**ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ**

Graphical user interface

Description automatically generated with medium confidence

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

**ΤΕΧΝΟΛΟΓΙΑ ΛΟΓΙΣΜΙΚΟΥ 6Ο ΕΞΑΜΗΝΟ**

ΑΡΓΥΡΑ ΟΥΡΑΝΙΑ Π20023

ΑΡΜΕΝΗ ΑΘΗΝΑ Π20025

ΠΑΝΑΓΟΠΟΥΛΟΣ ΑΛΕΞΑΝΔΡΟΣ-ΒΑΣΙΛΕΙΟΣ Π20150

ΓΕΩΡΓΙΑΔΗΣ ΝΙΚΟΛΑΟΣ Π19032

ΕΡΓΑΣΙΑ 2022-2023

**ΠΕΙΡΑΙΑΣ 2022-2023**

Στην εργασία μας ζητείται να υλοποιήσουμε μια desktop εφαρμογή δημιουργίας διαιτολογίου.

Αρχικά, θα ορίσουμε τους στόχους της εργασίας, οι οποίοι κρίνονται απαραίτητοι, καθώς λειτουργούν ως μέτρα επίδοσης και αξιολόγησης της προόδου και της επίτευξης των αποτελεσμάτων. Θέτοντας τους στόχους, μπορούμε να οργανώσουμε και να διαχειριστούμε την εφαρμογή μας με πιο αποδοτικό τρόπο όπως επίσης και να κατανοήσουμε τις ανάγκες και τις προσδοκίες του χρήστη.

1. **Εισαγωγή**

**1.1 Στόχοι της εργασίας**

1. **Αντικειμενικά χαρακτηριστικά :** Στόχος μας είναι να αναπτυχθεί λογισμικό, καθώς και να συνοδευτεί με την αντίστοιχη ανάπτυξή του σε μοντέλο κύκλου ζωής λογισμικού. Αυτό περιλαμβάνει την εισαγωγή κάποιων παραμέτρων, όπως για παράδειγμα στοιχεία ασθενή και προτιμήσεις σε φαγητά. Με βάση αυτά τα χαρακτηριστικά, ο διαιτολόγος, ως τελικός χρήστης της εφαρμογής, θα μπορεί να εκδώσει ένα εβδομαδιαίο πρόγραμμα με βάση τις ανάγκες του πελάτη.
2. **Ποιότητα και Απόδοση :**  Αποτελούν βασικές πτυχές του προγράμματος. Το πρόγραμμα μας, θα πρέπει να είναι αξιόπιστο για να μπορεί ο διαιτολόγος να βασιστεί πάνω σε αυτό, έγκυρο, εύκαμπτο, αποδοτικό και αποτελεσματικό. Άλλες θεμελιώδεις ιδιότητες που θέλουμε να πληροί το πρόγραμμά μας είναι η χρηστικότητα, η φορητότητα και η συντηρησιμότητα.
3. **Ασφάλεια :** Απαραίτητη ιδιότητα για την εφαρμογή μας, καθώς τα δεδομένα που θα εκχωρούνται θα πρέπει να προστατεύονται και να παραμένουν ιδιωτικά (ιδιωτικότητα). Το πρόγραμμα δε θα πρέπει να είναι ευπαθή και ευάλωτο, δηλαδή να έχει ελαττώματα ή αδυναμίες που θα μπορούσαν να χρησιμοποιηθούν από κάποιον κακόβουλο χρήστη που μπορεί να χειριστεί τα δεδομένα ή γενικότερα πληροφορίες της εφαρμογής.
4. **Επεκτασιμότητα :** Η εφαρμογή θα πρέπει να είναι εύκολα επεκτάσιμη και προσαρμόσιμη σε μελλοντικές απαιτήσεις και αλλαγές.
5. **Ικανοποίηση του χρήστη :** Κύριος στόχος μας είναι ο χρήστης να μείνει ικανοποιημένος από την εφαρμογή μας και να καλύπτονται οι ανάγκες του και οι απαιτήσεις του. Αυτό θα επιτευχθεί με την ευχρηστία της εφαρμογής, την επεξήγησή της, το εγχειρίδιο χρήσης, κτλ.
6. **Φιλικό προς το χρήστη :** Δε θα πρέπει να δυσχεραίνει το χρήστη και θα πρέπει να είναι κατανοητό προς αυτόν. Επίσης, θα υπάρχουν οδηγίες και πληροφορίες για την ευκολία εκμάθησης της εφαρμογής, έτσι ώστε να αναπτυχθεί μία «οικειότητα» με το χρήστη.
7. **Ανθεκτικότητα :** Η εφαρμογή θα λειτουργεί υπό κανονικές ή συνήθεις συνθήκες. Να υπάρχει δηλαδή λειτουργικότητα (υποστήριξη των ενεργειών που θα ήθελε να κάνει ο χρήστης) και να μπορεί να ανταπεξέλθει σε πιθανά λάθη που μπορεί να κάνει ο χρήστης (πχ εισαγωγή αρνητικού βάρους ή ύψους). Σε αυτό συμβάλει η προβλεψιμότητα των ενεργειών που θα κάνει ο χρήστης και η ανανηψιμότητα δηλαδή η δυνατότητα επανόρθωσης λάθους.
8. **Συνέπεια :** Χρησιμοποίηση ίδιων συμβάσεων καθ’ όλη τη διάρκεια της εφαρμογής . Για παράδειγμα, να μη γίνεται απότομη αλλαγή χρωματισμού, απότομη αλλαγή κουμπιών και γενικότερα διαφορετικές αναπαραστάσεις για το ίδιο πράγμα, καθώς αυτό θα δυσκολέψει το χρήστη κατά τη χρήση της εφαρμογής.
   1. **Ορισμός του προβλήματος προς επίλυση**

Γενικότερα, για να επιλύσουμε ένα πρόβλημα θα πρέπει πρώτα να το έχουμε ορίσει. Είναι ένα κρίσιμο βήμα το οποίο αναλύεται στα εξής στάδια:

1. **Καθορισμός του προβλήματος :** Χρειάζεται να διατυπώσουμε μία σαφή και συνοπτική δήλωση προβλήματος που να αποτυπώνει την ουσία του προβλήματος. Καλούμαστε να δημιουργήσουμε μία desktop εφαρμογή, η οποία θα δημιουργεί διαιτολόγια, ανάλογα με τα στοιχεία του πελάτη που θα εκχωρούνται. Για να γίνουν όμως αυτές οι ενέργειες, θα πρέπει να εκχωρήσουμε στην αρχή έναν πίνακα τροφίμων και την περιεκτικότητά τους σε διάφορα στοιχεία όπως πρωτεϊνες, υδατάνθρακες, ενέργεια (θερμίδες-kcal), λίπη, κτλ. Για να γίνει η δημιουργία λογαριασμού, πατάμε εγγραφή και συμπληρώνουμε τα στοιχεία μας όπως Όνομα, Επώνυμο, Κωδικό Πρόσβασης και E-mail. Τέλος, για να γίνει η διαφοροποίηση μεταξύ ασθενών και διαιτολόγων-διατροφολόγων, υπάρχουν 2 κουμπιά όπου ο κάθε πελάτης επιλέγει ως τι από τα 2 θέλει να εγγραφεί στην εφαρμογή. Ο -διατροφολόγος έχει τη δυνατότητα να εκδίδει πλήρη εβδομαδιαία προγράμματα και να βλέπει αυτά που έχει ήδη εκδώσει για τους πελάτες του, ενώ ο πελάτης θα βλέπει τα πλήρη εβδομαδιαία προγράμματα που έχουν εκδοθει μόνο για εκείνον και δε θα μπορεί να δημιουργήσει γενικά διαιτολόγια.
2. **Αναγνώριση των ενδιαφερόμενων φορέων :** Από αυτό το λογισμικό, επωφελούνται οι διαιτολόγοι-διατροφολόγοι και οι ασθενείς τους, αφού μπορούν σε ελάχιστο χρόνο να έχουν ένα πλήρη εβδομαδιαίο πρόγραμμα διατροφής, το οποίο μπορεί να εκδοθεί και εξ’ αποστάσεως.
3. **Συλλογή απαιτήσεων :** Για την έκδοση του εβδομαδιαίου προγράμματος θα πρέπει να ληφθούν υπόψη οι απαιτήσεις και ανάγκες του πελάτη, δηλαδή εάν ο πελάτης θέλει να αυξήσει, να μειώσει ή να κρατήσει σταθερό το βάρος του.
4. **Σύντομη παρουσίαση της RUP**

Η μεθοδολογία Rational Unified Process Methodology ή αλλιώς RUP είναι μία ευέλικτη μέθοδος ανάπτυξης λογισμικού, στην οποία ο κύκλος ζωής ενός έργου, ή η ανάπτυξη λογισμικού, χωρίζεται σε 4 φάσεις :

* **Μοντελοποίηση (Φάση Έναρξης)**
* **Ανάλυση και Σχεδιασμός (Φάση Επεξεργασίας)**
* **Υλοποίηση (Φάση Κατασκευής)**
* **Δοκιμή και Εφαρμογή (Μεταβατική Φάση)**

**Φάση Έναρξης (Inception)**

Κατά την πρώτη φάση καθορίζεται η βασική ιδέα και η δομή του έργου. Επίσης, καθορίζεται η αναγκαιότητα του έργου, η βιωσιμότητά του και η καταλληλότητά του και μας επιτρέπει να εκτιμήσουμε το κόστος και τον απαιτούμενο χρόνο.

**Φάση Επεξεργασίας (Elaboration)**

Σε αυτό το στάδιο γίνεται ο προγραμματισμός και η μοντελοποίηση. Επίσης, αξιολογούνται και αναλύονται οι απαιτήσεις της εφαρμογής. Εκτελείται λεπτομερές σχέδιο αξιολόγησης και ανάπτυξης και μειώνονται οι κίνδυνοι.

**Φάση Κατασκευής (Construction)**

Το έργο αναπτύσσεται και ολοκληρώνεται, δηλαδή το σύστημα λογισμικού κατασκευάζεται στο σύνολό του. Δημιουργείται σύστημα ή πηγαίος κώδικας και στη συνέχεια γίνεται δοκιμή. Επίσης, πραγματοποιείται κωδικοποίηση. Τέλος , πραγματοποιούνται και κάποια test cases, ενώ σε αυτή τη φάση περιλαμβάνεται και το εγχειρίδιο χρήσης.

**Μεταβατική Φάση (Transition)**

Το τελικό έργο κυκλοφορεί στο κοινό, δηλαδή είναι έτοιμο για χρήση. Μόλις ο χρήστης αρχίσει να χρησιμοποιεί το σύστημα, σχεδόν πάντα προκύπτουν προβλήματα που απαιτούν πραγματοποίηση αλλαγών στο σύστημα. Στόχος, ωστόσο, είναι να διασφαλιστεί μια θετική και ομαλή μετάβαση στον χρήστη. Πραγματοποιείται δοκιμή beta, και τα ελαττώματα αφαιρούνται από την εφαρμογή με βάση τα σχόλια από το κοινό.

Για τη δημιουργία του διαγράμματος RUP θα χρησιμοποιήσουμε την εφαρμογή drawio.com , <https://app.diagrams.net/> .

Για την εργασία δημιουργήσαμε το εξής προγραμματισμό ενεργειών για το διάγραμμα RUP ο οποίος περιλαμβάνει το όνομα της εφαρμογή (Task Name), τη διάρκεια που χρειάζεται για την πλήρη δημιουργία της (Duration), την ημερομηνία έναρξης της δημιουργίας της (Start) και την ημερομηνία λήξης της (ETA : Estimated Time of Arrival). Για τη **φάση της έναρξης (Inception)** χρειαζόμαστε 7 ημέρες για να δημιουργήσουμε τον πίνακα τροφίμων, να αναλύσουμε τα διατροφικά χαρακτηριστικά του, να συλλέξουμε τα στοιχεία του ασθενή και να αξιολογήσουμε τις διατροφικές ανάγκες του (για παράδειγμα αν θέλει να αυξήσει το βάρος του, χρειαζόμαστε μία διατροφή με περισσότερες θερμίδες από αυτές που προσλαμβάνει ημερησίως). Για τη **φάση της επεξεργασίας (Elaboration)** χρειαζόμαστε 11 ημέρες όπου σε αυτές θα πρέπει να γίνεται εγγραφή στην εφαρμογή-δημιουργία λογαριασμού, επιλογή εγγραφής ως διαιτολόγος ή ασθενής, σύνδεση στην εφαρμογή, εισαγωγή στοιχείων ασθενή (που συλλέξαμε στη φάση της έναρξης), δημιουργία ενός Use Case διαγράμματος (διάγραμμα περιπτώσεων χρήσης που περιγράφει τις υψηλού επιπέδου λειτουργίες και το πεδίο εφαρμογής ενός συστήματος και προσδιορίζονται οι αλληλεπιδράσεις μεταξύ συστήματος και χρηστών) και έλεγχος για το αν τα στοιχεία που δόθηκαν από το χρήστη είναι έγκυρα, για παράδειγμα ο δείκτης μάζας σώματος του χρήστη μπορεί να είναι κάτω από 18,5, το οποίο σημαίνει ότι ανήκει στο εύρος λιποβαρών, άρα δε μπορουμε να δώσουμε στο χρήστη διαιτολόγιο που θα στοχεύει στη μείωση του βάρους του, διότι κρίνεται επικίνδυνο για την υγεία του. Στη συνέχεια, χρειαζόμαστε 25 ημέρες για τη **φάση κατασκευής (Construction)** στην οποία θα γίνει η σχεδίαση συστήματος-λογισμικού (Code Design), η δημιουργία διαιτολογίου του ασθενή, έλεγχος για τυχόν απαιτούμενες αλλαγές στο διαιτολόγιο με βάση τις ανάγκες και απαιτήσεις του ασθενή, δοκιμές της εφαρμογής ή αλλιώς (Test Cases) και εγχειρίδιο χρήσης της εφαρμογής (User Manual). Τέλος, για τη **μεταβατική φάση (Transition)** χρειαζόμαστε 12 ημέρες για να γίνει έλεγχος για τυχόν προβλήματα της εφαρμογής, πραγματοποίηση αλλαγών στο σύστημα ή αλλιώς επίλυση των προβλημάτων που βρέθηκαν, έλεγχος απ’ το διαιτολόγο για την καταλληλότητα του διαιτολογίου της εφαρμογής, παράδοση του διαιτολογίου στον ασθενή, έλεγχος διαιτολογίου απ’ τον ασθενή (Δοκιμή beta) το οποίο είναι προσαρμοσμένο στις απαιτήσεις και τις ανάγκες του, ανάλυση κριτηκής του χρήστη και τέλος υποστήριξη και συντήρηση της εφαρμογής.

A screenshot of a computer

Description automatically generated

Λεξικό :

**Beta Testing** : Στην ανάπτυξη λογισμικού, μία δοκιμή beta είναι η δεύτερη φάση της δοκιμής λογισμικού κατά την οποία μία δειγματοληψία του κοινού για το οποίο προορίζεται δοκιμάζει την εφαρμογή και την καταλληλότητά της.

Βιβλιογραφία : <https://www.geeksforgeeks.org/> , <https://www.toolshero.com/> , Σημειώσεις μαθήματος «Τεχνολογία Λογισμικού», Σημειώσεις μαθήματος «Αλληλεπίδραση Ανθρώπου-Υπολογιστή», <https://www.techtarget.com/> ,