



ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ
ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΙΔΙΚΕΥΣΗ: “ΠΡΟΗΓΜΕΝΑ ΠΛΗΡΟΦΟΡΙΑΚΑ
ΣΥΣΤΗΜΑΤΑ”

Ανάπτυξη e-diabetes platform

Ανάπτυξη Πληροφοριακών Συστημάτων

Επιβλέποντες Καθηγητές: Γεώργιος Βασιλακόπουλος, Ανδρέας Μενύχτας

**Ιωάννα Κανδή (ME2136), Κωνσταντίνος Μαυρογιώργος (ME2144),
Δήμητρα Σεισάκη (ME2147), Παύλος-Νικόλαος Τουμλελής (ME2152),
Δήμητρα Φλωροπούλου (ME2154)**

Φεβρουάριος 2022

Πίνακας Περιεχομένων

Κατάλογος Εικόνων	3
Περίληψη	4
1. Εισαγωγή	5
1.1 Ορισμός του Προβλήματος	5
2.Κριτική Ανασκόπηση Βιβλιογραφίας	7
2.1. Υπηρεσιοστρεφής Αρχιτεκτονική	7
2.2. Μεθοδολογία	9
2.3. Μεθοδολογία Μαλακών Συστημάτων (ΜΜΣ) - Soft Systems Methodology (SSM)	10
2.3.1. Εισαγωγή στα ΜΜΣ	10
2.3.2. Μεθοδολογία των επτά σταδίων	11
2.4. Κύκλος Ζωής	15
2.4.1. Μεθοδολογία Καταρρακτοειδούς Ανάπτυξης Συστημάτων	19
2.4.2. Επαυξητική Μεθοδολογία	21
2.4.3. Μεθοδολογία Λειτουργικής Επαύξησης	21
2.4.4. Μεθοδολογία V	22
2.4.5. Σπειροειδής Μεθοδολογία	23
2.4.6. Μεθοδολογία εξελικτικών πρωτοτύπων	23
2.5. Μεθοδολογίες Agile	25
2.5.1. Οφέλη και περιορισμοί	27
2.6. Μεθοδολογία Scrum	29
3. Αξιοποιηθέντα εργαλεία DevOps και τεχνολογίες	33
3.1. Gitlab	33
3.2. Slack	35
3.3. Χρησιμοποιούμενες Τεχνολογίες - Γλώσσες Προγραμματισμού	36
3.4. Δημιουργία φακέλου infosysdev - Χρήση περιβάλλοντος git για ανάρτηση κώδικα στο Gitlab	37
3.5. Δοκιμές - Έλεγχος κώδικα	39
4. Στατικές και συμπεριφορικές όψεις του συστήματος με χρήση UML	40
4.1. Η UML συνοπτικά	40
4.2. Component Diagram	41
4.3. Use Case Diagram	43
4.4. Activity Diagram	45
	1

4.4.1. Ενδεικτικό διάγραμμα δραστηριοτήτων (activity diagram) σχετικά με την συμπλήρωση δεδομένων υγείας ασθενή ηλεκτρονικά	45
4.4.2. Ενδεικτικό διάγραμμα δραστηριοτήτων (activity diagram) σχετικά με την προβολή δεδομένων υγείας ασθενή από τον γιατρό ηλεκτρονικά	46
5. Εγχειρίδιο Χρήσης (User Manual)	48
5.1. Σύνδεση και Εγγραφή Χρηστών	48
5.1.1. Εγγραφή χρήστη στη πλατφόρμα	48
5.1.2. Σύνδεση χρήστη στη πλατφόρμα	49
5.2. Περιγραφή end-points γιατρού	50
5.2.1. Εισαγωγή μετρήσεων του ασθενή από τον γιατρό (patient's data import on doctor's end)	50
5.2.2. Εισαγωγή θεραπείας / συνταγής του ασθενούς από τον γιατρό (patient's prescription import from doctor's end)	51
5.2.3. Διαχείριση λογαριασμού γιατρού (doctor account management)	52
5.2.4. Προβολή ασθενών και των τελευταίων μετρήσεων τους από την πλευρά του γιατρού σε μορφή πίνακα (doctor's patients' data view in list form)	53
5.2.5. Ειδοποίηση γιατρού μέσω ηλεκτρονικού ταχυδρομείου για μη φυσιολογική μέτρηση ασθενή (doctor's email notification for patient's abnormal data import)	54
5.3. Περιγραφή end-points ασθενούς	55
5.3.1. Εισαγωγή μετρήσεων ασθενή (patient's data import)	55
5.3.2. Διαχείριση λογαριασμού ασθενή (Patient account management)	56
5.3.3. Προβολή συνταγής από τον ασθενή (Prescription view from patient's end)	57
5.3.4. Προβολή συχνών ερωτήσεων / απαντήσεων (F.A.Q.)	58
5.3.5. Προβολή Προσωπικού Φακέλου Υγείας του ασθενή (Patient's Personal Health Record view)	59
5.4. Responsiveness Εφαρμογής e-diabetes	60
Πηγές	61

Κατάλογος Εικόνων

Εικόνα 1: infosysdev Project Issues	34
Εικόνα 2: infosysdev Project Milestones	34
Εικόνα 3: infosysdev Gitlab & Slack Integration	35
Εικόνα 4: e-diabetes component diagram	41
Εικόνα 5: e-diabetes use case diagram	43
Εικόνα 6: e-diabetes activity diagram (patient)	45
Εικόνα 7: e-diabetes activity diagram about (doctor)	46
Εικόνα 8: e-diabetes registration form	48
Εικόνα 9: e-diabetes login form	49
Εικόνα 10: patient's data import on doctor's end	50
Εικόνα 11: patient's prescription import from doctor's end	51
Εικόνα 12: doctor account management	52
Εικόνα 13: doctor's patients' data view	53
Εικόνα 14: doctor's e-mail notification for patient's abnormal data import	54
Εικόνα 15: patient's data import	55
Εικόνα 16: patient account management	56
Εικόνα 17: Prescription view from patient's end	57
Εικόνα 18: F.A.Q. page	58
Εικόνα 19: Patient's Personal Health Record	59
Εικόνες 20-22: e-diabetes responsiveness	60

Περίληψη

Στην παρούσα εργασία παρατίθενται σημαντικές μεθοδολογίες ανάπτυξης πληροφοριακών συστημάτων και γίνεται μία αναφορά σε βασικές έννοιες γύρω από τη χρήση τους. Κατά το εισαγωγικό κομμάτι, περιγράφονται η Μεθοδολογία Μαλακών Συστημάτων, οι μέθοδοι/μεθοδολογίες που χρησιμοποιούνται στα πλαίσια ανάπτυξης ενός πληροφοριακού συστήματος και οι κυριότερες ευέλικτες μεθοδολογίες (Agile Methodologies). Μέσα από μία συγκεντρωτική περιγραφή, καταλήγουμε στην ευέλικτη μεθοδολογία της Scrum η οποία ύστερα από την πραγματοποίηση μιας συγκριτικής αξιολόγησης, κρίνεται ως η καταλληλότερη όλων των μεθοδολογιών και επιλέγεται για την ανάλυση του πληροφοριακού μας συστήματος στα πλαίσια του μαθήματος.

Η εργασία μας επικεντρώνεται στην ανάπτυξη μία ηλεκτρονικής πλατφόρμας για άτομα όλων των ηλικιών με τη χρόνια πάθηση του σακχαρώδη διαβήτη, στην οποία θα μπορούν να έχουν πρόσβαση από κοινού ο γιατρός και ο ασθενής στις λειτουργίες που προσφέρει η ηλεκτρονική πλατφόρμα μας (e-diabetes), ανάλογα με τον ρόλο τους.

1. Εισαγωγή

Στις μέρες μας, η ταχύτητα των εξελίξεων σε τεχνολογικό επίπεδο σε συνδυασμό με τη συνεχή απαίτηση για παραγωγή και προώθηση ποιοτικών προϊόντων και υπηρεσιών στους πελάτες, φέρνουν στην επιφάνεια την ανάγκη για δημιουργία συνεχώς ανανεωμένων και εξελιγμένων πληροφοριακών συστημάτων, που πρακτικά κάνουν τη ζωή μας πιο εύκολη μέσα από τις αυτοματοποιημένες διαδικασίες που προσφέρονται για συλλογή, εγγραφή, ανάκτηση, επεξεργασία, αποθήκευση και ανάλυση πληροφοριών.

Η ανάγκη για γρήγορη προσαρμοστικότητα, παραγωγικότερες πρακτικές, ταχύτερη ανταπόκριση στις απαιτήσεις, λιγότερη γραφειοκρατία, έφεραν στο προσκήνιο την ανάπτυξη ευέλικτων μεθόδων (Agile Methodology) που θα χρησιμοποιήσουμε και στην εργασία για την ανάπτυξη του πληροφοριακού μας συστήματος, με βασική προτεραιότητα τη μέγιστη ικανοποίηση του πελάτη [1].

1.1 Ορισμός του Προβλήματος

Μία πάθηση που συναντάμε σήμερα σε ανθρώπους όλων των ηλικιών, είναι ο σακχαρώδης διαβήτης. Πρόκειται για ένα μεταβολικό νόσημα όπου όταν ένας ασθενής διαγνωσθεί με κάποιον από τους τύπους αυτής της πάθησης (διαβήτης τύπου 1 – ή παιδικός διαβήτης, διαβήτης τύπου 2, διαβήτης κύησης), είναι απαραίτητη η παρακολούθησή του από έναν γιατρό ενδοκρινολόγο, διαβητολόγο ή παθολόγο με ειδικές γνώσεις διαβητολογίας [2].

Μια τέτοια πάθηση χρειάζεται διαρκή ιατρική παρακολούθηση. Οι επισκέψεις σε διαβητολόγο με προγραμματισμένα ραντεβού είναι τακτικές ώστε να ελέγχονται οι τιμές του ασθενή. Ο προσωπικός διαβητολόγος του κάθε ασθενή ελέγχει αν ο διαβήτης παραμένει σταθερός και αντίστοιχα χορηγεί νέα αγωγή σε περίπτωση που χρειάζεται. Ο εξειδικευμένος ιατρός δημιουργεί έναν φάκελο για τον κάθε έναν από τους ασθενείς του και προγραμματίζει το επόμενο ραντεβού ανάλογα με την πορεία της υγείας του.

Σκοπός αυτής της εργασίας είναι η ανάπτυξη ενός πληροφοριακού συστήματος, με το οποίο θα βελτιωθεί η εξυπηρέτηση του ασθενούς, καθώς θα πραγματοποιείται ηλεκτρονικά η παρακολούθηση του ασθενούς από το γιατρό κάνοντας χρήση μιας ηλεκτρονικής πλατφόρμας. Με αυτό τον τρόπο ο ασθενής εξοικονομεί χρόνο από τις συχνές επισκέψεις που αναγκάζεται να προγραμματίσει. Ακόμα, είναι περισσότερο τυπικός στα χρονικά περιθώρια μεταξύ των ραντεβού του με τον γιατρό, αφού δεν αποτελεί πλέον δύσκολη διαδικασία το κομμάτι της ενημέρωσης των νέων τιμών του σακχάρου του.

Το web-based πληροφοριακό σύστημα υγείας της παρούσας εργασίας χαρακτηρίζεται από τη δυνατότητα που δίνεται στον ασθενή να εισέλθει στην ηλεκτρονική πλατφόρμα με τα στοιχεία σύνδεσης του (username,password) και να μπορέσει να εισάγει τις καθημερινές του μετρήσεις. Επίσης, οι χρήστες του έχουν την ευχέρεια να προβάλλουν προηγούμενες μετρήσεις τους (στον προσωπικό τους ιατρικό φάκελο υγείας), καθώς και να ανατρέχουν στη συνταγή που τους χορήγησε ο γιατρός τους. Επίσης, δύνανται να αναζητήσουν πληροφορίες αναφορικά με την πάθηση του διαβήτη. Επιπλέον, οι χρήστες-γιατροί της πλατφόρμας έχουν δικαίωμα προβολής των στοιχείων και των εξετάσεων των ασθενών τους. Συγκεκριμένα, ο γιατρός μπορεί να ενημερώνεται ανά πάσα ώρα και στιγμή για τα δεδομένα υγείας που έχει εισάγει οι ασθενείς του, ενώ ταυτόχρονα, έχει πρόσβαση και στο ιστορικό των μετρήσεών τους. Περαιτέρω, σύμφωνα με τις εισαχθείσες μετρήσεις, ο γιατρός μπορεί να κάνει χρήση ενός εκπαιδευμένου αλγορίθμου μηχανικής μάθησης, έτσι ώστε να αποφασίσει αν ο ασθενής χρειάζεται ινσουλίνη ή όχι. Τέλος, κοινή δυνατότητα ασθενούς και γιατρού είναι η διαχείριση του λογαριασμού τους (όπως για παράδειγμα αλλαγή username).

2.Κριτική Ανασκόπηση Βιβλιογραφίας

2.1. Υπηρεσιοστρεφής Αρχιτεκτονική

Όταν αναφερόμαστε στην υπηρεσιοστρεφή αρχιτεκτονική (Service-oriented architecture - SOA), εννοούμε τη διαδικασία όπου σχεδιάζεται ένα σύστημα λογισμικού έτσι ώστε να παρέχει υπηρεσίες τόσο στους τελικούς χρήστες μιας εφαρμογής, όσο και σε άλλες υπηρεσίες μιας επιχείρησης που είναι διαμοιραζόμενες σε ένα δίκτυο. Η αρχιτεκτονική αυτή επιτρέπει στις υπηρεσίες να είναι διαθέσιμες στον παγκόσμιο ιστό. Ο βασικός στόχος της αρχιτεκτονικής αυτής είναι να επιτρέπει την επικοινωνία μεταξύ διαφορετικών συστημάτων και γλωσσών προγραμματισμού, χωρίς να υπάρχουν εμπόδια για την επικοινωνία μεταξύ των εφαρμογών που εκτελούνται σε διαφορετικές και ανεξάρτητες πλατφόρμες μέσω κατάλληλων πρωτοκόλλων επικοινωνίας [4].

Μια SOA είναι εύκολο να χρησιμοποιηθεί από πληροφοριακά συστήματα και να τα αναβαθμίσει. Καθώς επιτρέπει την εισαγωγή νέων δεδομένων στην δημιουργία εταιρικών μοντέλων και μειώνει το κόστος των επιχειρηματικών διαδικασιών.

Η αρχιτεκτονική της SOA αποτελείται από τέσσερις (4) λειτουργικές συνιστώσες οι οποίες είναι:

- 1) Μεταφορά (Transport layer): το επίπεδο αυτό είναι υπεύθυνο για τη μεταφορά των μηνυμάτων μεταξύ των υπηρεσιών. Το πεδίο αυτό έχει ως στόχο τη μετακίνηση αιτήσεων για υπηρεσίες (service requests) από τον καταναλωτή υπηρεσιών (service consumer) στον πάροχο υπηρεσιών (service provider), και αφορά τις αποκρίσεις των υπηρεσιών (service responses) από τον πάροχο στον καταναλωτή [3].
- 2) Πρωτόκολλο επικοινωνίας υπηρεσιών: το πεδίο αυτό είναι υπεύθυνο για την σύνταξη των μηνυμάτων χρησιμοποιώντας ένα κοινό πρότυπο της XML [4].
- 3) Περιγραφή υπηρεσίας: το πεδίο αυτό αναφέρεται στην περιγραφή μιας λειτουργίας και στα απαιτούμενα δεδομένα για την επιτυχή κλήση της [4].
- 4) Επιχειρησιακή διαδικασία: το σύνολο των υπηρεσιών, οι οποίες καλούνται με συγκεκριμένη σειρά έτσι ώστε να μπορούν να ανταποκριθούν σε μια επιχειρησιακή απαίτηση [4].

Μια υπηρεσία έχει ως στόχο την ικανοποίηση των απαιτήσεων της, οι οποίες πραγματοποιούνται μέσω μιας SOA. Δημιουργεί μια υπηρεσία διαθέσιμη στον ιστό, η οποία μπορεί να χρησιμοποιηθεί παραπάνω από μία φορές από τους χρήστες. Τα στάδια που ακολουθεί είναι αρχικά η σύνδεση (Bind) όπου ο πελάτης μπορεί να καλέσει την υπηρεσία και να την χρησιμοποιήσει με τον καταλληλότερο τρόπο. Δεύτερον η δημοσίευση (Publish), η υπηρεσία πρέπει να δημοσιεύεται, ώστε αυτή να μπορεί να είναι προσβάσιμη. Τέλος, η Εύρεση (Find) όπου ο αιτών (καταναλωτής) μιας υπηρεσίας την εντοπίζει μέσα από την αναζήτηση στο μητρώο υπηρεσιών αφού τα πρότυπα θα έχουν καθοριστεί με τέτοιο τρόπο ώστε να εξυπηρετούν την παραπάνω διαδικασία [4].

2.2. Μεθοδολογία

Όλα τα πληροφοριακά συστήματα παρουσιάζουν ένα κοινό χαρακτηριστικό που δεν είναι άλλο από τον τρόπο ανάπτυξής τους, χρησιμοποιώντας μία μεθοδολογία. Η μέθοδος σε σχέση με τον όρο της μεθοδολογίας παρουσιάζουν ορισμένες διαφορές αν λάβουμε υπόψιν μας πως η μεθοδολογία φαίνεται να είναι πιο ευέλικτη από τη μέθοδο, αναφορικά με τη δομή και τον τρόπο εφαρμογής της. Η μεθοδολογία ακολουθεί μία προσέγγιση, κατά την οποία εισάγει τον κίνδυνο της προβληματικής κατάστασης προς τη μεθοδολογία, ώστε το οποιοδήποτε πρόβλημα να καταστεί κατάλληλο για την εφαρμογή της. Ωστόσο μία πιο επιτυχής προσέγγιση, φαίνεται να είναι η προβληματοστρεφής. Σε αυτή την περίπτωση, η προβληματική κατάσταση διαμορφώνει τον τρόπο ανάπτυξης του πληροφοριακού συστήματος, δηλαδή την καταλληλότερη μορφή μεθοδολογίας που πρέπει να επιλεχθεί [19].

Ιδίως τα τελευταία χρόνια παρατηρείται ένα διαρκώς αυξανόμενο ενδιαφέρον για χρήση της μεθοδολογίας Μαλακών Συστημάτων για εργασίες που σχετίζονται με πληροφοριακά συστήματα. Η μεθοδολογία αυτή εκλαμβάνει όλα τα πληροφοριακά συστήματα που βασίζονται σε υπολογιστές, ως συστήματα που εξυπηρετούν την ανθρώπινη δράση και στην ουσία χωρίζεται σε δύο διακριτά συστήματα. Το πρώτο αφορά το σύστημα εξυπηρέτησης (πληροφοριακό σύστημα) και το δεύτερο σε ένα σύστημα που προσφέρεται για κάποιο σκοπό. Τα πληροφοριακά συστήματα (IS) εκφράζονται με την μορφή ενός μοντέλου “μοντέλο ανάπτυξης συστήματος πληροφοριών/ πληροφοριακού συστήματος” (IS/ISDM) [9].

2.3. Μεθοδολογία Μαλακών Συστημάτων (ΜΜΣ) - Soft Systems Methodology (SSM)

Στο κεφάλαιο αυτό, θα παρουσιάσουμε την ανάλυση της μεθοδολογίας μαλακών συστημάτων την οποία ακολουθήσαμε, προκειμένου να αναπτύξουμε το πληροφοριακό σύστημα.

Η Μέθοδος Μαλακών Συστημάτων χρησιμοποιείται σε ένα πλήθος προβληματικών καταστάσεων παρέχοντας ικανοποιητικά αποτελέσματα προς την κατεύθυνση βελτίωσης τους. Στόχος μας είναι να δείξουμε πώς η Μέθοδος Μαλακών Συστημάτων μπορεί να βοηθήσει στο σχεδιασμό ενός προγράμματος εξυπηρέτησης ασθενών με την βοήθεια μιας πλατφόρμας στον ιστότοπο.

2.3.1. Εισαγωγή στα ΜΜΣ

Η μεθοδολογία Μαλακών συστημάτων (SSM) εμφανίστηκε το 1980 στο πανεπιστήμιο του Lancaster στη μεγάλη Βρετανία. Ο καθηγητής που την ανέπτυξε ήταν ο Peter Checkland και οι συνεργάτες του, όπως ο Dr Brian Wilson [5]. Με τον όρο μαλακά συστήματα αναφερόμαστε σε μια μέθοδο δόμησης προβλημάτων μέσω της συστημικής μηχανικής. Έχει αποδειχθεί αποτελεσματική στην αντιμετώπιση ή την βελτίωση προβληματικών καταστάσεων σκοπεύοντας στην όσο δυνατόν καλύτερη αντιμετώπισή τους [5].

Η SSM ορίζει την αναγνώριση του προβλήματος. Εφαρμόζει μια σφαιρική τακτική ανάμεσα στο γενικό και στο συγκεκριμένο. Δεν στοχεύει στην δημιουργία ενός μοντέλου που περιέχει τα συστατικά στοιχεία του πραγματικού κόσμου. Αυτό συμβαίνει καθώς η πολυπλοκότητα του πραγματικού κόσμου είναι βέβαιο ότι ούτε βοηθάει στην αντιμετώπιση του προβλήματος, αφού η πλήρης απεικόνιση του είναι αδύνατη ούτε όμως στην απλοποίηση γιατί με τον τρόπο αυτό παραλείπονται σημαντικά στοιχεία που επηρεάζουν το τελικό αποτέλεσμα. Επομένως, η μεθοδολογία εφαρμόζει τακτικές οι οποίες δεν απεικονίζουν πλήρως τον πραγματικό κόσμο ούτε όμως είναι πλήρως θεωρητικές. Η SSM εστιάζει σε προβλήματα τα οποία δεν είναι εύκολα αναγνωρίσιμα και παρουσιάζουν δυσκολία στον εντοπισμό τους και υποστηρίζει ότι μόνο με τον τρόπο αυτό θα μπορέσουν να βρεθούν οι κατάλληλες

αλλαγές που πρέπει να γίνουν για την επίλυση ή την βελτίωση μιας προβληματικής κατάστασης [7].

2.3.2. Μεθοδολογία των επτά σταδίων

Η αρχική σύνθεση της SSM έχει επτά στάδια. Η ανάπτυξη του SSM ξεκίνησε τη δεκαετία του 1970 και μια πρώιμη έκδοση της μεθοδολογίας περιλαμβάνει τα ακόλουθα επτά στάδια:

(1) Η κατάσταση του προβλήματος θεωρείται προβληματική:

Στη πρώτη φάση, η SSM εστιάζει στη διερεύνηση μιας προβληματικής κατάστασης. Είναι πολύ δύσκολο να προσδιοριστούν τα όρια μιας κατάστασης και ειδικά όταν σε αυτή την κατάσταση εμπλέκονται πολλοί πόλοι. Η κάθε οπτική γωνία θεώρησης μιας κατάστασης έχει την δική της βαρύτητα και δυναμική. Οι διαφορετικές προσεγγίσεις οδηγούν σε διαφορετικούς τρόπους αντιμετώπισης, οι οποίοι είναι πλήρως υποκειμενικοί. Επομένως η SSM παρεμβαίνει για να βοηθήσει στην αντιμετώπιση της υποκειμενικότητας. Αυτό το επιτυγχάνει με τη συλλογή απόψεων από τους εμπλεκόμενους φορείς στην προβληματική κατάσταση. Σε αυτό το στάδιο, ο ερευνητής δημιουργεί μια εικόνα του προβλήματος με εικασίες που συγκεντρώνει από κοινωνικά στοιχεία (ερωτηματολόγια, συνεντεύξεις) [5].

(2) Εκφράζεται η κατάσταση του προβλήματος:

Στη δεύτερη φάση, οι διαφορετικές απόψεις που συγκεντρώθηκαν προηγουμένως καταγράφονται σε μία εικόνα. Η εικόνα αυτή ονομάζεται πλούσια εικόνα. Πρόκειται για ένα χειρόγραφο σχέδιο με ελάχιστο κείμενο, στο οποίο γίνεται προσπάθεια να αποδοθούν σχηματικά και με συμβολικά στοιχεία όπως: δομές, διεργασίες, κλίμα, άνθρωποι και συγκρούσεις. Στόχος της είναι η όσο το δυνατόν καλύτερη καταγραφή των στοιχείων ώστε να μπορέσουν να γίνουν κατανοητά από τον οποιοδήποτε και να βοηθήσει τους εμπλεκόμενους να αντιληφθούν την κατάσταση. Η πλούσια εικόνα είναι η δημιουργία ενός πλαισίου επικοινωνίας των εμπλεκόμενων. Προβάλλοντας όλες τις οπτικές γωνίες θα βοηθήσει στην κατανόηση του προβλήματος. Είναι σημαντική η δημιουργία μιας πλούσιας εικόνας γιατί συλλέγονται στοιχεία και δημιουργείται μια

σφαιρική άποψη για την προβληματική κατάσταση. Η δομή της αποτελείται από βέλη και σχήματα χωρίς να είναι αυστηρά καθορισμένος ο τρόπος αναπαράστασης. Η SSM αφήνει στην διάθεση του αναλυτή να δημιουργήσει την δική του εικόνα, καθώς δεν υπάρχει σαφής μέθοδος κατασκευής της πλούσιας εικόνας [6].

(3) Ορισμοί ρίζας του σχετικού συστήματος σκόπιμης δραστηριότητας:

Η τρίτη φάση, περιγράφει τη δύναμη παρέμβασης που έχει ο καθένας. Σε αυτή τη φάση ακολουθείται στρατηγική μέσω του μνημονικού CATWOE που επιτρέπει την κατανόηση πτυχών του μετασχηματισμού.

Ως ορισμό ρίζας του συστήματος μας, μπορούμε να αναφέρουμε μία ηλεκτρονική πλατφόρμα πρόσβασης ασθενών που πάσχουν από διαβήτη και η πλατφόρμα αυτή ανήκει ιδιωτικά στον εξειδικευμένο διαβητολόγο ο οποίος την χρησιμοποιεί για να εξυπηρετεί τους πελάτες του. Επιπλέον, η πλατφόρμα περιλαμβάνει ένα σύνολο εγκαταστάσεων και εξοπλισμού, όπως επίσης και ένα πλήθος ανθρωπίνου δυναμικού με την κατάλληλη βασική εμπειρία σε ηλεκτρονικές πλατφόρμες ώστε να εκτελούν εύκολα όσες λειτουργίες είναι απαραίτητες. Υπεύθυνος για την ηλεκτρονική πλατφόρμα είναι ο IT technician ο οποίος εκτελεί οποιαδήποτε αλλαγή ή εντολή ζητήσει ο διαχειριστής, δηλαδή ο διαβητολόγος - γιατρός, ο οποίος έχει αναλάβει να επιβλέπει τις εγγραφές και τα αρχεία των χρηστών.

Στο CATWOE, ένας πελάτης (C) είναι αυτός που θα ωφεληθεί ή θα χάσει όταν εκτελεστεί ο μετασχηματισμός (T). Οι δρώντες στο σύστημα (A) είναι αυτοί που θα κάνουν τη μεταμόρφωση και ο ιδιοκτήτης (O) αναθέτει και εκτελεί τη δουλειά που πρέπει να γίνει. Το περιβάλλον (E) είναι ο περιορισμός ο οποίος σχετίζεται με τους μετασχηματισμούς. Τέλος, το Weltanschauung (W) είναι μια γερμανική έκφραση που αναφέρεται στη κοσμοαντίληψη. Αντιστοιχεί στην οπτική γωνία θεώρησης του συστήματος [8].

Οι όροι του CATWOE δημιουργούν μία έννοια ενός συστήματος που κάνει (T), για (C), υλοποιείται από (A), λόγω (W), υπό την εντολή του (O) και περιορίζεται από (E).

Σύμφωνα με τον παραπάνω ορισμό, στο σύστημα που περιγράψαμε περιλαμβάνονται:

1. Οι πελάτες (customers - C), δηλαδή τα άτομα που επηρεάζονται (θετικά ή αρνητικά) από τις δραστηριότητες του συστήματος. Στην προκειμένη περίπτωση οι πελάτες είναι οι ασθενείς που πάσχουν από διαβήτη, οι οποίοι έχουν εγγραφεί στην ηλεκτρονική πλατφόρμα και είναι πελάτες του συγκεκριμένου διαβητολόγου.
2. Οι λειτουργοί (actors - A), δηλαδή τα άτομα που εκτελούν ή προκαλούν την εκτέλεση των κύριων δραστηριοτήτων του συστήματος. Στην προκειμένη περίπτωση είναι ο εξειδικευμένος γιατρός ο οποίος έχει αναλάβει την αγωγή του κάθε εγγεγραμμένου ασθενή καθώς γνωρίζει καλά την κάθε περίπτωση και επιβλέπει τις τιμές του κάθε πελάτη ώστε να του χορηγεί τη σωστή αγωγή.
3. Ο μετασχηματισμός (transformation - T), δηλαδή η επεξεργασία κατά την οποία τα εισερχόμενα μετατρέπονται σε εξερχόμενα. Ο μετασχηματισμός αποτελεί τον πυρήνα του συστήματος. Στην προκειμένη περίπτωση αφορά την διαδικασία, στην οποία εμπλέκονται οι ασθενείς που εγγράφονται στο συγκεκριμένη ηλεκτρονική πλατφόρμα προκειμένου να αποκτήσουν πρόσβαση και να γίνουν μέλη της. Ωστε να δημοσιεύουν τις μετρήσεις τους και έπειτα να παραλαμβάνουν την ηλεκτρονική αγωγή που θα τους επισυνάψει ο γιατρός.
4. Η κοσμοθεώρηση (world view - weltanschauungen - W), δηλαδή μία άποψη, ένα πλαίσιο ή μία παραστατική αντίληψη του κόσμου. Οι υπηρεσίες που παρέχονται στην πλατφόρμα είναι στο πλαίσιο της παροχής υπηρεσιών προς τους ασθενείς για την επίβλεψη της υγείας τους όσον αφορά τις τιμές του διαβήτη, ώστε να ελέγχουν τακτικά τις τιμές τους και να τους χορηγείται η σωστή αγωγή από τον γιατρό σε ψηφιακή μορφή.
5. Η ιδιοκτησία (ownership - O), δηλαδή τα άτομα που έχουν την κυριότητα του συστήματος, γνωρίζουν κατάλληλες λειτουργίες για να χειρίζονται την ιστοσελίδα του προγράμματος και έχουν, κατά συνέπεια, τη δύναμη να προκαλέσουν τον τερματισμό της ύπαρξης και της λειτουργίας του. Άτομα σαν αυτά είναι κυρίως οι IT technician που έχουν αναλάβει να υλοποιήσουν την πλατφόρμα που τους έχει ζητηθεί από τον γιατρό. Επομένως, λαμβάνουν εντολές από εκείνον και τις εκτελούν. Οι εντολές αυτές μπορούν να έχουν ως αποτέλεσμα την τροποποίηση ή ακόμα και την κατάργηση μιας λειτουργίας ή ολόκληρης της πλατφόρμας.

6. Το περιβάλλον (environment - E), δηλαδή τους περιορισμούς του συστήματος που επιβάλλονται από τον περιβάλλοντα χώρο στον οποίο υπάρχει και λειτουργεί. Το σύστημα που περιγράψαμε λειτουργεί στο πλαίσιο που ορίζεται από το πελατολόγιο του διαβητολόγου. Οι ηλεκτρονικές υπηρεσίες είναι διαθέσιμες σε συγκεκριμένο πελατολόγιο, το οποίο ο γιατρός γνωρίζει και εξυπηρετεί. Πρόκειται για ασθενείς που τους επιβλέπει ο ιατρός και γνωρίζει την κατάσταση και το στάδιο διαβήτη στο οποίο βρίσκονται [8].

(4) Εννοιολογικά μοντέλα των συστημάτων που αναφέρονται στους ριζικούς ορισμούς:

Η τέταρτη φάση, αφορά την ανάπτυξη των εννοιολογικών μοντέλων σύμφωνα με τους θεμελιακούς ορισμούς που διατυπώθηκαν στην προηγούμενη φάση. Το εννοιολογικό μοντέλο ακολουθεί την πρακτική ενός συστήματος το οποίο αποτελείται από μία οντότητα η οποία δέχεται εισροές και παράγει εκροές. Αρχικά, κατασκευάζεται ένα μοντέλο το οποίο αναπτύσσεται ιεραρχικά κατά την κρίση του αναλυτή, ο οποίος περιγράφει στον απαιτούμενο βαθμό λεπτομέρειας τις δραστηριότητες του συστήματος.

Η τέταρτη φάση αποτελείται από δύο υποφάσεις τις 4α και 4β.

Η 4α αναφέρεται στη χρήση ενός γενικού μοντέλου συστημάτων ανθρώπινης δραστηριότητας, το οποίο ελέγχει ότι τα εννοιολογικά μοντέλα που αναπτύχθηκαν δεν έχουν βασικές ελλείψεις.

Η 4β αναφέρεται σε τροποποιήσεις του μοντέλου της προηγούμενης φάσης ώστε να βοηθήσει στην βελτίωση της αναπαράστασης του προβλήματος [7].

(5) Σύγκριση μοντέλων και πραγματικού κόσμου:

Στη πέμπτη φάση, τα εννοιολογικά μοντέλα που κατασκευάστηκαν στο προηγούμενο στάδιο μεταφέρονται στον πραγματικό κόσμο και συγκρίνονται με την προβληματική κατάσταση. Έπειτα, οι εκπρόσωποι και οι αναλυτές του συστήματος αναπτύσσουν διάλογο μεταξύ τους προκειμένου να δώσουν λύσεις [7].

(6) Αλλαγές συστηματικά επιθυμητές και πολιτιστικά εφικτές:

Στην έκτη φάση, προσδιορίζονται οι εφικτές αλλαγές. Αφού προηγηθεί το πέμπτο βήμα, θα έχει ως αποτέλεσμα οι συγκρίσεις που θα γίνουν να αποφέρουν ένα σύνολο εφικτών συστάσεων, οι οποίες θα πρέπει να είναι συστημικά επιθυμητές και πολιτισμικά εφικτές [7].

(7) Δράση για τη βελτίωση της προβληματικής κατάστασης:

Μετά το έκτο στάδιο όπου έχουν γίνει οι επιθυμητές και εφικτές αλλαγές, ξεκινούν οι απαραίτητες ενέργειες για την υλοποίησή τους. Οι αλλαγές αυτές μπορεί να έχουν ως αποτέλεσμα τη δημιουργία νέων προβλημάτων που μπορούν να αντιμετωπιστούν εκτελώντας ξανά τη μεθοδολογία μαλακών συστημάτων [7].

2.4. Κύκλος Ζωής

Συχνά, οι διάφορες μεθοδολογίες/μέθοδοι που χρησιμοποιούνται στα πλαίσια ανάπτυξης ενός πληροφοριακού συστήματος, ονομάζονται και “κύκλοι ζωής ανάπτυξης συστημάτων” - ΚΖΑΣ (systems development life cycles- SDLC). Ο κύκλος ζωής αναφέρεται στην σταδιακή, κατά φάσεις, εκτέλεση της διεργασίας ανάπτυξης ενός ΠΣ, που εκτελούνται οργανωμένα και μεθοδικά κατά ακολουθιακό τρόπο [11]. Όλοι οι κύκλοι ζωής λειτουργούν για την επίλυση του προβλήματος, χωρίζοντας το σε πολλά επιμέρους τμήματα, που το καθένα αναλύεται και έχει δική του λύση. Το σύνολο των λύσεων των επιμέρους τμημάτων έχει ως αποτέλεσμα τη γενική επίλυση του προβλήματος.

Παλαιότερα, η ανάπτυξη ενός λογισμικού εξαρτιόταν από ένα προγραμματιστή, ο οποίος καλούνταν να γράψει κώδικα προκειμένου να λύσει ένα πρόβλημα ή να αυτοματοποιήσει μία διαδικασία. Στις μέρες μας, τα συστήματα είναι τόσο μεγάλα σε όγκο δεδομένων και πολύπλοκα, που υπάρχει ανάγκη ύπαρξης ομάδων αρχιτεκτόνων (architects), αναλυτών (analysts), προγραμματιστών (programmers), δοκιμαστών (testers) και χρηστών, οι οποίοι καλούνται να συνεργαστούν προκειμένου να δημιουργήσουν ένα προσαρμοσμένο γραμμένο κώδικα και να βασιστούν σε αυτό για να βγάλουν το επιθυμητό αποτέλεσμα [10]. Για να είναι εφικτό αυτό, δημιουργήθηκε πλήθος μοντέλων κύκλου ζωής ανάπτυξης συστημάτων (SDLC) τα οποία υλοποιούνται βάσει ορισμένων μεθοδολογιών. Αυτά είναι: η Μεθοδολογία Καταρρακτοειδούς Ανάπτυξης

Συστημάτων (Waterfall Systems Development Methodology), η Επαυξητική Μεθοδολογία (Incremental Methodology IM), η Μεθοδολογία Λειτουργικής Επαύξεσης - ΜΛΕ (Functional Incrementation Methodology - FIM), η Μεθοδολογία V (The V Methodology), η Μεθοδολογία Σπειροειδούς Ανάπτυξης Συστημάτων (Spiral System Development Methodology - SSDM) και η Μεθοδολογία Εξελικτικών Πρωτοτύπων - ΜΕΠ (Evolutionary Prototyping - EPM) [11].

Ο κύκλος ζωής ξεκινάει με μία ιδέα ή μία ανάγκη που μπορεί να ικανοποιηθεί πλήρως ή εν μέρει με το λογισμικό και τελειώνει με την απόσυρση του λογισμικού [12].

Ακολουθεί τέσσερα βασικά στάδια:

- σύλληψη
- κατασκευή/ εγκατάσταση
- λειτουργία/ συντήρηση
- απόσυρση

Αναλυτικότερα:

- Η σύλληψη αναφέρεται στους λόγους που πρέπει το σύστημα να κατασκευαστεί. Αυτό είναι εφικτό με τη δημιουργία μιας μελέτης εφικτότητας από άποψη τεχνική, οικονομική, λειτουργική ή οργανωσιακή και εν συνεχεία συντάσσεται μία σύμβαση επιπέδου υπηρεσιών (Service Level Agreement- SLA), με όλες τις τεχνικές και λειτουργικές προδιαγραφές του συστήματος.
- Με την αποδοχή αυτών περνάμε στο στάδιο της κατασκευής και εγκατάστασης του ΠΣ, το οποίο περιλαμβάνει τις φάσεις της ανάλυσης (βελτίωση των στόχων του έργου μέσω καθορισμένων λειτουργιών), σχεδιασμού (περιγραφή επιθυμητών χαρακτηριστικών και λειτουργιών με λεπτομέρεια, συμπεριλαμβανομένων επιχειρηματικών κανόνων, διαγραμμάτων και κώδικα), πραγμάτωσης (στάδιο της υλοποίησης μέσω της εγγραφής του κώδικα) και ελέγχων (όλα τα επιμέρους κομμάτια ενώνονται και ελέγχονται για τυχόν σφάλματα). Με τα διάφορα τεστ που θα πραγματοποιηθούν στο στάδιο των ελέγχων στην ουσία θα επαληθεύσουμε και θα επικυρώσουμε την ικανοποίηση των προδιαγραφών και θα επιδιορθώσουμε τυχόν κατασκευαστικά σφάλματα που έχουν προκύψει. Τέλος, η εγκατάσταση σε αυτό το σημείο κατόπιν ενδεδειγμένων ελέγχων θέτει το λογισμικό σε εφαρμογή.

- Η λειτουργία αφορά τη μετάβαση από την τρέχουσα κατάσταση στη νέα και η συντήρηση αφορά διάφορες αλλαγές, προσθήκες, μετακινήσεις σε διαφορετική υπολογιστική πλατφόρμα. Το συγκεκριμένο στάδιο, αποτελεί το πιο σημαντικό από όλα καθώς καθιστά το σύστημα συντηρήσιμο φαινομενικά για πάντα.
- Το τελευταίο στάδιο είναι η απόσυρση του συστήματος, λόγω μη απόδοσης της αναμενόμενης αξίας στον οργανισμό και συνεπώς πρέπει να αντικατασταθεί με νέο. Η απόσυρση (withdrawal) μπορεί να μην συνιστά την ολοκληρωτική απόσυρσή του και αντικατάσταση με νέο, αλλά σίγουρα την απόσυρσή του από την παρούσα μορφή [11].

Κάθε κύκλος ζωής δημιουργείται κατά την αντίληψη του κατασκευαστή ώστε να προσαρμόζεται στα χαρακτηριστικά του οργανισμού για τον οποίο θα κατασκευαστεί το πληροφοριακό σύστημα. Για το ερώτημα ποια τελικά είναι η κατάλληλη μεθοδολογία ή μέθοδος βάσει της οποίας θα κατασκευαστεί το ΠΣ, έγκειται στη συνάρτηση τεχνογνωσίας και εμπειρίας του κατασκευαστή, το μέγεθος του πληροφοριακού συστήματος και το πόσο συχνά μπορεί να αλλάξουν οι απαιτήσεις των χρηστών κατά τη διάρκεια ανάπτυξης του. Η επιλογή του κατάλληλου κύκλου ζωής για την ανάπτυξη, μπορεί να μην είναι μοναδική.

Γενικά μία μεθοδολογία ΚΖΑΣ αφορά τις ακόλουθες φάσεις:

- Αναγνώριση της προβληματικής κατάστασης: Από τη στιγμή που υφίσταται ήδη ένα ΠΣ, το σημαντικό είναι προσδιοριστούν τα ελαττώματά του. Αυτό επιτυγχάνεται μέσω συνεντεύξεων των χρηστών και παροχή συμβουλών από το προσωπικό υποστήριξης. Στην περίπτωση που πρέπει να αναπτυχθεί από την αρχή κάποιο ΠΣ, είναι αναγκαίο να προσδιοριστούν τα προβλήματα που συντάσσουν την προβληματική κατάσταση και πως αυτή θα μπορούσε να βελτιωθεί με αλλαγές που κρίνονται εφικτές και ευκαίριες. Μία τέτοια

μεθοδολογία αποτελεί η Μεθοδολογία Μαλακών Συστημάτων που αναφέρθηκε νωρίτερα.

- Προσδιορισμός των απαιτήσεων: Σε αυτή τη φάση είναι αναγκαίο να προσδιοριστούν οι απαιτήσεις που υπάρχουν και καθιστούν απαραίτητη τη δημιουργία νέου ΠΣ ή την αλλαγή του ήδη υπάρχοντος συστήματος, θέτοντας προτάσεις για τη βελτίωσή του.
- Σχεδιασμός συστήματος: Εδώ κατασκευάζονται οι κατάλληλες προδιαγραφές για το υλικό, το λογισμικό, τον προγραμματισμό και τα θέματα ασφαλείας που θα καλύπτει το σύστημα, λαμβάνοντας υπόψη τις διαθέσιμες τεχνολογίες και το πόσο εύκολη θα είναι η αφομοίωση της από τους χρήστες.
- Κατασκευή συστήματος: Σε αυτή τη φάση θα κατασκευαστούν τα νέα προγράμματα, αφού πρώτα ελεγχθούν και τροποποιηθούν σύμφωνα με τις ανάγκες του συστήματος ώστε να επιφέρουν αποδοτικότητα και αποτελεσματικότητα αναφορικά με τις απαιτήσεις των χρηστών και έπειτα θα εγκατασταθούν. Το σημαντικό ρόλο σε αυτή τη φάση έχουν οι χρήστες, οι οποίοι μετά την εγκατάσταση του συστήματος θα ζητήσουν τις τελικές ρυθμίσεις και όσες αλλαγές κρίνονται απαραίτητες.
- Λειτουργία συστήματος: Το νέο σύστημα ενδέχεται είτε να εισέλθει σταδιακά σε λειτουργία, είτε πολλές φορές κρίνεται εύλογο να υπάρξει απότομη αποκοπή από το παλιό σύστημα και εφαρμογή του νέου.
- Αξιολόγηση συστήματος: Αφού το σύστημα βρίσκεται σε λειτουργία για κάποιο χρονικό διάστημα, πρέπει να αξιολογηθεί για την πορεία της μέχρι τότε λειτουργίας του σε συνάρτηση με το αν ικανοποιούνται οι απαιτήσεις των χρηστών και αν χρήζει βελτίωσης να επέλθουν οι απαραίτητες τροποποιήσεις, κρατώντας όμως ενήμερους τους χρήστες για τις πρόσφατες αλλαγές.

Κάθε μεθοδολογία αποτελείται από φάσεις οι οποίες σε κάθε περίπτωση είναι διαφορετικές σε όνομα και αριθμό, όλες όμως κινούνται βάσει του γενικού πλαισίου που αναφέρθηκε νωρίτερα.

Η γενική μεθοδολογία ΚΖΑΣ διακρίνεται σε δύο κατηγορίες, όπου στη μία δίνεται βάση στα χρονοδιαγράμματα που πρέπει να τηρηθούν και στην άλλη ο προσδιορισμός και η

ικανοποίηση των χρηστών. Οι δύο αυτές κατηγορίες διακρίνονται σε τρεις τον αριθμό προσεγγίσεις:

- Ακολουθιακή προσέγγιση: Σε αυτή την προσέγγιση η μεταγενέστερη φάση δεν μπορεί να επιστρέψει στην προγενέστερη. Αυτό για τα περισσότερα συστήματα αποτελεί πρόβλημα, όμως σε ορισμένες περιπτώσεις μπορεί να αποδώσει, ιδίως σε περιπτώσεις που τα συστήματα ανάπτυξης δεν είναι κρίσιμα και είναι πολύ μικρά. Η καταρρακτοειδής προσέγγιση αποτελεί παράδειγμα αυστηρά ακολουθιακής προσέγγισης.
- Επαναληπτική προσέγγιση: Σε αυτή την προσέγγιση μπορεί να υπάρξει επιστροφή σε προγενέστερη φάση, επιφέροντας τις απαραίτητες αλλαγές βάσει των απαιτήσεων που διαρκώς αλλάζουν. Οι περισσότερες προσεγγίσεις κύκλου ζωής είναι επαναληπτικές με εξαίρεση την καταρρακτοειδής προσέγγιση.
- Αναδρομική προσέγγιση: Σε αυτή την προσέγγιση είναι δυνατή η επανεφαρμογή ολόκληρου του ΚΖΑΣ επί του συστήματος που έχει αναπτυχθεί με εφαρμογή του ίδιου ΚΖΑΣ.

Το πιο διαδεδομένο είναι το καταρρακτοειδές μοντέλο, στο οποίο το κάθε στάδιο που ολοκληρώνεται, στην ουσία αποτελεί την αρχή του επόμενου. Αυτό σημαίνει ότι οι απαιτήσεις πρέπει να είναι ξεκάθαρες πριν ξεκινήσει η επόμενη φάση και σε πολλές περιπτώσεις οι τυχόν αλλαγές σε απαιτήσεις δεν θα ληφθούν υπόψη. Στο τέλος κάθε βήματος, επανεξετάζεται αν έχει διασφαλιστεί η συμμόρφωση στις απαιτήσεις που καθορίστηκαν στη πρώτη φάση. Αυτό ξεκάθαρα το καθιστά ένα διαδοχικό μοντέλο ακολουθιακής προσέγγισης όπως αναφέραμε και νωρίτερα [13].

Ωστόσο, η χρήση αυτού του μοντέλου δημιουργεί σοβαρά προβλήματα καθώς οι απαιτήσεις αυξάνονται και αλλάζουν κατά τη διάρκεια της διαδικασίας, καθιστώντας απαραίτητες τόσο την ανατροφοδότηση όσο και τις επαναληπτικές διαδικασίες [10].

Πολλές ήταν οι μεθοδολογίες που στηρίχθηκαν στο καταρρακτοειδές μοντέλο, οι οποίες όμως αποδείχθηκαν ανελαστικές, καθώς αποτελούνταν από πολλά επιμέρους τμήματα μεγάλης διάρκειας και προσπαθούσαν να μαντέψουν τυχόν προβλήματα που

μπορεί να προέκυπταν στην πορεία ανάπτυξης του πληροφοριακού συστήματος. Για αυτό το λόγο, προκειμένου να υπάρχει καλύτερη προσαρμογή των νέων απαιτήσεων που προκύπτουν κατά τη διάρκεια ανάπτυξης ενός ΠΣ, προτιμάται να γίνεται χρήση των Ευέλικτων Μεθοδολογιών (Agile Methodologies).

2.4.1. Μεθοδολογία Καταρρακτοειδούς Ανάπτυξης Συστημάτων

Η μεθοδολογία καταρρακτοειδούς ανάπτυξης πληροφοριακού συστήματος βασίζεται στο καταρρακτοειδές μοντέλο ή κοινώς μοντέλο καταρράκτη. Κατά το μοντέλο αυτό, διενεργείται μία διεργασία που λαμβάνει χώρα κατά φάσεις, οι οποίες πραγματοποιούνται η μία μετά την άλλη. Αυτές οι φάσεις είναι οι εξής:

- Η ανάλυση και ο προγραμματισμός απαιτήσεων

Σε αυτή τη φάση καθορίζονται οι στόχοι, οι περιορισμοί και οι λειτουργίες του συστήματος από τους χρήστες. Εν συνεχεία, προσδιορίζονται και αναλύονται οι διάφορες απαιτήσεις ώστε να είναι αποδεκτές από όλους.

- Σχεδιασμός συστήματος

Σε αυτή τη φάση καθορίζεται η αρχιτεκτονική του συστήματος ως προς τις απαιτήσεις στο υλικό, δίκτυο, λογισμικό και λογισμικό των εφαρμογών. Ο σχεδιασμός του λογισμικού των εφαρμογών, αναφέρεται στη λειτουργία και τη λειτουργικότητα που πρέπει και τα δύο να μετατραπούν σε εκτελέσιμα προγράμματα.

- Πραγμάτωση και έλεγχος ενοτήτων

Σε αυτή τη φάση, κατασκευάζεται το λογισμικό των εφαρμογών του πληροφοριακού συστήματος ως ένα σύνολο προγραμμάτων και η κάθε εφαρμογή ελέγχεται ως προς την τήρηση των προδιαγραφών που έχουν έχουν καθοριστεί.

- Ολοκλήρωση και έλεγχος ολοκλήρωσης συστήματος

Ολοκληρώνονται οι εφαρμογές και ελέγχεται το σύστημα εκ νέου για το αν έχουν τηρηθεί οι προδιαγραφές που έχουν τεθεί ώστε κατόπιν του ελέγχου, το σύστημα να μπορεί να παραδοθεί στον τελικό χρήστη.

- Εγκατάσταση, λειτουργία και συντήρηση συστήματος

Το σύστημα εγκαθίσταται σε ένα παραγωγικό περιβάλλον και τίθεται σε δοκιμαστική λειτουργία. Το επόμενο βήμα είναι η συντήρηση του συστήματος. Κατά τη φάση αυτή, γίνονται οι διορθώσεις σε σφάλματα που έχουν προκύψει ή και σε σφάλματα που δεν ήταν εφικτό να γίνουν αντιληπτά κατά την εκτέλεση των δοκιμών, με στόχο τη βελτίωση λειτουργικότητας του συστήματος.

Στο συγκεκριμένο μοντέλο, όλες οι απαιτήσεις του συστήματός είναι εξ αρχής γνωστές και σαφείς και δεν μεταβάλλονται κατά την ανάπτυξη. Για να ξεκινήσει η εκτέλεση μιας φάσης, θα πρέπει να έχει ολοκληρωθεί με επιτυχία η προηγούμενή της. Συνεπώς, το συγκεκριμένο μοντέλο δεν επιτρέπει την επανεξέταση και την αναθεώρηση μιας φάσης που έχει ολοκληρωθεί [19].

2.4.2. Επαυξητική Μεθοδολογία

Αυτή η μεθοδολογία στοχεύει στην επιτάχυνση εκτέλεσης του κύκλου ζωής. Το μοντέλο της μεθοδολογίας αυτής, χρησιμοποιεί επαυξήσεις των χαρακτηριστικών για να δημιουργήσει τις διαφορετικές εκδόσεις. Κατά τη διάρκεια της ανατροφοδότησης, στα επόμενα στάδια προστίθενται και άλλες νέες επαυξήσεις, παρατηρώντας ότι η επαναλαμβανόμενη αυτή διαδικασία, συνεχίζεται μέχρι την επίτευξη του στόχου ή την απόσυρση του συστήματος.

Αρχικά το σύστημα προκειμένου να αναπτυχθεί, χωρίζεται σε μικρότερα τμήματα για να παρακολουθείται πιο εύκολα. Ωστόσο, ένα μειονέκτημα χρήσης αυτής της μεθοδολογίας είναι ότι εξαιτίας της επαυξητικής προσέγγισης, υπάρχει πιθανότητα λάθους ως προς τον προϋπολογισμό του κόστους ανάπτυξης ή όπως αναφέραμε νωρίτερα, λόγω των επαυξήσεων που έχουν ως αποτέλεσμα τη δημιουργία νέων

εκδόσεων, υπάρχει το ενδεχόμενο να παρουσιαστούν θέματα συμβατότητας με τις προγενέστερες εκδόσεις [19].

Συγκριτικά με τη μεθοδολογία καταρρακτοειδούς ανάπτυξης, φαίνεται να είναι πιο ευέλικτη, καθώς επιτρέπει την σταδιακή ενσωμάτωση απαιτήσεων διευκολύνοντας τις όποιες αλλαγές.

2.4.3. Μεθοδολογία Λειτουργικής Επαύξησης

Αυτή η μεθοδολογία στοχεύει στην επίλυση των αδυναμιών που δημιουργούνται από το καταρρακτοειδές μοντέλο. Το σύστημα χωρίζεται σε διάφορα τμήματα (υποσυστήματα) αφού έχει προηγηθεί ανάλυση και σχεδιασμός για τον προσδιορισμό τους. Το κάθε επιμέρους τμήμα ενώνεται με το προηγούμενο τμήμα και αυτή η διαδικασία επαναλαμβάνεται διαρκώς, έως ότου να αναπτυχθεί και ενσωματωθεί στο σύστημα και το τελευταίο τμήμα.

Η χρήση αυτής της μεθοδολογίας παρουσιάζει τη δυνατότητα παράλληλης ανάπτυξης των επιμέρους τμημάτων και την ανάπτυξη ολόκληρου του συστήματος σταδιακά αφού το κάθε επιμέρους τμήμα - όπως ήδη αναφέρθηκε- αναπτύσσεται και ενσωματώνεται στο προηγούμενο.

Ωστόσο δύο είναι τα κύρια μειονεκτήματα που εμφανίζει η συγκεκριμένη μεθοδολογία. Πρώτον, ότι οι απαιτήσεις του συστήματος πρέπει να είναι γνωστές εξ αρχής και να μην μεταβάλλονται κατά τη διάρκεια ανάπτυξης του συστήματος. Δεύτερον, ότι ο διαχωρισμός του συστήματος μπορεί να πραγματοποιηθεί από μη εξειδικευμένα άτομα, με αποτέλεσμα να μην είναι η κατάλληλη διάσπαση και να προκληθούν προβλήματα μεταγενέστερα [20].

2.4.4. Μεθοδολογία V

Σε αυτή τη μεθοδολογία, το κάθε βήμα μπορεί να επέλθει μετά την τεκμηρίωση του προηγούμενου βήματος, ελέγχοντας την κάθε διεργασία πριν προβούμε σε αυτή. Με τον συνεχή έλεγχο, είναι δυνατό να εντοπιστούν και να επιλυθούν διάφορα προβλήματα που είναι δυνατό να εμφανιστούν. Κατά τη διαδικασία ανάπτυξης του

συστήματος, υπάρχει ο προσδιορισμός των προδιαγραφών που τελικά κωδικοποιούνται [20]. Στη συγκεκριμένη μεθοδολογία αντί να συνεχίζεται αυτή η καθοδική κίνηση, υπάρχει παράλληλα και η ανοδική κίνηση στην οποία περιγράφονται τα στάδια των ελέγχων. Η χρήση της μεθοδολογίας αυτής, αποσκοπεί στο να κατανοήσουμε τη σημαντικότητα ύπαρξης των ελέγχων κατά τη διάρκεια της ανάπτυξης. Μέσω των ελέγχων αυτών που είναι διασπασμένοι σε όλη τη διεργασία, μπορούμε να εντοπίσουμε τα σφάλματα και να προβούμε στη διόρθωσή τους, μειώνοντας και το κόστος συντήρησης.

Τέλος, πρέπει να γίνει μία αναφορά στο γεγονός ότι η μεθοδολογία V μοιάζει αρκετά με το καταρρακτοειδές μοντέλο, όσον αφορά την αυστηρότητα της διεργασίας και τον ελάχιστο χώρο για ευέλικτη προσαρμογή, αφού οποιαδήποτε τροποποίηση στις απαιτήσεις ακυρώνει την υπάρχουσα τεκμηρίωση και τους ελέγχους που έχουν λάβει χώρα μέχρι εκείνη τη στιγμή.

2.4.5. Σπειροειδής Μεθοδολογία

Η Σπειροειδής Μεθοδολογία αναπαρίσταται με τη μορφή ελικοειδούς γραμμής και όχι ως ακολουθία δραστηριοτήτων. Σχηματίζει μια σπείρα όπου κάθε βρόχος της αντιπροσωπεύει και μία φάση της διαδικασίας. Καθένας από τους βρόγχους επιλέγεται ανάλογα με τις απαιτήσεις. Έτσι, καθ' όλη την διαδικασία γίνεται αξιολόγηση και επίλυση των κινδύνων.

Υπάρχουν τέσσερις βασικές φάσεις σε αυτό το μοντέλο :

1. Καθορισμός στόχων και αποφυγή της επανάληψης. Δίνονται εναλλακτικές λύσεις όταν εντοπίζονται οι κίνδυνοι.
2. Κατά τη διάρκεια απόφασης των εναλλακτικών λύσεων γίνεται ανάλυση των κινδύνων και εξετάζονται οι λύσεις έτσι ώστε να αποφασίσουν αν θα προχωρήσουν ή όχι στη διαδικασία ανάπτυξης.
3. Εφαρμογή της μεθόδου που επιλέχθηκε στη παραπάνω ανάλυση
4. Προγραμματισμός της επόμενης επανάληψης. Σε περίπτωση που επικυρωθούν τα αποτελέσματα του λογισμικού που αναπτύχθηκε, προγραμματίζεται η συνέχιση της ανάπτυξης.

Η μεθοδολογία αυτή βοηθάει στη μείωση των κινδύνων καθώς σε κάθε επανάληψη επιλέγει το κατάλληλο πρότυπο για την ανάπτυξη ενός τμήματος λογισμικού [20].

2.4.6. Μεθοδολογία εξελικτικών πρωτοτύπων

Η μεθοδολογία εξελικτικών πρωτοτύπων είναι ένα μοντέλο ανάπτυξης λογισμικού. Κατασκευάζονται τμηματικά μη ολοκληρωμένες εκδόσεις όπου κάθε μια από αυτές ονομάζεται πρωτότυπο. Στόχος είναι η συνεργασία με τον πελάτη και η άμεση δημιουργία μιας πρώτης έκδοσης του συστήματος που θα ξεκινά από μια αρχική περιγραφή με πολύ καλά κατανοητές απαιτήσεις. Το αρχικό πρωτότυπο αναπτύσσεται σταδιακά και εξελίσσεται στο τελικό σύστημα που θα εγκατασταθεί.

Η μέθοδος αυτή επιτρέπει στους χρήστες του συστήματος να αξιολογήσουν τις προτάσεις των μηχανικών λογισμικού. Οι χρήστες μπορούν να εντοπίσουν ασάφειες στις απαιτήσεις του λογισμικού και να διορθωθούν με αποτέλεσμα να υπάρχει σωστή επικοινωνία μεταξύ πελατών και υπεύθυνων ανάπτυξης. Πρόκειται για μία επαναληπτική διαδικασία η οποία αλλάζει συνεχώς μέχρι να βελτιωθεί πλήρως ώστε να ικανοποιούνται οι απαιτήσεις των πελατών.

Τα βήματα που ακολουθεί η διαδικασία αυτή είναι τα εξής :

1. Ο εντοπισμός και η καταγραφή των αναγκών του πελάτη
2. Η γρήγορη ανάπτυξη λειτουργικού πρωτοτύπου
3. Η δοκιμή πρωτοτύπου
4. Η διόρθωση και βελτίωση πρωτοτύπου
5. Η διαδικασία επιστρέφει στο βήμα 3 και επαναλαμβάνεται συνεχώς έως ότου ο χρήστης ικανοποιηθεί πλήρως [20].

2.5. Μεθοδολογίες Agile

Η λέξη agile στα ελληνικά σημαίνει ευκίνητος, εύστροφος. Επίσης ορίζεται ως η ικανότητα να κινείται κανείς γρήγορα και άνετα. Έτσι λοιπόν και οι agile μεθοδολογίες αποτελούν μεθοδολογίες οι οποίες είναι ευέλικτες, προσαρμόσιμες, γρήγορες και αποτελεσματικές.

Στόχος της παρούσας ενότητας είναι να παρουσιαστούν αναλυτικά οι Ευέλικτες (agile) Μεθοδολογίες. Αρχικά, θα γίνει μια γενική αναφορά στην έννοια των μεθοδολογιών αυτών και έπειτα θα παρουσιάσουμε συγκεκριμένα την scrum methodology όπου πρόκειται ουσιαστικά για μια agile μεθοδολογία, με την οποία θα ασχοληθούμε.

Οι μεθοδολογίες agile άρχισαν να εμφανίζονται την τελευταία δεκαετία. Οι μεθοδολογίες αυτές βασίζονται σε μια αρκετά διαφορετική προσέγγιση. Αρχικά, περιλαμβάνει μικρές εκδόσεις του λογισμικού με ταχύτατους κύκλους ανάπτυξης. Δεύτερον, περιλαμβάνει συνεργασία με τον πελάτη ο οποίος στην ουσία συμμετέχει στην υλοποίηση του έργου. Τέλος, περιλαμβάνει προσαρμοστικότητα δηλαδή υπάρχει δυνατότητα να γίνονται αλλαγές της τελευταίας στιγμής.

Οι μεθοδολογίες αυτές είναι πολλές αλλά με πολλά κοινά χαρακτηριστικά κυρίως στην φιλοσοφία υλοποίησης ενός έργου πληροφορικής.

Οι κυριότερες από τις agile μεθοδολογίες είναι οι εξής :

- Extreme programming (XP)
- SCRUM
- Crystal family
- Feature Driven Development
- Dynamic Systems Development Method
- Adaptive Software Development
- Lean Development

Αναλυτικότερα, οι μεθοδολογίες αυτές εστιάζουν πολύ στον ανθρώπινο παράγοντα. Οι προγραμματιστές δημιουργούν όσο το δυνατόν απλούστερο κώδικα με σκοπό να μειώσουν την τεκμηρίωση και παράγουν συνεχώς λειτουργικά λογισμικά σε τακτά χρονικά διαστήματα. Τέλος, οι προγραμματιστές όπως και ο πελάτης που αποτελούν την ομάδα ανάπτυξης έχουν την δυνατότητα να διαφοροποιούν τον σχεδιασμό όταν είναι απαραίτητο. Συνοπτικά, οι μεθοδολογίες αυτές προτείνουν διοικητικές και τεχνικές πρακτικές οι οποίες προσαρμόζονται συνεχώς σε αλλαγές που απαιτούνται κατά τη διάρκεια υλοποίησης του έργου, ανάλογα με τις απαιτήσεις των χρηστών και τις αλλαγές του περιβάλλοντος ανάπτυξης.

Σύμφωνα με τους δημιουργούς των agile methodologies, σε μία εταιρεία τα εργαλεία, η λεπτομερής τεκμηρίωση και τα σύμβολα είναι επουσιώδη. Αντίθετα, περισσότερη σημασία για τον πελάτη έχουν οι τέσσερις αρχές του agile manifesto:

- Πρώτον, ο ανθρώπινος παράγοντας έχει σημαντικό ρόλο. Οι άνθρωποι πρέπει να δίνουν το μέγιστο των δυνατοτήτων τους, χωρίς να περιορίζονται από εργαλεία και διαδικασίες που πρέπει να χρησιμοποιούν. Δηλαδή ενώ είναι σημαντικά τα εργαλεία και οι διαδικασίες, είναι πολύ σημαντικότερη η δημιουργία συνθηκών συνεργασίας μεταξύ ταλαντούχων ατόμων για την επιτυχή υλοποίηση ενός έργου.
- Δεύτερον, οι υποστηρικτές των agile μεθοδολογιών υποστηρίζουν ότι σε πολλές μεθοδολογίες ανάπτυξης έχει χαθεί η ανάπτυξη λειτουργικού και χωρίς σφάλματα λογισμικού. Δίνεται μεγαλύτερη βαρύτητα στην παράδοση εγγράφων, κειμένων που τεκμηριώνουν το λογισμικό, απαιτώντας από την ομάδα να αποδώσει περισσότερο μέρος της προσπάθειας της. Επομένως, ως κύριο στόχο θέτουν την ανάπτυξη λογισμικού και έπειτα ορίζουν κατά πόσο είναι απαραίτητο να ασχοληθούν με την τεκμηρίωση.
- Τρίτον, η συνεργασία με τον πελάτη πάνω από την διαπραγμάτευση συμβολαίων. Στόχος της διαπραγμάτευσης, είναι τα συμβαλλόμενα μέρη να κατανοήσουν ότι η βασική γνώση πάνω στο έργο ανακαλύπτεται κατά την

διάρκεια υλοποίησης του. Επομένως, δεν λειτουργούν με συμβόλαια τα οποία δεν τους επιτρέπουν τη δυνατότητα ευελιξίας του έργου.

- Τέταρτον, αντίδραση στην αλλαγή. Πολλές φορές είναι συχνό το φαινόμενο αλλαγών στις απαιτήσεις των χρηστών κατά την διάρκεια ανάπτυξης ενός έργου λογισμικού. Αρκετά συχνά, συναντάμε αδυναμία του πελάτη να εκφράσει το σύνολο των πραγματικών του αναγκών κατά την υλοποίηση ενός έργου και έτσι με αυτό τον τρόπο οι αλλαγές γίνονται όλο και περισσότερες. Παράλληλα όμως γίνεται προσπάθεια για όσο τον δυνατόν γρηγορότερη ανάπτυξη και παράδοση των τελικών συστημάτων. Συνεπώς, η ύπαρξη ενός πλάνου που θα καλύπτει το σύνολο των απαιτήσεων των χρηστών είναι απαραίτητη όμως οποιαδήποτε απόκλιση από το πλάνο και αλλαγή κατά την υλοποίηση του έργου από τους εμπλεκόμενους πρέπει να γίνεται αποδεκτή.

Για να θεωρηθεί σε ένα περιβάλλον ότι εκτελείται Agile μεθοδολογία, θα πρέπει να δίνεται πάντα προτεραιότητα στην ικανοποίηση του πελάτη, να δίνεται η δυνατότητα για πραγματοποίηση αλλαγών και τροποποιήσεων ακόμα και σε τελικά στάδια του έργου. Επίσης, είναι απαραίτητη η παράδοση λειτουργικού λογισμικού σε μικρά διαστήματα. Τα έργα εκτελούνται από στελέχη της ομάδας και η μεταφορά πληροφοριών γίνεται με συναντήσεις και συνομιλίες κατά πρόσωπο.

Οι ευέλικτες μεθοδολογίες ακολουθούν αειφόρο διεργασία ανάπτυξης. Δηλαδή όλοι οι συμμετέχοντες του έργου πρέπει να διατηρούν σταθερό ρυθμό. Επιπλέον, πρέπει να είναι προσηλωμένοι ώστε να υπάρχει πάντα σωστή τεχνική και καλός σχεδιασμός. Τέλος, η ομάδα ανάπτυξης πρέπει να έχει την ικανότητα αυτοδιαχείρισης και να κάνει ανά τακτά χρονικά διαστήματα αξιολογήσεις ώστε να προσαρμόζει και να διορθώνει τα λάθη της [14].

2.5.1. Οφέλη και περιορισμοί

Οι μεθοδολογίες αυτές έχουν πλεονεκτήματα και μειονεκτήματα. Πρώτο και σημαντικότερο πλεονέκτημα τους είναι η απόδοση της επένδυσης. Η στενή

συνεργασία με τον πελάτη σε κάθε στάδιο, βοηθάει το έργο να καλύπτει απόλυτα της ανάγκες του πελάτη. Έτσι, το έργο μπορεί να εκτελεστεί σε σύντομο χρονικό διάστημα με αποτέλεσμα η απόδοση της επένδυσης να είναι μεγαλύτερη και καλύτερη. Δεύτερο πλεονέκτημα, είναι η ταχύτερη ανάπτυξη του λογισμικού όπου με αυτόν τον τρόπο ο πελάτης μπορεί να παραλαμβάνει το λογισμικό σε μικρότερο χρόνο παράδοσης. Τρίτο πλεονέκτημα, είναι η δυνατότητα αποφυγής πρόωρων ακυρώσεων προβληματικών έργων. Οι μεθοδολογίες agile αναλύουν, σχεδιάζουν και υλοποιούν το λογισμικό σε μικρές επαναλήψεις και αξιολογούν την πρόοδο του λογισμικού. Έτσι δίνουν την ευκαιρία στον πελάτη να αναθεωρήσει το κόστος και τα οφέλη ή ακόμα και να ακυρώσει το έργο αντιμετωπίζοντας όμως λιγότερες ζημιές. Τέταρτο όφελος είναι ο έλεγχος.

Ο πελάτης με την τακτική των ευέλικτων μεθοδολογιών, ενημερώνεται σχεδόν σε κάθε στάδιο υλοποίησης του έργου με αποτέλεσμα να του παρέχεται βελτιωμένη εικόνα και έλεγχος του έργου. Πέμπτο πλεονέκτημα, είναι ότι η ανάπτυξη λογισμικού σε αυτές τις μεθοδολογίες γίνεται στα πλαίσια μιας ομάδας και οι πληροφορίες είναι διαθέσιμες σε όλους. Επομένως, αποφεύγεται η αποκλειστική υλοποίηση κάποιου σταδίου από ένα άτομο μεμονωμένα. Τέλος, επιτρέπουν τις αλλαγές σε απαιτήσεις του πελάτη ακόμα και σε τελικά στάδια αυξάνοντας τις πιθανότητες να παραδοθεί στον πελάτη ένα αποτέλεσμα πολύ κοντά στις επιθυμητές απαιτήσεις του.

Εκτός από τα πλεονεκτήματα των μεθοδολογιών αυτών υπάρχουν και μειονεκτήματα. Πρώτον, για την εφαρμογή τους απαιτείται εξειδικευμένο ανθρώπινο δυναμικό. Είναι απαραίτητη η ύπαρξη ατόμων που να αφιερώνουν το 100% του χρόνου τους στο έργο, γεγονός που ανεβάζει αρκετά το κόστος για τον πελάτη. Δεύτερον, είναι δύσκολο να εφαρμοστεί η άμεση επικοινωνία. Σε περιπτώσεις που λειτουργούν αποκεντρωμένα με ομάδες ανάπτυξης διασκορπισμένες σε διάφορες περιοχές ή χώρες η ανάγκη για συνεχή και κατά πρόσωπο επικοινωνία μεταξύ τους είναι δύσκολη. Έπειτα, σημαντικό πρόβλημα αντιμετωπίζουν στην δημιουργία κατάλληλων ομάδων έργου, αφού απαιτείται ύπαρξη στελεχών με σημαντική εμπειρία στην τεχνική και την εκπαίδευση πάνω σε συγκεκριμένες μεθοδολογίες. Ακόμα ένα μειονέκτημα είναι το υψηλό κόστος αφού η ανάπτυξη ενός λογισμικού είναι αρκετά δαπανηρή. Τέλος, μεγάλα εμπόδια αντιμετωπίζουν σε περιπτώσεις όπου το έργο είναι κρίσιμο και η ασφάλεια πολύ

σημαντικός παράγοντας όπως για παράδειγμα όταν αναλαμβάνουν τις εφαρμογές του στρατού [14].

2.6. Μεθοδολογία Scrum

Η διαδικασία ανάπτυξης ενός πληροφοριακού συστήματος είναι περίπλοκη και ορισμένες φορές αρκετά δύσκολη. Καθώς τα πάντα γύρω μας αλλάζουν διαρκώς, είναι απαραίτητο όταν κληθούμε να σχεδιάσουμε ένα ΠΣ να είναι με τέτοιο τρόπο, ώστε να μπορεί να ανταπεξέλθει σε ένα περιβάλλον συνεχώς μεταβαλλόμενο. Η επίτευξη αυτού μπορεί να προέλθει από την ευελιξία του συστήματος και την ύπαρξη ελέγχου.

Η επιλογή της καταλληλότερης μεθοδολογίας για την ανάπτυξη ενός ΠΣ, αποτελεί καθοριστικό παράγοντα για την αποτελεσματικότητά του. Οι μεθοδολογίες που υποστηρίζουν την ευελιξία (ευέλικτες), κάνουν το σύστημα πιο ανθεκτικό σε διάφορες αλλαγές που μπορούν να προκύψουν καθ'όλη τη διάρκεια ανάπτυξης του.

Το πιο σημαντικό, είναι το έργο στο τέλος να είναι αποτελεσματικό. Αυτό πιο απλά σημαίνει, ότι το έργο έχει παραδοθεί στην ώρα του, είναι στα πλαίσια του budget που έχει καθοριστεί από την αρχή και καλύπτει τις απαιτήσεις των πελατών [15].

Πιο συγκεκριμένα, η μεθοδολογία Scrum - που θα χρησιμοποιηθεί και στο δικό μας σύστημα - ανήκει στην οικογένεια των Agile Methodologies, η οποία αναγνωρίζει ότι οι αρχικές απαιτήσεις για την ανάπτυξη ενός ΠΣ δεν έχουν οριστεί πλήρως από την αρχή και για αυτό το λόγο χρησιμοποιεί μηχανισμούς ελέγχου για τη διαχείριση του απρόβλεπτου και τον έλεγχο κινδύνου, διασφαλίζοντας την ευελιξία, την ανταπόκριση και την αποδοτικότητα του. Πρόκειται για μία επαναληπτική και αυξητική προσέγγιση για την ανάπτυξη έργων που βασίζεται στη φιλοσοφική θεώρηση ότι “τα προβλήματα ανάπτυξης συστημάτων είναι εν πολλοίς αδόμητα και είναι δύσκολο να κατανοηθούν ή να οριστούν στην απαιτούμενη έκταση και στο απαιτούμενο βάθος”. Μπορεί να χρησιμοποιηθεί για τη βελτίωση και συντήρηση ενός υπάρχοντος έργου και επιπρόσθετα παρέχει ένα γενικότερο πλαίσιο διαχείρισης, μέσα από βήματα και

διαδικασίες σε πιο απαιτητικά έργα χρονικά, τα οποία έχουν πιο περίπλοκες απαιτήσεις και ιδιαιτερότητες. Μία βασική αρχή της μεθοδολογίας Scrum, είναι ότι ορισμένα προβλήματα τα οποία καλούμαστε να αντιμετωπίσουμε με την ανάπτυξη κάποιου ΠΣ, δεν μπορούν να αντιμετωπιστούν επιτυχώς με τη χρήση άλλων διεργασιοστρεφών μεθόδων όπως αυτή του καταρρακτοειδούς κύκλου ζωής) [16].

Η μεθοδολογία Scrum χαρακτηρίζεται από τις κάτωθι φάσεις:

- Φάση σχεδιασμού (Planning Phase ή Pregame): ορισμός μίας νέας έκδοσης που πρόκειται να σχεδιαστεί, κάνοντας μία εκτίμηση του χρονικού πλαισίου και του κόστους. Αν πρόκειται για ένα νέο ΠΣ, δίνεται πολύ μεγάλη έμφαση στην εννοιολόγηση και την ανάλυση του ενώ αν πρόκειται για τη βελτίωση ενός ήδη υπάρχοντος συστήματος, η ανάλυση θα είναι περιορισμένη. Σε αυτή τη φάση, βελτιώνεται η αρχιτεκτονική του συστήματος και γίνεται μία σχεδίαση υψηλού επιπέδου (προσδιορισμός αλλαγών που απαιτούνται για τις εργασίες που δεν έχουν εκτελεστεί ακόμα, προσδιορισμός προβλημάτων που έχουν προκύψει από την εφαρμογή ορισμένων αλλαγών, παρουσίαση νέων προσεγγίσεων και εφαρμογή αυτών).
- Φάση παιχνιδιού ή ανάπτυξης Σπριντ (Game or Development Sprints Phase): σε αυτή τη φάση τον κυρίαρχο ρόλο έχουν ο χρόνος, οι απαιτήσεις, η ποιότητα, το κόστος και ο ανταγωνισμός. Όλη η ανάπτυξη θα πρέπει να βασίζεται σε αυτές τις μεταβλητές, η αλληλεπίδραση με τις οποίες ορίζουν το τέλος αυτής της φάσης. Υπάρχουν πολλά επαναληπτικά σπριντς ή αλλιώς κύκλοι όπως λέγονται, που χρησιμοποιούνται για την εξέλιξη του συστήματος, μέχρι δηλαδή το προϊόν να κριθεί έτοιμο για διανομή. Τα σπριντς διεξάγονται ανά τακτά χρονικά διαστήματα, ανάλογα βέβαια και με την πολυπλοκότητα του προϊόντος [17].
- Φάση κλεισίματος (Closure Phase): αποτελείται από όλες εκείνες τις διαδικασίες προκειμένου να ολοκληρωθεί το έργο και να παραδοθεί στον πελάτη(προετοιμασία, σταδιακή δοκιμή, απελευθέρωση) [18].

Η μεθοδολογία της Scrum, φαίνεται να χρησιμοποιείται όλο και πιο συχνά λόγω των πλεονεκτημάτων που συνδέονται μαζί της.

Αναλυτικότερα, παρέχει μηχανισμούς ελέγχου δίνοντας τη δυνατότητα στους οργανισμούς να αλλάξουν το έργο κατά τη διαδικασία ανάπτυξης του συστήματος ανά πάσα στιγμή, δίνει τη δυνατότητα στους προγραμματιστές να επινοήσουν τις καλύτερες δυνατές λύσεις λαμβάνοντας υπόψη τις συνεχείς αλλαγές του περιβάλλοντος και παρέχεται ένα εξαιρετικό περιβάλλον εκπαίδευσης για όλα τα μέρη [17].

Ωστόσο, την εμφάνιση τους κάνουν και ορισμένα μειονεκτήματα. Ένα από αυτά το οποίο είναι πολύ σημαντικό, αποτελεί η συμμετοχή του πελάτη που θα πρέπει να είναι σε συνεχή επαφή με την ομάδα ανάπτυξης του έργου, καθώς κρίνεται απαραίτητο να αποσαφηνιστούν και να διευκρινιστούν οι απαιτήσεις του συστήματος. Ένα άλλο μειονέκτημα είναι η δυσκολία που μπορεί να αντιμετωπιστεί σε μεγάλης κλίμακας έργα. Πιο συγκεκριμένα, όσο μεγαλύτερο είναι ένα έργο τόσο πιο επιτακτική είναι η ανάγκη συμμετοχής των πελατών αυξάνοντας την πολυπλοκότητα στις επικοινωνιακές δραστηριότητες [15].

Η μεθοδολογία Scrum, αφορά μία μεθοδολογία επαναληπτική όπως ήδη αναφέρθηκε, στην οποία το έργο χωρίζεται σε μικρότερα τμήματα ώστε να είναι πιο εύκολη η διαχείριση τους, διατηρώντας συγκεκριμένα χρονικά περιθώρια, μέσα στα οποία διασφαλίζεται ένα ικανό χρονικό διάστημα προς αποφυγή οποιασδήποτε σπατάλης χρόνου.

Κατά τη διάρκεια του σπριντ, δεν γίνονται αλλαγές που να θέτουν σε κίνδυνο την πορεία και εξέλιξη του έργου και έτσι οι στόχοι που αφορούν σε θέματα ποιότητας σε καμία περίπτωση δεν μειώνονται. Με τα σπριντ, μπορεί να υπάρχει προβλεψιμότητα αλλά και προσαρμογή της προόδου στον στόχο, μέσα από τις τακτικές συναντήσεις όλων των μελών της ομάδας που έχουν αναλάβει αυτή την εργασία. Στόχος, είναι η ομάδα να λειτουργεί συλλογικά και όχι με ατομικές πρωτοβουλίες. Σαφώς, υπάρχει και η πιθανότητα να τερματίσει πρόωρα κάποιο σπριντ, αν η ομάδα αντιληφθεί ότι χάνει πολύτιμο χρόνο σε ένα σπριντ που δεν θα έχει τα επιθυμητά αποτελέσματα. Ωστόσο, αυτό είναι κάτι που δεν το συναντάμε συχνά αφού τα σπριντ έχουν πολύ μικρή διάρκεια.

Υπάρχει και το καθημερινό Scrum, που πραγματοποιείται σε καθημερινή βάση από όλα τα μέλη της ομάδας, αναφέροντας το κάθε μέλος ξεχωριστά, τι έκανε και που θα μπορούσε αυτό να βοηθήσει στην επίτευξη του σπριντ και στην μείωση της πολυπλοκότητας. Το αποτέλεσμα του σπριντ μετράει ως επαύξηση και το σύνολο όλων των επαυξήσεων από ένα σύνολο σπριντς που έχουν προηγηθεί, πρέπει να μπορεί να λειτουργεί αρμονικά.

Στο δικό μας πληροφοριακό σύστημα, επιλέχθηκε η συγκεκριμένη μεθοδολογία, γιατί το έργο χωρίζεται σε πολλά επιμέρους τμήματα τα οποία αναπτύσσονται κατόπιν συνεχών επαναλήψεων, κατά τις οποίες παραδίδονται τμήματα του προϊόντος.

Προσδιορίζονται όλες οι εργασίες που πρέπει να λάβουν χώρα μέσω των σπριντς και συνεπώς προβλέπεται ο χρόνος για κάθε εργασία. Μέσω αυτών των σπριντς και των τακτικών συναντήσεων μεταξύ των μελών της ομάδας, γίνεται γνωστό τι έχει υλοποιηθεί μέχρι εκείνη τη στιγμή, τι εκκρεμεί και σε ποιο επιμέρους τμήμα. Επιπλέον, κατά τη διάρκεια των τακτικών συναντήσεων, συζητείται αν τυχόν παρουσιάστηκε κάποιο πρόβλημα σε κάποιο από τα σπριντ, αν αντιμετωπίστηκε ή τι μέτρα θα μπορούσαν να ληφθούν για να βγει με επιτυχία εις πέρας έτσι ώστε να καταστεί βέλτιστο, αποτελεσματικό και έτοιμο προς παράδοση. Ολόκληρη η ομάδα είναι υπεύθυνη για την επίλυση των προβλημάτων. Μετά το τέλος του κάθε σπριντ, όλα τα μέλη της ομάδας αναθεωρούν την προηγούμενη κατάσταση, ελέγχουν την τωρινή και διατυπώνουν τυχόν αλλαγές ή τροποποιήσεις για μελλοντικές βελτιώσεις.

Από τη γενική μελέτη, παρατηρήθηκε ότι η εφαρμογή μιας τέτοιας μεθοδολογίας, βοηθάει το πληροφοριακό σύστημα να αναπτυχθεί μέσα από μία διαδικασία ευελιξίας και προσαρμοστικότητας σε αλλαγές κατά τη διάρκειά της. Όπως αναφέρθηκε και νωρίτερα, το κυριότερο χαρακτηριστικό της είναι πως το σύνολο των εργασιών ταξινομείται στα μέλη της ομάδας που αναλαμβάνει την ανάπτυξή του. Η καθημερινή επικοινωνία τους για την πρόοδο του έργου έχει σαν αποτέλεσμα την υψηλή λειτουργικότητα και ποιότητα του συστήματος.

3. Αξιοποιηθέντα εργαλεία DevOps και τεχνολογίες

3.1. Gitlab

Στα πλαίσια της εργασίας αξιοποιείται το Gitlab, το οποίο αποτελεί ένα από τα ευρέως χρησιμοποιούμενα εργαλεία αναφορικά με DevOps διαδικασίες. Το συγκεκριμένο εργαλείο παρέχει ποικίλες δυνατότητες. Μεταξύ άλλων, μέσω του Gitlab, διευκολύνεται η συνεργατική συγγραφή κώδικα, η παρακολούθηση της προόδου σε συνάρτηση με τις λειτουργικές απαιτήσεις του συστήματος και ο χρονοπρογραμματισμός αναφορικά με την υλοποίηση των απαιτήσεων.

Για την υλοποίηση του συστήματος της εργασίας, δημιουργήθηκε στο Gitlab το project InfoSysDev.

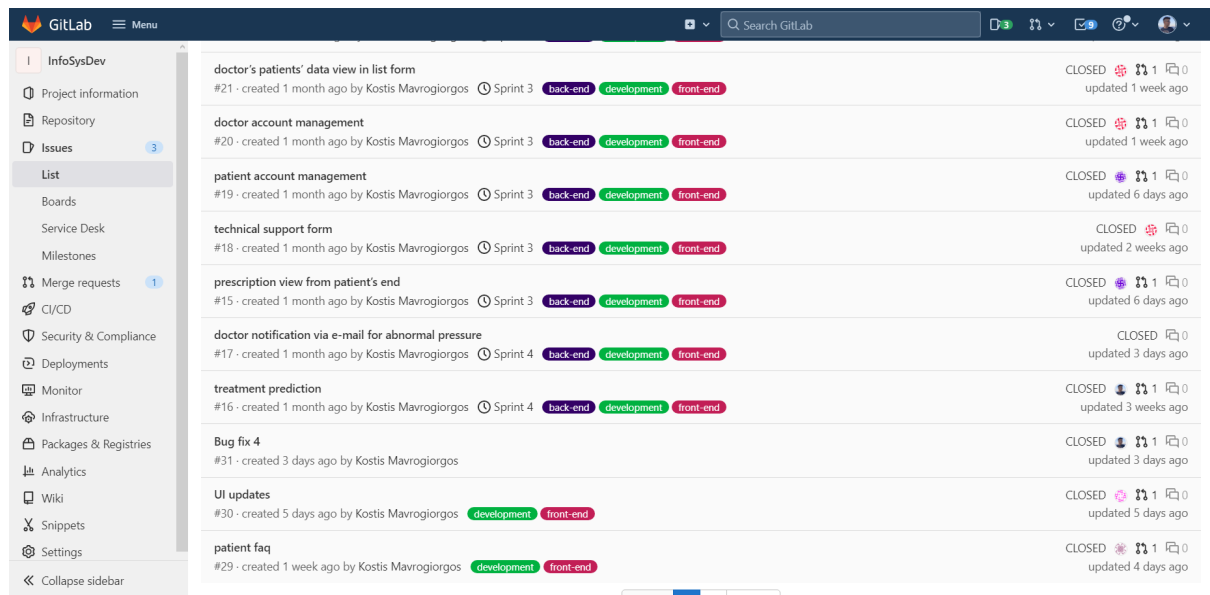
Το project είναι διαθέσιμο στη διεύθυνση: <https://gitlab.com/mavrogiorgos/infosysdev> και για να εγκριθεί η πρόσβαση σε αυτό, χρειάζεται αποστολή email σε αυτή τη διεύθυνση: kostismvg@gmail.com.

Για κάθε μία από τις απαιτήσεις του συστήματος δημιουργήθηκαν τα αντίστοιχα issues, ενώ το καθένα από αυτά ανατέθηκε και σε κάποιον προγραμματιστή. Επίσης, δημιουργήθηκαν συγκεκριμένα labels, έτσι ώστε να κατηγοριοποιηθούν τα issues και να είναι αποτελεσματικότερη η παρακολούθηση και η υλοποίησή τους. Τα labels είναι:

1. documentation: αφορά issues που σχετίζονται με τη συγγραφή του τελικού παραδοτέου εγγράφου (το συγκεκριμένο έγγραφο)
2. development: αφορά issues που σχετίζονται με τη συγγραφή κώδικα

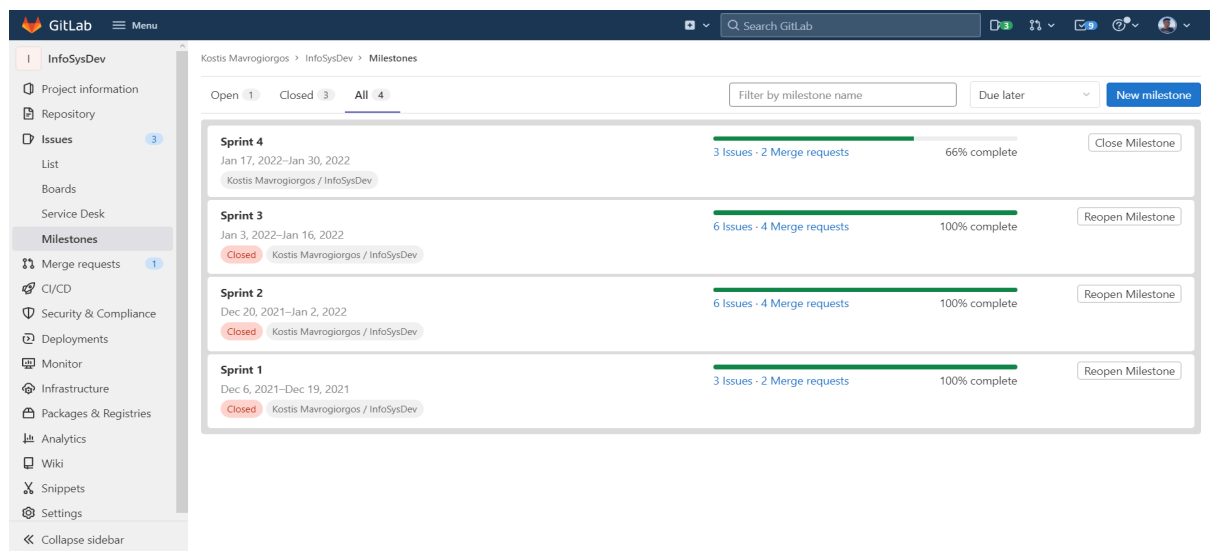
3. front-end: αφορά issues που σχετίζονται με το front-end κομμάτι της υλοποίησης (συνεπώς υπάρχει και το label development)
4. back-end: αφορά issues που σχετίζονται με το back-end κομμάτι της υλοποίησης (συνεπώς υπάρχει και το label development)

Στη παρακάτω εικόνα, φαίνονται ενδεικτικά κάποια από τα issues που δημιουργήθηκαν και υλοποιήθηκαν στα πλαίσια της εργασίας.



Εικόνα 1 : infosysdev Project Issues

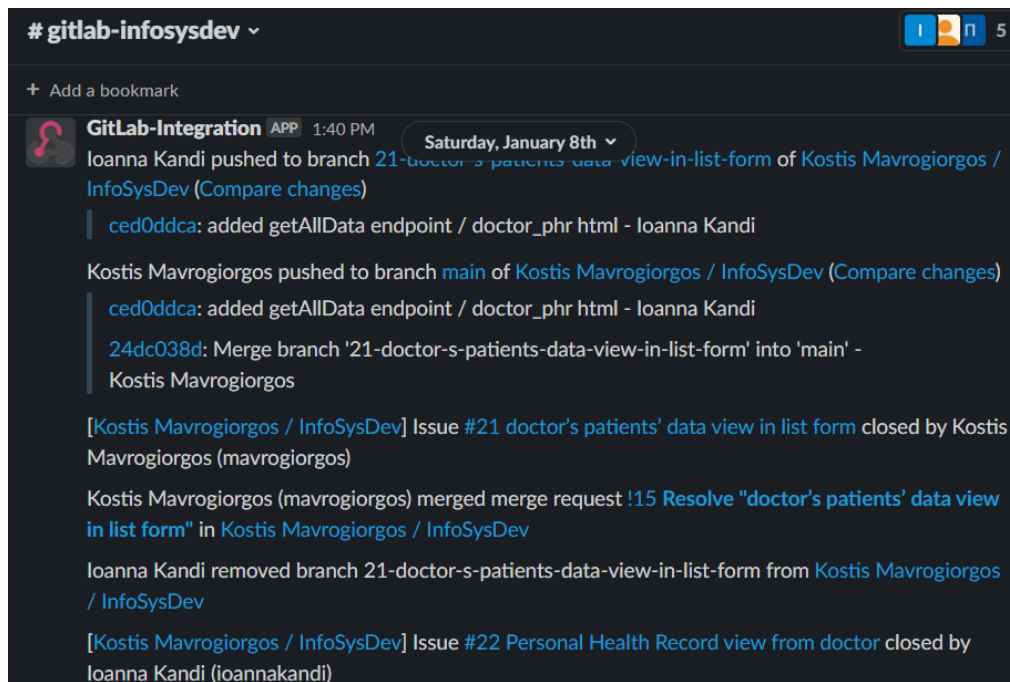
Αναφορικά με τον χρονοπρογραμματισμό των προαναφερθέντων issues, αυτός επιτεύχθηκε με τη δημιουργία των milestones. Κάθε milestone έχει διάρκεια δύο (2) εβδομάδων και περιλαμβάνει συγκεκριμένα issues. Κατά τη διάρκεια ενός milestone αναμένεται να υλοποιηθούν όλα τα issues που ανήκουν σε αυτό. Στη συγκεκριμένη εργασία, δημιουργήθηκαν τέσσερα (4) milestones, όπως παρουσιάζεται στη παρακάτω εικόνα.



Εικόνα 2 : infosysdev Project Milestones

3.2. Slack

Όσον αφορά την επικοινωνία μεταξύ των προγραμματιστών, αξιοποιήθηκαν τα εργαλεία Slack και Skype. Μέσω του Skype πραγματοποιήθηκαν οι εβδομαδιαίες κλήσεις μεταξύ των μελών της εργασίας, όπου συζητήθηκαν θέματα που αφορούσαν τη πρόοδο της υλοποίησης. Το Slack αποτελούσε ένα εργαλείο μέσω του οποίου υπήρχε η δυνατότητα ανταλλαγής μηνυμάτων μεταξύ των μελών αλλά και άμεσης ενημέρωσης σχετικά με τα commits που πραγματοποιούνταν καθώς αξιοποιήθηκε η δυνατότητα σύνδεσης του Slack με το Gitlab. Ειδικότερα, δημιουργήθηκε το κανάλι, το οποίο φαίνεται παρακάτω, με όνομα gitlab-infosysdev στο οποίο λαμβάνονταν αυτόματα ειδοποιήσεις που αφορούσαν οποιαδήποτε εξέλιξη σχετική με το gitlab (νέο commit, νέο merge request κτλ.).



Εικόνα 3 : infosysdev Gitlab & Slack Integration

3.3. Χρησιμοποιούμενες Τεχνολογίες - Γλώσσες Προγραμματισμού

Αναφορικά με τις τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση της πλατφόρμας, αυτές είναι οι ακόλουθες:

- **HTML5** για την οργάνωση των επιμέρους τμημάτων από τα οποία αποτελείται η διεπαφή χρήστη.
- **CSS3** για την μορφοποίηση των παραπάνω τμημάτων ούτως ώστε να είναι ευδιάκριτα και ελκυστικά προς τον χρήστη.
- **JavaScript** για την προσθήκη του στοιχείου της δυναμικότητας στην διεπαφή του χρήστη.
- **Bootstrap 4** για την μορφοποίηση τμημάτων της διεπαφής ούτως ώστε να είναι ευδιάκριτα και ελκυστικά προς τον χρήστη. Το Bootstrap είναι ένα framework το οποίο αποτελείται από HTML, CSS και JavaScript. Ουσιαστικά, παρέχει στους

προγραμματιστές έτοιμες κλάσεις τις οποίες μπορούν να χρησιμοποιήσουν έτσι ώστε να υλοποιήσουν γρήγορα μία πολύ ελκυστική διεπαφή χρήστη.

- **jQuery**, η οποία αποτελεί μία βιβλιοθήκη JavaScript και, ουσιαστικά, παίρνει πολλές κοινές εργασίες που απαιτούν πολλές γραμμές κώδικα JavaScript για να ολοκληρωθούν και τις «τυλίγει» σε μεθόδους που μπορούν να χρησιμοποιηθούν με μία γραμμή κώδικα. Μέσω της jQuery γίνονται και οι κλήσεις Ajax στα διάφορα endpoints που έχουν υλοποιηθεί στο back-end με χρήση της Python και της βιβλιοθήκης Flask.
- **Python** για την υλοποίηση των μηχανισμών της πλατφόρμας (back-end) και της αλληλεπίδρασης με τη MongoDB που αποτελεί τη βάση δεδομένων που αξιοποιείται από την πλατφόρμα.

3.4. Δημιουργία φακέλου infosysdev - Χρήση περιβάλλοντος git για ανάρτηση κώδικα στο Gitlab

Το κάθε μέλος της ομάδας δημιούργησε έναν φάκελο στην επιφάνεια εργασίας του, όπου σε αυτόν πάτησε δεξί κλικ, επέλεξε το “git bash here” και έτσι εμφανίστηκε το περιβάλλον του git. Στο git σαν πρώτη εντολή πληκτρολόγησε git clone (https address), δηλαδή την https Address που είχε αντιγράψει από το Project “Infosysdev” στο gitlab (όπου Infosysdev η ονομασία του μαθήματος εν συντομία) και έτσι προέκυπτε ένας νέος φάκελος με την ονομασία “infosysdev”, ο οποίος αρχικά περιείχε μόνο ένα αρχείο README.

Το μέλος το οποίο είχε αναλάβει τον συντονισμό της ομάδας (maintainer), τοποθέτησε σε αυτόν φάκελο ένα κενό αρχείο κώδικα Python με την ονομασία main.py και μετά πληκτρολόγησε στο git τις εξής εντολές: “git add main.py” , git commit -m “add main code” και “git push origin main”. Ο ίδιος, δημιούργησε νέα issues στο gitlab, οι ονομασίες

των οποίων ήταν σύμφωνα με τις λειτουργίες που είχαν επιλεγεί για την πλατφόρμα μας. Ο συντονιστής, ανέθεσε στο gitlab σε κάθε μέλος ξεχωριστά τα issues που είχαν συμφωνηθεί από κοινού να αναλάβουν τα μέλη, όπου τα issues αφορούσαν τις εξής κατηγορίες: documentation, back-end και front-end. Επιπροσθέτως, όρισε χρονοδιαγράμματα (sprints) μέσα στα οποία θα έπρεπε να ολοκληρωθούν.

Με αυτόν τον τρόπο, κάθε μέλος μπορούσε να προγραμματίσει την ολοκλήρωση των λειτουργιών (issues) που του είχαν ανατεθεί. Μόλις επιθυμούσε να ξεκινήσει να ασχολείται με ένα issue, έπρεπε πρώτα από όλα να δημιουργήσει ένα merge request για το issue που ήθελε να ασχοληθεί.

Μετά τη δημιουργία του merge request, ακολουθούσε την εξής διαδικασία: πρώτα αντέγραφε το branch name του issue, μετά πήγαινε στον φάκελο “infosysdev” όπου πρόσθετε το αρχείο με τον κώδικα που έγραφε (είτε Html είτε python) και άνοιγε το περιβάλλον του git. Κάθε φορά σαν πρώτη εντολή, πληκτρολογούσε git pull origin main ώστε να προστεθούν τοπικά όλες οι νεότερες εκδόσεις κώδικα.

Μετά για να ανεβάσει κώδικα για το Branch που τον ενδιέφερε, πληκτρολογούσε μια προς μια τις εξής εντολές:

- 1) git branch (name) ,
- 2) git checkout (name), όπου “name” το όνομα το branch που αντέγραφε,
- 3) git add app.py ή git add app.html όπου app είναι το όνομα του αρχείου που αποθήκευσε τον κώδικα και είχε προηγουμένως τοποθετήσει στον φάκελο infosysdev, όπου οι καταλήξεις αφορούν είτε γλώσσα Python είτε γλώσσα Html,
- 4) git commit -m”” όπου στα εισαγωγικά έγραφε τις αλλαγές που πραγματοποιούσε για εκείνο το προς ανάρτηση κομμάτι κώδικα,
- 5) git push origin (branch name), για να ανέβει στο gitlab.

Μόλις πληκτρολογούσε την τελευταία εντολή (git push), ερχόταν αυτόματα ειδοποίηση σε όλα τα μέλη της ομάδας στο Slack στην ενότητα “infosysdev” (που δημιουργήθηκε στο Slack για τις ανάγκες του μαθήματος) ώστε να είναι όλοι ενήμεροι ότι κάποιος ανέβασε ένα νέο ή τροποποιημένο κομμάτι κώδικα. Μόλις εκείνος που το ανέβασε στο

Gitlab ήταν σίγουρος ότι δεν πρόκειται να κάνει άλλη αλλαγή, έκανε κλικ στο check box με την επιλογή “Mark as ready” και ερχόταν ειδοποίηση στον maintainer ότι ο κώδικας που έχει ανέβει σε εκείνο το merge request είναι έτοιμος προς έλεγχο για να γίνει merge στην main.

Σε αυτό το σημείο, ο maintainer έκανε έλεγχο για τυχόν conflicts. Αν όντως υπήρχαν, τα διόρθωνε επιτόπου και μετά έκανε merge κάθε τμήμα του κώδικα στο κύριο κομμάτι (main.py).

Αν δεν υπήρχαν, τότε έκανε απευθείας χωρίς κάποια διόρθωση, merge κάθε τμήματος του κώδικα στο κύριο κομμάτι (main.py).

Προβήκαμε σε αυτόν τον τρόπο δοκιμών, τμήμα-τμήμα κώδικα, καθώς αν πηγαίναμε να κάνουμε απευθείας το τελικό merge όλου του κώδικα, θα ήταν δύσκολο να εντοπίσουμε τυχόν σφάλματα.

Εκτός από το παραπάνω, ο συντονιστής της ομάδας δημιούργησε ένα νέο φάκελο “templates”, ο οποίος περιελάμβανε κάποια έτοιμα προσυμπληρωμένα πρότυπα (templates), που μετονομάστηκαν από τα μέλη της ομάδας, σύμφωνα με τις λειτουργίες της πλατφόρμας. Επιπλέον, βασιστήκαμε στο κώδικα του κάθε προσυμπληρωμένου προτύπου και προβήκαμε στις αντίστοιχες αλλαγές σύμφωνα με τις λειτουργίες που θέλαμε να εμφανίζονται στο κάθε template.

3.5. Δοκιμές - Έλεγχος κώδικα

Στα πλαίσια της εργασίας μας, αξίζει να αναφερθεί ότι πραγματοποιούνταν τακτικές δοκιμές σε κάθε κομμάτι κώδικα. Αναλυτικότερα, αφού ολοκληρωνόταν το κάθε κομμάτι κώδικα που αφορούσε συγκεκριμένο endpoint, ο developer που το είχε αναλάβει το έτρεχε πρώτα στο προγραμματιστικό περιβάλλον της Python και αν δεν υπήρχαν σφάλματα, το έτρεχε στη συνέχεια στο postman παράλληλα με την Python ώστε να εμφανίζονται Responses. Ο developer για να δει τα αποτελέσματα (responses) θα έπρεπε να εκτελέσει ένα HTTP GET request στο postman και να πληκτρολογήσει το url, το οποίο μπορούσε να το βρει στην κονσόλα της Python μόλις έτρεχε τον κώδικα. Επιπλέον, στο τέλος του url ο προγραμματιστής προσέθετε “/” και το endpoint που ήθελε να δοκιμάσει και μετά έπρεπε να πατήσει “Send”.

Αμέσως μετά, περιηγούταν στο αντίστοιχο πρότυπο και πέρναγε πραγματικές τιμές στα πεδία key και values, όπου key οι μεταβλητές και values οι τιμές των μεταβλητών. Μόλις κλίκαρα την επιλογή “Send”, του εμφανιζόταν ένα μήνυμα ότι όλα είναι εντάξει ή ότι κάτι πήγε λάθος και πρέπει να ξαναπροσπαθήσει. Στην περίπτωση που δεν υπήρχε κάποιο σφάλμα, ανέβαζε το συγκεκριμένο κομμάτι κώδικα στο branch για το αντίστοιχο issue χρησιμοποιώντας το περιβάλλον git, σύμφωνα με τον τρόπο που αναφέρθηκε στην παράγραφο 3.4.

4. Στατικές και συμπεριφορικές όψεις του συστήματος με χρήση UML

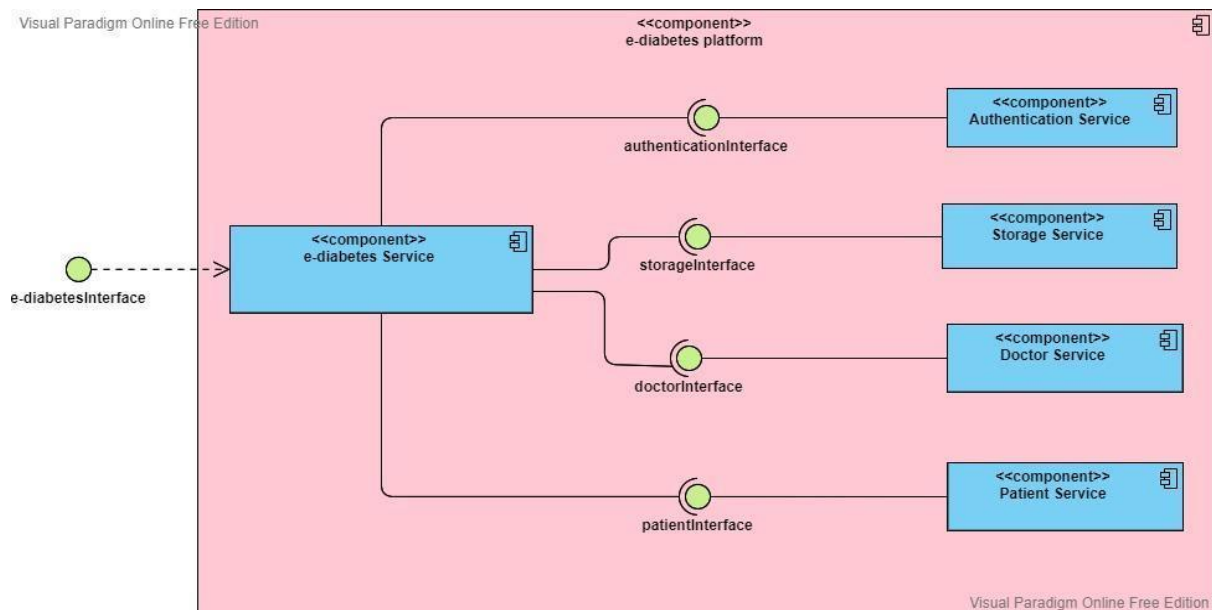
4.1. Η UML συνοπτικά

Η UML (Unified Modeling Language) αποτελεί μια γλώσσα γενικού σκοπού για μοντελοποίηση στα πλαίσια της μηχανικής λογισμικού. Βασικό χαρακτηριστικό της είναι το ότι μπορεί να γίνει εύκολα κατανοητή και από άτομα μη εξοικειωμένα με τον προγραμματισμό ή την πληροφορική γενικότερα. Η UML χρησιμοποιείται για τη δημιουργία ενός προτύπου που αποσκοπεί στην απεικόνιση ενός συστήματος και των εκάστοτε λειτουργιών του, σε όλες τις φάσεις, από το σχεδιασμό μέχρι την εγκατάσταση και λειτουργία του.

Οι λειτουργικές φάσεις ενός συστήματος αναπαρίστανται μέσω πληθώρας διαγραμμάτων που εξυπηρετούν διαφορετικές ανάγκες μοντελοποίησης, τα οποία η UML τα χωρίζει σε δύο (2) ευρύτερες κατηγορίες: τα στατικά (structural) και τα δυναμικά (behavioral). Οι προαναφερθείσες κατηγορίες, με τη σειρά τους, απαρτίζονται από διαφορετικού τύπου διαγράμματα. Ειδικότερα, στην εργασία αυτή χρησιμοποιήθηκαν

τρία (3) είδη διαγραμμάτων: το component diagram, το use case diagram και το activity diagram, τα οποία αναπαρίστανται και αναλύονται παρακάτω [21].

4.2. Component Diagram



Εικόνα 4: e-diabetes component diagram

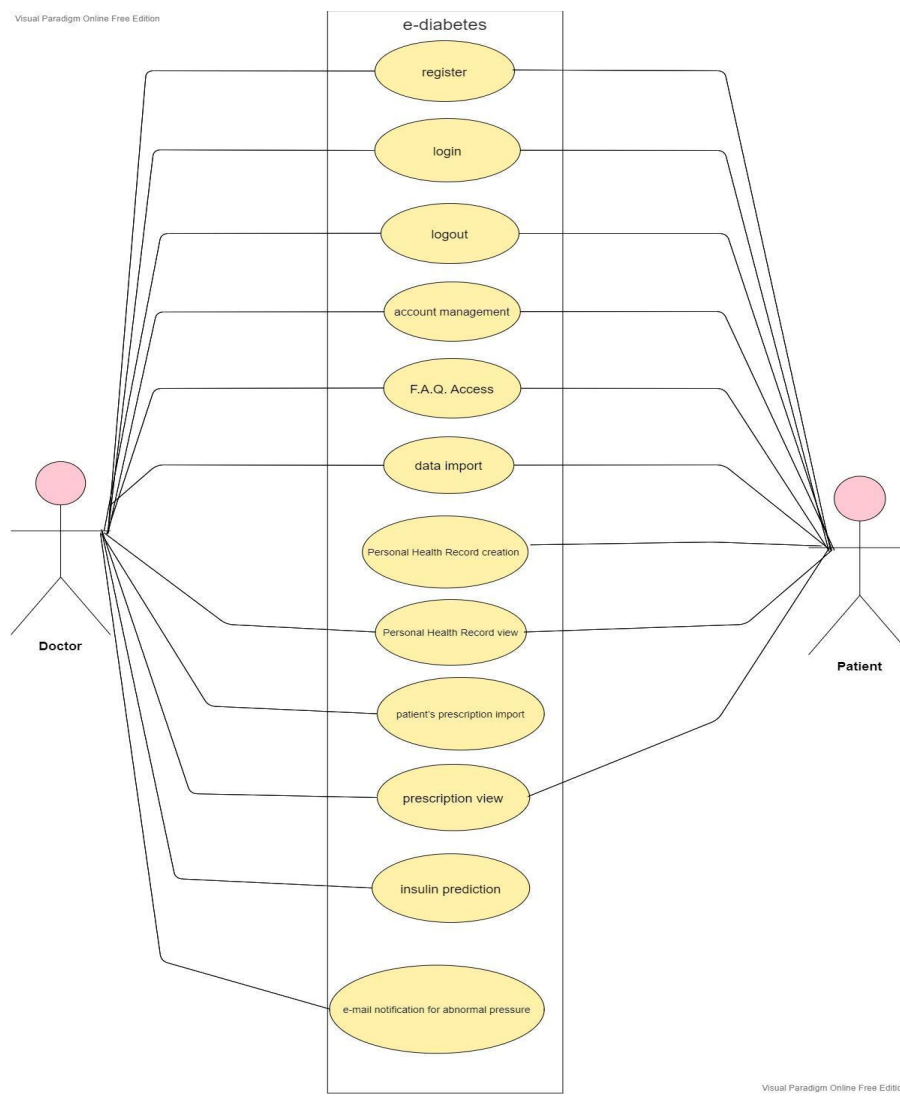
Το Component Diagram ανήκει στην ευρύτερη κατηγορία των Structural Diagrams της UML, τα οποία επικεντρώνονται σε θέματα δομής του συστήματος χρησιμοποιώντας αντικείμενα, παραμέτρους και σχέσεις. Το Component Diagram, ειδικότερα, απεικονίζει τα φυσικά στοιχεία ενός προγράμματος (π.χ. πακέτα, βιβλιοθήκες, αρχεία κ.ο.κ.), καθώς και τον τρόπο σύνδεσής τους, τα οποία απαρτίζουν το εκάστοτε σύστημα. Είναι σημαντικό να τονιστεί πως σε αυτό το διάγραμμα δεν επιδεικνύονται οι λειτουργίες του συστήματος αυτές καθ' αυτές, αλλά δίνεται έμφαση στα στοιχεία -φυσικά και λογικά- τα οποία είναι αναγκαία για την υλοποίησή του [21].

Τα στοιχεία/components από τα οποία απαρτίζεται η πλατφόρμα e-diabetes της παρούσας εργασίας είναι τα εξής:

- Authentication Service, το οποίο αφορά την διαχείριση δεδομένων εξουσιοδότησης για την εγγραφή και είσοδο των χρηστών στην πλατφόρμα. Για την επικοινωνία με το Authentication Service, αξιοποιείται το Authentication Interface.
- Storage Service, το οποίο πραγματεύεται την διαχείριση και αποθήκευση των δεδομένων των χρηστών. Για την επικοινωνία με το Storage Service, αξιοποιείται το Storage Interface.
- Doctor Service, το οποίο πραγματεύεται τη διαχείριση των δεδομένων των γιατρών. Για την επικοινωνία με το Doctor Service, αξιοποιείται το Doctor Interface.
- Patient Service, το οποίο πραγματεύεται τη διαχείριση των δεδομένων των ασθενών. Για την επικοινωνία με το Patient Service, αξιοποιείται το Patient Interface.
- E-diabetes Service, το οποίο επικοινωνεί με τα προαναφερθέντα components μέσω των αντίστοιχων interfaces.

Τέλος, για την επικοινωνία με την πλατφόρμα e-diabetes είναι απαραίτητη η χρήση του E-diabetes Interface.

4.3. Use Case Diagram



Εικόνα 5: e-diabetes use case diagram

Το Use Case Diagram υπάγεται στην κατηγορία των Behavioral Diagrams της UML, τα οποία επικεντρώνονται στη δυναμική συμπεριφορά του εκάστοτε συστήματος, καθώς και στο πώς τα διάφορα τμήματά του αλληλεπιδρούν μεταξύ τους. Αναλυτικότερα, το Use Case Diagram απεικονίζει τις λειτουργίες του συστήματος, τους χρήστες του και πώς αυτοί συνδέονται με ποιες λειτουργίες [21].

Η σύνδεση μεταξύ χρηστών και λειτουργιών της πλατφόρμας e-diabetes, απεικονίζεται στο ανωτέρω διάγραμμα της εικόνας 5. Το διάγραμμα περιγράφει, αρχικά, πως οι χρήστες είναι δύο (2), ο γιατρός (doctor) και ο ασθενής (patient).

Στη συνέχεια, φαίνεται πως και οι δύο (2) χρήστες έχουν πρόσβαση στις περισσότερες λειτουργίες της πλατφόρμας, οι οποίες είναι οι εξής:

- Εγγραφή στην πλατφόρμα e-diabetes (register)
- Είσοδος στην πλατφόρμα e-diabetes (login)
- Έξοδος από την πλατφόρμα e-diabetes (logout)
- Διαχείριση λογαριασμού (account management)
- Πρόσβαση σε Frequently Asked Questions για την καλύτερη κατάρτιση του χρήστη σε θέματα σχετικά με τον διαβήτη (F.A.Q. Access)
- Εισαγωγή δεδομένων στην πλατφόρμα e-diabetes (data import)
- Δημιουργία ή εμφάνιση του Προσωπικού Φακέλου του εκάστοτε ασθενούς (Personal Health Record View)

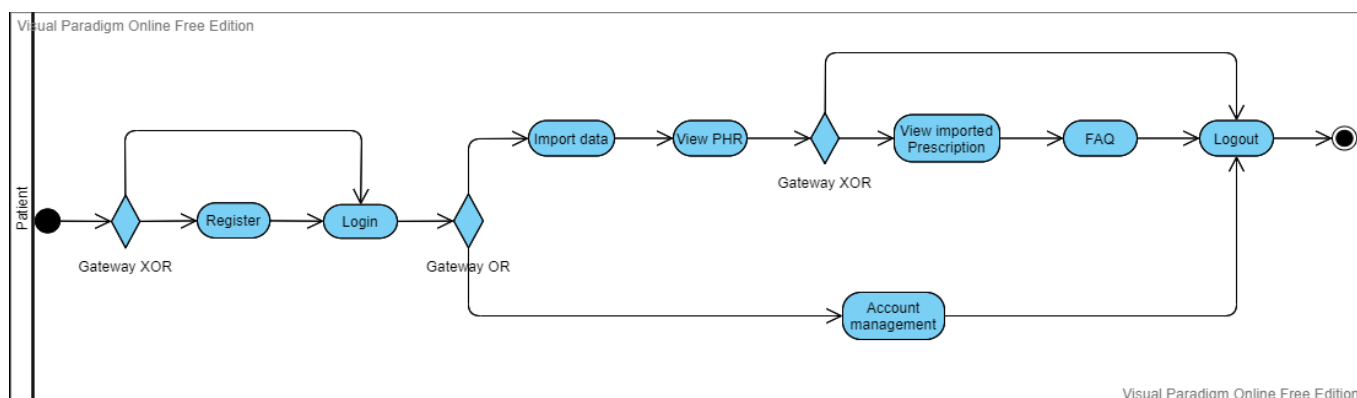
Εκτός από τις παραπάνω κοινές λειτουργίες, η πρόσβαση σε περαιτέρω λειτουργικότητες του e-diabetes εξαρτάται από την ιδιότητα του κάθε χρήστη. Δηλαδή, μόνο οι γιατροί έχουν τη δυνατότητα να παρέχουν συνταγές στους ασθενείς τους (patient's prescription import), να αξιοποιήσουν τον αλγόριθμο πρόβλεψης για τα επίπεδα ινσουλίνης κάποιου ασθενή τους (insulin prediction) και να λάβουν ειδοποιήσεις μέσω ηλεκτρονικού ταχυδρομείου για μη φυσιολογικές μετρήσεις κάποιου ασθενή τους (e-mail prediction for abnormal pressure). Τέλος, μόνο οι ασθενείς έχουν τη δυνατότητα να δημιουργήσουν τον δικό τους Προσωπικό Φάκελο Υγείας (personal health record creation) και να προβάλλουν την συνταγή που τους χορήγησε ο γιατρός τους.

4.4. Activity Diagram

Τα διαγράμματα αυτά απεικονίζουν την ακολουθία δραστηριοτήτων στο σύστημα. Επιπλέον, ένα διάγραμμα δραστηριοτήτων μπορεί να δείχνει τη ροή εργασιών από ένα σημείο έναρξης μέχρι ένα σημείο πέρατος για ένα λειτουργό, καταγράφοντας όλες τις λεπτομέρειες για τις αποφάσεις που μπορεί να υπάρχουν στην δραστηριότητα.

Αναλυτικότερα, στα πλαίσια της δικής μας εργασίας σχεδιάστηκαν δύο τέτοια διαγράμματα ένα σχετικά με την ενδεικτική ροή συμπλήρωσης δεδομένων υγείας ασθενή ηλεκτρονικά και - ένα σχετικά με την προβολή δεδομένων υγείας ασθενή από τον γιατρό ηλεκτρονικά [19].

4.4.1. Ενδεικτικό διάγραμμα δραστηριοτήτων (activity diagram) σχετικά με την συμπλήρωση δεδομένων υγείας ασθενή ηλεκτρονικά



Εικόνα 6: e-diabetes activity diagram (patient)

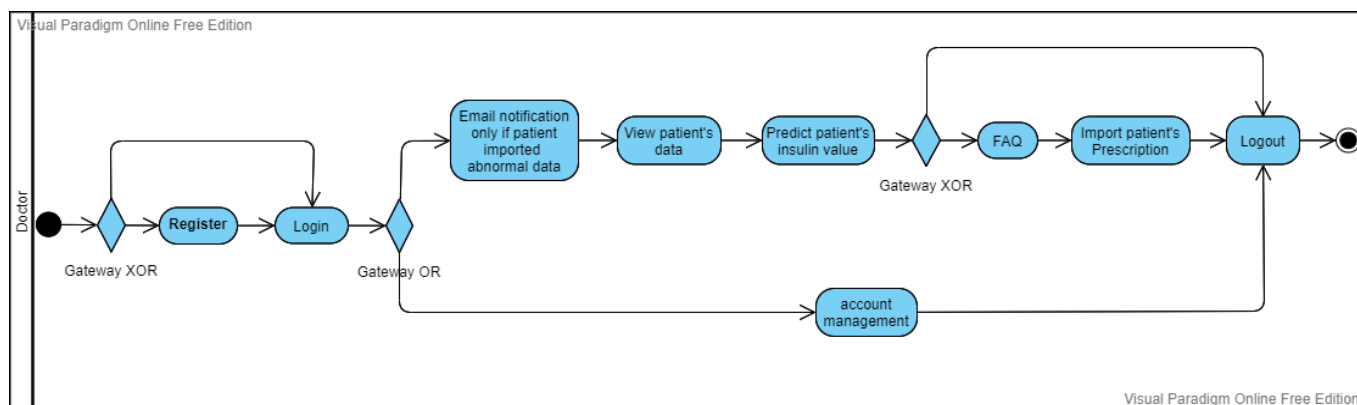
Ακολουθεί μία ενδεικτική ροή δραστηριοτήτων σχετικά με την εισαγωγή δεδομένων υγείας ηλεκτρονικά από τον ασθενή:

Ο ασθενής στην περίπτωση που δεν έχει λογαριασμό στην πλατφόρμα (e- diabetes), θα προβεί σε εγγραφή (registration) για να εισέλθει (login) σε αυτή. Στην περίπτωση που έχει ήδη προβεί σε εγγραφή του στο σύστημα, αρκεί να κάνει είσοδο (login) με τα στοιχεία του (username και password). Στη συνέχεια, του δίνεται η δυνατότητα είτε να

διαχειριστεί τον λογαριασμό του (account management) ώστε να αλλάξει κάποια στοιχεία του πχ το username του είτε να προβεί στην εισαγωγή των δεδομένων υγείας. Αν επέλεξε να διαχειριστεί τον λογαριασμό του, τότε μετά προβαίνει στην έξοδο από την πλατφόρμα κάνοντας αποσύνδεση (logout).

Αν επέλεξε να προβεί στην εισαγωγή των δεδομένων υγείας, τότε δύναται να προβάλλει τον Προσωπικό του Ιατρικό φάκελο υγείας (View PHR) που περιλαμβάνει όλες τις εισαχθείσες τιμές υγείας. Αμέσως μετά, είτε θα αποσυνδεθεί από την πλατφόρμα κάνοντας αποσύνδεση (logout) είτε θα προβάλλει τη συνταγή (View imported Prescription) που του επισύναψε ο γιατρός. Στην περίπτωση που επέλεξε να προβάλλει τη συνταγή του, τότε μετά μπορεί αν χρειάζεται κάποια ενημέρωση, να περιηγηθεί στις συχνές ερωτήσεις (FAQ) σχετικά με τον διαβήτη και τέλος προχωράει σε αποσύνδεση (logout) από την πλατφόρμα.

4.4.2. Ενδεικτικό διάγραμμα δραστηριοτήτων (activity diagram) σχετικά με την προβολή δεδομένων υγείας ασθενή από τον γιατρό ηλεκτρονικά



Εικόνα 7: e-diabetes activity diagram (doctor)

Ακολουθεί μία ενδεικτική ροή δραστηριοτήτων σχετικά με την προβολή δεδομένων υγείας ασθενή ηλεκτρονικά από τον γιατρό:

Ο γιατρός αν δεν έχει λογαριασμό στην πλατφόρμα (e- diabetes), θα προβεί σε εγγραφή (registration) για να εισέλθει (login) σε αυτή. Στην περίπτωση που έχει ήδη

προβεί σε εγγραφή του στο σύστημα, αρκεί να κάνει είσοδο (login) με τα στοιχεία του (username και password). Αξίζει να επισημανθεί, ότι αν ο ασθενής εισάγει κάποιο δεδομένο υγείας το οποίο είναι εκτός των φυσιολογικών ορίων, θα σταλεί ειδοποίηση μέσω e-mail στο γιατρό ενώ σε αντίθετη περίπτωση δεν θα σταλεί ειδοποίηση. Στη συνέχεια, του δίνεται η δυνατότητα είτε να διαχειριστεί τον λογαριασμό του (account management) ώστε να αλλάξει κάποια στοιχεία του πχ το username του είτε να προβεί στην προβολή των δεδομένων υγείας του ασθενή του.

Αν επέλεξε να διαχειριστεί τον λογαριασμό του, τότε μετά προβαίνει στην έξοδο από την πλατφόρμα κάνοντας αποσύνδεση (logout).

Αν επέλεξε να προβεί στην προβολή των δεδομένων υγείας του ασθενή του τότε ανάλογα με τις τιμές που έχει εισάγει ο ασθενής, δύναται - αν κρίνει ότι χρειάζεται - να προβεί σε πρόβλεψη της τιμής της ινσουλίνης (Predict patient's insulin value) που θα πρέπει να του χορηγήσει. Στην συνέχεια, είτε θα αποσυνδεθεί από την πλατφόρμα κάνοντας αποσύνδεση (logout) είτε θα περιηγηθεί στις συχνές ερωτήσεις (FAQ) για ενημέρωση σχετικά με τον διαβήτη. Στην περίπτωση που επέλεξε να περιηγηθεί στις συχνές ερωτήσεις (FAQ), τότε μετά θα επισυνάψει την συνταγή του ασθενή του (Import patient's Prescription) και τέλος θα προχωρήσει σε αποσύνδεση (logout) από την πλατφόρμα.

- Επίσης, αξίζει να επισημανθεί ότι εκτός από την εισαγωγή δεδομένων υγείας ηλεκτρονικά, η πλατφόρμα e-diabetes υποστηρίζει και την εισαγωγή δεδομένων του ασθενή από τον γιατρό κατά τη διάρκεια φυσικού (δια ζώσης) ραντεβού.

Στην περίπτωση αυτή το μόνο διαφορετικό με τα προαναφερθέντα, είναι ότι ο γιατρός μετά την σύνδεση του στην πλατφόρμα εισάγει τα δεδομένα υγείας του ασθενή του και ανάλογα με τα δεδομένα που εισήγαγε, στέλνεται ή όχι ειδοποίηση μέσω e-mail για μη φυσιολογική εισαγωγή δεδομένων. Μετά, συνεχίζει ο ενδεικτικός ρόλος του γιατρού ακριβώς όπως περιγράφηκε προηγουμένως μέχρι την αποσύνδεση του από την πλατφόρμα.

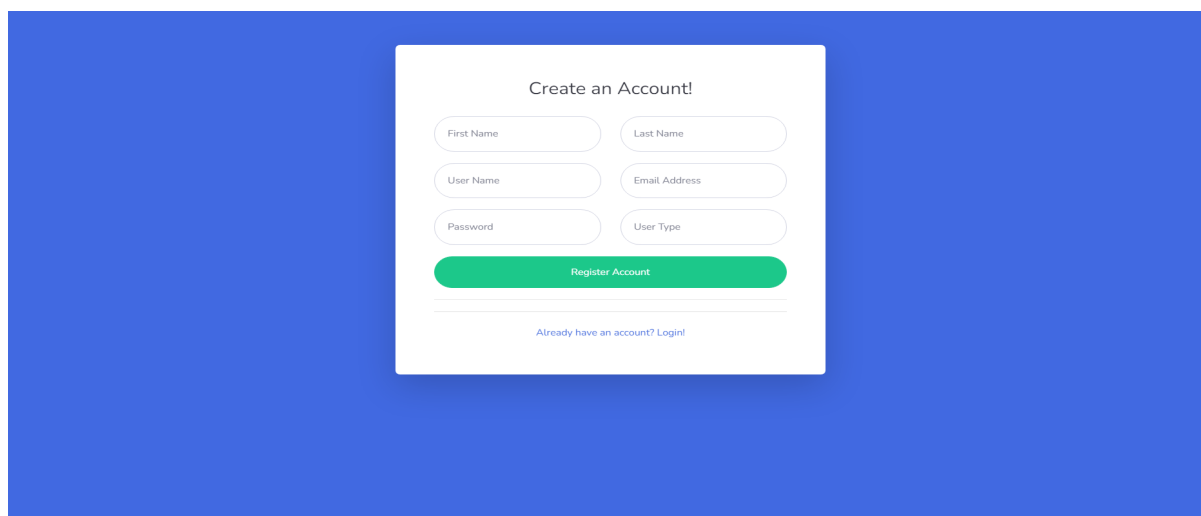
Σχετικά με τον ασθενή, μόλις ο γιατρός εισάγει τα δεδομένα, τότε ο ασθενής μπορεί να συνδεθεί στην πλατφόρμα και να προβάλλει τον προσωπικό του Ιατρικό φάκελο

υγείας ο οποίος περιλαμβάνει όλες τις εισαχθείσες τιμές υγείας. Μετά, συνεχίζει ο ενδεικτικός ρόλος του ασθενή ακριβώς όπως περιγράφηκε προηγουμένως μέχρι την αποσύνδεση του από την πλατφόρμα.

5. Εγχειρίδιο Χρήσης (User Manual)

5.1. Σύνδεση και Εγγραφή Χρηστών

5.1.1. Εγγραφή χρήστη στη πλατφόρμα

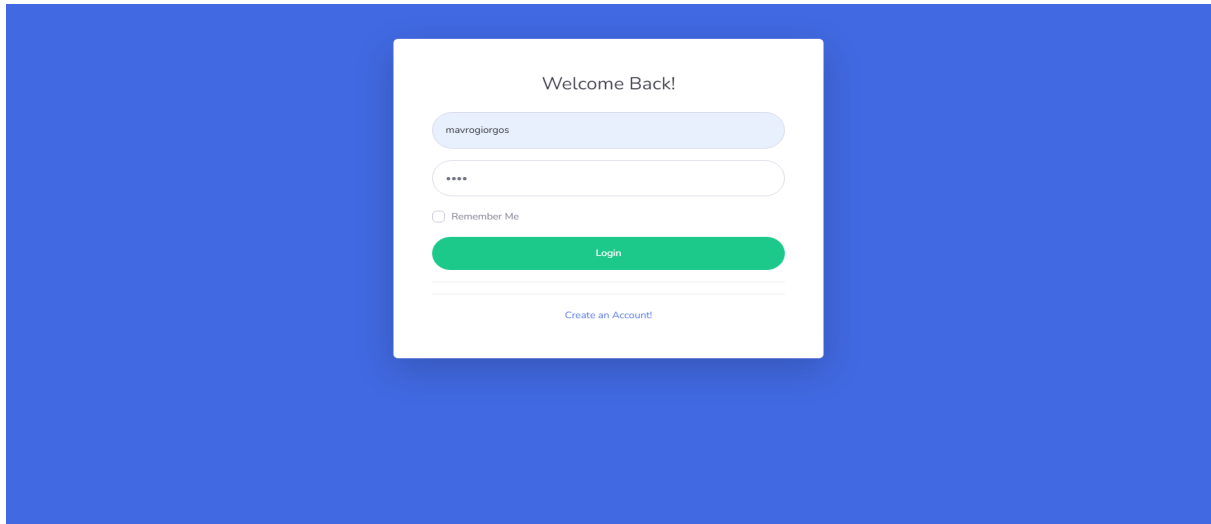


Εικόνα 8: e-diabetes registration form

Ο χρήστης προκειμένου να εγγραφεί στην πλατφόρμα καλείται να εισάγει τα εξής στοιχεία του: όνομα (First Name), επώνυμο (Last Name), το username της επιλογής του, email, κωδικό (password) του και το user type¹ του (doctor or patient). Μόλις συμπληρωθούν τα παραπάνω στοιχεία, ολοκληρώνεται η εγγραφή του στο σύστημα πατώντας το κουμπί Register Account.

¹ Το user type είναι απαραίτητο, διότι ανάλογα με τον τύπο χρήστη παρέχονται διαφορετικές υπηρεσίες στην πλατφόρμα.

5.1.2. Σύνδεση χρήστη στη πλατφόρμα



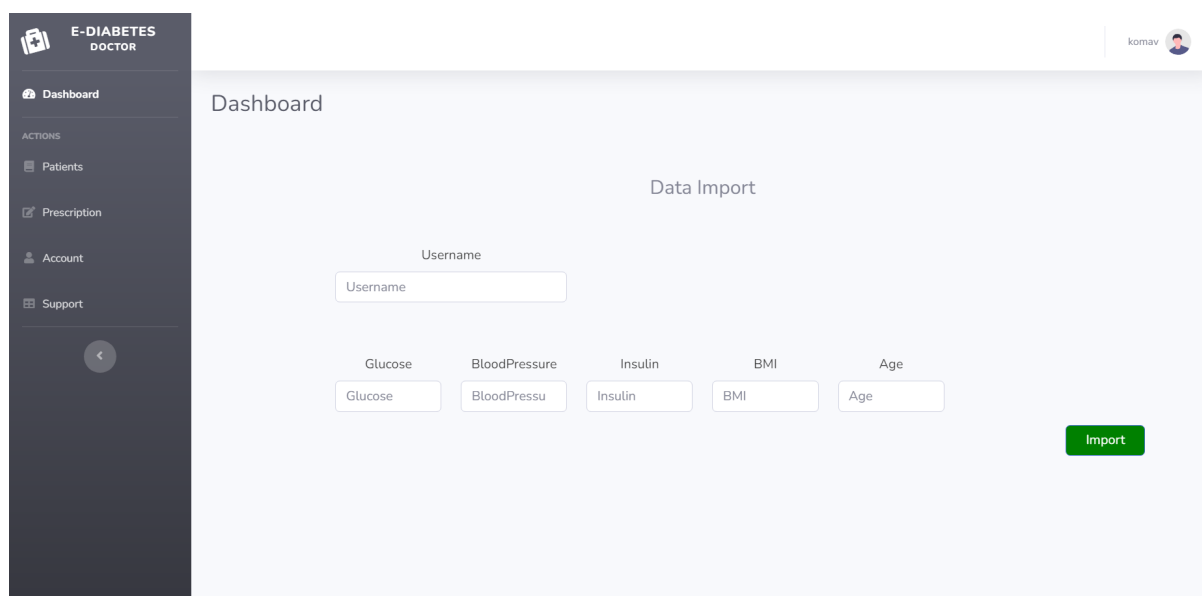
Εικόνα 9: e-diabetes login form

Ο χρήστης εφόσον έχει εγγραφεί, εισέρχεται στο σύστημα εισάγοντας το username και το password του. Στη συνέχεια, πατώντας το κουμπί Login, ανακατευθύνεται στην αντίστοιχη διεπαφή που συνάδει με το user type που αναφέρθηκε στην παραπάνω υποενότητα.

5.2. Περιγραφή end-points γιατρού

5.2.1. Εισαγωγή μετρήσεων του ασθενή από τον γιατρό (patient's data import on doctor's end)

Η πλατφόρμα e-diabetes, μεταξύ άλλων, παρέχει στο γιατρό τη δυνατότητα εισαγωγής των δεδομένων των ασθενών του. Αναλυτικότερα, εισάγει το όνομα χρήστη (username) του ασθενή και τις ακόλουθες τιμές: (α) ηλικία (age), (β) δείκτης μάζας σώματος (BMI), (γ) γλυκόζη (glucose) και (δ) αρτηριακή πίεση (BloodPressure). Στη συνέχεια, θα καταχωρήσει την τιμή της ινσουλίνης που κρίνει ότι θα πρέπει να λάβει ο ασθενής². Με την επιτυχή καταχώρηση των δεδομένων εμφανίζεται το μήνυμα "Data imported successfully!", διαφορετικά, εμφανίζεται το μήνυμα «Oops! Something went wrong. Please try again.».



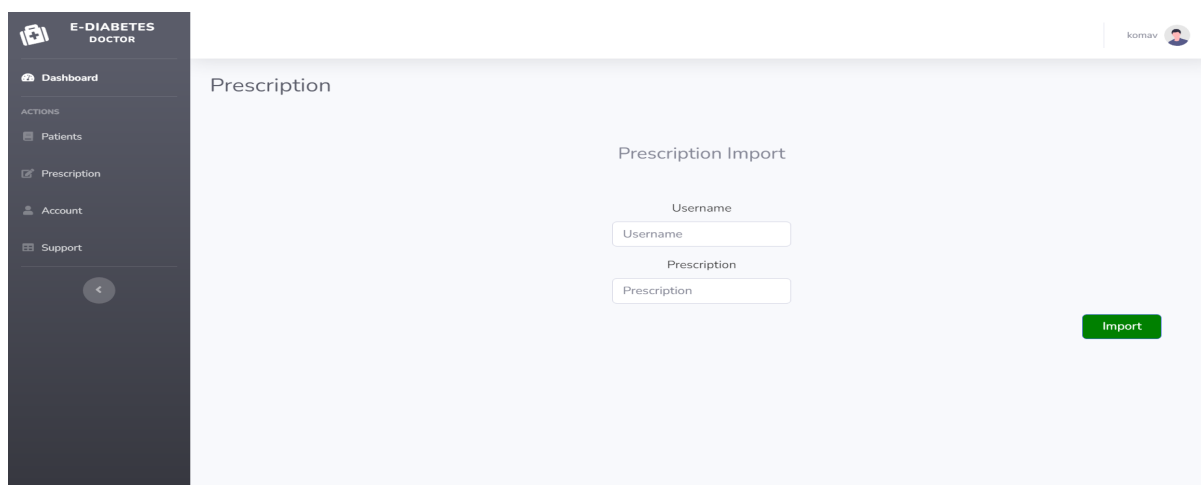
The screenshot displays the 'E-DIABETES DOCTOR' dashboard. On the left is a dark sidebar with navigation links: Dashboard, Patients, Prescription, Account, and Support. The main content area is titled 'Dashboard' and features a 'Data Import' section. This section includes a 'Username' input field, followed by five input fields for 'Glucose', 'BloodPressure', 'Insulin', 'BMI', and 'Age'. Each field has a label above it and a placeholder text below it. A green 'Import' button is located at the bottom right of the form.

Εικόνα 10: patient's data import on doctor's end

² Η καταχώρηση της τιμής της ινσουλίνης πραγματοποιείται μόνο σε ραντεβού διαζώσης.

5.2.2. Εισαγωγή θεραπείας / συνταγής του ασθενούς από τον γιατρό (patient's prescription import from doctor's end)

Η επόμενη λειτουργία αφορά την εισαγωγή θεραπείας/συνταγογράφησης. Όπως η εισαγωγή των μετρήσεων του ασθενή πραγματοποιείται από το γιατρό, έτσι και η συγκεκριμένη λειτουργία μπορεί εξίσου να πραγματοποιηθεί είτε εξ αποστάσεως είτε δια ζώσης. Εφόσον καταχωρηθούν οι νέες μετρήσεις του ασθενούς στο σύστημα είτε από το γιατρό είτε από τον ασθενή, ο γιατρός κρίνει αν χρήζει νέας φαρμακευτικής αγωγής ή όχι. Εφόσον κριθεί απαραίτητο ο ασθενής να ακολουθήσει μία διαφορετική αγωγή, ο γιατρός αναλαμβάνει τη σύνταξη μιας νέας συνταγής, την οποία αποθηκεύει στο σύστημα, αφού πρώτα καταχωρήσει το όνομα χρήστη (username) του ασθενή, προκειμένου να εμφανιστεί στο dashboard του αντίστοιχου ασθενή. Με το πέρας της διαδικασίας, απεικονίζεται το μήνυμα "Prescription imported successfully!", σε διαφορετική περίπτωση προβάλλεται το μήνυμα «Oops! Something went wrong. Please try again.».

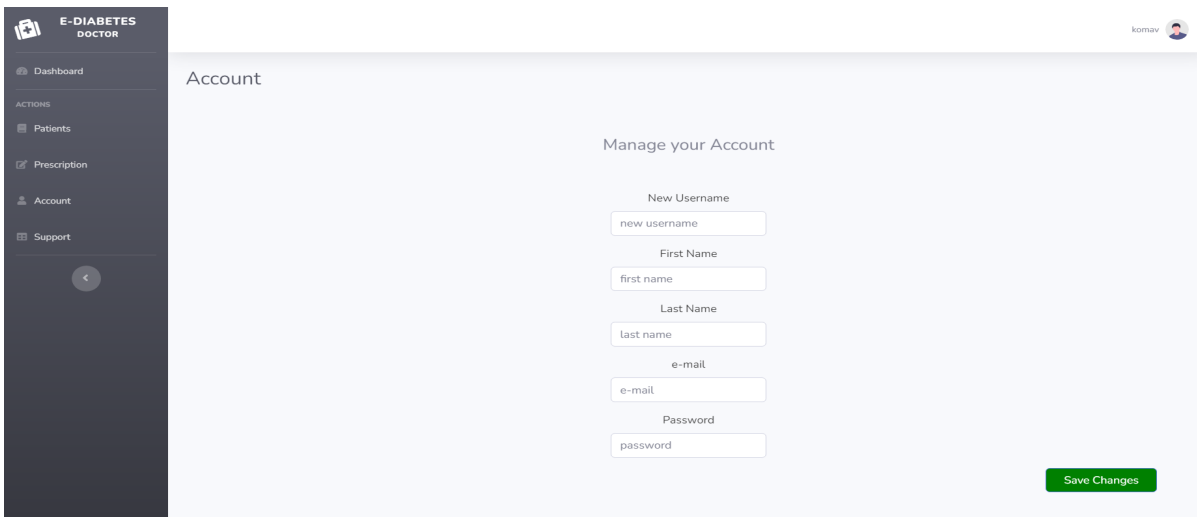


The screenshot displays the 'E-DIABETES DOCTOR' web application interface. On the left is a dark sidebar with a menu containing 'Dashboard', 'Patients', 'Prescription' (highlighted with a checkmark), 'Account', and 'Support'. The main content area is titled 'Prescription' and contains a 'Prescription Import' form. The form has two input fields: 'Username' and 'Prescription', both with placeholder text. A green 'Import' button is located at the bottom right of the form. In the top right corner of the application, there is a user profile icon and the name 'komav'.

Εικόνα 11: patient's prescription import from doctor's end

5.2.3. Διαχείριση λογαριασμού γιατρού (doctor account management)

Η τρίτη λειτουργία που παρέχεται στο γιατρό, είναι η τροποποίηση των στοιχείων του λογαριασμού του. Η συγκεκριμένη υπηρεσία προσπελάζεται μέσω της επιλογής Account που βρίσκεται στο sidebar menu της πλατφόρμας. Οι πληροφορίες που μπορεί να μεταβάλλει αντιστοιχούν στα εξής πεδία: (α) όνομα χρήστη (new username), (β) όνομα (First Name), (γ) επώνυμο (Last Name), (δ) email και (ε) κωδικός. Αυτό που δεν χρειάζεται να συμπληρώσει είναι το τρέχον όνομα χρήστη, εφόσον διατηρείται στο σύστημα μέσω των cookies, έως και την έξοδο του χρήστη από την πλατφόρμα. Εφόσον δεν προκύψει κάποιο πρόβλημα, το μήνυμα που εμφανίζεται στην οθόνη είναι «All set! Changes saved successfully.» ενώ σε άλλη περίπτωση προβάλλεται το μήνυμα «Oops! Something went wrong. Please try again.».



The screenshot shows the 'E-DIABETES DOCTOR' interface. On the left is a dark sidebar with a menu: Dashboard, Patients, Prescription, Account, and Support. The 'Account' section is active. The main area has a light blue header 'Account' and a title 'Manage your Account'. Below this are five input fields: 'New Username' (placeholder 'new username'), 'First Name' (placeholder 'first name'), 'Last Name' (placeholder 'last name'), 'e-mail' (placeholder 'e-mail'), and 'Password' (placeholder 'password'). A green 'Save Changes' button is located at the bottom right of the form area.

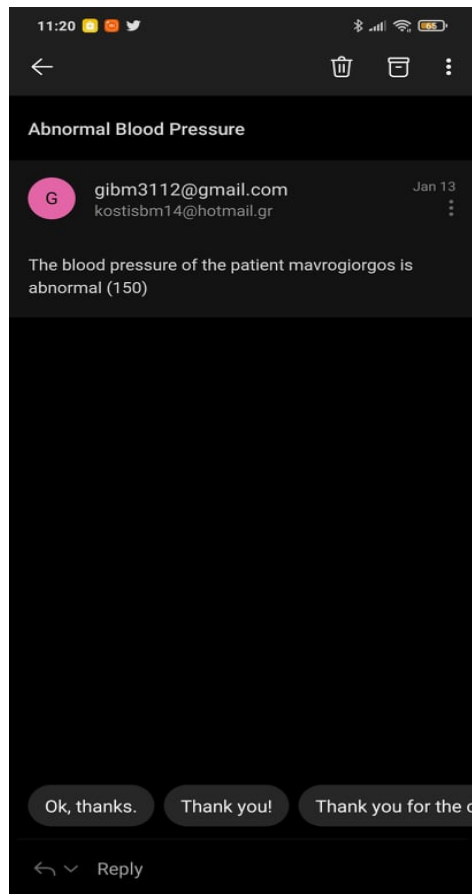
Εικόνα 12: doctor account management

5.2.4. Προβολή ασθενών και των τελευταίων μετρήσεων τους από την πλευρά του γιατρού σε μορφή πίνακα (doctor's patients' data view in list form)

Η επόμενη λειτουργία αφορά την προβολή των ασθενών και των τελευταίων μετρήσεων τους σε μορφή πίνακα. Πιο αναλυτικά, οι τιμές που παρουσιάζονται στον εν λόγω πίνακα είναι οι ακόλουθες: (α) το όνομα χρήστη του ασθενή (username), (β) η ηλικία του (age), (γ) ο δείκτης μάζας σώματος (BMI), (δ) η τιμή της γλυκόζης (glucose), (ε) η τιμή της αρτηριακής πίεσης (Blood Pressure), (στ) η τιμή της ινσουλίνης (Insulin) και (η) η ημερομηνία (date) εισαγωγής των μετρήσεων.

Στην ίδια σελίδα, ο γιατρός έχει τη δυνατότητα να προβλέψει αν ένας ασθενής χρειάζεται ινσουλίνη ή όχι, εισάγοντας τις μετρήσεις του, οι οποίες εμφανίζονται στον πίνακα. Η πρόβλεψη σχετικά με τη χορήγηση ινσουλίνης, πραγματοποιείται αξιοποιώντας έναν εκπαιδευμένο αλγόριθμο μηχανικής μάθησης (Naive Bayes) πάνω σε ένα σχετικό σύνολο δεδομένων³, το οποίο προέρχεται από το Kaggle (pima indians dataset). Αν ο ασθενής χρειάζεται ινσουλίνη, προβάλλεται το μήνυμα “Based on Naive Bayes ML algorithm, the patient needs insulin.” ενώ αν δεν χρειάζεται εμφανίζεται το μήνυμα “Based on Naive Bayes ML algorithm, the patient does not need insulin.”.

³ Οι τιμές που πραγματεύεται αφορούν τα πεδία: glucose, BloodPressure, Insulin, BMI, Age & Outcome (είναι το πεδίο προς πρόβλεψη και παίρνει τιμές 1 ή 0, ανάλογα με το αν ο ασθενής χρειάζεται ινσουλίνη ή όχι).

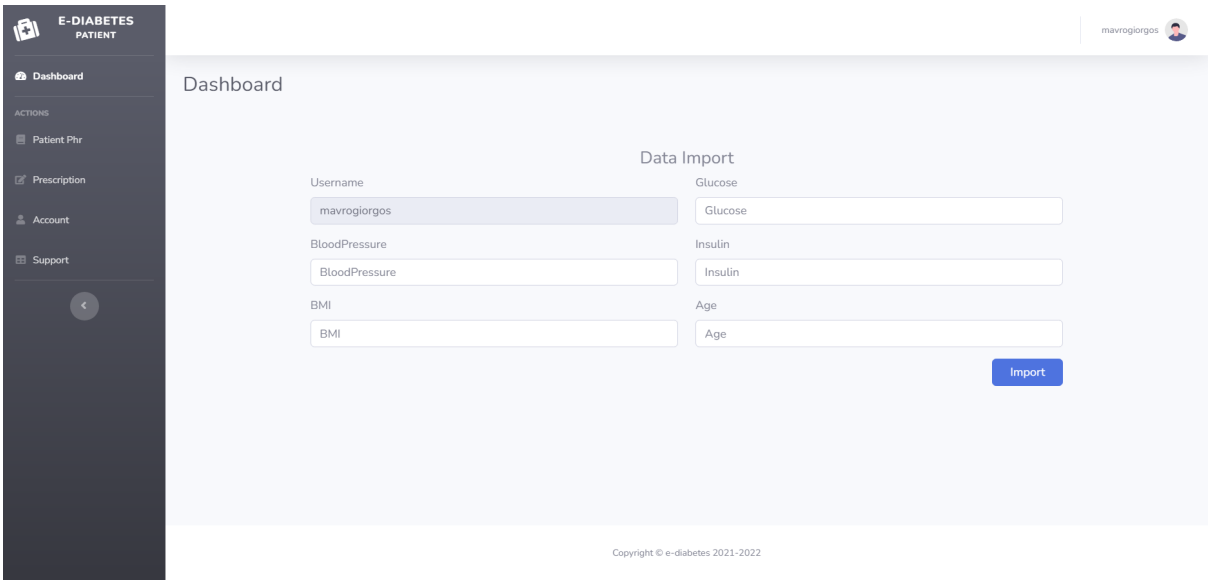


Εικόνα 14: doctor's e-mail notification for patient's abnormal data import

5.3. Περιγραφή end-points ασθενούς

5.3.1. Εισαγωγή μετρήσεων ασθενή (patient's data import)

Η πρώτη λειτουργία του e-diabetes patient αφορά την εισαγωγή των μετρήσεων από τον ίδιο τον ασθενή. Ο ασθενής έχει τη δυνατότητα να καταχωρήσει τις τιμές που μέτρησε για τη γλυκόζη (glucose), την αρτηριακή πίεση (BloodPressure) και την ινσουλίνη (insulin). Επίσης, μπορεί να καταχωρήσει τον δείκτη μάζας σώματος (BMI) και την ηλικία του (age). Εφόσον οι μετρήσεις αποθηκευτούν επιτυχώς εμφανίζεται στην οθόνη το μήνυμα “Successful data import”.



The screenshot displays the 'E-DIABETES PATIENT' dashboard. On the left is a dark sidebar with a 'Dashboard' button and a list of actions: 'Patient Phr', 'Prescription', 'Account', and 'Support'. The main area is titled 'Dashboard' and contains a 'Data Import' section. This section has six input fields arranged in two columns: 'Username' (pre-filled with 'mavrogiorgos'), 'Glucose', 'BloodPressure' (pre-filled with 'BloodPressure'), 'Insulin', 'BMI' (pre-filled with 'BMI'), and 'Age'. An 'Import' button is located at the bottom right of the form. The footer of the page reads 'Copyright © e-diabetes 2021-2022'.

Εικόνα 15: patient's data import

5.3.2. Διαχείριση λογαριασμού ασθενή (Patient account management)

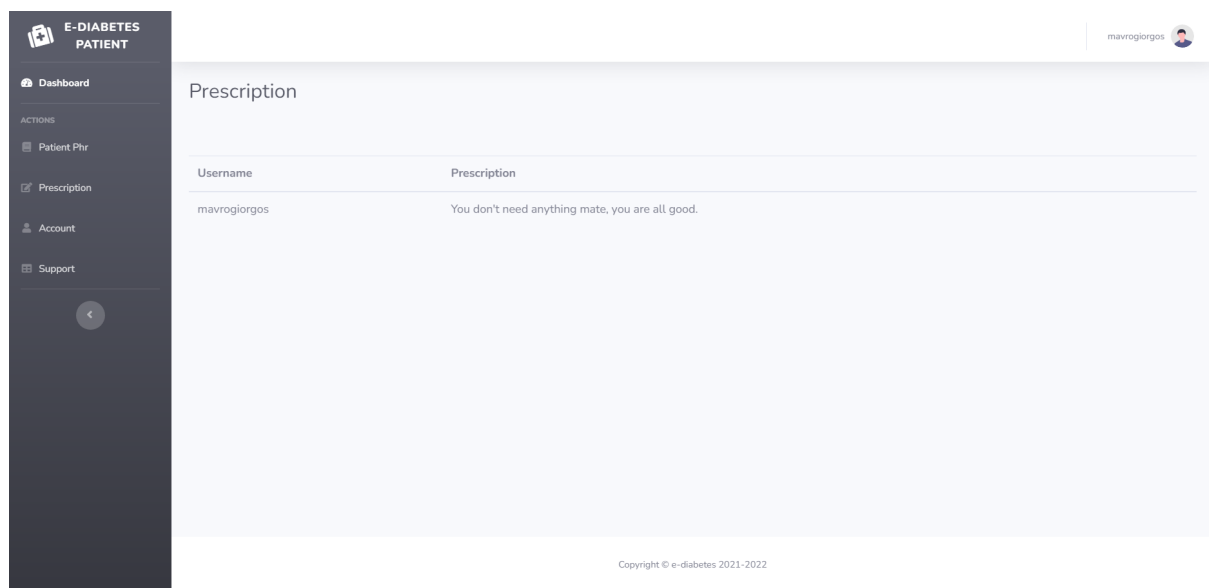
Η παρούσα διεργασία, επιτρέπει στον ασθενή να τροποποιήσει τα στοιχεία του. Ειδικότερα, τα στοιχεία που μπορεί να αλλάξει είναι: (α) το username, (β) το όνομα, (γ) το επώνυμο, (δ) το email και (ε) τον κωδικό του. Το παλιό όνομα χρήστη δεν χρειάζεται να συμπληρωθεί, καθώς διατηρείται στο σύστημα με τη χρήση των cookies έως την έξοδο του χρήστη από την πλατφόρμα. Με την ολοκλήρωση της προαναφερθείσας διαδικασίας, προβάλλεται το μήνυμα "All set! Changes saved successfully." ενώ διαφορετικά εμφανίζεται το μήνυμα "Oops! Something went wrong. Please try again".

The screenshot displays the 'E-DIABETES PATIENT' web application. On the left is a dark sidebar with a menu containing 'Dashboard', 'Patient Phr', 'Prescription', 'Account', and 'Support'. The 'Account' option is highlighted. The main content area is titled 'Account' and 'Manage your Account'. It contains a form with the following fields: 'First Name' (placeholder: first name), 'Last Name' (placeholder: last name), 'e-mail' (placeholder: e-mail), 'Password' (placeholder: password), 'Current Username' (displayed as mavrogiorgos), and 'New Username' (placeholder: new username). A blue 'Save Changes' button is located at the bottom right of the form. The footer of the page reads 'Copyright © e-diabetes 2021-2022'.

Εικόνα 16: patient account management

5.3.3. Προβολή συνταγής από τον ασθενή (Prescription view from patient's end)

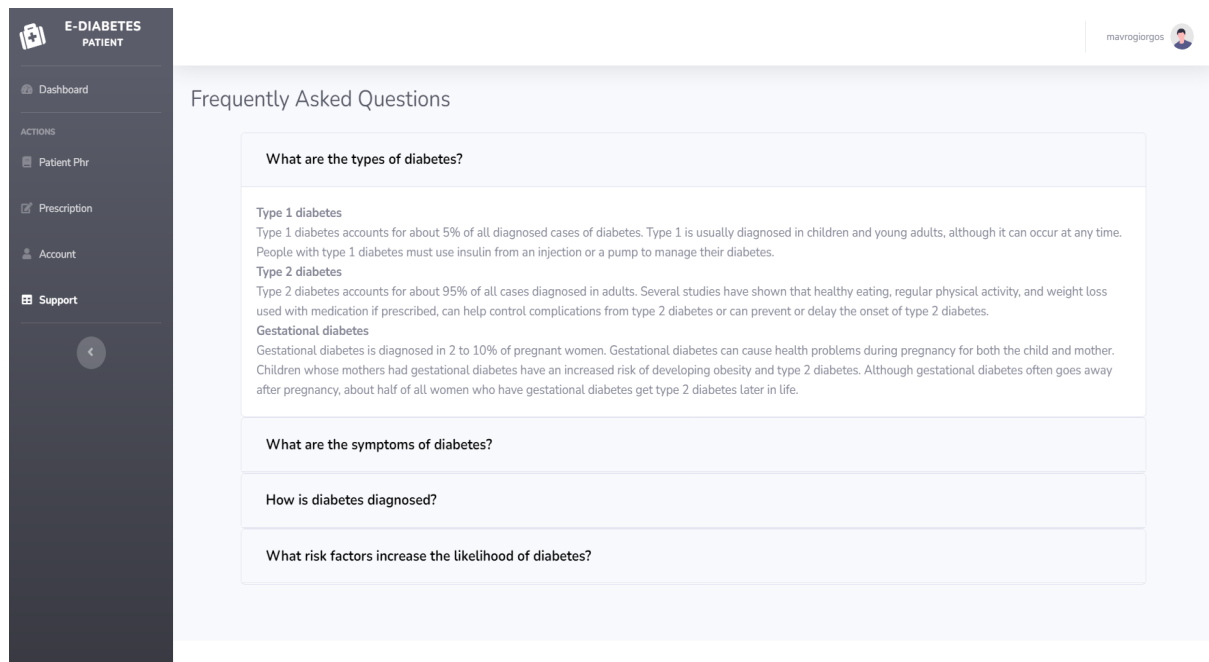
Η λειτουργία αυτή αφορά την προβολή της ηλεκτρονικής συνταγής του ασθενούς που έχει χορηγηθεί από τον γιατρό του. Η διαδικασία που ακολουθείται έχει ως εξής: Αφού ο ασθενής καταχωρήσει τις απαιτούμενες τιμές (Glucose, BloodPressure, Insulin, BMI, Age), ο γιατρός τις ελέγχει και τον ενημερώνει εάν χρειάζεται νέα αγωγή ή όχι , προβαίνοντας στην σύνταξη αντίστοιχου μηνύματος. Ο ασθενής μπορεί εξ αποστάσεως να διαβάσει την συνταγή που του χορηγεί ο γιατρός. Για να τη διαβάσει, επισκέπτεται την αντίστοιχη σελίδα (Prescription) μέσω του μενού επιλογών που βρίσκεται στο sidebar της πλατφόρμας e-Diabetes - Patient.



Εικόνα 17: Prescription view from patient's end

5.3.4. Προβολή συχνών ερωτήσεων / απαντήσεων (F.A.Q.)

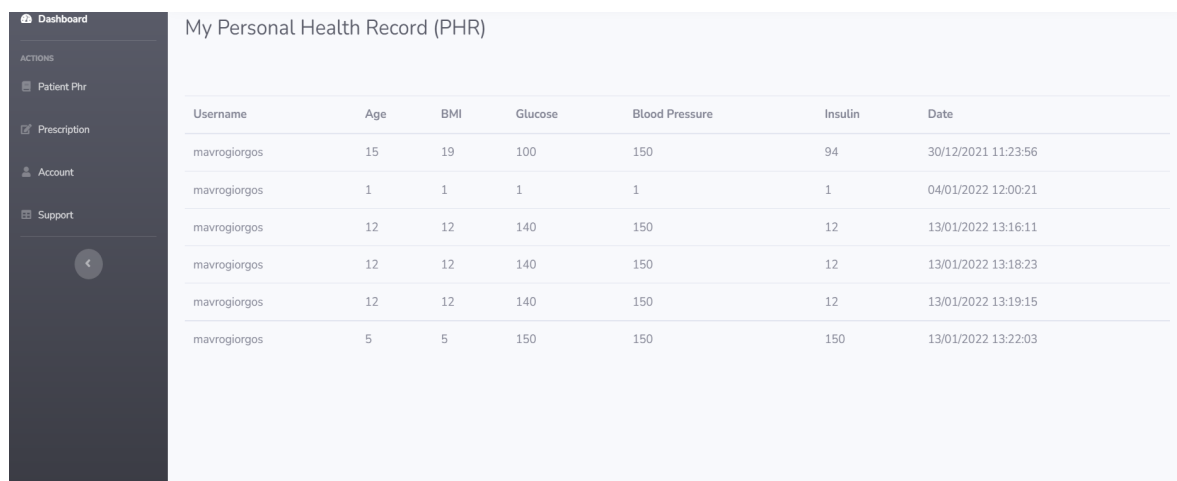
Η λειτουργία αυτή αποτελεί ένα βοηθητικό κομμάτι της πλατφόρμας, τόσο για τον ασθενή όσο και για τον γιατρό, παρέχοντας απαντήσεις σε συχνά ερωτήματα που σχετίζονται με τον διαβήτη. Αναλυτικότερα αυτές οι ερωτήσεις αφορούν: (α) τους τύπους του διαβήτη, (β) τα αντίστοιχα συμπτώματα, (γ) τον τρόπο διάγνωσης και (δ) τους παράγοντες που συμβάλλουν στην εμφάνιση διαβήτη.



Εικόνα 18: F.A.Q. page

5.3.5. Προβολή Προσωπικού Φακέλου Υγείας του ασθενή (Patient's Personal Health Record view)

Η λειτουργία αυτή αφορά την προβολή του προσωπικού φακέλου υγείας του ασθενή. Ειδικότερα, οι τιμές που προβάλλονται είναι οι ακόλουθες: (α) το όνομα χρήστη του ασθενή (username), (β) η ηλικία του (age), (γ) ο δείκτης μάζας σώματος (BMI), (δ) η τιμή της γλυκόζης (glucose), (ε) η τιμή της αρτηριακής πίεσης (Blood Pressure) και (στ) η ημερομηνία (date) εισαγωγής των μετρήσεων, η οποία καταχωρείται αυτόματα από το σύστημα.

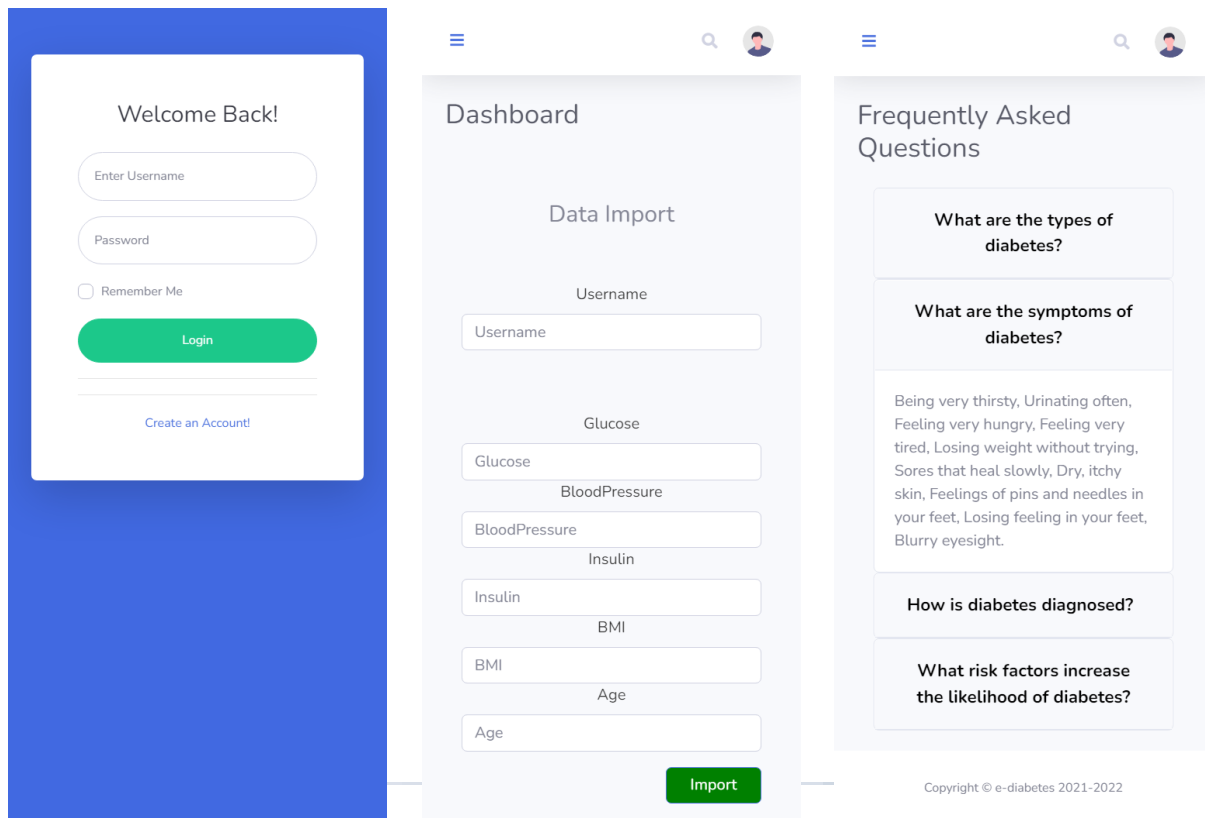


Username	Age	BMI	Glucose	Blood Pressure	Insulin	Date
mavrogiorgos	15	19	100	150	94	30/12/2021 11:23:56
mavrogiorgos	1	1	1	1	1	04/01/2022 12:00:21
mavrogiorgos	12	12	140	150	12	13/01/2022 13:16:11
mavrogiorgos	12	12	140	150	12	13/01/2022 13:18:23
mavrogiorgos	12	12	140	150	12	13/01/2022 13:19:15
mavrogiorgos	5	5	150	150	150	13/01/2022 13:22:03

Εικόνα 19: Patient's Personal Health Record view

5.4. Responsiveness Εφαρμογής e-diabetes

Η διεπαφή της εφαρμογής έχει σχεδιαστεί με τέτοιο τρόπο, ώστε να προσαρμόζεται σύμφωνα με τη συσκευή από την οποία προβάλλεται. Στις ακόλουθες εικόνες παρουσιάζονται, ενδεικτικά, κάποιες από τις σελίδες της εφαρμογής, όπως αυτές φαίνονται από ένα iPhone 12 Pro.



Εικόνες 20-22: e-diabetes responsiveness

Πηγές

[1]Σφέτσος Παναγιώτης, “Ευέλικτες Μέθοδοι Ανάπτυξης Λογισμικού” Διδακτορική Διατριβή, 2007,

<https://people.iese.edu/~sfetsos/Agile%20Methods1.html>

[2]Ρούλα Τσουλέα , Ιστοσελίδα Ιατροπαιδεία (iatropedia), “6 απρόσμενοι παράγοντες που απορρυθμίζουν το ζάχαρο”, 22 Νοεμβρίου 2017

<https://www.iatropedia.gr/ygeia/6-aprosmenoi-paragontes-pou-aporrythmizoun-zacharo/92668/>

[3]Νικόλαος Κούντουρος, “Υπηρεσιοστρεφής Αρχιτεκτονική και Ηλεκτρονική Διακυβέρνηση”, Διατριβή, 2015, <https://dione.lib.unipi.gr/xmlui/handle/unipi/8867>

[4]Docplayer, “Υπηρεσιοστρεφής Αρχιτεκτονική SOA (Service Oriented Architecture), <https://docplayer.gr/7434479-Ypiresiostrefis-arhitektoniki-soa-service-oriented-architecture.html>

[5]Rafael Francozo, Alberto Paucar- Cacaes, Mischel Carmen Neyra Belderrain “Combining Value-Focused thinking and soft systems methodology: A systemic framework to structure the planning process at a special educational needs school in Brazil”, 14 April 2020

<https://www.tandfonline.com/doi/full/10.1080/01605682.2021.1880298>

[6]Hanna Augustsson, Kate Churruca & Jeffrey Braithwaite, “Change and improvement 50 years in the making: a scoping review of the use of soft systems methodology in healthcare”, 23 November 2020

<https://link.springer.com/article/10.1186/s12913-020-05929-5#Sec3>

[7]Καλλιόπη Πεπόνη, “Ανάλυση και σύγκριση συστημικών μεθοδολογιών οργανωσιακής σχεδίασης” , Διπλωματική Εργασία , Μάιος 2007

<http://extev.syros.aegean.gr/bsc/peponi.pdf>

[8]Ιωάννα Κομνηνού, “Εικονικό Περιβάλλον μάθησης και εργασίας με την πλατφόρμα Microsoft Office 365”, Μεταπτυχιακή Διπλωματική Εργασία, Φεβρουάριος 2016

<https://dione.lib.unipi.gr/xmlui/bitstream/handle/unipi/9632/Komninou Ioanna.pdf?sequence=1&isAllowed=y>

- [9]Winter MarkC, Brown D. H., Checkland P. B. "A role for soft systems methodology in information systems development". European Journal of Information Systems, 1995, 4.3: 130-142, <https://link.springer.com/article/10.1057/ejis.1995.17>
- [10]Network I. D. "System development life cycle".Computerworld, 2002, QuickStudy, http://cs.franklin.edu/~smithw/ITEC495_Resources/SDLC_Doesitwork.pdf
- [11]Διαφάνειες Βασιλακόπουλου Γ. στο μάθημα Ανάπτυξης Πληροφοριακών Συστημάτων
- [12]Eric Simon, Christophe Kunzi, Kilian Stoffel, "Scalable Social Protocols to Formalize Systems Development Life Cycles". In: Proceedings of IADIS International Conference e-Society, 2007, p. 177-184, https://www.researchgate.net/profile/Kilian-Stoffel/publication/267936348_SCALABLE_SOCIAL_PROTOCOLS_TO_FORMALIZE_SYSTEMS_DEVELOPMENT_LIFE_CYCLES/links/5527f64b0cf29b22c9b9d05c/SCALABLE-SOCIAL-PROTOCOLS-TO-FORMALIZE-SYSTEMS-DEVELOPMENT-LIFE-CYCLES.pdf
- [13]Oriana Karina Eason, Information systems development methodologies transitions: "An analysis of waterfall to agile methodology", 2016, <https://scholars.unh.edu/cgi/viewcontent.cgi?article=1288&context=honors>
- [14]Βαγγέλης Μονοχρήστου, "Μεθοδολογίες διαχείρισης έργων πληροφορικής: Ευέλικτες Agile Μεθοδολογίες και η χρήση τους σε δημόσια έργα πληροφορική", Διδακτορική Διατριβή, 2010, https://dspace.lib.uom.gr/bitstream/2159/14303/2/Monochristou_PhD2011.pdf
- [15]Georgia M Kapitsaki., Marios Christou Marios. "Where is Scrum in the current Agile world?". In: 2014 9th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE). IEEE, 2014, p. 1-8, <https://www.scitepress.org/Papers/2014/48677/48677.pdf>
- [16]Διαφάνειες Γ. Βασιλακόπουλου, Μάθημα "Ανάπτυξη Πληροφοριακών Συστημάτων".
- [17]Ken Schwaber "Scrum development process. In: Business object design and implementation". Springer, London, 1997, p. 117-134, <http://damiantgordon.com/Methodologies/Papers/Business%20Object%20Design%20and%20Implementation.pdf>

- [18]Popli Rashmi, Chauhan Nar. "Scrum: an agile framework. International Journal of Information Technology and Knowledge Management", 2011, 4.1: 147-149,
<http://csjournals.com/IJITKM/PDF%204-1/30.Rashmi%20Popli1%20&%20Naresh%20Chauhan2.pdf>
- [19]Γ. Βασιλακόπουλος, "Προσεγγίσεις Ανάπτυξη Πραγμάτωση", Αθήνα 2018
- [20]Μαρία Ρηγοπούλου, "Η μεθοδολογία SCRUM και η Εφαρμογή της στην Ανάπτυξη πληροφοριακών συστημάτων", Διπλωματική εργασία, 2014,
https://eclass.uth.gr/modules/document/file.php/DE_P_101/%CE%A5%CE%A0%CE%9F%CE%A3%CE%A4%CE%97%CE%A1%CE%99%CE%9A%CE%A4%CE%99%CE%9A%CE%9F%20%CE%A5%CE%9B%CE%99%CE%9A%CE%9F/%CE%94%CE%99%CE%A0%CE%9B%CE%A9%CE%9C%CE%91%CE%A4%CE%99%CE%9A%CE%95%CE%A3%20%CE%95%CE%A1%CE%93%CE%91%CE%A3%CE%99%CE%95%CE%A3%20%CE%A3%CE%A7%CE%95%CE%A4%CE%99%CE%9A%CE%95%CE%A3%20%CE%9C%CE%95%20%CE%95%CE%A5%CE%95%CE%9B%CE%99%CE%9A%CE%A4%CE%95%CE%A3%20%CE%9C%CE%95%CE%98%CE%9F%CE%94%CE%9F%CE%A5%CE%A3/master_thesis_rigopoulou.pdf
- [21]Ανδρέας Μενύχτας, διαφάνειες μαθήματος "Ανάπτυξη Πληροφοριακών Συστημάτων", 2021-2022.