Εθνικό και Καποδιστριακό Πανεπιστήμιο Αθηνών
Τμήμα Πληροφορικής και Τηλεπικοινωνιών

Ανάλυση και Σχεδίαση Συστημάτων Λογισμικού
Υποχρεωτική Εργασία Εαρινού Εξαμήνου 2025

Ονοματεπώνυμο : Πάτρος Μάριος AM : 1115202200143
email: mpatros13@gmail.com

Ονοματεπώνυμο : Τσάκωνας Βασίλης AM : 1115202200245
email: billts2004@gmail.com

Περιεχόμενα

1 Εισαγωγή

Καταλήξαμε στα παρακάτω διαγράμματα από κοινού. Ο τρόπος που συνεργαστήκαμε είναι ο εξής:
Συναντιόμασταν από κοντά και ο καθένας αναλάμβανε 1-2 διαγράμματα, στα οποία δούλευε μέχρι την επόμενη φορά που θα συναντιόμασταν ξανά. Τότε τα συζητούσαμε και με βάση τι είχε κάνει αυτός που είχε αναλάβει το εκάστοτε διάγραμμα καταλήγαμε στο τελικό έπειτα από συζήτηση και τυχόν αλλαγές. Τα διαγράμματα που είχε αναλάβει ο Βασίλης είναι τα: διάγραμμα ακολουθίας (Σενάριο 2) και τα διαγράμματα Δομημένης Ανάλυσης. Τα διαγράμματα του Μάριου είναι τα: διάγραμμα περιπτώσεων χρήσης, διάγραμμα ακολουθίας(Σενάριο 1), διάγραμμα δραστηριοτήτων(Σενάριο 2). Στα υπόλοιπα δουλέψαμε μαζί κατά τη διάρκεια των συναντήσεών μας. Οι συναντήσεις γινόντουσαν ανά 2-3 μέρες έτσι ώστε να έχουμε χρόνο να προετοιμάσουμε τα διαγράμματα κάνοντας ταυτόχρονα και μια μικρή επανάληψη στη θεωρία.

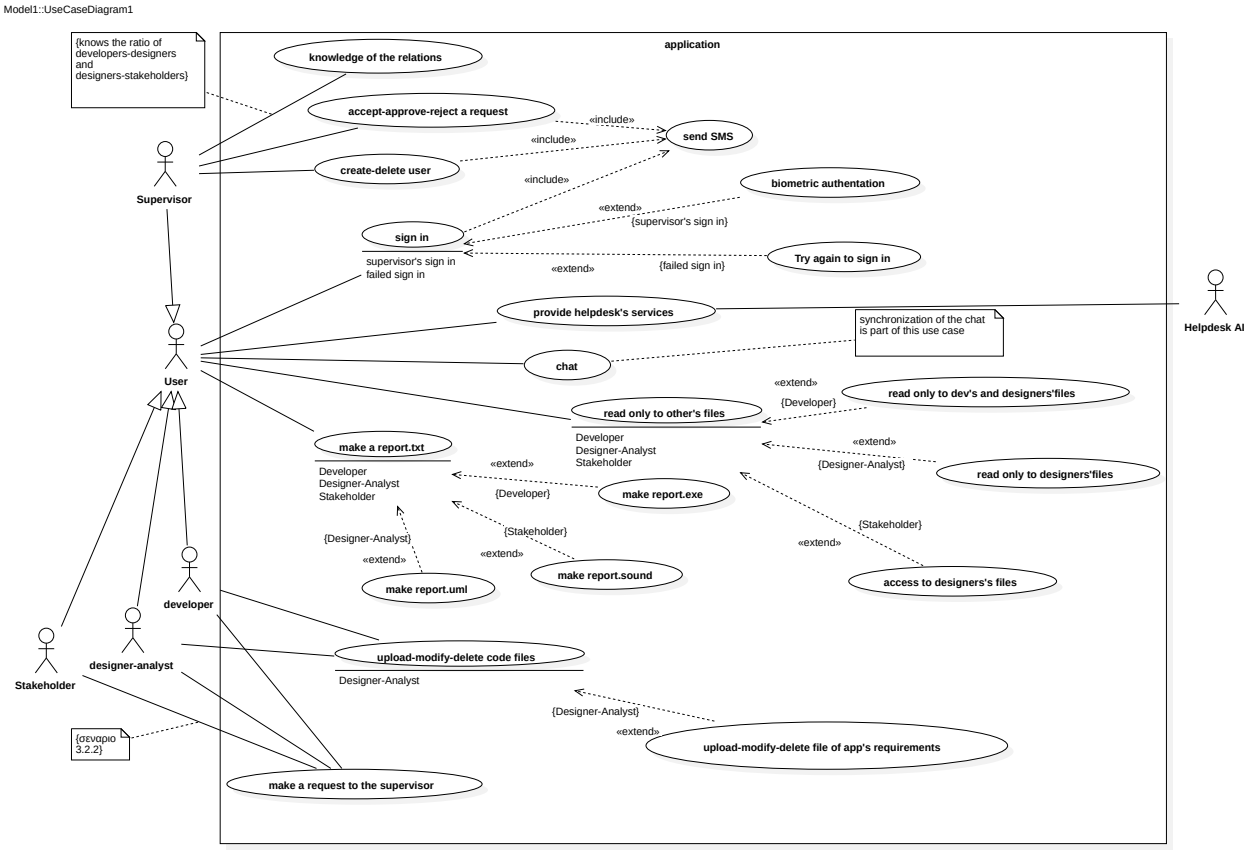
Όσον αφορά την εκφώνηση, δεν είχαμε κάποιο θέμα. Ήταν κατανοητή, κατατοπιστική και σε συνδυασμό με τις μικρές διευκρινήσεις σας κατά τη διάρκεια των διαλέξεων δεν είχαμε κάποιο θέμα σε σχέση με την κατανόηση της.
Η μόνη παρατήρηση μας είναι ότι στο κείμενο διαπιστώσαμε ότι υπάρχει ένα typo οπότε όταν η εκφώνηση έγραφε Stateholder εμείς το θεωρήσαμε Stakeholder.

Σύμφωνα με την εκφώνηση, ο Stakeholder δεν έχει τη δυνατότητα να στείλει κάποιο αίτημα στον επόπτη, αλλά στο σενάριο 2 λέει ότι τελικά μπορεί. Οπότε εμείς πήγαμε με βάση το σενάριο 2 για όλα τα διαγράμματα, όπως είχε διευκρινισθεί και στις συζητήσεις του e-class.

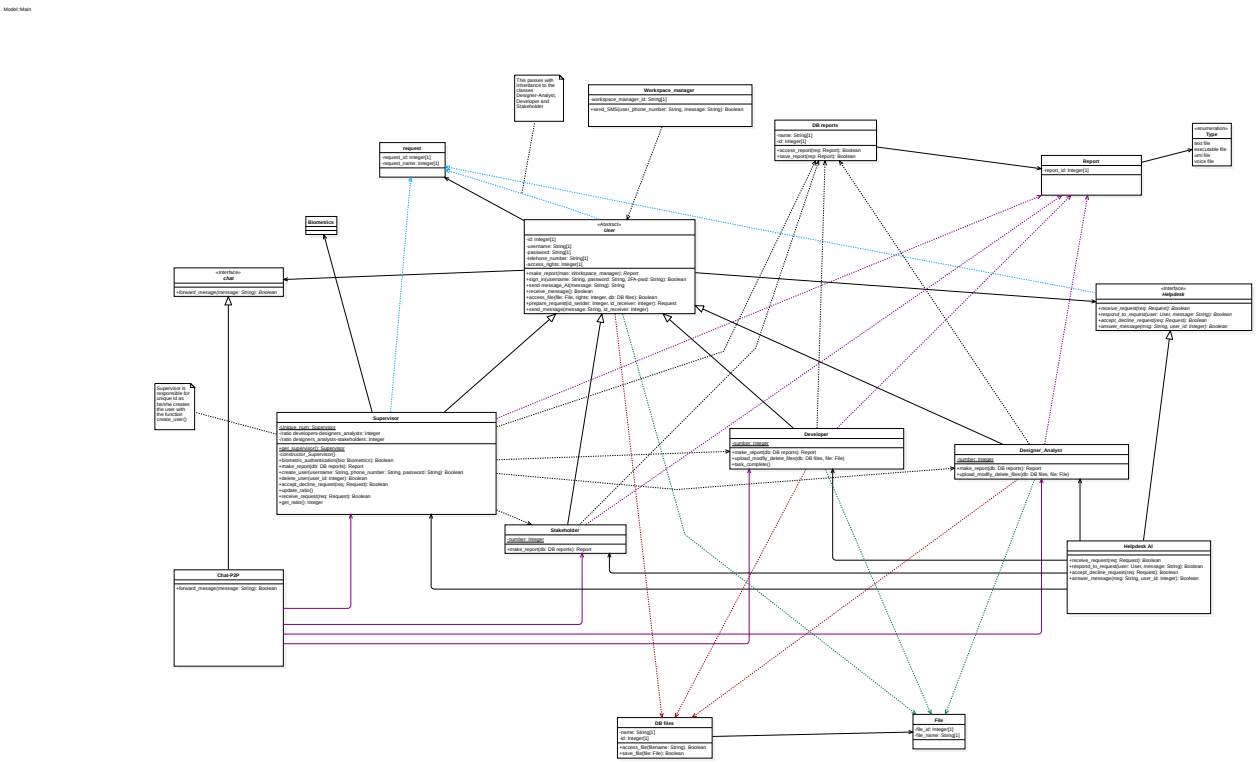
Τα διαγράμματα που μας δυσκόλεψαν περισσότερο ήταν το διάγραμμα μηχανής καταστάσεων και τα διαγράμματα ακολουθίας και δραστηριοτήτων του 2ου σεναρίου. Στο διάγραμμα μηχανής καταστάσεων δυσκολευτήκαμε καθώς ήταν στην αρχή δύσκολο να διακρίνουμε τις καταστάσεις στις οποίες μπορούσε να βρεθεί το σύστημα. Το σενάριο 2 είχε αρκετά περιεκτικές και συμπιεσμένες πληροφορίες και αυτό μας δυσκόλεψε αρκετά στην αρχή.

2 Use Case Diagram

Διευκρινήσεις:
Το κύριο σενάριο επιτυχίας στο Use case "Make a report.txt" αφορά τον supervisor και τα άλλα 3 extends είναι των υπολοίπων χρηστών όπως αποτυπώνεται και στο διάγραμμα.
Το κύριο σενάριο επιτυχίας στο Use case "read only other's files" αφορά τον supervisor και τα άλλα 3 extends είναι των υπολοίπων χρηστών όπως αποτυπώνεται και στο διάγραμμα.
Το κύριο σενάριο επιτυχίας στο Use case "upload-modify-delete code files" αφορά τον developer και το extend είναι του designer-analyst.
Ο συγχρονισμός του chat εμπεριέχεται στο use case "chat".



3 Class Diagram



Διευκρινήσεις-Παραδοχές:
Η κλάση Workspace_manager είναι ο admin της εφαρμογής αυτός είναι υπεύθυνος για να στέλνει τα SMS ενημέρωσης.
Η κλάση biometrics περιέχει τα βιομετρικά στοιχεία του επόπτη.
Η συνάρτηση send_message_AI() στέλνει κάποιο μήνυμα στο Helpdesk.
Το prepare_request() είναι συνάρτηση μέσα στο User,η οποία γίνεται στη συνέχεια override στη κλάση Supervisor καθώς αυτός μπορεί να στείλει αίτημα μόνο στο Helpdesk ενώ οι Stakeholders, Developers και Designers-Analysts μπορούν να στείλουν και στον επόπτη και στο Helpdesk.
Το number στους Stakeholders, Developers και Designers-Analysts είναι static και λειτουργεί ως ένας counter για το πόσα αντικείμενα έχουν φτιαχτεί. Έτσι ο Supervisor το χρησιμοποιεί για να φτιάξει τις αναλογίες που χρειάζεται.
Η μεταβλητή rights είναι ένας αριθμός που δείχνει τι rights έχει ο κάθε user και σε ποια αρχεία έχει πρόσβαση. Η λογική είναι όμοια με τη λογική που υπάρχει στα files του υπολογιστή μας με read, write, execute access(π.χ. το 6 σημαίνει κάποιος έχει και τα 3 rights).
Μια παραδοχή δική μας που δεν ήταν στις απαιτήσεις του συστήματος ήταν να βάλουμε σαν κλάσεις 2 βάσεις δεδομένων έτσι ώστε να αποθηκεύονται τα files και τα reports για αυτόν τον λόγο υπάρχουν οι εξαρτήσεις μεταξύ των users(Developers, Stakeholders, etc) και των DB files, file, DB reports, report. Καταλήξαμε σε αυτό καθώς θέλαμε να αποτυπώσουμε τη λογική των reports και των φακέλων με μια πιο ρεαλιστική υλοποίηση και πιο εύχρηστη για τον προγραμματιστή.
Ο Επόπτης(Supervisor) είναι αυτός που δημιουργεί τους χρήστες οπότε αυτός είναι υπεύθυνος στο να περάσει μοναδικό id έτσι ώστε να μην έχουμε διπλότυπα.
Ο λόγος που υπάρχουν dependencies μεταξύ του Supervisor και των υπολοίπων χρηστών είναι επειδή εκείνος του

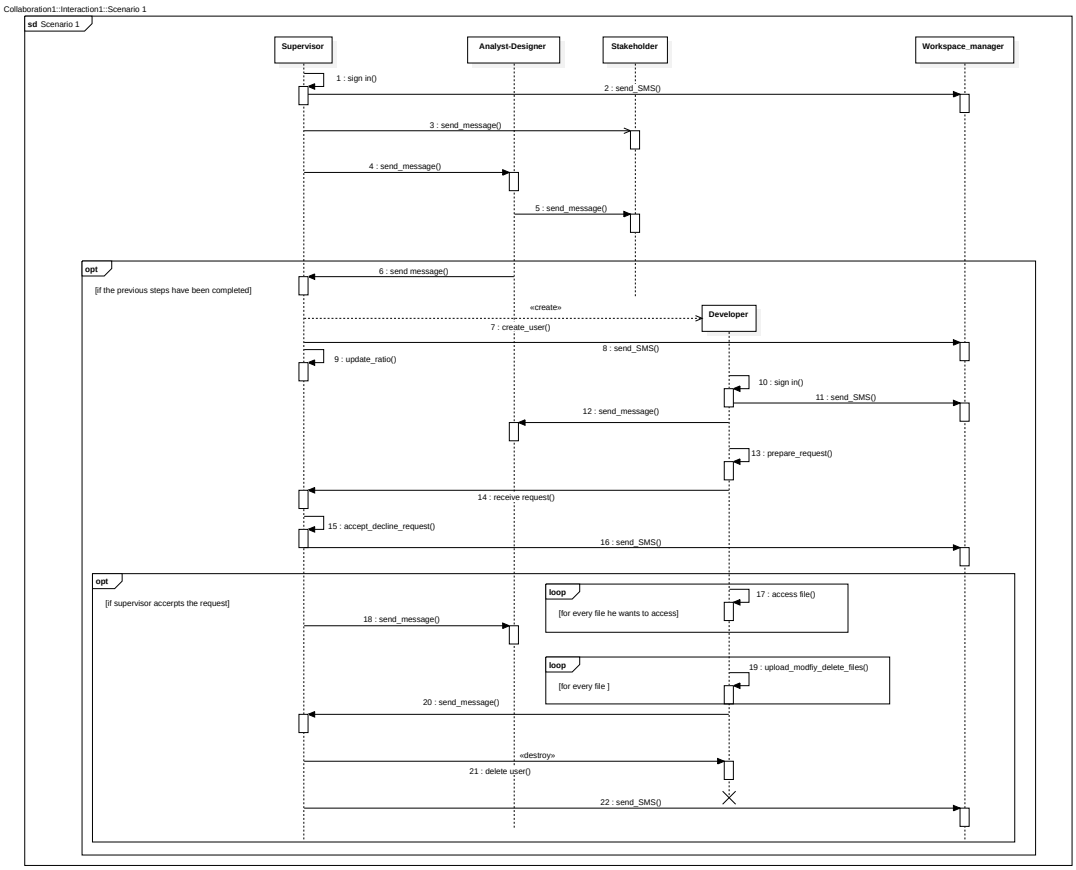
δημιουργεί ή διαγράφει.
Η dependency σχέση από τον User στο request κληρονομείται και στις κλάσεις Developer, Stakeholder και Designer-Analyst.
Ένα από τα πρότυπα σχεδίασης που χρησιμοποιήθηκαν σε αυτό το class diagram είναι το πρότυπο Singleton για να αναπαραστήσει ότι ο επόπτης είναι μόνο ένας. Ένα άλλο πρότυπο σχεδίασης που χρησιμοποιήθηκε 2 φορές είναι το πρότυπο Mediator για την αναπαράσταση του chat μεταξύ 2 χρηστών και ενός χρήστη με το Helpdesk. Έτσι οι χρήστες μπορούν να επικοινωνήσουνε μεταξύ τους και το Helpdesk μπορεί να επικοινωνήσει με όλους (έτσι αποφεύχθηκαν εξαρτήσεις των κλάσεων που θα έκαναν το διάγραμμα πολύ μπερδεμένο και δυσνόητο).

4 Sequence Diagram

Διευρκινήσεις:
Και στα 2 σενάρια θεωρούμε πως είναι απαραίτητο να φαίνεται η κλήση της συνάρτησης send_SMS(create_user, delete_user,sign_in,accept_delete_user), καθώς ενημερώνει τον εκάστοτε user για την απόφαση του επόπτη ή για κάποια άλλη πληροφορία. Επομένως αποτελεί σημαντικό μέρος των μεθόδων και για αυτό το αποτυπώνουμε και εμείς.

4.1 Σενάριο νο1

Όταν στην εκφώνηση λέει ότι 2 άνθρωποι επικοινωνούν, αυτό που καταλαβαίνουμε είναι ότι αυτός που επικοινωνεί στέλνει μήνυμα στον άλλον και ο δεύτερος του απαντάει χωρίς όμως να χρειαστεί να αποτυπωθεί στο διάγραμμα καθώς ευκόλως εννοείται και ταυτόχρονα δεν είναι απαραίτητο για την ομαλή υλοποίηση του σεναρίου.



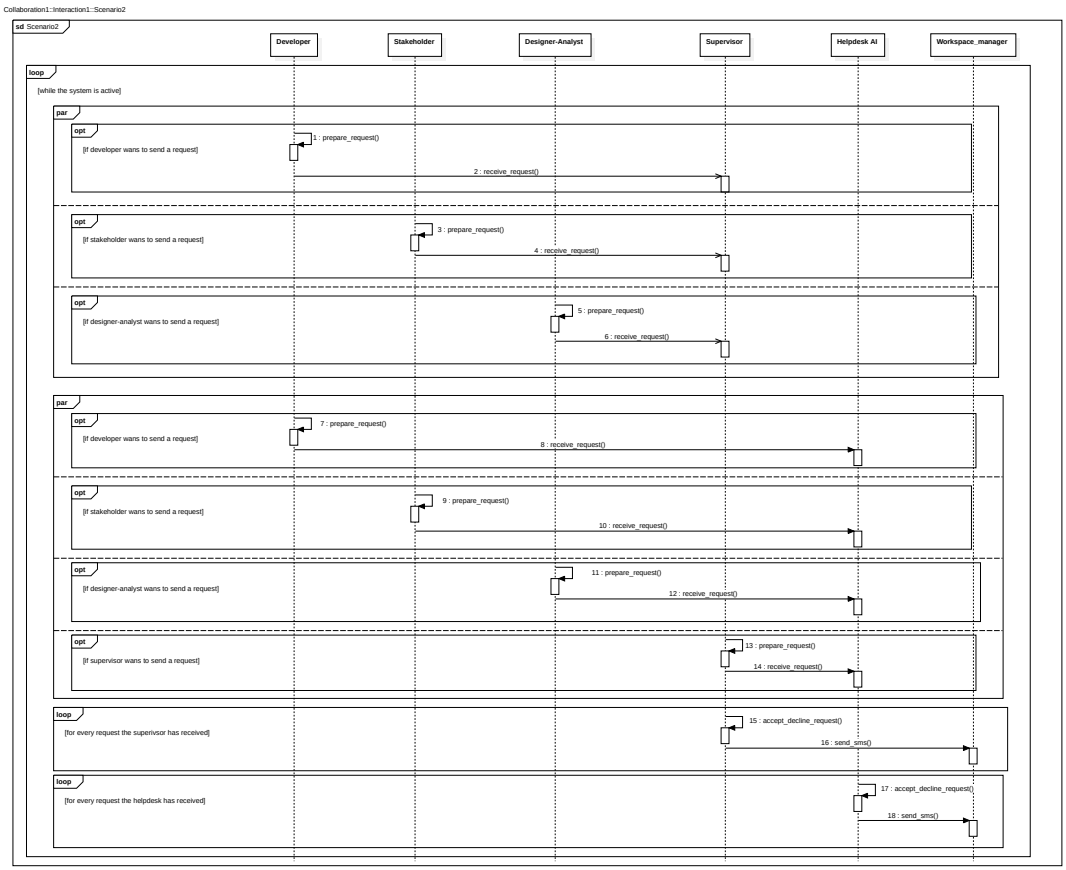
4.2 Σενάριο νο2

Διευκρινήσεις:

Οι χρήστες μπορούν να στείλουν παράλληλα αιτήματα στον επόπτη και στη συνέχεια μπορούν να στείλουν πάλι παράλληλα αιτήματα στο helpdesk αλλά δε μπορούν να στείλουν παράλληλα και στους 2. Οπότε πρώτα θα πρέπει να στείλουν στον επόπτη (όποιος επιθυμεί) και στη συνέχεια να στείλουν στο Helpdesk το αίτημά τους(όποιος πάλι επιθυμεί), μόνο με αυτή τη σειρά.

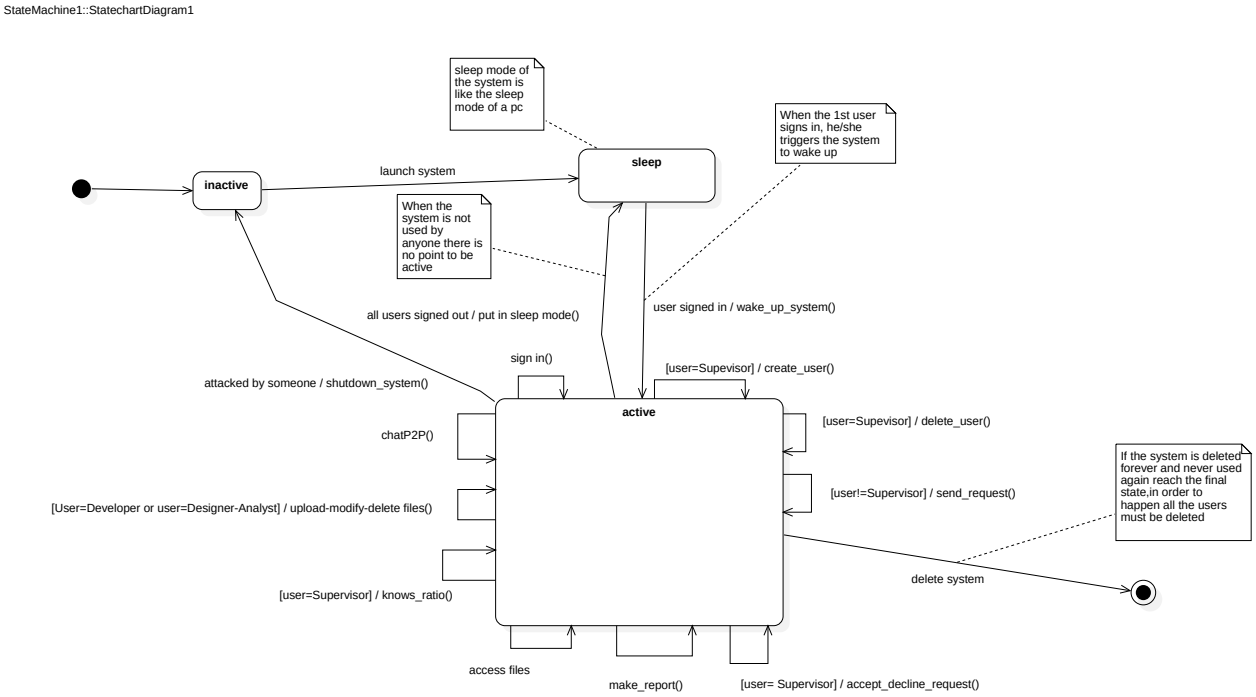
Όταν κάποιος(supervisor/Helpdesk) λαμβάνει ένα αίτημα η απάντηση του είναι ένα μήνυμα (flag) σαν ok το έλαβα και όταν βρει χρόνο κάνει accept ή decline και στη συνέχεια ενημερώνει τον αιτούντα μέσω SMS. Η απάντηση δεν φαίνεται στο διάγραμμα αλλά όπως είναι φυσικό εννοείται. Οι χρήστες δε χρειάζεται να λάβουν αυτό το ok για να προχωρήσουν όταν κάνουν κάποιο αίτημα στον επόπτη αλλά πρέπει να το λάβουν από το Helpdesk για να συνεχίσουν.

Προσθέσαμε ότι μετά την υποβολή των requests σε Helpdesk και επόπτη τότε και οι δυο αποδέχονται ή απορρίπτουν το αίτημα, αναλόγως, και ενημερώνουν τον εκάστοτε αιτούντα. Δεν αποτελούσε μέρος του δεύτερου σεναρίου αλλά είναι η φυσική ροή του και για αυτό το προσθέσαμε.



5 Statechart Diagram

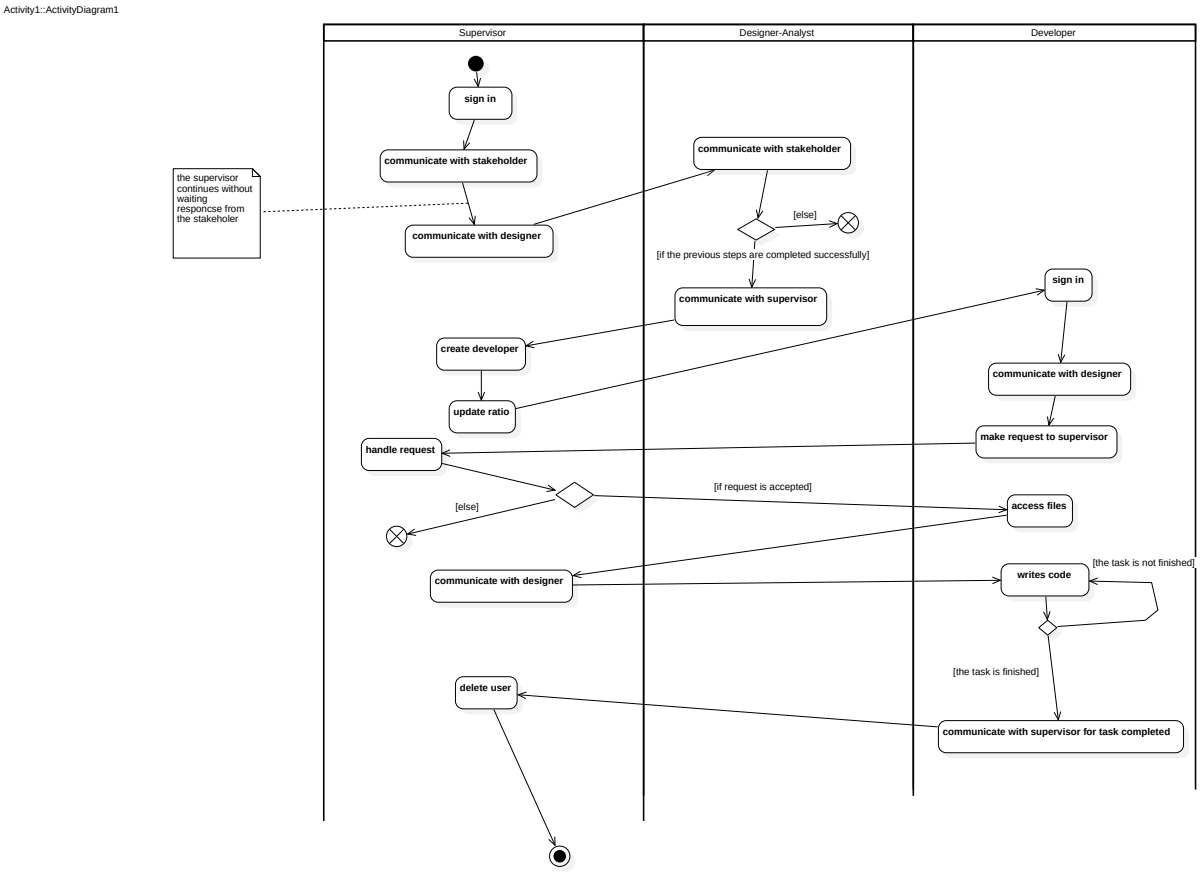
Διευκρινήσεις/Παραδοχές:
Το σύστημα μας μπορεί να βρίσκεται σε 3 καταστάσεις οι οποίες είναι: sleep, inactive, active. Η κατάσταση sleep αντιστοιχεί στο sleep mode που υπάρχει σε έναν υπολογιστή έτσι ώστε να εξοικονομούνται πόροι. Για να είναι σε sleep mode θα πρέπει κανένας user να μη είναι signed in και όταν τουλάχιστον ένας κάνει sign in τότε βγαίνει από sleep mode και πάει στην κατάσταση active.
Για να πάει σε κατάσταση inactive από active πρέπει κάτι να μη πάει καλά και να κλείσει το σύστημα. Αυτό μπορεί να γίνει αν επιτεθεί στην εφαρμογή κάποιος κακόβουλος. Όταν αυτό γίνει αντιληπτό το σύστημα κάνει shutdown αυτομάτως για να προστατεύσει τα δεδομένα του. Αυτή ήταν μια δική μας παραδοχή η οποία δεν αποτυπώνεται στην εκφώνηση.
Για να φτάσει στο final state θα πρέπει για κάποιο λόγο να πάψει να υπάρχει το σύστημα και αυτός που το έχει να θέλει να το διαγράψει και να μη το χρησιμοποιεί πια.(παραδοχή δική μας)
Το sign in που είναι self transistion στη κατάσταση active αφορά το sign ενός χρήστη αφότου έχει κάνει sign in ο πρώτος που θα το "ξυπνήσει".



6 Activity Diagram

6.1 Σενάριο νο1

Διευκρινήσεις:
Στις περιπτώσεις όπου γίνεται ένας έλεγχος (π.χ. άμα έχει ολοκληρωθεί η διαδικασία ή αν ο επόπτης έχει αποδεχθεί το αίτημα) και αυτός αποτύχει τότε το διάγραμμα καταλήγει σε flow final το οποίο σημαίνει ότι απλά τερματίζει η διαδικασία γιατί κάτι απέτυχε.
Επιπλέον το action "writes code" αντιστοιχεί στην συγγραφή κώδικα που αναφέρεται στην εκφώνηση και κάνει αυτή την ενέργεια μέχρι να ολοκληρώσει το έργο που του έχει ανατεθεί(loop).



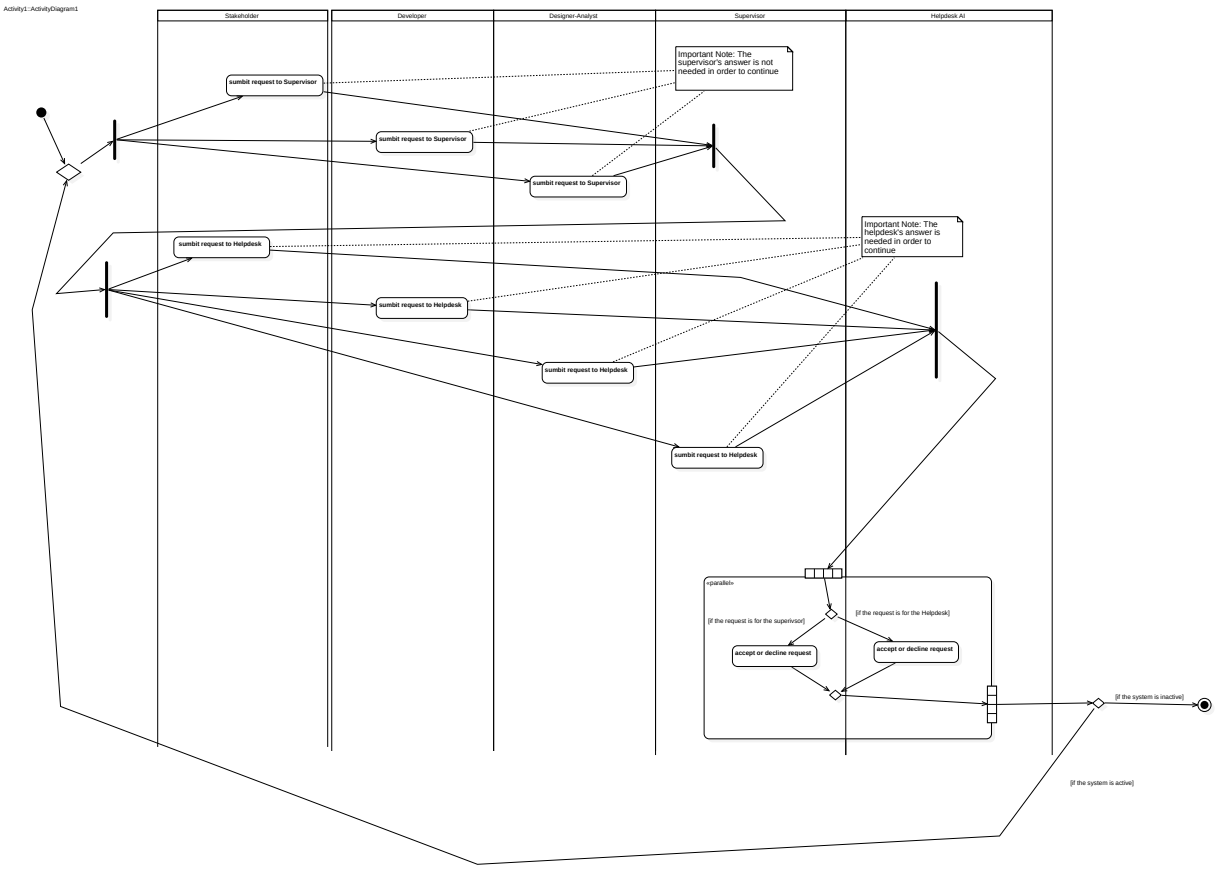
6.2 Σενάριο νο2

Διευκρινήσεις:

Όπως και στο διάγραμμα ακολουθίας θεωρούμε ότι οι χρήστες μπορούν να στείλουν σειριακά αιτήματα στον επόπτη και στο helpdesk, δηλαδή παράλληλα οι χρήστες μπορούν να στείλουν στον επόπτη και μετά πάλι όλοι παράλληλα να στείλουν κάποιο αίτημα στο Helpdesk.

Επιπλέον όπως πάλι και στο διάγραμμα ακολουθίας του σεναρίου 2, προσθέσαμε ότι μετά την υποβολή των requests σε Helpdesk και επόπτη τότε και οι δυο αποδέχονται ή απορρίπτουν το αίτημα, αναλόγως, και ενημερώνουν τον εκάστοτε αιτούντα. Δεν αποτελούσε μέρος του δεύτερου σεναρίου αλλά είναι η φυσική ροή του.

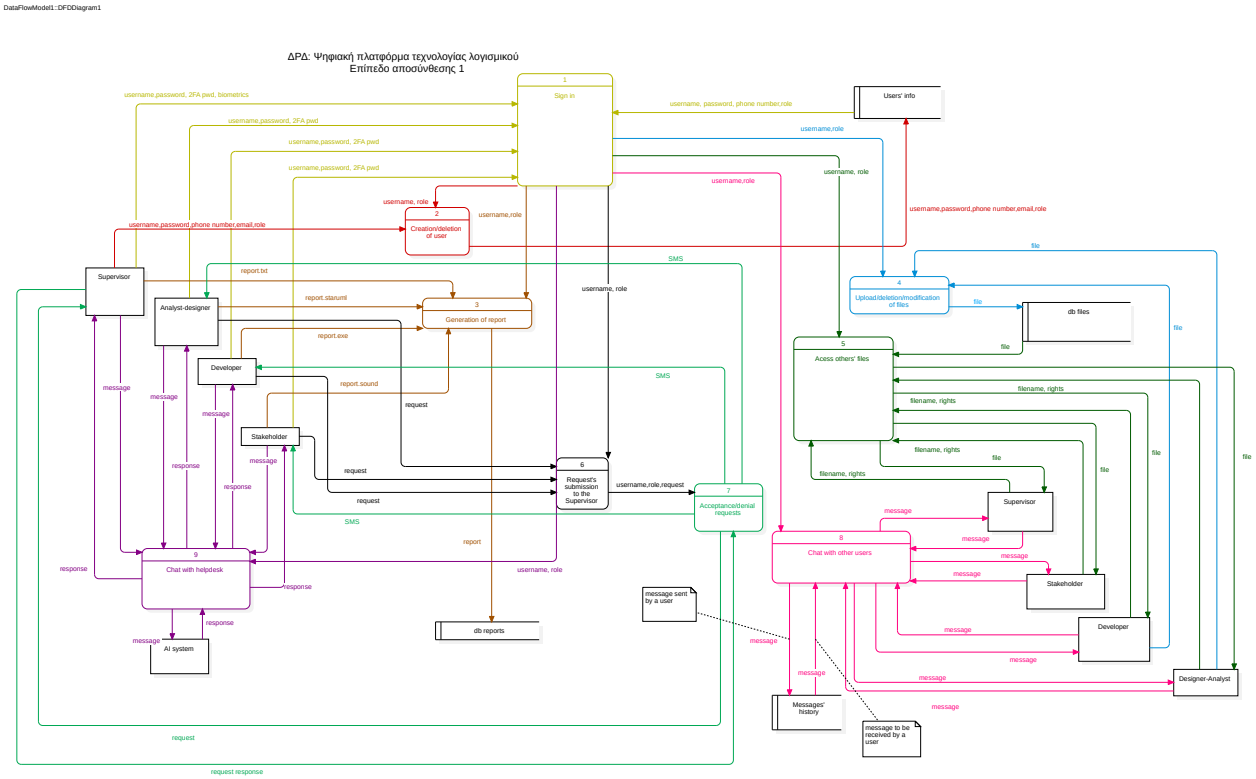
Η περιοχή επέκτασης με όνομα «parallel» δείχνει ότι ο επόπτης ή το Helpdesk, αναλόγως για ποιον είναι το αίτημα, το αποδέχεται ή το απορρίπτει και αυτό συμβαίνει για κάθε αίτημα που υπάρχει. Στο σχεδιάγραμμα παρακάτω το περίγραμμά του είναι με συνεχόμενη γραμμή ενώ στο πρόγραμμα Staruml είναι με διακεκομμένη όπως δηλαδή απεικονίζεται μια περιοχή επέκτασης με βάση τη θεωρία που έχουμε δει. Απλά θέλουμε να τονίσουμε ότι είναι περιοχή επέκτασης.



7 Dataflow Diagram

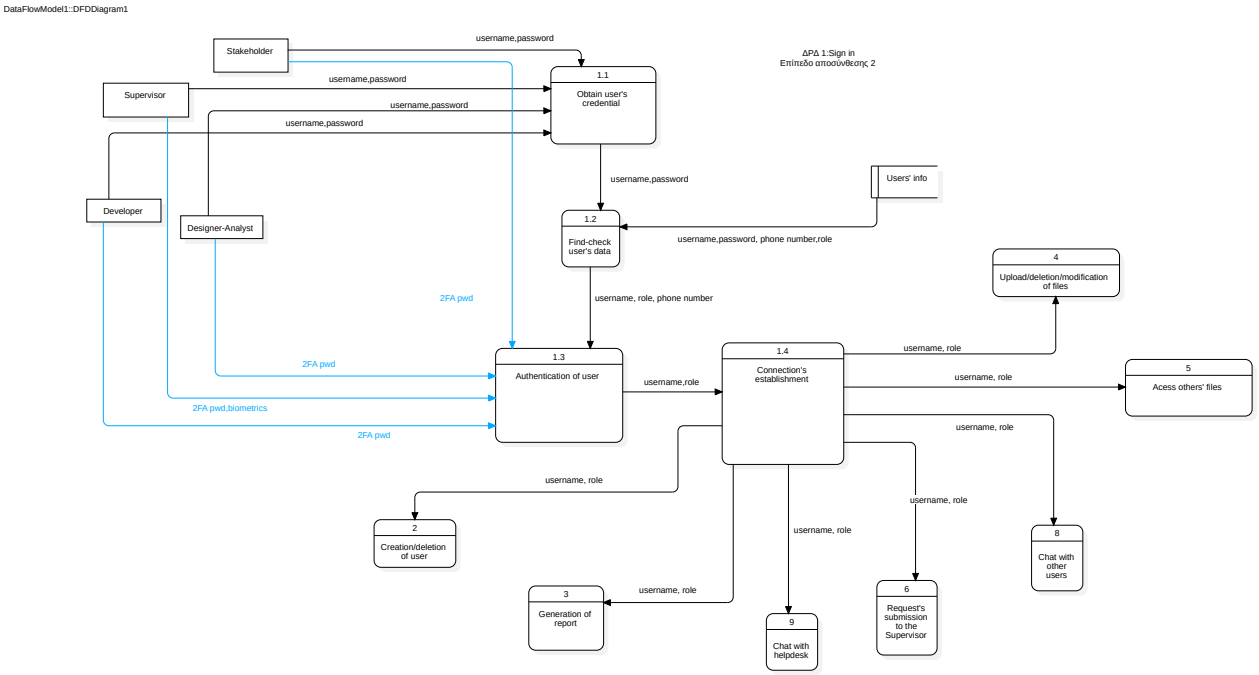
7.1 Επίπεδο Αφαίρεσης 1

Στο παρακάτω ΔΡΔ επιπέδου 1 αποτυπώνονται οι κύριες διεργασίες της εφαρμογής. Η πρώτη είναι το sign in και όλες οι υπόλοιπες μπορούν να γίνουν αμέσως μετά αναλόγως ποιος είναι ο χρήστης και τι λειτουργία θα επιλέξει. Έχουμε βάλει τα external entities(Supervisor, Stakeholder, Developer, Designer-Analyst) δυο φορές για καλύτερη, ευκολότερη κατανόηση και αποφυγή ενός πιο "μπερδεμένου" διαγράμματος. Η επιλογή των χρωμάτων έγινε για τον ίδιο λόγο πάλι. Τα συγκεκριμένα χρώματα έχουν επιλεγεί για να υπάρχει αντίθεση μεταξύ των διάφορων διεργασιών αλλά και του background.
Στο process "Creation/deletion user" τα username, password, phone number, email, role από τον Supervisor και προς το db "User's info" είναι τα στοιχεία του νέου/διαγραμμένου χρήστη. Το username, role που προέρχεται από το process 1 "Sign in" είναι το username, role του χρήστη προκειμένη περίπτωση(Supervisor).
Επιπλέον, η αποστολή SMS δεν είναι μέρος του συστήματος που υλοποιούμε εμείς καθώς έρχεται στον αριθμό τηλεφώνου του εκάστοτε χρήστη και για αυτό δεν αποτυπώνεται σε κανένα από τα δυο ΔΡΔ.
Επιπλέον, για την επικοινωνία μεταξύ των users έχουμε μια βάση όπου κρατάμε το ιστορικό των μηνυμάτων γιατί είναι πιο πρακτικό κάποιος να έχει πρόσβαση σε όλο το ιστορικό μιας συνομιλίας παρά να βλέπει μόνο το πιο πρόσφατο μήνυμα.



7.2 Επίπεδο Αφαίρεσης 2

Η διεργασία με τις περισσότερες ροές δεδομένων(12) είναι η διεργασία 1 με όνομα "sign in". Παρακάτω είναι το ΔΡΔ Επιπέδου αποσύνθεσης 2 της διαδικασίας sign in.



7.3 Πίνακας Αποφάσεων

Ο πίνακας που παρατίθεται είναι πίνακας αποφάσεων μικτών καταχωρήσεων.

	1	**2**	**3**	**4**	**5**
Επόπτης συμφωνεί	N	Y	Y	Y	Y
Stakeholder συμφωνεί	-	N	Y	Y	Y
Designer-Analyst συμφωνεί	-	-	N	Y	Y
Developer συμφωνεί	-	-	-	Y	N
Παράδοση λογισμικού	Μη-παράδοση	Μη-παράδοση	Μειωμένη χρέωση	Πλήρης χρέωση	Πλήρης χρέωση

Διευκρινήσεις:

Από τα δεδομένα της εκφώνησης έλειπε το σενάριο 5 όπου συμφωνούν όλοι εκτός του Developer. Η παραδοχή μας σε αυτήν την περίπτωση είναι πως το λογισμικό παραδίδεται στον πελάτη με πλήρη χρέωση.