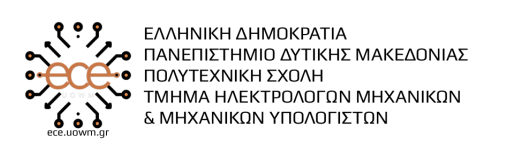
****

**Αναπτυξη Υπηρεσιας Διαχειρισης Ανθρωπινου Δυναμικου Μικρομεσαιων Επιχειρησεων Πληροφορικης.**

**ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ**

****

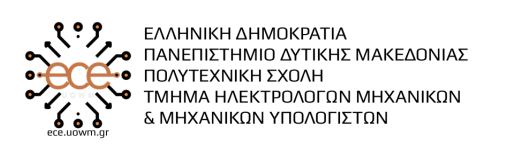
**του**

**ΑΝΑΣΤΑΣΟΠΟΥΛΟΥ ΓΕΩΡΓΙΟΥ**

**Επιβλεπων :** **Σαρηγιαννιδης Παναγιωτης, Καθηγητης**

**ΚΟΖΑΝΗ/ΜΗΝΑΣ/ΕΤΟΣ**

**ΑΥΤΗ Η ΣΕΛΙΔΑ ΕΙΝΑΙ ΣΚΟΠΙΜΑ ΛΕΥΚΗ**



**Development Of A Human Resource Management System For Small And Medium Software Enterprises**

**Thesis**

****

**Of**

**ANASTASOPOULOS GEORGIOS**

**Επιβλεπων : Σαρηγιαννιδης Παναγιωτης, Καθηγητης**

**ΚΟΖΑΝΗ/ΜΗΝΑΣ/ΕΤΟΣ**

**ΑΥΤΗ Η ΣΕΛΙΔΑ ΕΙΝΑΙ ΣΚΟΠΙΜΑ ΛΕΥΚΗ**

ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ ΚΑΙ ΑΝΑΛΗΨΗΣ ΠΡΟΣΩΠΙΚΗΣ ΕΥΘΥΝΗΣ

Δηλώνω ρητά ότι, σύμφωνα με το άρθρο 8 του Ν. 1599/1986 και τα άρθρα 2,4,6 παρ. 3

του Ν. 1256/1982, η παρούσα Διπλωματική Εργασία με τίτλο “ Αναπτυξη Υπηρεσιας Διαχειρισης Ανθρωπινου Δυναμικου Μικρομεσαιων Επιχειρησεων Πληροφορικης.” καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας και αναφέρονται

ρητώς μέσα στο κείμενο που συνοδεύουν, και η οποία έχει εκπονηθεί στο Τμήμα

Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Πανεπιστημίου Δυτικής

Μακεδονίας, υπό την επίβλεψη του μέλους του Τμήματος κ. Σαρηγιαννιδη Παναγιωτη αποτελεί αποκλειστικά προϊόν προσωπικής εργασίας και δεν προσβάλλει κάθε μορφής πνευματικά

δικαιώματα τρίτων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που

χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον. Τα σημεία

όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή / και πηγές άλλων συγγραφέων,

αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική

αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη

περιγραφή. Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας

εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η

ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή

ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να

διατηρείται το παρόν μήνυμα. Ερωτήματα που αφορούν τη χρήση της εργασίας για

κερδοσκοπικό σκοπό πρέπει να απευθύνονται προς τον συγγραφέα. Οι απόψεις και τα

συμπεράσματα που περιέχονται σε αυτό το έγγραφο εκφράζουν τον συγγραφέα και μόνο.

Copyright (C) Αναστασοπουλος Γεωργιος, Σαρηγιαννιδης Παναγιωτης, 2023, Κοζάνη

Copyright (C) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ , \_\_\_\_\_\_\_ , \_\_\_\_\_\_\_\_\_\_\_\_

Υπογραφή Φοιτητή: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Δίνεται η περίληψη της μεταπτυχιακής διπλωματικής εργασίας με 150-200 λέξεις. Αναφέρονται ο σκοπός της διπλωματικής εργασίας, καθώς και η μεθοδολογία που χρησιμοποιήθηκε. Επίσης αναφέρονται οι τεχνολογίες που χρησιμοποιήθηκαν

Γίνεται αναφορά στα αποτελέσματα και τα συμπεράσματα.

**Λέξεις Κλειδιά:**<<……>>

**Abstract**

*Software development is a complex process that requires the collaboration of many different disciplines. In order for companies to produce high-quality software, they must organize the stages of development very well, so that the work is manageable, sustainable and ultimately serves the needs of the customer. The stages through which a software goes, from the conception of the idea to the disposal of the final product, are so manyis a very common fact that makes mistakes very frequent during the development by the development teams. The purpose of this thesis is to present the SDLC and its aspects and how through the models it includes to highlight the benefits it has especially in small and medium enterprises. In a nutshell, the software development life cycle (SDLC) is a methodology used to organize and manage application development as it divides and divides application development into various phases, each with its own goals and tasks. In order to complete the analysis of the SDLC, this diploma is accompanied by an application where it was developed with popular technologies in the field. In particular, the use of React.js was chosen for the front end of the application and Spring Boot for the back end, due to the flexibility they provide and the speed of developing quality code. The application aims to offer a simple and understandable environment of a personnel management platform that satisfies the needs of a small business, showing that even at an early stage, such an application can only bring positive results.*

**Keywords:**<<……>>

**Πίνακας περιεχομένων**

1 Εισαγωγή (Introduction) 4

1.1<Τίτλος που έχει σχέση με τον γενικότερο χώρο εφαρμογής της διπλωματικής (Security in critical infastructures)> 5

1.2Αντικείμενο διπλωματικής 6

*1.2.1Συνεισφορά 7*

1.3Οργάνωση κειμένου 8

2 Θεωρητικό υπόβαθρο (Technology background) 9

2.1Software Development Life Cycle (SDLC) 10

2.2SDLC Models & Methodologies 11

2.3Προγραμματιστικά Framework (Programming/Software frameworks) 12

2.4 Προγραμματιστικά Framework (Programming/ Software frameworks)

3 Σχεδιασμός και ανάπτυξη συστήματος (System development) 13

3.1 Περιγραφη συστηματος, Χρηστες ( User Roles)

3.2 Απαιτήσεις και λειτουργιες συστήματος

3.3 Περιπτωσεις Χρησης (Use Cases)

3.4 Αρχιτεκτονική συστήματος (System architecture) 15

3.5 Βαση Δεδομενων 16

4 <Λειτουργίες και διεπαφή χρήστη (UI/UX)> 18

5 Επίλογος (Conclusions/Future work) 19

5.1Σύνοψη και συμπεράσματα 20

5.2Μελλοντικές επεκτάσεις 21

6 Βιβλιογραφία 22

7 Παράρτημα 23

8 Οδηγίες Εγκατάστασης Υπηρεσίας 25

***1. Εισαγωγή (Introduction)***

Η ανάπτυξη εφαρμογών είναι μια πολύπλοκη διαδικασία που απαιτεί τη συνεργασία πολλών διαφορετικών ειδικοτήτων. Ο κύκλος ζωής ανάπτυξης λογισμικού (SDLC) είναι μια μεθοδολογία που χρησιμοποιείται για να οργανώσει και να διαχειριστεί την ανάπτυξη εφαρμογών. Ο SDLC διαιρεί την ανάπτυξη εφαρμογών σε διάφορες φάσεις, καθεμία από τις οποίες έχει τους δικούς της στόχους και εργασίες.

Η εφαρμογή που θα αναπτυχθεί στο πλαίσιο αυτής της διπλωματικής εργασίας θα είναι μια πλατφόρμα διαχείρισης έργων που θα βοηθήσει τις επιχειρήσεις να διαχειριστούν αποτελεσματικότερα τα έργα τους. Η πλατφόρμα θα παρέχει δυνατότητες για την παρακολούθηση του προόδου των έργων, την διαχείριση των πόρων, την επικοινωνία μεταξύ των μελών των ομάδων έργων και την αναφορά.

Η πλατφόρμα θα αναπτυχθεί χρησιμοποιώντας το Spring Boot και το React. Το Spring Boot είναι ένα Java framework που διευκολύνει την ανάπτυξη και την ανάπτυξη εφαρμογών Java. Η React είναι μια βιβλιοθηκη της JavaScript που χρησιμοποιείται για την ανάπτυξη δυναμικών εφαρμογών ιστού.

Η διπλωματική εργασία θα χωριστεί σε τέσσερις κύριες ενότητες. Η πρώτη ενότητα θα παρουσιάσει μια επισκόπηση του SDLC. Η δεύτερη ενότητα θα συζητήσει τα διαφορετικά μοντέλα και μεθοδολογίες SDLC. Η τρίτη ενότητα θα περιγράψει τον σχεδιασμό και την ανάπτυξη της πλατφόρμας. Η τέταρτη ενότητα θα παρουσιάσει τις λειτουργίες και τη διεπαφή χρήστη της πλατφόρμας.

Η διπλωματική εργασία θα ολοκληρωθεί με ένα συμπέρασμα και μια συζήτηση για μελλοντικές εργασίες.

***1.1 <Τίτλος που έχει σχέση με τον γενικότερο χώρο εφαρμογής της διπλωματικής (Security in critical infastructures)>***

Εδώ αυτή κάνουμε μια γενική περιγραφή του χώρου εφαρμογής της διπλωματικής. Αναφέρουμε τα χαρακτηριστικά του χώρου και καταλήγουμε στα γενικότερα προβλήματα που αντιμετωπίζει ο χώρος. Η συζήτηση των προβλημάτων θα πρέπει να προϊδεάζει τον αναγνώστη για το τι θα προσπαθήσει να αντιμετωπίσει η διπλωματική, χωρίς ακόμα να αναφερόμαστε συγκεκριμένα στο αντικείμενο της διπλωματικής.

***1.2 Αντικείμενο διπλωματικής***

Εδώ αναφερόμαστε συγκεκριμένα στο τί θα κάνει η διπλωματική. Αναφέρουμε λεπτομερώς α) τα προβλήματα που θα λύσει (και που ήδη έχουν περιγραφεί γενικά στην προηγούμενη ενότητα), και β) πώς σκοπεύει να τα λύσει.

Είναι σημαντικό κάποιος που θα διαβάσει την ενότητα αυτή να καταλάβει σε σημαντικό βαθμό τον σκοπό της διπλωματικής σας και τις τεχνικές δυσκολίες της, χωρίς να είναι αναγκαίο να δει όλα τα άλλα κεφάλαια. Η ενότητα αυτή θέλει πολύ προσοχή και καλύτερα να τη γράψετε αφού έχετε γράψει όλα τα υπόλοιπα κεφάλαια.

***1.2.1 Συνεισφορά***

Εδώ παραθέτουμε αριθμητικά συγκεκριμένες ενέργειες/λύσεις/μεθοδολογίες που παρουσιάζει η διπλωματική και λύνουν τα προβλήματα που υποσχεθήκαμε στην προηγούμενη ενότητα ότι θα λύσει η διπλωματική. Συνήθως η υποενότητα αυτή έχει την παρακάτω μορφή:

Η συνεισφορά της διπλωματικής συνοψίζεται ως εξής:

1. Μελετήσαμε συστήματα κ.λ.π.
2. Υλοποιήσαμε 3 αλγορίθμους υπολογισμού κ.λ.π.

Αξιολογήσαμε την επίδοση των αλγορίθμων και βρήκαμε ότι κ.λ.π.

1. Ενσωματώσαμε τους αλγορίθμους σε πρότυπο σύστημα κ.λ.π.
2. ...

***1.3 Οργάνωση κειμένου****Το παρόν κεφάλαιο αποτελεί την εισαγωγή της διπλωματικής εργασίας, στο οποίο παρουσιάζουμε τον κεντρικό τομέα της έρευνας, τη σημασία του προβλήματος και τους στόχους που έχουν τεθεί. Στη συνέχεια, θα προχωρήσουμε στο κεφάλαιο 2, όπου θα αναλύσουμε το θεωρητικό υπόβαθρο του έργου, παρουσιάζοντας το SDLC και εξετάζοντας διάφορες μεθοδολογίες που περιλαμβανει..  
Συνεχίζοντας, στο κεφάλαιο 3 θα εστιάσουμε στον σχεδιασμό και την ανάπτυξη του συστήματος, αναλύοντας τις απαιτήσεις και τις λειτουργιες του συστήματος, τα σεναρια χρησης, το μεσο αποθηκευσης, και γενικοτερα τις τεχνολογιες που χρησιμοποιηθηκαν.  
Στο κεφάλαιο 4, θα εξετάσουμε τον σχεδιασμό της διεπαφής χρήστη (UI) και της εμπειρίας χρήστη (UX), με έμφαση στη φιλικότητα και την ευχρηστία της.  
Στο κεφάλαιο 5, θα παρουσιάσουμε τα συμπεράσματα και τις μελλοντικές επεκτάσεις της εργασίας. Θα ανασκοπήσουμε τους στόχους που έχουν επιτευχθεί και θα αξιολογήσουμε τα αποτελέσματα. Θα επισημάνουμε επίσης περαιτέρω βελτιώσεις και δυνατότητες ανάπτυξης για το μέλλον.  
Στο κεφάλαιο 6, θα παραθέσουμε τη βιβλιογραφία που χρησιμοποιήσαμε για την έρευνα και τον σχεδιασμό της εφαρμογής.  
Στο κεφάλαιο 7, θα παραθέσουμε παραρτήματα με πληροφορίες που δεν είναι απαραίτητες για τον βασικό κείμενο αλλά είναι χρήσιμες για επιπλέον κατανόηση και αναφορά.*

*Τέλος, στο κεφάλαιο 8 θα παρέχουμε λεπτομερείς οδηγίες εγκατάστασης και χρήσης της εφαρμογής, προσφέροντας έναν πλήρη οδηγό για τους χρήστες και τους διαχειριστές.  
Μέσω αυτής της διάρθρωσης, επιδιώκουμε να παρουσιάσουμε μια συνολική και συστηματική προσέγγιση στον τομέα του SDLC και στον σχεδιασμό και ανάπτυξη της εφαρμογής. Η διπλωματική εργασία αναζητά την αποτελεσματικότερη και αποδοτικότερη λύση για την διαχείριση και την ανάπτυξη σύγχρονου λογισμικού, ενισχύοντας την αξία και την επιτυχία των οργανισμών και των κρίσιμων υποδομών.*

***Κεφάλαιο 2  
Θεωρητικό υπόβαθρο (Technology background)***

Το κεφάλαιο 2 παρουσιάζει το θεωρητικό υπόβαθρο της ανάπτυξης λογισμικού και εφαρμογών. Ξεκινάει εξηγώντας το Software Development Life Cycle (SDLC) και επισημαίνουμε τα οφέλη του, ιδίως από οικονομικής άποψης αλλά και τους τομείς εφαρμογής του. Στη συνέχεια, εξετάζεται η λειτουργία του SDLC, σε ποιες φάσεις χωρίζονται οι διαδικασίες αναπτυξης, τους ρόλους που παίρνουν τα μέλη σε αυτές τις φάσεις, καθώς και τα διάφορα μοντέλα υλοποίησης του SDLC αναλόγως εφαρμογής. Τέλος, γίνεται μια εισαγωγή στην έννοια των προγραμματιστικών frameworks και πώς αυτά διευκολύνουν τους προγραμματιστές να δημιουργήσουν λογισμικό μεγαλύτερης αποτελεσματικότητας και πιο γρήγορα, παρουσιάζοντας κάποια από τα πιο γνωστά, με μεγαλύτερη έμφαση σε αυτά που χρησιμοποιούνται για την υλοποίηση της εφαρμογής.  
  
Το κεφάλαιο 2 έχει σαν στόχο να παρέχει μια συνοπτική εξηγηση του ορου SDLC στον αναγνωστη και να κανει ξεκαθαρες όλες τις απαιτησεις που θα μπορουσε να εχει μια εφαρμογη στα κύρια θέματα που θα εξεταστούν στην παρούσα διπλωματική, προσφέροντας την απαραίτητη βάση για την κατανόηση και ανάλυση της ανάπτυξης λογισμικού, καθώς και σαν εισαγωγή, για την ευκολότερη κατανόηση του κεφαλαίου 3, που αναφέρεται σε πιο τεχνικά θέματα, σχετικά με την υλοποίηση της εφαρμογής που συνοδεύει την διπλωματική.  
 ***2.1*** ***Software Development Life Cycle (SDLC)***

***Ορισμός του SDLC***

Ο όρος "SDLC" αναφέρεται στον κύκλο ζωής ανάπτυξης λογισμικού (Software Development Life Cycle) και αντιπροσωπεύει την διαδικασία οργάνωσης, σχεδίασης, ανάπτυξης, δοκιμής και συντήρησης λογισμικού. Το SDLC περιγράφει τις φάσεις και τις διαδικασίες που ακολουθούνται κατά την ανάπτυξη λογισμικού από την αρχική ιδέα μέχρι την παράδοση του προϊόντος στον τελικό χρήστη, η ακομα και μετα την παραδοση, στην συντηρηση του λογισμικου.

***Σημασία του SDLC στην ανάπτυξη λογισμικού***

Η σημασία του SDLC είναι κρίσιμη για την επιτυχημένη ανάπτυξη λογισμικού και τη διασφάλιση υψηλής ποιότητας του προϊόντος. Χρησιμοποιώντας το SDLC, οι ομάδες ανάπτυξης μπορούν να διαχειριστούν αποτελεσματικά το έργο τους, να εντοπίζουν προβλήματα εγκαίρως και να λαμβάνουν τα απαραίτητα μέτρα για την αντιμετώπισή τους. Ο χχρο

***Οφέλη του SDLC***

Μείωση δαπανών ανάπτυξης: Ο σχεδιασμός και ο καλός προγραμματισμός που επιτυγχάνεται με το SDLC μειώνουν τις πιθανότητες εμφάνισης σφαλμάτων και ανωτάτων δαπανών για τη διόρθωσή τους αργότερα.

Αύξηση παραγωγικότητας: Η μεθοδολογία SDLC ενισχύει τη συνεργασία και την επικοινωνία μεταξύ των μελών της ομάδας ανάπτυξης, βοηθώντας στην αποτελεσματική διαχείριση του χρόνου και των πόρων.

Μείωση χρόνου ανάπτυξης: Το SDLC βοηθά στην οργάνωση της διαδικασίας ανάπτυξης, μειώνοντας τον χρόνο που απαιτείται για την παραγωγή και την παράδοση του λογισμικού.

Αύξηση ποιότητας του προϊόντος: Η συστηματική και δομημένη προσέγγιση του SDLC βελτιώνει την ποιότητα του παραγόμενου λογισμικού, προσφέροντας ένα πιο αξιόπιστο και αποτελεσματικό προϊόν.

Πιο αποτελεσματική συντήρηση: Οι καλά καθορισμένες διαδικασίες και η τεκμηρίωση που προκύπτουν από το SDLC διευκολύνουν την αντιμετώπιση αλλαγών και βελτιώσεων στο λογισμικό, κάνοντας τη συντήρηση πιο αποτελεσματική και αποδοτική.

Μείωση απωλειών λόγω σφαλμάτων: Η ακριβής ανάλυση και δοκιμή που πραγματοποιούνται κατά το SDLC μειώνει την πιθανότητα εμφάνισης σφαλμάτων στο τελικό προϊόν, μειώνοντας τις απώλειες που μπορεί να προκληθούν λόγω μη λειτουργικού λογισμικού.

***Οργανισμοί και επιχειρήσεις που χρησιμοποιούν το SDLC***

Το SDLC είναι ευρέως χρησιμοποιούμενο από διάφορες επιχειρήσεις και οργανισμούς που επιθυμούν να αναπτύξουν λογισμικό και εφαρμογές υψηλής ποιότητας. Είναι κρίσιμο για εταιρείες που εξαρτώνται από λογισμικό για τις δραστηριότητές τους, ανεξαρτήτως μεγέθους και βιομηχανίας.  
Τεχνολογία Πληροφορικής: Εταιρείες που αναπτύσσουν λογισμικό και παρέχουν υπηρεσίες πληροφορικής.

Χρηματοοικονομικοί Φορείς: Τράπεζες, χρηματοπιστωτικές εταιρείες και ασφαλιστικές εταιρείες που χρησιμοποιούν λογισμικό για τις χρηματοοικονομικές τους λειτουργίες.

Λιανικό Εμπόριο: Επιχειρήσεις που διαχειρίζονται διαδικτυακά καταστήματα (e-commerce) ή χρησιμοποιούν λογισμικό για τη διαχείριση αποθηκών, αποστολές, κλπ.  
  
Υγειονομική Περίθαλψη: Νοσοκομεία, ιατρικά κέντρα και φαρμακευτικές εταιρείες που χρησιμοποιούν λογισμικό για τη διαχείριση ασθενών, ιατρικών εγγραφών και άλλων λειτουργιών.

Κυβέρνηση: Κρατικοί φορείς και δημόσιες υπηρεσίες που χρησιμοποιούν λογισμικό για την υλοποίηση διαδικασιών και την παροχή δημόσιων υπηρεσιών.

***Ρόλος των μελών της ομάδας ανάπτυξης στο SDLC***

Ο ρόλος των μελών της ομάδας ανάπτυξης είναι κρίσιμος για την επιτυχία του Software Development Life Cycle (SDLC). Κάθε μέλος συμβάλλει με τις δεξιότητές του και την ειδίκευσή του σε κάθε φάση του SDLC, επιτρέποντας την ομαλή πρόοδο και την αποτελεσματική ολοκλήρωση του έργου. Κυριότεροι ρόλοι των μελών είναι οι εξής :  
  
Project Manager (Διευθυντής Έργου): Ο ρόλος του Project Manager είναι να διαχειρίζεται το έργο συνολικά. Αναλαμβάνει την προγραμματισμένη εκτέλεση των φάσεων του SDLC, τον προσδιορισμό των προθεσμιών και των στόχων, καθώς και την ανάθεση αρμοδιοτήτων στα μέλη της ομάδας.

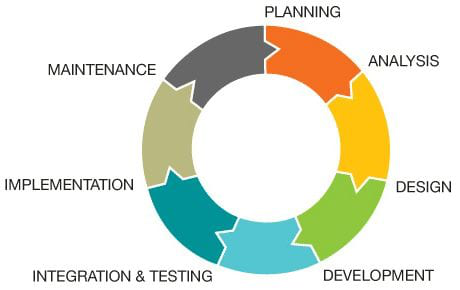
Business Analyst (Αναλυτής Επιχειρησιακών Απαιτήσεων): Ο Business Analyst είναι υπεύθυνος για την κατανόηση των αναγκών και απαιτήσεων του πελάτη ή του τελικού χρήστη. Συνεργάζεται με τους ενδιαφερόμενους φορείς για τη συλλογή, την ανάλυση και την καταγραφή των λειτουργικών και μη λειτουργικών απαιτήσεων του συστήματος.

Software Developer (Προγραμματιστής Λογισμικού): Οι προγραμματιστές είναι υπεύθυνοι για την υλοποίηση του λογισμικού σύμφωνα με τις προδιαγραφές και απαιτήσεις του έργου. Αναπτύσσουν τον κώδικα της εφαρμογής, επιλέγοντας τα κατάλληλα προγραμματιστικά frameworks και τεχνολογίες.

Quality Assurance (Αξιολόγηση Ποιότητας): Η ομάδα Quality Assurance είναι υπεύθυνη για την εκτέλεση δοκιμών και ελέγχων για τον έλεγχο της ποιότητας του λογισμικού. Βεβαιώνονται ότι το λογισμικό λειτουργεί σωστά, ανιχνεύοντας και διορθώνοντας τυχόν σφάλματα ή προβλήματα.

UI/UX Designer (Σχεδιαστής Χρήσης και Εμπειρίας Χρήστη): Ο σχεδιαστής UI/UX δημιουργεί τον σχεδιασμό της διεπαφής χρήστη (UI) και βελτιστοποιεί την εμπειρία του χρήστη (UX). Στόχος του είναι να δημιουργήσει μια ευχάριστη και λειτουργική εμπειρία για τους χρήστες.

Συνοψίζοντας, το SDLC αποτελεί μια κρίσιμη μεθοδολογία για την ανάπτυξη λογισμικού, προσφέροντας οφέλη σε οργανισμούς και επιχειρήσεις που επιδιώκουν την παραγωγή υψηλής ποιότητας προϊόντων. Με το SDLC, οι επιχειρήσεις μπορούν να επιτύχουν αποτελεσματική διαχείριση έργου, μείωση του ρίσκου και βελτίωση της ποιότητας του παραγόμενου λογισμικού. Στο επόμενο κεφάλαιο, θα επικεντρωθούμε στις διάφορες φάσεις του SDLC και τις λεπτομερείς διαδικασίες που ακολουθούνται σε κάθε μία από αυτές.

***2.2 SDLC Phases***

Εικόνα 1

*(https://digitalmarketing.temple.edu/kassijata/wp-content/uploads/sites/366/2021/07/System-Development-Life-Cycle-Phases.jpg)*

Το SDLC χωρίζεται σε 6 ή 7 στάδια (ανάλογα με τις ανάγκες του εκάστοτε έργου): Σχεδιασμός, Ανάλυση, Σχεδίαση, Κατασκευή, Δοκιμή, Κυκλοφορία, Συντήρηση.

**Στάδιο Σχεδιασμού (Planning)**

Στο στάδιο του σχεδιασμού, ξεκινά η διαδικασία προετοιμασίας για την επιτυχή ανάπτυξη του λογισμικού. Η ομάδα που εμπλέκεται στο έργο έχει ως βασικό στόχο να καθορίσει το πεδίο εφαρμογής του έργου, τους στόχους που πρέπει να επιτευχθούν και τις λύσεις που αναμένονται. Οι στόχοι αυτοί θα πρέπει να είναι σαφείς, ρεαλιστικοί με το πεδίο εφαρμογής και με την πραγματοποίηση τους καθώς και να συμφωνούν με τα χρονικά περιθώρια.

Στον στάδιο αυτό, πραγματοποιείται μια διεξοδική ανάλυση των αναγκών και απαιτήσεων των πελατών και των χρηστών του λογισμικού. Αναπτύσσεται εάν αποτελεσματικό πλάνο, που βοηθά στον εντοπισμό προβλημάτων, πριν αυτά επηρεάσουν την ανάπτυξη και επισημαίνονται οι λειτουργίες που πρέπει να υποστηριχθούν, οι τεχνολογίες που θα χρησιμοποιηθούν και οι περιορισμοί που πρέπει να ληφθούν υπόψη.

Μια σωστά προετοιμασμένη φάση σχεδιασμού εξασφαλίζει την σωστή χρηματοδότηση και τους πόρους που χρειάζονται για την ανάληψη του εκάστοτε συστήματος. Εξασφαλίζει επίσης την μεγαλύτερη αποδοτικότητα κατά την ανάπτυξη και μειώνει τον κίνδυνο εμφάνισης σφαλμάτων.

Θεωρείται από τα πιο σημαντικά στάδια, καθώς επίσης καθορίζει το χρονοδιάγραμμα ολοκλήρου του έργου. Ιδίως σε εμπορικές συμφωνίες και με την συνεχή αύξηση των απαιτήσεων των εταιρειών, εάν σωστό χρονοδιάγραμμα θα επιφέρει μικρότερο κόστος και θα μειωθεί ο χρόνος παράδοσης.

**Στάδιο Ανάλυσης (Analysis)**

Το στάδιο της ανάλυσης θεωρείται από τα πιο κρίσιμα στάδια καθώς τα μέλη του έργου καθορίζουν και συγκεντρώνουν όλες τις τεχνικές απαιτήσεις του συστήματος, τις εισροές και τις εκροές της ροής δεδομένων μέσα και έξω από το σύστημα και γενικότερα όλες τις προδιάγραφες του έργου, που απαιτούνται για το νέο σύστημα.

Πραγματοποιούνται συνεντεύξεις με τους ενδιαφερόμενους φορείς και συλλέγονται πληροφορίες για τις λειτουργίες που πρέπει να υποστηριχθούν, τα δεδομένα που θα αναλυθούν, και τους περιορισμούς που πρέπει να ληφθούν υπόψη. Έτσι εξασφαλίζεται η άμεση και σωστή αντιμετώπιση των απαιτήσεων και η ακριβής ανάλυση των επιχειρηματικών διαδικασιών που πρέπει να καλυφθούν και να επιλυθούν από το μελλοντικό σύστημα, με γνώμονα σαφώς τις απαιτήσεις του πελάτη.

Σε αυτό το στάδιο, τεκμηριώνονται όλες οι απαιτήσεις και οι προδιάγραφες του έργου, προσδιορίζονται απαιτήσεις υλικού και απαιτήσεις δικτύου, προτείνονται πρωτότυπες ιδέες και αξιολογούνται εναλλακτικές δοκιμές λύσεις. Σε αυτό το σημείο του έργου, οι προγραμματιστές δημιουργούν ένα Έγγραφο Προδιαγραφής Απαιτήσεων (Software Requirements Specification – SRS), στο οποίο περιγράφεται τι θα κάνει η εφαρμογή και πως αναμένεται να λειτουργήσει. Δημιουργούνται μοντέλα όπως οι λειτουργικές απαιτήσεις (functional requirements) και οι μη λειτουργικές απαιτήσεις (non-functional requirements), προκειμένου να περιγράψουν με σαφήνεια τι πρέπει να πετύχει το λογισμικό και ποια περιβάλλοντα πρέπει να υποστηρίζει.

Αυτή η φαση είναι στενά συνδεδεμένη με την αποφυγή υπερβολικής χρηματοδότησης ή πόρων σε σχέση με ομάδες αναπτυξης στον ίδιο τομέα, αρά και στην μείωση των πιθανοτήτων μη ανάληψης του έργου τελικά. Οπότε η ομάδα συνήθως χρειάζεται αλλά και αφιερώνει αρκετό χρόνο για να αναλύσει σωστά κάθε λεπτομέρεια του έργου

**Στάδιο Σχεδίασης (Design)**

Επόμενο στάδιο και από τα πιο απαραίτητα στάδια του SDLC, είναι το στάδιο της σχεδίασης. Τα μέλη του έργου, αφού έχουν καθορίσει σωστά τις απαιτήσεις και την τεκμηρίωση που υλοποίησαν στην φαση της ανάλυσης, καλούνται όλα αυτά τα στοιχεία, να τα χρησιμοποιήσουν για να αναπτύξουν την τεχνική, πλέον, τεκμηρίωση του έργου.

Στοιχεία όπως η αρχιτεκτονική του λογισμικού, οι διεπαφής του χρήστη, του συστήματος, το δίκτυο, οι απαιτήσεις του, η διαχείριση των δεδομένων αλλά και οι βάσεις τους, οι οροί και οι προϋποθέσεις μεταξύ των φάσεων, θα θέσουν τις βάσεις για το σχέδιο του έργου. Η διεπαφή θα πρέπει να είναι φιλική και εύχρηστη, ώστε οι χρήστες να μπορούν να αλληλοεπιδρούν με το λογισμικό χωρίς δυσκολία και προβλήματα, αλλά ταυτόχρονα να είναι σύμφωνη και με τις απαιτήσεις των πελατών. Καθορίζονται επίσης τα μετρά που θα ληφθούν για την ασφάλεια της εφαρμογής, καθώς και οι πλατφόρμες στις οποίες θα εκτελείται.

Όλα αυτά τα στοιχεία αναλύονται και παρουσιάζονται στο Έγγραφο Προδιαγραφής Απαιτήσεων Επιχειρησιακού Επιπέδου (Business Requiremens Specification – BRS), το οποίο είναι ένα επιχειρηματικό έγγραφο, που περιγράφει τις υψηλού επιπέδου απαιτήσεις από την οπτική γωνιά της επιχείρησης για το σύστημα που θα αναπτυχθεί. Περιλαμβάνει πληροφορίες σχετικά με τους σκοπούς, τους στόχους και τα οφέλη του συστήματος.

Με την ολοκλήρωση της καταγραφής, τα μέλη της ομάδας θα είναι σε θέση να γνωρίζουν τι πρέπει να κάνουν σε κάθε στάδιο του SDLC, αλλά δίνει την δυνατότητα και στην άλλη πλευρά. αυτήν του πελάτη, να κατανοήσει τα σχέδια λειτουργίας, Το έγγραφο SRS που δημιουργήθηκε στην προηγουμένη φαση, θα μετατραπεί σε μια πιο απλή γλώσσα, ούτως ώστε να μπορεί να μετατραπεί σε μια γλώσσα προγραμματισμού. Επίσης, στο στάδιο του σχεδιασμού, οι ομάδες συνηθίζουν να κατασκευάζουν διαγράμματα UML, για να αποδώσουν πιο κατανοητά την αρχιτεκτονική του συστήματος.

**Στάδιο Ανάπτυξης (Development)**

Μια από τις πιο χρονοβόρες αλλά ενδιαφέρουσες, ταυτόχρονα. φάσεις ενός έργου είναι αυτή της κατασκευής. Είναι το στάδιο όπου οι προγραμματιστές γράφουν πραγματικό κώδικα και κατασκευάζουν την εφαρμογή σύμφωνα με τα έγγραφα σχεδιασμού και τις περιγραφόμενες προδιαγραφές, που έχουν αναλυθεί στα προηγούμενα στάδια. Διαμορφώνονται τα cloud συστήματα, τα πρωτοκολλά και το υλικοτεχνικό κομμάτι, εκτός της σύνταξης του κώδικα, για αυτό και αυτό το στάδιο είναι τόσο σημαντικό.

Τα διαγράμματα UML, που κατασκευάστηκαν νωρίτερα, κατέχουν σημαντικό ρολό στην έγκαιρη και ορθή εκπλήρωση αυτού του σταδίου. Λανθασμένα διαγράμματα θα καθυστερούσαν συνολικά το έργο, δεδομένο που δείχνει ακόμα πιο έντονα την αναγκαιότητα της όσο πιο σωστής ολοκλήρωσης της κάθε φάσης.

Παράλληλα με τον κώδικα, οι προγραμματιστές δημιουργούν ένα λεπτομερές έγγραφο για τον κώδικα τους, γνωστό και ως Έγγραφο Σχέδιου Αποδοχής (Acceptance Test Plan – ATP), που περιγράφει πώς θα δοκιμαστεί το σύστημα λογισμικού για να διασφαλιστεί ότι πληροί τις απαιτήσεις που καθορίζονται στα προηγούμενα δυο έγγραφα, το SRS και το BRS. Αυτή η τεκμηρίωση βοηθά άλλα μέλη της ομάδας να κατανοήσουν τον κώδικα, τον σκοπό του, τη χρήση του και τυχόν εξαρτήσεις. Η κατάλληλη τεκμηρίωση είναι ζωτικής σημασίας για τη μελλοντική συντήρηση και ενημέρωση, αλλά και για την φαση της δοκιμής στο επόμενο στάδιο.

Οι προγραμματιστές ακολουθούν συγκεκριμένες οδηγίες και χρησιμοποιήσουν συγκεκριμένα εργαλεία και compilers, debuggers όσον αφορά τον κώδικα, με βάση τις ανάγκες που έχουν συμφωνηθεί σε προηγούμενο στάδιο από τον πελάτη.

**Στάδιο Δοκιμής (Testing)**

Σε αυτή τη φάση, το λογισμικό που έχει αναπτυχθεί ελέγχεται διεξοδικά, ξανά και ξανά, για τυχόν ελαττώματα και σφάλματα, έως ότου τα αποτελέσματα ταιριάζουν με το αναμενόμενο αποτέλεσμα. Αν υπάρχουν και διαπιστωθούν, ανατίθενται στους προγραμματιστές για να κάνουν εκ νέου διορθώσεις και στην συνέχεια επανεξετάζονται, μέχρις ότου το λογισμικό να είναι σύμφωνο με τις προσδοκίες του πελάτη και η εμπειρία του τελικού χρήστη να μην επηρεαστεί αρνητικά σε κανένα σημείο.

Η δοκιμή πρέπει να διασφαλίζει ότι κάθε λειτουργία και διαφορετικά μέρη της εφαρμογής λειτουργούν ταυτόχρονα σωστά, με στόχο να μειωθούν τυχόν καθυστερήσεις και δυσλειτουργίες αλλά και να βελτιστοποιηθεί το ποσοστό χρήσης.

Υπάρχουν διάφοροι τύποι δοκιμών κατά τη φάση της δοκιμής, πιο γνωστοί από αυτούς είναι των δοκιμών διασφάλισης ποιότητας (QA), δοκιμών ολοκλήρωσης συστήματος (SIT) και δοκιμών αποδοχής χρήστη (UAT).

**STLC (Software Testing Life Cycle)**

Το STLC αναφέρεται στη διαδικασία που χρησιμοποιείται για τις δοκιμές λογισμικών και περιγράφει τη σειρά των βημάτων που πρέπει να ακολουθήσει η ομάδα δοκίμων για να διασφαλιστεί η ποιότητα του λογισμικού.

Κάθε επιτυχημένο SDLC περιλαμβάνει τη φάση των δοκιμών, και για τον λόγο αυτό, το STLC θεωρείται ως ένα υποσύνολο του SDLC και συνδέεται άρρηκτα με τις άλλες φάσεις της αναπτυξης λογισμικού. Καθορίζει τις διαδικασίες, τις τεχνικές και τα εργαλεία που χρησιμοποιούνται για την εκτέλεση των δοκιμών και τον έλεγχο της ποιότητας του λογισμικού πριν από την παράδοση του στον πελάτη. Οι βασικές φάσεις του STLC περιλαμβάνουν τα εξής:

* Σχεδιασμός των δοκιμών: Σε αυτό το στάδιο ορίζονται οι στόχοι και το σχέδιο των δοκιμών και δημιουργούνται τα σενάρια δοκίμων. Καθορίζεται η στρατηγική δοκιμών, ο σχεδιασμός των σεναρίων, και ορίζονται οι πόροι που θα χρησιμοποιηθούν.
* Ανάπτυξη των δοκιμών: Σε αυτή την φαση των δοκίμων, δημιουργούνται οι δοκιμαστικοί κώδικες και τα σενάρια δοκίμων και ετοιμάζονται τα δεδομένα που θα χρησιμοποιηθούν κατά την δοκιμή
* Ανάπτυξη των σεναρίων :
* Εκτέλεση των δοκιμών: Οι δοκιμαστικοί κώδικες και τα προγραμματισμένα σενάρια εκτελούνται και καταγράφονται τα αποτελέσματα. Εάν στα αποτελέσματα ανιχνευθούν σφάλματα, τότε αναφέρονται και παρακολουθείται η διαδικασία επιδιόρθωσής τους.
* Αναφορά Σφαλμάτων: Οι προγραμματιστές αναλύουν τα σφάλματα της προηγουμένης φάσης και προχωρούν στην επιδιόρθωση τους. Οι επιδιορθωμένοι κώδικες επαναδοκιμάζονται με σκοπό την επαλήθευση της επιτυχούς επίλυσης των σφαλμάτων.

* Αναθεώρηση δοκιμών: Σε αυτό το στάδιο περιλαμβάνεται η αναθεώρηση των αποτελεσμάτων των δοκίμων και δημιουργείται μια συνοπτική αν απτική αναφορά για αυτά. Ελέγχεται η ποιότητα του λογισμικού και τυχόν προβλήματα σημειώνονται για επίλυση σε επόμενη φαση. Ο πελάτης αξιολογεί με την σειρά του το λογισμικό και αποφασίζει εάν είναι έτοιμο για παράδοση ή όχι.

Με την εφαρμογή του STLC στο στάδιο της δοκιμής, επιτυγχάνεται η ασφάλεια και η αξιοπιστία, μειώνοντας την εμφάνιση κρίσιμων σφαλμάτων και βελτιώνοντας συνολικά την ποιότητα του λογισμικού και την εμπειρία των χρηστών. Επιτυγχάνεται επίσης εξοικονόμηση χρόνου και πόρων από την έγκαιρη ανίχνευση και διόρθωση των σφαλμάτων, διασφαλίζοντας την ευελιξία και την λειτουργικότητα, αφού είναι προσαρμοσμένο στις ανάγκες των πελατών. Μέσω της συμμόρφωσης και της πλήρης κατανόησης των προδιαγραφών, των αναγκών και των κανονισμών που έχουν τεθεί, επιτυγχάνεται η αξιοπιστία του φορέα και η εμπιστοσύνη των πελατών και θέτονται θεμέλια μακροχρόνιας συνεργασίας.  
Συνεπώς, η διαδικασία του STLC είναι ένα απαραίτητο εργαλείο για την εξασφάλιση υψηλής ποιότητας λογισμικού και απαραίτητο μέρος του συνολικά του SDLC.

Η φαση της δοκιμής είναι κρίσιμη, καθώς αποτελεί την τελευταία φαση πριν διατεθεί το λογισμικό στον τελικό χρήστη, διασφαλίζοντας ταυτόχρονα ότι θα πληροί τα κριτήρια ποιότητας που είχαν οριστεί και συμφωνηθεί εγγράφως στα προηγούμενα στάδια.

Αναλόγως λοιπόν, με την πολυπλοκότητα του λογισμικού, τις απαιτήσεις του τελικού χρήστη, τον αριθμό των σφαλμάτων αλλά και τις ικανότητες των ομάδων αναπτυξης και δοκίμων, το στάδιο της δοκιμής μπορεί είτε να είναι μια σύντομη διαδικασία , είτε να αποδειχθεί πολύ χρονοβόρα τελικά.

**Στάδιο Εφαρμογής/Υλοποίησης (Implementation)**

Αφού τελειώσει η φαση της δοκιμής, σειρά έχει αυτή της εφαρμογής, της διάθεσης δηλαδή του λογισμικού στον χρήστη. Το λογισμικό αφού εγκατασταθεί και ενσωματωθεί στο περιβάλλον λειτουργίας του από τους προγραμματιστές και δοθούν οι οδηγίες χρήσης στους χρήστες, θα είναι έτοιμο για χρήση.  
Για αυτό τον λόγο, αυτό το στάδιο περιλαμβάνει την εκπαίδευση και την υποστήριξη των χρηστών πριν η και κατά την διάρκεια λειτουργίας του λογισμικού.

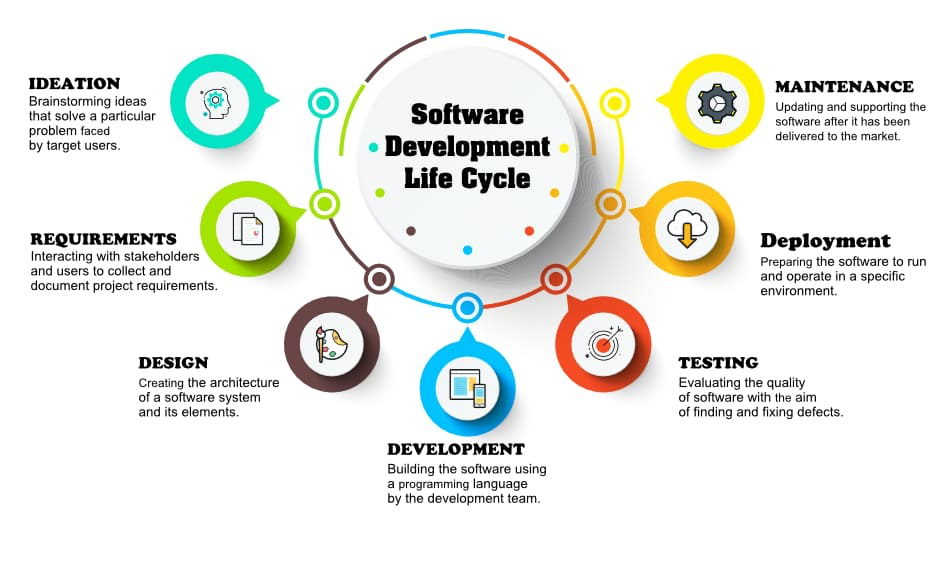
Ωστόσο, ακόμα και μετά την κυκλοφορία της εφαρμογής, πραγματοποιείται ακόμα ένας μικρός κύκλος δοκίμων, αν αυτό κρίνεται απαραίτητο, κυρίως για επίλυση θεμάτων συμβατότητας η παρατηρήσεων από τους χρήστες, για την βελτίωση του λογισμικού

**Στάδιο Συντήρησης (Maintenance)**

Η φαση της συντήρησης είναι πολύ σημαντική για την εξέλιξη και την αδιάλειπτη λειτουργία της νέας εφαρμογής, καθώς κύριος στόχος αυτής της φάσης είναι να διασφαλίσει ότι οι ανάγκες συνεχίζουν να ικανοποιούνται και ότι το σύστημα συνεχίζει να λειτουργεί σύμφωνα με τις προδιαγραφές που αναφέρονται στην πρώτη φάση.

Ακόμα όμως και να ικανοποιούνται πλήρως οι ανάγκες, πέραν της διαχείρισης και της διόρθωσης σφαλμάτων, οι προγραμματιστές είναι υπεύθυνοι για την τροποποίηση , τον σχεδιασμό προσθέτων δυνατοτήτων αλλά και των αναβαθμίσεων σε νεότερες βελτιστοποιημένες εκδόσεις.   
Κατά τη φάση συντήρησης, παρέχονται υπηρεσίες υποστήριξης και προστασίας του λογισμικού από πιθανά σφάλματα ή προβλήματα. Οι πελάτες μπορούν να ζητήσουν και να αναπτύξουν νέες λειτουργίες ή να προσαρμόσουν το λογισμικό με βάση τις εξελίξεις της αγοράς και τις ανάγκες τους. Η συντήρηση και ανάπτυξη του λογισμικού εξασφαλίζει ότι η εφαρμογή παραμένει χρήσιμη και λειτουργική για τους τελικούς χρήστες.

Όπως είναι λογικό, όλες αυτές οι εργασίες και ιδιαίτερα αυτή της συνεχής αναβάθμισης μπορεί να δημιουργούν νέους κύκλους αναπτυξης, οπότε ουσιαστικά αυτό το στάδιο, θα συνεχίζει να είναι σε λειτουργία και να απασχολεί τους προγραμματιστές όσο η εφαρμογή είναι και θα συνεχίσει να είναι διαθέσιμη προς πώληση η προς οποία γενικότερη εκμετάλλευση.



(https://www.ntaskmanager.com/wp-content)

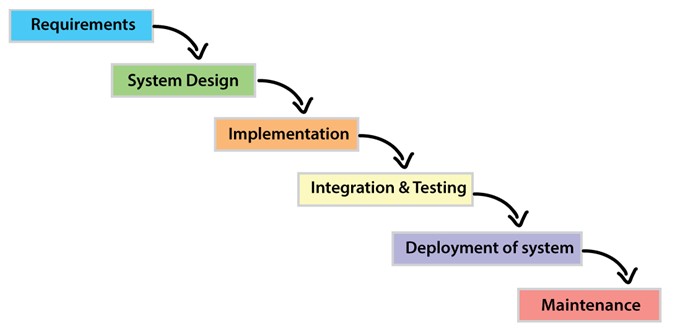
***2.3 SDLC Models & Methodologies***

Η μέθοδος SDLC χρησιμοποιεί πολλά διαφορετικά μοντέλα που καθορίζουν και σχεδιάζουν την διαδικασία της αναπτυξης, τα οποία ακολουθούνται σε όλη την φάση της. Αυτά τα μοντέλα ονομάζονται επίσης "Μοντέλα Διαδικασίας Ανάπτυξης Λογισμικού". Κάθε μοντέλο διαδικασίας ακολουθεί μια σειρά συγκεκριμένων φάσεων ανάλογα τον τύπο του με σκοπό να εξασφαλίσει την επιτυχία του εκάστοτε βήματος της ανάπτυξης λογισμικού.

Τα πιο σημαντικά από τα μοντέλα είναι τα εξής :

Waterfall Model, V-Shaped Model, Iterative Model, Spiral Model, Agile Model, RAD Model, Big Bang Model κτλ. Παρακάτω θα αναλυθούν μερικά από αυτά.

* + 1. **Waterfall Model**



Το μοντέλο του “καταρράκτη” είναι το πιο παλιό και διαδεδομένο μοντέλο διαδικασίας αναπτυξης λογισμικού καθώς και το πιο ευθύ από όλα τα υπόλοιπα μοντέλα. Η μέθοδος του συγκεκριμένου μοντέλου είναι απλή και βασίζεται στην διαίρεση του έργου σε διάφορες φάσεις. Κάθε φαση έχει το δικό της και κάθε φαση, αφού τελειώσει, διαδέχεται την επόμενη. Σαν τον καταρράκτη, έτσι και αυτή η μέθοδος φαίνεται να ρέει σταθερά προς τα κάτω, μέσω των φάσεων και προτιμάται σε έργα οπού το κόστος και χρόνος δεν είναι τόσο σημαντικά όσο η ποιότητα. Με αυτό το μοντέλο διασφαλίζεται η συνέπεια του κάθε σταδίου και κατ’ επέκταση την συνέπεια ολόκληρου του συστήματος, αφού κάθε στάδιο επαληθεύεται πριν γίνει η μετάβαση στο επόμενο.

Πλεονεκτήματα αυτού του μοντέλου είναι η απλότητα που προσφέρει, οπότε επιτυγχάνεται η σταθερότητα ως προς τον σχεδιασμό, τον κώδικα, και τα παραδοτέα σε κάθε φαση. Επιτυγχάνεται ακόμα η ελαχιστοποίηση των απαιτούμενων πόρων σε σύγκριση με άλλες μεθόδους και η σαφήνεια του έργου ως προς τα μέλη της ομάδας αλλά και τους πελάτες.

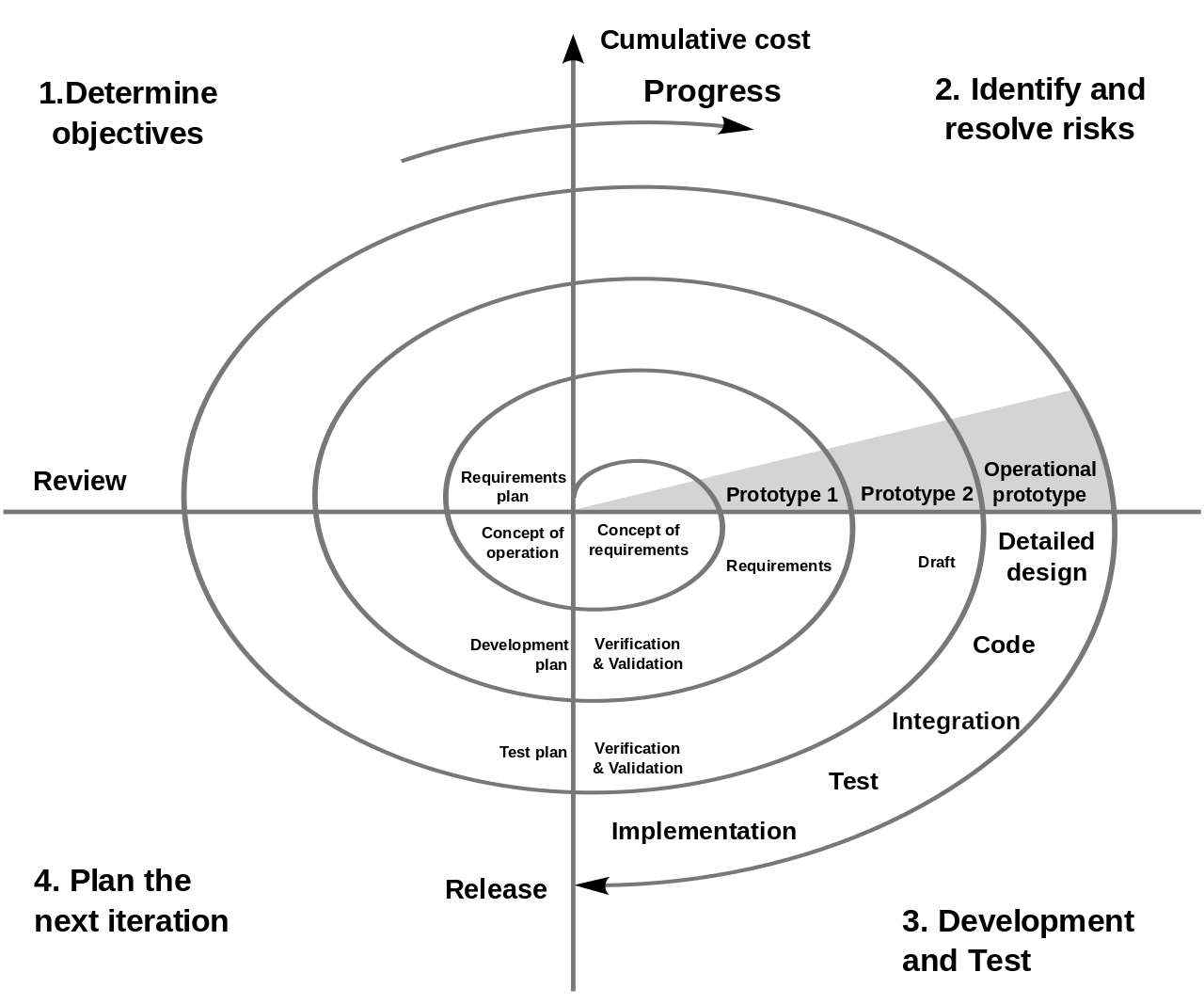
Το waterfall είναι ένα γραμμικό μοντέλο ανάπτυξης λογισμικού, που σημαίνει ότι κάθε στάδιο πρέπει να ολοκληρωθεί πριν ξεκινήσει το επόμενο. Λόγω της σταθερότητας όμως που προσφερει το μοντέλο του “καταρράκτη”, αποτυγχάνεται η ευελιξία. Χάνεται η δυνατότητα επιστροφής και αλλαγής απαιτήσεων (από το στάδιο του σχεδιασμού και έπειτα), αφού μια τέτοια αλλαγή θα οδηγούσε σε αναπήδηση προηγουμένων σταδίων, οπότε και σε ελαττώματα στον σχεδιασμό και στον κώδικα . Σφάλματα, που οι προγραμματιστές αναγνωρίζουν δηλαδή, συνεχίζουν να υπάρχουν μέχρι το τέλος του έργου οδηγώντας σε νέες εργασίες επεκτείνοντας και άλλο τον χρόνο αναμονής για τον πελάτη. Επίσης η ομάδα δοκίμων εμπλέκεται μόνο στην φαση των δοκίμων και όχι σε όλο το έργο συνολικά. Ο πελάτης, τελικά, δεν θα μείνει ικανοποιημένος, είτε λόγω των χρονικών περιθωρίων, είτε από το τελικό αποτέλεσμα, αφού αν θέλει να είναι εντός χρονικών περιθωρίων, θα πρέπει να συμβιβαστεί με την μη προσθήκη των αλλαγών στο τελικό προϊόν.

Εν κατακλείδι, το μοντέλο του καταρράκτη, προτείνεται για έργα με σταθερές και γνωστές απαιτήσεις και όχι για μακροπρόθεσμα έργα οπού οι απαιτήσεις μπορεί να αλλάζουν συχνά.

**2.3.2 Spiral Model**

Το μοντέλο Spiral αποτελεί μια προηγμένη μέθοδο ανάπτυξης που ενσωματώνει τα θετικά στοιχεία του μοντέλου Waterfall και του μοντέλου Iterative. Το σπειροειδές μοντέλο συνδυάζει το γραμμικό μοντέλο, που χωρίζει τη διαδικασία ανάπτυξης σε μια σειρά από διαδοχικές φάσεις και κάθε φάση ολοκληρώνεται πριν ξεκινήσει η επόμενη και το επαναληπτικό μοντέλο, που χωρίζει τη διαδικασία ανάπτυξης σε μια σειρά από επαναλήψεις και κάθε επανάληψη περιλαμβάνει ένα μικρό σύνολο εργασιών που ολοκληρώνονται πριν ξεκινήσει η επόμενη επανάληψη. Αυτη η επαναλαμβανόμενη διαδικασία ανάπτυξης, επιτρέπει την προσαρμογή στις μεταβαλλόμενες απαιτήσεις, με κάθε επανάληψη να ονομάζεται "Σπείρα". Το ονομα προκυπτει από το γεγονος ότι η γραφική αναπαράσταση του μοντέλου, μοιάζει με σπείρα με πολλές στροφές, οπου ο ακριβής αριθμός των στροφών της σπείρας είναι άγνωστος και μπορεί να διαφέρει από έργο σε έργο. Κάθε στροφή της σπείρας ονομάζεται Φάση της διαδικασίας ανάπτυξης λογισμικού και αντιπροσωπεύει έναν πλήρη κύκλο ανάπτυξης λογισμικού, από την συλλογή και ανάλυση απαιτήσεων έως το σχεδιασμό, την υλοποίηση, τη δοκιμή και τη συντήρηση.

Ο ακριβής αριθμός των φάσεων που απαιτούνται για την ανάπτυξη του προϊόντος μπορεί να ποικίλλει κυριως επειδη το μοντελο αυτό δινει εμφαση στον κινδύνο, δεδομένου κιολας ότι ο project manager προσδιορίζει δυναμικά τον αριθμό των φάσεων. ο project manager έχει σημαντικό ρόλο στην ανάπτυξη ενός προϊόντος στο μοντέλο Spiral. Κάθε σπείρα περιλαμβάνει τέσσερις κύριες φάσεις: Σχεδιασμό, Ανάλυση Κινδύνου, Υλοποίηση, Αξιολόγηση.



Στην αρχική φάση του Spiral, καθορίζονται οι αρχικές απαιτήσεις και πραγματοποιείται ένας αρχικός σχεδιασμός. Στη συνέχεια, γίνεται η ανάλυση κινδύνων, προσδιορίζονται οι κρίσιμες περιοχές του συστήματος και προτεινονται πιθανες λυσεις για την καλυτερη προσεγγιση. Στη φάση της υλοποίησης, ξεκινάει ο πραγματικός προγραμματισμός του συστήματος. Μετα την φαση της υλοποίησης, η επόμενη στροφή ξεκινάει με μια νέα φαση σχεδιασμού, η μια νέα σπείρα, με βάση τα αποτελέσματα της αξιολόγησης.  
  
Στο μοντελο spiral σε μορφη διαγραμματος, το κοστος του project είναι αναλογο της ακτίνας της σπείρας σε οποιοδήποτε σημείο και η γωνια που σχηματιζεται αντιπροσωπεύει την πρόοδο που έχει σημειωθεί μέχρι στιγμής στην τρέχουσα φάση.

Κάθε φάση του μοντέλου Spiral είναι χωρισμένη σε τέσσερις κατηγορίες, όπως φαίνεται στο παραπάνω διάγραμμα. Οι λειτουργίες αυτών των τεσσάρων κατηγοριών είναι οι εξής:

1. Καθορισμός στόχων και εναλλακτικών λύσεων : Καθορίζονται και συγκεντρώνονται οι απαιτήσεις των πελατών.
2. Αναγνώριση και επίλυση κινδύνων: Αξιολογούνται όλες οι πιθανές λύσεις και αναγνωρίζονται οι κίνδυνοι που συνδέονται με αυτές τις λύσεις και επιλύονται μέσω της καλύτερης δυνατής στρατηγικής, για να επιλεγεί η καλύτερη δυνατή λύση.
3. Υλοποίηση και δοκιμή νέας έκδοσης : Αναπτύσσεται η νέα έκδοση της εφαρμογής, βαση των απαιτησεων που συλλεχθηκαν στην προηγουμενη επαναληψη, με όσο το δυνατόν καλύτερη λύση και γινεται επαληθεύεται η ορθότητα των αλλαγών.
4. Ανασκόπηση και σχεδιασμός για την επόμενη φάση : Οι πελάτες αξιολογούν την μέχρι τώρα ανεπτυγμένη έκδοση του λογισμικού προκειμενου να δουν αν τους ικανοποιει, η επαναληψη τελειωνει και αν κριθει απαραιτητο ξεκινάει η φαση σχεδιασμου για την επόμενη επαναληψη

Ένα από τα σημαντικότερα χαρακτηριστικά του μοντέλου Spiral είναι η διαρκής αξιολόγηση και αναθεώρηση των κινδύνων κατά τη διάρκεια της ανάπτυξης. Ενθαρρύνεται η επικοινωνία μεταξύ πελάτη και ομάδων αναπτυξης, βοηθώντας στην βελτίωση της κατανόησης των απαιτήσεων, επιτρέποντας στις ομάδες να προβλέπουν πιθανά προβλήματα και να λαμβάνουν μέτρα πρόληψης, μειώνοντας έτσι τον κίνδυνο αποτυχίας του έργου συνολικά. Για αυτό το λόγο χρησιμοποιείται συχνότερα σε πιο μεγάλα και σύνθετα έργα αναπτυξης λογισμικού καθώς επιτρέπει μια πιο ευέλικτη και προσαρμοστική προσέγγιση δεκτικη στις αλλαγες, με υψηλοτέρα ίσως επίπεδα κίνδυνου, αλλά από την άλλη υψηλότερες απαιτήσεις ποιότητας .  
  
Από την άλλη, το μοντελο spiral θεωρειται από τα πιο περιπλοκα μοντελα, με σημαντικότερη να είναι η δυσκολία διαχείρισης χρόνου και πόρων λόγω του αγνώστου αριθμού φάσεων στις αρχές του έργου, που μπορεί να αποβεί δαπανηρό τελικά και ιδίως για μικρότερα έργα. Το μοντελο αυτό βασιζεται εντονα στην αναλυση του ρισκου, γεγονος που απαιτει πολύ καλους ειδκους, ρποκειμενου να μην αποτυχει το συνολικο εργο, οποτε εξαρταται και από την εκαστοτε ομαδα.  
  
Συνοψίζοντας, το μοντέλο Spiral είναι ένα ευέλικτο και προσαρμοστικό μοντέλο ανάπτυξης λογισμικού που μπορεί να χρησιμοποιηθεί για έργα με υψηλό επίπεδο αβεβαιότητας ή κινδύνου. Ενθαρρύνει την επικοινωνία, τη συνεργασία και την έγκαιρη λήψη αποφάσεων. Το μοντέλο αυτό μπορεί να είναι πιο ακριβό και χρονοβόρο από άλλα μοντέλα ανάπτυξης λογισμικού, αλλά μπορεί να βοηθήσει στη μείωση του κινδύνου αποτυχίας του έργου και στη βελτίωση της ποιότητας του τελικού προϊόντος. Περιλαμβάνει τα πιο γνωστά μοντέλα του SDLC, όπως τα μοντέλα Waterfall, Spiral και Iterative.

**2.3.3 Agile Model**

Η ταχεία εξέλιξη των τεχνολογιών και η πολυπλοκότητα των προς επίλυση ζητημάτων καθιστούν δύσκολο τον ακριβή προσδιορισμό των απαιτήσεων για την υλοποίηση προγραμματιστικων εργων, οποτε η υιοθέτηση μοντελων όπως το waterfall δεν ενθαρρύνουν την επικοινωνία και την συνεργασία των εμπλεκόμενων μερών. Σε τετοιες περιπτωσεις το μοντέλο Agile προσαρμοζεται καλυτερα στις αναγκες του project γιατι επικεντρώνεται στην ικανοποίηση του πελάτη και στην γρήγορη παράδοση του συστήματος. Η συνεργασία όλων των εμπλεκόμενων μερών είναι απαραίτητη για την κατανόηση των πραγματικών αναγκών των πελατών και των επιχειρήσεων.

Η ιδεα πισω από την μεθοδο Agile, είναι στην υλοποιηση της σε επαναληπτικους κυκλους χωρίζοντας το σε μικρά, συνεχή επαναλαμβανόμενα βήματα, γνωστά ως "επαναλήψεις" ή "σπριντ" και συνήθως διαρκούν από μια έως τρεις βδομάδες. Σε καθεμιά από αυτές τις επαναλήψεις, εργάζονται ταυτόχρονα αρκετές ομάδες μεταξύ τους σε τομείς όπως την ανάλυση απαιτήσεων, την σχεδίαση, τον κώδικα κλπ., και τους αντιμετωπίζουν διαφορετικά από έργο σε έργο. Οι απαιτήσεις κάθε έργου είναι διαφορετικές και το μοντέλο Agile προσαρμόζει την μέθοδο αναπτυξης στις εκάστοτε ανάγκες. Το μοντέλο αυτό εφαρμόζει, επίσης, επαναληπτική προσέγγιση, έως ότου, μέσα από όλες τις σταδιακές επαναλήψεις, το έργο να καταλήξει στην τελική μορφή με τα όλα τα χαρακτηριστικά που εξυπηρετούν τον πελάτη. Η συμφωνία των εμπλεκόμενων μερών για το αντικείμενο που θα υλοποιηθεί στον επόμενο κύκλο είναι απαραίτητη για την εξασφάλιση της συνοχής του έργου. Η συμφωνία αυτή πρέπει να βασίζεται στα αποτελέσματα των ενδιάμεσων ελέγχων και στις ανάγκες των τελικών χρηστών. Οι εργασίες χωρίζονται σε μικρά χρονικά πλαίσια, έτσι ώστε σε κάθε πλαίσιο να παραδίδεται συγκεκριμένο χαρακτηριστικό, παρέχοντας απλότητα και την ύπαρξη λιγοστών κανόνων στην ανάπτυξη.

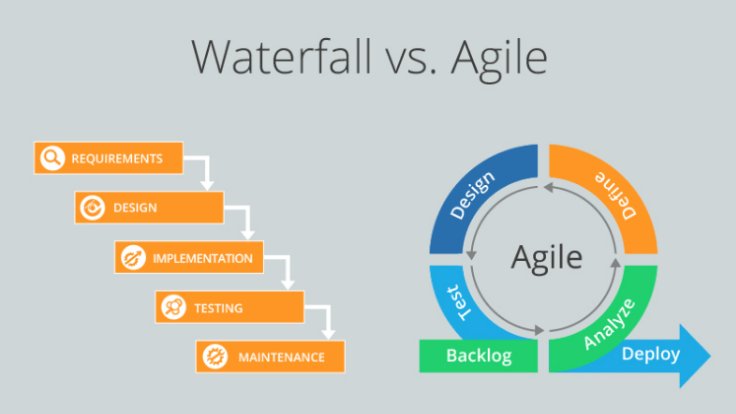
Οι κύριες αρχές του Agile είναι η επικοινωνία και η συνεργασία μεταξύ των μελών της ομάδας, η ανταπόκριση σε αλλαγές και η υποβολη συχνών, λειτουργικών παραδόσεων. Οι βασικές φάσεις του μοντέλου Agile περιλαμβάνουν:

1. Σχεδιασμός και καθορισμός απαιτήσεων: Καθορίζονται οι αρχικές απαιτήσεις και δημιουργείται ένα αρχικό σχέδιο για την ανάπτυξη.
2. Επαναλήψεις (Sprints): Η ανάπτυξη γίνεται σε μικρά, προκαθορισμένα χρονικά διαστήματα, με την ομάδα να παραδίδει λειτουργικό λογισμικό σε κάθε επανάληψη.
3. Δοκιμές και αξιολόγηση: Κάθε παράδοση δοκιμάζεται και αξιολογείται από την ομάδα και τους πελάτες.
4. Επανεκτίμηση και προσαρμογή: Μετά από κάθε επανάληψη, η ομάδα επανεκτιμά τις απαιτήσεις και προσαρμόζει τον σχεδιασμό ανάλογα.

Γενικότερα, η μέθοδος Agile γίνεται όλο και πιο διαδεδομένη στο χώρο της αναπτυξης λογισμικού, αφού σαν μοντέλο δεν απαιτεί πολλούς πόρους και είναι εύκολο στην διαχείριση. Οι προγραμματιστές του έργου είναι πιο ευέλικτοι, καθώς το μοντέλο τους επιτρέπει να εργάζονται σε έργα που οι απαιτήσεις μεταβάλλονται, ελαχιστοποιώντας τον γενικό σχεδιασμό και προσφέροντας μια γρήγορη και ρεαλιστική εικόνα του λογισμικού εκ των προτέρων. Επίσης λόγω της ταυτόχρονης αναπτυξης και υλοποίησης των σταδίων, προωθείται η ομαδικότητα μεταξύ των μελών και των ομάδων.

Βέβαια όπως και σε κάθε άλλο μοντέλο, το μοντέλο Agile έχει και αυτό κάποια μειονεκτήματα. Η υλοποίηση του έργου εξαρτάται σε μεγάλο βαθμό από τον πελάτη, ο οποίος πρέπει να είναι συγκεκριμένος ως προς τις απαιτήσεις, για να μην οδηγηθεί η ομάδα σε λάθος συμπεράσματα. Λόγω όμως τις απλότητας και των λιγοστών κανόνων του μοντέλου, το έργο εξαρτάται αρκετά και από τους εργαζομένους σε ατομικό επίπεδο, επειδη μπορει να είναι δύσκολο να διαχειριστεί κανείς τις συχνες αλλαγές πράγμα που μπορεί να δημιουργήσει αστοχίες στο τελικό προϊόν. Το κοστος καθως και το χρονοδιαγραμμα του εργου είναι πιο δυσκολο να εκτιμηθουν αφου οι επαναληψεις που θα γινουν και η εργασια από την ομαδα να αυξηθουν, ειδικα αν το εργο είναι μεγαλο ή περιπλοκο.

Συνοψίζοντας, η μέθοδος Agile δεν βασίζεται στην γραμμικη προσεγγιση όπως οι πιο παραδοσιακές μέθοδοι (Waterfall, V-Shaped, κλπ.), αλλά χρησιμοποιεί μια πιο προσαρμοστική προσέγγιση. όπου δεν υπάρχει λεπτομερής σχεδιασμός και υπάρχει σαφήνεια σχετικά με τις μελλοντικές εργασίες μόνο σε σχέση με τα χαρακτηριστικά που πρέπει να αναπτυχθούν. Υπάρχει ανάπτυξη με γνώμονα τα χαρακτηριστικά και η ομάδα προσαρμόζεται δυναμικά στις μεταβαλλόμενες απαιτήσεις του προϊόντος. Το προϊόν δοκιμάζεται πολύ συχνά, μέσω των επαναλήψεων απελευθέρωσης, ελαχιστοποιώντας τον κίνδυνο οποιωνδήποτε σοβαρών αστοχιών στο μέλλον.

[](https://i0.wp.com/aayushjain.com/wp-content/uploads/2020/05/agile-vs-waterfall.png?resize=736%2C414&ssl=1)

**2.3.4 V-Shaped Model**

Το μοντέλο V, γνωστό και ως μοντέλο επαλήθευσης και επικύρωσης (V&V) ακολουθεί μια διαδικασία σχεδιασμού όπως το μοντέλο καταρράκτη, καθως κάθε φάση του μοντελου πρέπει να ολοκληρωθεί πριν ξεκινήσει η επόμενη φάση, με την διάφορα ότι απαλείφει τα μειονεκτήματα του δευτέρου. Σε αυτό το μοντέλο η ομάδα δοκιμής συμμετέχει από τις πρώτες φάσεις του έργου, διορθωνοντας και αποφεύγοντας σφάλματα που ενδεχομένως θα προεκύπταν, μειώνοντας έτσι τον χρόνο εργασίας στο στάδιο δοκίμων. Η ιδέα είναι ότι από την μια πλευρα του “V” βρισκονται οι φασεις επαληθευσης και από την άλλη πλευρα οι φασεις επικυρωσης, και μια προς μια συνδεονται παραλληλα μεσω των δοκιμών, ώστε να διασφαλίζεται ότι κάθε απαίτηση του αναπτυσσόμενου συστήματος ελέγχεται σε κάθε στάδιο των δοκιμών και κάθε στάδιο ανάπτυξης (development) συσχετίζεται με το αντίστοιχο στάδιο δοκιμών (testing), σχηματίζοντας "την βαση του V".

Τα παραδοτέα εκτελούνται παράλληλα καθώς ενώ, η ομάδα προγραμματισμού εργάζεται με τα έγγραφα απαιτήσεων συστήματος SRS (System Requirement Specification), η ομάδα δοκίμων εργάζεται με τα επιχειρησιακα έγγραφα απαιτησεων BRS (Business Requirement Specification) και προετοιμάζουν το σχέδιο δοκιμής αποδοχής ATP (Acceptance Test Plan) και τα σενάρια δοκιμής αποδοχής ATC (Acceptance Test Cases) και ουτω καθεξης. Από την στιγμη που οι developers παραδωσουν το τελικο προιον, οι testers θα αμεσα έτοιμοι να αναλαβουν τις δοκιμες με όλα τα απαιτούμενα πλανα (όπως το σχέδιο δοκιμής, τα test cases).

Επαλήθευση (Validation) : Η φάση επαλήθευσης αναφέρεται στην πρακτική αξιολόγησης της διαδικασίας ανάπτυξης προϊόντος για να διασφαλιστεί ότι η ομάδα πληροί τις καθορισμένες απαιτήσεις.

Επικύρωση (Validation): Η επικύρωση είναι η διαδικασία αξιολογησης του λογισμικού μετά την ολοκλήρωση της διαδικασίας ανάπτυξης, για να προσδιοριστεί εάν το λογισμικό ανταποκρίνεται στις προσδοκίες και τις απαιτήσεις των πελατών.

Τα σταδια της επαληθευσης είναι τα εξης :

Ανάλυση απαιτήσεων πελατη (Business requirement analysis): Στο πρωτο βημα γινεται η πρωτη επαφη με τον πελατη και γινεται λεπτομερης συζητηση σχετικα με τις απιτησεις του προιοντος. Είναι μια πολύ σημαντικη διαδικασια καθως ο πελατης μπορει να μην εχει σαφης απαιτησεις από την αρχη.

Σχεδιαση συστήματος (System Design) : Σε επομενο σταδιο οι developers αναλυουν τις απαιτησεις του εγγραφου απιτησεων και σχεδιαζουν την αρχιτεκτονικη του συστηματος. Επισης το σχέδιο δοκιμής συστήματος (ATP) αναπτύσσεται με βάση τη σχεδιαση του συστήματος, αφηνοντας περισσότερο χρόνο για την πραγματική εκτέλεση της δοκιμής αργότερα

Σχεδιαση αρχιτεκτονικής (Architecture Design): Σε αυτό το στάδιο γινεται κατανοητες οι αρχιτεκτονικες προδιαγραφες. Προσδιοριζονται οι τεχνικες προσεγγισεις ενώ ο σχεδιασμος (αναφερεται και σαν High Level Design, HLD) του συστηματος διαιρειται σε μικροτερα διαφορετικα και πιο τεχνικα κομματια, καθως εδώ καθοριζονται τα μεσα μεταφορας, οι βασεις δεδομενων, η αλληλεπιδραση οι σχεσεις και οι εξαρτησεις των modules και τεχνικες λεπτομερειες.

Σχεδιαση module (Module Design): Σε αυτή τη φάση, καθορίζεται η λεπτομερής εσωτερικη σχεδιαση και η συμβατοτητα όλων των μονάδων του συστήματος. Βαση αυτης της εσωτερικης σχεδιασης αναπτυσσονται το σχέδιο δοκιμής αποδοχής ATP (Acceptance Test Plan) και τα σενάρια δοκιμής αποδοχής ATC (Acceptance Test Cases) που είναι πολύ σημαντικα για την εξάλειψη σφαλμάτων και ασυμβατοτήτων σε πρώιμο στάδιο.

Τα σταδια της επικυρωσης είναι τα εξης :

Unit Testing : Τα τεστ μονάδων εκτελούνται για να ελέγξουν τον κώδικα για ατομικά modules. Τα τεστ μονάδων βοηθούν στην εξάλειψη των σφαλμάτων σε πρώιμο στάδιο, αν και δεν μπορούν να εντοπίσουν όλα τα σφάλματα.

Integration Testing: Τα τεστ ολοκλήρωσης εκτελούνται για να ελέγξουν τη συνύπαρξη και την επικοινωνία μεταξύ των modules εντός του συστήματος. Τα τεστ ολοκλήρωσης βοηθούν να διασφαλιστεί ότι τα modules μπορούν να συνεργαστούν σωστά μεταξύ τους.

System Testing: Τα τεστ συστήματος εκτελούνται για να ελέγξουν ολόκληρη τη λειτουργικότητα του συστήματος και τη επικοινωνία του συστήματος υπό ανάπτυξη με εξωτερικά συστήματα. Τα τεστ συστήματος βοηθούν να διασφαλιστεί ότι το σύστημα πληροί όλες τις απαιτήσεις και ότι είναι συμβατό με άλλα συστήματα.

Acceptance Testing: Τα τεστ αποδοχής εκτελούνται για να ελέγξουν το προϊόν σε περιβάλλον χρήστη. Τα τεστ αποδοχής βοηθούν να διασφαλιστεί ότι το προϊόν είναι αξιόπιστο και ικανό να ανταποκριθεί στις ανάγκες των χρηστών.

Η "βαση του V" (Coding Phase) : Αυτή είναι η τελικη φαση οπου συνδεονται όλα και όπου γράφεται ο πραγματικός κώδικας για τις μονάδες συστήματος. Επιλέγεται η καλύτερη γλώσσα προγραμματισμού και ο κώδικας γράφεται σύμφωνα με τις οδηγίες και τα πρότυπα κωδικοποίησης του έργου. Στη συνέχεια ελέγχεται και βελτιστοποιείται πριν παραδοθει στον πελατη και διατεθει στην αγορα.

Αυτό το μοντέλο έχει εφαρμογή και στοχεύει κυριως σε μικροτερα Projects αφου είναι απλο και ευκολο στην κατανοηση του αλλα και σε περιπλοκα και υψηλης ποιοτητας εργα, με καθορισμένο βεβαια χρόνο αναπτυξης και απαιτησεων αλλα και για εργα που απαιτούν ασφάλεια, καθως δίνεται έμφαση στην ενδελεχή δοκιμή και στην εξασφαλιση της αξιοπιστιας και της ποιοτητας του λογισμικου, όπως για παράδειγμα σε αμυντικά και αεροδιαστημικά συστήματα. Είναι ένα γενικα αυστηρο μοντελο, καθως κάθε φαση εχει συγκεκριμενα παραδοτεα και ολοκληρωνονται η μια μετα την άλλη. Εξοικονομείται όμως πολύς χρόνος, αρά και πόροι αφού, αφου μέχρι η ομάδα προγραμματισμού να φτάσει στο στάδιο της έκδοσης, η ομάδα δοκίμων έχει ήδη ετοιμάσει το σχέδιο δοκίμων της. Έχει αρκετά ακόμα πλεονεκτήματα όπως την έναρξη των δοκιμών από τα αρχικά στάδια της αναπτυξης του προϊόντος έως και το τέλος της, δίνοντας ένα πιο ολοκληρωμένο προϊόν και με μικρότερο προϋπολογισμό λόγω των μειωμένων δοκιμών.

Στα μειονεκτήματα αυτού του μοντέλου είναι ο υψηλότερος κίνδυνος αλλά και ο υψηλοτερος αρχικός προϋπολογισμός, αφού και οι ομάδα προγραμματισμού και η ομάδα δοκιμών ξεκινάνε να εργάζονται από το πρώτο κιόλας στάδιο. Επίσης αν χρειαστεί κάποια αλλαγή στις απαιτήσεις, χρειαζεται πολυς χρονος προσαρμογης καθώς απαιτείται εκ νεου πολύ τεκμηρίωση (documentation) και δοκιμή. Ακριβώς επειδή δίνει πολύ έμφαση στην τεκμηρίωση, το έργο μπορεί να εξαρτηθεί υπερβολικά από αυτή και να αποκλίνει τελικά της πραγματικής υλοποίησης.

(https://producttribstg.wpengine.com/upl

**2.3.5 Iterative Model**

Το επαναληπτικό μοντέλο θυμιζει αρκετα το μοντελο agile, αφου και τα δυο χρησιμοποιουν επαναληπτικα sprints. Κάθε επαναληψη sprint περιεχει τα σταδια Planning, Design, Development και Testing που σε γενικες γραμμες και σε αυτό το μοντελο συμβαινουν οι ενεργειες που εχουν ηδη αναφερθει. Το μοντελο Iterative εστιάζει σε μια πιο απλοποιημένη προσέγγιση στην αρχή της υλοποίησης, η οποία σταδιακά αποκτά μεγαλύτερη πολυπλοκότητα και βελτιώνεται επαναληπτικά μέχρι το πλήρες σύστημα να είναι έτοιμο για ανάπτυξη. Έτσι, στην αρχή δεν εφαρμόζεται όλο το σύνολο των απαιτήσεων, αλλά ένα κομμάτι αυτών, ξεκινώντας με τον καθορισμό και την εφαρμογή κάποιων από τις απαιτήσεις. Έπειτα, ελέγχονται για προσδιορισμό άλλων απαιτήσεων που θα προκύψουν , οι οποίες προστίθενται στην εφαρμογή σε κάθε επανάληψη και σταδιακά δημιουργείται μια πλήρως αναπτυγμένη εκδοχή του λογισμικού.

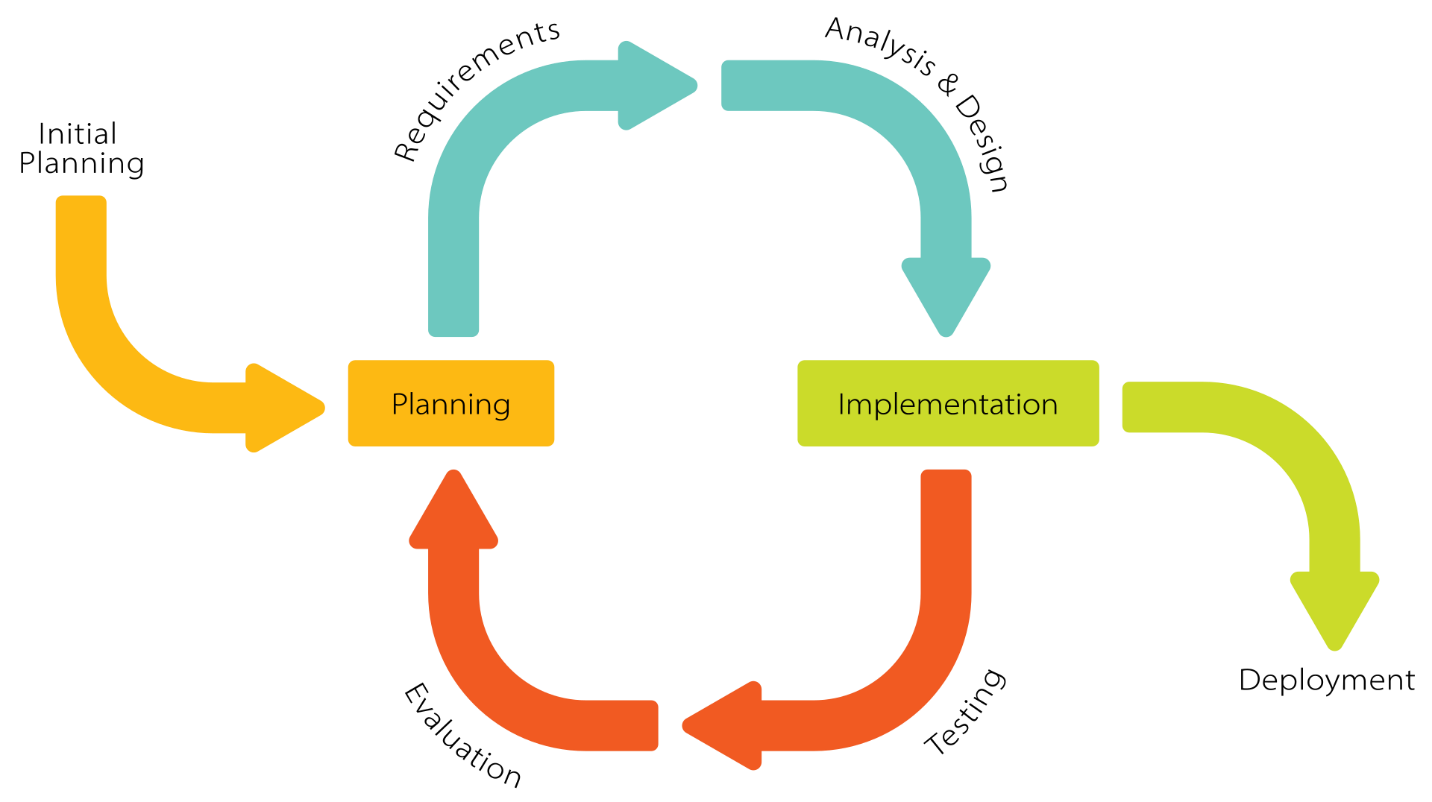
**Πλεονεκτήματα:**

* Ευκολη διαχειριση : Οι προγραμματιστες μπορουν να κανουν πιο ευκολα αλλαγες αφου η διαδικασια αναπτυξης είναι διαιρεμενη σε μικροτερα πιο διαχειρισιμα κομματια και ετσι μειωνεται και το κοστος των αλλαγων που μπορει να προκυψουν.
* Ευελιξία: Το μοντέλο αυτό είναι εξαιρετικά ευέλικτο, δίνοντας τη δυνατότητα προσαρμογής των προγραμματιστων στις αλλαγες που μπορει να προκυψουν από τους χρηστες κατά τη διάρκεια της ανάπτυξης, οποτε είναι μια καλη επιλογη για περιπλοκα εργα η για εργα με πιο απροσδιοριστες απαιτησεις.
* Επαναπροσδιορισμός: Η δυνατότητα επανεκτίμησης των απαιτήσεων και του σχεδιασμού μετά από κάθε επανάληψη βοηθάει στη βελτίωση του τελικού προϊόντος.
* Διαρκής βελτίωση: Η δυνατότητα ενσωμάτωσης αναδυόμενων απαιτήσεων στις επόμενες επαναλήψεις οδηγεί σε διαρκή βελτίωση του λογισμικού.

**Αρνητικά:**

* Δυσκολία στον έλεγχο του χρονοδιαγράμματος: Η ανεπίσημη φύση του μοντέλου μπορεί να καθυστερήσει τον προγραμματισμένο χρόνο παράδοσης, καθώς η συνεχής επανάληψη μπορεί να παρατείνει τη διάρκεια της ανάπτυξης.
* Πιθανότητα προστασίας στο σφάλμα: Κάθε επανάληψη μπορεί να εισάγει νέα σφάλματα στο λογισμικό, ενώ τα προβλήματα που αναδύονται μετά από κάθε επανάληψη μπορεί να είναι δύσκολο να εντοπιστούν. Λογω του ότι δεν συγκεντρωνονται ολες οι απαιτησεις εξαρχης μπορει να προκυψουν ζητηματα αρχιτεκτονικης η σχεδιασμου και να καθυστερησουν η να κανουν αβεβαιο το τελος του Project.
* Αύξηση του κόστους: Η επαναληπτική φύση του μοντέλου μπορεί να αυξήσει το κόστος της ανάπτυξης λόγω της ανάγκης για συνεχή αναθεώρηση και επανεκτίμηση, αλλα και υψηλοι ποροι για την αναλυση κινδυνου.

Συνοψίζοντας, το Iterative Model έχει εφαρμογή σε έργα που οι απαιτήσεις δεν είναι, γενικώς ή από την αρχή, ξεκάθαρες ή συνεχώς αλλάζουν κατά την διάρκεια της αναπτυξης και που έχουν ανάγκη από ευελιξία σε τροποποιήσεις. Είναι κατάλληλο για έργα που υπάρχει ανάγκη συχνής αναθεώρησης και αξιολόγησης της προόδου και συμμέτοχης από τους ενδιαφερομένους. Σε γενικές γραμμές, το επαναληπτικό μοντέλο είναι κατάλληλο για περιπτώσεις όπου η αβεβαιότητα, οι αλλαγές και η δυναμικότητα είναι κυρίαρχες, και όπου η συνεχής προσαρμογή και βελτίωση είναι ουσιώδης για την επίτευξη των στόχων του έργου.



***2.4 Προγραμματιστικά Framework (Programming/ Software frameworks)***

Κατηγορίες (πχ. Web frameworks, Front-End Frameworks κτλ).

1. Web Frameworks
2. Front-End Frameworks
3. Back-End Frameworks
4. Mobile App Frameworks
5. Content Management Frameworks
6. Data Science Frameworks

Prons / Cons

Μπορείς να αναφερθείς επίσης σε

Frameworks vs. Programming Languages

Frameworks vs. Libraries

Παραδείγματα από γνωστά frameworks (όχι εκτενή περιγραφή)

Σας κλείσιμο του κεφαλαίου έναν πίνακα συνοπτικό με τα παραπάνω  
  
Ένα προγραμματιστικό framework είναι ένα σύνολο εργαλείων και οδηγιών που χρησιμοποιείται για την ανάπτυξη λογισμικού. Τα προγραμματιστικά frameworks μπορούν να βοηθήσουν τους προγραμματιστές να αναπτύξουν λογισμικό πιο γρήγορα και αποτελεσματικά. Υπάρχουν πολλά διαφορετικά προγραμματιστικά frameworks διαθέσιμα, καθένα από τα οποία έχει τα δικά του πλεονεκτήματα και μειονεκτήματα.

**Τύποι προγραμματιστικών frameworks**

Υπάρχουν διάφοροι τύποι προγραμματιστικών frameworks, καθένας από τους οποίους χρησιμοποιείται για την ανάπτυξη διαφορετικών τύπων λογισμικού. Οι πιο κοινοί τύποι προγραμματιστικών frameworks περιλαμβάνουν:

* Web frameworks: Τα web frameworks χρησιμοποιούνται για την ανάπτυξη web εφαρμογών. Τα πιο δημοφιλή web frameworks περιλαμβάνουν το Django, το Flask και το Ruby on Rails.
* Front-end frameworks: Τα front-end frameworks χρησιμοποιούνται για την ανάπτυξη της μπροστινής πλευράς μιας web εφαρμογής. Τα πιο δημοφιλή front-end frameworks περιλαμβάνουν το React, το Angular και το Vue.js.
* Back-end frameworks: Τα back-end frameworks χρησιμοποιούνται για την ανάπτυξη της πίσω πλευράς μιας web εφαρμογής. Τα πιο δημοφιλή back-end frameworks περιλαμβάνουν το Laravel, το Symfony και το Spring Boot.
* Mobile app frameworks: Τα mobile app frameworks χρησιμοποιούνται για την ανάπτυξη mobile εφαρμογών. Τα πιο δημοφιλή mobile app frameworks περιλαμβάνουν το React Native, το Flutter και το Ionic.
* Content management frameworks: Τα content management frameworks χρησιμοποιούνται για την ανάπτυξη συστημάτων διαχείρισης περιεχομένου. Τα πιο δημοφιλή content management frameworks περιλαμβάνουν το WordPress, το Drupal και το Joomla.
* Data science frameworks: Τα data science frameworks χρησιμοποιούνται για την ανάπτυξη εφαρμογών μηχανικής μάθησης και τεχνητής νοημοσύνης. Τα πιο δημοφιλή data science frameworks περιλαμβάνουν το TensorFlow, το PyTorch και το scikit-learn.

**Πλεονεκτήματα και μειονεκτήματα των προγραμματιστικών frameworks**

Τα προγραμματιστικά frameworks προσφέρουν μια σειρά από πλεονεκτήματα στους προγραμματιστές, όπως:

* Μαγνητίζουν το κώδικα: Τα προγραμματιστικά frameworks μπορούν να βοηθήσουν τους προγραμματιστές να αναπτύξουν λογισμικό πιο γρήγορα και αποτελεσματικά, παρέχοντας ένα σύνολο προκαθορισμένων λειτουργιών και δομών δεδομένων που μπορούν να χρησιμοποιηθούν ξανά και ξανά. Αυτό μπορεί να εξοικονομήσει πολύ χρόνο και προσπάθεια στους προγραμματιστές.
* Βελτιώνουν την ποιότητα του κώδικα: Τα προγραμματιστικά frameworks μπορούν να βοηθήσουν τους προγραμματιστές να αναπτύξουν λογισμικό που είναι πιο ασφαλές και αξιόπιστο, παρέχοντας ένα σύνολο προκαθορισμένων κανόνων και προτύπων που πρέπει να ακολουθούνται. Αυτό μπορεί να βοηθήσει τους προγραμματιστές να αποφύγουν λάθη και να αναπτύξουν λογισμικό που είναι πιο πιθανό να είναι επιτυχημένο.
* Βελτιώνουν τη συμβατότητα: Τα προγραμματιστικά frameworks μπορούν να βοηθήσουν τους προγραμματιστές να αναπτύξουν λογισμικό που είναι πιο συμβατό με άλλα συστήματα και εφαρμογές, παρέχοντας ένα κοινό σύνολο API που μπορούν να χρησιμοποιηθούν από διαφορετικά προγράμματα. Αυτό μπορεί να διευκολύνει την ενσωμάτωση διαφορετικών συστημάτων και εφαρμογών και τη δημιουργία πιο ολοκληρωμένων λύσεων.

Ωστόσο, τα προγραμματιστικά frameworks έχουν επίσης ορισμένα μειονεκτήματα, όπως:

* Μπορεί να περιορίσουν την ευελιξία των προγραμματιστών: Τα προγραμματιστικά frameworks μπορεί να περιορίσουν την ευελιξία των προγραμματιστών, καθώς απαιτούν τους προγραμματιστές να ακολουθήσουν έναν συγκεκριμένο τρόπο ανάπτυξης. Αυτό μπορεί να είναι ένα πρόβλημα για τους προγραμματιστές που θέλουν να έχουν τον πλήρη έλεγχο της διαδικασίας ανάπτυξης.
* Μπορεί να είναι δύσκολο να μάθουν και να χρησιμοποιήσουν: Τα προγραμματιστικά frameworks μπορεί να είναι δύσκολο να μάθουν και να χρησιμοποιήσουν, ειδικά για τους προγραμματιστές που δεν έχουν εμπειρία με το συγκεκριμένο framework. Αυτό μπορεί να είναι ένα σημαντικό εμπόδιο για τους προγραμματιστές που θέλουν να χρησιμοποιήσουν ένα συγκεκριμένο framework, αλλά δεν έχουν την απαραίτητη εμπειρία.
* Μπορεί να είναι δαπανηρά: Τα προγραμματιστικά frameworks μπορεί να είναι δαπανηρά, ειδικά για τα μεγάλα και πολύπλοκα frameworks. Αυτό μπορεί να είναι ένα πρόβλημα για τις μικρές επιχειρήσεις ή τους προγραμματιστές με περιορισμένο προϋπολογισμό.

Συνοψίζοντας, τα προγραμματιστικά frameworks είναι ένα σημαντικό εργαλείο για τους προγραμματιστές. Μπορούν να βοηθήσουν τους προγραμματιστές να αναπτύξουν λογισμικό πιο γρήγορα και αποτελεσματικά, να βελτιώσουν την ποιότητα του κώδικα και να αυξήσουν τη συμβατότητα. Ωστόσο, τα προγραμματιστικά frameworks μπορεί επίσης να περιορίσουν την ευελιξία των προγραμματιστών και να είναι δύσκολα να μάθουν και να χρησιμοποιήσουν. Είναι σημαντικό οι προγραμματιστές να σταθμίσουν τα πλεονεκτήματα και τα μειονεκτήματα των προγραμματιστικών frameworks πριν αποφασίσουν να χρησιμοποιήσουν ένα συγκεκριμένο framework.

**3.****Σχεδιασμός και ανάπτυξη συστήματος** **(System development)**

Το τρίτο κεφάλαιο αποτελεί τον πυρήνα της εργασίας, όπου θα εξεταστούν εκτενώς οι βασικές φάσεις του σχεδιασμού και της ανάπτυξης του συστήματος. Κατά τη διάρκεια αυτών των φάσεων, οι σχεδιαστικές αποφάσεις και οι τεχνικές επιλογές λαμβάνουν χώρα, προσαρμόζοντας τις απαιτήσεις σε μια λειτουργική αρχιτεκτονική και μετατρέποντάς τις σε πραγματικότητα.

Το κεφάλαιο ξεκινά με την ανάλυση των απαιτήσεων του συστήματος, όπου γίνεται η κατανόηση και καταγραφή των λειτουργικών και μη λειτουργικών απαιτήσεων που πρέπει να πληροί το σύστημα. Η εμβάθυνση σε αυτό το στάδιο βοηθά στο να διασφαλιστεί ότι οι ανάγκες των χρηστών και του περιβάλλοντος λαμβάνονται υπόψη.

Στη συνέχεια, ακολουθεί η αρχιτεκτονική του συστήματος, όπου γίνεται η επιλογή της κατάλληλης αρχιτεκτονικής για την υλοποίηση των απαιτήσεων. Αυτή η φάση επικεντρώνεται στον σχεδιασμό της δομής και των συνιστωσών του συστήματος, προκειμένου να επιτευχθεί αποδοτική και αξιόπιστη λειτουργία.

Ένα σημαντικό εργαλείο στον σχεδιασμό του συστήματος αποτελεί το σχεσιακό διάγραμμα βάσης δεδομένων (ERD), το οποίο παρουσιάζει τη δομή της βάσης δεδομένων και τις σχέσεις μεταξύ των διαφόρων πινάκων. Μέσω αυτού του διαγράμματος, επιτυγχάνεται η βελτιστοποίηση της αποθήκευσης και διαχείρισης των δεδομένων.

Τέλος, η φάση του σχεδιασμού και της ανάπτυξης συμπεριλαμβάνει και τις ρουτίνες του συστήματος, δηλαδή τις λειτουργίες που υλοποιούνται για την κάλυψη των απαιτήσεων. Στο κομμάτι αυτό, αναλύονται λεπτομερώς οι ενέργειες που αναλαμβάνει το σύστημα, καθώς και ο τρόπος αλληλεπίδρασής του με τους χρήστες.

Αυτό το κεφάλαιο αποτελεί το σημαντικό βήμα προς την πραγματοποίηση των απαιτήσεων, προετοιμάζοντας το έδαφος για την περαιτέρω ανάπτυξη, υλοποίηση και δοκιμή του συστήματος.

**3.1** **Περιγραφη Συστηματος, Χρηστες (System Analysis,User Roles)**

Η παρούσα εφαρμογή αναπτύχθηκε με στόχο να παρέχει μια σειρα λειτουργιών για τη διαχείριση έργων συμφωνα με την διαδικασια SDLC που αναφερθηκε παραπανω αλλα και του προσωπικού που το απαρτηζουν, με έμφαση στην απλότητα και την χρηστικότητα. Η ανάπτυξη τέτοιων λειτουργιων απαιτεί να εντοπιστουν και να εξεταστους οι υπάρχουσες ανάγκες και να οριστουν οι διαφορετικοι ρόλοι των χρηστών, η συμπεριφορά τους σε διαφορετικές περιπτώσεις χρήσης καθως και οι απαιτησεις ολου του συστηματος που θα πρεπει να ικανοποιουνται τελικα.

Η εφαρμογη χωριζεται, χρονολογικα αλλα και χωρικα σε τεσσερις μεγαλες κατηγοριες: Επαληθευση Χρηστων, Καταχωρηση Πληροφοριων, Τροποποιησεις, Οπτικοποιηση Δεδομενων.

Η επαληθευση των χρηστων αναφερεται σε ολες τις ενεργειες που ακνει ο χρηστης προκειμνεου να συνδεθει με καποιο ρολο στην εφαρμογη και να του δοθει προσβαση και δικαιωματα χρησης σε ορισμενες απο τις λειτουργιες.

Η καταχωρηση Πληροφοριων αναφερεται σε ολες εκεινες τις λειτουργιες με τις οποιες ο χρηστης μπορει να αρχικοποιησει οποιαδηποτε πληροφορια στο συστημα, οποιαδηποτε εισροη και ειναι ισως η πιο σημαντικη κατηγορια της συγκεκριμενης εφαρμογης. Εχει δοθει μεγαλη προσοχη στην υλοποιηση αυτου του κομματιου, προκειμενου οι αναγκαιες πληροφοριες να συλλεγονται σωστα και να αποθηκευονται στην συνεχεια στο συστημα. Κυριος υπευθυνος της της κατηγοριας Καταχωρησης Πληροφοριων ειναι ο διαχειριστης, αφου το μεγαλυτερο μερος ολων των δεδομενων που παρεχονται στην εφαρμογη καταχορουνται απο τον ιδιο, απο τα στοιχεια του καθε προτζεκτ μεχρι και τα προσωπικα στοιχεια των υπαλληλων

Η κατηγορια της τροποποιησης αναφερεται σε ολες τις λειτουργιες στις οποιες γινεται επεξεργασια η αλλαγη ηδη υπαρχοντων δεδομενων. Βασικες λειτουργιες ειναι η επεξεργασια υπαλληλου και η παρακολουθηση της προοδου που θα αναλυθει και παρακατω.

Η κατηγορια της οπτικοποιησης δεδομενων εχει να κανει με ολες τις λειτουργιες οι οποιες εξυπηρετουν τον χρηστη ως προς την προβολη ολων αυτων των στοιχειων, που δοθηκαν στο συστημαο, με οργανωμενο και αναλογο για την χρηση τροπο.

Αυτες οι τεσσερις κατηγοριες μπορουν να συνδυαστουν και ο συνδυασμος αυτων αφορα την συνθεση των Project με Employees και την παρακολουθηση αυτων προκειμενου να δημιουργηθεί μια πλατφορμα που θα βοηθήσει τον χρήστη να κατανοήσει τη διαδικασία Ανάπτυξης Λογισμικού (SDLC) στην πράξη και να παρεχει δυνατοτητες διαχειρισης και προβολης πληροφοριων προς ολους τους εμπλεκομενους στην αναληψη ενός προγραμματιστικου εργου με διαφορετικες απαιτησεις, διαφορετικα χρονικα πλαισια.

**Ρολοι Χρηστων Συστηματος**

Το πρωτο βημα στην σχεδιαση της εφαρμογης, είναι η αναγνωριση των χρηστων που προβλεπεται να χρησιμοποιησουν την εφαρμογη, καθως και οι λειτουργιες που αναμενονται να εξυπηρετησουν κάθε χρηστη. Τρεις διαφορετικοι ρολοι χρηστων οριζονται : διαχειριστης (admin), υπευθυνοι εργου (managers) και υπαλληλοι (employees) .

Όπως σε κάθε υπηρεσια, ετσι και στην παρουσα εφαρμογη, ο χρηστης με το ρολο του διαχειριστη, εχει προσβαση σχεδον σε κάθε λειτουργια του συστηματος, σε σχεση με αλλους χρηστες με διαφορετικους ρολους, που οι λειτουργιες τους περιοριζονται. Ο αριθμός των διαχειριστών χρηστών περιορίζεται σε έναν για λόγους ασφαλείας και απλοτητας. Κυριες λειτουργιες του διαχεριστη είναι η αρχικοποιηση και η παρακολουθηση της ομαλης διεξαγωγης των εργων, της προσθηκης εργαζομενων στο συστημα και η επιβλεψη των πληρωμων τους. Να σημειωθει ότι μονο σε εναν μικρο αριθμο λειτουργιων δεν εχει προσβαση ο διαχειριστης, καθως δεν τον αφορουν εμμεσα, αλλα θα αναλυθουν ολες οι λειτουργιες εκτενεστερα στην συνεχεια.

Οι υπευθυνοι εργων (managers) αποτελουν ξεχωριστο κομματι και ταυτοχρονα μερος των υπαλληλων και διαφοροποιουνται σε σχεση με αυτους, καθως επιλεγονται από τους διαχειριστες κατά την διαρκεια της αρχικοποιησης ως υπευθυνοι συγκεκριμενων εργων. Ως εκ τουτου διαφοροποιουνται και οι δυνατοτητες που τους παρεχονται, με σημαντικοτερη αυτή της παρακολουθησης και της ανατροφοδοτησης της πορειας του εργου στον διαχεριστη και της επιλογης καταλληλων υπαλληλων για το εκαστοτε εργο.

Οι υπαλληλοι (employees) είναι ατομα που απασχολουνται από την επιχειρηση και συνεπως και από την πλατφορμα και εχουν προσληφθει και εκχωρηθει στο συστημα από τον διαχειριστη. Όπως αναφερθηκε παραπανω, μερικοι από αυτους ενδεχομενως να εχουν και δευτερο ρολο, αυτόν του manager, αλλα στην περιπτωση που μονο συμμετεχουν σε project, η προσβαση τους περιοριζεται στην προβολη των λεπτομεριων των project στο οποια συμμετεχουν και των πληρωμων που εχουν πραγματοποιηθει η θα πραγματοποιηθουν μελλοντικα και τους αφορουν.

Είναι σημαντικό να σημειωθεί ότι αυτές οι λειτουργίες αποτελούν τη βάση για την ανάπτυξη περαιτέρω χαρακτηριστικών και προσθηκών στο μέλλον, ώστε να ανταποκρίνονται στις εξελισσόμενες ανάγκες των χρηστών και της αγορας και να προσφέρουν ακόμη περισσότερη ευελιξία και αποδοτικότητα.  
  
  
ΕΔΩ ΜΠΟΡΩ ΝΑ ΒΑΛΩ TERMINOLOGY…..αργοτερα

**3.2** **Απαιτησεις και Λειτουργιες Συστηματος**

Το κεφάλαιο των Απαιτήσεων Συστήματος αποτελεί θεμέλιο για τον σχεδιασμό και την ανάπτυξη κάθε πληροφοριακού συστήματος καθως ορίζει και καθορίζει πώς το πληροφοριακό σύστημα πρέπει να λειτουργεί, επιτυγχάνοντας τον στόχο του και ικανοποιώντας τις ανάγκες των χρηστών. Εδώ αναδεικνύονται oλες οι λειτουργιες που πρεπει το συστημα να περιλαμβανει για κάθε χρηστη και που οδηγούν σε μια πλατφορμα που απλοποιει την διαδικασία της ανάπτυξης και αναλυονται ολες οι περιπτωσεις χρησης για κάθε πιθανο σεναριο χρησης της εφαρμογης.

**Συνδεση Χρηστη στην πλατφορμα**

Η σύνδεση του χρηστη στην πλατφόρμα, γίνεται με την χρηση των διαπιστευτήριων του, όπως σε κάθε συγχρονη εφαρμογη, δηλαδή το email του χρήστη (Email) και τον κωδικό πρόσβασης (Password). Στο κομματι της συνδεσης, μαζι με τα διαπιστευτηρια, ο χρηστης θα πρεπει να επιλεξει και τον ρολο, που θελει να παρει κατά την συνδεση του στο συστημα μεσω των επιλογων στην καρτελα συνδεσης. Ο Admin όπως εχει ηδη αναφερθει είναι ηδη καταχωρημενος εξ ορισμου από το συστημα. Προκειμενου οι συνδεση των υπαλληλων η των manager να είναι επιτυχης, ο Admin θα πρεπει να εχει καταχωρησει τους αντιστοιχους υπαλληλους στο παρελθον. Όπως εχει προαναφερθει, η υπαρξη του manager είναι και αυτή αμεσα συνδεδεμενη με τον διαχειριστη της πλατφορμας, αφου και σε αυτή την περιπτωση ο admin είναι υπευθυνος για τον ορισμο και την δημιουργια manager. Επομένως το σύστημα δεν έχει και σελίδα εγγραφής (Register page) για καινούργιους χρήστες αφου όλα τα employees accounts δημιουργουνται και παρεχονται ηδη από τον admin για καθε χρηστη αυτοματα με την εισαγωγη του στο συστημα.

**Καταχωρηση Υπαλληλων**

Η καταχωρηση υπαλληλων στο συστημα, ειναι η πρωτη σημαντικοτερη λειτουργια που πρεπει να προσφερει η εφαρμογη και αφορα τον admin και μονο. Αφου επιλεξει την καταχωρηση ενος υπαλληλου, ο admin συμπληρωνει τα απαραιτητα στοιχεια που ζητουνται απο το συστημα , οπως το ονομα του, τον κωδικο προσβασης του, τα μισθοδοτικα του στοιχεια κλπ. Τα στοιχεια αυτα θα χρησιμοποιηθουν σε διαφορετικα σημεια της εφαρμογης και με το περας της καταχωρησης ενος υπαλληλου, δημιουργειται ο λογαριασμος του. Πλεον και επειτα απο συνεννοηση με τον διαχεριστη, για να ενημερωθει για τον κωδικο προσβασης,ο υπαλληλος εχει την δυνατοτητα να συνδεθει στην πλατφορμα με δικαιωματα υπαλληλου. Επισης με την νεα καταχωρηση του υπαλληλου στο συστημα, αυτος θα ειναι ορατος πλεον στην σελιδα των υπαλληλων και θα ειναι διαθεσιμος να του οριστουν καθηκοντα σε ενα ενα project, διαδικασια που θα αναλυθει παρακατω.

**Καταχωρηση Project**

Η επομενη πιο σημαντικη λειτουργια σε μια τετοια εφαρμογη και ουσιαστικα και ο κυριος λογος δημιουργιας της ιδιας της εφαρμογης είναι προφανως, η καταχωρηση νεων προτζεκτ στην πλατφορμα. Κάθε προτζεκτ πρεπει να είναι σωστα ορισμενο ως προς τις φασεις του και τους εργαζομενους που απασχολουνται σε αυτό. Κατά την αρχικοποιηση του προτζεκτ ο χρηστης πρεπει να είναι σε θεση να επιλεγει τις φασεις που το απαρτιζουν, να οριζει ποτε ξεκινουν, τον προυπολογισμο τους και ενδεχομενως καποια tasks. Ο τροπος που ο χρηστης επιλεγει τις φασεις θα πρεπει να είναι τετοιος, που να είναι σε θεση να επιλεξει το μοντελο sdlc που τον εξυπηρετει να χρησιμοποιησει. Αυτό σημαινει να μπορει να ορισει επικαλυπτομενες ημερομηνιες στις φασεις, η επαναλαμβανομενα σετ φασεων, οπως θα χρειαζοταν για παραδειγμα στο μοντελο spiral. Οι φασεις θα πρεπει να είναι συμφωνες χρονικα με τα χρονικα ορια του project συνολικα και τα budget τους να μην υπερβαινουν το προβλεπομενο budget ολου του εργου. Για να είναι λειτουργικη η εφαρμογη, ο admin θα πρεπει να είναι σε θεση να επιλεγει και να οριζει υπαλληλους για κάθε φαση. Σημασια στην επιλογη εχουν και οι διαθεσιμοι ποροι αλλα και οι διαθεσιμες ημερομηνιες των φασεων του εκαστοτε rpoject οποτε η εφαρμογη θα πρεπει να προσφερει με οσο το δυνατον πιο ξεκαθαρο τροπο την ολη διαδικασια. Για αυτο το λογο, η αρχικοποιηση ενος προτζεκτ στην συγκεκριμενη εφαρμογη εχει χωριστει σε τρεις ειδικοτερες φασεις, μοιραζοντας την διαδικασια σε τρεις κατηγοριες αναλογα με το τι πραγματευεται η καθεμια και ειναι οι εξης : Προσθηκη Project ( 1η φαση), Προσθηκη Φασεων και Ορισμος Manager ( 2η φαση) και Προσθηκη Υπαλληλου για καθε φαση (3η Φαση). Καθε φαση της τιμης status που οριζει την τρεχουσα κατασταση του project, η πλατφορμα προβαλλει στον εκαστοτε χρηστη τις αναλογες ενεργειες που μπροει να εκτελεσει την προκειμενη στιγμη. Η τρεχουσα κατασταση δεν αναφερεται μονο στην αρχικοποιηση του project, οσο και μετα το περας αυτης.θα αναλυθει παρακατω και αφορα το status καθε projectΠαρακατω θα αναλυθουν οι απαιτησεις που πρεπει να πληροι η καθε φαση αρχικοποιησης, ωστε ολες μαζι να ικανοποιουν συνολικα την δομη ενος νεου Project βασιζομενες στις αναγκες και τις απαιτησεις μιας πραγματικης εταιρειας sofware.

***1η Φαση Αρχικοποιησης – Προσθηκη Project***

To πρωτο κομματι της αρχικοποιησης του project, αφορα αποκλειστικα και μονο τον διαχεριστη, αφου μονος αυτος εχει προσβαση σε αυτη την λειτουργια και εχει να κανει με τον καθορισμο των βασικοτερων στοιχειων του project. Στοιχεια οπως το ονομα του προτζεκτ, η διαρκεια του (σε μηνες) και ο συνολικος του προυπολογισμος, επειτα απο την επιλογη “Add New Project”, συμπληρωνονται απο τον διαχεριστη, για να εκχωρησει στο συστημα μια πρωτη “εικονα” με τα βασικα για αρχη, δεδομενα που θα χρειαστουν για να δομησουν ολοκληρο το |project στην δευτερη φαση αρχικοποιησης.

Η λειτουργια αυτη ειναι στοχευμενα απλη στην χρηση της αφου ο διαχειριστης δηλωνει μονο τις πολυ βασικες πληροφοριες και δεν απαιτει ιδιαιτερη προσοχη. Με την ολοκληρωση της πρωτης φασης, οι δοθεισες πληροφοριες καταγραφονται στο συστημα και ο διαχεριστης μπορει πλεον να δει το συγκεκριμενο project στην λιστα με τα συνολικα projects. Να σημειωθει επισης πως αφου ολοκληρωθει η 1η φαση, τοτε ενεργοποιειται μεσω της πλατφορμας, η επιλογη αρχικοποιησης της 2ης φασης που θα αναλυθει παρακατω.

***2η Φαση Αρχικοποιησης – Προσθηκη Φασεων και Manager***

To δευτερο κομματι της αρχικοποιησης του project, αφορα επισης αποκλειστικα και μονο τον διαχεριστη, αφου μονος αυτος εχει προσβαση σε αυτη την λειτουργια και εχει να κανει με τον καθορισμο των φασεων του εκαστοτε project και την επιλογη Manager. Η λειτουργια αυτη ειναι πιο περιπλοκη, αφου σε αυτη οριζονται ο αριθμος των φασεων, η επιλογη του καταλληλου SDLC μοντελου, η καταχωρηση των απαραιτητων πληροφοριων για καθε φαση που επιλεχθει αλλα και του καταλληλου manager για το εργο. Για αυτο τον λογο και για να επιτευχθει η απλοτητα και σε αυτη τη φαση, η αρχικοποιηση εχει χωριστει σε τρια σταδια. Στο πρωτο σταδιο γινεται η επιλογη των φασεων, στο δευτερο σταδιο γινεται η αρχικοποιηση των δεδομενων των φασεων και στο τριτο σταδιο γινεται η επιλογη του manager.

**1) Επιλογη Φασεων**

Με την εναρξη της δευτερης φασης, αρχικα ο διαχειριστης καλειται να επιλεξει τις απαραιτητες φασεις που εχουν οριστει για την αναληψη και ολοκληρωση του εκαστοτε project. Η εφαρμογη του δινει την δυνατοτητα, ειτε να επιλεξει απο τις ηδη υπαρχουσες φασεις, που αφορουν τις φασεις του μοντελου waterfall, ειτε να προσθεσουν τις δικες τους custom φασεις. Η επιλογη που παρεχεται στον διαχεριστη να προσθετει την δικια του φαση εξυπηρετει την προφανη αναγκη επιλογης καποιου αλλου μοντελου SDLC, οπως για παραδειγμα του μοντελου Spiral η του μοντελου Agile, οποτε και οι ηδη υπαρχουσες απο το συστημα φασεις να μην τον εξυπηρετουν. Ο διαχεριστης μπορει να δωσει οποιοδηποτε ονομα θελει στην φαση και να την προσθεσει στην λιστα των επιλεγμενων φασεων, ειτε μαζι με καποιες απο τις ηδη υπαρχουσες φασεις ειτε αποκλειστικα και μονο με δικες του φασεις. Δινεται η ελευθερια πληρους επιλογης, καθως αυτο το σημειο για καθε προτζεκτ μπορει να ειναι πολυ διαφορετικο απο ενα αλλο, οποτε η λογικη των προεπιλεγμενων φασεων, η συγκεκριμενου μοντελου, δεν θα ηταν ουτε χρηστικη ουτε ευελικτη. Ο μονος περιορισμος που συναντα ο διαχειριστης ειναι η απαιτηση απο το συστημα ο αριθμος των φασεων να ειναι μεγαλυτερος απο τεσσερις και αυτο γιατι αναληψη εργου με λιγοτερες απο 4 φασεις δεν υφισταται στις σημερινες απαιτησης αναπτυξης λογισμικου.

**2) Αρχικοποιηση Φασεων**

Αμεσως μετα την επιλογη των φασεων, ο διαχειριστης καλειται να αρχικοποιησει με δεδομενα την καθε φαση ξεχωριστα. Εξισου απαιτητικη ειναι και αυτη η λειτουργια, καθως η λεπτομερης και μελετημενη καταχωρηση στοιχειων στις φασεις απαιτειται για την ορθη διεξαγωγη του project συνολικα. Οποιοδηποτε λαθος στην καταχωρηση θα εχει σιγουρο αντικτυπο στην συνεχιση της αρχικοποιησης του project και τελικα στη λανθασμενη διαχειριση αυτου. Ο λογος που αυτη η διαδικασια πρεπει να ειναι ακριβης ως προς την υλοποιηση της και ως προς το περιεχομενο της, ειναι επειδη σε αυτο το σταδιο οριζονται η διαρκεια (σε μηνες), το budget και τα tasks αλλα και η ημερομηνια εναρξης της καθε φασης.

Η διαρκεια οριζεται ως ενα υποσυνολο της συνολικης διαρκειας του project που ειχε οριστει στο πρωτο σταδιο αρχικοποιησης και η ελαχιστη διαρκεια καθε φασης ειναι 1 μηνας. Να σημειωθει οτι ο πρωτος περιορισμος που προκυπτει απο το συστημα ειναι η καταλληλη δηλωση μηνων σε καθε φαση ετσι ωστε η συνολικη διαρκεια ολων των φασεων συνολικα, να ειναι συμφωνη με την ορισμενη διαρκεια του project.

Ο δευτερος περιορισμος προκυπτει απο το budget που οριζεται και αυτο ως ενα υποσυνολο του συνολικου προυπολογισμου του project που ειχε οριστει στο πρωτο σταδιο αρχικοποιησης. Ελαχιστο budget ανα φαση δεν υπαρχει, αλλα και σε αυτην την περιπτωση ο συνολικος αριθμος ολων των δοθεντων συγκεντρωτικα budgets καθε φασης πρεπει να ειναι συμφωνος με τον προυπολογισμο που ειχε οριστει στο προτζεκτ στην πρωτη φαση. Να σημειωθει πως η εφαρμογη ενημερωνει σε πραγματικο χρονο καταλληλα τον διαχειριστη, τοσο για το αν εχει υπερβει την συνολικη διαρκεια οσο και τον συνολικο προυπολογισμο αλλα και για τις εναπομηνασες φασεις και προυπολογισμο, προκειμενου να συμφωνουν με τα πρωτα αρχικα δεδομενα του project.

Οσον αφορα την ημερομηνια εναρξης, ειναι ισως το πιο σημαντικο στοιχειο και αν συνδυαστει και με τα task με συγκεκριμενο τροπο μπορουν να διαρθρωσουν και να προσομοιωσουν οποιοδηποτε μοντελο SDLC. Ο διαχειριστης οριζει την ημερομηνια εναρξης για καθε φαση, και σε συνδυασμο επισης με την διαρκεια της φασης, οριζεται και η τελικη ημερομηνια. Οι ημερομηνιες μεταξυ τους μπορουν ειτε να επικαλυπτονται, ειτε να τελειωνει η μια και να ξεκιναει η αλλη, η να τρεχει καποιο σετ φασεων παραλληλα και γενικως υπαρχει ελευθερια στην επιλογη και στην διαρθρωση του project. Ειδικα ομως εξαιτιας αυτης της ελευθεριας, ειναι πολυ ευκολο ο διαχεριστης να μην καθορισει σωστα τις φασεις, οποτε στην προκειμενη, η ελευθερια δημιουργει απαιτησεις και αυστηρα καθορισμενες φασεις. Ο τροπος που θα οριστουν οι ημερομηνιες εναρξης θα καθορισει το μοντελο που θα χρησιμοποιηθει για την αναπτυξη της εφαρμογης.

Τα tasks αναφερονται στον αριθμο των tasks που πρεπει να εκτελεστουν για τον τερματισμο καθε φασης και εν συνεχεια και του προτζεκτ και στην ουσια υποδηλωνουν την προοδο καθε φασης. Να σημειωθει πως αν ο διαχεριστης θελει να ακολουθησει το μοντελο Agile η Spiral παραδειγματος χαρη, ο αριθμος των tasks θα δηλωνε τον αριθμο των φασεων της μιας επαναληψης στο Agile η το αριθμο των φασεων καθε σπειρας στο Spiral, προκειμενου να μπορουν να αποδοθουν ολες οι τεχνικες λεπτομερειες αναλογως μοντελου επιλογης.

**2) Επιλογη Manager**

Αφου λοιπον ο διαχεριστης τελειωσει και με το δευτερο σταδιο της δευτερης φασης αρχικοποιησης, καλειται να ορισει εναν υπαλληλο ως manager του project. Η αναγκη υπαρξης manager σε ενα project προκυπτει απο το γεγονος οτι το project θα ειναι πιο λειτουργικο και θα απελευθερωνει τον admin απο καποια καθηκοντα οπως την επιλογη των καταλληλων υπαλληλων και την επιβλεψη της προοδου του project. Ουτως η αλλως και σημερα, οι τωρινες απαιτησεις των εταιρειων προβλεπουν την θεση του manager και μαλιστα ως πολυ σημαντικη, καθως επεκτεινεται η ιεραρχια, μοιραζονται οι ευθυνες και τα project γινονται πιο διαχερισιμα. Η επιλογη του manager περαν δεξιοτητων εξαρταται και απο τα μισθολογικα του στοιχεια. Για αυτο το λογο η εφαρμογη προβαλλει μια λιστα με οσους υπαλληλους ειναι διαθεσιμοι και δινει την δυνατοτητα επιλογης του πιο καταλληλου υπαλληλου. Η πλατφορμα σε αυτο το σημειο εχει περιορισει την λιστα με τους υπαλληλους, ετσι ωστε να προβαλλονται μονο οσοι εχουν προσληφθει σαν “full time”. Για καθε υπαλληλο η λιστα δειχνει το ονομα του καθως και την μηνιαια αμοιβη του. Αναλογως την επιλογη του, ο διαχειριστης ενημερωνεται σε πραγματικο χρονο για την συνολικη αμοιβη του manager για ολοκληρο το project, προκειμενου να εχει μια εικονa του ποιου μερους του budget θα μοιραστει στους υπαλληλους και ποιο μερος θα διατεθει στο manager με την συγκεκριμενη επιλογη υπαλληλου. Να σημειωθει πως, αφου γινει η επιλογη, η τελικη αμοιβη του manager, θα αφαιρεθει ισοποσα απο τους προυπογισμους των φασεων που εχουν οριστει.

Με την ολοκληρωση της δευτερης φασης, οι δοθεισες πληροφοριες καταγραφονται στο συστημα και ο διαχεριστης μπορει πλεον να δει το ενημερωμενο project στην λιστα με τα συνολικα projects. Αφου ολοκληρωθει η 2η φαση, τοτε ενεργοποιειται μεσω της πλατφορμας, η επιλογη αρχικοποιησης της 3ης φασης που θα αναλυθει παρακατω. Με το περας ομως της διαδικασιας αυτης μια αλλη σημαντικη λειτουργια ειναι διατεθειμενη προς τους υπαλληλους της πλατφορμας, οπως ειχε αναφερθει και παραπανω. Σε αυτη την περιπτωση αναφερομαστε σε αυτους τους υπαλληλους που εχουν και δικαιωματα manager σε καποιο project, εκτος της ιδιοτητας τους ως υπαλληλοι. Πλεον αυτοι οι υπαλληλοι μπορουν να συνδεθουν στην πλατφορμα με τον ρολο του Manager, δινοντας τους καποιες παραπανω λειτουργιες οσον αφορα το project.

***3η Φαση Αρχικοποιησης – Επιλογη Υπαλληλων για το Project***

Οπως αναφερθηκε παραπανω, η χρηση manager ειναι αναγκαια για να λειτουργησει το συστημα με βαση τα σημερινα δεδομενα. Για αυτο το λογο, τo τριτο κομματι της αρχικοποιησης του project, το οποιο εχει να κανει με τον καθορισμο υπαλληλων στις φασεις, σε αντιθεση με τα προηγουμενα δυο parts, αφορα αποκλειστικα και μονο τον μανατζερ. Mε βαση τον τροπο που κατασκευαστηκε το συστημα, η υποχρεωση-καθηκον της επιλογης προσωπικου για το εκαστοτε project βαραινει αποκλειστικα και μονο τον manager, αφου μονος αυτος εχει προσβαση σε αυτη την λειτουργια. Η λειτουργια αυτη ειναι θεωρητικα απλη στην εκτελεση της και με το περας της τερματιζεται συνολικα ολη η διαδικασια αρχικοποιησης. Να σημειωθει επισης πως δημιουργονται και οι αναλογες πληρωμες απο το συστημα για τους εκαστοτε υπαλληλους που επιλεγονται. Ο τροπος επιλογης θα πρεπει να ειναι απλος και να συμφωνει με τροπο λειτουργιας του ολου project. Οι περιορισμοι που υπαρχουν σε αυτη την φαση, εχουν να κανουν με τα ποσα πληρωμων αλλα και την ημερομηνια διεξαγωγης της εκαστοτε φασης.

Οι πληρωμες των υπαλληλων οριζονται ως υποσυνολα του συνολικου προυπολογισμου της εκαστοτε φασης που ειχε οριστει στο δευτερο σταδιο αρχικοποιησης. Το συνολο των πληρωμων ολων των επιλεγμενων υπαλληλων για καθε φαση πρεπει να ειναι συμφωνος με τον προυπολογισμο που ειχε οριστει στο δευτερο σταδιο αρχικοποιησης.

Αναφορικα με την ημερομηνια διεξαγωγης της φασης, το συστημα εχει δημιουργηθει με τετοιο τροπο ωστε να εξασφαλισει στον manager, την επιλογη μονο των υπαλληλων που δεν εργαζονται σε καποιο αλλο project κατα την διαρκεια διεξαγωγης της συγκεκριμενης φασης, προκειμενου να μην υπαρξoυν επιπλοκες.

H εφαρμογη ενημερωνει σε πραγματικο χρονο καταλληλα τον manager, για το συνολικο ποσο πληρωμων, προκειμενου να συμφωνει και να ειναι στο αποδεκτα ορια του προυπλογισμου της φασης, οπως οριστηκε στο δευτερο σταδιο αρχικοποιησης.

Με την εκτελεση αυτου του σταδιου, το project ολοκληρωνεται, ως προς την αρχικοποιηση του και οι εμπλεκομενοι υπαλληλοι καθως και ο admin μπορουν πλεον να δουν το project στην λιστα με τα projects τους.

**Καταργηση κατα την Αρχικοποιηση**

Αφου αναλυθηκαν τα σταδια αρχικοποιησης ενος προτζεκτ και ο τροπος χρησης τους, γινεται αντιληπτο, πως λογω της πολυπλοκοτητας καποιων σημειων στα σταδια αυτα, αρα και ενδεχομενου λαθους, ο admin πρεπει να εχει την δυνατοτητα καταργησης ενος project, σε οποιοδηποτε φαση και αν βρισκεται, προκειμενου να μπορεσει να το επαναρχικοποιησει και να το διορθωσει με τις σωστες πληροφοριες. Αυτη η λειτουργια αφορα επισης και τον manager, εφοσον το project βρισκεται στο τριτο σταδιο αρχικοποιησης. Αυτη η λειτουργια προλαμβανει αστοχιες που μπορει να προεκυψαν με την καταχωρηση λανθασμενων πληροφοριων και κρινεται απαραιτητη για την ευρυθμη λειτουργια της πλατφορμας συνολικα. Την λειτουργια αυτη ο εκαστοτε χρηστης μπορει να την βρει στην λιστα με τα projects, σαν action button για καθε Project ξεχωριστα και αν επιλεχθει η καταργηση, ο χρηστης μεσω ενος modal επιβεβαιωνει την επιλογη του. Με το περας αυτης της λειτουργιας, το Project και ολες οι πληροφοριες, οπως οι φασεις, οι υπαλληλοι κλπ, διαγραφονται ολοκληρωτικα απο την βαση δεδομενων και απο τις λιστες με τα project των χρηστων. Αναφορικα με την διαγραφη, να σημειωθει πως αυτη η λειτουργια ειναι η μονη που κανει πληρη διαγραφη πληροφοριων απο το συστημα και δεν θα ξαναεμφανιστει αναλογη λειτουργια στην υπολοιπη εφαρμογη.

**Τερματισμος κατα την Εκτελεση**

Αλλη μια λειτουργια που ειναι απαραιτητη στην εφαρμογη ειναι και αυτη που τερματιζει το project. Δυνατοτητα τερματισμου του project δινεται μονο στο admin και ο τερματισμος μπορει να αναφερεται ειτε στην επιτυχη ολοκληρωση του project, ειτε στον προωρο τερματισμο του, γεγονος που το καθιστα πλεον μη ενεργο, αρα και διαχειρισιμο. Με το περας αυτης της λειτουργιας, το status του project τιθεται με την τιμη 4, γεγονος που δηλωνει οτι στην λιστα με τα Projects, το συγκεκριμενο project θα εχει την δυνατοτητα πλεον μονο να προβληθει οπως αναφερθηκε παραπανω.

**Προβολη ολων των project**

Οπως αναφερθηκε παραπανω τα projects θα πρεπει να εμφανιζονται συγκεντρωτικα στο μενου του εκαστοτε χρηστη. Με την μορφη λιστας ο εκσατοτε χρηστης θα μπορει να δει ολα τα projects η να αναζητησει ενα συγκεκριμενο project, προκειμενου να μπορει να εχει μια συνολικη εικονα των τρεχοντων η τον ολοκληρωμενων projects. Αναλογα με τον ρολο που εχει συνδεθει ο χρηστης θα μπορει να δει συγκεκριμενα projects στην λιστα του. Σαν Admin, o χρηστης θα μπορει να δει μια λιστα με ολα τα projects που εχουν εκχωρηθει στην εφαρμογη, ειτε αυτα ειναι στην φαση της αρχικοποιησης, ειτε αυτα ειναι ενεργα, ειτε ειναι ανενεργα. Η λιστα αυτη δεν θα ειναι ιδια για τον Manager και τον υπαλληλο, αφου θα περιεχει μονο οσα projects, στα οποια ο χρηστης εχει οριστει ως manager η ως υπαλληλος αντιστοιχα. Η λιστα περιεχει πληροφοριες για το καθε project, οπως το ονομα του, την διαρκεια του, τον συνολικο προυπολογισμο, τα συνολικα tasks των φασεων καθως και ενα πεδιο actions. Το πεδιο actions σχετιζεται με τις ενεργειες που μπορει ο χρηστης να κανει την δεδομενη για το project στιγμη, και συνδεεται αμεσα με τo status του.  
Το status ειναι μια μεταβλητη που οριζεται στο project αυτοματα στην καταχωρηση του κατα την πρωτη φαση αρχικοποιησης και αφορα την τρεχουσα κατασταση του project. Η χρηση της μεταβλητης απο την εφαρμογη υπαρχει καθαρα για σκοπους χειρισμου και οχι προβολης, οποτε και δεν ειναι αμεσα εμφανης στον χρηστη, παρα μονο απο το πεδιο actions και το τι αυτο εμφανιζει. To status απο 1 εως και 2, αναφερεται στην δευτερη και τριτη φαση αρχικοποιησης, αντιστοιχα. Το status με τιμη 3, υποδηλωνει οτι το project εκτελειται και το status με τιμη 4, υποδηλωνει οτι το project εχει τερματισει. Ετσι οταν το project status ειναι ενα ( 1), το πεδιο tasks της λιστας ειναι κενο ( αφου αρχικοποιειται στην επομενη φαση) και στο πεδιο actions εμφανιζονται δυο επιλογες, αυτη της εναρξης της δευτερης φασης αρχικοποιησης και αυτη της καταργησης του project. Οταν το project status εχει την τιμη δυο (2), πλεον το πεδιο tasks της λιστας θα δειχνει τον συνολικο αριθμο των tasks του project και στο πεδιο action μαζι με την επιλγοη καταργησης που αναφερθηκε νωριτερα, πλεον θα εμφανιζεται η επιλογη εναρξης της τριτης φασης αρχικοποιησης αντι αυτης της δευτερης. Η επιλογη αυτη ομως θα ειναι δυνατη μονο ως προς τον manager, συμφωνα με τις απαιτησεις της εφαρμογης, ενω στην αντιστοιχη λιστα του admin η λειτουργια αυτη θα ειναι απενεργοποιημενη (not clickable). Οταν το project status εχει την τιμη τρια (3), το πεδιο tasks θα δειχνει, αναλογα το progress του project, της εκτελεσμενες φασεις σε σχεση με τις συνολικες φασεις και στο πεδιο actions για τον Admin θα εμφανιζονται οι επιλογες προβολης και τερματισμου του project, ενω στις αντιστοιχες λιστες του manager και του υπαλληλου θα εμφανιζεται μονο η επιλογη προβολης του project. Τελος, οταν το project status εχει την τιμη 4 (4), το πεδιο tasks θα δειχνει ολα τα task τερματισμενα, ή ημιτελη σε περιπτωση προωρου τερματισμου και στο πεδιο action θα εμφανιζεται μονο η προβολη του project για ολους πλεον τους χρηστες την εφαρμογης.

**Προβολη ολων των υπαλληλων**

Με παρομοιο τροπο θα γινεται και η προβολη των υπαλληλων στον admin. Με την μορφη λιστας ο admin θα μπορει να δει ολους τους υπαλληλους η να αναζητησει εναν συγκεκριμενο υπαλληλο, προκειμενου να εχει μια συνολικη και γρηγορη εικονα των ενεργων η μη υπαλληλων της πλατφορμας. Η λιστα πρεπει να προβαλει πληροφοριες για τον καθε υπαλληλο, οπως το ονοματεπωνυμο του, το email του, την ημερομηνια γεννησης του, το Pid του και τα μισθολογικα του στοιχεια. Οπως και στην λιστα των projects, ετσι και στην λιστα των υπαλληλων περιεχεται και ενα πεδιο actions, το οποιο σχετιζεται με τις λειτουργιες που μπορει ο διαχειριστης να επιλεξει σχετικα με τον εκαστοτε υπαλληλο. Οι λειτουργιες εξ ορισμου ειναι τρεις, με πρωτη την προβολη του υπαλληλου, δευτερη την τροποποιηση των στοιχειων του και τριτη την απολυση-απαλλαγη του υπαλληλου. Αν ο υπαλληλος ειναι μη ενεργος στο συστημα, τοτε το πεδιο actions περιεχει μονο την προβολη του.

**Επεξεργασια Υπαλληλων**

Οπως αναφερθηκε παραπανω, στην λιστα των υπαλληλων, στο πεδιο actions εμφανιζεται η επιλογη επεξεργασιας του εκαστοτε υπαλληλου. Επειδη ομως η συγκεκριμενη λειτουργια αφορα εκτος του διαχειριστη και τον υπαλληλο, το συστημα παρεχει την λειτουργια επεξεργασιας και στο μενου του υπαλληλου. Στο συστημα το οποιο εχει αναπτυχθει, ο χρηστης θα δυναται να αλλαξει τις καταχωρημενες πληροφοριες του υπαλληλου. Με την παροδο του χρονου, στοιχεια όπως ο μισθος η το συμβολαιο, ενδεχεται να χρειαζεται να αλλαξουν, αλλα και για διορθωση λαθων κατα την αρχικοποιηση, οπως επισης και ο κωδικος προσβασης, ενδεχομενως για λογους ασφαλειας, οποτε και η τροποποιηση στοιχειων πρεπει να προσφερεται σαν λειτουργια. Επισης απο την πλευρα του υπαλληλου ως χρηστη, αυτος θα εχει την δυνατοτητα να τροποποιησει τα προσωπικα του στοιχεια, αλλα και τον κωδικο προσβασης του. Πιο συγκεκριμενα, στο κομματι της επεξεργασιας, οι δυνατοτητες που θα εχει ο admin θα ειναι μονο να αλλαξει τα μισθολογικα στοιχεια του εκαστοτε υπαλληλου, ενω ο υπαλληλος δεν θα μπορει να τροποποιησει τα σχετικα αυτα στοιχεια, παρα μονο οπως ειναι λογικο, μονο τα πεδια με τα στοιχεια του και τον κωδικο προσβασης. Αυτη η λειτουργια δινει στο συστημα μια πιο ολολκηρωμενη λυση οσον αφορα την διαχεριση προσωπικου και κατα συνεπεια των projects, καθως οι αλλαγες στα μισθολογικα του στοιχεια, επηρεαζουν τις πληρωμες του και το ρολο που μπορει να παρει στα project, αφου για παραδειγμα, οπως ειχε αναφερθει, με την αλλαγη ενος υπαλληλου σε full time, θα μπορει να αναλαμβανει χρεη manager.

**Προεπισκοπηση Υπαλληλου**  
  
Για λογους χρηστικοτητας, μια ακομα λειτουργια που προστεθηκε στην εφαρμογη ειναι η προεπσικοπηση ενος υπαλληλου. Οι πληρωμες για καθε υπαλληλο, που οπως αναφερθηκε νωριτερα, δημιουργουνται με την ολοκληρωση της τριτης φασης, πρεπει να ειναι προσβασιμες προς προβολη απο το συστημα. Η παρακολουθηση των πληρωμων ενος υπαλληλου, τοσο απο τον διαχεριστη οσο και απο τον ιδιο τον υπαλληλο, κρινεται αναγκαια, καθως θα παρεχεται και στους δυο, μια καθαρη εικονα ολων των πληρωμων και ολων των project που ο υπαλληλος εχει απασχοληθει συγκεντρωτικα.

Ο διαχειριστης μεσω της λιστας και απο το πεδιο actions που αναφερθηκε ηδη, θα μπορει να επιλεξει την ενεργεια προεπισκοπησης, ενω ο υπαλληλος, θα μπορει να κανει την αντιστοιχη επιλογη απο την αρχικη του σελιδα. Μαζι με τα στοιχεια του υπαλληλου προβαλλεται και μια λιστα με τις αντιστοιχες πληρωμες, οπου ο χρηστης αρχικα βλεπει τις βασικες λεπτομερειες, οπως το status της πληρωμης, το συνολικο ποσο πληρωμης αλλα και τις ωρες απασχολησης του, ενω στην συνεχεια μπορει να επιλεξει να δει περισσοτερες λεπτομερειες τις συγκεκριμενης πληρωμης, οπως την ημερομηνια εναρξης και ληξης της φασης της πληρωμης, το ονομα του project κλπ. Να σημειωθει οτι ο εκαστοτε χρηστης μπορει να κανιε αναζητηση οποιασδηποτε πληρωμης, με βαση το ονομα του project η του τυπου της φασης αυτης της πληρωμης.

**Αποδεσμευση Υπαλληλου**

Οπως αναλυθηκε η λειτουργια του τερματισμου ενος project, ετσι και για τον υπαλληλο εχει δοθει μια παρομοια λειτορυγια προς τον διαχεριστη. Ο διαχεριστης εχει την δυνατοτητα οποιαδηποτε στιγμη να αποδεσμευσει τον υπαλληλο απο τα καθηκοντα του, οριζοντας τον ως πλεον μη ενεργο στο συστημα. Οταν ενας υπαλληλος εχει οριστει ως μη ενεργος, σημαινει ειτε την παραιτηση του ειτε την απολυση του και εν συνεχεια δεν θα ειναι διαθεσιμος για επιλογη και αναληψη καποιου project κατα την αρχικοποιηση του απο τους εκαστοτε managers. Απο την λιστα των υπαλληλων, στο πεδιο actions ο χρηστης θα μπορει να επιλεξει την συγκεκριμενη ενεργεια και εφοσον την επιλεξει θα επιβεβαιωνει την επιλογη μεσω του modal που προβαλλεται. Να σημειωθει πως για λογους ιστορικου και ρποσβασης σε ηδη καταχωρημενες πληροφοριες του, ο υπαλληλος δεν διαγραφεται εντελως απο το συστημα. Με το περας αυτης της λειτουργιας, το state του υπαλληλου ανανεωνεται και πλεον στην λιστα των εργαζομενων, μπορει να γινει μονο προβολη των ηδη υπαρχοντων στοιχειων του.

**Παρακολουθηση Project**

Για να θεωρηθει πετυχημενη η εκαστοτε εφαρμογη που παραγεται, καθε συγχρονη εταιρεια που αναλαμβανει την διαχειριση projects, θα πρεπει να δινει στον χρηστη την δυνατοτητα να παρακολουθει και να ενημερωνεται λεπτομερως για το project στο οποιο εργαζεται. Η παρακολουθηση των project, ειναι ισως η πιο σημαντικη λειτουργια μιας τετοιας εφαρμογης, αφου ολες αυτες οι πληροφοριες που συλλεχθηκαν κατα τα σταδια αρχικοποιησης προβαλλονται συγκεντρωτικα και με ευκολια στον χρηστη. Η λειτουργια αφορα ολους τους χρηστες του συστηματος, συμφωνα παντα με την λιστα των projects που αφορουν τον καθενα και προκειμενου να ειναι σε θεση να κατανοησουν πληρως την φυση του project, ιδανικη κριθηκε η χρηση αναλογων διαγραμματων. Μεσω των διαγραμματων δοθηκε ιδιαιτερη εμφαση στην χρονικη αναλυση του project, αλλα και στο πως ειναι δομημενες οι φασεις και οι υπαλληλοι σε αυτο.

***Hierarchy Tree***

Το πρωτο διαγραμμα που παρεχεται στον χρηστη ειναι ενα διαγραμμα τυπου Hierarchy Tree, οπου χωριζει και αναλυει ολο το προτζεκτ σε τρια επιπεδα. Το πρωτο επιπεδο αναφερεται στον πρωτο κομβο του διαγραμματος που ειναι το ιδιο το προτζεκτ και πιο συγκεκριμενα το ονομα του και επεκτεινεται στους κομβους του δευτερου επιπεδου. Οι κομβοι του δευτερου επιπεδου αναφερονται στον τυπο των φασεων του συγκεκριμενου project και καθε φαση στην συνεχεια επεκτεινεται στους αντιστοιχους υπαλληλους της, με το ονοματεπωνυμο τους και αυτοι παρομοια, σαν κομβοι του τριτου επιπεδου. Παραλληλα η εφαρμογη παρεχει στον χρηστη την δυνατοτητα να επιλεξει οποιονδηποτε απο αυτους τους κομβους, ετσι ωστε αν θελει να μπορει να δει περισσοτερες λεπτομερειες σχετικα με το αντικειμενο του κομβου. Αν ο χρηστης για παραδειγμα επιλεξει τον πρωτο κομβο, οι πληροφοριες που θα του προβαλλονται θα ειναι αναφορικαμε το συγκεκριμενο project, οπως το budget, o manager αλλα και το ποσοστο ολοκληρωσης του. Αν επιλεξει καποιον απο τους κομβους του δευτερου επιπεδου, θα του προβληθουν οι αντιστοιχες πληροφοριες της φασης που επελεξε. Αν τελος επιλεξει καποιον απο τους κομβους του τριτου επιπεδου, αυτοι θα προβαλλουν τις αντιστοιχες πληροφοριες του υπαλληλου μαζι με μια γρηγορη επισκοπηση της πληρωμης του για την εκαστοτε φαση. Συνολικα, το συγκεκριμενο διαγραμμα ειναι πολυ χρησιμο για οποιον το παρακολουθει, αφου μπορει αμεσα να δει ποιοι εργαζομενοι δουλευουν σε καθε φαση, ποσες ειναι οι φασεις και ποσοι ειναι συνολικα οι εργαζομενοι.

***Gannt Diagram***

Το δευτερο διαγραμμα που παρεχεται εχει την μορφη του γνωστου διαγραμματος gannt. Η εφαρμογη παρεχει στον χρηστη ενα διαγραμμα με βαση τις ημερομηνιες διεξαγωγης των εκαστοτε φασεων του project. Ετσι για καθε φαση, το διαγραμμα δειχνει μια μπαρα με αρχη την ημερομηνια εναρξης και τελος την ημερομηνια ληξης. Με τον τροπο επισης που κατασκευαστηκε το διαγραμμα gannt, η κατασταση των tasks της καθε φασης αντανακλουν χρωματικα στην μπαρα προβαλλοντας και την προοδο της εκαστοτε φασης. Οπως ειχε αναφερθει, οι φασεις μπορει να ταυτιζονται με σπειρες η επαναληψεις, με βαση το μοντελο spiral η agile, οποτε το συγκεκριμενο διαγραμμα κανει ακομα πιο κατανοητο και βολικο τον τροπο που λειτουργει καθε μοντελο και κανει πιο ξεκαθαρο το ποτε λαμβανει μερος το καθε “υποκομματι” του project.

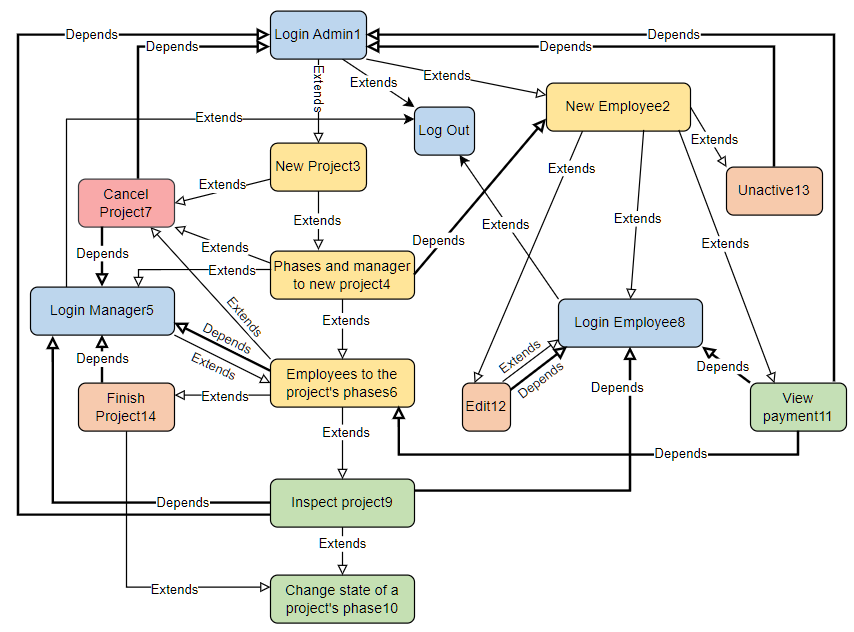
**Παρακολουθηση Προοδου**

Χαρη στην παραπανω λειτουργια και μεσω του περιβαλλοντος που εχει σχεδιαστει για την παρακολουθηση των project, η εφαρμογη παρεχει την δυνατοτητα στον χρηστη και στην συγκεκριμενη περιπτωση του manager, να παρακολουθει και να σημειωνει την προοδο ενος project. Ο σκοπος αυτης της λειτουργιας ειναι να προσφερει στο manager μεγαλυτερη διαδραστικοτητα σε σχεση με την πλατφορμα αλλα και σε σχεση με τις υποχρεωσεις που εχει ως υπευθυνος εργου απεναντι στο ιδιο το εργο. Ο χρηστης πρεπει σε οποιαδηποτε φαση της υλοποιησης να μπορει να ενημερωνει το status και την προοδο του εργου, προκειμνεου και οι υπολοιποι χρηστες να μπορουν να γνωριζουν σε ποια φαση βρισκονται αναφορικα με τα παραδοτεα και τα tasks. Μεσω του πρωτου διαγραμματος της παρακολουθησης project, ο manager επιλεγει μια απο τις φασεις- κομβους του δευτερου επιπεδου και στις επιπλεον λεπτομερειες που προβαλλονται τους μεφανιζεται και ενα κουμπι. Το κουμπι αυτο εχει να κανει με τον τερματισμο ενος task και την εναρξη του επομενου. Με την επιλογη του τερματισμου του εκαστοτε task, ξεκιναει το επομενο η τερματιζεται συνολικα η φαση, αν προκειται για το τελευταιο task. Απο την αλλαγη αυτη προκυπτει οτι και το διαγραμμα gannt που ειχε αναφερθει παραπανω, θα ανανεωθει αυτοματα και αμεσα, καθως οποιαδηποτε αλλαγη task θα επεκταθει και στην μπαρα προοδου της συγκεκριμενης φασης. Ενημερωνεται επισης και η λιστα των project, αφου ενα απο τα πεδια της αναφερεται στην προοδο του Project συνολικα. Γενικα, η χρηση πληροφοριων σχετικα με την προοδο του εργου ειναι συχνη στην πλατφορμα, αντανακλωντας ετσι, στην αναγκη μιας λειτουργιας που χειριζεται η τερματιζει υποκομματια του εργου. Ετσι διατηρειται η συνεπεια των παραδοτεων και των χρονικων πλαισιων των φασεων και η υλοποιηση του εργου ειναι πιο διαχειρισιμη οσον αφορα τον χρονο.

**Αποσυνδεση Χρηστη**

Για ευνοητους λογους, εχει δοθει στους χρηστες της πλατφορμας και η δυνατοτητα αποσυνδεσης. Η αναγκη υπαρξης αυτης της δυνατοτητας προκυπτει απο το γεγονος οτι η πλατφορμα φιλοξενει και συνδεδεμενους χρηστες. Ο συνδεδεμενος χρηστης σε οποιοδηποτε σημειο της εφαρμογης εχει την δυνατοτητα ειτε να αποσυνδεθει εντελως ειτε να θελει να εισελθει με αλλο ρολο απο αυτο που ηδη εχει. Σε οποιαδηποτε περιπτωση, πρεπει το συστημα να παρεχει και αυτη την δυνατοτητα.

**3.3** **Περιπτωσεις Χρησης ( Use Cases )**

Από την αναλυση των απαιτησεων του συστηματος προκυπτει οτι οι διαφορετικοι χρηστες της εφαρμογης για να εκτελεσουν μια λειτουργια, προχωρουν σε μια ακολουθια συγκεκριμενων ενεργειων και βηματων. Αυτές οι ενέργειες ονομάζονται περιπτώσεις χρήσης και ορίζονται από μια λίστα ενεργειών ή γεγονοτων που ορίζουν την αλληλεπίδραση μεταξύ ενός χρήστη και του συστήματος για την ολοκληρωση μιας λειτουργιας.

Ακολουθούν δεκα τέσσερις (15) περιπτώσεις χρήσης της πλατφόρμας και παρουσιάζονται σε μορφή πινάκων που παρέχουν όλες τις απαραίτητες πληροφορίες για να γίνουν κατανοητές, να μοντελοποιηθουν και τελικά να εφαρμοστούν βοηθώντας στο σχεδιασμό και την ανάπτυξη της πλατφόρμας.

Use Case (UC01) : Login σαν Διαχειριστης

|  |  |
| --- | --- |
| Actor | Admin |
| Brief | Ο διαχειριστης μπορει να συνδεθει στην εφαρμογη και να χρησιμοποιησει τις λειτουργιες της. |
| Προυποθεση | Να υπαρχει admin στο συστημα ( οριζεται με το συστημα ) |
| Αποτελεσμα | Ο χρηστης συνδεεται στην εφαρμογη και μπορει να δει το dashboard. |
| Βασικη Ροη | * 1. Ο διαχειριστης βλεπει την φορμα login στην αρχικη σελιδα   2. Ο διαχειριστης συμπληρωνει το email και το κωδικο του και επιλεγει τον ρολο “admin”.   3. Ο χρηστης συνδεεται στην εφαρμογη και ανακατευθυνεται στο dashboard. |
| Εναλλακτικη Ροη | Τα στοιχεια του διαχειριστη είναι λανθασμενα και μπορει να προσπαθησει να συνδεθει ξανα με νεα στοιχεια.  Δεν εγινε επιλογη του ρολου “admin”. |
| Εξαρταται από | ---- |
| Εκτεινεται σε | UC02, UC03 |

Use Case (UC02) : Καταχωρηση Νεου Υπαλληλου

|  |  |
| --- | --- |
| Actor | Admin |
| Brief | Ο διαχειριστης μπορει να καταχωρησει έναν νέο υπαλληλο στην πλατφορμα. |
| Προυποθεση | Ο χρηστης να είναι συνδεδεμενος ως admin και να μην εχει καταχωρησει καποιον υπαλληλο με τα ιδια στοιχεια στο παρελθον. |
| Αποτελεσμα | Ο διαχειριστης καταχωρει ένα νέο υπαλληλο στην εφαρμογη. |
| Βασικη Ροη | * 1. Ο διαχειριστης συνδεεται στην πλατφορμα και κανει επιλογη του Employees από το αρχικο Dashboard.   2. Ο διαχεριστης ανακατευθυνεται στο μενου των Employees και επιλεγει την Προσθηκη Νεου Υπαλληλου.   3. Ο διαχειριστης συμπληρωνει τα απαιτουμενα στοιχεια από το αναδυομενο modal και επιλεγει το κουμπι “Τελος”.   4. Ο διαχεριστης ανακατευθυνεται παλι στο μενου Employees και μπορει τωρα να δει τον νέο υπαλληλο στην λιστα των Employees. |
| Εναλλακτικη Ροη | Ο διαχειριστης επιλεγει “Ακυρωση” κατά την διαρκεια της εκχωρησης. |
| Εξαρταται απο | UC01 |
| Εκτεινεται σε | UC08, UC11, UC12, UC13 |

Use Case (UC03) : Καταχωρηση Νεου Project (1η φαση)

|  |  |
| --- | --- |
| Actor | Admin |
| Brief | Ο διαχειριστης μπορει να καταχωρησει ένα νέο project στην πλατφορμα. |
| Προυποθεση | Ο χρηστης να είναι συνδεδεμενος ως admin και να μην εχει καταχωρησει project με τα ιδια στοιχεια στο παρελθον. |
| Αποτελεσμα | Ο διαχειριστης καταχωρει ένα νέο project (1η φαση) στην εφαρμογη και ενεργοποιειται το επομενο σταδιο αρχικοποιησης. |
| Βασικη Ροη | * 1. Ο διαχειριστης συνδεεται στην πλατφορμα και κανει επιλογη του Projects από το αρχικο Dashboard.   2. Ο χρηστης ανακατευθυνεται στο μενου των Projects και επιλεγει την Προσθηκη Νεου Project.   3. Ο χρηστης συμπληρωνει τα απαιτουμενα στοιχεια από το αναδυομενο modal και επιλεγει “Εκχώρηση”   4. Ο διαχειριστης ανακατευθυνεται στο μενου των Projects και μπορει τωρα να δει το νέο project στην λιστα των Projects. |
| Εναλλακτικη Ροη | Ο χρηστης επιλεγει “Ακυρωση” κατά την διαρκεια της εκχωρησης. |
| Εξαρταται απο | UC01 |
| Εκτεινεται σε | UC04, UC07 |

Use Case (UC04) : Καταχωρηση Φασεων και Manager σε Project (2η φαση)

|  |  |
| --- | --- |
| Actor | Admin |
| Brief | Ο admin μπορει να ορισει manager και φασεις για ενα συγκεκριμενο project κατά την διαρκεια της αρχικοποιησης. |
| Προυποθεση | Ο χρηστης να είναι συνδεδεμενος ως admin.  Ο admin να εχει καταχωρησει υπαλληλους στο συστημα και να εχει ολοκληρωσει το πρωτο μερος της αρχικοποιησης του συγκεκριμενου project. |
| Αποτελεσμα | Ο χρηστης καταχωρει τις απαιτουμενες φασεις και καποιον υπαλληλο ως manager του συγκεκριμενου προτζεκτ (2η φαση) και ενεργοποιειται το επομενο σταδιο αρχικοποιησης. |
| Βασικη Ροη | * 1. Ο διαχειριστης συνδεεται στην πλατφορμα και κανει επιλογη του Projects από το αρχικο Dashboard.   2. Ο διαχειριστης ανακατευθυνεται στο μενου με την λιστα όλων των project και αναζητει ένα συγκεκριμενο.   3. Ο διαχεριστης επιλεγει το αναλογο κουμπι αρχικοποιησης (2ης φασης) και αναδυεται ένα modal.   4. Ο διαχεριστης επιλεγει, από την λιστα που εμφανιζεται, τις φασεις του project. Μπορει να δημιουργησει και custom φασεις σε αυτό το σημειο, αν θελει και επιλεγει συνεχεια.   5. Ο χρηστης αρχικοποιει με πληροφοριες την κάθε φαση και επιλεγει το κουμπι “Συνεχεια”.   6. Ο χρηστης διαλεγει τον μανατζερ από μια λιστα υπαλληλων και επιλεγει “Τελος”. |
| Εναλλακτικη Ροη | Ο χρηστης επιλεγει “Ακυρωση” οποιαδηποτε στιγμη της διαδικασιας αρχικοποιησης των φασεων. |
| Εξαρταται απο | UC03, UC02 |
| Εκτεινεται σε | UC05, UC06, UC07 |

Use Case (UC05) : Login σαν Manager

|  |  |
| --- | --- |
| Actor | Manager |
| Brief | Ο μανατζερ μπορει να συνδεθει στην εφαρμογη και να χρησιμοποιησει τις λειτουργιες της. |
| Προυποθεση | Ο συγκεκριμενος υπαλληλος να εχει καθηκοντα και δικαιωματα μανατζερ σε ένα τουλαχιστον Project. |
| Αποτελεσμα | Ο manager συνδεεται στην εφαρμογη και μπορει να δει τα project στα οποια συμμετεχει. |
| Βασικη Ροη | * 1. Ο χρηστης βλεπει την φορμα login στην αρχικη σελιδα   2. Ο χρηστης αρχικα συμπληρωνει το email και το κωδικο του και στην συνεχεια επιλεγει τον ρολο του “manager”.   3. Ο manager επιλεγει το κουμπι “Login”, συνδεεται στην εφαρμογη και ανακατευθυνεται στην λιστα με τα projects που συμμετεχει. |
| Εναλλακτικη Ροη | Τα στοιχεια του manager είναι λανθασμενα και μπορει να προσπαθησει να συνδεθει ξανα με νεα στοιχεια.  Δεν εγινε επιλογη του ρολου “manager”.  Ο συγκεκριμενος υπαλληλος δεν είναι υπευθυνος σε κανενα project και ετσι η λιστα των projects είναι κενη. |
| Εξαρταται από | UC04 |
| Εκτεινεται σε | UC06 |

Use Case (UC06) : Καταχωρηση Υπαλληλων σε Φασεις ( 3η φαση)

|  |  |
| --- | --- |
| Actor | Manager |
| Brief | Ο manager μπορει να καταχωρησει υπαλληλους σε κάθε φαση ενός project. |
| Προυποθεση | Ο χρηστης να είναι συνδεδεμενος ως manager.  Ο admin να εχει καταχωρησει υπαλληλους στο συστημα καθως και να εχει ολοκληρωσει το μερος της αρχικοποιησης που τον αφορα για το συγκεκριμενο project.  Να εχει οριστει ο συγκεκριμενος manager ως υπευθυνος του συγκεκριμενου project από τον admin. |
| Αποτελεσμα | Ο manager καταχωρει έναν η περισσοτερους υπαλληλους σε κάθε φαση του project (3η φαση) και τερματιζει η αρχικοποιηση του συγκεκριμενου Project. |
| Βασικη Ροη | * 1. Ο manager συνδεεται στην πλατφορμα και ανακατευθυνεται στην λιστα με τα projects που συμμετεχει.   2. Ο manager βλεπει την λιστα με όλα τα projects και αναζητει ενα συγκεκριμενο.   3. Ο manager επιλεγει το αναλογο κουμπι αρχικοποιησης (3ης φασης) και αναδυεται ένα modal.   4. Ο manager επιλεγει, από την λιστα που εμφανιζεται, τους υπαλληλους για την συγκεκριμενη φαση και επιλεγει το κουμπι “Συνεχεια”.   5. Η διαδικασια επαναλαμβανεται τοσες φορες, οσες και οι φασεις του project και ο manager επιλεγει το κουμπι “Τελος”. |
| Εναλλακτικη Ροη | Ο manager δεν μπορει να επιλεξει υπαλληλους γιατι δεν υπαρχoυν διαθεσιμοι για το χρονικο διαστημα της εκαστοτε φασης. |
| Εξαρταται απο | UC05, UC04 |
| Εκτεινεται σε | UC07, UC09, UC14 |

Use Case (UC07) : Καταργηση Project

|  |  |
| --- | --- |
| Actor | Admin, Manager |
| Brief | Ο χρηστης μπορει να καταργησει ένα συγκεκριμενο project που βρισκεται ακομα στην διαδικασια της αρχικοποιησης. Γινεται ολικη διαγραφη ολων των στοιχειων του project από το συστημα |
| Προυποθεση | Ο χρηστης να είναι συνδεδεμενος ως Admin ή Manager.  Ο admin να εχει καταχωρησει τουλαχιστον στην 1η φαση του το συγκεκριμενο προτζεκτ.  Το συγκεκριμενο Project να βρισκεται στην 3η φαση, για να μπορει να το καταργησει ο manager. |
| Αποτελεσμα | Ο χρηστης καταργει ένα συγκεκριμενο προτζεκτ από την εφαρμογη. |
| Βασικη Ροη | * 1. Ο χρηστης συνδεεται στην πλατφορμα και αν είναι ο admin επιλεγει από το αρχικο Dashboard το κουμπι Projects, αλλιως στην αντιθετη περιπτωση αν είναι ο μανατζερ ανακατευθυνεται στην λιστα με τα projects που συμμετεχει   2. Ο χρηστης βλεπει την λιστα με όλα τα προτζεκτς και αναζητει ενα συγκεκριμενο.   3. Ο χρηστης επιλεγει το κουμπι καταργησης του συγκεριμενου προτζεκτ και αναδυεται ένα modal.   4. Ο χρηστης επιβεβαιωνει με την επιλογη του και ανακατευθυνεται στο μενου Projects.   5. Ο χρηστης μπορει τωρα να δει την ανανεωμενη λιστα των projects, χωρις το συγκεκριμενο. |
| Εναλλακτικη Ροη | Ο admin δεν βρισκει το προτζεκτ γιατι δεν υπαρχει στο συστημα.  Ο manager δεν βρισκει το project γιατι δεν είναι ορισμενος ως υπευθυνου του συγκεκριμενου εργου.  Ο χρηστης επιλεγει “Ακυρωση” κατά την προτροπη επιβεβαιωση της επιλογης του. |
| Εξαρταται απο | UC01/UC05, UC03/UC04/UC06 |
| Εκτεινεται σε | ---- |

Use Case (UC08) : Login σαν Employee

|  |  |
| --- | --- |
| Actor | Employee |
| Brief | Ο υπαλληλος μπορει να συνδεθει στην εφαρμογη και να χρησιμοποιησει τις λειτουργιες της. |
| Προυποθεση | Να εχει καταχωρηθει ο συγκεκριμενος υπαλληλος από τον admin στο συστημα στο παρελθον. |
| Αποτελεσμα | Ο υπαλληλος συνδεεται στην εφαρμογη και μπορει να δει τα project στα οποια συμμετεχει. |
| Βασικη Ροη | * 1. Ο υπαλληλος βλεπει την φορμα login στην αρχικη σελιδα   2. Ο υπαλληλος αρχικα συμπληρωνει το email και το κωδικο του και στην συνεχεια επιλεγει τον ρολο του “employee”.   3. Ο υπαλληλος επιλεγει το κουμπι “Login”, συνδεετεαι στην εφαρμογη και ανακατευθυνεται στην λιστα με τα projects που συμμετεχει. |
| Εναλλακτικη Ροη | Τα στοιχεια του υπαλληλου είναι λανθασμενα και μπορει να προσπαθησει να συνδεθει ξανα με νεα στοιχεια.  Δεν εγινε επιλογη του ρολου “employee”. |
| Εξαρταται από | UC02 |
| Εκτεινεται σε | ---- |

Use Case (UC09) : Προεπισκοπηση ενός Project

|  |  |
| --- | --- |
| Actor | Admin, Manager, Employee |
| Brief | Ο χρηστης μπορει να δει ολες τις σχετικες λεπτομερειες για ένα συγκεκριμενου project. |
| Προυποθεση | Ο χρηστης να είναι συνδεδεμενος στο συστημα.  Ο admin να εχει καταχωρησει το συγκεκριμενο προτζεκτ στο παρελθον.  Ο admin και ο manager να εχουν τελειωσει με την διαδικασια αρχικοποιησης του project.  Αν ο χρηστης συνδεθει σαν employee η manager, να συμμετεχει στο συγκεκριμενο project. |
| Αποτελεσμα | Ο χρηστης βλεπει ολες τις πληρoφοριες ενός project. |
| Βασικη Ροη | * 1. Ο χρηστης συνδεεται στην πλατφορμα ως admin και από το αρχικο Dashboard επιλεγει το κουμπι Projects και ανακατευθυνεται στην λιστα με όλα τα projects του συστηματος.  Η χρηστης συνδεεται στην πλατφορμα ως μανατζερ η employee και ανακατευθυνεται αυτοματα στην λιστα με τα projects που συμμετεχει   2. Ο χρηστης βλεπει την αντιστοιχη λιστα με τα προτζεκτς και αναζητει ενα συγκεκριμενο προτζεκτ.   3. Ο χρηστης επιλεγει το αντιστοιχο κουμπι προβολης και αναδυεται ένα modal.   4. Ο χρηστης βλεπει τις πληροφοριες του project και αναλυτικα διαγραμματα αυτου. |
| Εναλλακτικη Ροη | Ο χρηστης δεν μπορει να επιλεξει ένα συγκεκριμενο project γιατι δεν εχει τελειωσει η αρχικοποιηση του.  Ο manager η ο υπαλληλος δεν μπορουν να βρουν το συγκεκριμενο project γιατι δεν εχουν λαβει μερος σε αυτο |
| Εξαρταται από | UC06, UC01/UC05/UC08 |
| Εκτεινεται σε | UC10 |

Use Case (UC10) : Ολοκληρωση ενός task μιας φασης

|  |  |
| --- | --- |
| Actor | Manager |
| Brief | Ο manager μπορει να αλλαξει την κατασταση μιας φασης επιβεβαιωνοντας την ολοκληρωση ενός task της. |
| Προυποθεση | Ο χρηστης να είναι συνδεδεμενος ως Manager και να μην εχει ολοκληρωσει όλα τα task της συγκεκριμενης φασης.  Ο admin και ο manager να εχουν τελειωσει με την διαδικασια αρχικοποιησης του project. |
| Αποτελεσμα | Ο manager τερματιζει το task μιας φασης και ειτε η φαση τερματιζει ειτε συνεχιζει στο επομενο task. |
| Βασικη Ροη | * 1. Ο manager συνδεεται στην πλατφορμα και ανακατευθυνεται στην λιστα με τα projects που συμμετεχει.   2. Ο manager βλεπει την λιστα με όλα τα projects και αναζητει ενα συγκεκριμενο.   3. Ο manager επιλεγει το κουμπι “Προβολη Project” και αναδυεται ένα modal.   4. Ο manager επιλεγει μια φαση και στην συνεχεια με το κουμπι “Next” τερματιζει το παρον task της φασης ενός συγκεκριμενου project. |
| Εναλλακτικη Ροη | Ο manager δεν μπορει να επιλεξει ένα συγκεκριμενο project γιατι ειτε δεν εχει τελειωσει ειτε δεν εχει ξεκινησει η αρχικοποιηση του.  Ο manager δεν μπορει να επιλεξει “Next” στο επομενο task, επειδη όλα τα task τις συγκεκριμενης φασης εχουν ολοκληρωθει. |
| Εξαρταται από | UC09 |
| Εκτεινεται σε | ---- |

Use Case (UC11) : Προεπισκοπηση Πληρωμων ενός υπαλληλου

|  |  |
| --- | --- |
| Actor | Admin, Υπαλληλος |
| Brief | Ο χρηστης μπορει να δει ολες τις πληρωμες ενός υπαλληλου. |
| Προυποθεση | Ο χρηστης να είναι συνδεδεμενος ως Admin ή Employee.  Ο Admin να εχει καταχωρησει τον συγκεκριμενο υπαλληλο στο παρελθον και ο υπαλληλος να εχει οριστει σε καποιο project στο παρελθον. |
| Αποτελεσμα | Ο χρηστης βλεπει ολες τις πληρωμες ή μια συγκεκριμενη για έναν υπαλληλο. |
| Βασικη Ροη | * 1. a) Ο χρηστης συνδεεται στην πλατφορμα ως admin και από το αρχικο Dashboard κανει επιλογη του Employees και ανακατευθυνεται στην λιστα με όλους τους employees του συστηματος.  b) Ο χρηστης συνδεεται στην πλατφορμα ως employee και ανακατευθυνεται αυτοματα στο Employee μενου.   2. a) Ο admin αναζητει εναν συγκεκριμενο υπαλληλο, επιλεγει του κουμπι “Προβολη” και αναδυεται ένα modal.   b) O υπαλληλος επιλεγει από το μενου του το κουμπι “Προβολη” και αναδυεται ένα modal.   * 1. Ο χρηστης μπορει να δει τωρα την λιστα με ολες τις πληρωμες του υπαλληλου (του ιδιου αν είναι logged in ως employee)   2. Ο χρηστης μπορει να αναζητησει μια συγκεκριμενη πληρωμη και να την δει λεπτομερως. |
| Εναλλακτικη Ροη | Ο admin δεν βρισκει τον συγκεκριμενο υπαλληλο γιατι δεν υπαρχει στο συστημα.  Ο admin η ο υπαλληλος, δεν μπορει να δει πληρωμες για ένα συγκεκριμενο υπαλληλο η για τον ιδιο αντιστοιχα γιατι δεν υπαρχουν καταχωρημενες στο συστημα. |
| Εξαρταται από | UC01/UC08, UC06 |
| Εκτεινεται σε | ---- |

Use Case (UC12) : Επεξεργασια υπαλληλου

|  |  |
| --- | --- |
| Actor | Employee |
| Brief | Ο υπαλληλος μπορει να επεξεργαστει τα ηδη καταχωρημενα στοιχεια του. |
| Προυποθεση | Ο χρηστης να είναι συνδεδεμενος ως Employee και να εχει καταχωρηθει από τον Admin στο παρελθον. |
| Αποτελεσμα | Ο υπαλληλος επεξεργαστηκε και καταχωρησε τα στοιχεια του στην πλατφορμα. |
| Βασικη Ροη | * 1. Ο υπαλληλος συνδεεται στην πλατφορμα ως employee και ανακατευθυνεται αυτοματα στο Employee μενου.   2. Ο υπαλληλος επιλεγει το κουμπι “Edit” και αναδυεται ένα modal.   3. Ο υπαλληλος συμπληρωνει τα στοιχεια προς αλλαγη.   4. Ο υπαλληλος επιλεγει το κουμπι “Τελος” και ανακατευθυνεται στο μενου Employees. |
| Εναλλακτικη Ροη | Ο υπαλληλος επιλεγει “Ακυρωση” κατά την διαρκεια της επεξεργασιας. |
| Εξαρταται από | UC08 |
| Εκτεινεται σε | UC08 |

Use Case (UC13) : Αποδεσμευση υπαλληλου

|  |  |
| --- | --- |
| Actor | Admin |
| Brief | Ο διαχειριστης μπορει να αποδεσμευσει εναν ενεργο υπαλληλο. |
| Προυποθεση | Ο χρηστης να είναι συνδεδεμενος ως Admin και να εχει καταχωρησει τον υπαλληλο στο παρελθον. |
| Αποτελεσμα | Ο διαχειριστης αποδεσμευει έναν συγκεκριμενο υπαλληλο από την εφαρμογη. Ωστοσο δεν διαγραφεται από το συστημα, παρα μονο το status του employee αλλαζει σε μη ενεργο. |
| Βασικη Ροη | * 1. Ο διαχειριστης συνδεεται στην πλατφορμα ως admin και από το αρχικο Dashboard κανει επιλογη του Employees και ανακατευθυνεται στην λιστα με όλους τους employees του συστηματος.   2. Ο διαχειριστης αναζητει εναν συγκεκριμενο υπαλληλο.   3. Ο διαχειριστης επιλεγει το κουμπι “αποδεσμευση” του συγκεριμενου υπαλληλου και αναδυεται ένα modal.   4. Ο διαχειριστης επιβεβαιωνει με την επιλογη του και ανακατευθυνεται στο μενου Employees. |
| Εναλλακτικη Ροη | Ο χρηστης επιλεγει “Ακυρωση” κατά την επιβεβαιωση της επιλογης του. |
| Εξαρταται από | UC01, UC02 |
| Εκτεινεται σε | ---- |

Use Case (UC14) : Τερματισμος Project

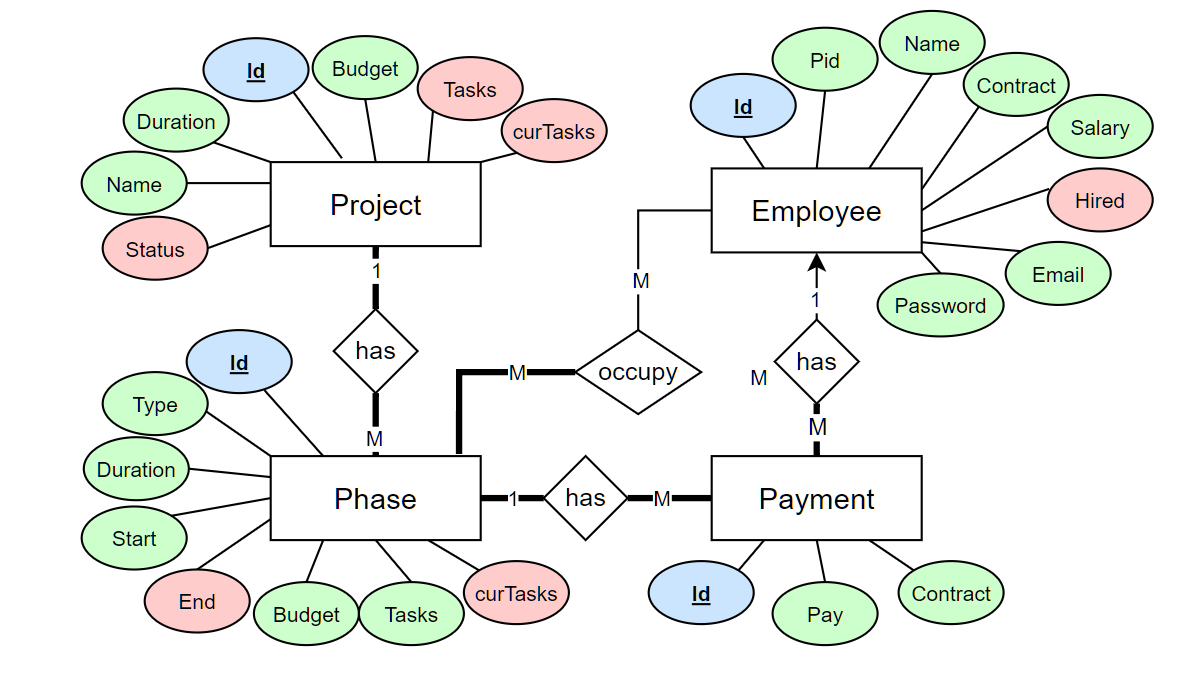
|  |  |
| --- | --- |
| Actor | Manager |
| Brief | Ο υπευθυνος μπορει να τερματισει οριστικα η προωρα ένα συγκεκριμενο project από το συστημα. |
| Προυποθεση | Ο χρηστης να είναι συνδεδεμενος ως Manager και να εχει αρχικοποιησει με υπαλληλους τις φασεις του συγκεκριμενου project. |
| Αποτελεσμα | Ο manager τερματιζει ένα συγκεκριμενο προτζεκτ από την εφαρμογη. Ωστοσο δεν διαγραφεται από το συστημα, παρα μονο το status του project αλλαζει σε μη ενεργο. |
| Βασικη Ροη | * 1. Ο manager συνδεεται στην πλατφορμα και ανακατευθυνεται στην λιστα με τα projects που συμμετεχει.   2. Ο manager αναζητει ενα συγκεκριμενο προτζεκτ από την λιστα που εμφανιζεται, επιλεγει το κουμπι ”τερματισμου” του συγκεριμενου προτζεκτ και αναδυεται ένα modal.   3. Ο manager επιβεβαιωνει την επιλογη του και ανακατευθυνεται στο μενου Projects.   4. Ο manager μπορει τωρα να δει την ανανεωμενη λιστα των projects. |
| Εναλλακτικη Ροη | Ο χρηστης επιλεγει “Ακυρωση” κατά την προτροπη επιβεβαιωση της επιλογης του. |
| Εξαρταται από | UC05, UC06 |
| Εκτεινεται σε | UC10 |

Use Case (UC15) : Log Out Χρηστη

|  |  |
| --- | --- |
| Actor | Admin, Manager, Employee |
| Brief | Ο χρηστης μπορει να τερματισει οριστικα η προωρα ένα συγκεκριμενο project από το συστημα. |
| Προυποθεση | Ο χρηστης να είναι συνδεδεμενος ως Manager και να εχει αρχικοποιησει με υπαλληλους τις φασεις του συγκεκριμενου project. |
| Αποτελεσμα | Ο manager τερματιζει ένα συγκεκριμενο προτζεκτ από την εφαρμογη. Ωστοσο δεν διαγραφεται από το συστημα, παρα μονο το status του project αλλαζει σε μη ενεργο. |
| Βασικη Ροη | * 1. Ο manager συνδεεται στην πλατφορμα και ανακατευθυνεται στην λιστα με τα projects που συμμετεχει.   2. Ο manager αναζητει ενα συγκεκριμενο προτζεκτ από την λιστα που εμφανιζεται, επιλεγει το κουμπι ”τερματισμου” του συγκεριμενου προτζεκτ και αναδυεται ένα modal.   3. Ο manager επιβεβαιωνει την επιλογη του και ανακατευθυνεται στο μενου Projects.   4. Ο manager μπορει τωρα να δει την ανανεωμενη λιστα των projects. |
| Εναλλακτικη Ροη | Ο χρηστης επιλεγει “Ακυρωση” κατά την προτροπη επιβεβαιωση της επιλογης του. |
| Εξαρταται από | UC05, UC06 |
| Εκτεινεται σε | UC10 |

***3.5 Βαση δεδομένων***

Για την μοντελοποιηση των δεδομενων και της δομης μιας βασης δεδομενων είναι ευρεως διαδεδομενη η χρηση του διαγραμματος οντοτήτων-συσχετίσεων (ER diagram), καθως αποτελει σημαντική βοήθεια στην κατανόηση και στην αναπαρασταση των αναγκαιων μεταβλητων που αποθηκευονται στο συστημα. Το διάγραμμα ER χρησιμοποιείται συνήθως στο πρώτο στάδιο της διαδικασίας σχεδιασμού βάσης δεδομένων, κατά την ανάλυση των απαιτήσεων της βάσης δεδομένων και σκοπός του είναι να κατανοήσει τις ανάγκες των χρηστών και να δημιουργήσει ένα μοντέλο δεδομένων που να τις ικανοποιεί. Είναι ένα σημαντικό εργαλείο για τον σχεδιασμό βάσης δεδομένων επειδή παρέχει μια οπτική αναπαράσταση των δεδομένων. Αυτό βοηθά τους σχεδιαστές να κατανοήσουν καλύτερα τα δεδομένα και να δημιουργήσουν ένα μοντέλο που είναι αποτελεσματικό και αποδοτικό. Καθως η εφαρμογη χρησιμοποιει ένα σχεσιακο μοντελο βασης δεδομενων, το διαγραμμα ER θα δωσει μια πιο ξεκαθαρη εικονα μεταξυ των οντοτητων, τους στοιχειων τους και των συσχετισεων.



Προκειμενου η εφαρμογη να συλλεγει και να χρησιμοποιει τα δεδομενα που καταχωρει ο χρηστης με ταχυτητα, αξιοπιστια και ευελιξια, η βαση δεδομενων δημιουργηθηκε με το MySQL. Το MySQL είναι ένα open source συστημα, που χρησιμοποιεί τη γλώσσα SQL (Structured Query Language) για να επεμβαίνει στα δεδομένα της βάσης δεδομένων και χρησιμοποιειται σε ιστοσελιδες και σε εφαρμογες ηλεκτρονικου εμποριου, διαχειρισης πελατων η υπαλληλων κλπ.

Όπως φαινεται από το παραπανω σχημα της βασης δεδομενων, καποια από τα χαρακτηριστικα των οντοτητων, συμβολιζονται με διαφορετικο χρωμα. Κάθε χρωμα συμβολιζει μια διαφορετικη κατηγορια χαρακτηριστικου. Τα attributes με ανοιχτο μπλε χρωμα συμβολιζουν τα primary keys κάθε οντοτητας, τα attributes με πρασινο ανοιχτο χρωμα συμβολιζουν τα βασικα χαρακτηριστικα της οντοτητας, τα attributes με ανοιχτο μωβ χρωμα συμβολιζουν τα foreign keys και τα attributes με ανοιχτο κοκκινο χρωμα συμβολιζουν τα χαρακτηριστικα της οντοτητας, που είναι δευτερευοντα και βοηθητικα μονο ως προς το συστημα η δημιουργουνται από αυτό. Ως δευτερευοντα οριζονται τα χαρακτηριστικα τα οποια ειτε προκυπτουν από διαδικασιες της πλατφορμας και όχι από τον χρηστη, ειτε δεν εμφανιζονται καθολου στον χρηστη και είναι χρησιμα μονο για το συστημα. Οι οντοτητες της βάσης αναφέρονται παρακάτω μαζί με τα στοιχεία που τις αποτελούν καθώς και μία λεπτομερή επεξήγηση των συσχετίσεων τους.

**3.5.1 Οντότητες**

1. Οντοτητα «Projects»

Ο πινακας των projects αναφερεται στα Projects που εχουν καταχωρηθει σε ολη την διαρκεια χρησης του συστηματος από την εκαστοτε επιχειρηση και αποτελειται από 7 χαρακτηριστικα τα οποια είναι :

* id: Ένα αναγνωριστικό για το project.
* name: Το όνομα του project.
* duration: Η διάρκεια του project σε μηνες.
* budget: Το προϋπολογισμό του project.
* status: Η κατάσταση του project.
* tasks: Ο αριθμός των εργασιών στο project.
* curTasks: Ο αριθμός των τρεχουσών εργασιών στο project.

Το id του project είναι μοναδικο, αυξανεται αυτοματα με κάθε καταχωρηση, είναι τυπου int και αποτελει το primary key του πινακα. Το name είναι τυπου varchar και τα υπολοιπα χαρακτηριστικα είναι τυπου int. Το status, το tasks και το curTasks εμφανιζονται στο σχημα με ανοιχτο κοκκινο χρωμα, καθως ο χρηστης δεν εμπλεκεται αμεσα με την αρχικοποιηση αυτων των τιμων, και αυτές χρησιμοποιουνται μονο από το συστημα παρα μονο για την παρακολουθηση της καταστασης του project και του τερματισμου.

1. Οντοτητα «Employees»

O πινακας employees αναφερεται στους υπαλληλους που εργαζονται στην εκαστοτε επιχειρηση και που εχουν καταχωρηθει στο συστημα. Αποτελειται από 9 χαρακτηριστικα τα οποια είναι :

* id: Ένα αναγνωριστικό για τον υπάλληλο.
* pid: Ο αριθμος ταυτοτητας του υπαλλήλου.
* email: Η διεύθυνση ηλεκτρονικού ταχυδρομείου του υπαλλήλου.
* name: Το όνομα του υπαλλήλου.
* contract: Ο τύπος σύμβασης του υπαλλήλου.
* salary: Το μισθό του υπαλλήλου.
* birth: Η ημερομηνία γέννησης του υπαλλήλου.
* password: Ο κωδικός πρόσβασης του υπαλλήλου.
* hired: Δεικτης που δηλωνει αν είναι ενεργος ο υπαλληλος.

Το id του employee είναι μοναδικο, αυξανεται αυτοματα με κάθε καταχωρηση, είναι τυπου int και αποτελει το primary key του πινακα. Το name, το pid, το password και το email είναι τυπου varchar και τo birth, salary, hired και contract να είναι τυπου int, με το contract να παιρνει τις τιμες 4 και 8, αναλογα με την συμβαση τους. Το hired εμφανιζεται στο σχημα με ανοιχτο κοκκινο χρωμα, καθως η τιμη συμπληρωνεται αυτοματα με την αρχικοποιηση του υπαλληλου και δεν σχετιζεται αμεσα με τον χρηστη, αλλα χρησιμοποιειται κυριως από το συστημα για να διαχειριζεται τους ενεργους και μη υπαλληλους αναλογως.

1. Οντοτητα «Phases»

Ο πινακας phase αναφερεται σε ολες τις ενεργες η μη φασεις των project του συστηματος και αποτελειται από 9 χαρακτηριστικα, τα οποια είναι :

* id: Ένα μοναδικό αναγνωριστικό για τη φάση.
  + type: Ο τύπος-ονομα της φάσης.
  + duration: Η διάρκεια της φάσης σε μηνες.
  + start: Η ημερομηνία έναρξης της φάσης.
  + end: Η ημερομηνία λήξης της φάσης.
  + budget: Ο προϋπολογισμός της φάσης.
  + tasks: Ο αριθμός των εργασιών στη φάση.
  + curTasks: Ο αριθμός των τρεχουσών εργασιών στη φάση.
  + project\_id: Το έργο στο οποίο ανήκει η φάση.

Το id της φασης είναι μοναδικο, αυξανεται αυτοματα με κάθε καταχωρηση, είναι τυπου int και αποτελει το primary key του πινακα. Το type είναι τυπου varchar, το duration, το curTasks και tasks είναι τυπου number, ενώ το start και το end είναι τυπου date. To end και το curTasks εμφανιζονται στο σχημα με ανοιχτο κοκκινο χρωμα, καθως το end δεν υπολογιζεται αμεσα από τον χρηστη, αλλα από το start και το duration που δινει αυτος, οποτε το end υπολογιζεται από το συστημα. Το curTasks επισης χρησιμοποιειται και αυτό κυριως από το συστημα, παρα μονο για την παρακολουθηση της καταστασης της φασης και του τερματισμου της. Το project\_id είναι foreign key που αναφερεται στο αναγνωριστικο καποιου project και υποδεικνυει σε ποιο project ανηκει η φαση.

1. Οντοτητα «Payments»

Ο πινακας payments αναφερεται σε ολες τις πληρωμες του συστηματος και δηλωνει το ποιοι υπαλληλοι εχουν παρει μερος σε ποια φαση. Αποτελειται από 5 χαρακτηριστικα, τα οποια είναι :

* id: Ένα μοναδικό αναγνωριστικό για την πληρωμή.
* pay: Το ποσό χρημάτων που καταβλήθηκε στον υπάλληλο.
* contract: Οι ωρες απασχολησης του υπαλληλου για την συγκεκριμενη φαση.
* phase\_id: Η φάση για την οποία έγινε η πληρωμή.
* employee\_id: Ο υπάλληλος που έλαβε την πληρωμή.

Το id της φασης είναι μοναδικο, αυξανεται αυτοματα με κάθε καταχωρηση, είναι τυπου int και αποτελει το primary key του πινακα. Το pay και το contract είναι τυπου int, αλλα το pay προκυπτει από το συστημα και από την επιλογη του contract που θα κανει ο manager, οποτε εμφανιζεται και αυτό με ανοιχτο κοκκινο χρωμα, σε σχεση με το contract. Το phase\_id και το employee\_id είναι τα foreign keys που αναφερονται στα αναγνωριστικα καποιας φασης και καποιου υπαλληλου αντιστοιχα και υποδεικνυουν σε ποια φαση και σε ποιον υπαλληλο αναφερεται η συγκεκριμενη πληρωμη.

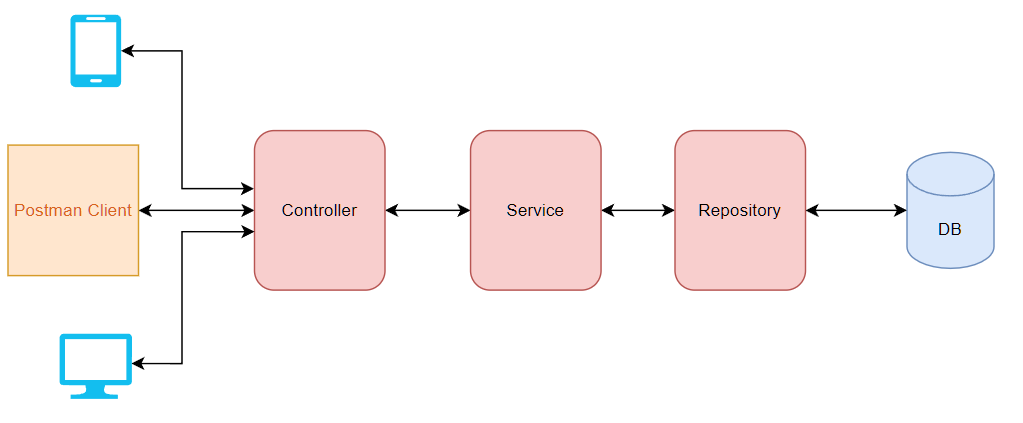
**3.5.2** **Σχεσεις Οντοτητων**

Σε γενικες γραμμες, όπως φαινεται και από το διαγραμμα, η βαση δεδομενων της εφαρμογης είναι σχετικα απλη, αποτελειται από τεσσερις οντοτητες εκ των οποιων οι δυο μπορουν να είναι ανεξαρτητες και οι υπολοιπες δυο προυποθετουν την υπαρξη των πρωτων δυο. Αναλυτικοτερα, η υπαρξη ενός υπαλληλου δεν προυποθετει την υπαρξη ενός project, ουτε μιας φασης ή μιας πληρωμης. Το γεγονος ότι η εφαρμογη χτιστηκε, με βαση να χωρισει την αρχικοποιηση ενός project σε σταδια, κανει θεωρητικα και την οντοτητα project ανεξαρτητη από ολες τις υπολοιπες οντοτητες, αν και μεσω του συστηματος και του δευτερου σταδιου αρχικοποιησης, η οντοτητα project συνδεεται με την οντοτητα phases. Η πιο σκουροχρωμη γραμμη στο διαγραμμα υποδηλωνει την εξαρτηση μιας οντοτητας από την άλλη, οποτε στην συγκεκριμενη περιπτωση, ένα project δεν είναι απαραιτητο να εχει φασεις, αλλα μια φαση πρεπει οπωσδηποτε να εχει ένα project για να υπαρχει, καθως δεν νοοειται φαση χωρις project. Η σχεση μεταξυ project και φασης είναι σχεση “1 προς πολλα” (one-to-many), που σημαινει πως ένα project μπορει να εχει πολλες φασεις, αλλα μια φαση μπορει να εχει μονο ένα project, για αυτό όπως αναφερθηκε παραπανω, γινεται και η χρηση του foreign key project\_id στο πινακα phases.

Η οντοτητα phase από την άλλη, σχετιζεται και με τις τρεις άλλες οντοτητες της βασης. Η πρωτη συσχετιση είναι με την project που αναφερθηκε μολις και η δευτερη συσχετιση ερχεται με τον υπαλληλο. Στην συγκεκριμενη σχεση, ουτε ο υπαλληλος εξαρταται από την υπαρξη φασης, ουτε η φαση από την υπαρξη υπαλληλου, αν και σε αυτή την περιπτωση επισης, μεσω του τριτου σταδιου αρχικοποιησης πρακτικα κάθε φαση εχει τελικα υπαλληλους που είναι «occupied» σε αυτην. Η σχεση μεταξυ τους είναι μια σχεση “πολλα προς πολλα” (many-to-many), αφου ενας υπαλληλος μπορει να παιρνει μερος σε πολλες φασεις και μια φαση μπορει να εχει πολλους υπαλληλους. Η τριτη συσχετιση ερχεται με την οντοτητα payment. Σε αυτή την σχεση η υπαρξη payment προυποθετει την υπαρξη φασης, ενω η υπαρξη φασης θεωρητικα δεν προυποθετει την υπαρξη payment (πρακτικα όμως δεν ισχυει). Η σχεση μεταξυ τους είναι σχεση “ένα προς πολλα” (one-to-many), που σημαινει πως μια φαση μπορει να εχει πολλες πληρωμες, ενώ μια πληρωμη πρεπει να εχει μια συγκεκριμενη φαση, δεδομενου ότι αναφερεται σε καποιον υπαλληλο, οποτε αυτοματα δημιουργειται και η συσχετιση μεταξυ οντοτητας payment και οντοτητας employee. Σε αυτή την σχεση η υπαρξη πληρωμης δεν είναι απαραιτητη για την υπαρξη υπαλληλου, αφου ενας υπαλληλος μπορει να μην εχει παρει ποτε μερος σε καποια φαση, αλλα η υπαρξη πληρωμης προυποθετει την υπαρξη υπαλληλου. Η σχεση μεταξυ τους είναι επισης «ένα προς πολλα», αφου ενας υπαλληλος μπορει να εχει περισσοτερες από μια πληρωμες, ενώ μια πληρωμη αναφερεται σε ακριβως έναν υπαλληλο (employee-id) της δεδομενης φασης (phase-id)

***3.6 Τεχνολογιες Αναπτυξης εφαρμογης***

Για την βαση δεδομενων της εφαρμογης επιλεχθηκε όπως αναφερθηκε και παραπανω η χρηση της MySQL για λογους απλοτητας σταθεροτητας και ευκολιας. Για το πισω μερος της εφαρμογης, η αλλιως backend, χρησιμοποιηθηκε το Spring Boot. Το Spring Boot είναι ένα framework της Java, που είναι πολύ ευελικτο και ισχυρο και που διευκολυνει το backend κομματι στην αναπτυξη εφαρμογων. Η εντονη ενεργητικοτητα που παρατηρειται, από τις μεγαλες επιχειρησεις και εργα καθως και η πληθωρα πλεονεκτηματων που προσφερει το framework αυτό, εκαναν την επιλογη του ευκολοτερη και με την χρηση της δοθηκε και μεγαλυτερη αξια στην ιδια της εφαρμογη αλλα και στην προσπαθεια υλοποιησης της. Το Spring Boot είναι δημοφιλες από μικρές έως μεγάλες επιχειρήσεις και χρησιμοποιείται για web apps και εφαρμογές λογισμικου. Φημιζεται για τις λειτουργιες που προσφερει και που επιταχυνουν τη διαδικασια αναπτυξης, αλλα και για την ευκολη ρυθμιση της παρεχοντας πολλες προεπιλεγμενες ρυθμισεις, που εξοικονομουν χρονο από τον επιπλεον κωδικα. Παρεχονται πολλες αυτοποιημενες ενεργειες που κανουν ευκολοτερη την συνδεση με την βαση δεδομενων αλλα και για την επεξεργασια των αιτηματων και την ασφαλεια της εφαρμογης Για την συνδεση της MySQL βασης δεδομενων με το Spring Boot, χρησιμοποιηθηκε η Spring Data JPA. Το framework αυτό της Spring διευκολυνει και εξοικονομει πολύτιμο χρονο στην αναπτυξη της βασης δεδομενων, των οντοτητων και των συσχετισεων αφου αυτοματοποιει τις οντοτητες, απλα σε κλασεις, οπου για κάθε ονοτητα της βασης θα υπαρχει η αντιστοιχη κλαση. Ως προς την διαρθρωση του back end, η διακριση γινεται σε 4 μερη. Ο πρωτος φακελος models περιεχει τα classes των οντοτητων, που οπως εχει ηδη αναλυθει είναι τεσσερις, αρα και τεσσερα classes. Ο δευτερος φακελος repositories περιεχει τις κλασεις με τα JPA repositories που παρεχουν ευκολη προσβαση στις αντιστοιχες οντοτητες της βάσης δεδομένων. Ο τριτος φακελος services περιεχει όλες τις classes services και τις classes services implementations κάθε οντοτητας. Ο φακελος αυτος ειναι υπευθυνος να παρεχει ολες τις λειτουργιες της εφαρμογης για αυτό και συνηθως χρησιμοποιουνται τα repositories των models εδώ. Ο τελευταιος φακελος controllers, περιέχει τις κλάσεις controller κάθε οντοτητας και υλοποιούν την επιχειρησιακη λογική της εφαρμογής. Οι controllers είναι υπεύθυνοι για την επεξεργασία των αιτημάτων από τους χρήστες και την επιστροφή των αποτελεσμάτων. Με τις δεδομενες απαιτησεις της εφαρμογης τα περισσοτερα Api Requests ηταν τυπου Post και Get, οποτε η εφαρμογη ακολουθησε τις αρχες των RESTful Apis και τα δεδομενα που μεταφερονται μεταξύ του πελάτη και του εξυπηρετητή εχουν την μορφη json. Για την αμεση και ευκολη δοκιμη της βασης δεδομενων συνολικα, των υπηρεσιων και των controllers που κατασκευαστηκαν, χρησιμοποιηθηκε το Postman. Μια από τις βασικότερες λειτουργίες που χαρακτηριζουν το Postman είναι η ευκολία της εκτέλεσης οποιουδηποτε τυπου HTTP request σε οποιοδήποτε API. Ετσι αναλογως της απαντησης που θα επιστρεψει το εργαλειο, ο προγραμματιστης μπορει να εντοπισει συγκεκριμενα errors και να προλαβει αστοχιες στο back end.

. 

Η χρηση του framework Spring Boot απαιτει αντιστοιχης βαρυτητας συστημα τοσο στο front end της εφαρμογης. Ενα από τα πιο δημοφιλή εργαλεία για την ανάπτυξη επαγγελματικών και διαδραστικών ιστοσελίδων και εφαρμογών είναι η React. Η React είναι μια βιβλιοθηκη της Javascript που χρησιμοποιεται για τη δημιουργία διεπαφών χρήστη (UIs). Επιτρεπει την συνταξη JSX, δινοντας την δυνατοτητα χρησης HTML και Javascript στον ιδιο κωδικα αλλα το μεγαλυτερο πλεονεκτημα είναι ότι χρησιμοποιει components για να δομησει το UI μιας εφαρμογης. Αυτά τα components μπορουν να είναι επαναχρησιμοποιουμενα σε διαφορα μερη της εφαρμογης μειωνοντας το μεγεθος του κωδικα και της επαναληψης και το UI ανταποκρίνεται δυναμικά στις αλλαγές των components, χωρίς την ανάγκη για επαναφόρτωση της σελίδας.

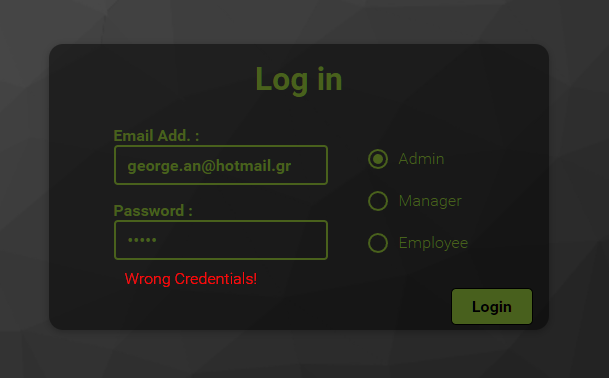
Στην συνεχεια, χρησιμοποιηθηκε το εργαλειο διαχειρισης πακετων nmp (Node Package Manager) για την εγκατασταση και την διαχεριση των απαραιτητων πακετων που χρειαστηκε η εφαρμογη, όπως και της React βιβλιοθηκης. Για τα αιτηματα που δημιουργουνται στο front end χρησιμοποιηθηκε η βιβλιοθηκη Axios, γιατι μπορει να χρησιμοποιει ασυγχρονες λειτουργιες και να επεξεργαστει σφαλματα η να ακυρωσει αιτησεις αλλα και γιατι υποστηριζει διαφορες μορφες δεδομενων όπως JSON. Στο κομματι του front-end και της React χρησιμοποιηθηκε CSS για την μορφοποιηση των στυλιστικων παραμετρων αλλα εγινε και αρκετη χρηση της βιβλιοθηκης Material UI.

***4 <******Λειτουργίες και διεπαφή χρήστη (UI/UX)>***

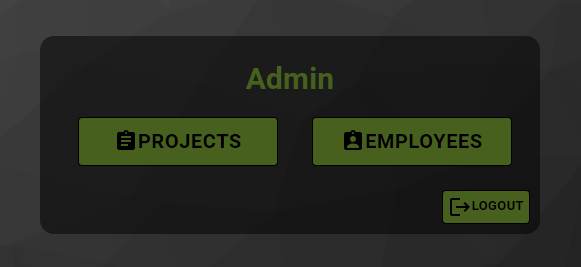
Στο κεφάλαιο αυτό θα περιγραφεί ο τρόπος λειτουργίας της εφαρμογής που αναπτύχθηκε στο πλαίσιο της διπλωματικής εργασίας. Θα γινει εκτενης αναλυση στο γραφικο περιβαλλον, το οποιο δημιουργηθηκε και θα αναδειχθουν ολες οι λειτουργιες των χρηστων. Κυρια στοιχεια του συγκεκριμενου κεφαλαιου θα είναι τα ενδεικτικα screenshots που θα περιλαμβανονται, προκειμενου ο αναγνωστης να μπορεσει να κατανοησει πληρως ολες τις διαδικασιες και να ξεκαθαριστουν οποιαδηποτε κενα του εχουν δημιουργηθει εως τωρα. Ολες οι λειτουργιες και οι ιδιαιτεροτητες τους θα αποσαφηνιστουν και θα αναλυθουν και διαφορετικα σεναρια σε αυτές τις λειτουργιες.

Ξεκινοντας με την αναλυση των απαιτησεων του περιβαλλοντος του συστηματος, η εφαρμογη πρεπει να είναι οσο πιο απλη γινεται στον χρηστη, να είναι καλοσχεδιασμενη και ευστοχη. Οι διαφορες λειτουργιες θα πρεπει να εμφανιζονται με τροπο εξυπνο και λιτο, ετσι ώστε το περιβαλλον της εφαρμογης να μην είναι πολυπλοκο και να εξασφαλιζει την ευκολη κατανοηση των ενεργειων που πρεπει η μπορει να κανει ο χρηστης. Για αυτό τον λογο, η πρωτη σελιδα που εμφανιζεται, που είναι η συνδεση των χρηστων, είναι σχεδιασμενη οσο πιο απλα και ξεκαθαρα γινεται. Επισης και οι σελιδες που ακολουθουν μετα την συνδεση του εκαστοτε χρηστη είναι σχεδιασμενες για να παρεχουν τις καταλληλες πληροφοριες και ενεργειες και μπορουν να παρομοιαστουν με σελιδες τυπου dashboard.

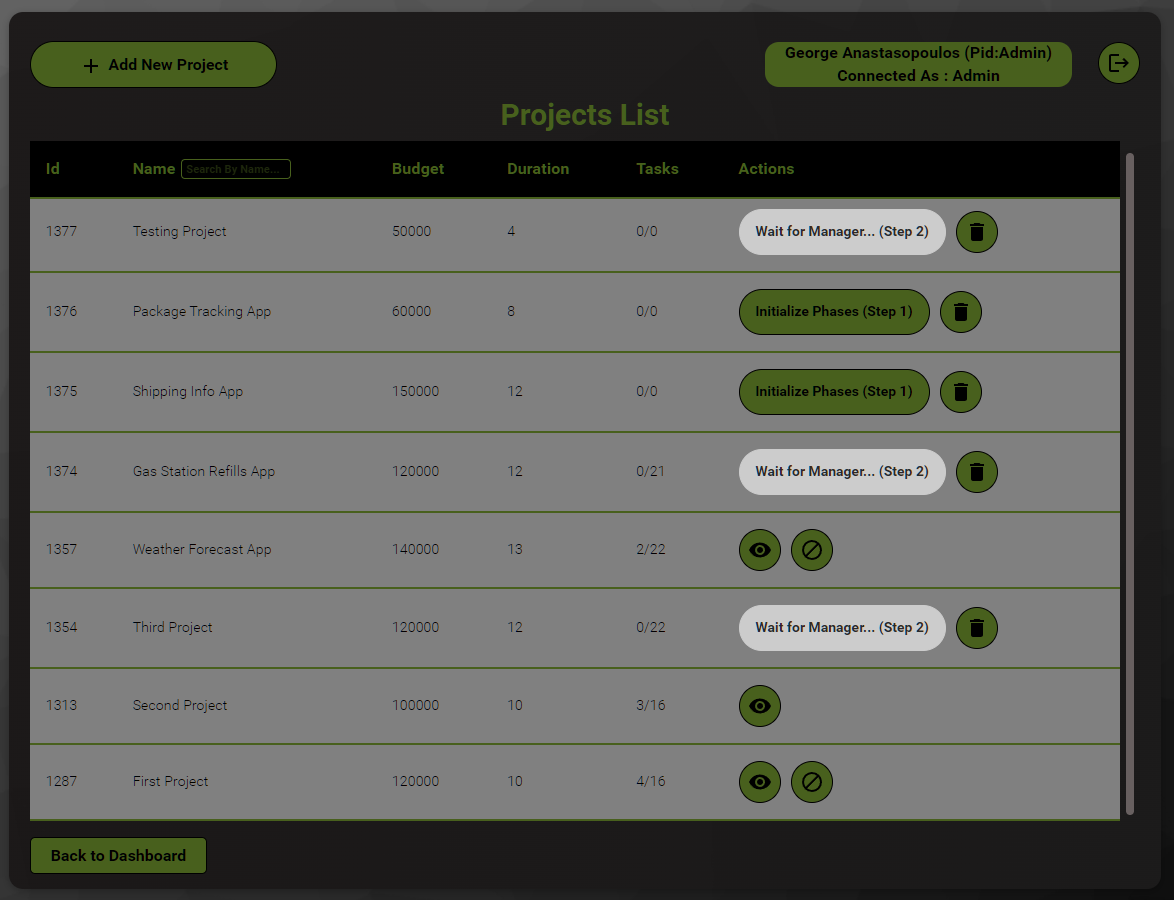
Περαν αυτων των σελιδων, θεωρηθηκε πως η χρηση modal components θα εδινε μια καλυτερη εμπειρια χρησης και θα εκανε την εφαρμογη πιο οργανωμενη και πλοηγησιμη, οποτε ολες οι υπολοιπες λειτουργιες αναδυονται κάθε φορα με το αντιστοιχο modal.   
  
Αρχικα ο χρηστης με την εναρξη της χρησης της εφαρμογης, βλεπει την αρχικη σελιδα της, που είναι η σελιδα συνδεσης. Εκει ο χρηστης εχει την επιλογη να επιλεξει ρολο χρησης του συστηματος και με την μορφη radio buttons, εμφανιζονται οι επιλογες Admin, Manager, Employee.   
  
Επισης τα πεδια συμπληρωσης εμφανιζονται με την μορφη text field και η εφαρμογη ενημερωνει καταλληλα τον χρηστη με ένα μηνυμα σε κοκκινο χρωμα, σε περιπτωση λανθασμενων στοιχειων.



Από το back end της εφαρμογης και των καταλληλων api requests, ελεγχεται η εγκυροτητα των στοιχειων του χρηστη και επειτα από την επιτυχη συνδεση του, αναλογα με την ιδιοτητα που εχει επιλεξει, μεταφερεται και στην αντιστοιχη σελιδα. Λαμβανοντας υποψιν τις απαιτησεις και τους περιορισμους που αναλυθηκαν στο κεφαλαιο 3.2, οι αρχικες οθονες της εκαστοτε ιδιοτητας κατασκευαστηκαν με τροπο τετοιο ώστε να εμφανιζουν στον εκαστοτε χρηστη τις επιθυμητες πληροφοριες και ενεργειες. Σε γενικες γραμμες, οι σελιδες και των τριων χρηστων είναι παρομοιες ως προς την δομη τους, καθως οι λειτουργιες τους μοιαζουν η τουλαχιστον ο τροπος εμφανισης καποιων από αυτων είναι παρομοιος. Η μονη διαφορα είναι πως στην περιπτωση του Admin, που εχει την επιλογη να δει και να επεξεργαστει υπαλληλους, μετα την επιτυχη συνδεση μεταφερεται σε μια σελιδα που εχει να επιλεξει αναμεσα στα projects και τους employees.



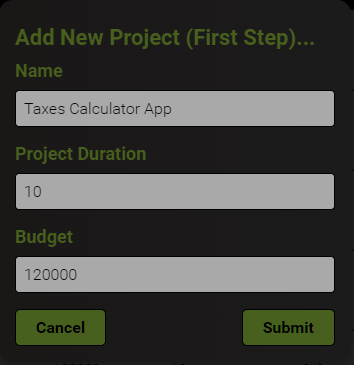
Επιλεγοντας ένα από τα δυο, ο admin θα ανακατευθυνθει στην αντιστοιχη αρχικη σελιδα και συγκεκριμενα αν επιλεξει το κουμπι projects, θα μεταφερθει στην σελιδα των projects, όπως γινεται στην περιπτωση συνδεσης των αλλων δυο ιδιoτητων. Να σημειωθει πως, αναλογα την συνδεση του χρηστη, η σελιδα των projects θα εμφανιζει μονο τα projects που αφορουν τον καθενα, οποτε στην περιπτωση του admin θα εμφανιζονται όλα τα projects που εχουν εκτελεστει στην εφαρμογη.



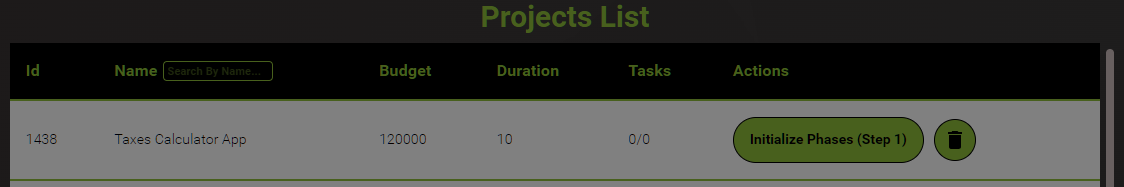
Για την αρχικη σελιδα των projects, αναγκαια και χρησιμη ηταν η προβολη αυτων σε μια αναλυτικη λιστα. Ετσι με ένα api get request, η εφαρμογη δεχεται όλα τα project που υπαρχουν στην βαση, οποτε η λιστα περιλαμβανει τα πολύ βασικα στοιχεια μονο, με πιο σημαντικο να είναι το πεδιο actions για το καθενα από αυτά. Το πεδιο actions περιλαμβανει τις διαθεσιμες λειτουργιες που μπορει ο χρηστης να επιλεξει, αναλογα βεβαια με τον ρολο συνδεσης και κυριως το εκαστοτε status του project. Στην περιπτωση της εικονας …. , ο χρηστης εχει συνδεθει ως Admin, οποτε εχει την δυνατοτητα της αρχικοποιησης των φασεων, της καταργησης, της διαγραφης και της παρακολουθησης του project, ενώ η δυνατοτητα επιλογης υπαλληλων εμφανιζεται ως disabled, καθως είναι λειτουργια-υποχρεωση του manager. Ο χρηστης ενημερωνεται καταλληλα από το πεδιο πανω δεξια της εικονας, για τον ρολο με τον οποιο εχει συνδεθει στο συστημα και ανα πασα στιγμη ο χρηστης εχει την δυνατοτητα να κανει logout.

Στην περιπτωση του Admin, κατω από την λιστα εμφανιζεται και ένα κουμπι Back to Dashboard, που δινει την δυνατοτητα στον admin να επιστρεψει στην αρχικη του σελιδα, ενδεχομενως για να επιλεξει να μεταφερθει στην αρχικη σελιδα των employees, που μονο αυτος εχει προσβαση.   
  
Επισης για την προσθηκη νεου Project και την εκτελεση του πρωτου σταδιου αρχικοποιησης του, εχει τοποθετηθει το καταλληλο custom button πανω αριστερα από την λιστα. Να σημειωθει πως αυτό το κουμπι θα εμφανιζεται στην αρχικη σελιδα μονο του admin, καθως μονο αυτος είναι αρμοδιος για αυτην την λειτουργια, όπως προκυπτει από τις απαιτησεις του συστηματος.

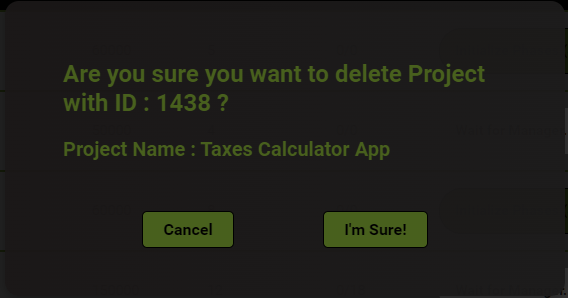
Με την επιλογη του κουμπιου αναδυεται ενα modal με την μορφη φορμας, οπου ο admin πληκτρολογει τα βασικα στοιχεια του project στα αναλογα textfields.



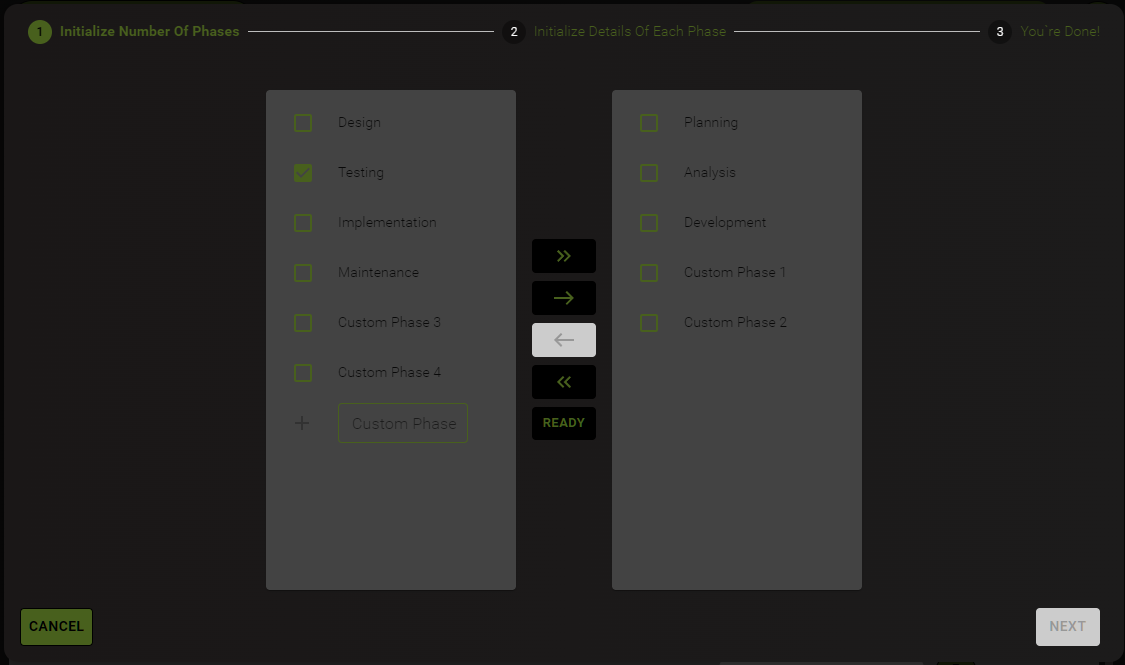
Με την ολοκληρωση της συμπληρωσης, ο χρηστης καλειται να επιλεξει το κουμπι submit για να αποθηκευτει το project στην βαση δεδομενων και να ολοκληρωσει το πρωτο σταδιο αρχικοποιησης. Μολις μεταφερθει παλι στην αρχικη σελιδα των projects, η λιστα ανανεωνεται αυτοματα και πλεον είναι ορατο και το νέο project σε αυτην. Να σημειωθει πως το κουμπι submit είναι clickable μονο εάν εχουν συμπληρωθει τα στοιχεια σωστα και πως ο χρηστης εχει την δυνατοτητα να ακυρωσει την διαδικασια με την επιλογη του κουμπιου cancel.



Όπως φαινεται στην εικονα …., το project πλεον είναι εμφανες μαζι με τα υπολοιπα και το πεδιο actions του, περιεχει δυο buttons. Με την επιλογη του δευτερου button, ο admin μπορει να δαιγραψει το συγκεκριμενο project, μεσω του modal που εμφανιζεται και του αντιστοιχου api delete request που γινεται στο back end της εφαρμογης.



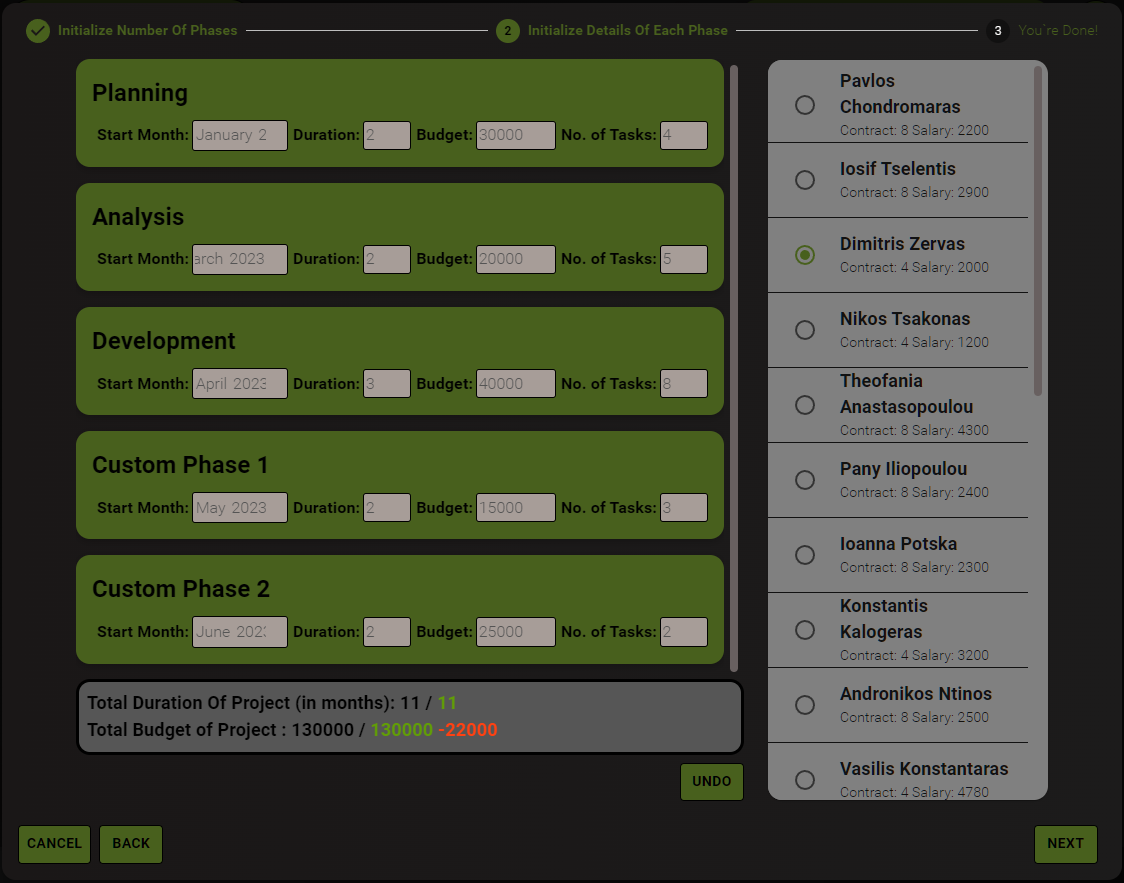
Με την επιλογη του πρωτου button, ο admin μπορει να συνεχεισει με την αρχικοποιηση του project και να προσθεσει τις φασεις, τις λεπτομερειες τους και τον manager του εργου. Με την χρηση παραθυρου modal, το πρωτο πραγμα που πρεπει να κανει ο admin είναι να επιλεξει τις φασεις που θελει. Όπως εχει αναφερθει στις απαιτησεις του συστηματος, ο χρηστης μπορει ειτε να επιλεξει από τις ηδη προτεινομενες φασεις που παρεχει το συστημα και ακολουθουν το μοντελο waterfall του SDLC, ειτε να προσθεσει και δικες του custom φασεις στις ηδη υπαρχουσες. Χρησιμοποιηθηκε λοιπον μια transferlist, οπου στην αριστερη λιστα είναι οι διαθεσιμες προς επιλογη φασεις και στην δεξια λιστα είναι οι επιλεγμενες φασεις. Ο σχεδιασμος του γραφικου περιβαλλοντος με τετοιο τροπο, εξασφαλιζει την σαφηνεια της διαδικασιας και κανει αυτονοητο αμεσα στο χρηστη τι πρεπει να κανει, οποτε δεν μπορουν να δημιουργηθουν παρανοησεις η να προκυψουν λαθη.



Παραδειγματος χαρην οπως φαινεται στην εικονα …., ο admin, περαν των προκαθορισμενων φασεων, εχει προσθεσει και τεσσερις custom φασεις. Από αυτές τις φασεις εχει επιλεξει τελικα τρεις φασεις από τις προκαθορισμενες και δυο από τις custom φασεις, όπως φαινεται από την δεξια λιστα. Το button «Ready», αρχικα είναι disabled, καθως το συστημα εχει περιορισει το αριθμο των επιλεγμενων φασεων σε περισσοτερες από τεσσερις. Όταν ο χρηστης μπορει και επιλεξει το button «Ready», η λιστα “κλειδωνει” και γινεται enabled το button «Next», προκειμενου ο admin να συνεχισει με το αρχικοποιηση των στοιχειων της κάθε φασης από τις επιλεγμενες. Να σημειωθει πως το button «ready», αλλαζει και γινεται button «Edit», σε περιπτωση που ο χρηστης θελει να αλλαξει κατι πριν προχωρησει στο επομενο βημα.

**Περιπτωση Agile Project**

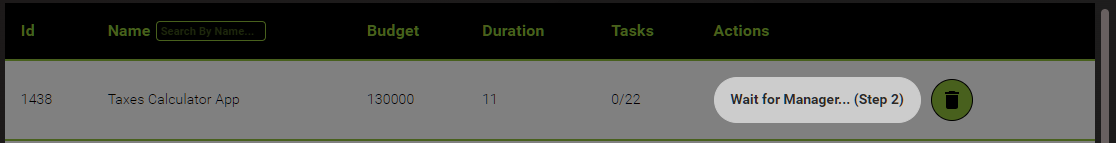
Αφου λοιπον επιλεξει «Next», μεταφερεται στο επομενο σταδιο, οπου του προβαλλεται μια φορμα, οπου για κάθε φαση του προηγουμενου σταδιου, περιεχει τα αντιστοιχα textfields, για την αρχικοποιηση των φασεων. Τα textfields αφορουν τον μηνα εκκινησης, την διαρκεια, τον προυπολογισμο και τον αριθμο των tasks, της κάθε φασης. Επισης, το συστημα ενημερωνει καταλληλα τον χρηστη για το εάν οι πληροφοριες που εχει προσθεσει συμφωνουν με τις απαιτησεις του project μεσω του αναλογου πεδιου. Μολις ο admin επιλεξει το button «Ready», κλειδωνουν τα πεδια συπμληρωσης και εμφανιζεται διπλα από την φορμα μια λιστα με υπαλληλους. Από αυτή την λιστα ο διαχεριστης καλειται να επιλεξει έναν από αυτους για Manager του Project και μολις το κανει, ενεργοποιειται το button «Next», που εξαρχης ηταν disabled. Επισης, το συστημα ενημερωνει και για την συνολικη πληρωμη του επιλεγμενου manager για το project, σε σχεση με το συνολικο προυπολογισμο του όπως φαινεται και στην εικονα ….



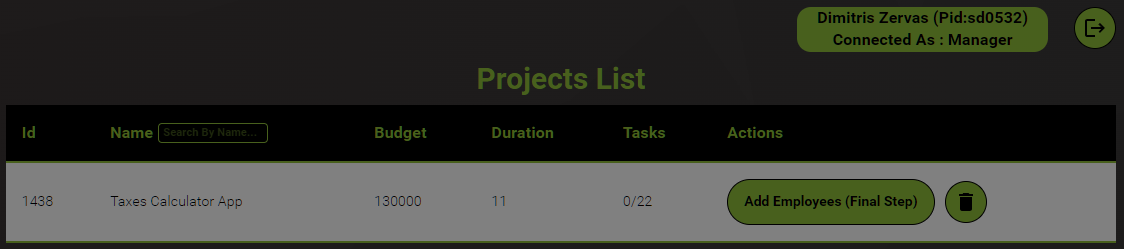
Μολις ο admin τελειωσει και με την επιλογη manager και πατησει το button «Next», ουσιαστικα τερματιζει και η διαδικασια της δευτερης φαση αρχικοποιησης και ο χρηστης μεταφερεται στην σελιδα του επομενου βηματος οπου προβαλλεται μια συνοψη των πληροφοριων που καταχωρηθηκαν εως τωρα.



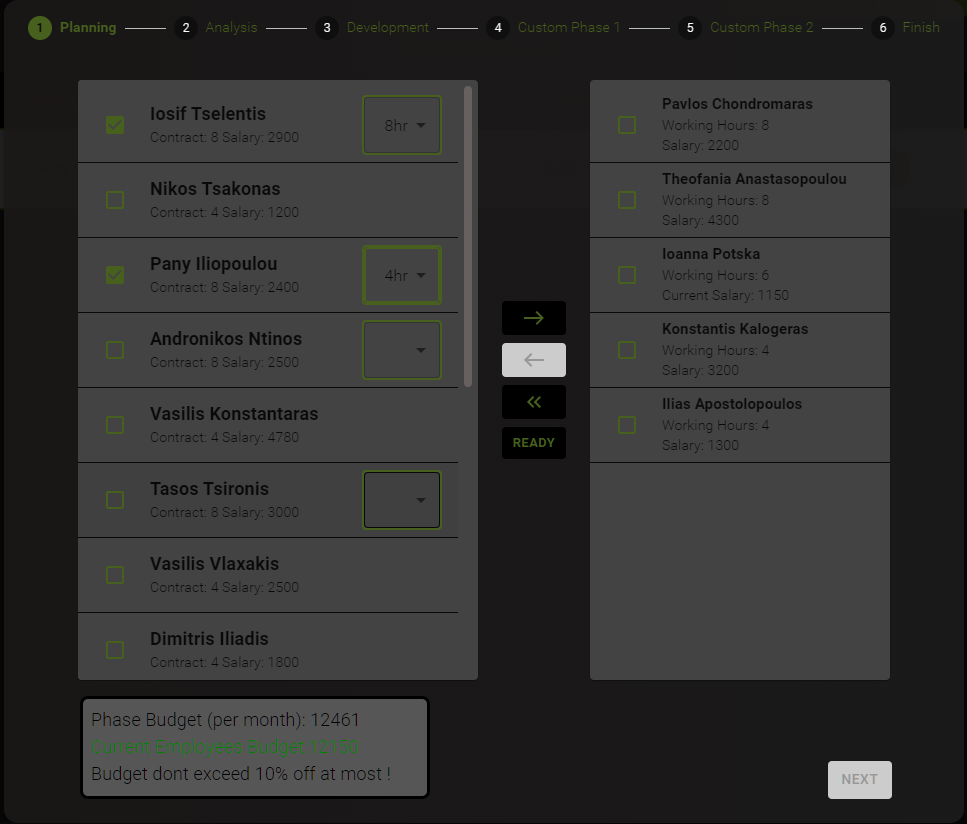
Με την επιλογη κλεισιματος της τελικης καρτελας, κλεινει και το modal και ο admin μεταφερεται στην αρχικη σελιδα των Project οπου βλεπει το ενημερωμενο project πλεον στην λιστα με τα υπολοιπα. Το πεδιο tasks ενημερωθηκε με τις συνολικα task των φασεων όπως επισης και το πεδιο actions. Στο πεδιο actions, περαν του button διαγραφης, πλεον βρικσεται και ένα disabled button που αναφερεται στο τριτο σταδιο αρχικοποιησης τους project. Ο λογος για τον οποιο φαινεται ως disabled είναι επειδη αυτή η λειτουργια είναι διαθεσιμη μονο για τον ορισμενο manager του project.



Οποτε για να συνεχιστει η διαδικασια αρχικοποιησης, θα πρεπει ο συγκεκριμενος υπαλληλος να κανει login ως manager. Εκει από την αρχικη σελιδα και την λιστα των projects, ο Manager βλεπει το Project όπως φαινεται στην παρακατω εικονα.

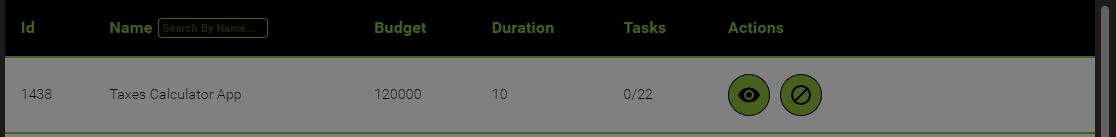


Με την επιλογη της ενεργειας αρχικοποιησης του τριτου σταδιου του project, ο manager θα πρεπει να καταχωρησει υπαλληλους στο Project, μια διαδικασια πολύ σημαντικη, οποτε η αναγκη ευκολιας και απλοτητας ηταν πρωταρχικου ρολου. Για αυτό τον λογο στο επομενο modal, η διαδικασια χωριστηκε σε σταδια και συγκεκριμενα σε τοσα, οσες και οι φασεις, προκειμενου να διαιρεθει σε πιο απλα και κατανοητα μερη. Ετσι εμφανιζεται ένα modal στον manager, οπου παρατηρει παλι την χρηση transferlist.

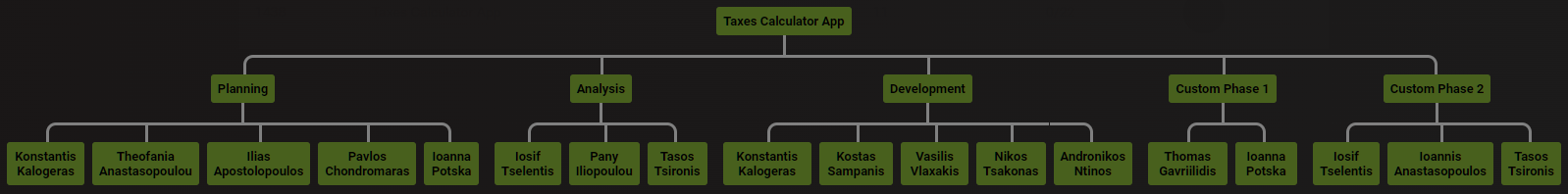


Πιο συγκεκριμενα στην αριστερη λιστα εμφανιζονται οι διαθεσιμοι, για την χρονικη περιοδο της φασης, υπαλληλοι και στην δεξια λιστα οι αντιστοιχοι επιλεγμενοι για αυτην. Το συστημα κρατα ενημερο τον χρηστη σχετικα με τον προυπολογισμο και τις πληρωμες των επιλεγμενων υπαλληλων καθολη την διαρκεια της διαδικασιας, για να κανει πιο ευκολη την επιλογη των καταλληλοτερων υπαλληλων. Επισης το συστημα δινει την επιλογη στον manager, να ορισει υπαλληλους από 8ωρους σε 4ωρους. Μολις επιλεξει τους υπαλληλους, επιλεγει το button «Ready» και αμεσως «Next» και εκτελειται ένα api post request προς την βαση δεδομενων για να αποθηκευσει τους επιλεγμενους υπαλληλους και συνεχιζει στην αρχικοποιηση των επομενων φασεων ακριβως με τον ιδιο τροπο που παρουσιαστηκε μολις.

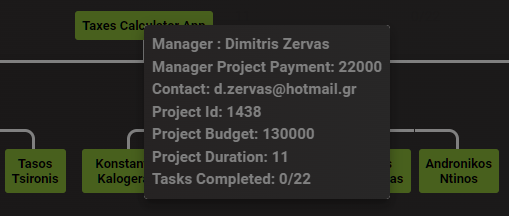
Μολις τελειωσει με ολες τις φασεις ο manager, τερματιζει το τριτο και τελευταιο σταδιο αρχικοποιησης του Project και ενημερωνεται καταλληλα η λιστα της αρχικης σελιδας με το ανανεωμενο Project. Πιο συγκεκριμενα, πλεον ο κάθε χρηστης, ανεξαιρετου ρολου θα μπορει να παρακολουθησει το Project από το ανανεωμενο πεδιο actions. Για τον admin μονο , δινεται και η δυνατοτητα τερματισμου του project επισης, από το δευτερο button που παρεχεται.



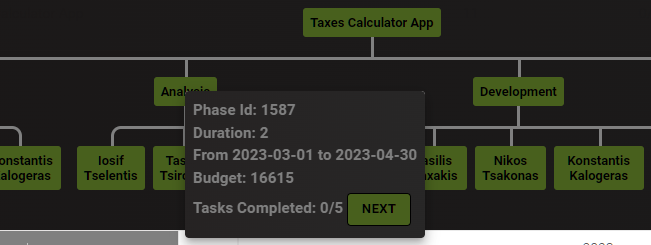
Επιλεγοντας ο χρηστης την προβολη του project, πρεπει να μπορει να δει συγκεντρωτικα και οργανωμενα ολες τις πληροφοριες τις οποιες συλλεχθηκαν. Το συστημα δημιουργηθηκε με τετοιο τροπο ώστε μεσω των διαγραμματων που χρησιμοποιηθηκαν ο χρηστης να εχει μια πιο οπτικοποιημενη εικονα των φασεων, των υπαλληλων και των χρονικων περιθωριων του project. Όπως και στις προηγουμενες περιπτωσεις, ετσι και εδώ εμφανιζεται ένα modal το οποιο περιεχει δυο διαγραμματα, ενός hierarchy tree και ενός διαγραμματος gannt. Το συγκεριμενο hierarchy tree κατασκευαστηκε ώστε να χρησιμοποιει σαν κυριους κομβους, τον κομβο project σε πρωτο επιπεδο, με παρακλαδια του τους κομβους phases σε δευτερο επιπεδο, και σε τριτο επιπεδο τους employees που αφορουν την κάθε φαση.



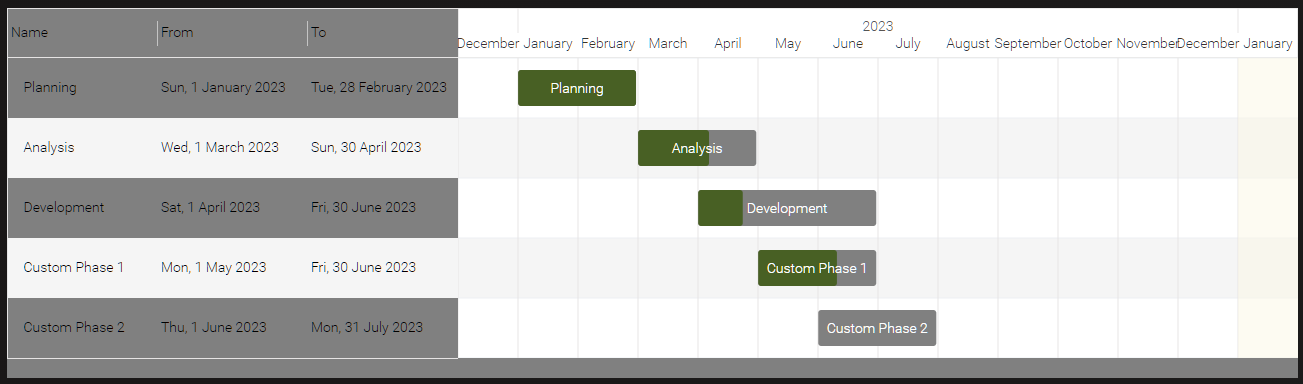
Επισης εχει δοθει η δυνατοτητα στον χρηστη να μπορει να εμφανιζει τα στοιχεια κάθε κομβου η κάθε υπαλληλου κανοντας click πανω στον αντιστοιχο κομβο, όπως για παραδειγμα με τον πρωτο κομβο στην εικονα…..



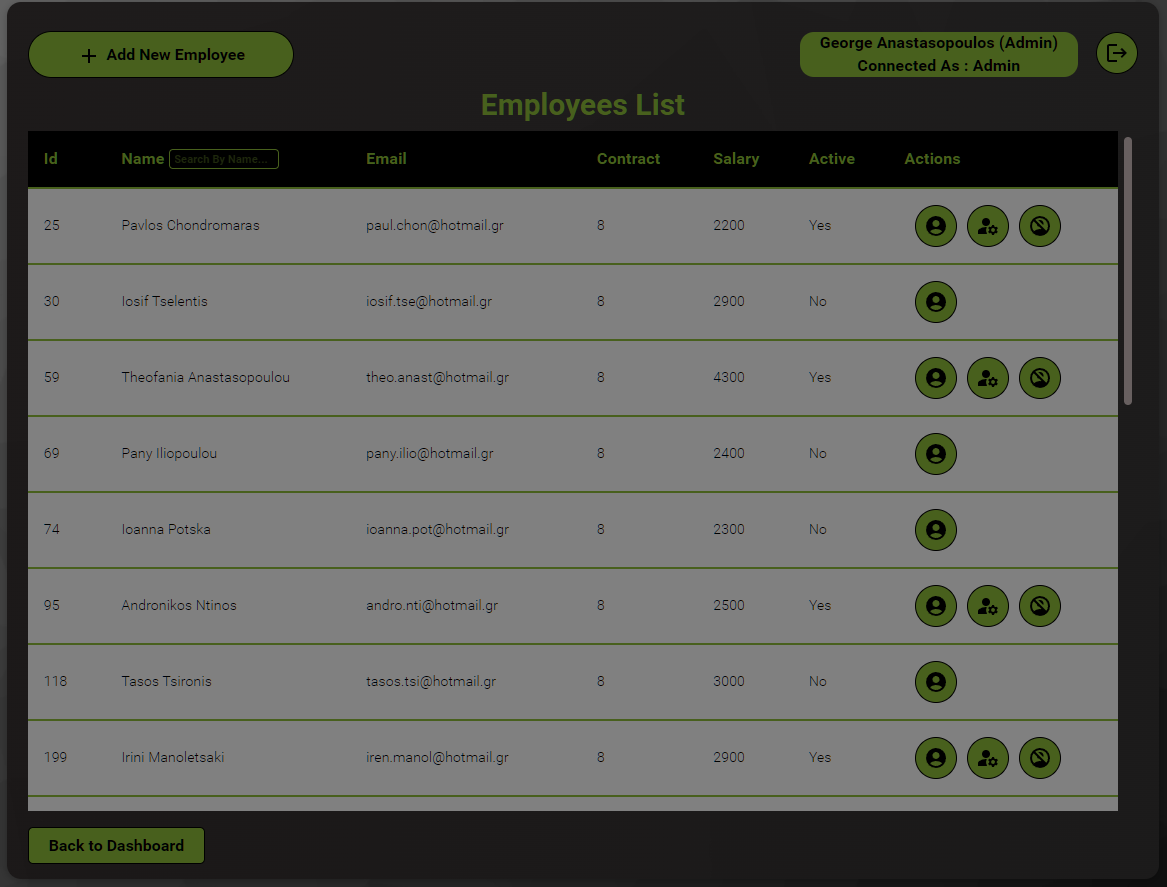
Στην περιπτωση που η συνδεση και η προβολη γινονται από τον υπευθυνο manager, το συστημα δινει την επιλογη σε αυτόν, κανοντας click πανω σε οποιονδηποτε κομβο των φασεων, να επιλεξει μεσω του κουμπιου next, αλλαγη του task και συνεχεια στο επομενο, προσφεροντας μια διαδραστικοτητα κατά την διαδικασια αναπτυξης, καθως ο manager θα ενημερωνει την πλατφορμα για την προοδο κάθε φασης μεσω των tasks της καθεμιας. Αξιζει να σημειωθει, πως μετα την ολοκληρωση ολων των tasks ολων των φασεων, ολοκληρωνεται και το project συνολικα και αυτοματα ενημερωνεται και το status του ως ολοκληρωμενο.



Η αλλαγη του task της φασης είναι αμεσα αντιληπτη μεσω του διαγραμματος gannt που παρεχεται ακριβως κατω από το hierarchy tree. Στην εικονα …, φαινονται ολες οι φασεις και οι χρονικες τους απαιτησεις όπως επισης και η προοδος σε σχεση με τα tasks, με την μορφη bar μεσα στο διαγραμμα.

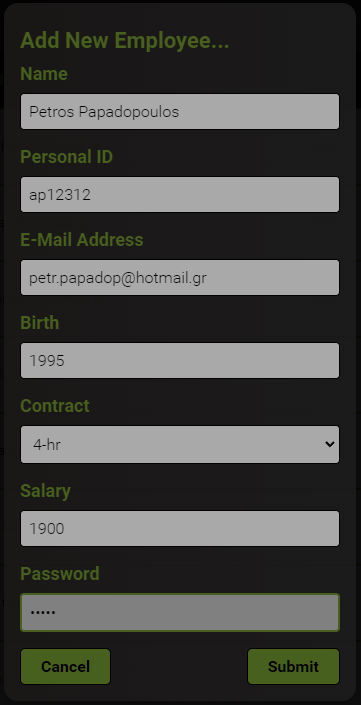


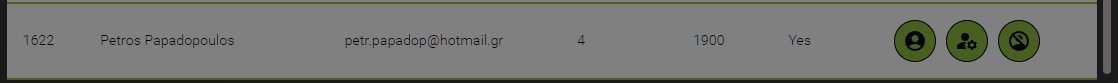
Όπως ειχε αναφερθει παραπανω, ο διαχειριστης με την επιτυχη συνδεση του μεταφερεται σε μια σελιδα, οπου εχει να επιλεξει αναμεσα στα projects, που αναλυθηκε ηδη, αλλα και στους employees. Αν επιλεξει τους employees, θα ανακατευθυνθει στην αρχικη σελιδα των employees οπου εμφανιζεται μια λιστα με ολους τους υπαλληλους, που εχουν απασχοληθει από την εταιρεια, ενεργους και μη.



Όπως και με την αρχικη σελιδα των projects, ετσι και αυτή την αρχικη, απαραιτητη ηταν η προβολη και η οργανωση των υπαλληλων σε μια αναλυτικη λιστα. Η λιστα περιλαμβανει τα απαραιτητα στοιχεια, αλλα και τα μισθολογικα, όπως επισης και ενα πεδιο active και actions. Στο πεδιο actions περιεχονται τρια κουμπια με την μορφη iconbutton και τα δυο πεδια αυτά σχετιζονται με το αν ο υπαλληλος είναι ενεργος η όχι την τωρινη στιγμη στο συστημα. Και στις δυο περιπτωσεις, περιλαμβανεται η λειτουργια της προβολης των λεπτομερειων του υπαλληλου, ενώ αν αυτος είναι ενεργος, διαθεσιμες είναι και οι λειτουργιες της επεξεργασιας και της αποδεσμευσης του. Επισης ο admin ενημερωνεται καταλληλα από το πεδιο πανω δεξια της εικονας…., για τον ρολο με τον οποιο εχει συνδεθει στο συστημα και ανα πασα στιγμη εχει την δυνατοτητα να κανει logout, μεσω του αντιστοιχου iconbutton.

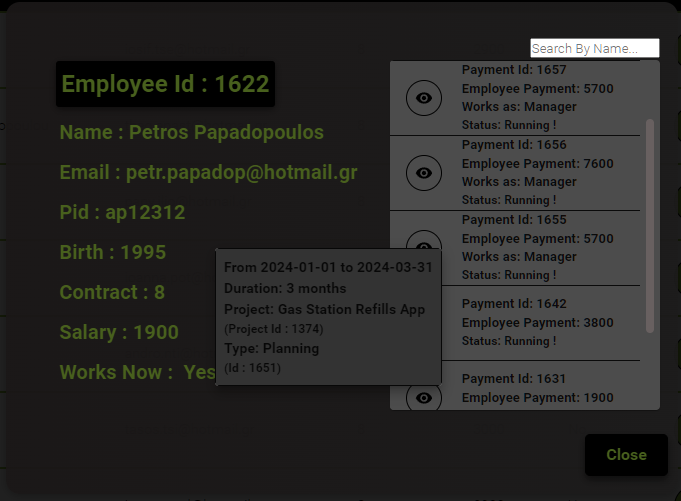
Για την προσθηκη νεου υπαλληλου στο συστημα, εχει τοποθετηθει καταλληλο custom button πανω αριστερα από την λιστα και με την επιλογη του κουμπιου αναδυεται ενα modal με την μορφη φορμας, οπου ο admin μπορει να πληκτρολογησει τα απαραιτητα και απαιτουμενα στοιχεια του νεου υπαλληλου στα αναλογα textfields.



Με την ολοκληρωση της συμπληρωσης, ο admin καλειται να επιλεξει το κουμπι submit για να προσθεσει τον νεο υπαλληλο στο συστημα και να μεταφερθει παλι στην αρχικη σελιδα των employees, οπου πλεον θα είναι ορατος και στην αρχικη λιστα. Να σημειωθει πως το κουμπι submit είναι clickable μονο εάν εχουν συμπληρωθει τα στοιχεια σωστα και πως ο χρηστης εχει την δυνατοτητα να ακυρωσει την διαδικασια με την επιλογη του κουμπιου cancel.

Όπως φαινεται στην εικονα …., ο employee πλεον εμφανιζεται μαζι με τους υπολοιπους και το πεδιο actions του, περιεχει τα τρια iconbuttons που αναφερθηκαν παραπανω.

Με την επιλογη του πρωτου iconbutton, ο admin μπορει να δει ολες τις λεπτομερειες του συγκεκριμενου υπαλληλου μεσω του modal που εμφανιζεται. Στο modal, από την αριστερη πλευρα εμφανιζονται τα στοιχεια του υπαλληλου και για την δεξια πλευρα δημιουργηθηκε μια λιστα με όλες τις πληρωμες που εχει λαβει. Για την λιστα εχει τοποθετηθει και ένα search textfield, οπου ο διαχειριστης μπορει να αναζητησει στις πληρωμες του με βαση το ονομα του project η με βαση την φαση οπου εξ ορισμου από το συστημα για κάθε φαση που εχει συμμετασχει αντιστοιχει και μια πληρωμη. Επισης, για κάθε πληρωμη, περα από τα βασικα της στοιχεια, όπως για παραδειγμα το αν εκτελουσε χρεη manager, o admin μπορει να επιλεξει το αντιστοιχο κουμπι προβολης επιπλεον πληροφοριων, προκειμενου να εχει καλυτερη εικονα σχετικα με την συγκεκριμενη πληρωμη.

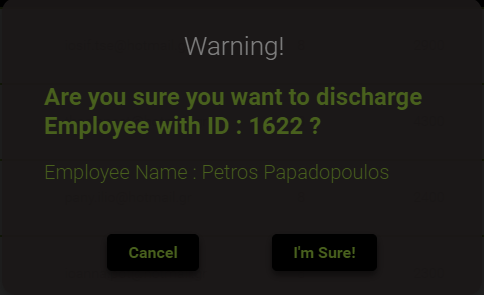


Με την επιλογη του δευτερου iconbutton, ο admin εχει την δυνατοτητα να τροποποιησει τα στοιχεια ενός υπαλληλου. Αυτή η λειτουργια εξυπηρετει και την ανανεωση των προσωπικων στοιχειων του υπαλληλου, αλλα και την αυξηση του μισθου του η την αλλαγη της συμβασης του από part-time σε full-time και αντιστροφα. Δημιουργηθηκε λοιπον μια φορμα παρομοια με αυτή της προσθηκης υπαλληλου που αναφερθηκε παραπανω, με την μονη προφανη αλλαγη ότι ερχεται συμπληρωμενη με τα στοιχεια που ληφθησαν από το back end της εφαρμογης και του αντιστοιχου api get request.

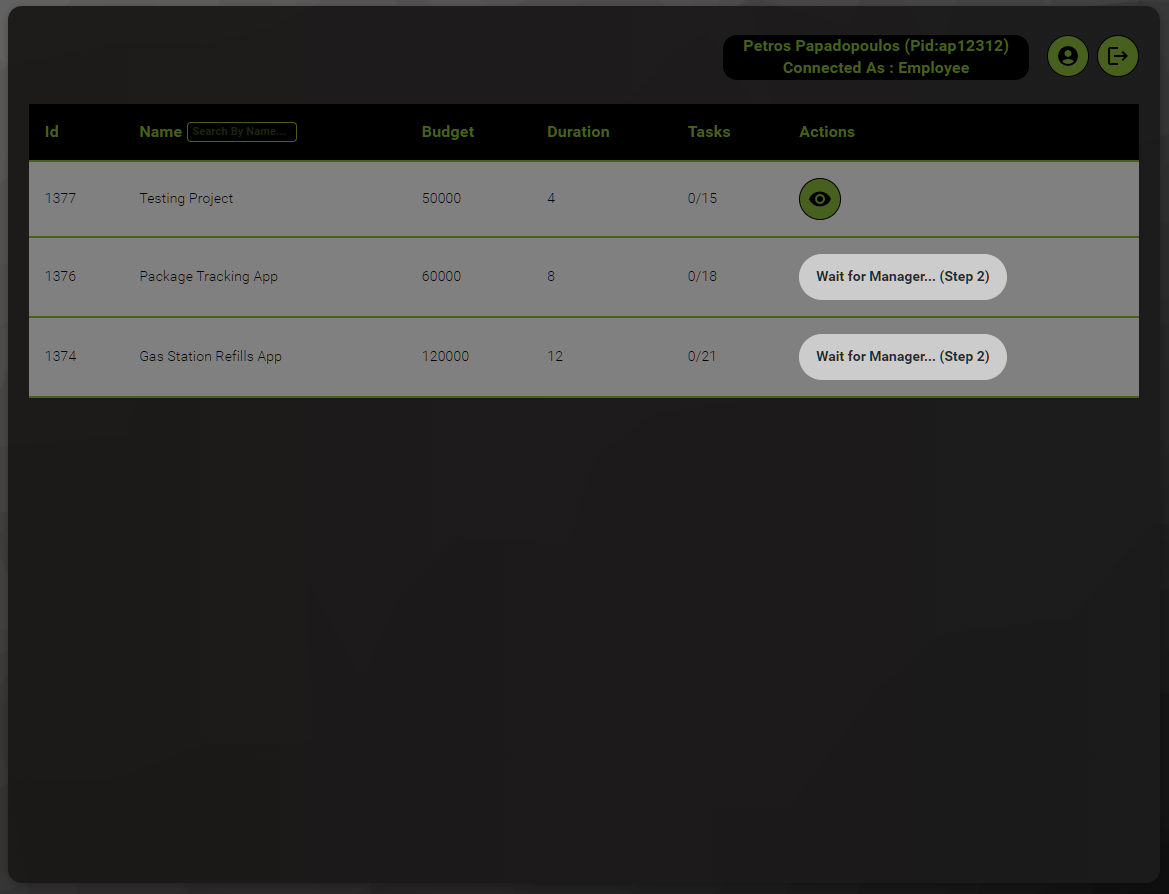


Μολις ο διαχειριστης ολοκληρωσει την τροποποιηση των στοιχειων επιλεγει το κουμπι Edit και τα νεα δεδομενα του υπαλληλου αποθηκευονται εκ νέου στην βαση δεδομενων, αλλα μπορει να πασα στιγμη να ακυρωσει την τροποποιηση μεσω και ενός κουμπιου Cancel.

Με την επιλογη του τριτου iconbutton, ο admin εχει την δυνατοτητα να αποδεσμευσει τον υπαλληλο από το συστημα. Επι της ουσιας και όπως ηδη εχει αναφερθει, ο υπαλληλος δεν διαγραφεται, αλλαζει μονο το state του στην βαση δεδομενων μεσω του αντιστοιχου api request. Με την χρηση ενός warning modal, ο admin ερωταται για την αποφαση του και επιλεγει το κουμπι Sure, αλλιως επιλεγει Cancel και ακυρωνει την διαδικασια.



Η τελευταια αρχικη σελιδα του συστηματος είναι αυτή που ο χρηστης συνδεεται ως employee. Και σε αυτή την τελευταια περιπτωση, η αρχικη κατασκευαστηκε με ιδιο τροπο με τις προηγουμενες αρχικες, με τις μονες αλλαγες να είναι στο περιεχομενο που εχει προσβαση ο υπαλληλος. Ετσι λοιπον στην λιστα που του εμφανιζεται, περιεχονται μονο τα projects στα οποια εχει λαβει μερος. Η μονη λειτουργια που του είναι διαθεσιμη είναι μονο η προβολη των project και των πληροφοριων και διαγραμματων τους.



Επισης η εφαρμογη επρεπε να παρεχει την δυνατοτητα ελεγχου των πληρωμων και των στοιχειων από τον ιδιο τον εκαστοτε υπαλληλο, οποτε τοποθετηθηκε καταλληλο iconbutton στο πανω δεξια μερος της αρχικης σελιδας, όπως και ένα iconbutton διπλα του που σχετιζεται με το logout. Το modal της προβολης των στοιχειων του και των πληρωμων του είναι ιδιο με το modal της προβολης υπαλληλου που κατασκευαστηκε από την πλευρα του admin.

Συνοψιζοντας, συνολικα το γραφικο περιβαλλον της εφαρμογης ακολουθησε μια λιτη σχεδιαση, προκειμενου να είναι ευκολα προσπελασιμα και κατανοητα όλα τα σταδια και οι ενεργειες στον χρηστη. Η χρηση της σχεδιαστικης γραμμης των modals, επιτυγχανει την καθαροτητα στις λειτουργιες, απομονωνει τις επιμερους διαδικασιες που λαμβανουν μερος, ενώ η αρχικη σελιδα υπαρχει παντα στο προσκηνιο και παραμενει οσο πιο καθαρη και απλη γινεται.  
Το logout button είναι αμεσα διαθεσιμο σε ολες τις σελιδες και συνολικα τα κουμπια που τοποθετηθηκαν στην εφαρμογη είναι ευδιακριτα και αλλαζουν καταλληλα, κατά το hover η την κατασταση ondisabled. Οπου κριθηκε απαραιτητο σε πολλα σημεια της εφαρμογης τοποθετηθηκαν τα καταλληλα cancel buttons ενώ με την κατασταση ondisabled κυριως σε submit τυπου buttons, αποφευχθηκε η προωρη υποβολη στοιχειων, οποτε και λειτουργουν σαν ένα σταδιο ελεγχου εγκυροτητας. Ελεγχος εγκυροτητας επισης εχει προστεθει στις φορμες εισαγωγης μεσω των textfields. Οι αρχικες λιστες που δημιουργηθηκαν, από το 8ο αντικειμενο και επειτα γινονται scrollable, ενώ οπου θεωρηθηκε απαραιτητο τοποθετηθηκε και ένα textfield αναζητησης της εκαστοτε λιστας. Η εφαρμογη οπου είναι αναγκαιο εχει προνοησει και προβαλει τα αντιστοιχα error messages, προλαμβανοντας και επιδεικνυοντας λαθη στο χρηστη.

***2 Επίλογος (Conclusions/Future work)***

Εδώ αναφέρουμε ότι θα ακολουθήσει σύνοψη της διπλωματικής.

***2.1Σύνοψη και συμπεράσματα***

Εδώ συνοψίζουμε τα αποτελέσματα της διπλωματικής και περιγράφουμε τα συμπεράσματα που προέκυψαν, αρνητικά και θετικά. Επιβεβαιώνουμε τη συνεισφορά της διπλωματικής στα προβλήματα που αναφέραμε στην εισαγωγή.

***2.2Μελλοντικές επεκτάσεις***

Εδώ δίνουμε ιδέες για επέκταση της διπλωματικής.

***3 Βιβλιογραφία***

<https://www.javatpoint.com/software-engineering-sdlc-models>

<https://stackify.com/what-is-sdlc/>

<https://phoenixnap.com/blog/software-development-life-cycle> <https://www.clouddefense.ai/blog/system-development-life-cycle>

<https://svitla.com/blog/system-development-life-cycle>

<https://www.softwaretestingmaterial.com/waterfall-model-in-sdlc/>

https://www.geeksforgeeks.org/software-paradigm-and-software-development-life-cycle-sdlc/

https://www.tutorialspoint.com/sdlc/sdlc\_v\_model.htm

<https://www.softwaretestingmaterial.com/v-model-in-sdlc/>

https://digitalstrategy.gov.gr/website/static/website/assets/uploads/digital\_strategy.pdf

***4 Παράρτημα***

***5 Οδηγίες Εγκατάστασης Υπηρεσίας***