναπτύχθηκε προηγουμένως. Εκεί θα επιβεβαιώσουμε την ευκολία που παρέχει το JSON-LD στον σημασιολογικό μετασχηματισμό των δεδομένων.

4.1.3 Χρήση βάσης δεδομένων γραφημάτων για την αναπαράσταση δεδομένων

Ένας ακόμη τρόπος για να διαχειριστούμε εύκολα δεδομένα σε οντολογική μορφή, είναι η γραφική αναπαράστασή τους μέσω μιας βάσης δεδομένων γραφημάτων. Η πιο διαδεδομένη βάση τέτοιου τύπου είναι η **Neo4j**. Όπως προαναφέραμε, χρησιμοποιεί κόμβους (nodes) και ιδιότητες (properties) για να αποθηκεύει τα δεδομένα της, ενώ ιδιαίτερη σημασία έχουν οι σχέσεις (relationship) μεταξύ των κόμβων.

Από την πρώτη στιγμή λοιπόν, είναι προφανές ότι η αναπαράσταση της οντολογίας με γράφο σε μια τέτοια βάση δεδομένων γίνεται εύκολα και άμεσα. Εξάλλου όταν μιλάει κανείς για οντολογία, αναφέρεται στο σχήμα της οντολογίας το οποίο δεν είναι κάτι παραπάνω από **κλάσεις που αντιστοιχούν στους κόμβους του γράφου, από ιδιότητες δεδομένων που αντιστοιχούν στις ιδιότητες του κόμβου και από ιδιότητες**

αντικειμένων που αντιστοιχούν σε σχέσεις μεταξύ κόμβων(δηλαδή κλάσεων). Επιπλέον, κάθε αντικείμενο μιας κλάσης αναπαριστάται με έναν κόμβο, που πέρα από τις ιδιότητες του και τις σχέσεις του με άλλα αντικείμενα, διαθέτει και μια σχέση για να συνδέεται με τον κόμβο της γενικής κλάσης της οντολογίας και για να δείχνει τον τύπο του.

Η τέλεια αντιστοίχιση της οντολογίας σε έναν γράφο του Neo4j δικαιολογεί την επιλογή μας να διαχειριστούμε τα σημασιολογικά δεδομένα με χρήση αυτής της βάσης δεδομένων. Ωστόσο ορθώς αναρωτιέται κανείς γιατί να επιλεγεί και δεύτερος τρόπος αναπαράστασης οντολογικών δεδομένων μετά τη μορφή JSON-LD. Η απάντηση είναι ότι μέσω του Neo4j παρέχονται **περαιτέρω δυνατότητες επεξεργασίας**, που είναι και αυτές επιθυμητές εφόσον έχει εξασφαλιστεί η ευκολία αναπαράστασης και επικοινωνίας μέσω της μορφής JSON-LD.

Είναι σημαντικό, επομένως, να εστιάσουμε στους λόγους για τους οποίους η χρήση της βάσης Neo4j επεκτείνει τις δυνατότητες του συστήματος. Αρχικά, έχει μελετηθεί ότι με τη χρήση βάσης δεδομένων γραφημάτων, η μεταβάσεις από έναν κόμβο σε έναν άλλον γίνονται πολύ γρήγορα, αφού γίνεται χρήση της σχέσης μεταξύ των κόμβων που έχει κύριο ρόλο σε μια τέτοια βάση. Αντίθετα, σε μία βάση διαφορετικού τύπου, δεν υπάρχει η έννοια της σχέσης με αποτέλεσμα να είναι πιο δύσκολη η μεταπήδηση μεταξύ αντικειμένων. Ακόμα, θα ήταν και πιο χρονοβόρα αφού θα έπρεπε να γίνει χρήση πεδίου που θα χρησιμοποιούνταν σαν ξένο κλειδί για την σύνδεση ενός αντικειμένου με ένα άλλο.

Αναλύωντας τώρα πιο συγκεκριμένα τις δυνατότητες που μας προσφέρει το παραπάνω χαρακτηριστικό της βάσης δεδομένων Neo4j, αξίζει να αναφέρθεί ότι μπορούν να πραγματοποιηθούν εύκολα και γρήγορα πολλά **σύνθετα ερωτήματα (queries)** προς την βάση. Ερωτήματα, δηλαδή, που παίρνουν πληροφορία από πολλούς κόμβους και συνδυάζοντάς την οδηγούν σε ενδιαφέροντα αποτελέσματα θα μπορούσαν να είναι ιδιάιτερα χρήσιμα για την **εξαγωγή γενικών συμπερασμάτων**.

Τέτοιες δυνατότητες μπορεί να μην αξιοποιούνται στο έπακρο από το σύστημα μας, διότι συλλέγονται δεδομένα για έναν χρήστη, ωστόσο αυτή η επιλογή της βάσης γραφημάτων έγινε για να υπάρχει η **δυνατότητα επεκτασιμότητας** του συστήματος. Αν, λοιπόν, σκεφτεί κανείς ένα σύστημα με πολλούς χρήστες, θα έβλεπε ότι η βάση δεδομένων γραφημάτων που θα σχηματιζόταν θα είχε έναν κεντρικό γράφο με κόμβους που αναπαριστούν τις κλάσεις της οντολογίας και πάνω σε αυτόν το γράφο θα συνδεόταν ένας συνολικός γράφος για κάθε χρήστη, δημιουργώντας έτσι συνδέσεις τόσο μεταξύ κόμβων του ίδιου ατόμου όσο και μεταξύ κόμβων διαφορετικών ατόμων.

Γίνεται κατανοητό, επομένως, ότι με την παραπάνω δομή, επιτυγχάνεται η διασύνδεση των δεδομένων και η **δυνατότητα εξαγωγής εμπλουτισμένης πληροφορίας** που θα μπορούσε να χρησιμοποιηθεί για διάφορους σκοπούς. Για παράδειγμα, σε μια πιθανή επέκταση του συστήματος, όπου θα συνδεόταν σε αυτό και άλλοι ρόλοι χρηστών, όπως ερευνητές, γιατροί και άλλοι, θα ήταν ιδιαίτερα χρήσιμη η γραφική αναπαράσταση. Με άλλα λόγια, κάθε χρήστης, ανάλογα με το είδος του, θα μπορούσε να βγάλει εξειδικευμένα αποτελέσματα **είτε για ερευνητικούς σκοπούς, είτε για ιατρικούς** αξιοποιώντας την πληροφορία που είναι αποθηκευμένη στη βάση Neo4j. Φυσικά, οποιαδήποτε πρόσβαση σε αυτά τα δεδομένα θα πρέπει να γίνεται με τρόπο που να διασφαλίζεται η ιδιωτικότητα του χρήστη. Καθίσταται

αναγκαίο δηλαδή, η πρόσβαση να γίνεται ανώνυμα και κατόπιν ελέγχου του χρήστη που ζητάει πληροφορία με ευαίσθητο περιεχόμενο.

Στο κεφάλαιο της υλοποίησης, θα παρουσιαστεί στιγμιότυπο του γραφήματος σε Neo4j προκειμένου να γίνει κατανοητός ο τρόπος με τον οποίο αναπαριστώνται τα σημασιολογικά δεδομένα στη μορφή γράφου της βάσης.

4.2 Εξυπηρετητής

Στη παρούσα ενότητα, θα περιγραφεί αναλυτικά η αρχιτεκτονική του εξυπηρετητή μέσω και ενός ειδικού σχήματος αρχιτεκτονικής, ενώ επίσης θα παρουσιαστούν οι λειτουργίες που επιτελεί σε κάθε επίπεδο.

4.2.1 Αρχιτεκτονική εξυπηρετητή

Παρατίθεται σε αυτή την ενότητα το Σχήμα 4.3, όπου παρουσιάζονται πιο ειδικά οι υπηρεσίες κάθε επιπέδου, οι βάσεις δεδομένων που απαιτούνται, καθώς και οι τεχνολογίες που χρησιμοποιήθηκαν για επικοινωνία μεταξύ συγκεκριμένων υπηρεσιών.

Στην επόμενη ενότητα γίνεται αναφορά σε κάθε υπηρεσία ξεχωριστά και στον ρόλο της μέσα στο σύστημα.

4.2.2 Λειτουργίες εξυπηρετητή

Παρατηρώντας το παραπάνω σχήμα αρχιτεκτονικής, γίνεται εν συνεχεία ανάλυση του κάθε συστατικού στοιχείου του εξυπηρετητή.

- **Υπηρεσία Λήψης Ακατέργαστων δεδομένων**

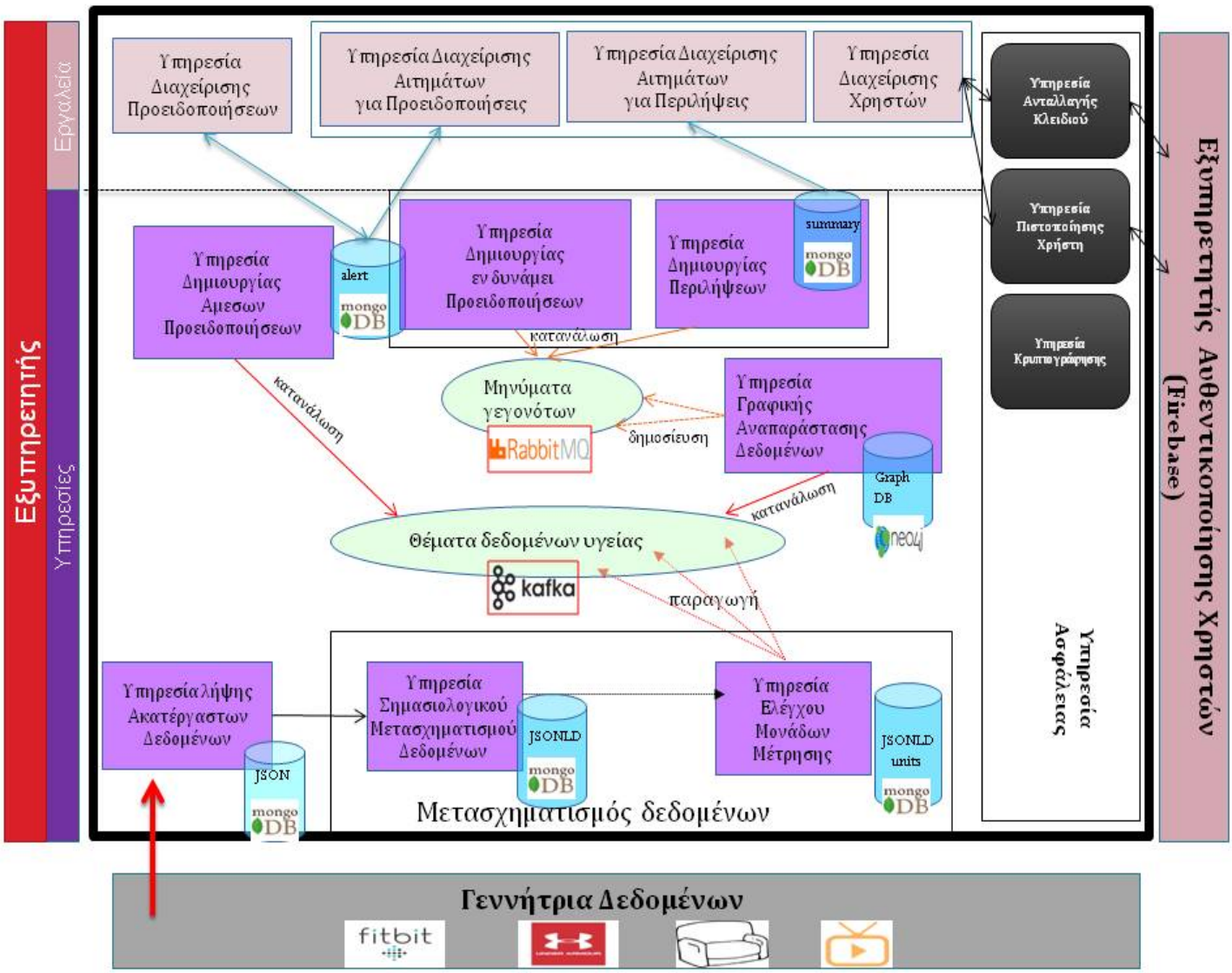
  Η υπηρεσία αυτή υλοποιεί το επίπεδο λήψης δεδομένων. Λαμβάνει κρυπτογραφημένα τα δεδομένα υγείας από την γεννήτρια δεδομένων σε μορφή JSON(JavaScript Object Notation) , και αφού τα αποκρυπτογραφήσει με βάση τον αλγόριθμο που έχει συμφωνηθεί, τα αποθηκεύει απευθείας σε μια μη σχεσιακή βάση δεδομένων (JSON mongoDB), όπου κρατάει όλα τα ακατέργαστα δεδομένα του χρήστη.

- **Υπηρεσία Σημασιολογικού Μετασχηματισμού Δεδομένων**

  Πρόκειται για υπηρεσία που ανήκει στο επίπεδο επεξεργασίας και ο ρόλος της είναι να μετατρέψει τα δεδομένα που συλλέχθηκαν στο προηγούμενο στάδιο σε σημασιολογικά. Με άλλα λόγια, τα δεδομένα μετατρέπονται σε μορφή JSON-LD(JavaScript Object Notation- Linked Data) με βάση το κοινό λεξιλόγιο της οντολογίας που αναπτύχθηκε. Φυσικά, κατα τον μετασχηματισμό αυτό πρέπει να δώσουμε ιδιάιτερη προσοχή ώστε να μην αλλοιωθεί το περιεχόμενο των δεδομένων που λαμβάνονται σε JSON. Σημειώνεται ότι αυτή η υπηρεσία αποθηκεύει τα οντολογικά δεδομένα που παράγει σε κατάλληλη μη σχεσιακή βάση του εξυπηρετητή (JSON-LD mongoDB).

- **Υπηρεσία Ελέγχου Μονάδων Μέτρησης**

  Σε αυτό το επιμέρους στάδιο επεξεργασίας, τα δεδομένα επιδέχονται μία ακόμα μετατροπή. Συγκεκριμένα, ελέγχονται οι μονάδες μέτρησης προκειμένου να



Σχήμα 4.3: Αρχιτεκτονική εξυπηρετητή

δημιουργηθούν νέα JSON-LD, τα οποία έχουν ίδιες μονάδες μέτρησης στα πεδία που εκφράζουν ίδιου είδους πληροφορία. Αυτό γίνεται για να διευκολυνθεί σε επόμενα στάδια η λογική διαχείριση της πληροφορίας. Και εδώ η αποθήκευση γίνεται σε αντίστοιχη μη σχεσιακή βάση δεδομένων (JSON-LD mongoDB).

Με την ολοκλήρωση του σημασιολογικού μετασχηματισμού, τα δεδομένα πρέπει να περάσουν στο δεύτερο επίπεδο επεξεργασίας. Για το λόγο αυτό με τη χρήση της τεχνολογίας Apache Kafka, γίνεται παραγωγή (publish) μηνυμάτων με περιεχόμενο την πληροφορία σε JSON-LD. Αυτά τα μηνύματα διανέμονται σε συγκεκριμένα θέματα δεδομένων υγείας (topics) που θα παρουσιαστούν στο κομμάτι της υλοποίησης.

Σημειώνεται ότι ο λόγος που διαχωρίζονται οι δύο προηγούμενες λειτουργίες είναι για να δειχθεί ότι σε ένα ιδανικό σενάριο, η μόνη μετατροπή που θα έπρεπε να πραγματοποιηθεί θα ήταν αυτή που επιτελεί η Υπηρεσία Ελέγχου Μονάδων Μέτρησης, καθώς η συλλογή των δεδομένων θα γινόταν απευθείας σε οντολογική μορφή λόγω του κοινού λεξιλογίου.

- **Υπηρεσία Δημιουργίας Άμεσων Προειδοποιήσεων**

  Η Υπηρεσία Δημιουργίας Άμεσων Προειδωποιήσεων καταναλώνει (consume) τα μηνύματα από θέματα δεδομένων υγείας που απαιτούν άμεση διαχείριση, καθώς είναι ζωτικής σημασίας για το άτομο (όπως ο καρδιακός παλμός). Πρόκειται, ουσιαστικά, για διαχείριση δεδομένων σε πραγματικό χρόνο για να αποφευχθεί τυχόν καθυστέρηση επεξεργασίας. Ελέγχονται, λοιπόν, οι τιμές των συγκεκριμένων δεικτών ευημερίας και αν παρατηρηθεί κάποιος κίνδυνος, δημιουργείται μια προειδοποίηση σε οντολογική μορφή και αποθηκεύεται σε σχετική μη σχεσιακή βάση για τις προειδοποιήσεις (alert mongoDB).

- **Υπηρεσία Γραφικής Αναπαράστασης Δεδομένων**

  Αυτή η υπηρεσία καταναλώνει (consume) κάθε είδους μήνυμα που παράγεται μέσω Apache Kafka. Συλλέγει, δηλαδή, όλα τα σημασιολογικά δεδομένα σε JSON-LD προκειμένου να τα αποθηκεύσει στην βάση δεδομένων γραφημάτων Neo4j. Σημειώνεται ότι η υπηρεσία αναλαμβάνει να φορτώσει στη βάση αυτή και τους κόμβους που σχετίζονται με τις κλάσεις της οντολογίας. Τα σημασιολογικά δεδομένα λοιπόν, εισέρχονται στο γράφο του χρήστη και συνδέονται με τους κατάλληλους κόμβους-κλάσεις της οντολογίας προκειμένου να επιτευχθεί η διασύνδεση της πληροφορίας.

  Σε επόμενο στάδιο, ελέγχεται το είδος της πληροφορίας και βάσει γεγονότων, που παρατηρούνται από τα δεδομένα υγείας του χρήστη, γίνεται δημοσίευση (publish) συνοπτικών μηνυμάτων με χρήση της τεχνολογίας RabbitMQ. Τέτοια μηνύματα δηλώνουν την εμφάνιση συγκεκριμένων συμβάντων στη μέρα του χρήστη και κατανέμονται ανάλογα με το είδος τους σε αντίστοιχες ουρές μηνυμάτων.

- **Υπηρεσία Δημιουργίας Εν Δυνάμει Προειδοποιήσεων**

  Η παρούσα υπηρεσία, καταναλώνει από τις ουρές του RabbitMQ μηνύματα, που μπορεί να οδηγήσουν σε πιθανές προειδοποιήσεις. Ειδικότερα, υπάρχουν περιπτώσεις που κάποιο δεδομένο μπορεί να σημάνει μια πιθανή προειδοποίηση όχι άμεσα, αλλά μετά από ένα χρονικό διάστημα αν το άτομο δεν κάνει κάποιες συγκεκριμένες ενέργειες.

Η υπηρεσία αυτή παρακολουθεί αυτές τις περιπτώσεις και δημιουργεί προειδοποιήσεις τις οποίες αποθηκεύει σε JSON-LD μορφή με βάση την οντολογία, στη σχετική μη σχεσιακή βάση προειδοποιήσεων (alert mongoDB). Αναλυτίκα, θα δούμε στο κεφάλαιο της υλοποίησης τέτοιες πιθανές προειδοποιήσεις, ωστόσο μπορούμε να αναφέρουμε σε αυτό το σημείο συνοπτικά ένα παράδειγμα. Έστω, λοιπόν, ότι ο χρήστης μπαίνει σε κατάσταση, κατά την οποία είναι καθιστός (Rest State). Εκείνη τη στιγμή η Υπηρεσία Δημιουργίας Εν Δυνάμει Προειδοποιήσεων θα ενημερωθεί με σχετικό μήνυμα και θα δημιουργήσει μια πιθανή προειδοποίηση, η οποία θέλουμε να εμφανιστεί μετά από κάποιο χρονικό διάστημα(π.χ. 3 ώρες) στον χρήστη, εφόσον εκείνος παραμείνει σε αυτή τη κατάσταση για αυτό το διάστημα. Αν ο χρήστης αλλάξει κατάσταση πιο νωρίς, η εν δυνάμει προειδοποίηση θα διαγραφεί από την ίδια υπηρεσία.

- **Υπηρεσία Δημιουργίας Περιλήψεων**

  Η Υπηρεσία Δημιουργίας Περιλήψεων καταναλώνει μηνύματα από ουρές του RabbitMQ, τα οποία αφορούν γεγονότα που οδηγούν σε περιλήψεις δραστηριότητας ή κατάστασης. Με άλλα λόγια, όταν ο χρήστης ολοκληρώσει μια δραστηριότητα, που είναι άξια αναφοράς, η παρούσα υπηρεσία θα λάβει γνώση αυτού του γεγονότος και θα δημιουργήσει τις περιλήψεις που απαιτούνται. Η περίληψη αυτή θα είναι βασισμένη στις μετρήσεις της εκάστοτε πηγής δεδομένων που κατέγραψε την συγκεκριμένη δραστηριότητα/κατάσταση. Σημειώνεται, επιπλέον, ότι η υπηρεσία υπολογίζει και περιλήψεις ημέρας όταν λάβει συγκεκριμένα μηνύματα. Όλες αυτές οι περιλήψεις θα αποθηκευτούν σε αντίστοιχη μη σχεσιακή βάση των περιλήψεων (summary mongoDB), σε αναπαράσταση JSON-LD με βάση το σχήμα οντολογίας. Αναλυτικά, θα δούμε τις λειτουργίες της υπηρεσία αυτής στο κεφάλαιο της υλοποίησης.

- **Υπηρεσία Διαχείρισης Προειδοποιήσεων**

  Η Υπηρεσία Διαχείρισης Προειδοποιήσεων που ανήκει στο επίπεδο εργαλείων του εξυπηρετητή αναλαμβάνει την παρακολούθηση των προειδοποιήσεων που υπάρχουν στην σχετική βάση (alert mongoDB) και τη διαχείρισή τους, προκειμένου να είναι γνωστό ανά πάσα στιγμή για ποιές προειδοποιήσεις πρέπει να ενημερωθεί ο χρήστης.

- **Υπηρεσία Διαχείρισης Αιτημάτων για Προειδοποιήσεις**

  Στο επίπεδο εργαλείων και συγκεκριμένα στις υπηρεσίες πελάτη του εξυπηρετητή υπάρχει η Υπηρεσία Διαχείρισης Αιτημάτων για Προειδοποιήσεις. Το έργο της είναι να λαμβάνει αιτήματα (requests) από την εφαρμογή για αποστολή των προειδοποιήσεων του χρήστη που ανήκουν σε ένα συγκεκριμένο χρονικό διάστημα. Η υπηρεσία, λοιπόν, αντλεί από τη μη σχεσιακή βάση των προειδοποιήσεων (alert mongoDB) εκείνες που ανήκουν σε αυτό το διάστημα και τις στέλνει στην εφαρμογή. Φυσικά η επικοινωνία πρέπει να γίνεται με ασφάλεια και γι΄ αυτό το λόγο το αίτημα λαμβάνεται κρυπτογραφημένο, ενώ αντίστοιχα η απάντηση κρυπτογραφείται πριν σταλεί.

- **Υπηρεσία Διαχείρισης Αιτημάτων για Περιλήψεις**

  Ακόμα μία υπηρεσία πελάτη του εξυπηρετητή είναι η Υπηρεσία Διαχείρισης Αιτημάτων για Περιλήψεις, η οποία καλείται να απαντήσει αιτήματα για αποστολή περιλήψεων κατάστασης ή δραστηριότητας του χρήστη που ανήκουν σε ένα ορισμένο από το αίτημα

χρονικό διάστημα. Αναζητά επομένως στη μη σχετική βάση (summary mongoDB) τις περιλήψεις που αποτελούν μέρος της απάντησης και τις αποστέλλει στην εφαρμογή. Και πάλι απαιτούνται ενέργειες για την αποκρυπτογράφηση του αιτήματος και την κρυπτογράφηση της απάντησης για να διασφαλιστεί η προστασία του ευαίσθητου περιεχομένου της πληροφορίας.

- **Υπηρεσία Διαχείρισης Χρηστών**

  Η τρίτη και τελευταία υπηρεσία πελάτη είναι η Υπηρεσία Διαχείρισης Χρηστών, που όπως φανερώνει και το όνομα, αναλαμβάνει την διαχείριση των δεδομένων των χρηστών. Με άλλα λόγια, ο ρόλος της σχετίζεται με τις διαδικασίες εγγραφής και σύνδεσης των χρηστών προκειμένου να γίνουν οι απαραίτητες ενέργειες για να πραγματοποιηθεί η χειραψία (handshake) μεταξύ εφαρμογής χρήστη και εξυπηρετητή. Η υπηρεσία αυτή διαχειρίζεται μια σχετική μη σχεσιακή βάση για τους χρήστες (users mongoDB).

- **Υπηρεσία Πιστοποίησης Χρήστη**

  Στο κομμάτι των υπηρεσιών ασφαλείας υπάρχει η Υπηρεσία Πιστοποίησης Χρήστη. Ο ρόλος της είναι να πιστοποιήσει τον χρήστη κατά την εγγραφή, χρησιμοποιώντας κατάλληλες υπηρεσίες που παρέχονται από τον Εξυπηρετητή Αυθεντικοποίησης Χρηστών (Firebase).

- **Υπηρεσία Ανταλλαγής Κλειδιού**

  Η Υπηρεσία Ανταλλαγής Κλειδιού αποτελεί μια δεύτερη υπηρεσία των υπηρεσιών ασφαλείας, η οποία παίζει καθοριστικό ρόλο στην επίτευξη κρυπτογραφημένης επικοινωνίας με την εφαρμογή. Το έργο της είναι να πιστοποιήσει τον χρήστη κατά την εγγραφή μέσω της προηγούμενης υπηρεσίας και στη συνέχεια να υλοποιήσει την διαδικασία για την ασφαλή ανταλλαγή κλειδιού κρυπτογράφησης μεταξύ χρήστη και εξυπηρετητή. Κάθε επικοινωνία πλέον μεταξύ του εκάστοτε χρήστη και του εξυπηρετητή θα γίνεται με χρήση αυτού του κλειδιού το οποίο και αποθηκεύεται στην βάση των χρηστών (users mongoDB).

- **Υπηρεσία Κρυπτογράφησης**

  Μια τελευταία υπηρεσία ασφάλειας είναι η Υπηρεσία Κρυπτογράφησης. Διαθέτει τις απαραίτητες συναρτήσεις για κρυπτογράφηση και αποκρυπτογράφηση προκειμένου να κληθούν από τις Υπηρεσίες Πελάτη για να επιτευχθεί σωστά και με ασφάλεια η ανταλλαγή δεδομένων μεταξύ εφαρμογής και εξυπηρετητή.

Συνοψίζοντας, η παρούσα ενότητα αναφέρθηκε γενικά στις υπηρεσίες από τις οποίες συγκροτούνται τα τρία επίπεδα του εξυπηρετητή (επίπεδο συλλογής δεδομένων, επίπεδο επεξεργασίας δεδομένων και επίπεδο αξιοποίησης). Ειδικότερα, έγινε μια συνοπτική περιγραφή του ρόλου που αναλαμβάνει κάθε υπηρεσία. Στο επόμενο κεφάλαιο θα αναλυθεί εκτενώς ο τρόπος με τον οποίο υλοποιήθηκαν οι λειτουργίες κάθε υπηρεσίας.

4.3 Εφαρμογή

Σε αυτή την ενότητα, θα γίνει περιγραφή του αρχιτεκτονικού σχεδιασμού της εφαρμογής. Πιο συγκεκριμένα, παρουσιάζονται αρχικά οι απαιτήσεις της εφαρμογής σχετικά με τις οθόνες που

θα προβάλλονται στον χρήστη. Στη συνέχεια παρατίθεται αναλυτικό σχήμα των υπηρεσιών της εφαρμογής και τέλος αναφέρονται γενικά οι λειτουργίες που αναλαμβάνει κάθεμια.

4.3.1 Οθόνες εφαρμογής

Πρώτα απ΄όλα θα πραγματοποιηθεί ο σχεδιασμός του γραφικού περιβάλλοντος της εφαρμογής. Οι απαιτήσεις της εφαρμογής σχετικά με τις οθόνες που πρέπει παρουσιάζει στον χρήστη καθορίζονται από τις λειτουργίες που θέλουμε να παρέχουμε και από την ανάγκη για ένα φιλικό περιβάλλον

Αρχικά, είναι προφανές ότι απαιτείται **οθόνη για την σύνδεσή των χρηστών στην εφαρμογή**. Παρόλο που τα δεδομένα που προσομοιώσαμε αφορούν έναν χρήστη, η σύνδεση παραπάνω χρηστών στο σύστημά μας είναι κάτι επιθυμητό. Για αυτό το λόγο, η πρώτη οθόνη που θα ανοίγει κατά την πλοήγηση στην εφαρμογή, θα παρέχει την δυνατότητα σύνδεσης ενός εγγεγραμμένου χρήστη συμπληρώνοντας τα σχετικά πεδία για την αυθεντικοποίηση του(βλ. ενότητα 5.3.1). Παράλληλα, αν ο χρήστης δεν έχει εγγραφεί ακόμη, θα παρακινείται να ξεκινήσει την διαδικασία της εγγραφής σε μια άλλη σχετική οθόνη. Επομένως, η δεύτερη απαραίτητη **οθόνη** είναι **για την εγγραφή ενός νέου χρήστη**. Καλεί τον χρήστη να συμπληρώσει τόσο πεδία που είναι χρήσιμα για την αυθεντικοποίηση του, όσο και πεδία που σχετίζονται με άλλα προσωπικά δεδομένα τα οποία θα χρησιμοποιηθούν από την εφαρμογή μελλοντικά(βλ. ενότητα 5.3.1).

Σε επόμενο στάδιο, με την εγκεκριμένη είσοδο ενός χρήστη στην εφαρμογή, χρειαζόμαστε περαιτέρω οθόνες. Ο διαχωρισμός τους καθορίζεται από τα είδη των γενικών λειτουργιών που επιθυμούμε να παρέχουμε ενώ η εναλλάγη από την μία στην άλλη θα μπορούσε να γίνει με τη χρήση μιας καρτέλας (tab) ανά λειτουργία. Σαν αποτέλεσμα, χρειαζόμαστε μια οθόνη με το προφίλ του χρήστη για να μπορεί ο ίδιος να κάνει βασικές επιλογές, μια δεύτερη οθόνη για να έχει την δυνατότητα παρακολούθησης των δεικτών ευημερίας του και μια τρίτη για την διαχείριση των στόχων του. Ο λόγος που επιλέγουμε λειτουργία σχετική με στόχους του χρήστη είναι για να υπάρχει η δυνατότητα σύγκρισης τους με την πραγματικότητα προκειμένου να δίνονται κίνητρα να τους πετύχει για να βελτιώσει εν τέλει την ποιότητα ζωής του.

Ξεκινώντας από το **προφίλ του χρήστη**, μπορούμε να του δώσουμε την επιλογή να εισάγει δική του φωτογραφία παρέχοντας έτσι ένα πιο οικείο και φιλικό περιβάλλον. Επίσης, είναι απαραίτητο σε αυτή την οθόνη να φαίνονται οι πηγές δεδομένων που έχει συνδεδεμένες ο χρήστης στην εφαρμογή, καθώς και να παρέχεται η δυνατότητα είτε εισαγωγής μιας νέας συσκευής είτε διαγραφής μιας προϋπάρχουσας. Τέλος, είναι επιθυμητό να επιλέγονται μέσω συγκεκριμένου μενού, λειτουργίες για την αποσύνδεση του χρήστη και για τις προτιμήσεις του σχετικά με την εφαρμογή.

Δευτερον, σχεδιάζοντας την **οθόνη για παρακολούθηση των δεικτών ευημερίας** από τον χρήστη, πρέπει να σκεφτούμε τις επιμέρους επιλογές παρακολούθησης που θα δίνονται από την εφαρμογή. Συγκεκριμένα, για να υπάρχει αμεσότητα, είναι καλό να παρέχεται σε πραγματικό χρόνο στον χρήστη η δυνατότητα να παρακολουθεί την εκάστοτε κατάστασή του μέσα στη μέρα. Για αυτό το λόγο, μια οθόνη παρακολούθησης των δεδομένων υγείας της τρέχουσας ημέρας θα ήταν ιδιαίτερα χρήσιμη. Επιπλέον, θα ήταν εξίσου χρήσιμη μία οθόνη για τις περιλήψεις μετρικών προηγούμενων ημερών με στόχο την δυνατότητα παροχής ιστορικού στον χρήστη. Τέλος, ενδέχεται ο χρήστης να επιθυμεί παρακολούθηση πιο ειδικών

συμπερασμάτων τα οποία θα μπορούσαν να του δώσουν καθοριστική πληροφορία για τη βελτίωση του τρόπου ζωής του. Έτσι, υπάρχει η απαίτηση για μια τρίτη επιλογή στην οθόνη παρακολούθησης που θα έχει πιο εξειδικευμένη πληροφορία.

Εμβαθύνοντας τώρα στις επιλογές στην οθόνη παρακολούθησης, πρέπει να σχεδιάσουμε αρχίκα το τρόπο με τον οποίο θα εμφανίζεται στην οθόνη η **τρέχουσα μέρα**. Είναι γνωστό ότι τα δεδομένα που έχουμε συλλέξει μέσα στη μέρα αφορούν μετρικές ευημερίας που σχετίζονται με την καθημερινή του δραστηριότητα. Επιπροσθέτως, είναι γνωστό ότι θα υπάρχουν συγκεκριμένοι στόχοι για τις τιμές των μετρικών, που θα έχουν τεθεί από τον ίδιο μέσω της λειτουργίας των στόχων χρήστη. Επομένως, ένας πίνακας με τους δείκτες ευημερίας του ατόμου, τις τιμές που έχουν καταγραφεί την τρέχουσα μέρα και τις τιμές -στόχους που έχουν τεθεί, θα αποτελούσε μια φιλική και συνάμα ιδιαίτερα χρήσιμη αναπαράσταση της τελικής πληροφορίας.

Παρομοίως, για την **επιλογή της περιλήψης μετρικών προηγούμενης ημέρας**, μια αναπαράσταση ίδια με την προαναφερθείσα θα ικανοποιούσε τις ανάγκες του χρήστη. Φυσικά, για να πληροί και την ανάγκη παροχής ιστορικού χρήστη, θα ήταν απαραίτητο να δοθεί στη συγκεκριμένη οθόνη και η επιλογή για αναζήτηση περίληψης μιας συγκεκριμένης μέρας.

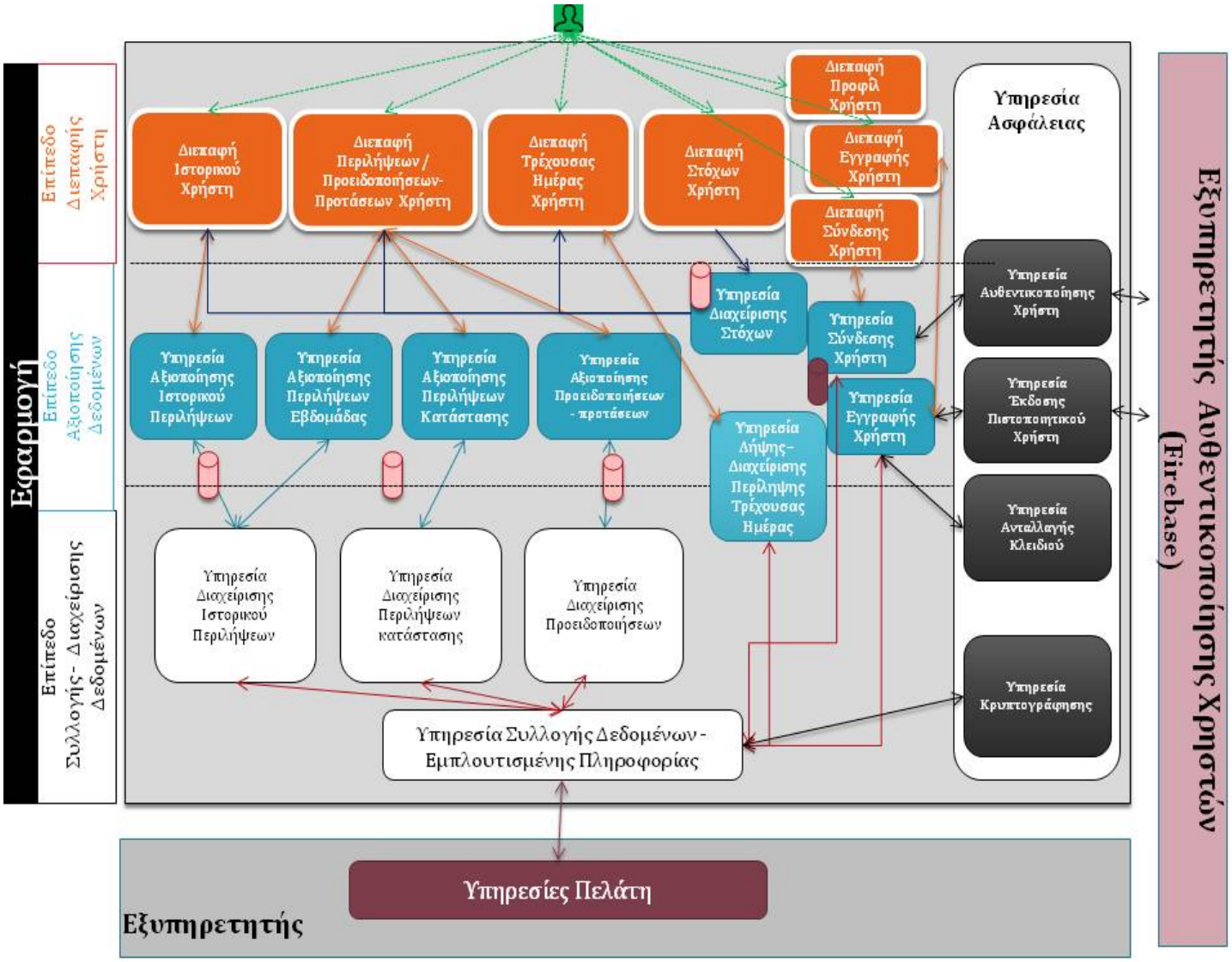
Όσον αφορά τώρα την επιλογή στην οθόνη παρακολούθησης για πιο εξειδικευμένη πληροφορία, είναι επιθυμητό το σύστημά μας να παρέχει στον χρήστη **προειδοποιήσεις** για τυχόν κινδύνους στη καθημερινότητά του, αντίστοιχες **προτάσεις** για αντιμετώπιση αυτών, και τέλος **περιλήψεις σημαντικών καταστάσεων ή δραστηριοτήτων** του. Ακόμη, μια αναπαράσταση των ημερήσιων τιμών ενός δείκτη ευημερίας, που είναι άξιος μελέτης στη διάρκεια μιας εβδομάδας θα ήταν κάτι εξίσου χρήσιμο. Καταλήγουμε, λοιπόν, στη διαμέριση της συγκεκριμένης οθόνης σε τρείς πίνακες. Ο πρώτος περιλαμβάνει τις τελευταίες προειδοποιήσεις και τις αντίστοιχες προτάσεις προς τον χρήστη, ο δεύτερος τις τελευταίες περιλήψεις κατάστασης και ο τρίτος την δυνατότητα επιλογής ενός δείκτη ευημερίας για αναπαράσταση της διακύμανσής του την τελευταία εβδομάδα.

Ολοκληρώνοντας, θα γίνει σχεδίαση και της τρίτης οθόνης, που αφορά τους**στόχους του χρήστη**. Ειδικότερα, μέσω αυτής της οθόνης, θα είναι δυνατή η αποθήκευση νέων στόχων χρήστη, οι οποίοι θα ισχύουν από την τρέχουσα μέρα. Το κύριο περιεχόμενο αυτής της οθόνης θα είναι ένας συγκεντρωτικός πίνακας όλων των δεικτών ευημερίας που μετρώνται για ένα άτομο από τις πηγές δεδομένων κατά τη διάρκεια μιας μέρας. Σε κάθε δείκτη θα δίνεται αντίστοιχο πεδίο, για συμπλήρωση της τιμής- στόχου που επιθυμεί να θέσει ο χρήστης.

Έχοντας, λοιπόν, μια συνολική εικόνα της διεπαφής της εφαρμογής με τον χρήστη και έχοντας επίσης κατανοήσει τις λειτουργίες που επιθυμούμε να υλοποιήσουμε μπορούμε να συνεχίσουμε στις επόμενες ενότητες. Εκεί θα μελετήσουμε τον σχεδιασμό της αρχιτεκτονικής και των υπηρεσιών της εφαρμογής του συστήματος.

4.3.2 Αρχιτεκτονική εφαρμογής

Παρατίθεται σε αυτήν την ενότητα, το σχήμα αρχιτεκτονικής της εφαρμογής (Σχήμα 4.4), όπου φαίνονται οι υπηρεσίες κάθε επιπέδου και οι βάσεις δεδομένων που διαχειρίζεται καθε μία.



Σχήμα 4.4: Αρχιτεκτονική εφαρμογής

Γίνονται ευδιάκριτα, επομένως, τα τρία επίπεδα της αρχιτεκτονικής της εφαρμογής (επίπεδο συλλογής-διαχείρισης δεδομένων, επίπεδο αξιοΠροειποίησης δεδομένων και επίπεδο διεπαφής χρήστη). Διαχωρίζονται, επιπλέον, οι αντίστοιχες υπηρεσίες του καθε στρώματος, καθώς και οι υπηρεσίες ασφαλείας που ενεργούν κάθετα σε όλα τα επίπεδα της εφαρμογής. Τέλος, είναι εμφανής και ο τρόπος με τον οποίο αλληλεπιδρά η εφαρμογή με τον εξυπηρετητή του συστήματος, με τον Εξυπηρετητή Αυθεντικοποίησης Χρηστών (Firebase) και με τον τελικό χρήστη της εφαρμογής.

4.3.3 Λειτουργίες εφαρμογής

Παρατηρώντας το σχήμα της αρχιτεκτονικής, γίνεται εν συνεχεία ανάλυση του κάθε συστατικού στοιχείου της εφαρμογής.

- **Υπηρεσία Συλλογής Δεδομένων-Εμπλουτισμένης Πληροφορίας**

  Αυτή η υπηρεσία είναι υπεύθυνη για την αποστολή αιτημάτων των υπηρεσιών της εφαρμογής προς τον εξυπηρετητή του συστήματος. Υλοποιεί, με άλλα λόγια, την επικοινωνία με τον εξυπηρετητή, φροντίζοντας και τις απαιτήσεις ασφαλείας κρυπτογραφώντας τα αιτήματα που στέλνει και αποκρυπτογραφώντας τις απαντήσεις που λαμβάνει.

- **Υπηρεσία Διαχείρισης Περιλήψεων Κατάστασης**

  Ο ρόλος της συγκεκριμένης υπηρεσίας είναι να ενημερώνεται για τυχόν περιλήψεις κατάστασης ή δραστηριότητας του χρήστη. Συντάσσει, επομένως, κατάλληλα αιτήματα, που θα σταλούν μέσω της προηγούμενης υπηρεσίας στον εξυπηρετητή, προκειμένου να λάβει το σύνολο των περιλήψεων για ένα συγκεκριμένο χρονικό διάστημα. Ειδικότερα, λαμβάνει πολλαπλές περιλήψεις για την ίδια δραστηριότητα ή κατάσταση αφού ενδέχεται να έχει καταγραφεί από διαφορετικές πηγές δεδομένων. Καθίσταται αναγκαίο, λοιπόν, να γίνει μια σύγκριση των τιμών σε κάθε δείκτη ευημερίας με σκοπό να δημιουργηθεί ένα τελικό συμπέρασμα για την εκάστοτε δραστηριότητα / κατάσταση του χρήστη. Αυτή η υπηρεσία, επομένως, επιτελεί και αυτή τη λειτουργία αποθηκεύοντας τη τελική περίληψη στην αντίστοιχη συλλογή (collection) περιλήψεων της μη σχεσιακής βάσης του χρήστη. Παράλληλα, ενημερώνει και την Υπηρεσία Αξιοποιήσης Περιλήψεων Κατάστασης του ανώτερου επιπέδου για την περίληψη που έφτιαξε.

- **Υπηρεσία Διαχείρισης Ιστορικού Περιλήψεων**

  Η Υπηρεσία Διαχείρισης Ιστορικού Περιλήψεων είναι υπεύθυνη για την συλλογή των συνολικών περιλήψεων μια ολόκληρης ημέρας. Ο εξυπηρετητής, όπως είδαμε παρέχει τέτοια δυνατότητα στους πελάτες του, οπότε η παρούσα υπηρεσία συντάσσει ανάλογα αιτήματα, προκειμένου να συλλέξει περιλήψεις ημέρας, μέσω της Υπηρεσίας Συλλογής Δεδομένων-Εμπλουτισμένης Πληροφορίας. Αυτές τις αποθηκεύει σε σχετική συλλογή της βάσης του χρήστη. Επιπλέον, ενημερώνει και την Υπηρεσία Αξιοποίησης Ιστορικού Περιλήψεων του επόμενου επιπέδου.

- **Υπηρεσία Διαχείρισης Προειδοποιήσεων**

  Μία ακόμα υπηρεσία αυτού του επιπέδου είναι η Υπηρεσία Διαχείρισης Προειδοποιήσεων, η οποία αποτελείται από μια σειρά επιμέρους υπηρεσιών

για την συλλογή των προειδοποιήσεων του χρήστη. Διαχειρίζεται τις προειδοποιήσεις που λαμβάνει εξετάζοντας και συγκρίνοντας τις τιμές μετρικών που κατέγραψε κάθε πηγή δεδομένων. Εν τέλει, φτιάχνει τελικές προειδοποιήσεις και σχετικές προτάσεις για τον χρήστη αποθηκεύοντας τες σε αντίστοιχη συλλογή της βάσης του χρήστη και ενημερώνοντας την Υπηρεσία Αξιοποίησης Προειδοποιήσεων/Προτάσεων για τη νέα πληροφορία.

- **Υπηρεσία Αξιοποίησης Περιλήψεων Κατάστασης**

  Στο επίπεδο αξιοποίησης δεδομένων βρίσκεται η Υπηρεσία Αξιοποίησης Περιλήψεων Κατάστασης, η οποία ενημερώνεται πρώτον από την βάση του χρήστη για τις τελευταίες περιλήψεις κατάστασης ή δραστηριότητας και δεύτερον από την αντίστοιχη υπηρεσία διαχείρισης για τις νέες περιλήψεις. Ο ρόλος της είναι να γνωστοποιήσει στον χρήστη τα συγκεντρωτικά αποτελέσματα μέσω των κατάλληλων διεπαφών χρήστη (οθόνες), που βρίσκονται στο γραφικό περιβάλλον της εφαρμογής. Ανανεώνει, επομένως με κατάλληλο τρόπο τις οθόνες της εφαρμογής συγχρονίζοντάς τες με τα δεδομένα που συλλέγονται.

- **Υπηρεσία Αξιοποίησης Ιστορικού Περιλήψεων**

  Παρομοίως με την λογική που ακολουθεί η προηγούμενη υπηρεσία, ο ρόλος της Υπηρεσίας Αξιοποίησης Ιστορικού Περιλήψεων είναι να ανανεώνει τις αντίστοιχες διεπαφές χρήστη, με το ιστορικό περιλήψεων ημέρας που καλείται η εφαρμογή να παρουσιάσει στο χρήστη. Και πάλι η απόκτηση αυτής της πληροφορίας γίνεται είτε από την αντίστοιχη υπηρεσία διαχείρισης, είτε από την βάση του χρήστη.

- **Υπηρεσία Αξιοποίησης Προειδοποιήσεων**

  Ακριβώς με τον ίδιο τρόπο, η παρούσα υπηρεσία, φροντίζει να ανανεώνει τις οθόνες της εφαρμογής με τις τελευταίες προειδοποιήσεις χρήστη που συλλέγει από την υπηρεσία διαχείρισης ή από την αντίστοιχη συλλογή στη βάση του χρήστη.

- **Υπηρεσία Αξιοποίησης Περιλήψεων Εβδομάδας**

  Η συγκεκριμένη υπηρεσία αντλεί από την συλλογή περιλήψεων ημέρας της βάσης του χρήστη όλη την απαραίτητη πληροφορία προκειμένου να φτιάξει περιλήψεις εβδομάδας για έναν δείκτη ευημερίας. Κατόπιν, παρέχει τα αποτελέσματα στην αντίστοιχη διεπαφή με σκοπό να τα παρουσιάσει στον χρήστη, όταν ζητηθούν.

- **Υπηρεσία Λήψης-Διαχείρισης Περίληψης Τρέχουσας Ημέρας**

  Αυτή η υπηρεσία έχει τοποθετηθεί εσκεμμένα μεταξύ του επιπέδου λήψης και διαχείρισης και του επιπέδου αξιοποίησης δεδομένων. Ο λόγος είναι ότι η λήψη περίληψης της τρέχουσας ημέρας από τον εξυπηρετητή γίνεται δυναμικά τη στιγμή που ζητηθεί από τον χρήστη με στόχο τον συγχρονισμό της εφαρμογής με τα εκάστοτε δεδομένα του εξυπηρετητή. Τέλος, η ίδια υπηρεσία αξιοποιεί την πληροφορία που λαμβάνει ανανεώνοντας την κατάλληλη διεπαφή χρήστη της εφαρμογής.

- **Υπηρεσία Διαχείρισης Στόχων**

  Η Υπηρεσία Διαχείρισης Στόχων, όπως φαίνεται και από την ονομασία της, διαχειρίζεται τους στόχους που θέτει ο χρήστης για τις μετρικές που του παρουσιάζονται μέσω της εφαρμογής. Ο ρόλος της υπηρεσίας είναι να λαμβάνει τους στόχους αυτούς από

αντίστοιχη διεπαφή του χρήστη, να τους αποθηκεύει σε σχετική συλλογή της βάσης του χρήστη και στη συνέχεια να τους αξιοποιεί δίνοντας την δυνατότητα σύγκρισης των πραγματικών τιμών με τις τιμές-στόχους. Ανανεώνει, επομένως κατάλληλα τις αντίστοιχες οθόνες της εφαρμογής.

- **Υπηρεσία Εγγραφής Χρήστη**

  Η παρούσα υπηρεσία πραγματοποιεί όλες τις ενέργειες που πρέπει να γίνουν για να εγγραφεί ο χρήστης στο σύστημά μας. Τα δεδομένα εγγραφής του χρήστη τα λαμβάνει από την αντίστοιχη διεπαφή της εφαρμογής, έλεγχοντας την ορθότητά τους. Στη συνέχεια, αναλαμβάνει μέσω υπηρεσιών ασφαλείας, την εγγραφή στον Εξυπηρετητή Αυθεντικοποίησης Χρηστών (Firebase), την έκδοση πιστοποιητικού από τον ίδιο εξυπηρετητή και τέλος την ασφαλή ανταλλαγή κλειδιού για την κρυπτογραφημένη επικοινωνία με τον εξυπηρετητή του συστήματος. Τα απαραίτητα δεδομένα του χρήστη, πέρα από την αποστολή τους στον εξυπηρετητή, αποθηκεύονται και σε σχετική μη σχεσιακή βάση των χρηστών της εφαρμογής.

- **Υπηρεσία Σύνδεσης Χρήστη**

  Η Υπηρεσία Σύνδεσης Χρήστη πραγματοποιεί αρχικά την ανάγνωση των δεδομένων εισόδου από την οθόνη σύνδεσης της εφαρμογής και σε επόμενο στάδιο την αυθεντικοποίηση μέσω του Εξυπηρετητή Αυθεντικοποίησης Χρηστών. Στη συνέχεια υλοποιεί την ενημέρωση του εξυπηρετητή του συστήματος για την σύνδεση του συγκεκριμένου χρήστη. Για τις παραπάνω διεργασίες, η υπηρεσία χρειάζεται πρόσβαση και στα δεδομένα του χρήστη που διατηρούνται στη βάση χρηστών της εφαρμογής.

- **Διεπαφή Εγγραφής Χρήστη**

  Πρόκειται για διεπαφή στο γραφικό περιβάλλον της εφαρμογής, μέσω της οποίας ο χρήστης ζητάει την εκτέλεση λειτουργιών που σχετίζονται με την εγγραφή νέου χρήστη. Η διεπαφή αυτή απευθύνεται στην Υπηρεσία Εγγραφής Χρήστη για να φέρει εις πέρας τις λειτουργίες που απαιτούνται.

- **Διεπαφή Σύνδεσης Χρήστη**

  Η Διεπαφή Σύνδεσης Χρήστη χρησιμοποιείται για να πραγματοποιήσει σύνδεση στην εφαρμογή. Η διεπαφή αυτή συνεργάζεται με την Υπηρεσία Σύνδεσης Χρήστη για να υλοποιηθεί το αίτημα του χρήστη.

- **Διεπαφή Προφίλ Χρήστη**

  Η Διεπαφή Προφίλ Χρήστη αλληλεπιδρά με τον χρήστη μέσω της αντίστοιχης οθόνης της εφαρμογής και φροντίζει να χρησιμοποιεί κατάλληλες λειτουργίες για να διαχειριστεί αιτήματα του τελικού χρήστη σχετικά με το προφίλ του και τις γενικές προτιμήσεις του.

- **Διεπαφή Ιστορικού Χρήστη**

  Πρόκειται για μια διεπαφή στην οθόνη παρακολούθησης των δεικτών ευημερίας του χρήστη. Μέσω αυτής, παρουσιάζονται ολοκληρωμένες περιλήψεις προηγούμενων ημερών. Η ίδια η οθόνη ανανεώνεται από την Υπηρεσία Αξιοποίησης Ιστορικού Περιλήψεων και την Υπηρεσία Διαχείρισης Στόχων του κατώτερου στρώματος της εφαρμογής.

- **Διεπαφή Τρέχουσας Ημέρας Χρήστη**

  Παρομοίως, η διεπαφή αυτή ανήκει στην οθόνη παρακολούθησης δεικτών ευημερίας και επικοινωνεί με την Υπηρεσία Λήψης-Διαχείρισης Περίληψης Τρέχουσας Ημέρας και την Υπηρεσία Διαχείρισης Στόχων. Σκοπός της είναι να παρουσιάσει στον χρήστη, εφόσον εκείνος το επιλέξει, τα συγκεντρωτικά αποτελέσματα της μέρας σχετικά με τις μετρικές υγείας που καταγράφονται.

- **Διεπαφή Περιλήψεων/ Προειδοποιήσεων-Προτάσεων Χρήστη**

  Ακόμη μία διεπαφή σχετική με την οθόνη παρακολούθησης, είναι η Διεπαφή Περιλήψεων/ Προειδοποιήσεων-Προτάσεων Χρήστη. Μέσω αυτής γίνεται παρουσίαση του ιστορικού των προειδοποιήσεων, των περιλήψεων κατάστασης ή δραστηριότητας και των εβδομαδιαίων περιλήψεων μετρικών ευημερίας. Για την εμφάνιση αυτής της πληροφορίας η διεπαφή χρησιμοποιεί τις Υπηρεσίες Αξιοποίησης Προειδοποιήσεων-Προτάσεων, Αξιοποίησης Περιλήψεων Κατάστασης και Περιλήψεων Εβδομάδας.

- **Διεπαφή Στόχων Χρήστη**

  Η Διεπαφή Στόχων Χρήστη αλληλεπιδρά με τον χρήστη σχετικά με τους στόχους που ο ίδιος θέτει στη αντίστοιχη οθόνη. Για την εκτέλεση σχετικών με τους στόχους λειτουργιών, η διεπαφή απευθύνεται στην Υπηρεσία Διαχείρισης Στόχων.

- **Υπηρεσία Έκδοσης Πιστοποιητικού Χρήστη**

  Αυτή η υπηρεσία ανήκει στις Υπηρεσίες Ασφαλείας και αναλαμβάνει να εκδόσει ένα πιστοποιητικό για τον εκάστοτε χρήστη, καλώντας κατάλληλη λειτουργία του Εξυπρετητή Αυθεντικοποίησης Χρηστών (Firebase).

- **Υπηρεσία Αυθενικοποίησης Χρήστη**

  Η Υπηρεσία Αυθεντικοποίησης χρήστη που είναι και αυτή μια από τις υπηρεσίες ασφαλείας χρησιμοποιεί κατάλληλες λειτουργίες του Εξυπηρετητή Αυθεντικοποίσης προκειμένου να επιβεβαιώσει τα στοιχεία εισόδου του χρήστη στην εφαρμογή κατά την σύνδεσή του.

- **Υπηρεσία Ανταλλαγής Κλειδιού**

  Είναι ακόμα μία υπηρεσία ασφάλειας που παρέχει υλοποίηση λειτουργιών που σχετίζονται με την ασφαλή ανταλλαγή κλειδιού μεταξύ εφαρμογής και εξυπηρετητή με σκοπό την χρήση του κατά την κρυπτογραφημένη επικοινωνία.

- **Υπηρεσία Κρυπτογράφησης**

  Η συγκεκριμένη υπηρεσία ασφαλείας αναλαμβάνει την κρυπτογράφηση δεδομένων που εξέρχονται από την εφαρμογή και την αποκρυπτογράφηση δεδομένων που εισέρχονται σε αυτήν.

Κεφάλαιο 5

Υλοποίηση συστήματος

Στο Κεφάλαιο αυτό θα εστιάσουμε στον τρόπο με τον οποίο υλοποιήθηκαν τόσο τα προσομοιωτικά δεδομένα, όσο και το σύστημα (εξυπηρετητής, εφαρμογή) συνολικά. Σε πρώτη φάση, δηλαδή, θα αναλύσουμε τον τρόπο με τον οποίο η Γεννήτρια Δεδομένων παράγει τα παροσομοιωτικά δεδομένα ενός χρήστη, ενώ σε δεύτερη φάση, θα μελετήσουμε λεπτομερώς την υλοποίηση των υπηρεσιών του εξυπηρετητή του συστήματος και της εφαρμογής κινητών τηλεφώνων.

5.1 Γεννήτρια Δεδομένων

Αρχικά, είναι σημαντικό να εξετάσουμε τον τρόπο με τον οποίο κατασκευάζονται τα δεδομένα προσωμοίωσης του χρήστη, καθώς αυτά αποτελούν την είσοδο στο σύστημα μας. Θα ξεκινήσουμε από την επιλογή των πηγών δεδομένων που χρησιμοποιήθηκαν και των δεδομένων υγείας που τελικά καταγράφονται από αυτές. Στη συνέχεια, θα παρουσιαστεί ο αλγόριθμος που κατασκευάστηκε για να προσομοιώσει κατάλληλα την ημέρα του χρήστη και τέλος θα γίνει μελέτη των επιμέρους αλγορίθμων που χρειάστηκαν για να παραχθούν αληθοφανείς τιμές για τους διάφορους δείκτες ευημερίας του ατόμου.

5.1.1 Επιλογή πηγών δεδομένων και μετρικών

Έχει γίνει αναφορά και στο κεφάλαιο της ανάλυσης ότι η Γεννήτρια Δεδομένων θα υλοποιηθεί με τη χρήση ενός **εξυπηρετητή**, ο οποίος **παίζει τον ρόλο όλων των πηγών δεδομένων** που καταγράφουν δείκτες ευημερίας του ατόμου. Επίσης στο ίδιο κεφάλαιο εξετάστηκαν οι πηγές δεδομένων που τελικά επιλέχθηκαν για την συλλογή των ακατέργαστων δεδομένων.

Συγκεκριμένα, οι πηγές που χρησιμοποιήθηκαν για την δημιουργία δεδομένων είναι η ενδυτή συσκεύη **Fitbit** που φοριέται στο καρπό του χεριού, η εφαρμογή κινητών τηλεφώνων **UnderArmour** και οι αισθητήρες στο έξυπνο σπίτι του χρήστη που καταγράφουν αν κάθεται στον καναπέ (couch sensor) και αν η τηλεόραση είναι ανοικτή (tv sensor).

Η επιλογή των δύο πρώτων έγινε, διότι είναι διαδεδομένες πηγές καταγραφής τέτοιων δεδομένων, που μελετάνε ένα μεγάλο εύρος μετρικών της καθημερινότητας του ατόμου. Επιπλέον, διαθέτουν **ανοικτές διεπαφές προγραμματισμού εφαρμογών**, δηλαδή

φανερά στο κοινό APIs (Application Programming Interface), μέσω των οποίων μπορούμε να ακολουθήσουμε την μορφή με την οποία αναπαριστώνται τα δεδομένα. Σημειώνεται ότι και οι δύο αυτές πηγές επιλέγουν **αναπαράσταση σε JSON** με συγκεκριμένα πεδία η κάθε μια. Έτσι, πετυχαίνουμε την πιστή αναπαράσταση ενός πραγματικού σεναρίου, παρόλο που τα δεδομένα είναι προσομοιωτικά.

Επιπροσθέτως, η επιλογή των δύο αισθητήρων έγινε, καθώς πρόκειται για αισθητήρες που θα μπορούσε να βρει κάποιος σε ένα έξυπνο σπίτι. Βεβαίως στη παρούσα φάση, δεν γίνεται ιδιαίτερη χρήση τέτοιων αισθητήρων. Ωστόσο, επιλέξαμε να καταγράψουμε και τέτοιου είδους δεδομένα, προκειμένου να μελετήσουμε τη συμπεριφορά του συστήματος μας σε ένα μελλοντικό σενάριο που το σπίτι του ατόμου θα μετατραπεί σε ένα έξυπνο σπίτι. Σαν αποτέλεσμα, στα πλαίσια του Διαδικτύου των Πραγμάτων, δεδομένα από τέτοιου είδους αισθητήρες θα ήταν άξια μελέτης από το σύστημά μας. Τέλος, σημειώνεται ότι δεν βρέθηκε η μορφή αναπαράστασης αυτών των δεδομένων από κάποιον αισθητήρα πίεσης στον καναπέ ή από κάποιον αισθητήρα λειτουργίας της τηλεόρασης. Επομένως, επιλέχθηκε να υλοποιήσουμε από μόνοι μας τον τρόπο αναπαράστασης αυτών των δεδομένων. Φυσικά, αυτό μπορεί να γίνει χωρίς βλάβη της γενικότητας, γιατί πρόκειται για πολύ απλού τύπου δεδομένα. Γι΄ αυτό το λόγο, δεν έχει ιδιαίτερη σημασία να αντιγράψουμε επακριβώς τον τρόπο αναπαράστασης τους από σχετικούς αισθητήρες.

Έχοντας δικαιολογήσει την επιλογή των πηγών δεδομένων στη προσομοίωση ενός πραγματικού σεναρίου, αξίζει να αναφερθεί κανείς σε αυτό το σημείο, στις παραδοχές που έγιναν σχετικά με το είδος των δεδομένων που κατασκευάζονται από την Γεννήτρια Δεδομένων. Αρχικά, από τις διεπαφές προγραμματισμού εφαρμογών (APIs) τόσο του Fitbit όσο και του UnderArmour, γίνεται γνωστό το πώς αναπαριστώνται σε JSON τα τελικά συμπεράσματα κατάστασης ή δραστηριότητας του ατόμου, και όχι με ποιό τρόπο καταγράφονται σε πραγματικό χρόνο οι τιμές κάθε μετρικής. Επομένως, αυτό που μας επιτρέπουν να συλλέξουμε είναι συγκεντρωτικά αποτελέσματα μιας κατάστασης του ατόμου που έχει ολοκληρωθεί. Για τον συγκεκριμένο λόγο, έγινε η επιλογή να κατασκευάζονται για κάθε κατάσταση ή δραστηριότητα του χρήστη **ένα JSON τη στιγμή που αρχίζει την δραστηριότητα** με null τιμές στα σχετικά πεδία και **ένα τη στιγμή που τελειώνει την δραστηριότητα** με τις τελικές τιμές των μετρικών.

Μία ακόμη παραδοχή για την προσομοίωση των δεδομένων σχετίζεται με το εύρος των αθλητικών δραστηριοτήτων που θεωρούμε ότι κάνει το άτομο. Ειδικότερα, η συσκευή Fitbit και η εφαρμογή UnderArmour έχουν την δυνατότητα καταγραφής πολλών και διαφορετικών αθλητικών δραστηριοτήτων. Ωστόσο, για τις ανάγκες της εργασίας, επιλέγουμε χωρίς βλάβη της γενικότητας να θεωρήσουμε ότι οι δραστηριότητες που κάνει ο χρήστης μας είναι συγκεκριμένες. Θεωρούμε, ειδικότερα ότι περιορίζονται στο **περπάτημα**, το **τρέξιμο** και τη **ποδηλασία**. Έτσι, θα φτιάξουμε δεδομένα μόνο για αυτές τις δραστηριότητες, έχοντας γνώση ότι πιθανή προσθήκη κάποιας επιπλέον δραστηριότητας θα γινόταν με ανάλογο τρόπο.

Επιπλέον, είναι γνωστό ότι αυτές οι πηγές δεδομένων καταγράφουν τεράστιο όγκο δεδομένων ανά δευτερόλεπτο, ειδικά όταν πρόκειται για μετρικές ζωτικής σημασίας του ατόμου όπως ο καρδιακός παλμός. Στο σύστημά μας ωστόσο, επειδή θέλουμε να μελετήσουμε την προτυποποίηση των δεδομένων και όχι την διαχείρηση του όγκου τους, δεν θα δημιουργούμε τόσες πολλές τιμές ανά δευτερόλεπτο για τέτοιες μετρικές. **Θα περιορίσουμε**, δηλαδή,

σημαντικά τον όγκο των δεδομένων διατηρώντας όμως την μορφή τους, προκειμένου να μελετήσουμε τον τρόπο διαχείρησης τους από το σύστημα.

Έχοντας υπόψιν μας τις παραπάνω παραδοχές καθόλη την υλοποίηση της γεννήτριας δεδομένων είναι σημαντικό σε αυτό το σημείο να γίνει λεπτομερής περιγραφή των δραστηριοτήτων και των δεικτών ευημερίας που καταγράφει κάθε μία πηγή δεδομένων. Έχουμε, λοιπόν ανά πηγή δεδομένων τα εξής:

- **Fitbit**

  Από δραστηριότητες ή καταστάσεις του χρήστη, επιλέγουμε την καταγραφή της κατάστασης του ύπνου και της ξεκούρασης συμπεριλαμβάνοντας στην τελευταία κάθε κατάστασης, κατά την οποία ο χρήστη κάθεται, είτε βρίσκεται σπίτι του, είτε αλλού (π.χ. στον εργασιακό του χώρο). Σημειώνεται ότι από τη διεπαφή προγραμματισμού εφαρμογών (API) του Fitbit δεν είναι εμφανής η δυνατότητα καταγραφής της ξεκούρασης, αλλά φαίνεται ότι μπορεί να μετρήσει το χρόνο που ο χρήστης κάθεται μέσα στη μέρα. Επιπλέον, υπάρχει μια συγκεκριμένη δομή αναπαράστασης δεδομένων για κάθε είδους δραστηριότητα, οπότε μπορούμε να θεωρήσουμε για τις ανάγκες της εργασίας ότι η κατάσταση, που ο χρήστης κάθεται, ακολουθεί αυτή τη δομή σαν ένα είδος δραστηριότητας. Επίσης, από άλλες δραστηριότητες καταγράφονται το περπάτημα, το τρέξιμο και η ποδηλασία. Για τα παραπάνω δημιουργούνται δεδομένα σχετικά με δείκτες που μετράει το Fitbit, οι οποίοι είναι γνωστοί μέσα από τις διεπαφές προγραμματισμού εφαρμογών (APIs) που παρέχονται. Επιπλέον, όσον αφορά τις μετρικές του σώματος που είναι πιο γενικές και δεν σχετίζονται άμεσα με την εκάστοτε δραστηριότητα, επιλέγουμε την μέτρηση του καρδιακού παλμού, του φαγητού και του νερού που καταναλώνει ο χρήστης και τέλος του βάρους του. Φυσικά, ο τρόπος με τον οποίο μετρώνται οι παραπάνω δείκτες από το Fitbit δεν είναι γνωστός, ωστόσο για τους σκοπούς της διπλωματικής αυτό που αρκεί να γνωρίζουμε είναι πρώτον η δυνατότητα της συσκευής να καταγράφει αυτές τις μετρικές και δεύτερον η μορφή με την οποία τις αναπαριστά. Και οι δύο αυτές απαιτήσεις ικανοποιούνται σε αυτή τη περίπτωση, οπότε η προσομοίωση που θέλουμε να πετύχουμε για τα δεδομένα θα είναι αξιόπιστη.

- **UnderArmour**

  Σχετικά με τη συγκεκριμένη εφαρμογή κινητών τηλεφώνων έχουμε μελετήσει ότι υποστηρίζεται η καταγραφή της κατάστασης του ύπνου και των δραστηριοτήτων του περπατήματος, του τρεξίματος και της ποδηλασίας. Μετρώνται, αντίστοιχα, συγκεκριμένοι δείκτες σχετικοί με την εκάστοτε κατάσταση του χρήστη. Επιπλέον από πιο γενικούς δείκτες που σχετίζονται με το σώμα του χρήστη επιλέξαμε την μέτρηση του καρδιακού παλμού.

  Παρατηρούμε, λοιπόν, αμέσως ότι υπάρχουν από τη πρώτη στιγμή κάποιες διαφοροποιήσεις στο είδος των μετρικών που επιλέξαμε να καταγράφονται από τις δύο παραπάνω πηγές δεδομένων. Στη συνέχεια, θα παρατηρήσουμε και άλλες διαφορές στις μετρικές που μετράει κάθε συσκευή για την ίδια δραστηριότητα, καθώς επίσης και διαφορές στις μονάδες μέτρησης που χρησιμοποιούνται. Όλες αυτές οι διαφορές οδηγούν σε μια καλή προσέγγιση του πραγματικού σεναρίου, καθότι από την πληθώρα των συσκευών που υπάρχουν, κάθεμια μετράει συγκεκριμένους δείκτες ευημερίας του ατόμου με συγκεκριμένο τρόπο.

- **Couch sensor**

  Ο αισθητήτας πίεσης στον καναπέ, μετράει πολύ απλά το αν ο χρήστης κάθεται στον καναπέ του ή όχι. Σε αυτό το σημείο σημειώνεται ότι όταν ο αισθητήρας δείχνει ότι κάποιος κάθεται στον καναπέ, θεωρούμε ότι αυτός είναι ο ίδιος ο χρήστης και δεν υπάρχει πιθανότητα να είναι κάποιος άλλος. Προφανώς σε ένα πραγματικό σενάριο έξυπνου σπιτιού, υπάρχει πιθανότητα να κάθονται πολλά άτομα στον ίδιο καναπέ, όμως εμείς για λόγους ευκολίας κάναμε την παραπάνω παραδοχή.

- **TV sensor**

  Ομοίως, ο αισθητήρας λειτουργίας της τηλεόρασης δείχνει αν η τηλεόραση είναι ανοικτή ή όχι. Εδώ έχουμε συνυπολογίσει την περίπτωση η τηλεόραση να είναι σε λειτουργία, αλλά ο χρήστης να μην είναι εκεί. Ειδικότερα, αν ο χρήστης κάθεται στον καναπέ και η τηλεόραση είναι ανοικτή, τότε έχουμε θεωρήσει πως ο χρήστης παρακολουθεί τηλεόραση. Και πάλι, αυτό δεν είναι κάτι απόλυτο στη πραγματικότητα, αλλά η προαναφερθείσα παραδοχή έγινε για λόγους ευκολίας προκειμένου να δουμε την συμπεριφορά του συστήματος σε ένα τέτοιο σενάριο, όπου θα γνωρίζαμε επακριβώς τι κάνει ο χρήστης σε αυτές τις ειδικές περιπτώσεις.

Ολοκληρώνοντας, αφού έχουμε μελετήσει το είδος των καταστάσεων και δραστηριοτήτων που καταγράφονται για τον χρήστη καθώς και τις μετρικές που σχετίζονται γενικότερα με το σώμα του, θα ακολουθήσει η παρουσίαση του αλγορίθμου που χρησιμοποιήθηκε για να προσομοιώθει η μέρα του χρήστη. Σε επόμενη ενότητα, θα αναφερθούμε πιο ειδικά στα JSON που δημιουργούνται από την Γεννήτρια Δεδομένων και στα αντίστοιχα πεδία τους, βασισμένοι στη μορφή αναπαράστασης των δεδομένων από τις πηγές που επιλέξαμε.

5.1.2 Αλγόριθμος προσομοίωσης ημέρας χρήστη

Σε αυτήν την ενότητα, λοιπόν, θα μελετηθεί ο τρόπος με τον οποίο φτιάχνονται τα δεδομένα για έναν συγκεκριμένο χρήστη, ο οποίος θα λέγαμε ότι συμπεριφέρεται λίγο πολύ σαν έναν **μέσο χρήστη**. Με άλλα λόγια, θέλουμε να προσομοιώσουμε την καθημερινή του δραστηριότητα με βάση την καθημερινότητα ενός μέσου ατόμου, που για παράδειγμα κοιμάται και ξυπνάει συνηθισμένες ώρες, ή επίσης τρώει, πίνει νερό ή αθλείται και πάλι σε λογικές ώρες μέσα στη μέρα.

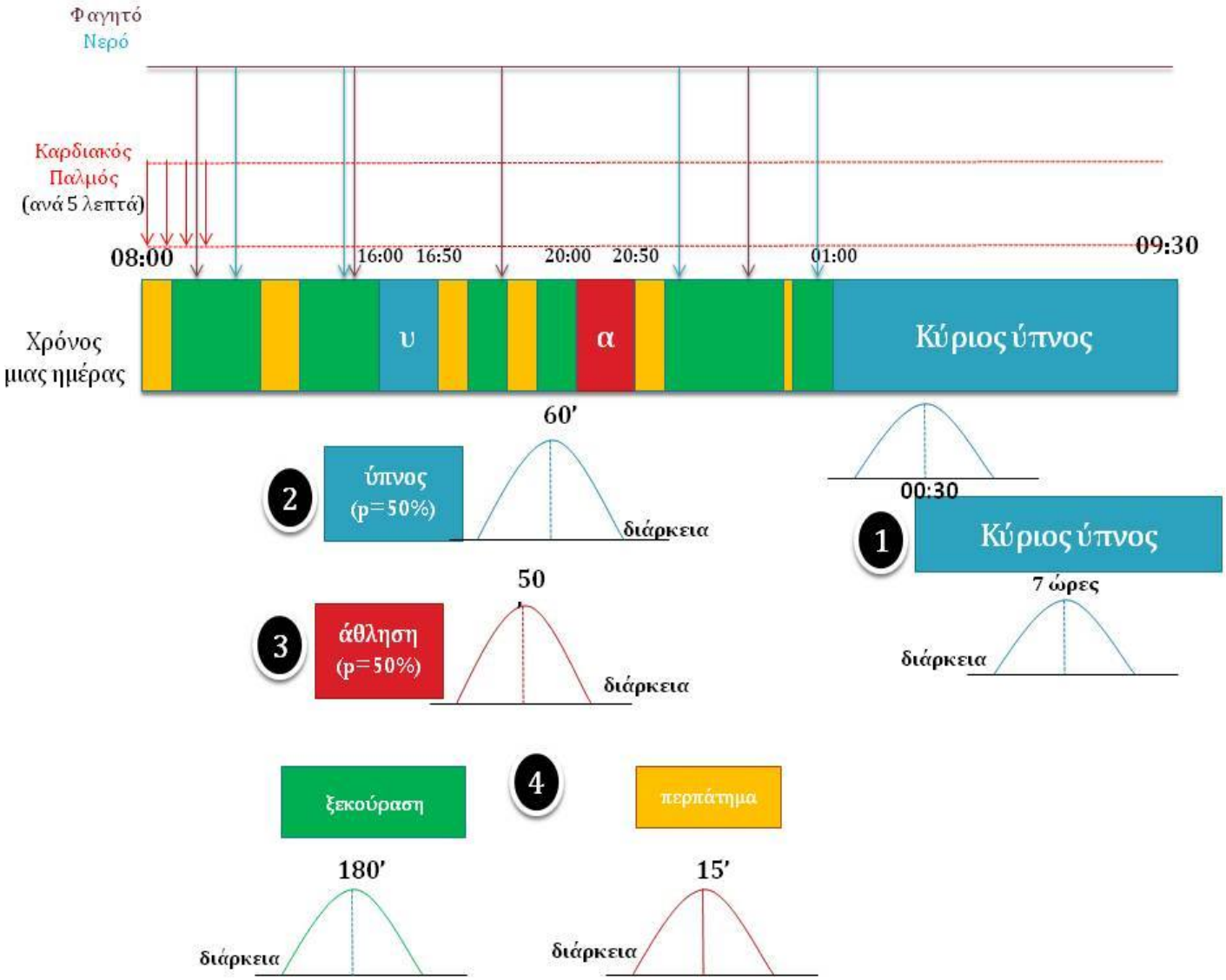
Κάνοντας μια πρώτη προσέγγιση του αλγορίθμου που θα χρησιμοποιηθεί, μπορεί να σκεφτεί κάποιος ότι υπάρχουν συγκεκριμένες καταστάσεις μέσα σε μια μέρα, από τις οποίες ένας μέσος άνθρωπος θα περάσει σε συγκεκριμένες ώρες. Για παράδειγμα, γνωρίζουμε με ασφάλεια ότι ο χρήστης θα πάει για βραδινό ύπνο (**κύριο ύπνο** δηλαδή) και μάλιστα μπορούμε να θεωρήσουμε ότι αυτό θα το κάνει μέσα σε ένα συγκεκριμένο χρονικό πλαίσιο της μέρας. Στη συνέχεια, γνωρίζουμε ότι θα κοιμηθεί για ένα λογικό χρονικό διάστημα και ότι θα ξυπνήσει μέσα σε συγκεκριμένες πρωινές ώρες. Με βάση τα παραπάνω, είναι λογικό να ξεκινήσουμε καθορίζοντας τον κύριο ύπνο του ατόμου, ο οποίος θα καθορίσει με τη σειρά του και την ημέρα του ατόμου. Με άλλα λόγια, θα θεωρήσουμε ότι μια ημέρα του χρήστη ορίζεται από την στιγμή που ξύπνησε μέχρι την στιγμή που θα τελείωσει ο κύριος ύπνος του την άλλη μέρα το πρωί.

Σε επόμενο στάδιο, μπορούμε να συμπληρώσουμε την υπόλοιπη μέρα του ατόμου. Οι καταστάσεις οι οποίες μπορούν να ενσωματωθούν στη μέρα είναι κάποιος **μικρός ύπνος**, μια **αθλητική δραστηριότητα** (εννοώντας τρέξιμο ή ποδηλασία) ή καταστάσεις που ο χρήστης **κάθεται** ή **περπατάει**. Χωρίς βλάβη της γενικότητας και για λόγους ευκολίας επίσης, θεωρούμε με ασφάλεια ότι ο μέσος χρήστης μας θα πάει για έναν μικρό ύπνο το πολύ μία φορά μέσα στη μέρα, ενώ αντίστοιχα για αθλητική δραστηριότητα πάλι το πολύ μια φορά μέσα σε λογικές ώρες της μέρας. Φυσικά, δίνεται και πιθανότητα να μην κοιμηθεί κι άλλο, πέρα από τον κύριο ύπνο, ή αντίστοιχα να μην αθληθεί καθόλου. Βέβαια, αν κάνει κάτι από τα δυο ή και τα δυο, θεωρούμε ότι θα γίνουν σε ώρες που αντιστοιχούν στη καθημερινότητα ενός μέσου ατόμου. Για παράδειγμα, ένας μικρός ύπνος συμβαίνει συνήθως το μεσημέρι, δηλαδή περίπου στο μέσο του χρόνου που ο χρήστης είναι ξύπνιος. Κατ΄ αντιστοιχία, μια αθλητική δραστηριότητα θα λαμβάνει χώρα, είτε στο πρώτο τέταρτο, είτε στο τρίτο τέταρτο της χρονικής διάρκειας που ο χρήστης είναι ξύπνιος.

Ορίζοντας με αυτό το τρόπο, πιθανές εισαγωγές δεύτερου ύπνου ή αθλητικής δραστηριότητας (τρέξιμο, ποδήλατο) μέσα στη μέρα του χρήστη, μένει τώρα να δούμε πώς θα γεμίσει η υπόλοιπη μέρα του ατόμου. Όπως αναφέραμε και πιο πάνω είναι λογικό όλη η υπόλοιπη μέρα του χρήστη να είναι είτε καταστάσεις κατά τις οποίες κάθεται είτε καταστάσεις στις οποίες περπατάει. Η κατάσταση ξεκούρασης και το περπάτημα, επομένως, θα εναλλάσσονται μέσα στη μέρα προκειμένου να γεμίσουν τα κενά διαστήματα που δεν έχουν συμπληρωθεί προηγουμένως.

Σαν αποτέλεσμα, έχουμε πετύχει μια λογική προσομοίωση καταστάσεων και δραστηριοτήτων ενός μέσου ατόμου μέσα σε μια μέρα. Πάνω σε αυτή τη μέρα, **είναι αναγκαίο να προσθέσουμε καταγραφές για το καρδιακό παλμό, το φαγητό, το νερό και το βάρος του χρήστη**, καθώς είναι δείκτες που έχουν επιλεγεί για μέτρηση από τις πηγές δεδομένων που προσομοιώνει η Γεννήτρια Δεδομένων. Επιλέγουμε επομένως, την καταγραφή του καρδιακού παλμού κάθε 5 λεπτά και φαγητού ή νερού του χρήστη, κατά τη διάρκεια που εκείνος κάθεται (κατάσταση ξεκούρασης), και εφόσον η ώρα είναι λογική, ή εφόσον έχει περάσει αρκετό διάστημα από την τελευταία φορά που έφαγε ή ήπιε νερό. Τέλος, θεωρούμε ότι κάθε μέρα, παίρνουμε μέτρηση για το βάρος του χρήστη, λίγο πριν αυτός κοιμηθεί για βράδυ.

Συνοψίζοντας, παρατίθεται στη συνέχεια ο αλγόριθμος, γραμμένος σε **ψευδογλώσσα**, που υλοποιεί όλα τα παραπάνω, όπως επίσης και ένα σχήμα για την καλύτερη κατανόηση του αλγορίθμου. Βεβαίως, μελετώντας τον αλγόριθμο μπορεί να παρατηρήσει κανείς το πώς επιλέγονται οι ώρες κάθε κατάστασης του χρήστη πιθανοτικά. Ο λόγος που επιθυμούμε την επιλογή κάθε κατάστασης βάσει πιθανοτήτων, είναι διότι στην καθημερινή δραστηριότητα ενός μέσου ατόμου, υπάρχουν διακυμάνσεις από μέρα σε μέρα, τις οποίες θέλουμε να συμπεριλάβουμε στην προσομοίωση. Ειδικότερα, χρησιμοποιούμε **κανονική κατανομή** (κατανομή **Gauss**) με συγκεκριμένη κάθε φορά μέση τιμή και απόκλιση που εξαρτώνται από την καθημερινότητα ενός μέσου χρήστη. Η χρήση κανονικής κατανομής δικαιολογείται εύκολα, αν σκεφτεί κανείς ότι **προσεγγίζει κατά πολύ τις διακυμάνσεις** της χρονικής στιγμής, κατά την οποία ο χρήστης, κάνει κάτι μέσα στη μέρα. Με άλλα λόγια, σχεδόν πάντα υπάρχει μία μέση ώρα για την έναρξη μιας συγκεκριμένης κατάστασης/δραστηριότητας του ατόμου και καθώς απομακρυνόμαστε από αυτή την ώρα, μειώνεται **ομοιόμορφα** η πιθανότητα να επιλεγεί σαν την ώρα έναρξης της εκάστοτε κατάστασης/δραστηριότητας.



Σχήμα 5.1: Αλγόριθμος Προσομοίωσης ημέρας χρήστη

Στις επόμενες σελίδες ακολουθούν το σχήμα που συνοψίζει τα παραπάνω(Σχήμα 5.1), καθώς και ο αλγόριθμος που αναπτύχθηκε γραμμένος σε ψευδογλώσσα.

```
1
2   //**** DATA GENERATOR ALGORITHM ****//
3
4  // Define start_date and end_date of the simulation
5  startDate = <start_date>;
6  endDate   = <end_date>;
7
8  //Define wakeUp time for the first day
9  wakeUpTimestamp = <wakeUpTime>;    // 08:00
10
11
12 FOR day = (from: startDate, until: endDate)
13 {
14   //Define Main Sleep
15   mainSleep_startTime = random_Gaussian('avg'= 00:30, 'dev'= 2h, 'upLimit'= 02:00, '
       downLimit'= 22:00);
16   mainSleep_duration  = random_Gaussian('avg'= 7h,    'dev'= 1h, 'upLimit'= 9h,    '
       downLimit'= 6h);
17   nextWakeUp_time   = goToBed_time + mainSleep_duration;
18
19   //Define Nap if needed
20   nap_selected =  random(50% true, 50% false);
21   IF (nap_selected == true)
22   {
23     nap_startTime   = random_Gaussian('avg'= middleOf(wakeUpTimestamp, mainSleep_startTime
         ),  'dev'=1h);
24     nap_duration  = random_Gaussian('avg'=50mins, 'dev'= 30mins);
25     nap_endTime   = nap_startTime + nap_duration;
26   }
27
28   //Define Workout IF needed
29   wo_selected =  random(50% true, 50% false);
30   IF (wo_selected == true)
31   {
32
33     wo_startTime  = random_Gaussian('avg'= firstOrThirdQuarterOf(wakeUpTimestamp,
         mainSleep_startTime),  'dev'= 1h);
34     wo_duration   = random_Gaussian('avg'= 30mins, 'dev'= 10mins);
35     wo_endTime    = wo_startTime + wo_duration;
36   }
37
38   //main loop of day
39   state = null; prevState = null;
40
41   FOR time = (from:wakeUpTimestamp, until:nextWakeUp_time)
42   {
43
44     IF (time MOD '5mins' == 0)    //every 5 minutes generate heart rate data.
45       createHeartRate_JSON();
46
47     IF (state=="nap")
48     {
49       IF (time==nap_endTime)
50       {
51         createNap_finalJSON();
```

```
        state      = null;
        prevState    = "nap";
      }
    }
    ELSE IF (state=="workout")
    {
      IF (time==wo_endTime)
      {
        createWO_finalJSON();
        state      = null;
        prevState    = "workout";
      }
    }
    ELSE IF (state=="mainSleep")
    {
      IF (time==nextWakeUp_time)
      {
        createMainSleep_finalJSON();
        state      = null;
        prevState    = "mainSleep";
      }
    }
    ELSE IF (state=="walking")
    {
      IF (time == state_endTime)
      {
        createWalking_finalJSON();
        state      = null;
        prevState = "walking";
      }
    }
    ELSE IF (state=="rest")
    {
      IF (time == state_endTime)
      {
        createRest_finalJSON();
        state      = null;
        prevState = "rest";
      }
      ELSE{
        //check last food and water log, and create a new based on a probability.
        IF (time == a long time after lastFood_time)
        {
          createFood_JSON();
          lastFood_time = time;
        }
        IF (time == a long time after lastWater_time)
        {
          createWater_JSON();
          lastWater_time = time;
        }
      }
    }
    ELSE
    {
```

```
      IF (time==nap_startTime)
      {
        createNap_nullJSON();
        state = "nap";
      }
      ELSE IF (time==wo_startTime)
      {
        createWO_nullJSON();
        state = "workout";
      }
      ELSE IF (time==mainSleep_startTime)
      {
        createMainSleep_nullJSON();
        state = "mainSleep";
      }
      ELSE
      {
        IF (prevState=="walking")
        {
          createRest_nullJSON();
          state = "rest";
          state_duration  = random_Gaussian('avg'=3h, 'dev'= 1h, upLimit= 6h, downLimit= 5
              mins);
          state_endTime   = time + state_duration;
        }
        ELSE IF (prevState=="rest")
        {
          createWalking_nullJSON();
          state = "walking";
          state_duration  = random_Gaussian('avg'=15mins, 'dev'= 10mins, upLimit= 30mins,
               downLimit= 5mins);
          state_endTime   = time + state_duration;
        }
        ELSE
        {
          // create state(rest or walking) exactly like above.
          createRandomly_WalkingOrRestState(50% walking, 50% rest);
        }
      }
    }
  }
  // with 'a while before main sleep' timestamp create weight data.
  createWeight_JSON();

  //define wakeUp timestamp for the next day.
  wakeUpTimestamp = nextWakeUp_time;
}
```

Data Generator Algorithnm in pseudocode

Με την ολοκλήρωση αυτής της ενότητας, έχουμε γνώση του τρόπου με τον οποίο προσομοιώνεται η μέρα του ατόμου. Είναι επιτακτική η ανάγκη να παρουσιαστεί και η δομή των JSON της κάθε πηγής δεδομένων, προκειμένου να κατανοήσουμε πώς ο αλγόριθμος φτιάχνει τελικά τα προσομοιωτικά δεδομένα.

5.1.3 Δομή των δεδομένων

Στη παρούσα ενότητα, θα εμβαθύνουμε στον τρόπο αναπαράστασης των δεδομένων. Προφανώς, η αναπαράσταση θα ακολουθεί τις δομές των JSON που χρησιμοποιεί κάθε πηγή δεδομένων. Αυτές οι δομές αντλήθηκαν από τα **APIs** του Fitbit[22] και του UnderArmour[23].

Για να έχουμε μια συνολική εικόνα, μπορούμε να δούμε τα JSON που δημιουργεί η Γεννήτρια Δεδομένων ανά κατάσταση/δραστηριότητα ή μετρική του σώματος. Στις περιπτώσεις που υπάρχουν πάνω από μια πηγή για την καταγραφή ίδιου τύπου δεδομένων, θα παρουσιαστεί η μορφή που χρησιμοποιεί κάθε μια ξεχωριστά, προκειμένου να δοθεί επιπλέον και η δυνατότητα σύγκρισης. Σημειώνεται επίσης, ότι από τα πεδία των JSON που χρησιμοποιούν οι εφαρμογές, μερικά δεν μας χρειάζονται για τους σκοπούς της διπλωματικής και γι΄ αυτό το λόγο κρατήσαμε αυτούσια μόνο εκείνα που μας αφορούν. Επομένως, ανά κατάσταση/δραστηριότητα του χρήστη έχουμε:

- **Κατάσταση ύπνου**

  Η κατάσταση του ύπνου, είτε πρόκειται για κύριο ύπνο είτε όχι, καταγράφεται και από το Fitbit και από το UnderArmour.

  Ξεκινώντας από το Fitbit παρατηρήσαμε ότι κάνει διαχωρισμό ανάμεσα στα δύο είδη ύπνου με αποτέλεσμα η δομή του JSON του ενός να είναι λίγο διαφορετική από του άλλου. Και τα δύο μετράνε την ώρα έναρξης και την διάρκεια του ύπνου, καθώς και δείκτες σχετικούς με το πώς κατανέμεται η συνολική διάρκεια του ύπνου.Για παράδειγμα, μετράται πόση ώρα χρειάστηκε για να αποκοιμηθεί ο χρήστης, πόση ώρα κοιμήθηκε τελικά, πόση ώρα παρέμεινε στο κρεβάτι πριν σηκωθεί και λοιπά. Τέλος, και οι δύο δομές των JSON περιλαμβάνουν τις φάσεις του ύπνου και τη διάρκεια της καθεμίας. Εδώ υπάρχει και η διαφοροποίηση των δύο δομών, αφού το Fitbit επιλέγει να ορίζει διαφορετικά της φάσεις του ύπνου ανάλογα με το αν είναι κύριος ή όχι. Παρατίθεται στη συνέχεια η δομή των δυο αυτών JSON:

”Main Sleep (Fitbit)”

```
{
  "dateOfSleep": <date>,
  "duration": <value in
      milliseconds>,
  "efficiency": <value>,
  "isMainSleep": true,
  "levels": {
    "summary": {
      "deep": {
        "count": <value>,
        "minutes": <value>
      },
      "light": {
        "count": <value>,
        "minutes": <value>
      },
      "rem": {
        "count": <value>,
        "minutes": <value>
      },
      "wake": {
        "count": <value>,
        "minutes": <value>
      }
        }
   },
  "logId": <log_id>,
  "minutesAfterWakeup": <
      value>,
  "minutesAsleep": <value>,
  "minutesAwake": <value>,
  "minutesToFallAsleep": <
      value>,
  "startTime": <dateTimestamp
      >,
  "timeInBed": <value in
      minutes>,
  "type": "stages"
}
```

”Nap (Fitbit)”

```
{
  "dateOfSleep": <date>,
  "duration": <value in milliseconds
      >,
  "efficiency": <value>,
  "isMainSleep": false,
  "levels": {
    "summary": {
      "asleep": {
        "count": <value>,
        "minutes": <value>
      },
      "awake": {
        "count": <value>,
        "minutes": <value>
      },
      "restless": {
        "count": <value>,
        "minutes": <value>
      }
    }
  },
  "logId": <log_id>,
  "minutesAfterWakeup": <value>,
  "minutesAsleep": <value>,
  "minutesAwake": <value>,
  "minutesToFallAsleep": <value>,
  "startTime": <dateTimestamp>,
  "timeInBed": <value in minutes>,
  "type": "classic"
}
```

Παρομοίως, το UnderArmour, που χρησιμοποιεί μία δομή για τον ύπνο, καταγράφει την αρχή και το τέλος του, καθώς και μετρικές, σχετικές με την κατανομή της διάρκειας του ύπνου του χρήστη. Επιπλέον, καταγράφει τις διάφορες φάσεις του ύπνου.

Η αναπαράσταση σε JSON φαίνεται παρακάτω:

"Sleep (UnderArmour)"

```
{
  "updated_datetime": <dateTimestamp>,
  "created_datetime": <dateTimestamp>,
  "aggregates": {
    "sum": <value_in_seconds>,
    "details": {
        "deep_sleep": {
            "sum": <value_in_seconds>
        },
        "awake": {
            "sum": <value_in_seconds>
        },
        "time_to_sleep": {
            "sum": <value_in_seconds>
        },
        "times_awakened": {
            "sum": <value>
        },
        "light_sleep": {
            "sum": <value_in_seconds>
        }
    }
  },
  "links": {
    "self": [
      {
      "href": "/v7.1/sleep/"+<log_id>+"/",
      "id": <log_id>
      }
    ]
  }
}
```

- **Κατάσταση ξεκούρασης**

  Πρόκειται για κάθε κατάσταση κατά την οποία το άτομο κάθεται. Με βάση τις πηγές δεδομένων που προσομοιώνουμε, θεωρούμε ότι η καταγραφή αυτής της κατάστασης γίνεται μόνο από το Fitbit σαν μια από τις δραστηριότητες, όπως εξηγήθηκε πιο πάνω. Γενικότερα, για κάθε δραστηριότητα, το Fitbit χρησιμοποιεί έναν μοναδικό αριθμό για να την προσδιορίζει. Γι΄ αυτό επιλέγουμε κι εμείς τον αριθμό 10000 για τον προσδιορισμό της κατάστασης της ξεκούρασης, έχοντας σιγουρευτεί ότι δε χρησιμοποιείται σαν ταυτοποιητικό από τις άλλες δραστηριότητες που καταγράφουμε. Με βάση λοιπόν αυτή τη δομή, πέρα από την ώρα έναρξης και την διάρκεια αυτής της κατάστασης, υπάρχει η δυνατότητα μέτρησης και των θερμιδών που έκαψε ο χρήστης. Έτσι, προσαρμόζοντας το JSON που χρησιμοποιείται και για τις άλλες δραστηριότητες του Fitbit φτιάξαμε την ακόλουθη δομή:

"Sedentary (Fitbit)"

```
{
  {
    "activityId":10000,
    "calories": <value_in_kcal>,
    "distance":0,
    "duration":<value_in_milliseconds>,
    "hasStartTime":true,
    "logId":<log_id>,
    "name":"Sedentary",
    "startTime": <dateTimestamp>,
    "steps":0
  }
}
```

- **Δραστηριότητα (περπάτημα, τρέξιμο, ποδηλασία)**

  Η κατηγοριοποίηση όλων των ειδών δραστηριότητας σε μια δομή γίνεται, διότι έχουμε μελετήσει ότι και η ενδυτή συσκευή Fitbit και η εφαρμογή UnderArmour χρησιμοποιούν μια συγκεκριμένη αναπαράσταση η καθεμιά για τα δεδομένα κάθε δραστηριότητας. Προφανώς έχουν σχετικά πεδία, όπου ορίζεται το είδος της δραστηριότητας, τόσο με έναν αριθμό-ταυτοποιητικό (activity_id), όσο και με το ίδιο το όνομα (activity_name) της δραστηριότητας. Σημειώνεται ότι ο μοναδικός αυτός αριθμός δεν είναι πάντα εμφανής από τα APIs και γι αυτό το λόγο, σε αρκετά σημεία επιλέξαμε έναν τυχαίο αριθμό για ταυτοποιητικό, χωρίς να υπάρχει βλάβη της γενικότητας. Όσον αφορά τα υπόλοιπα δεδομένα που συλλέγουν, οι δείκτες που καταγράφουν, πέρα από την ώρα έναρξης και την διάρκεια, είναι σχετικοί με σημαντικές μετρικές των δραστηριοτήτων, όπως οι θερμίδες που έκαψε το άτομο, η απόσταση που έκανε, η μέση ταχύτητα του, καθώς και τα βήματα που έκανε αν πρόκειται για περπάτημα ή τρέξιμο. Τα παραπάνω φαίνονται πιο αναλυτικά, στη δομή των JSON που παρουσιάζεται ακολούθως:

”Activity (Fitbit)”

```
{
"activityId": <activity_id>,
"calories": <value_in_kcal>,
"description": <description (
    avg_speed_in_mph)>,
"distance": <value_in_miles>,
"duration": <value_in_milliseconds>,
"logId": <log_id>,
"name": <activity_name>,
"startTime": <dateTimestamp>,
"steps": <value>
}
```

”Activity (UnderArmour)”

```
{
 "name": <activity_name>,
 "updated_datetime": <dateTimestamp>,
 "created_datetime": <dateTimestamp>,
 "_links":{
   "self":[
   {
    "href":"\/v7.1\/workout\/<log_id>\/
        ",
    "id":<log_id>
   }
   ],
   "activity_type": [{
     "href": "\/v7.1\/activity_type\/<
         activity_id>\/",
     "id": <activity_id>
   }]
 },
 "aggregates": {
   "active_time_total": <
       value_in_seconds>,
   "distance_total": <value_in_meters>,
   "steps_total": <value>,
   "speed_avg": <value_in_meters/second
       >,
   "elapsed_time_total": <
       value_in_seconds>,
   "metabolic_energy_total": <
       value_in_joules>
  }
}
```

Αντίστοιχα, κατασκευάζονται τα JSON για τις υπόλοιπες μετρικές του σώματος, που καταγράφονται ανεξάρτητα από την κατάσταση/δραστηριότητα του χρήστη. Ανά μετρική, επομένως, έχουμε:

- **Καρδιακός παλμός**

  Πρόκειται για μια μετρική ζωτικής σημασίας που καταγράφεται και από τις δύο κύριες πηγές δεδομένων της προσομοίωσης μας. Μέσω του Fitbit μπορούμε να πάρουμε τις τιμές του καρδιακού παλμού για ένα συγκεκριμένο χρονικό διάστημα που θα ζητήσουμε. Στο σενάριο μας, επειδή όπως αναφέρθηκε δεν θέλουμε πληθώρα τιμών ανά δευτερόλεπτο, θεωρούμε ότι παίρνουμε μία τιμή ανά 5 λεπτά. Σαν αποτέλεσμα η δομή του JSON για το Fitbit θα περιορίζεται στην χρονική στιγμή που έγινε η μέτρηση και στην τιμή του καρδιακού παλμού. Σχετικά με την εφαρμογή UnderArmour δεν υπήρξε η δυνατότητα να βρούμε τον ακριβή τρόπο που μπορούμε να πάρουμε την τιμή του καρδιακού παλμού μια δεδομένη χρονική στιγμή. Ωστόσο, γνωρίζουμε ότι η εφαρμογή αυτή μετράει όντως τον καρδιακό παλμό, με παρόμοιο τρόπο με το Fitbit, οπότε μπορούμε και πάλι χωρίς βλάβη της γενικότητας να χρησιμοποιήσουμε την ίδια δομή και για το UnderArmour. Προφανώς, οι μονάδες

που χρησιμοποιούνται είναι παλμοί ανά λεπτό και για τις δύο πηγές δεδομένων. Η δομή, επομένως, για τον καρδιακό παλμό είναι η ακόλουθη.

"Heart rate (Fitbit and UnderArmour)"

```
{
  "time": <dateTimestamp>,
  "value": <value in beats/min>
}
```

- **Φαγητό και νερό**

  Οι μετρήσεις αυτές πραγματοποιούνται μόνο από το Fitbit. Όπως και σε άλλες περιπτώσεις δεν θα ασχοληθούμε με τον τρόπο που η συσκευή συλλέγει αυτά τα δεδομένα. Μας αρκεί ότι μπορεί να τα καταγράψει, ακόμα και στη χειρότερη περίπτωση που ο χρήστης τα εισάγει από την ίδια την συσκευή. Επίσης, όσον αφορά το φαγητό υπάρχει και δυνατότητα καταγραφής επιμέρους στοιχείων πέρα από τις θερμίδες, όπως οι υδατάνθρακες, οι πρωτεΐνες και άλλα, που όμως δεν θα συμπεριληφθούν στην περίπτωση μας.

  Από την διεπαφή προγραμματισμού εφαρμογών (API), έχουμε την δομή που χρησιμοποιεί το Fitbit. Ειδικότερα, κάνει χρήση μιας συγκεκριμένης δομής και για το φαγητό και για το νερό που καταναλώνει ο χρήστης, καθώς περιλαμβάνει ταυτόχρονα πεδία και για τις δύο μετρικές. Αυτή είναι και η δομή που θα προσομοιώσουμε, έχοντας υπόψιν μας ότι όταν πρόκειται για καταγραφή φαγητού, το σχετικό πεδίο για το νερό θα έχει την τιμή null. Το αντίστοιχο θα γίνεται και για κάθε καταγραφή του νερού που καταναλώνεται. Το JSON που συνοψίζει τα παραπάνω είναι το εξής:

"Food and water (Fitbit)"

```
{
  "logDate": <dateTimestamp>,
  "logId": <log_id>,

  "nutritionalValues":{
    "calories": <value_in_kcal>,
    "water": <value_in_mL>
  }
}
```

- **Βάρος**

  Όπως και με το προηγούμενο είδος μετρικής, και το βάρος του χρήστη καταγράφεται με κάποιο τρόπο από το Fitbit. Συγκεκριμένα, υπολογίζει το βάρος του χρήστη και τον δείκτη μάζας του σώματος του. Η δομή, λοιπόν, που βρέθηκε στο API της συσκευής είναι ο ακόλουθος:

"Weight (Fitbit)"

```
{
  "bmi": <value>,
  "date": <date>,
  "logId": <log_id>,
  "time": <dateTimestamp>,
  "weight": <value_in_kilograms>
}
```

Τέλος, σχετικά με τους **αισθητήρες στο περιβάλλοντα χώρο του χρήστη**, μπορούμε να ορίσουμε από μόνοι μας τον τρόπο αναπαράστασης των δεδομένων που καταγράφουν, μιας και δεν έχουμε κάποιο έτοιμο παράδειγμα τέτοιου αισθητήρα.

Γενικότερα, αυτό που θέλουμε να υπολογίζουν οι αισθητήρες αυτοί, είναι το πότε το άτομο κάθεται στον καναπέ και το πότε η τηλεόραση είναι ανοικτή. Επομένως, μια δομή που μπορούμε να χρησιμοποιήσουμε είναι ένα JSON, όπου θα ορίζεται σίγουρα η ώρα που αρχίζει η εκάστοτε κατάσταση (η στιγμή δηλαδή που ο χρήστης έκατσε στον καναπέ ή αντίστοιχα η στιγμή που άνοιξε την τηλεόραση). Ακόμη, θα προσδιορίζεται η διάρκεια που η συγκεκριμένη κατάσταση παρέμεινε ως έχει (δηλαδή η διάρκεια μέχρι να σηκωθεί ο χρήστης από τον καναπέ ή αντίστοιχα η διάρκεια που παρέμεινε ανοικτή η τηλεόραση). Σε αυτά θα προστεθούν και κάποια ακόμη σχετικά πεδία, όπως ο αύξων αριθμός της μέτρησης του αισθητήρα (log_id). Συνοψίζοντας τα παραπάνω σε μορφή JSON έχουμε τα εξής:

"couch sensor"

```
{
  "logId": <log_id>,
  "onCouch":true,
  "startTime": <dateTimestamp>,
  "duration": <value_in_milliseconds>
}
```

"TV sensor"

```
{
  "logId": <log_id>,
  "tvOn":true,
  "startTime": <dateTimestamp>,
  "duration": <value_in_milliseconds>
}
```

Όλα τα παραπάνω, επομένως, αποτελούν το σύνολο των δομών που θα χρησιμοποιήσουμε για να αναπαραστήσουμε τα δεδομένα που προσομοιώνει η Γεννήτρια Δεδομένων. Αν παρατηρήσει κάποιος πιο προσεκτικά τις δομές των JSON, θα δει τις διαφοροποιήσεις που υπάρχουν ανάμεσα στις πηγές δεδομένων, που καταγράφουν τέτοιου είδους δείκτες. Συγκεκριμένα, είναι αρχικά εμφανές, ότι υπάρχουν διαφορές στους δείκτες που μετράνε για κάποια κατάσταση/δραστηριότητα. Επιπλέον, ακόμα κι αν υπολογίζουν μια ίδια μετρική, το όνομα που της αποδίδουν διαφέρει, με αποτέλεσμα να μην γνωρίζουμε εκ των προτέρων ότι αναφέρονται στον ίδιο δείκτη ευημερίας. Τέλος, εξίσου αξιοσημείωτο είναι το γεγονός ότι οι μονάδες μέτρησης που χρησιμοποιούν για ίδιου είδους μετρικές παρουσιάζουν σημαντικές διαφοροποιήσεις.

Συνοψίζοντας, οι διαφορές που θα εμφανιστούν ανάμεσα στις αναπαραστάσεις των προσομοιωτικών δεδομένων είναι κάτι που **προσεγγίζει σε μεγάλο βαθμό την πραγματικότητα**. Αυτό είναι και ο κύριος στόχος της προσομοίωσής μας. Εξάλλου, και σε ένα πραγματικό σενάριο, η έλλειψη κοινού λεξιλογίου οδηγεί στην τεράστια ποικιλία αναπαράστασης ίδιου τύπου δεδομένων.

5.1.4 Αλγόριθμοι υπολογισμού τιμών μετρικών ευημερίας

Η τελευταία ενότητα για την υλοποίηση της Γεννήτριας Δεδομένων αφιερώνεται στους **επιμέρους αλγορίθμους** που εφαρμόστηκαν, **προκείμενου να υπολογιστούν οι τιμές των μετρικών** που συμπεριλαμβάνονται στις προαναφερθείσες δομές. Ειδικότερα, μας ενδιαφέρει να εξετάσουμε τον τρόπο με τον οποίο θα παράγουμε τιμές για κάποιους συγκεκριμένους δείκτες ευημερίας. Ανά κατηγορία, επομένως, έχουμε:

- **Κατάσταση ύπνου**

  Για τον ύπνο του χρήστη, παρατηρώντας και τα αντίστοιχα JSON, βλέπουμε ότι χρειαζόμαστε μετρήσεις σχετικές με το πόση ώρα κοιμόταν κανονικά, και με το πόση ώρα καθόταν στο κρεβάτι του ξύπνιος. Ακόμη, μετράμε την διάρκεια κάθε φάσης του ύπνου και το πόσες φορές πέρασε από κάθε φάση. Υπολογίζοντας, βέβαια, ακριβώς τις τιμές που σχετίζονται με τις φάσεις του ύπνου (awake, light, deep, rem phase) μπορούμε να υπολογίσουμε και τους υπόλοιπους δείκτες του ύπνου.

  Ο αλγόριθμος που ακολουθήσαμε για τον υπολογισμό των φάσεων, βασίζεται στο πώς κατανέμεται ο συνολικός χρόνος του ύπνου. Συγκεκριμένα, πήραμε από σελίδα της ίδιας της συσκευής του Fitbit[24], την μέση περίπου διάρκεια κάθε φάσης, σαν ποσοστό επί της συνολικής διάρκειας ύπνου του χρήστη. Έτσι, θεωρήσαμε προσεγγιστικά, ότι η φάση που ο χρήστης είναι ξύπνιος (awake phase) καταλαμβάνει το 5-10% του συνολικού χρόνου, η φάση ελαφρύ ύπνου το 50-60%, η φάση βαθύ ύπνου το 20-25% και τέλος η φάση REM το 20-25%. Τέλος, γνωρίζοντας την μέση διάρκεια ενός ολοκληρωμένου κύκλου ύπνου (δηλαδή ενός περάσματος από όλες τις φάσεις ύπνου), ο οποίος είναι περίπου 90 λεπτά, μπορέσαμε να υπολογίσουμε και το πόσες περίπου φορές περνάει ο χρήστης από κάθε φάση.

  Φυσικά, για τον υπολογισμό των παραπάνω, χρησιμοποιήσαμε και πάλι επιλογή τυχαίας τιμής μέσα σε λογικά πλαίσια, που ορίζονται από τις μέσες τιμές διάρκειας κάθε φάσης ύπνου. Έτσι, πετυχαίνουμε και κάποιες αποκλίσεις από αυτές τις τιμές, το οποίο είναι επιθυμητό για λόγους πιστής προσομοίωσης της πραγματικότητας. Ομοίως επιλέχθηκε βάσει πιθανοτήτων, ο χρήστης να μην περνάει πάντα από όλα τα στάδια μέσα σε έναν κύκλο ύπνου, καθώς και στον πραγματικό κοσμό συμβαίνει, για παράδειγμα, να ξυπνήσει κάποιος στη μέση του ύπνου από έναν εξωτερικό παράγοντα. Αυτό έχει σαν αποτέλεσμα πιθανή διακοπή της ροής που ακολουθεί ένας κύκλος ύπνου. Αυτή η περίπτωση επομένως συμπεριλαμβάνεται με τον παραπάνω τρόπο στην προσομοίωση μας.

- **Δραστηριότητα**

  Όσον αφορά τις δραστηριότητες του χρήστη **(περπάτημα, τρέξιμο, ποδηλασία)**, υπάρχουν σχετικές μετρικές, για τις οποίες πρέπει να παράξουμε τιμές. Αρχικά, η διάρκεια της δραστηριότητας είναι κάθε φορά γνωστή, μέσω του αλγορίθμου που παρουσιάστηκε πιο πάνω για την προσομοίωση της μέρας του χρήστη. Αν βρούμε, λοιπόν, και μια μέση ταχύτητα για την εκάστοτε δραστηριότητα του χρήστη, τότε μπορούμε να υπολογίσουμε βάσει διαδεδομένων τύπων και τις υπόλοιπες μετρικές (απόσταση, βήματα, θερμίδες).

  Για την επιλογή, λοιπόν, της μέσης ταχύτητας μιας δραστηριότητας του χρήστη βασιζόμαστε στην ταχύτητα ενός μέσου ατόμου. Συγκεκριμένα, για το περπάτημα, μια μέση ταχύτητα είναι γύρω στα 3 μίλια/ώρα ή αλλίως περίπου 5 χιλιόμετρα/ώρα[26]. Για το τρέξιμο, θεωρούμε ότι μια λογική τιμή για την ταχύτητα θα είναι μια μέση ταχύτητα τζόκινγκ του ατόμου. Επομένως, επιλέξαμε μέση ταχύτητα γύρω στα 5 με 6 μίλια ανά ώρα[25]. Αντίστοιχα για την ποδηλασία, βρήκαμε ότι μια λογική ταχύτητα για αθλητική άσκηση ενός ατόμου είναι τα 11-12 μίλια/ώρα[27]. Με βάση αυτές τις μέσες τιμές επιλέγουμε κάθε φορά βάσει κανονικής κατανομής μια ταχύτητα για την εκάστοτε δραστηριότητα του χρήστη.

  Έχοντας, λοιπόν, φτιάξει τιμή για την ταχύτητα στην εκάστοτε δραστηριότητα, μπορούμε να υπολογίσουμε εύκολα τώρα, την απόσταση που διένυσε ο χρήστης σαν το γινόμενο του χρόνου που διήρκησε η δραστηριότητα και της ταχύτητας που είχε αυτό το διάστημα. Επιπλέον, για τον μέσο βηματισμό του ατόμου, τόσο στο περπάτημα όσο και στο τρέξιμο, χρησιμοποιούμε τύπους που δίνουν, βάσει του ύψους του ατόμου, τα εκατοστά που διανύει με ένα βήμα. Φυσικά, αυτό είναι μια πολύ γενική προσέγγιση και μπορεί να μην ισχύει για όλους τους χρήστες. Ωστόσο, μας δίνει μια τιμή εντός λογικών πλαισίων με αποτέλεσμα να μπορούμε να την χρησιμοποιήσουμε με ασφάλεια. Ειδικότερα, για το περπάτημα θεωρείται ότι το ύψος σε εκατοστά πολλαπλασιασμένο με το συντελεστή 0.415 για άντρες και 0.413 για γυναίκες δίνει μια καλή προσέγγιση του βήματος[29]. Παρομοίως για ένα χαλαρό τρέξιμο μια τιμή γύρω 0.58 επί το ύψος του ατόμου αποτελεί μια λογική τιμή. Διαιρώντας, επομένως την συνολική απόσταση με το μέσο βήμα του χρήστη βρίσκουμε και τα βήματα που έκανε κατά την δραστηρίότητα.

  Τέλος, μια ιδιαίτερα χρήσιμη μετρική είναι οι θερμίδες που έκαψε ο χρήστης. Για τον σωστό υπολογισμό διαχωρίζουμε κάθε δραστηριότητα σε επιμέρους περιπτώσεις ανάλογα με την ένταση εκτέλεσης της. Ο διαχωρισμός αυτός γίνεται διότι σε κάθε περίπτωση ορίζεται μια διαφορετική τιμή METS (Metabolic Equivalent Task) ανά δραστηριότητα ανάλογα με την ένταση της. Αφού, λοιπόν, βρούμε την τιμή αυτή από σχετικούς πίνακες[30], για την εκάστοτε δραστηριότητα, υπολογίζουμε στη συνέχεια βάσει του τύπου [1]:

$$Calorie = BMR * Mets * durationInHours/24 \tag{5.1}$$

  τις θερμίδες καύσης σε μονάδες kcal. Σημειώνεται ότι το BMR στον παραπάνω τύπο είναι ο βασικός μεταβολικός ρυθμός του ατόμου που έχει τύπο[31]:

$$BMR = 10 * weight(kg) + 6.25 * height(cm) - 5 * age(y) + 5 \tag{5.2}$$

[1] http://keisan.casio.com/menu/system/000000000120

για τους άντρες και

$$BMR = 10 \times weight(kg) + 6.25 \times height(cm) - 5 \times age(y) - 161 \quad (5.3)$$

για τις γυναίκες.

Ολοκληρώθηκε επομένως και η καταγραφή των θερμιδών για τις σωματικές δραστηριότητες του ατόμου. Σημειώνεται ότι αφού συμπεριλάβαμε και τη κατάσταση που ο χρήστης κάθεται σαν ένα είδος δραστηριότητας, πρέπει να υπολογίσουμε και γι΄ αυτή τη περίπτωση τις θερμίδες που καίει. Γι΄ αυτό το σκοπό, χρησιμοποιούμε τον βασικό μεταβολικό ρυθμό του χρήστη, που επί της ουσίας μας δείχνει τις θερμίδες που καίει μέσα σε μια μέρα χωρίς να κάνει καμία σωματική δραστηριότητα. Σαν αποτέλεσμα, λοιπόν, του γινομένου αυτής της ποσότητας με τον χρόνο που καθόταν σε ώρες διά 24(ώρες της μέρας), βρίσκουμε τις θερμίδες που έκαψε.

- **Καρδιακός παλμός**

  Για τον υπολογισμό του καρδιακού παλμού, εξετάζουμε κάθε φορά την εκάστοτε κατάσταση του χρήστη και βάσει αυτής ορίζουμε τα λογικά όρια στις τιμές που μπορεί να λάβει η μετρική αυτή. Στη συνέχεια, επιλέγουμε τυχαία μια τιμή μέσα στο λογικά αυτά όρια. Βεβαίως, αφήνουμε περιθώριο κάποιες φορές να υπάρχει απόκλιση από αυτά τα όρια. Ο λόγος που θέλουμε κάτι τέτοιο στην προσομοίωση μας, είναι διότι και στον πραγματικό κόσμο υπάρχουν περιπτώσεις που ένα άτομο έχει μη φυσιολογικό σφυγμό.

  Συνοπτικά, αντλώντας σχετική πληροφορία από το διαδίκτυο[32], βρήκαμε ό,τι χρειαστήκαμε σχετικά με τον καρδιακό παλμό. Έτσι, γνωρίζουμε αρχικά, ότι ο καρδιακός παλμός κατά την διάρκεια που ο χρήστης ξεκουράζεται κυμάινεται από 60 μέχρι το πολύ 100 παλμούς ανά λεπτό. Επιπλέον, γνωρίζουμε ότι ο μέγιστος καρδιακός παλμός ενός ατόμου δίνεται από τον τύπο

  $$maxHR = 220 - age(years) \quad (5.4)$$

  Με χρήση αυτής της τιμής ορίζουμε και τα όρια του καρδιακού παλμού ανάλογα με την ένταση της δραστηριότητας. Πιο συγκεκριμένα, κατά την κατάσταση που ο χρήστης αθλείται με μέτρια ένταση, ο σφυγμός του είναι περίπου στο 50-70% του μέγιστου (maxHR). Ακόμα, κατά τη διάρκεια μιας έντονης άσκησης η τιμή θα κυμαίνεται στο 70-85% της μέγιστης τιμής (maxHR).

- **Φαγητό**

  Σε αυτό το σημείο θα αναφερθούμε στις θερμίδες που καταναλώνει ένα άτομο στα γεύματα του. Συγκεκριμένα, έχουμε θεωρήσει και με βάση τον αλγόριθμο προσομοίωσης της μέρας, ότι ο χρήστης κάνει ως επί των πλείστων 5 γεύματα μέσα στη μέρα, δηλαδή 3 κύρια γεύματα (πρωί, μεσημέρι, βράδυ) και 2 μικρότερα ανάμεσα στα προηγούμενα 3. Μελετώντας τις θερμίδες κάθε γεύματος ενός μέσου χρήστη[28], θεωρήσαμε ότι λογική τιμή για ένα κύριο γεύμα ενός άντρα είναι περίπου 600-650 θερμίδες και για κάθε ενδιάμεσο γεύμα περίπου 200 θερμίδες. Επιλέξαμε εν τέλει τιμές για κάθε γεύμα, έτσι ώστε να ακολουθούν μια κανονική κατανομή με μέση τιμή 625 θερμίδες για κύριο και 200 για ενδιάμεσο γεύμα. Οι αποκλίσεις που θα προκύψουν από την μέση τιμή κάθε φορά, είναι επιθυμητές στην προσομοίωση ενός σεναρίου του πραγματικού κόσμου.

- **Νερό**

  Σχετικά με το νερό, έχουμε επιλέξει και πάλι την κατανάλωση ως επί των πλείστων 5 φορών μέσα στη μέρα, σε κοντινές με τα γεύματα ώρες. Γνωρίζοντας από σχετικές έρευνες ότι μια μέση ποσότητα νερού είναι 2 λίτρα την ημέρα, μπορούμε να χωρίσουμε αυτή τη ποσότητα σε $2000\text{mL} / 5 = 400\text{mL}$ την φορά. Φυσικά και πάλι επιλέγουμε τυχαία μια τιμή ώστε να ακολουθείται κανονική κατανομή με μέση τιμή τα 400mL.

- **Βάρος**

  Όσον αφορά το βάρος του ατόμου, το οποίο υπολογίζεται μια φορά προς το τέλος της μέρας, θα χρησιμοποιήσουμε για τον υπολογισμό τις θερμίδες που κατανάλωσε και τις θερμίδες που έκαψε. Μετά από μια σύντομη έρευνα, μπορεί να καταλάβει κανείς ότι δεν υπάρχει γενικός τύπος που να ορίζει την αντιστοιχία ανάμεσα σε θερμίδες και κιλά. Υπάρχουν διαφοροποιήσεις από άτομο σε άτομο και είναι πολλοί οι παράγοντες που παίζουν ρόλο. Πολύ προσεγγιστικά όμως, μπορούμε να βασιστούμε στο ισοζύγιο των θερμίδων και να μεταβάλλουμε λίγο το βάρος ανάλογα με το πόσο απέχουν μεταξύ τους οι εισερχόμενες και οι εξερχόμενες από το σώμα θερμίδες.

Τα παραπάνω συνοψίζουν όλες τις μεθόδους που ακολουθήσαμε προκειμένου να υπολογιστούν τιμές για κάθε μετρική. Ειδικότερα, σημειώνεται ότι όταν είχαμε δύο πηγές δεδομένων (Fitbit, UnderArmour) που καταγράφουν έναν ίδιο δείκτη, επιλέξαμε η μια πηγή (Fitbit) να υπολογίζει μια πρώτη τιμή με βάση τα παραπάνω, ενώ η δεύτερη (UnderArmour) να υπολογίζει την δική της τιμή με βάση την πρώτη. Με άλλα λόγια, με σκοπό την **εμφάνιση μικρών αποκλίσεων μεταξύ των δύο πηγών**, παίρνουμε κάθε φορά την πρώτη τιμή για μια μετρική και επιλέγουμε τυχαία μια άλλη τιμή κοντά σε αυτή. Έτσι, ενσωματώνουμε στη προσομοίωση μας και την περίπτωση που κάποιες συσκευές μετράνε διαφορετικά τους δείκτες ευημερίας του ατόμου.

Έχοντας, λοιπόν, ολοκληρώσει την ενότητα της υλοποίησης της Γεννήτριας Δεδομένων, γνωρίζουμε ακριβώς τον τρόπο με τον οποίο προσομοιώνονται τα δεδομένα του χρήστη και η μορφή που αυτά αναπαριστώνται. Αυτά τα δεδομένα θα αποτελέσουν την τροφοδοσία του συστήματος μας, καθώς θα σταλούν στον εξυπηρετητή με κρυπτογραφημένο φυσικά τρόπο. Μάλιστα, η κρυπτογράφηση τους θα γίνει με χρήση του αλγορίθμου **AES-128** (Advanced Encryption Algorithm- 128bits) με προκαθορισμένο κλειδί, θεωρώντας ότι ο εξυηρετητής έχει πιστοποιηθεί από τις πηγές δεδομένων και έχει την δυνατότητα να ζητάει τα δεδομένα του χρήστη. Στην επόμενη ενότητα επομένως, θα μελετήσουμε τον τρόπο συλλογής και διαχείρησης τους από τον εξυπηρετητή.

5.2 Εξυπηρετητής

Μετά τον σχεδιασμό του εξυπηρετητή στο κεφάλαιο 4, έχουμε αποκτήσει μια γενική εικόνα της αρχιτεκτονικής του και των υπηρεσιών που διαθέτει. Στη παρούσα ενότητα, λοιπόν, θα εμβαθύνουμε σε κάθε υπηρεσία ξεχωριστά προκειμένου να κατανοήσουμε τον τρόπο με τον οποίο επιτελούν τις λειτουργίες τους.

Ξεκινώντας, επομένως, από το **επίπεδο συλλογής των δεδομένων** πρέπει να αναφερθούμε στην υλοποίηση της Υπηρεσία Λήψης Ακατέργαστων Δεδομένων.

- **Υπηρεσία Λήψης Ακατέργαστων Δεδομένων**

  Σε αυτή την υπηρεσία αποστέλλονται τα δεδομένα, τα οποία γνωρίζουμε πλέον ότι αναπαριστώνται σε μια από τις δομές που παρουσιάστηκαν στη προηγούμενη ενότητα. Η υπηρεσία λοιπόν, έχει ορίσει συγκεκριμένες διευθύνσεις (URLs - Uniform Resource Location) για την λήψη διαφορετικού είδους δεδομένων. Διαθέτει, δηλαδή, διαφορετικές διευθύνσεις ανά πηγή δεδομένων σε πρώτο επίπεδο και ανά είδος δεδομένων σε δεύτερο. Σαν αποτέλεσμα, αυτό που λαμβάνει σε κάθε διεύθυνση είναι το JSON που αντιστοιχεί σε αυτήν, κρυπτογραφημένο με βάση τον αλγόριθμο (AES - 128bits) και το κλειδί που έχουν συμφωνήσει ο εξυπηρετητής και η Γεννήτρια Δεδομένων.

  Κάθε φορά, επομένως, αναλαμβάνει να αποκρυπτογραφήσει το μήνυμα που λαμβάνει, προκειμένου να παράξει το αρχικό JSON όπως το έστειλε η Γεννήτρια Δεδομένων. Αυτό στη συνέχεια το αποθηκεύει σε μια μη σχεσιακή βάση του συγκεκριμένου χρήστη σε mongoDB, όπου θα συλλέγονται όλα τα ακατέργαστα δεδομένα. Σημειώνεται ότι αυτή η βάση έχει συλλογές (collections) ανά είδος δεδομένων (δηλαδή activity, sleep, heart_rate, weight, food_and_water, sensor_data), όπου αποθηκεύονται τα αντίστοιχα JSON από όποια πηγή και αν προέρχονται αφού πρώτα συμπληρωθούν με ένα πεδίο στη δομή τους, το οποίο υποδηλώνει την συσκεύη που τα κατέγραψε. Τέλος, τα ακατέργαστα αυτά δεδομένα προωθούνται στο επίπεδο επεξεργασίας για την διαχείριση τους.

Εμβαθύνοντας, τώρα, στο **επίπεδο επεξεργασίας**, η υλοποίηση των λειτουργιών ανά υπηρεσία φαίνεται ακολούθως.

- **Υπηρεσία Σημασιολογικού Μετασχηματισμού Δεδομένων**

  Η παρούσα υπηρεσία παίζει έναν από τους πιο σημαντικούς ρόλους στο σύστημα μας, αφού θα μετατρέψει βάσει οντολογίας τα δεδομένα σε σημασιολογικά, επιβεβαιώνοντας και πρακτικά την δυνατότητα προτυποποίησης των δεικτών ευημερίας με τη χρήση ενός κοινού λεξιλογίου. Με πιο απλά λόγια αναλαμβάνει τον **μετασχηματισμό των JSON** που λάβαμε στο προηγούμενο επίπεδο **σε JSON-LD** το οποίο ακολουθεί το σχήμα της οντολογίας, όπως αυτό αναλύθηκε κατά την σχεδίαση.

  Ειδικότερα, αξίζει σε αυτό το σημείο να εστιάσουμε στον τρόπο που οι δομές των JSON μετατρέπονται σε οντολογική μορφή. Αρχικά, κάθε JSON μπορούμε να παρατηρήσουμε ότι αποτελεί μια παρατήρηση (Observation) για το συγκεκριμένο άτομο (Person). Μάλιστα, η παρατήρηση έγινε μια δεδομένη χρονική στιγμή (observationTimestamp). Επιπλέον, κάθε δομή JSON περιέχει πεδία που αφορούν μετρικές που καταγράφηκαν, οι οποίες μπορούν να θεωρηθούν τύποι παρατηρήσεων (ObservationType) της συνολικής παρατήρησης. Φυσικά κάθε μετρική έχει έναν συγκεκριμένο τύπο (VitalSign, BodyMeasurement, ActivityMetric, SleepMetric, Other), ανάλογα με το τί αναπαριστά. Έτσι έχουμε συνοπτικά, τον καρδιακό παλμό με τύπο VitalSign, το βάρος, τον δείκτη μάζας σώματος, τις θερμίδες και την κατανάλωση νερού με τύπο BodyMeasurement, τους σχετικούς με μια δραστηριότητα δείκτες σαν ActivityMetric , τους σχετικούς δείκτες ενός ύπνου σαν SleepMetric, και άλλου είδους δεδομένα (αισθητήρας τηλεόρασης ή καναπέ) με τύπο Other. Σημειώνεται,

επιπλέον, ότι κάθε μετρική, δηλαδή κάθε τύπος παρατήρησης (ObservationType) έχει μια συγκεκριμένη τιμή (observationTypeValue) και συγκεκριμένη μονάδα μέτρησης (Unit), ενώ επίσης ενδέχεται να έχει και συγκεκριμένο ταυτοποιητικό (observationTypeLoincNum) αν ορίζεται κάπως από το σύστημα LOINC (Logical Observation Identifiers Names and Codes[2]).

Πέρα λοιπόν από τις ιδιότητες της, η παρατήρηση (Observation), ενδέχεται να σχετίζεται (raisedWithin) με μία συγκεκριμένη κατάσταση (State) του χρήστη. Αυτή η κατάσταση αρχίζει και τελειώνει σύμφωνα με το σχετικό JSON μια δεδομένη στιγμή (startTime και endTime αντίστοιχα), έχει καθορισμένη διάρκεια (Duration) και διαθέτει επιπλέον έναν αριθμό-ταυτοποιητικό (stateLogId) που προέρχεται από το <log_id> του JSON. Ακόμη, έχει τύπο Activity, Rest ή Sleep και ανάλογα με αυτόν το τύπο διαθέτει επιπλέον πεδία (ActivityType ή SleepPhase κλπ).

Έχοντας μελετήσει την αντιστοίχιση των πεδίων των JSON στην οντολογία, μπορούμε να κατανοήσουμε πώς θα γίνεται η μετατροπή σε JSON-LD. Γενικά, λοιπόν, κάθε αντικείμενο στο JSON-LD θα έχει αρχικά ένα πεδίο για την ατιστοίχιση του με την οντολογία που αναπτύξαμε ("@context" = <ontology_IRI>). Επίσης, κάθε JSON-LD θα ξεκινάει με ένα αντικείμενο της κλάσης Observation ("@type"="Observation"), αφού όπως είπαμε κάθε JSON είναι μια παρατήρηση. Ακόμα, σε κάποια αντικείμενα δίνεται μέσω του πεδίου @id ένα συγκεκριμένο όνομα προκειμένου να δηλωθεί η συσχέτιση του με τα υπόλοιπα JSON-LD που έχουν στο σώμα τους ίδιου τύπου δεδομένα. Επιπλέον, κάθε ιδιότητα της κλάσης του αντικειμένου θα γίνεται πεδίο στο JSON-LD και θα συμπληρώνεται με την σχετική πληροφορία από το JSON. Ειδικότερα, κάθε πεδίο του JSON που αντιστοιχεί σε ιδιότητα δεδομένων, θα παίρνει στο JSON-LD σαν όνομα, το όνομα που αποδώσαμε κατά την ανάπτυξη της οντολογίας, και σαν τιμή, την τιμή που καταγράφηκε στο JSON. Όταν, όμως, πρόκειται για ιδιότητα αντικειμένων, θα έχουμε στο JSON-LD σαν ζευγάρι πεδίου-τιμής, το όνομα της ιδιότητας (πεδίο) με τιμή ένα άλλο αντικείμενο (τιμή) της οντολογίας όπως καθορίζεται από την συγκεκριμένη ιδιότητα.

Σε γενικές γραμμές επομένως, αυτός είναι ο τρόπος μετατροπής των JSON σε JSON-LD. Σημειώνεται επίσης, ότι κατα τον μετασχηματισμό ελέγχεται αν η παρατήρηση αφορά τέλος σωματικής δραστηριότητας και αν ναι, προσδιορίζεται και η έντασή της (intensity) βασισμένη στο είδος της δραστηριότητας και στην μέση ταχύτητα που είχε ο χρήστης κατά τη διάρκεια της.

Ενδεικτικά, μπορούμε να δούμε σε αυτό το σημείο ένα παράδειγμα μετασχηματισμού του JSON σε JSON-LD. Επιλέξαμε συγκεκριμένα πληροφορία που καταγράφηκε από το Fitbit για μια δραστηριότητα τρεξίματος (Running) του χρήστη. Έχουμε δηλαδή:

[2]https://loinc.org

”JSON Running (Fitbit)”


```
{
      "activityId": 12030,
      "calories": 479,
      "description": "6.46mph",
      "distance": "3.66",
      "duration": 2040000,
      "hasStartTime": true,
      "logId": 176,
      "name": "Running",
      "startTime": "2017-10-19T20:19:00Z
          ",
      "steps": "2696"
}
```


”JSON-LD Running”


```
{
  "@context": <ontology_IRI>,
  "@type": "Observation",
  "observationTimestamp": "2017-10-19T20
     :53:00Z",
  "observationLogId": 176,
  "hasObservationType": [
    {
      "@type": "BodyMeasurement",
      "@id": <caloriesBurned_IRI>
      "observationTypeLoincNum": "41981-
          2",
      "observationTypeValue": 479,
      "hasObservationTypeUnit": {
        "@type": "Unit",
        "@id": <energyUnit_IRI>,
        "unitDescription": "kcal"
        }
     },
     {
      "@type": "ActivityMetric",
      "@id": <distance_IRI>,
      "observationTypeValue": "5890.2",
      "hasObservationTypeUnit": {
        "@type": "Unit",
        "@id": <distanceUnit_IRI>,
        "unitDescription": "meters"
       }
     },
     {
      "@type": "ActivityMetric",
      "@id": <speed_IRI>,
      "observationTypeValue": "2.89",
      "hasObservationTypeUnit": {
        "@type": "Unit",
        "@id": <speedUnit_IRI>,
        "unitDescription": "meters/
            second"
       }
  },
  {
      "@type": "ActivityMetric",
      "@id": <steps_IRI>,
      "observationTypeValue": "2696",
      "hasObservationTypeUnit": {
        "@type": "Unit",
        "@id": <noUnit_IRI>,
        "unitDescription": ""
       }
    }
  ],
  "raisedWithin": {
    "@type": "Activity",
    "@id": <activity_IRI>,
    "startTime": "2017-10-19T20:19:00Z",
    "endTime": "2017-10-19T20:53:00Z",
    "stateLogId": 10208,
    "hasActivityType": {
      "@type": "ActivityType",
      "@id": <running_IRI>,
      "activityTypeId": 12030
     },
    "hasDuration": {
      "@type": "Duration",
      "durationValue": "34.00",
      "hasDurationUnit": {
        "@type": "Unit",
        "@id": <timeUnit_IRI>,
        "unitDescription": "minutes"
       }
    },
    "intensity": "vigorous"
    },
  "recordedBy": {
    "@type": "MobileApplication",
    "@id": <fitbit_IRI>,
    "dataSourceName": "fitbit",
    "dataSourceId": 1
  },
  "refersToPerson": {
    "@type": "Person",
    "personId": "1"
  }
}
```


Μετά από την παραπάνω ανάλυση επομένως, γνωρίζουμε πώς κάθε JSON θα μετατραπεί σε JSON-LD από την υπηρεσία. Στη συνέχεια, τα δεδομένα που θα

παραχθούν, αποθηκεύονται σε σχετική μη σχεσιακή βάση (mongoDB) του χρήστη σε συλλογές (collections) ίδιες με αυτές που αποθηκεύουμε τα JSON (heart_rate, food_and_water, weight, activity, sleep, sensor_data). Τα σημασιολογικά πλέον δεδομένα προωθούνται στην επόμενη υπηρεσία.

- **Υπηρεσία Ελέγχου Μονάδων Μέτρησης**

  Τα σημασιολογικά δεδομένα που παράχθηκαν από την προηγούμενη υπηρεσία σε JSON-LD έχουν σαν μονάδες μέτρησης των μετρικών τους, τις μονάδες που χρησιμοποιεί η εκάστοτε συσκευή. Σαν αποτέλεσμα, υπάρχουν διαφοροποιήσεις ακόμα και για ίδιους δείκτες ευημερίας. Για την εύκολη διαχείριση στα επόμενα στάδια, χρησιμοποιούμε την Υπηρεσία Ελέγχου Μονάδων Μέτρησης ώστε όλες οι μονάδες μέτρησης για ίδιου τύπου δεδομένα να γίνουν ίδιες. Συγκεκριμένα, οι μονάδες μέτρησης που επιλέχθηκαν ανά μετρική καθορίστηκαν όπου είναι εφικτό από τα πρότυπα που έχουν τεθεί από το σύστημα LOINC. Έτσι, βρήκαμε ότι το βάρος του χρήστη μετράται σε γραμμάρια (g), ο δείκτης μάζας σώματος σε κιλά διά μέτρα στο τετράγωνο (kg/m^2), ο καρδιακός παλμός σε παλμούς ανά λεπτό (beats/min), και οι θερμίδες σε kcal. Θεωρήσαμε ακόμα για μονάδες μέτρησης, τα μέτρα (meters) για την απόσταση, τα μέτρα ανά δευτερόλεπτο (meters/second) για τη ταχύτητα, και τα λεπτά (minutes) γενικότερα για τη χρονική διάρκεια.

  Παίρνωντας, επομένως, τα παραπάνω JSON-LD ελέγχουμε κάθε αντικείμενο που περιέχει σαν τιμή σχετικού πεδίου ένα αντικείμενο της κλάσης Unit. Βλέπουμε, λοιπόν, την τιμή του unitDescription και το πεδίο @id στο συγκεκριμένο αντικείμενο προκειμένου να μάθουμε τόσο τί μονάδες μετρησης έχει όσο και τι τύπου μετρική είναι. Αν η μονάδα μέτρησης του εκάστοτε αντικειμένου δεν ταυτίζεται με αυτή που ορίσαμε παραπάνω, τότε βρίσκουμε την τιμή της μετρικής και με κατάλληλες πράξεις την μετατρέπουμε. Τέλος, αλλάζουμε και το unitDescription του αντικειμένου κλάσης Unit ώστε να ακολουθείται η κοινή μονάδα μέτρησης του δείκτη.

  Με αυτό το τρόπο, παράγουμε τα τελικά JSON-LD με τις κοινές μονάδες μέτρησης ανά μετρική. Στη συνέχεια τα αποθηκεύουμε σε άλλη μη σχεσιακή (mongoDB) βάση του χρήστη χωρισμένη σε συλλογές (collections) ακριβώς όπως και οι προαναφερθείσες βάσεις.

  Επιπλέον, ανάλογα με το είδος των δεδομένων που συλλέχθηκαν, κάνουμε μέσω της τεχνολογίας Apache Kafka δημοσίευση (produce) του τελικού JSON-LD στο σχετικό θέμα (topic) δεδομένων υγείας που αντιστοιχεί. Τα θέματα που έχουμε στη παρούσα φάση είναι τα εξής: heartRate, state, calorieIntake, water, weight, sensorData. Εύκολα μπορούμε να κατανοήσουμε επομένως σε ποιό θέμα δημοσιεύται κάθε JSON-LD. Στις επόμενες υπηρεσίες θα εξετάσουμε πώς καταναλώνονται αυτά τα μηνύματα.

- **Υπηρεσία Δημιουργίας Άμεσων Προειδοποιήσεων**

  Από τα παραπάνω θέματα δεδομένων υγείας, αυτά που χρήζουν άμεσου ελέγχου καταναλώνονται από τη συγκεκριμένη υπηρεσία. Στο σύστημα μας συγκεκριμένα, μόνο ο καρδιακός παλμός είναι ζωτικής σημασίας και γι΄ αυτο η παρούσα υπηρεσία εγγράφεται (subscribe) στο σχετικό θέμα (heartRate topic) για τον παλμό και δέχεται JSON-LD

τέτοιας μορφής. Επίσης, πολύ σημαντικό για αυτήν την υπηρεσία είναι να γνωρίζει σε τί κατάσταση σωματική βρίσκεται ο χρήστης, δηλαδή αν κάθεται, αν κοιμάται ή αν αθλείται. Μάλιστα, αυτή η πληροφορία θα χρειαστεί και για τον έλεγχο του καρδιακού παλμού, αφού η τιμή που καταγράφεται πρέπει να έλεγχεται με συγκεκριμένα κάθε φορά όρια που ορίζονται από την εκάστοτε κατάσταση του ατόμου.

Έτσι, κάθε φορά που η υπηρεσία καταναλώνει ένα JSON-LD με τον καρδιακό παλμό, παίρνει την τιμή που καταγράφηκε και ελέγχει αν είναι στα λογικά όρια με βάση την κατάσταση του χρήστη. Ειδικότερα, αν ο χρήστης κάθεται, τα όρια πρέπει να είναι απο 60 μέχρι το πολύ 100 παλμούς ανά λεπτό. Αν αντίστοιχα κάνει κάποια δραστηριότητα πρέπει να έχει παλμό που κυμάινεται περίπου στο 50-70% του μέγιστου παλμού του για μέτριας έντασης άσκηση και 70-85% για έντονη άσκηση (όπου maxHR = 220 - age_in_years). Αν, λοιπόν, σε αυτό το σημείο παρατηρηθεί απόκλιση από τα λογικά όρια, τότε δημιουργείται μία προειδοποίηση για τον χρήστη με πεδίο triggered ίσον με true, προκειμένου να ενεργοποιηθεί αυτόματα και να ενημερωθεί ο χρήστης. Τα υπόλοιπα πεδία του JSON-LD της προειδοποίησης που είναι απαραίτητα, συμπληρώνονται εύκολα, με την χρονική στιγμή που φτιάχτηκε η προειδοποίηση (notificationCreatedTimestamp) και με την τιμή που καταγράφηκε για τον καρδιακό παλμό (alertValue). Τέλος, στο πεδίο notificationDescription εισάγουμε μια περιγραφή της προειδοποίησης, δηλαδή αν ο καρδιακός παλμός ήταν χαμηλότερος ή υψηλότερες από τα κανονικά όρια της εκάστοτε κατάστασης. Το JSON-LD της προειδοποίησης (alert) εισάγεται σε σχετική μη σχεσιακή βάση (mongoDB) του χρήστη που περιλαμβάνει όλες τις προειδοποιήσεις σε συλλογές ανάλογα με το είδος τους (heart_rate, food, water, sedentary).

- **Υπηρεσία Γραφικής Αναπαράστασης Δεδομένων**

  Σαν καταναλωτής Apache Kafka όλων των θεμάτων δεδομένων υγείας λειτουργεί η Υπηρεσία Γραφικής Αναπαράστασης Δεδομένων. Με άλλα λόγια, αναλαμβάνει την συλλογή όλων των JSON-LD προκειμένου να τα εισάγει στην βάση δεδομένων γραφημάτων (neo4j). Σημειώνεται ότι η παρούσα υπηρεσία διαχειρίζεται όλη τη γραφική αναπαράσταση των δεδομένων ξεκινώντας πρώτα απ΄ όλα από την εισαγωγή της οντολογίας μας στον γράφο. Μετατρέπει, δηλαδή κάθε κλάση της οντολογίας σε κόμβο και πραγματοποιεί τις απαραίτητες συνδέσεις μεταξύ τους, όπως τις σχέσεις κλάσης-υποκλάσης (isA) που υπάρχουν. Στη συνέχεια, λαμβάνει κάθε είδους JSON-LD και το προσθέτει στον γράφο συνδέοντας τον και με τους απαραίτητους κόμβους-κλάσεις της οντολογίας.

  Για την αναπαράσταση του JSON-LD σε μορφή γράφου, η μέθοδος που ακολουθούμε είναι να παίρνουμε αρχικά κάθε αντικείμενο που ορίζεται μέσα στο JSON-LD και να φτιάχνουμε έναν κόμβο για αυτό, συνδέοντας τον με σχέση isA με τον κόμβο-κλάση στην οποία ανήκει. Στη συνέχεια, προσθέτουμε σαν ιδιότητα (property) του κόμβου κάθε ιδιότητα τύπου δεδομένων του αντικειμένου και σαν σχέση (relationship) κάθε ιδιότητα τύπου αντικειμένων. Φυσικά, αυτή η σχέση θα ενώνει τον εκάστοτε κόμβο με έναν άλλον ο οποίος αναπαριστά ένα άλλο αντικείμενο. Αν αυτό το αντικείμενο υπάρχει ήδη από προηγούμενα JSON-LD τότε πραγματοποιείται κανονικά η σύνδεση στον γράφο, ενώ αν δεν υπάρχει πρέπει να κατασκευαστεί με παρόμοιο τρόπο. Για

παράδειγμα, μια περίπτωση όπου ο κόμβος υπάρχει ήδη, είναι όταν αναφερόμαστε σε ίδιες μονάδες μέτρησης κάποιων αντικειμένων. Σε αυτή τη περίπτωση, αν δύο κόμβοι έχουν ίδιες μονάδες μέτρησης (hasObservationTypeUnit) θα ενωθούν με τον ίδιο κόμβο στον γράφο, ο οποίος αντιπροσωπεύει την συγκεκριμένη μονάδα μέτρησης.

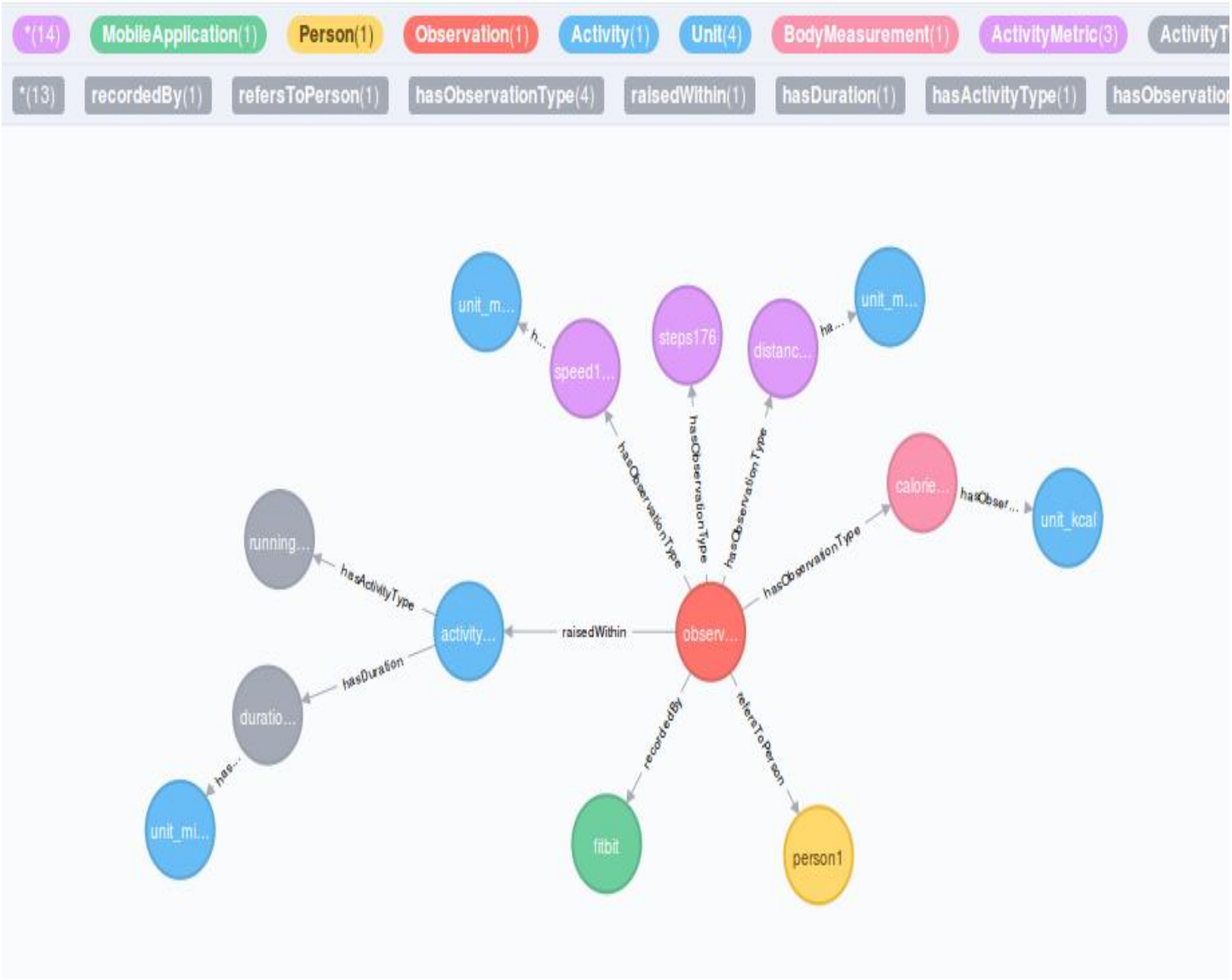
Τέλος, ορίζουμε για κάθε κόμβο μία ιδιότητα για το όνομά του (nodeName), η οποία αντιπροσωπεύει ουσιαστικά το όνομα του αντικειμένου. Γνωρίζουμε, εκ των προτέρων το είδος του κόμβου (δηλαδή αν πρόκειται για συγκεκριμένη μετρική ή γενικά για παρατήρηση κλπ), οπότε με βάση αυτό μπορούμε να προσδιορίσουμε το όνομα του σε πρώτη φάση (observation για Observation, caloriesBurned για την σχετική μετρική κλπ). Σε δεύτερη φάση, αρκεί να ακολουθείται αυτό το όνομα και από έναν αύξων αριθμό ώστε να μην έχουμε ίδια ονόματα κόμβων. Επιλέγουμε, λοιπόν, για κάθε JSON-LD που λαμβάνουμε, να προσθέσουμε στα ονόματα των κόμβων που θα δημιουργηθούν, τον άυξων αριθμό της εκάστοτε παρατήρησης. Με άλλα λόγια όλοι οι κόμβοι που σχετίζονται με την ίδια παρατήρηση (Observation) θα έχουν σαν τέλος στο όνομά τους (nodeName) τον ταυτοποιητικό αριθμό που φέρει και η παρατήρηση.

Με σκοπό να κατανοηθεί καλύτερα ο τρόπος εισαγωγής ενός JSON-LD στον γράφο της βάσης μας, παρατίθεται στη συνέχεια στιγμιότυπο της βάσης neo4j στο Σχήμα 5.2. Φαίνεται συγκεκριμένα πώς το παράδειγμα που χρησιμοποιήσαμε και πιο πάνω μετατρέπεται σε κόμβους και σχέσεις στον γράφο. Πέρα από τα ονόματα των κόμβων και τις σχέσεις που φαίνονται σε αυτό το στιγμιότυπο, αξίζει να αναφέρουμε ότι κάθε ένας από αυτούς του κόμβους έχει τις δικές του ιδιότητες, ενώ ακόμη συνδέεται με σχέση isA με τον εκάστοτε κόμβο-κλάση της οντολογίας. Η κλάση βέβαια τοποθετείται και σαν ετικέτα στον κόμβο και αυτό φαίνεται και στο σχήμα αν παρατηρήσουμε την διαφορά στο χρώμα των κόμβων διαφορετικής κλάσης.

Listing 5.1: "JSON Running (Fitbit)"

```
{
      "activityId": 12030,
      "calories": 479,
      "description": "6.46mph",
      "distance": "3.66",
      "duration": 2040000,
      "hasStartTime": true,
      "logId": 176,
      "name": "Running",
      "startTime": "2017-10-19T20:19:00Z",
      "steps": "2696"
}
```

Αφότου το JSON-LD μετατράπηκε και σε κόμβους στον γράφο μας, η υπηρεσία αναλαμβάνει ανάλογα με το είδος της πληροφορίας (γεγονός- event) να φτιάξει συγκεκριμένο μήνυμα κάθε φορά , το οποίο περιέχει την ώρα που παρατηρήθηκε ένα γεγονός (event) και το όνομα του κόμβου της παρατηρησης στη βάση neo4j.
Το μήνυμα αυτό θα δημοσιευτεί (publish) μέσω της τεχνολογίας RabbitMQ σε συγκεκριμένη ουρά, ανάλογα με το γεγονός που παρατηρείται για τον χρήστη (π.χ. τέλος δραστηριότητας, αρχή ύπνου, κατανάλωση θερμίδων και άλλα).



Σχήμα 5.2: Παράδειγμα μετασχηματισμού σε Neo4j

Στη συνέχεια, θα δούμε ότι οι καταναλωτές των μηνυμάτων χωρίζονται σε δύο μεγάλα κανάλια που το ένα αντιπροσοπεύει γεγονότα που χρειάζονται για τον έλεγχο δημιουργίας προειδοποιήσεων (alertChannel), και το άλλο γεγονότα χρήσιμα για την εξαγωγή συμπερασμάτων- περιλήψεων κατάστασης ή δραστηριότητας (summaryChannel). Πρόκειται ουσιαστικά για τις επόμενες δύο υπηρεσίες, οι οποίες αναλαμβάνουν την επεξεργασία των σημασιολογικών δεδομένων κατά δεσμίδες (batch processing).

- **Υπηρεσία Δημιουργίας εν δυνάμει Προειδοποιήσεων**

  Η υπηρεσία αυτή διαθέτει καταναλωτές μηνυμάτων (consumers) που ανήκουν στο κανάλι που σχετίζεται με τις προειδοποιήσεις. Έχουμε, λοιπόν, έναν καταναλωτή (consumer) για τον έλεγχο γεγονότων που σχετίζονται με το φαγητό (food_alertQueue), έναν για το νερό που καταναλώνει ο χρήστης (water_alertQueue) και έναν για την καθιστική του κατάσταση (rest_alertQueue).

  Ξεκινώντας, λοιπόν, από τον καταναλωτή για τον έλεγχο του φαγητού, αυτό που κάνει σε γενικές γραμμές είναι να δημιουργεί, όταν παίρνει μήνυμα φαγητού, μια εν δυνάμει προειδοποίηση, η οποία θα ενημερώσει τον χρήστη μετά το πέρας ενός χρονικού διαστήματος 3 ωρών ότι έχει μείνει αρκετή ώρα νηστικός. Για την υλοποίηση αυτής της προειδοποίησης, φτιάχνεται αντίστοιχο JSON-LD, στο οποίο εισάγεται σαν notificationCreatedTimestamp η ώρα της παρατήρησης του φαγητού, σαν notificationTriggerTimestamp η ώρα που θέλουμε να γίνει ενεργή η προειδοποίηση (δηλαδή μετά από 3 ώρες σε περίπτωση φαγητού) και σαν περιγραφή (notificationDescription) ένα μήνυμα για το ότι ο χρήστης είναι αρκετή ώρα νηστικός. Φυσικά, τα πεδία triggered και executed θα έχουν αρχικά τιμή false αφού πρόκειται για εν δυνάμει προειδοποίηση που δεν είναι ενεργή (triggered = false) από την αρχή και προφανώς δεν έχει αντιμετωπιστεί (executed = true). Επιπλέον, πριν την εισαγωγή αυτής της προειδοποίησης πρέπει να πραγματοποιηθεί και έλεγχος σε περίπτωση που υπάρχει ήδη άλλη προειδοποίηση που είτε δεν είναι ενεργή ακόμα (triggered = false), είτε είναι ενεργή αλλά δεν εχει αντιμετωπιστεί από τον χρήστη (triggered = true, executed = false). Η πρώτη περίπτωση εμφανίζεται ουσιαστικά όταν το άτομο καταναλώσε φαγητό πριν περάσει το διάστημα των 3 ωρών, ενώ η δεύτερη αφότου έχει περάσει αυτό το διάστημα και η προειδοποίηση που δημιουργήθηκε τότε πρόλαβε να γίνει ενεργή. Επομένως, στην πρώτη περίπτωση η εν δυνάμει προειδοποίηση διαγράφεται αφού δεν πρόλαβε να γίνει ενεργή, ενώ στην δεύτερη, παραμένει στη βάση, αφού αλλάξει το αντίστοιχο πεδίο της (executed) σε true. Με βάση τα παραπάνω, μέσα στη μη σχεσιακή βάση των προειδοποιήσεων (user1_alert mongoDB) θα υπάρχει το πολύ ένα JSON-LD στη συλλογή του φαγητού (food) που είτε δεν θα είναι ενεργό, είτε θα είναι αλλά δεν θα έχει αντιμετωπιστεί από τον χρήστη ακόμη. Τέλος, σημειώνεται ότι αν ληφθεί μήνυμα που υποδηλώνει την έναρξη κύριου ύπνου όλες οι εν δυνάμει προειδοποιήσεις θα διαγραφούν. Κατ΄ αναλογία, όταν ληφθεί μήνυμα για το τέλος κύριου ύπνου, τότε σημαίνει ότι ο χρήστης μόλις ξεκίνησε την μέρα του, οπότε η δημιουργία μιας εν δυνάμει προειδοποίησης για φαγητό σε μια ώρα θα ήταν επιθυμητή για έναν μέσο χρήστη.

  Ακριβώς η ίδια λογική εφαρμόζεται και στον καταναλωτή του RabbitMQ για τον

έλεγχο του νερού που καταναλώνεται. Και πάλι το χρονικό διάστημα για να γίνει ενεργή μια προειδοποίηση νερού ορίζεται στις 3 ώρες. Επίσης, τα σχετικά JSON-LD αποθηκεύονται σε συλλογή για τις προειδοποιήσεις νερού (water) στην μη σχεσιακή βάση των προειδοποιήσεων του χρήστη (user˙alert mongoDB). Τέλος, και αυτός ο καταναλωτής ενημερώνεται για έναρξη ή λήξη κύριου ύπνου προκειμένου να διαγράψει ή να δημιουργήσει αντίστοιχα τις κατάλληλες προειδοποιήσεις.

Σχετικά τώρα με τη κατάσταση που ο χρήστης κάθεται ο σχετικός καταναλωτής (consumer) λαμβάνει μηνύματα γεγονότων για τη στιγμή που ο χρήστης έκατσε και για την στιγμή που σηκώθηκε. Στη πρώτη περίπτωση, δημιουργεί μια εν δυνάμει προειδοποίηση με ανάλογη περιγραφή για καθιστική ζωή, η οποία θα γίνει ενεργή μετά από 3 ώρες. Η λογική, επομένως, για την δημιουργία είναι η ίδια, μόνο που τώρα χρειαζόμαστε και το γεγονός της λήξης της κατάστασης του χρήστη. Σε αυτή τη περίπτωση, ο καταναλωτής διαχειρίζεται με τον γνωστό τρόπο τις προειδοποιήσεις. διαγράφει τις εν δυνάμει και αλλάζει το πεδίο executed στις ενεργές, σε true.

- **Υπηρεσία Δημιουργίας Περιλήψεων**

  Από την υπηρεσία αυτή καταναλώνονται μηνύματα του καναλιού του RabbitMQ, που σχετίζεται με την δημιουργία περιλήψεων κατάστασης/δραστηριότητας. Υπάρχει συγκεκριμένα, ένας καταναλωτής για γεγονότα που αφορούν το τέλος μια σωματικής δραστηριότητας του χρήστη, η οποία όμως ήταν τουλάχιστον μέτριας έντασης. Αν, αντίθετα ήταν ήπιας έντασης δεν παράγεται καν μήνυμα από την Υπηρεσία Γραφικής Αναπαράστασης Δεδομένων. Ακόμα υπάρχουν καταναλωτές για μηνύματα κάθε φορά που τελείωνει ο κύριος ή ο μεσημεριανός, αντίστοιχα, ύπνος του ατόμου. Τέλος, η υπηρεσία διαθέτει και καταναλωτή για περιλήψεις βάρους του χρήστη κάθε φορά που καταγράφεται από μια πηγή δεδομένων.

  Αρχικά, με το μήνυμα για λήξη μιας αθλητικής δραστηριότητας, ο αντίστοιχος καταναλωτής γνωρίζει το όνομα του κόμβου της σχετικής παρατήρησης, με αποτέλεσμα να έχει τη δυνατότητα να κάνει κατάλληλα ερωτήματα (queries) στην βάση του neo4j και να συλλέξει την πληροφορία που χρειάζεται. Με αυτήν φτιάχνει την δεδομένη χρονική στιγμή μια περίληψη για την δραστηριότητα, όπως αυτή μετρήθηκε από την εκάστοτε πηγή δεδομένων. Αυτή η περίληψη θα εισαχθεί τόσο σε συλλογή (collection 'activity') σε μη σχεσιακή βάση του χρήστη για τις περιλήψεις (user_1summary mongoDB), όσο και στον γράφο στην βάση δεδομένων γραφημάτων του χρήστη. Τα συμπεράσματα της δραστηριότητας εισάγονται σαν notificationDescription, ενώ η περίληψη (Summary) θεωρείται ενεργή κατευθείαν (triggered = true, notificationCreatedTimestamp = notificationTriggerTimestamp).

  Δεύτερον, οι δύο καταναλωτές για τον ύπνο (κύριο και μεσημεριανό), εκτελούν παρόμοια διαδικασία με τον καταναλωτή δραστηριοτήτων προκειμένου να δημιουργήσουν περίληψη για τον ύπνο του χρήστη. Ο λόγος που έγινε διαχωρισμός ανάμεσα στους δύο καταναλωτές γεγονότων ύπνου, είναι διότι αυτός που σχετίζεται με τον κύριο ύπνο επιτελεί ακόμη μία λειτουργία. Ειδικότερα, αναλαμβάνει να φτιάξει μια περίληψη για όλη τη μέρα του χρήστη. Κάνει επομένως, κατάλληλα ερωτήματα (queries) στην βάση neo4j και συλλέγει από όλες τις πηγές, δεδομένα που σχετίζονται με αξιοσημείωτες μετρικές κατά τη διάρκεια της μέρας. Βεβαίως, όταν πρόκειται

για καταστάσεις/δραστηριότητες που καταγράφηκαν από παραπάνω από μια συσκευές πρέπει να γίνει υπολογισμός της τελικής τιμής των ίδιων δεικτών ευημερίας. Στην περίπτωση μας, έχουμε το πολύ δύο πηγές δεδομένων που ενδέχεται να καταγράψουν μια ίδια κατάσταση ή δραστηριότητα. Επομένως, θα έχουμε το πολύ δύο διαφορετικές τιμές ανά μετρική και σαν αποτέλεσμα η τιμή που θα κρατήσουμε θα είναι ο μέσος όρος των δύο. Οι ειδοποιήσεις λοιπόν του ύπνου αποθηκεύονται στη συλλογή του ύπνου (collection 'sleep') στην μη σχεσιακή βάση των περιλήψεων του χρήστη (user1_summary mongoDB), ενώ οι ειδοποιήσεις για τα συμπεράσματα ημέρας αποθηκεύονται στην ίδια βάση σε συλλογή για τις περιλήψεις ημέρας (day).

Τέλος, για την παρακολούθηση των δεδομένων του βάρους, υπάρχει ακόμη ένας καταναλωτής ο οποίος παίρνει κατ΄ αντίστοιχο τρόπο σχετικό μήνυμα κάθε φορά που καταγράφεται το βάρος του χρήστη. Φτιάχνει λοιπόν, με τη σειρά του περίληψη για το βάρος με περιγραφή που εμπεριέχει τα δεδομένα που μετρήθηκαν (κιλά, δείκτης μάζας σώματος).

Σε αυτό το σημείο είναι εξέχουσας σημασίας να σημειωθεί ότι ακόμα μια λειτουργία που θέλουμε να υποστηρίζει η Υπηρεσία Δημιουργίας Περιλήψεων, είναι να υπολογίζει συμπεράσματα για μετρικές για την τρέχουσα ημέρα του ατόμου. Με άλλα λόγια, επιθυμούμε να καταγράφει μια περίληψη για την τρέχουσα μέρα την οποία θα μπορεί να παρέχει με τα μέχρι τότε δεδομένα. Φυσικά, κάθε φορά που συμβαίνει ένα γεγονός το οποίο επηρεάζει μια μετρική που παρουσιάζεται και στα τελικά συμπεράσματα, είναι αναγκαίο να ανανεωθεί το JSON-LD της τρέχουσας μέρας. Αυτή η πληροφορία αποθηκεύεται στην ίδια συλλογή με τις περιλήψεις ημέρας, ωστόσο για να διαχωρίζεται από τις άλλες περιλήψεις, προσθέσαμε ένα πεδίο isFinal στο JSON-LD στο οποίο δίνουμε την τιμή false αφού δεν έχει ολοκληρωθεί ακόμα η μέρα.

Έχοντας αναλύσει λοιπόν, και αυτή την υπηρεσία, ολοκληρώνουμε το επίπεδο επεξεργασίας των δεδομένων. Στις επόμενες υπηρεσίες θα μελετήσουμε πώς χρησιμοποιείται η εμπλουτισμένη πληροφορία (προειδοποιήσεις, περιλήψεις) από το επίπεδο αξιοποίησης.

- **Υπηρεσία Διαχείρισης Προειδοποιήσεων**

  Με την παρούσα υπηρεσία, πραγματοποιείται ανά λεπτό, έλεγχος των προειδοποιήσεων που έχουν παραχθεί από το κατώτερο επίπεδο. Συγκεκριμένα, η Υπηρεσία Διαχείρισης Προειδοποιήσεων παρακολουθεί τις εν δυνάμει προειδοποιήσεις (triggered=false) και τις μετατρέπει σε ενεργές όταν αυτό είναι απαραίτητο (δηλαδή όταν notificationTriggerTimestamp = <current_time>). Ακόμα, παρακολουθεί ποιές προειδοποιήσεις είναι ενεργές και δεν έχουν αντιμετωπιστεί από τον χρήστη. Σε αυτές ανανεώνει την χρονική στιγμή αναπαραγωγής (notificatioTriggerTimestamp), προκειμένου να ενημερωθεί ξανά ο χρήστης μετά από μία ώρα, εφόσον η προειδοποίηση παραμείνει ενεργή μέχρι τότε. Συνοψίζοντας, το έργο που επιτελεί η υπηρεσία είναι να διαχειρίζεται τις τρέχουσες προειδοποιήσεις του χρήστη στη σχετική βάση (user1_alert moongoDB).

- **Υπηρεσία Διαχείρισης Αιτημάτων για Προειδοποιήσεις**

  Η υπηρεσία αυτή δέχεται αιτήματα από την εφαρμογή σε συγκεκριμένες διευθύνσεις (URLs). Ανάλογα με τις προειδοποιήσεις που ζητούνται, γίνεται κατάλληλο

ερώτημα προς τη μη σχεσιακή βάση και συγκεντρώνονται όλες οι προειδοποιήσεις που συγκροτούν την απάντηση. Δημιουργείται επομένως ένα JSON, το οποίο αποστέλλεται πίσω χρησιμοποιώντας μεθόδους κρυπτογράφησης της ομώνυμης υπηρεσίας.

- **Υπηρεσία Διαχείρισης Αιτημάτων για Περιλήψεις**

  Παρομοίως, η λειτουργία της Υπηρεσίας Διαχείρισης Αιτημάτων για Περιλήψεις δέχεται από την εφαρμογή αιτήματα σε συγκεκριμένες διευθύνσεις (URLs) και με βάσει αυτά σχηματίζει ένα τελικό JSON που περιλαμβάνει τις περιλήψεις που ζητήθηκαν. Κρυπτογραφώντας αυτό το τελικό JSON παράγει την απάντηση που θα σταλεί πίσω στην εφαρμογή.

- **Υπηρεσία Διαχείρισης Χρηστών**

  Αυτή η υπηρεσία, δέχεται από την εφαρμογή δεδομένα που σχετίζονται με την εγγραφή και την σύνδεση χρηστών.

  Κατά την εγγραφή ενός νέου χρήστη, γνωρίζουμε ότι εφόσον ολοκληρωθεί η διαδικασία εγγραφής του στον Εξυπηρετητή Αυθεντικοποίησης Χρηστών, πρέπει να ενημερωθεί και ο εξυπηρετητής μας για να προβεί στις κατάλληλες ενέργειες. Συγκεκριμένα, για έναν νέο χρήστη καθίσταται αναγκαίο να ανταλλαχθεί με ασφάλεια ένα κλειδί για την μετέπειτα κρυπτογραφημένη επικοινωνία. Έτσι η Υπηρεσία Διαχείρισης Χρηστών, λαμβάνει από έναν νέο χρήστη ένα αίτημα που περιλαμβάνει το ταυτοποιητικό του στην Υπηρεσία Αυθεντικοποίησης, ένα πιστοποιητικό και ένα δημόσιο κλειδί. Αφότου η υπηρεσία πιστοποιήσει τον χρήστη μέσω της Αρχής Πιστοποίησης (Firebase), ξεκινά την διαδικασία για να παράξει ένα κλειδί για την κρυπτογραφημένη επικοινωνία (AES-128) με τον συγκεκριμένο χρήστη. Αυτό το κλειδί το κρυπτογραφεί με το δημόσιο κλειδί του χρήστη στα πλαίσια του αλγορίθμου RSA που χρησιμοποιείται για την ανταλλαγή. Το αποστέλλει τελικά στην εφαρμογή και από εδώ και πέρα ακολουθεί η κρυπτογραφημένη με AES-128 επικοινωνία. Σε επόμενο στάδιο η υπηρεσία δέχεται και τα προσωπικά στοιχεία του χρήστη τα οποία και αποθηκεύει σε μη σχεσιακή βάση για τους χρήστες (users mongoDB) μάζι με το κοινό τους κλειδί. Σημειώνεται ότι ο τρόπος με τον οποίο αποθηκεύονται τα προσωπικά δεδομένα του χρήστη είναι σε μορφή JSON-LD που βασίζεται στο οντολογικό σχήμα.

  Κατά την σύνδεση τώρα, όταν ο εξυπηρετητής λαμβάνει μήνυμα ότι ένας χρήστης συνδέθηκε επιτυχώς, η υπηρεσία αυτή αντλεί από την βάση το κοινό τους κλειδί προκειμένου να πραγματοποιηθεί η εδραίωση μιας κρυπτογραφημένης επικοινωνίας. Ομοίως αν ο χρήστης αποσυνδεθεί η υπηρεσία ενημερώνεται με κατάλληλο μήνυμα.

- **Υπηρεσία Πιστοποίησης Χρήστη**

  Όπως προαναφέρθηκε, υλοποιεί την επιβεβαίωση του πιστοποιητικού του χρήστη, που λαμβάνεται σαν JSON Web Token (jwt), καλώντας κατάλληλη συνάρτηση (verifyIdToken) που παρέχει ο Εξυπηρετητή Αυθεντικοποίσης.

- **Υπηρεσία Ανταλλαγής Κλειδιού**

  Με αυτήν την υπηρεσία, υλοποιούνται οι λειτουργίες που απαιτούνται για την παραγωγή ενός νέου κλειδιού για τον αλγόριθμο AES-128 που θα χρησιμοποιηθεί. Γενικά,

η υπηρεσία χρησιμοποιείται από την Υπηρεσία Διαχείρισης Χρηστών προκειμένου να ανταλλαχθεί το κοινό κλειδί μεταξύ εφαρμογής και εξυπηρετητή.

- **Υπηρεσία Κρυπτογράφησης**

  Η παρούσα υπηρεσία αναλαμβάνει την κρυπτογράφηση εξερχομένων μηνυμάτων και την αποκρυπτογράφηση εισερχομένων. Όταν πρόκειται για επικοινωνία με την Γεννήτρια Δεδομένων, χρησιμοποιείται το κλειδί που έχει συμφωνηθεί μεταξύ τους από πριν, ενώ όταν πρόκειται για επικοινωνία με την εφαρμογή χρησιμοποιείται το κλειδί που ανταλλάχθηκε κατά την εγγραφή του εκάστοτε χρήστη.

Συνοψίζοντας, μελετήθηκε διεξοδικά ο τρόπος με τον οποίο υλοποιήθηκαν οι υπηρεσίες του εξυπηρετητή. Μπορούμε, τώρα να προχωρήσουμε στην υλοποίηση της εφαρμογής για να έχουμε την ολοκληρωμένη εικόνα για την υλοποίηση του συστήματός μας.

5.3 Εφαρμογή

Το τελευταίο συστατικό στοιχείο του συστήματος μας είναι η εφαρμογή. Με τη χρήση της, έχουμε την δυνατότητα να δουμε πώς αξιοποιείται η τελική πληροφορία που παράγει ο εξυπηρετητής μετά την συλλογή των ακατέργαστων δεδομένων από τις πηγές. Είναι σημαντικό, επομένως, να αναλύσουμε τον τρόπο υλοποίησης της, σχετικά τόσο με τις οθόνες που παρέχονται στον τελικό χρήστη, όσο και με τις λειτουργίες κάθε υπηρεσίας της εφαρμογής.

5.3.1 Οθόνες εφαρμογής

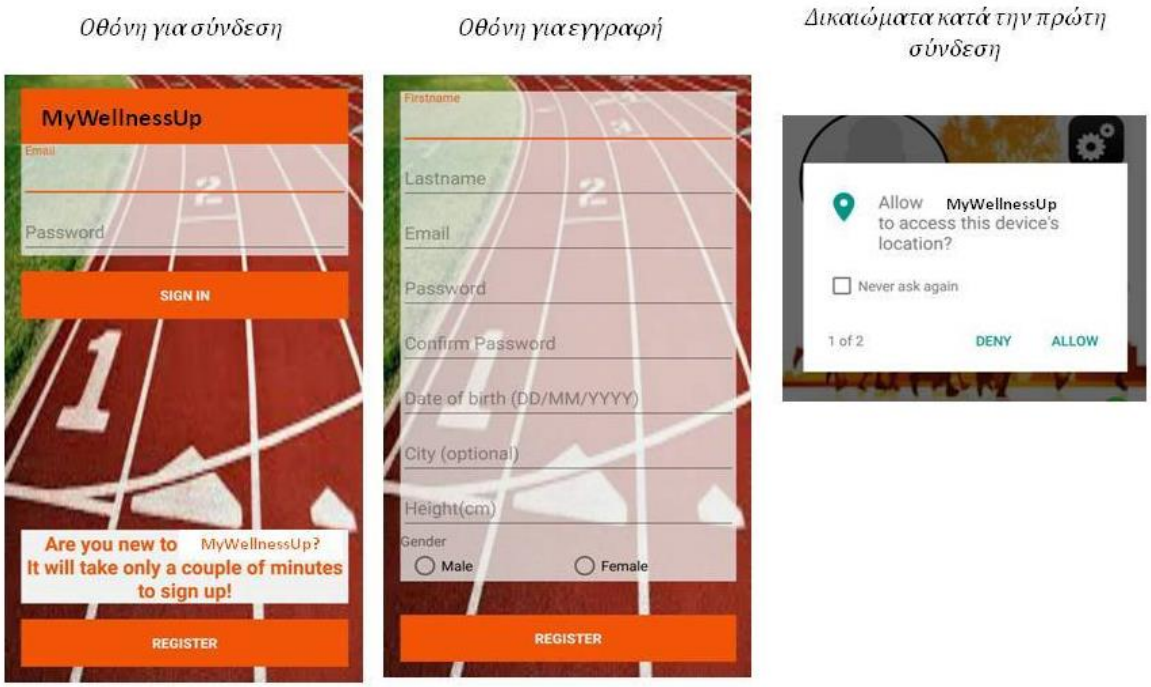
Πρώτα απ΄ όλα, θα αναφερθούμε στις τελικές οθόνες του χρήστη, όπως αυτές υλοποιήθηκαν με βάση τον σχεδιασμό που έγινε στο κεφάλαιο 4.

Αρχικά, κατά το άνοιγμα της εφαρμογής, εμφανίζεται η **οθόνη σύνδεσης**. Όπως φαίνεται και στο Σχήμα 5.3, ο χρήστης καλείται να συμπληρώσει τη διεύθυνση ηλεκτρονικού ταχυδρομείου και τον κωδικό του προκειμένου να εισέλθει πατώντας το σχετικό κουμπί (Login). Αν ωστόσο ο χρήστης επιθυμεί να εγγραφεί πατάει το κουμπί Register για να μεταβεί στην οθόνη εγγραφής.

Η **οθόνη εγγραφής** τώρα, ζητάει την εισαγωγή προσωπικών στοιχείων του χρήστη για να δημιουργηθεί το προφίλ του. Συγκεκριμένα, υπάρχουν πεδία για συμπλήρωση ονόματος, επωνυμου, διεύθυνσης ηλεκτρονικού ταχυδρομίου, κωδικού, ημερομηνίας γέννησης, πόλης κατοικίας, φύλου και ύψους. Όταν η φόρμα συμπληρωθεί κανονικά, μπορεί να σταλεί στη αρμόδια υπηρεσία της εφαρμογής με το κουμπί Register.

Κατά την εγγραφή τώρα του χρήστη, ζητείται από την εφαρμογή **δικαίωμα για πρόσβαση στον αποθηκευτικό χώρο του κινητού τηλεφώνου και στην τοποθεσία του χρήστη**, εφόσον εκείνος το επιθυμεί.

Με την σύνδεση του χρήστη, παίρνει σειρά η **κύρια οθόνη**, η οποία χωρίζεται σε **τρία κομμάτια (fragments)**, όπως φαίνεται στο Σχήμα 5.4. Το πρώτο είναι το προφίλ του χρήστη, το οποίο εμφανίζεται αυτόματα κατά την σύνδεση, το δεύτερο είναι η οθόνη παρακολούθησης δεικτών ευημερίας (monitoring) και το τρίτο η οθόνη των στόχων (goals).



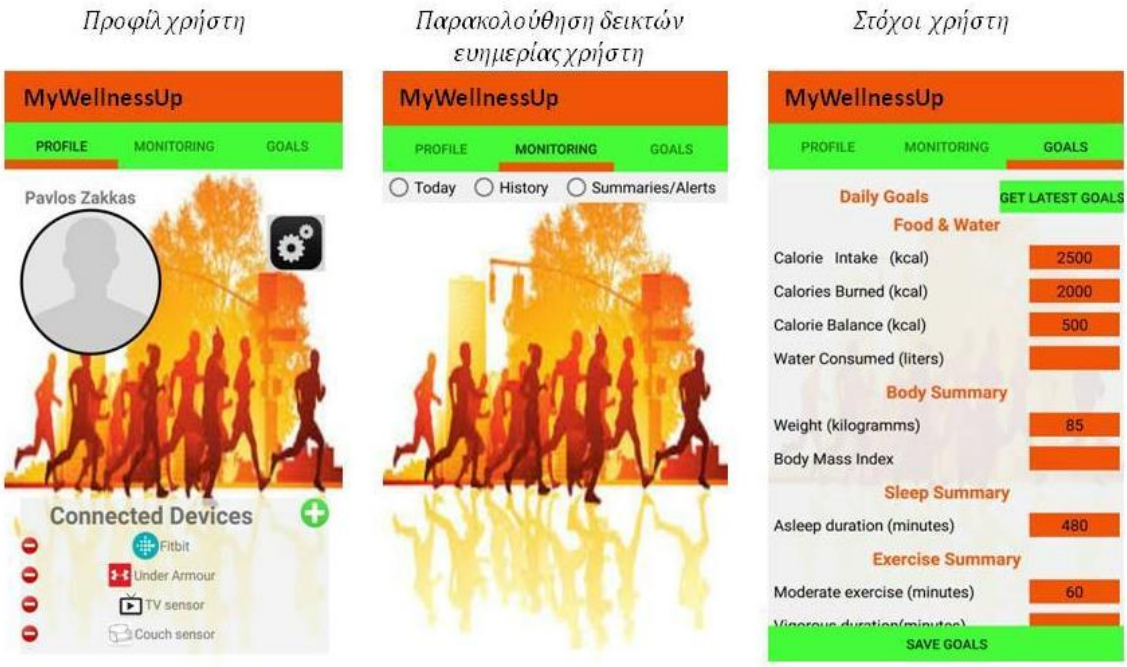
Σχήμα 5.3: Αρχικές οθόνες εφαρμογής

Με την χρήση καρτέλων (tabs) επιτρέπεται η πλοήγηση ανάμεσα σε αυτά τα συστατικά της κύριας οθόνης.

Όσο αφορά την **οθόνη προφίλ**, βλέπουμε ότι περιλαμβάνει μια εικόνα στην οποία ο χρήστης μπορεί να εισάγει μια δική του φωτογραφία. Ακόμη, περιέχει σαν πληροφορία τις πηγές δεδομένων που έχει συνδέδει ο χρήστης και του δίνεται η δυνατότητα να εισάγει άλλες ή να αφαιρέσει ήδη υπάρχουσες. Στη περίπτωσή μας αυτή η λίστα είναι στατική αφού οι συσκευές που έχουμε συνδέσει είναι προκαθορισμένες (Fitbit, UnderArmour, couch_sensor, tv_sensor). Επιπλέον υπάρχει και ένα μενού επιλογών όπου ο χρήστης μπορεί να επιλέξει μια από τις εξής ενέργειες: άλλαγή εικόνας προφίλ (Change profile photo), προτιμήσεις (Preferences), αποσύνδεση (Sign out). Η πρώτη επιλογή, τον κατευθύνει στον αποθηκευτικό χώρο του κινητού για να επιλέξει φωτογραφία. Η δεύτερη του βγάζει ένα μενού για καθορισμό των προτιμήσεων του, όπως το αν θέλει η τοποθεσία του να χρησιμοποιείται από την εφαρμογή. Τέλος, η τρίτη επιλογή τον αποσυνδέει από την εφαρμογή.

Σχετικά, τώρα με την **οθόνη παρακολούθησης των δεικτών ευημερίας** (monitoring) του δίνεται η δυνατότητα για τρεις επιλογές. Η μία του παρουσιάζει τα μέχρι εκείνη τη στιγμή συμπεράσματα της τρέχουσας ημέρας. Η άλλη του παρουσιάζει την περίληψη κάποιας παλιάς μέρας σαν ιστορικό, και τέλος η τρίτη, του δείχνει πίνακες σχετικούς με περιλήψεις κατάστασης/ δραστηριότητας και με προειδοποιήσεις. Οι επιλογές αυτές αναλύονται στην συνέχεια.

Η τρίτη επιμέρους οθόνη της κύριας οθόνης αφορά τους **στόχους του χρήστη** (goals). Εκεί υπάρχει μια λίστα με τους δείκτες ευημερίας που επιλέγουμε να παρουσιάζουμε στο χρήστη στα συμπεράσματα ημέρας. Για κάθε έναν από αυτούς τους δείκτες, ο χρήστης μπορεί να εισάγει τις δικές του τιμές-στόχους. Με την επιλογή Get Latest Goals του παρουσιάζονται οι στόχοι που είχε θέσει την τελευταία φορά, ενώ με την επιλογή Save Goals αποθηκεύονται



Σχήμα 5.4: Κύριες οθόνες εφαρμογής

οι τιμές που έθεσε σαν στόχους. Αυτοί οι στόχοι θα ισχύουν από την τρέχουσα μέρα και μετά.

Εμβαθύνοντας τώρα στις επιλογές της οθόνης παρακολούθησης μετρικών (Monitoring), παρατίθεται το Σχήμα 5.5. Πρώτα, μπορούμε να αναφερθούμε στην επιλογή **Today** που αφορά την **τρέχουσα ημέρα**. Εκεί παρουσιάζονται όλοι οι δείκτες ευημερίας που συλλέγονται από τον εξυπηρετητή, ανά κατηγορία. Βλέπουμε δηλαδή σε μια λίστα, μετρικές ζωτικής σημασίας, μετρικές του σώματος του χρήστη, της συνολικής αθλητικής του δραστηριότητας και τέλος μετρικές ξεχωριστά για κάθε δραστηριότητα που μπορεί να έκανε. Αντίστοιχα για κάθε δείκτη φαίνεται η τιμή που έχει μέχρι την συγκεκριμένη στιγμή της μέρας και δίπλα από αυτή τη τιμή φαίνεται ο στόχος που είχε θέσει ο χρήστης. Τέλος, ειδικά για τον καρδιακό παλμό υπάρχει η επιλογή View με την οποία δίνεται η δυνατότητα παρακολούθησης, μέσω γραφικής, των τιμών που έλαβε την τελευταία ώρα. Βέβαια, ο χρήστης έχει και την επιλογή Update για ανανέωση των δεδομένων της τρέχουσας μέρας.

Όσον αφορά την οθόνη του **ιστορικού (History)**, παρουσιάζεται ακριβώς με την ίδια λογική, το σύνολο των μετρικών με τις τιμές τους και τους στόχους, που αντιστοιχούν στην τελευταία ολοκληρωμένη μέρα του χρήστη, δηλαδή στη χθεσινή. Η μόνη μετρική που δεν φαίνεται είναι ο καρδιακός παλμός, καθώς δεν έχει ιδιαίτερο νόημα για προηγούμενη μέρα. Επιπλέον, ενδιαφέρον παρουσιάζει και η επιλογή για αναζήτηση περίληψης ημέρας. Δηλαδή, ο χρήστης μπορεί να ανατρέξει στο ιστορικό του, και να ζητήσει τα συμπεράσματα που είχαν συλλεγεί για μια προηγούμενη μέρα.

Τέλος, θα αναφερθούμε στη τρίτη επιλογή της οθόνης παρακολούθησης που αφορά τις **περιλήψεις κατάστασης ή δραστηριότητας και τις προειδοποιήσεις (Summaries/ Alerts)**. Εκεί υπάρχουν τρεις πίνακες. Ο πρώτος περιέχει τις τελευταίες

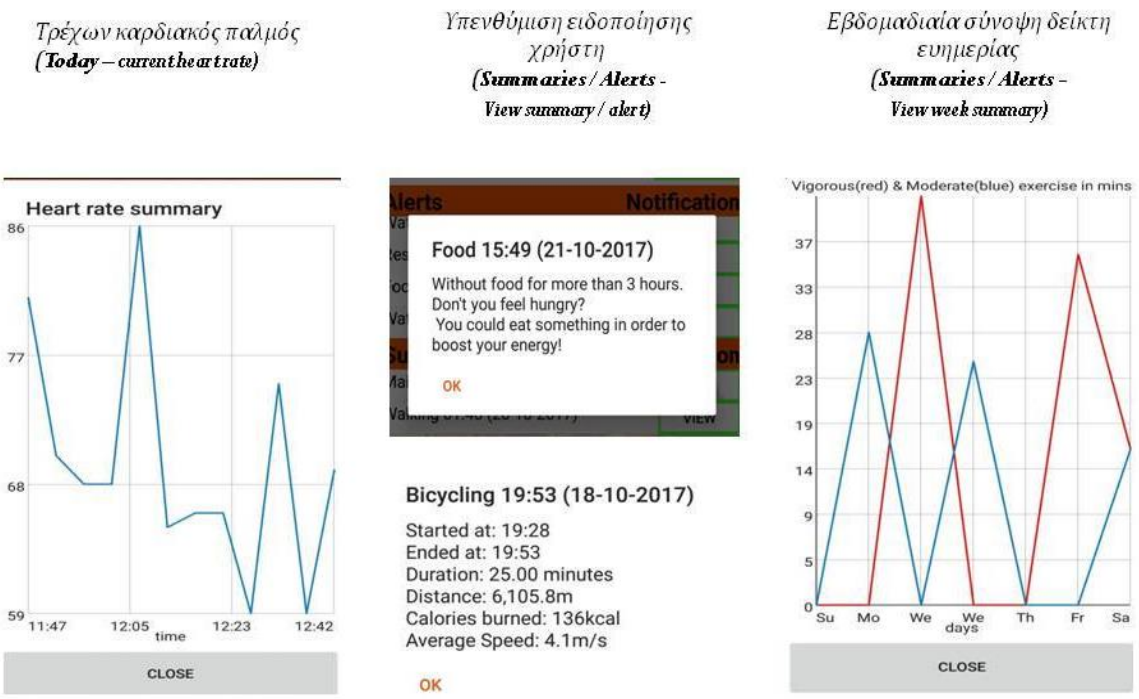
Σχήμα 5.5: Βασικές λειτουργίες στην οθόνη παρακολούθησης

δέκα προειδοποιήσεις που εμφανίστηκαν στον χρήστη ανεξαρτήτως τύπου (φαγητό, νερό, καθιστική ζωή, καρδιακός παλμός). Ο δεύτερος περιέχει τις τελευταίες δέκα περιλήψεις είτε κατάστασης είτε δραστηριότητας που είναι άξιες αναφοράς (κύριος ύπνος, μεσημεριανός ύπνος, έντονο περπάτημα, τρέξιμο, ποδηλασία). Σε αυτούς τους δύο πίνακες έχει την δυνατότητα να πατήσει την επιλογή View και να δει την αντίστοιχη ειδοποίηση που του είχε εμφανιστεί σε πραγματικό χρόνο στην γραμμή εργασιών του κινητού. Τέλος, ο τρίτος πίνακας περιλαμβάνει κάποιους δείκτες (θερμίδες, άθληση, νερό, ύπνος, βήματα), που είναι άξιοι μελέτης σε εβδομαδιαία κλίμακα. Με την επιλογή View λοιπόν, παρατίθεται στον χρήστη μια γραφική παράσταση με τις διακυμάνσεις αυτού του δείκτη ευημερίας ανά μέρα σε βάθος μιας εβδομάδας.

Αξίζει να παρατεθούν σε αυτό το σημείο, συγκεκριμένα αποτελέσματα που εμφανίστηκαν στην εφαρμογή, κατά την λειτουργία της με τα προσομοιωτικά δεδομένα, που φτιάξαμε στο αρχικό στάδιο υλοποίησης.

Αρχικά, στην καρτέλα παρακολούθησης (Monitoring), επιμέρους λειτουργίες που παρατηρούνται φαίνονται στο Σχήμα 5.6. Έχουμε επομένως, ένα παράδειγμα με την **γραφική παράσταση του καρδιακού παλμού** της τελευταίας ώρας. Ένα ακόμα παράδειγμα αφορά την **υπενθύμιση της ειδοποίησης** που σχετίζεται με μια προειδοποίηση (φαγητό) ή με μια περίληψη δραστηριότητας (ποδήλατο) στην σχετική οθόνη. Τέλος, ένα τρίτο παράδειγμα είναι η **γραφική παράσταση της εβδομαδιαίας διακύμανσης μιας μετρικής** (αθλητική άσκηση).

Σαφώς, μια σημαντική λειτουργία της εφαρμογής είναι οι **ειδοποιήσεις σε πραγματικό χρόνο, είτε προειδοποιήσεων, είτε περιλήψεων**. Οι ειδοποιήσεις φυσικά θα γεμισουν και τους σχετικούς πίνακες στην οθόνη της παρακολούθησης, ωστόσο



Σχήμα 5.6: Επιμέρους Λειτουργίες στην οθόνη παρακολούθησης

πρώτα θα εμφανιστούν στη γραμμή εργασιών του κινητού. Παραδείγματα για τις διαφορετικού τύπου ειδοποιήσεις φαίνονται στο Σχήμα 5.7 και στο Σχήμα 5.8.

5.3.2 Υπηρεσίες εφαρμογής

Σχετικά με τις υπηρεσίες της εφαρμογής, αυτή η ενότητα θα αφιερωθεί στον τρόπο με τον οποίο υλοποιήθηκαν οι λειτουργίες τους. Έτσι ανά υπηρεσία της αρχιτεκτονικής της εφαρμογής έχουμε τις παρακάτω υλοποιήσεις.

- **Υπηρεσία Συλλογής Δεδομένων- Εμπλουτισμένης Πληροφορίας**

  Αυτή η υπηρεσία αναλαμβάνει να στέλνει τα αιτήματα της εφαρμογής σε κατάλληλες διευθύνσεις του εξυπηρετητή, αφού πρώτα τα κρυπτογραφήσει χρησιμοποιώντας την Υπηρεσία Κρυπτογράφησης. Αντίστροφα, αναλαμβάνει και την λήψη δεδομένων από τον εξυπηρετητή τα οποία παρέχει στις υπηρεσίες που τα ζήτησαν αφού τα αποκρυπτογραφήσει.

Σχήμα 5.7: Παραδείγματα προειδοποιήσεων χρήστη

Σχήμα 5.8: Παραδείγματα περιλήψεων χρήστη

- **Υπηρεσία Διαχείρισης Ιστορικού Περιλήψεων**

  Πρόκειται για μια υπηρεσία της εφαρμογής που τρέχει στο υπόβαθρο (background), δηλαδή ακόμα και όταν ο χρήστης έχει κλειστή την εφαρμογή, εφόσον όμως έχει παραμείνει συνδεδεμένος. Συγκεκριμένα, η υπηρεσία καλείται, όταν η εφαρμογή πρέπει να λάβει από τον εξυπηρετητή την περίληψη μιας μέρας του χρήστη η οποία έχει ολοκληρωθεί. Σαν αποτέλεσμα, λαμβάνει τα συμπεράσματα που έχει υπολογίσει ο εξυπηρετητής με την σχετική υπηρεσία του, και τα αποθηκεύει σε αντίστοιχη μη σχεσιακή βάση (δηλαδή σε συλλογή day_summary της mongoDB βάσης app_user1 του χρήστη). Ακόμα ενημερώνει κατάλληλα την Υπηρεσία Αξιοποίησης Ιστορικού Περιλήψεων για την νέα περίληψη ημέρας που έλαβε. Επίσης δημιουργεί μέσω της Υπηρεσίας Δημιουργίας Ειδοποιήσεων μια ειδοποίηση για την γραμμή εργασιών του χρήστη προκειμένου να του παρουσιαστούν τα αποτελέσματα χωρίς να χρειαστεί να ανοίξει την εφαρμογή.

- **Υπηρεσία Διαχείρισης Περιλήψεων Κατάστασης**

  Είναι επιπλέον μία υπηρεσία που τρέχει, ακόμα και όταν η εφαρμογή είναι κλειστή. Ειδικότερα, ενεργοποιείται κάθε 10 λεπτά και στέλνει, μέσω της Υπηρεσίας Συλλογής, αιτήματα στον εξυπηρετητή για λήψη των περιλήψεων κατάστασης ή δραστηριότητας. Για αυτά τα αιτήματα ορίζει τα χρονικά όρια μέσα στα οποία ζητάει να ανήκει η περίληψη. Σαν άνω όριο ορίζει όπως είναι λογικό, την εκάστοτε χρονική στιγμή που στέλνει το αίτημα, ενώ σαν κάτω όριο ορίζει την τελευταία χρονική στιγμή που έλαβε σχετική περίληψη. Με αυτό το τρόπο εξασφαλίζεται ότι θα λάβει όλα τα σχετικά δεδομένα που έχει υπολογίσει ο εξυπηρετητής. Σημειώνεται επίσης, ότι δεν αποτελεί πρόβλημα το να καθυστερήσει να λάβει μια περίληψη, καθώς δεν πρόκειται για δεδομένα που απαιτούνται να παρουσιαστούν τόσο άμεσα στον χρήστη. Γι΄ αυτό το λόγο επιλέχθηκε και το διάστημα των 10 λεπτών.

  ΑΦού λοιπόν, ληφθούν τα δεδομένα αυτά, πρέπει η εφαρμογή να εφαρμόσει μια συγκεκριμένη λογική προκειμένου να φτιάξει μια τελική περίληψη κατάστασης ή δραστηριότητας την οποία και θα παρουσιάσει στον χρήστη. Υπενθυμίζουμε ότι ενδέχεται να υπάρχουν πάνω από μια περιλήψεις για την ίδια κατάσταση/δραστηριότητα, επομένως μπορεί να υπάρχουν διαφορετικές τιμές για μια συγκεκριμένη μετρική. Στην προσομοίωση μας, έχουμε το πολύ δύο συσκευές (Fitbit, UnderArmour) που να καταγράφουν ίδιες καταστάσεις. Οπότε θα έχουμε το πολύ δύο τιμές και γι΄ αυτό η τελική τιμή που θα κρατήσουμε, θα είναι ο μέσος όρος τους, καθώς δεν μπορούμε να κάνουμε κάποιον διαχωρισμό (όπως θα γινόταν αν είχαμε παραπάνω μετρήσεις για τον ίδιο δείκτη). Κατασκευάζουμε, εν τέλει, μια περίληψη με τις τελικές τιμές ανά δείκτη ευημερίας. Αυτήν θα την αποθηκεύσουμε σε σχετική συλλογή (notification_summary) στην μη σχεσιακή βάση που αντιστοιχεί στον χρήστη της εφαρμογής (app_user1 mongoDB). Τέλος, ενημερώνεται κατάλληλα και η Υπηρεσία Δημιουργίας Ειδοποιήσεων για να κατασκευάσει μια στοχευμένη ειδοποίηση στο κινητό του χρήστη.

- **Υπηρεσία Διαχείρισης Προειδοποιήσεων**

  Αποτελείται από ένα σύνολο υπηρεσιών που τρέχουν ανεξάρτητα από το αν η εφαρμογή είναι ανοικτή ή κλειστή. Κάθε μια από αυτές, αναλαμβάνει την διαχείριση

προειδοποιήσεων κάθε είδους, όπως αυτές δημιουργούνται από τον εξυπηρετητή. Έχουμε, επομένως, επιμέρους υπηρεσίες για προειδοποιήσεις φαγητού, νερού, καθιστικής ζωής και καρδιακού παλμού.

Η υπηρεσία για το φαγητό και το νερό, ενεργοποιείται κάθε 5 λεπτά, αφού είναι πιο άμεση από πριν η ανάγκη για παρουσίαση μιας προειδοποίησης στον χρήστη. Το αίτημα που στέλνει συμπεριλαμβάνει τα χρονικά όρια της προειδοποίησης που καθορίζονται από την στιγμή που συνδέθηκε ο χρήστης μέχρι την εκάστοτε χρονική στιγμή του αιτήματος. Εδώ δεν μας ενδιαφέρουν προειδοποιήσεις που δημιουργήθηκαν κατά τη διάρκεια που ο χρήστης ήταν αποσυνδεδεμένος, αφού δεν υπάρχει ιδιαίτερος λόγος να τον ενημερώσουμε εκ των υστέρων. Σαν αποτέλεσμα του αιτήματος, συγκεντρώνονται ανά κατηγορία (φαγητό, νερό) όλες οι προειδοποιήσεις. Σε αυτό το σημείο γίνεται μια επεξεργασία της πληροφορίας για να φτιαχτεί η τελική προειδοποίηση. Ωστόσο στην προσομοίωση μας, έχουμε μόνο το Fitbit για την καταγραφή φαγητού και νερού, επομένως σε μια συγκεκριμένη χρονική στιγμή θα έχουμε το πολύ μία προειδοποίηση ανά κατηγορία. Σε αυτή θα βασιστεί και η υπηρεσία για να φτιάξει μια τελική προειδοποίηση, η οποία θα εμφανιστεί και στην οθόνη του κινητού.

Δεύτερον, η υπηρεσία για τον έλεγχο καθιστικής ζωής, αναλαμβάνει δράση και αυτή κάθε 5 λεπτά, πραγματοποιώντας αιτήματα με τελείως ανάλογο τρόπο με την προαναφερθείσα υπηρεσία. Λαμβάνει, έτσι, προειδοποιήσεις για καθιστική ζωή (υπερβολική ξεκούραση), οι οποίες και σε αυτή τη περίπτωση θα είναι το πολύ μία ανά κατάσταση ξεκούρασης του χρήστη, αφού μόνο το Fitbit έχει δυνατότητα τέτοιας καταγραφής. Για την διαχείριση αυτής της πληροφορίας η υπηρεσία συνδυάζει πρόσθετα δεδομένα με σκοπό της εξαγωγή πιο στοχευμένων συμπερασμάτων. Ειδικότερα, λαμβάνει με σχετικό αίτημα στον εξυπηρετητή, την εκάστοτε κατάσταση του χρήστη, στην οποία φαίνεται και αν κάθεται στον καναπέ ή αν βλέπει τηλεόραση. Με βάση αυτή τη πληροφορία από τους αισθητήρες έξυπνου σπιτιού, μπορεί να κατασκευάσει πιο εύστοχα μηνύματα. Επιπλέον, λαμβάνει και δεδομένα για τον καιρό που επικρατεί, είτε στην περιοχή που βρίσκεται ο χρήστης, αν έχουμε πρόσβαση στην τοποθεσία του, είτε στην πόλη διαμονής που συμπλήρωσε κατά την εγγραφή. Ανάλογα επομένως με τον καιρό, ενδέχεται να εμφανιστεί μήνυμα που τον παροτρύνει να βγει για μια βόλτα ή για μια αθλητικη δραστηριότητα σε περίπτωση καλού καιρού ή του προτείνει μια δραστηριότητα στο σπίτι σε περίπτωση κακοκαιρίας. Με τον συνδυασμό, δηλαδή, διαφορετικού είδους πληροφορίας, πετυχαίνουμε την σωστή καθοδήγηση του χρήστη και την κινητοποίηση του προκειμένου να ελαττώσει τον καθιστικό τρόπο ζωής του.

Τρίτον, έχουμε και την επιμέρους υπηρεσία για προειδοποιήσεις καρδιακού παλμού. Αυτή, λοιπόν, ενεργοποιείται κάθε λεπτό, καθώς πρόκειται για μετρική ζωτικής σημασίας. Με αντίστοιχο τρόπο που χρησιμοποιούν και οι παραπάνω υπηρεσίες, λαμβάνει από τον εξυπηρετητή τις σχετικές προειδοποιήσεις. Εδώ, ενδέχεται να έχουμε δύο προειδοποιήσεις για μια δεδομένη χρονική στιγμή, διότι υπάρχουν δύο συσκευές (Fitbit, UnderArmour), που μετράνε τον παλμό. Συγκεκριμένα, αν η προειδοποίηση είναι μία, σημαίνει ότι η τιμή που κατέγραψε η άλλη συσκευή ήταν εντός λογικών ορίων. Έτσι, επιλέγουμε να απορρίψουμε την αρχική προειδοποίηση αφού ενδέχεται να προέρχεται από λάθος μέτρηση της συσκευής. Φυσικά σε ένα σύστημα με πολλές συσκευές η διαχείριση του καρδιακού παλμού θα γινόταν με πιο αξιόπιστο τρόπο.

Ωστόστο στην προσομοίωση μας, το μόνο που μπορούμε να κάνουμε είναι να θεωρούμε μια προειδοποίηση έγκυρη, εφόσον προέρχεται και από τις δύο πηγές. Ύστερα λοιπόν από τον κατάλληλο έλεγχο, διαμορφώνεται όπου είναι αναγκαίο (δηλαδή όπου και οι δύο συσκευές κατέγραψαν τιμή εκτός ορίων), μια τελική προειδοποίηση προκειμένου να εμφανιστεί σχετικό μήνυμα στην οθόνη.

Ολοκληρώνοντας, σημειώνεται ότι οι τελικές προειδοποιήσεις αποθηκέυονται σε σχετική συλλογή (notification_alert) της μη σχεσιακής βάσης του χρήστη (app_user1 mongoDB). Παράλληλα ενημερώνεται και η Υπηρεσία Αξιοποίησης προειδοποιήσεων για να ανανεώσει τις κατάλληλες οθόνες, ενώ επίσης γίνεται χρήση και της Υπηρεσίας Δημιουργίας Ειδοποιήσεων για να ενημερωθεί έγκαιρα ο χρήστης.

- **Υπηρεσία Αξιοποίησης Ιστορικού Περιλήψεων**

  Η υπηρεσία αυτή αναλαμβάνει να ανανεώσει την σχετική οθόνη παρακολούθησης ιστορικού (History), είτε όταν ενημερωθεί από την αντίστοιχη υπηρεσία κατώτερου επιπέδου ότι λήφθηκε περίληψη ημέρας, είτε όταν δεχθεί αίτημα από την Διεπαφή Ιστορικού για αναζήτηση περίληψης συγκεκριμένης μέρας. Την πληροφορία που χρειάζεται την αντλεί από την σχετική συλλογή περιλήψεων μέρας (day_summary) στην μη σχεσιακή βάση που κρατάει η εφαρμογή.

- **Υπηρεσία Αξιοποίησης Περιλήψεων Εβδομάδας**

  Πρόκειται για την υπηρεσία που χρησιμοποιείται, όταν ο χρήστης κάνει αίτημα από την Διεπαφή Περιλήψεων/ Προειδοποιήσεων-Προτάσεων για γραφική αναπαράσταση των τιμών ενός δείκτη ευημερίας ανά μέρα. Ο ρόλος, επομένως, της υπηρεσίας είναι να συλλέξει τα σχετικά δεδομένα από την βάση των περιλήψεων μέρας και να υλοποιήσει την γραφική παράσταση για την διεπαφή.

- **Υπηρεσία Αξιοποίησης Περιλήψεων Κατάστασης**

  Πρόκειται για μια υπηρεσία που αναλαμβάνει δράση τόσο όταν ανοίγει η εφαρμογή, προκειμένου να πάρει από την βάση των περιλήψεων ('notification_summary' collection) τις τελευταίες δέκα εγγραφές, όσο και όταν δημιουργηθεί μια περίληψη κατάστασης σε πραγματικό χρόνο από την Υπηρεσία Διαχείρισης Περιλήψεων. Ανανεώνει, λοιπόν, με βάση τα παραπάνω την σχετική διεπαφή της εφαρμογής.

- **Υπηρεσία Αξιοποίησης Προειδοποιήσεων**

  Ακολουθώντας ακριβώς την ίδια λογικά με την προαναφερθείσα υπηρεσία ανανεώνει την αντίστοιχη διεπαφή. Δηλαδή και αυτή η υπηρεσία, συγκεντρώνει τις τελευταίες 10 προειδοποιησεις από την σχετική βάση ('notification_alert' collection), ενώ επίσης, ενημερώνεται σε πραγματικό χρόνο για τυχόν δημιουργία νέων προειδοποιήσεων με σκοπό την αξιοποίηση τους.

- **Υπηρεσία Λήψης- Διαχείρισης Περίληψης Τρέχουσας Ημέρας**

  Η υλοποίηση της παρούσας υπηρεσίας πραγματοποιείται, πρώτον με την δημιουργία αιτήματος στον εξυπηρετητή για περίληψη τρέχουσας μέρας και δεύτερον με την λήψη και αξιοποίηση της απάντησης προκειμένου να παρουσιαστεί μέσω της Διεπαφής Τρέχουσας Ημέρας. Ειδικότερα, αναλαμβάνει δράση κάθε φορά που ο χρήστης επιθυμεί να δει

τις τιμές των μετρικών ευημερίας της εκάστοτε μέρας. Σημειώνεται ότι την περίληψη τρέχουσας ημέρας που λαμβάνει δεν την αποθηκεύει κάπου, αφού πρόκειται για δεδομένα που αλλάζουν από λεπτό σε λεπτό.

- **Υπηρεσία Δημιουργίας Ειδοποιήσεων**

  Στην Υπηρεσία Δημιουργίας Ειδοποιήσεων απευθύνεται κάθε άλλη, η οποία επιθυμεί να παρουσιάσει ένα μήνυμα στον χρήστη μέσω ειδοποίησης στη γραμμή εργασιών του κινητού του. Ο ρόλος λοιπόν της υπηρεσίας είναι να εμφανίσει την εκάστοτε ειδοποίηση ακόμα και στη περίπτωση που ο χρήστης είναι μεν συνδεδεμένος, αλλά έχει κλειστή την εφαρμογή.

- **Υπηρεσία Διαχείρισης Στόχων**

  Ανήκει και αυτή στο επίπεδο αξιοποίησης δεδομένων, διότι παίρνει από την Διεπαφή Στόχων αιτήματα, σχετικά με την αποθήκευση (Save Goals) ή την παρουσίαση (Get Latest Goals) των στόχων του χρήστη. Οι στόχοι αυτοί αποθηκεύονται σε σχετική συλλογή (goals) της βάσης της εφαρμογής για τον χρήστη (app_user1 mongoDB). Στη συνέχεια, εμφανίζονται και στις άλλες οθόνες για σύγκριση με τις πραγματικές τιμές. Γι΄ αυτό, μια τελευταία λειτουργία της υπηρεσίας είναι να παρέχει τους αντίστοιχους στόχους στις διεπαφές της εφαρμογής, κάθε φορά που απαιτούνται για την αλληλεπίδραση με τον χρήστη.

- **Υπηρεσία Εγγραφής Χρήστη**

  Η υπηρεσία αυτή δέχεται αιτήματα από την Διεπαφή Εγγραφής προκειμένου να πραγματοποιήσει την εγγραφή ενός νέου χρήστη στο σύστημα. Αρχικά, λαμβάνει την φόρμα που συμπλήρωσε ο χρήστης και ελέγχει τα πεδία(όνοματεπώνυμο, email, κωδικός, πόλη, φύλο, ύψος, ημερομηνία γέννησης) ώστε πρώτον, όλα να είναι συμπληρωμένα και δεύτερον να είναι έγκυρα. Αν δεν είναι, ενημερώνει τον χρήστη με κατάλληλο μήνυμα. Εάν όμως είναι έγκυρα, κάνει χρήση μιας λειτουργίας (createUserWithEmailAndPassword) που παρέχει ο Εξυπηρετητής Αυθεντικοποίησης (Firebase) με σκοπό την εγγραφή του χρήστη. Αν η διαδικασία ολοκληρωθεί ανεπιτυχώς, εμφανίζεται κατάλληλο μήνυμα με το πρόβλημα που προέκυψε (π.χ. ήδη υπάρχουσα διεύθυνση ηλεκτρονικού ταχυδρομίου). Αν ωστόσο η διαδικασία ολοκληρωθεί με επιτυχία, ο χρήστης δημιουργείται και τα δεδομένα του αποθηκεύονται σε σχετική μη σχεσιακή βάση για τους χρήστες της εφαρμογής (συλλογή user_info στην mongoDB βάση app_users).

  Ακόμη, πρέπει να ενημερωθεί και ο εξυπηρετητής και να ανταλλαχθεί ένα κοινό κλειδί για την επικοινωνία. Επομένως η υπηρεσία, κάνοντας χρήση υπηρεσιών ασφαλείας παίρνει ένα ζευγάρι δημόσιου και ιδιωτικού κλειδιου με βάση τον αλγόρθμο RSA και ένα πιστοποιητικό σε μορφή jwt (JSON Web Token). Το πιστοποιητικό αυτό, μαζί με το δημόσιο κλειδί και με τον μοναδικό αριθμό (ταυτοποιητικό) που πήρε κατά την εγγραφή, στέλνονται στην αρμόδια υπηρεσία στον εξυπηρετητή (Υπηρεσία Διαχείρισης Χρηστών). Στη συνέχεια, λαμβάνει το κοινό τους κλειδί κρυπτογραφημένο. Το αποκρυπτογραφεί με βάση το ιδιωτικό κλειδί και έτσι αποκτά εν τέλει το κοινό κλειδί, που θα χρησιμοποιηθεί για την κρυπτογραφημένη επικοινωνία με χρήση του αλγορίθμου AES-128. Φυσικά,

αυτό το κλειδί το αποθηκεύει στην βάση των χρηστών, και τέλος το χρησιμοποιεί, για να στείλει στον εξυπηρετητή κρυπτογραφημένα όλα τα δεδομένα εγγραφής του χρήστη.

- **Υπηρεσία Σύνδεσης Χρήστη**

  Κατά την προσπάθεια σύνδεσης του χρήστη, η υπηρεσία ελέγχει αν η φόρμα είναι συμπληρωμένη και εφόσον είναι, ξεκινάει την διαδικασία για είσοδο του χρήστη στην εφαρμογή. Ειδικότερα, καλεί κατάλληλη υπηρεσία ασφαλείας (Υπηρεσία Αυθεντικοποίησης) προκειμένου να επιβεβαιώσει τα δεδομένα εισόδου τον χρήστη. Αν η σύνδεση αποτύχει, ο χρήστης ενημερώνεται κατάλληλα. Αν όμως συνδεθεί κανονικά, τότε η παρούσα υπηρεσία ενημερώνει τον εξυπηρετητή, ενώ επίσης φορτώνει και τα απαραίτητα δεδομένα του χρήστη (συνδεδεμένες συσκευές, προτιμήσεις, εικόνα προφίλ) με σκοπό να τα δώσει στην Διεπαφή Προφίλ Χρήστη.

- **Υπηρεσία Αυθεντικοποίησης**

  Αναλαμβάνει την επιβεβαίωση των δεδομένων εισόδου του χρήστη. Ουσιαστικά, καλεί κατάλληλη μέθοδο (signInWithEmailAndPassword) του Εξυπηρετητή Αυθεντικοποίησης (Firebase). Μέσω αυτής, παίρνει απάντηση για την εγκυρότητα των δεδομένων και παρέχει το αποτέλεσμα στην αιτούσα υπηρεσία.

- **Υπηρεσία Έκδοσης Πιστοποιητικού Χρήστη**

  Ο ρόλος της είναι να δημιουργήσει ένα πιστοποιητικό, μοναδικό για τον εκάστοτε χρήστη. Καλεί, επομένως σχετική μέθοδο (getToken) του Εξυπηρετητή Αυθεντικοποίησης και το λαμβάνει σε μορφή jwt (JSON Web Token).

- **Υπηρεσία Ανταλλαγής Κλειδιού**

  Χρησιμοποιείται για την ασφαλή ανταλλαγή κλειδιού με τον εξυπηρετητή. Μέσω αυτής της υπηρεσίας παράγεται το ζεύγος ιδιωτικού και δημόσιου κλειδιού, τα οποία θα χρησιμοποιηθούν για να ανταλλαχθεί πετυχημένα το κλειδί κρυπτογραφημένης επικοινωνίας εφαρμογής και εξυπηρετητή.

- **Υπηρεσία Κρυπτογράφησης**

  Οποιαδήποτε κρυπτογράφηση ή αποκρυπτογράφηση δεδομένων βασίζεται στην παρούσα υπηρεσία. Συγκεκριμένα, αναλαμβάνει να κρυπτογραφήσει τα εξερχόμενα προς τον εξυπηρετητή δεδομένα και να αποκρυπτογραφήσει τα εισερχόμενα, με χρήση του αλγορίθμου AES-128 και του κοινού κλειδιού που ανταλλάχθηκε.

Στην παραπάνω περιγραφή των λειτουργιών κάθε υπηρεσίας δεν αναφέρθηκαν ξεχωριστά οι **διεπαφές της εφαρμογής**, καθώς ο ρόλος τους είναι προφανής. Αυτό που κάνουν είναι να αλληλεπιδρούν με τον χρήστη μέσω των οθονών που παρουσιάστηκαν προηγουμένως. Συγκεκριμένα, δέχονται αρχικά αιτήματα για ενέργειες από τον χρήστη και τελικά του παρουσιάζουν ένα αποτέλεσμα, με χρήση των υπηρεσιών που βρίσκονται στο στρώμα αξιοποίησης.

Έχοντας μελετήσει κανείς και το κεφάλαιο της υλοποίησης έχει μια ολοκληρωμένη εικόνα για το πώς υλοποιήθηκε κάθε υπηρεσία ξεχωριστά, προκειμένου να επιτευχθεί ένα σύνολο λειτουργιών στην εφαρμογή. Είναι εμφανές, λοιπόν, ότι με χρήση της λειτουργικότητας του

εξυπηρετητή, είναι δυνατόν να αναπτυχθεί μια εφαρμογή φιλική προς τον χρήστη, η οποία θα μπορεί να συμβάλει στο ευ ζην του έχοντας παρακινητικό και όχι παρεμβατικό ρόλο.

Κεφάλαιο 6

Συμπεράσματα και Μελλοντικές επεκτάσεις

Στο έκτο και τελευταίο κεφάλαιο αυτής της διπλωματικής, θα παρουσιαστούν τα συμπεράσματα στα οποία καταλήξαμε, ενώ επίσης θα γίνει αναφορά και σε πιθανές μελλοντικές επεκτάσεις. Συγκεκριμένα, έχοντας μελετήσει κανείς τα προηγούμενα κεφάλαια έχει αποκτήσει μια συνολική εικόνα του συστήματος που αναπτύχθηκε για την ασφαλή συλλογή και διαχείρηση δεικτών ευημερίας του ατόμου. Είναι σημαντικό, επομένως, σε αυτό το σημείο, να γίνει πρώτον, μια σύνοψη όλων των συμπερασμάτων και δεύτερον, μια παράθεση ιδεών για τις επόμενες επεκτάσεις του συστήματος.

6.1 Συμπεράσματα

Στην παρούσα ενότητα, θα γίνει μια περίληψη των συμπερασμάτων που εξάχθηκαν κατά την εκπόνηση αυτής της διπλωματικής εργασίας.

Αρχικά, ο πρώτος στόχος της εργασίας ήταν να δημιουργήσουμε ένα **κοινό λεξιλόγιο, το οποίο μοντελοποιεί όλους τους δείκτες ευημερίας** που μπορούν να καταγραφούν για ένα άτομο. Ειδικότερα, είδαμε πως υπάρχει **πληθώρα ενδυτών συσκευών και εφαρμογών**, ικανών να μετρήσουν δεδομένα υγείας των χρηστών τους. Υπάρχουν, επίσης, πολλοί **αισθητήρες στο περιβάλλοντα χώρο** του ατόμου, οι οποίοι μπορούν να καταγράψουν πληροφορία που είναι χρήσιμη αν συνδυαστεί σωστά, στα πλαίσια του Διαδικτύου των Πραγμάτων. Παρατηρήσαμε, επομένως, ότι αποτέλεσμα αυτών, είναι η καταγραφή ενός τεράστιου όγκου πληροφορίας, που δυσχεραίνει τις προσπάθειες αξιοποίησης του.

Έγινε, ακόμη, εμφανές πως η **δυσκολία διαχείρισης του όγκου των δεδομένων** έγκειται κυρίως στο γεγονός ότι κάθε πηγή αναπαριστά με διαφορετικό τρόπο τις μετρικές ευημερίας που συλλέγει, με αποτέλεσμα να υπάρχει μια πολλή μεγάλη ποικιλία αναπαραστάσεων ακόμα και του ίδιου τύπου πληροφορίας. Απόρροια αυτού του προβλήματος, είναι το ότι γίνεται επιτακτική η ανάγκη να δημιουργηθεί ένας κοινός κώδικας επικοινωνίας, που θα ακολουθείται από κάθε είδους πηγή δεδομένων υγείας.

Για τον σκοπό αυτό, αναπτύχθηκε στο κεφάλαιο της σχεδίασης, μια **οντολογία** με

χρήση γλώσσας οντολογικού ιστού. Βασίζοντας την έρευνά μας στο τρόπο με τον οποίο κάθε πηγή δεδομένων αναπαριστά τους δείκτες ευημερίας του ατόμου, καταλήξαμε σε ένα τελικό οντολογικό σχήμα. Επιλέχθηκαν, με βάση τις ανάγκες των δεδομένων, οι κλάσεις της οντολογίας, όπως επίσης και οι ιδιότητες τους. Στη συνέχεια, παρουσιάστηκε η δομή αναπαράστασης με **JSON-LD**, με την οποία επιβεβαιώθηκε η δυνατότητα προτυποποίησης των δεικτών ευημερίας του ατόμου. Έγινε, επιπλέον παρουσίαση της γραφικής αναπαράστασης της οντολογίας στην βάση δεδομένων γραφημάτων **Neo4j**, ενώ τέλος αναφερθήκαμε στις ανάγκες που καλύπτει κάθε είδους αναπαράσταση στην διαχείριση των δεδομένων.

Πέρα, ωστόσο, από τον στόχο μας να προτυποποιήσουμε τα δεδομένα με την ανάπτυξη μιας οντολογίας, είχαμε εξαρχής και έναν δεύτερο στόχο. Αυτός ο δεύτερος σκοπός της διπλωματικής ήταν να αναπτυχθεί ένα **ολοκληρωμένο σύστημα ασφαλούς συλλογής και διαχείρισης των δεδομένων υγείας**. Σχεδιάστηκε και υλοποιήθηκε επομένως, ο **εξυπηρετητής** και η **εφαρμογή σε κινητά τηλέφωνα με λειτουργικό Android**. Μέσω αυτού του συστήματος, είδαμε μέσω του εξυπηρετητή πώς μπορούμε να συλλέξουμε δείκτες ευημερίας από διαφορετικές πηγές και να τους επεξεργαστούμε για να παράξουμε **εμπλουτισμένη πληροφορία**. Σε δεύτερη φάση, είδαμε μέσω της εφαρμογής, πώς γίνεται αυτή η εμπλουτισμένη πληροφορία να αξιοποιηθεί προκειμένου να δοθεί μια ολοκληρωμένη εικόνα στον χρήστη. Αποτέλεσμα των παραπάνω, είναι η **συμβολή του συστήματός μας στη βελτίωση της ποιότητας ζωής του ατόμου**, μέσω σωστής καθοδήγησης από τις διεπαφές της εφαρμογής.

Πριν την ανάπτυξη του συστήματος, κατασκευάστηκε μια Γεννήτρια Δεδομένων προκειμένου να προσομοιώσει τις πηγές δεδομένων υγείας. Μετά από μια σύντομη έρευνα των ενδυτών συσκευών, των εφαρμογών και των αισθητήρων που χρησιμοποιούνται για την καταγραφή δεικτών ευημερίας, επιλέξαμε ένα συγκεκριμένο σύνολο από αυτές για να υλοποιήσουμε την προσομοίωση μας. Στη συνέχεια, αναπτύχθηκε αλγόριθμος για να προσομοιωθεί η ημέρα ενός χρήστη ενώ παρουσιάστηκε λεπτομερώς και η διαδικασία δημιουργίας των δεδομένων που θα σταλούν στον εξυπηρετητή στην μορφή που ακολουθούν οι συγκεκριμένες πηγές δεδομένων.

Όσον αφορά τώρα τα κύρια κομμάτια του συστήματος, αναπτύχθηκε αρχικά ο εξυπηρετητής. Παρουσιάστηκε η αρχιτεκτονική με την οποία σχεδιάστηκε, ενώ στη συνέχεια, έγινε εμφανής και η υλοποίηση των υπηρεσιών του. Μελετήθηκε, συγκεκριμένα, ο τρόπος με τον οποίο συλλέγει τα δεδομένα και τα μετασχηματίζει σε σημασιολογικά με βάση την οντολογία. Στη συνέχεια, υλοποιήθηκε η διαδικασία διαχείρισης των σημασιολογικών δεδομένων προκειμένου να παραχθεί εμπλουτισμένη πληροφορία και να σταλεί στην εφαρμογή.

Το δεύτερο συστατικό στοιχείο του συστήματος είναι η εφαρμογή κινητών τηλεφώνων. Σχεδιάστηκε αρχικά το σχήμα της αρχιτεκτονικής της και ύστερα αναπτύχθηκαν οι υπηρεσίες της. Έτσι, υλοποιήθηκαν οι λειτουργίες για συλλογή της εμπλουτισμένης πληροφορίας και για αξιοποίηση της. Τέλος, παρουσιάστηκε ο τρόπος με τον οποίο αλληλεπιδρά με τον χρήστη, με σκοπό να τον καθοδηγήσει με φιλικό τρόπο στην καθημερινότητα του. Συγκεκριμένα, με την εξαγωγή ορθών συμπερασμάτων και κατάλληλων προτάσεων επιτυγχάνεται η βελτιώση στον τρόπο ζωής του.

Βέβαια, καθόλη την διάρκεια ανάπτυξης του συστήματος, δόθηκε ιδιαίτερη σημασία στην **ασφάλεια των προσωπικών δεδομένων** του ατόμου. Γι΄ αυτό το λόγο,

χρησιμοποιήθηκαν κατάλληλες τεχνικές κρυπτογραφίας προκειμένου να διαφυλαχθεί το ευαίσθητο περιεχόμενο της πληροφορίας.

Συνοψίζοντας, τα συμπεράσματα που αποκομίζει κάποιος από αυτή τη διπλωματική εργασία, είναι ότι **η προτυποποίηση των δεικτών ευημερίας είναι εφικτή** και ότι η υλοποίηση της μπορεί να **συμβάλλει καθοριστικά στο ευ ζην του ατόμου**.

6.2 Μελλοντικές επεκτάσεις

Ολοκληρώνοντας, αξίζει να αναφερθούμε και στους τρόπους με τους οποίους μπορεί να επεκταθεί η παρούσα διπλωματική εργασία στο μέλλον. Ειδικότερα, το σύστημα που υλοποιήθηκε, μπορεί να αναπτυχθεί περαιτέρω προς διάφορες κατευθύνσεις.

Πρώτα απόλα, σημειώνεται ότι μια από τις παραδοχές που κάναμε ήταν ότι τα δεδομένα υγείας που φτιάξαμε αφορούν έναν χρήστη. Επομένως, μια πιθανή επέκταση του συστήματος, θα ήταν η ανάπτυξη ενός εξυπηρετητή που έχει την δυνατότητα να διαχειριστεί ταυτόχρονα **πολλούς συνδεδεμένους χρήστες**. Η υλοποίηση αυτή επιτάσσει την χρήση τεχνολογιών υπολογιστικού νέφους (cloud computing), καθώς ο όγκος των δεδομένων που συλλέγονται θα είναι μεγάλος και ραγδαία αυξανόμενος.

Μια δεύτερη επέκταση του συστήματος, θα ήταν η ανάπτυξη της ιδέας για εγγραφή και σύνδεση χρηστών που έχουν **διαφορετικούς ρόλους**. Για παράδειγμα, μια πιθανή σύνδεση γιατρών, φυσιοθεραπευτών, ερευνητών και άλλων ειδικοτήτων θα οδηγούσε σε μια πλήρως ανεπτυγμένη πλατφόρμα. Οι δυνατότητες αυτής θα ήταν εξαιρετικά χρήσιμες, αφού θα ήταν εφικτή μια καθολική συλλογή των δεικτών ευημερίας του ατόμου, οι οποίοι θα μπορούσαν να αξιοποιηθούν στο έπακρο από τους διαφορετικούς ρόλους χρηστών.

Ακόμα μια πιθανή επέκταση θα ήταν η **σύνδεση παραπάνω συσκευών** στο σύστημα, προκειμένου να μελετήσουμε την διαχείριση ακόμα περισσότερων δεδομένων υγείας. Σε αυτό το σενάριο, ιδιαίτερη σημασία πρέπει να δοθεί στους αλγορίθμους που πρόκειται να χρησιμοποιηθούν προκειμένου να φιλτραριστεί η πληροφορία και να παρουσιαστεί ένα τελικό αποτέλεσμα. Επίσης, σημειώνεται ότι απώτερη φιλοδοξία είναι η δυνατότητα συλλογής δεδομένων υγείας από οποιαδήποτε πηγή. Για την επίτευξη αυτού του στόχου πρέπει να γίνει αποδεκτό από τις εταιρίες το κοινό λεξιλόγιο που αναπτύχθηκε στην παρούσα διπλωματική. Επομένως, θα ήταν επιθυμητό να συνταχθεί μια ολοκληρωμένη πρόταση προς κάθε πηγή τέτοιων δεδομένων, με σκοπό την προτυποποίηση των δεικτών ευημερίας που συλλέγουν.

Τέλος, μια εξέχουσας σημασίας προέκταση της εργασίας αφορά το κομμάτι της ασφάλειας όπως παρουσιάζεται και στην δημοσίευση "The Internet of Things in Healthcare Potential Applications and Challenges"[8]. Συγκεκριμένα, σε όλες τις παραπάνω προτάσεις πρέπει να επιστήσουμε την προσοχή μας στην **διασφάλιση της ιδιωτικότητας** του ατόμου. Επομένως, καθίσταται αναγκαίο να γίνει μια ολοκληρωμένη μελέτη για τις μεθόδους ασφαλείας που πρέπει να υλοποιεί κάθε τέτοιο σύστημα. Τα δεδομένα που διαχειριζόμαστε έχουν ευαίσθητο περιεχόμενο, καθότι αφορούν την υγεία του ανθρώπου και για αυτό το λόγο η προστασία τους πρέπει να είναι κύριο μέλημα. Επιπλέον, σε μια πλατφόρμα με χρήστες διαφορετικών ρόλων είναι επιτακτική η ανάγκη να καθοριστούν τα δικαιώματα που έχει πάνω στα δεδομένα υγείας κάθε ρόλος.

Συνοψίζοντας, εύκολα παρατηρεί κάποιος ότι υπάρχουν πολλές πιθανές επεκτάσεις της πλατφόρμας που αναπτύχθηκε σε αυτήν την διπλωματική. Μελλοντικές, επομένως εργασίες μπορούν να επικεντρωθούν στις παραπάνω προτάσεις στα πλαίσια της ανάπτυξης ενός ακόμα πιο ολοκληρωμένου συστήματος ασφαλούς συλλογής και διαχείρισης δεικτών ευημερίας του ατόμου.

Βιβλιογραφία

[1] Despina T. Meridou, Maria-Eleftheria Ch. Papadopoulou, Andreas P. Kapsalis, Panagiotis Kasnesis, Athanasios I. Delikaris, Charalampos Z. Patrikakis, Iakovos S. Venieris, and Dimitra I. Kaklamani, "Improving Quality of Life with the Internet of Everything", in Beyond the Internet of Things, pp. 377-408. Springer International Publishing, 2017.

[2] "Internet of Things (IoT) in healthcare: benefits, use cases and evolutions", [Online]. Available at: https://www.i-scoop.eu/internet-of-things-guide/internet-things-healthcare/ Last accessed: 02/11/2017.

[3] Pedro Castillejo, Jose-Ferman Martinez, Jesus Rodriguez-Molina, Alexandra Cuerva, and Grys-Citsem, "Integration of wearable devices in a wireless sensor network for an E-health application", in IEEE Wireless Communications, Volume 20, Issue 4, August 2013, pg. 1536-1584.

[4] Mostafa Haghi, MSc, Kerstin Thurow, Dr. Ing. Habil, Regina Stoll, and Dr. Med. Habil, "Wearable Devices in Medical Internet of Things: Scientific Research and Commercially Available Devices", in Healthc Inform Res., January 2017, pg. 4-15.

[5] Mary M. Rodgers, Vinay M. Pai, and Richard S. Conroy, "Recent Advances in Wearable Sensors for Health Monitoring", in IEEE Sensors Journal, Volume 15, Issue 6, June 2015, pg. 3119-3126.

[6] Shyamal Patel, Hyung Park, Paolo Bonato, Leighton Chan, and Mary Rodgers, "A review of wearable sensors and systems with application in rehabilitation", in Journal of NeuroEngineering and Rehabilitation, 2012.

[7] David Metcalf, Sharlin T.J. Milliard, Melinda Gomez, and Michael Schwartz, "Wearables and the Internet of Things for Health", in IEEE Pulse, Volume 7, Issue 5, Sept.-Oct. 2016, pg. 35-39.

[8] Phillip A. Laplante, and Nancy Laplante, "The Internet of Things in Healthcare Potential Applications and Challenges", in IT Professional, Volume 18, Issue 3, May-June 2016, pg. 2–4.

[9] Google Developers, "Firebase Authentication", [Online]. Available at: https://firebase.google.com/products/auth/ Last accessed: 02/11/2017.

[10] "Platform Architecture", [Online]. Available at: https://developer.android.com/guide/platform/index.html Last accessed: 02/11/2017.

[11] Linux Foundation, "About Node.js", [Online]. Available at: https://nodejs.org/en/about/ Last accessed: 02/11/2017.

[12] Gosling James, Joy Bill, Steele Guy, Bracha Gilad, Buckley Alex, "The Java Language Specification", March 2015.

[13] Neo4j, Inc., "A Graph Platform Reveals and Persists Connections", [Online]. Available at: https://neo4j.com/ Last accessed: 02/11/2017.

[14] Neo4j, Inc., "About Cypher", [Online]. Available at: https://neo4j.com/developer/cypher-query-language/ Last accessed: 02/11/2017.

[15] MongoDB, Inc., "What is MongoDB?", [Online]. Available at: https://www.mongodb.com/what-is-mongodb Last accessed: 02/11/2017.

[16] MongoDB, Inc., "Quick-Start Guide to mLab", [Online]. Available at: http://docs.mlab.com/ Last accessed: 02/11/2017

[17] Tutorials Point, "Apache Kafka - Cluster Architecture", [Online]. Available at: https://www.tutorialspoint.com/apache_kafka/apache_kafka_cluster_architecture.htm Last accessed: 02/11/2017.

[18] Pivotal Software, Inc., "Understanding AMQP, the protocol used by RabbitMQ", [Online]. Available at: https://spring.io/blog/2010/06/14/understanding-amqp-the-protocol-used-by-rabbitmq/ Last accessed: 02/11/2017.

[19] Ecma International, "Introducing JSON", [Online]. Available at: http://www.json.org/ Last accessed: 02/11/2017.

[20] "JSON for Linking Data", [Online]. Available at: https://json-ld.org/ Last accessed: 02/11/2017.

[21] OWL Working Group, "Web Ontology Language (OWL)", December 2013, [Online]. Available at: https://www.w3.org/OWL/ Last accessed: 02/11/2017.

[22] Fitbit Inc, "Reference API Documentation", [Online]. Available at: https://dev.fitbit.com/reference/ Last accessed: 02/11/2017.

[23] Under Armour, "Designed to Make Athletes Better", [Online]. Available at: https://developer.underarmour.com/docs/ Last accessed: 02/11/2017.

[24] Danielle Kosecki, "REM, Light, Deep: How Much of Each Stage of Sleep Are You Getting?", March 2017, [Online]. Available at: https://blog.fitbit.com/sleep-stages-explained/ Last accessed: 02/11/2017.

[25] William Mccoy, "Normal Speed for Jogging", September 2017, [Online]. Available at: https://www.livestrong.com/article/526358-normal-speed-for-jogging/ Last accessed: 02/11/2017.

[26] Rebeka Stowe, "What is the Average Treadmill Walking Speed?", September 2017, [Online]. Available at: https://www.livestrong.com/article/512777-what-is-the-average-speed-for-walking-on-a-treadmill/ Last accessed: 02/11/2017.

[27] Ryan Haas, "The Average Bike Riding Speed", September 2017, [Online]. Available at: https://www.livestrong.com/article/413599-the-average-bike-riding-speed/ Last accessed: 02/11/2017.

[28] Andrea Cespedes, "Recommended Calorie Intake for One Meal", October 2017, [Online]. Available at: https://www.livestrong.com/article/440135-recommended-calorie-intake-for-one-meal/ Last accessed: 02/11/2017.

[29] Science Buddies, "Stepping Science: Estimating Someone's Height from Their Walk", November 2014, [Online]. Available at: https://www.scientificamerican.com/article/bring-science-home-estimating-height-walk/ Last accessed: 02/11/2017.

[30] Harvard T.H. Chan, "Measuring Physical Activity", [Online]. Available at: https://www.hsph.harvard.edu/nutritionsource/mets-activity-table/ Last accessed: 02/11/2017.

[31] Allan Robinson, "How to Calculate BMR Manually", July 2017, [Online]. Available at: https://www.livestrong.com/article/184215-how-to-calculate-bmr-manually/ Last accessed: 02/11/2017.

[32] U.S. Department of Health and Human Services, "Target Heart Rate and Estimated Maximum Heart Rate", [Online]. Available at: https://www.cdc.gov/physicalactivity/basics/measuring/heartrate.htm Last accessed: 02/11/2017.