

Εφαρμογή Διαχείρισης Φαρμακαποθήκης

Pharmacy Information System

By PHARMASIX

Project Manager: Κωνσταντίνος Παπανάγνου 4378

Developers: Παναγιώτης Σκλίδας 4360 & Ελένη Γιβανούδη 4314

Πελάτες: Ντενίτσα Γκρούνοβα 4326 & Έλλη Ράπτη 4319

Ελεγκτής: Αθανάσιος Μελισσός 4375

Περιεχόμενα

[Contributions 4](#_Toc74078448)

[Διοίκηση Έργου 4](#_Toc74078449)

[Προσωπικές διαφοροποιήσεις μοντέλου SCRUM 5](#_Toc74078450)

[Στελέχωση 7](#_Toc74078451)

[Αρχικά Διαγράμματα 7](#_Toc74078452)

[Διάγραμμα PERT 7](#_Toc74078453)

[Διάγραμμα Gantt 9](#_Toc74078454)

[Αναθέσεις Προσωπικού 9](#_Toc74078455)

[Μοντέλο Κύκλου Ζωής 10](#_Toc74078456)

[Τελικά Διαγράμματα: 11](#_Toc74078457)

[Διάγραμμα PERT: 11](#_Toc74078458)

[Διάγραμμα Gantt 13](#_Toc74078459)

[Εκτίμηση Κόστους 13](#_Toc74078460)

[Αναδρομική Ανάλυση Έργου 14](#_Toc74078461)

[Last Part 16](#_Toc74078462)

[Test πρώτου sprint. 16](#_Toc74078463)

[Πρώτο μέρος test λειτουργικότητας εφαρμογής 16](#_Toc74078464)

[Πρώτο διάγραμμα γενικού πλάνου τεστ (Log-in screen): 17](#_Toc74078465)

[Διάγραμμα τεστ εισαγωγής απλού χρήστη: 18](#_Toc74078466)

[Διάγραμμα τεστ εισαγωγής admin: 19](#_Toc74078467)

[Διάγραμμα τεστ δημιουργίας ενώς χρήστη: 20](#_Toc74078468)

[Διάγραμμα τεστ αλλαγής στοιχείων χρήστη: 21](#_Toc74078469)

[Διάγραμμα τεστ διαγραφής στοιχείων χρήστη: 22](#_Toc74078470)

[Περιγραφή ανάλυση και αποτελέσματα Unit Testing του Log-In Screen 23](#_Toc74078471)

[Πρώτο μέρος test διεπαφής 26](#_Toc74078472)

[Test δεύτερου sprint. 31](#_Toc74078473)

[Δεύτερο μέρος test λειτουργικότητας της εφαρμογής (Μενού φαρμάκων) 31](#_Toc74078474)

[Διάγραμμα test ελέγχου εισόδων εξόδων (2ου μέρους): 33](#_Toc74078475)

[Διάγραμμα test ελέγχου αποθηκάριου: 34](#_Toc74078476)

[Διάγραμμα test ελέγχου μάρκετινγκ: 35](#_Toc74078477)

[Διάγραμμα test ελέγχου πωλητή φαρμάκων: 36](#_Toc74078478)

[Δεύτερο μέρος δοκιμών διεπαφής (Μενού φαρμάκων) 37](#_Toc74078479)

[Test τρίτου Sprint. 44](#_Toc74078480)

[Τρίτο μέρος test λειτουργικότητας της εφαρμογής (Ανανεωμένα μενού) 44](#_Toc74078481)

[Διάγραμμα ελέγχου λειτουργίας προσθήκης και αλλαγής στοιχείων πελάτη 46](#_Toc74078482)

[Διάγραμμα ελέγχου λειτουργίας διαγραφή πελάτη 47](#_Toc74078483)

[Διάγραμμα ελέγχου προσθήκης παραγγελίας μέσω επιλογής φαρμακοποιού 49](#_Toc74078484)

[Τρίτο μέρος test διεπαφής (Ανανεωμένα μενού) 50](#_Toc74078485)

[Αναφορές με χρονική σειρά. 56](#_Toc74078486)

[Αναφορά προβληματικής συμπεριφοράς εφαρμογής (1.1) 56](#_Toc74078487)

[Αναφορά προβληματικής συμπεριφοράς εφαρμογής (2.1) 60](#_Toc74078488)

[Ανάλυση εύρεσης λάθους (διάγραμμα) 61](#_Toc74078489)

[Αναφορά προβληματικής συμπεριφοράς εφαρμογής (2.2) 62](#_Toc74078490)

[Ανάλυση εύρεσης λάθους (διαγράμματα) 64](#_Toc74078491)

[Αναφορά προβληματικής συμπεριφοράς εφαρμογής (3.1) 65](#_Toc74078492)

[Ανάλυση εύρεσης λάθους (διάγραμμα) 66](#_Toc74078493)

[Αναφορά προβληματικής συμπεριφοράς εφαρμογής (3.2) 67](#_Toc74078494)

[Ανάλυση εύρεσης λάθους (διάγραμμα) 67](#_Toc74078495)

[Αναφορά προβληματικής συμπεριφοράς εφαρμογής (3.3) 70](#_Toc74078496)

[Ανάλυση εύρεσης λάθους (διάγραμμα) 70](#_Toc74078497)

[Αναφορά προβληματικής συμπεριφοράς εφαρμογής (3.4) 73](#_Toc74078498)

[Ανάλυση εύρεσης λάθους (διάγραμμα) 73](#_Toc74078499)

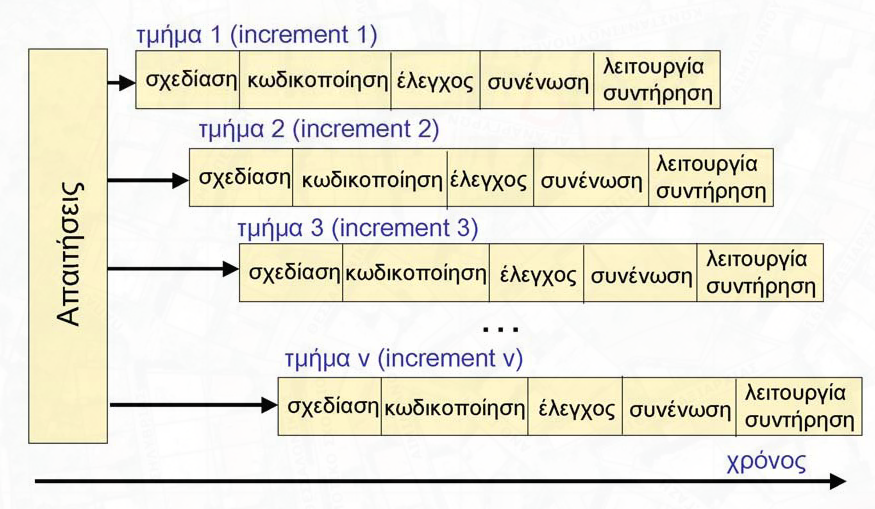
[Βιβλιογραφία 75](#_Toc74078500)

# Contributions

Σε αυτήν την εργασία δούλεψαν όλα τα μέλη στα πόστα τους, και βοήθησαν επίσης όπου μπορούσαν. Πιο συγκεκριμένα, ο Project Manager ασχολήθηκε με όλα τα διαδικαστικά του project και το μοντέλο κύκλου ζωής. Επίσης, έπαιξε και τον ρόλο του Scrum Master λόγω του μοντέλου κύκλου ζωής που επιλέχθηκε για την καλύτερη δυνατή οργάνωση του Project. Παράλληλα με αυτά, βοήθησε σημαντικά στον προγραμματισμό της εφαρμογής, στην οργάνωση και την συνένωση καθώς έπαιξε σημαντικό ρόλο στην αλλαγή της γλώσσας προγραμματισμού από java σε C#. Βοήθησε τους προγραμματιστές να εξοικειωθούν με την γλώσσα προγραμματισμού C# και ήταν παρών για ότι πρόβλημα προέκυπτε. Έδωσε λύσεις και βοήθησε κατά την εξαγωγή προδιαγραφών και πιο συγκεκριμένα στον σχεδιασμό των πινάκων προδιαγραφών καθώς βοήθησε ελάχιστα και στην διεξαγωγή των ελέγχων, ενημερώνοντας τον ελεγκτή για πιθανά λάθη που εντόπισε. Από τους 2 προγραμματιστές, η συνεισφορά του Παναγιώτη Σκλίδα, ήταν αξιοσημείωτη καθώς πέρα από το δικό του πόστο, βοήθησε και τον Project Manager στην συνένωση αυτού του εγγράφου και είναι υπεύθυνος εξ ολοκλήρου για την δημιουργία του User Manual και τον προγραμματισμό του Backend της εφαρμογής μας. Η δεύτερη προγραμματίστρια, Ελένη Γιβανούδη ήταν υπεύθυνη για το αισθητικό κομμάτι της εφαρμογής, συνεπώς ανέλαβε να σχεδιάσει τις οθόνες προς έγκριση και μετέπειτα να τις υλοποιήσει για την εφαρμογή. Οι πελάτες μας δούλεψαν το κομμάτι τους μαζί μετά από δική τους συνεννόηση και τέλος ο ελεγκτής ήταν υπεύθυνος για όλους τους ελέγχους του προγράμματος και την ενημέρωση των προγραμματιστών για τα ενδεχόμενα και επιβεβαιωμένα bugs.

# Διοίκηση Έργου

Το κύριο μέλημα μας σαν ομάδα είναι να δουλεύουμε όλοι μαζί όσο πιο μεθοδικά γίνεται για να έχουμε όσο το δυνατόν καλύτερη μεταχείριση του χρόνου και των πόρων που μας είναι διαθέσιμοι. Για αυτόν λόγο, η μέθοδος διοίκησης έργου που επιλέχτηκε από την ομάδα μας είναι η μέθοδος SCRUM με μερικές ιδιαιτερότητες. Η μέθοδος SCRUM πέρα από του ότι είναι ευέλικτη σε περιπτώσεις λαθών, μας επιτρέπει να χωρίσουμε το έργο που αναλαμβάνουμε σε υποέργα από τα οποία μπορεί να προκύψει μία αρχική έκδοση του προγράμματος (Θυμίζει λίγο επαυξητικό μοντέλο αλλά δεν είναι απόλυτα επαυξητικό, παρότι δανειστήκαμε αρκετά στοιχεία από τα επαυξητικά μοντέλα.)



*Επαυξητικό Μοντέλο*

## Προσωπικές διαφοροποιήσεις μοντέλου SCRUM

Μέσα σε κάθε Sprint του SCRUM μοντέλου μας, θα δανειστούμε μερικά στοιχεία του επαυξητικού μοντέλου. Μέσα σε κάθε sprint πελάτες και προγραμματιστές θα δουλεύουν μεταξύ τους σε ζευγάρια (Προγραμματιστής-Πελάτης, Προγραμματιστής-Πελάτης). Με αυτόν τον τρόπο επιτυγχάνουμε μια πιο εξειδικευμένη παραλληλοποίηση της εργασίας. Κάθε ζευγάρι θα αναλάβει ένα κομμάτι (front-end, back-end with database) για να φτιάξει. Ο πελάτης θα βγάλει απαιτήσεις που αφορούν μόνο το κομμάτι που επέλεξε (Back-end with database ή front-end) και καθώς τα αναλύει και βγάζει προδιαγραφές, ο προγραμματιστής του ζευγαριού τα κοιτάει και ξεκινάει την παράλληλη σχεδίαση και υλοποίηση αυτών που βγάζει ο πελάτης. Τέλος καθώς οι προγραμματιστές ολοκληρώνουν ένα κομμάτι υλοποίησης, αναλαμβάνει ο ελεγκτής να το ελέγξει και να ειδοποιήσει για πιθανά λάθη και προβλήματα. Επομένως σε αυτό το σχέδιο έχουμε μια πολύ στενή συνεργασία μεταξύ πελατών και προγραμματιστών στην αρχή του sprint και στενή συνεργασία μεταξύ προγραμματιστών και ελεγκτή από την μέση μέχρι και το τέλος του sprint με αποτέλεσμα να δουλεύουν όλοι μαζί και συνέχεια. Πιστεύω πως με αυτόν τον τρόπο πετυχαίνουμε μια 80% τουλάχιστον αποτελεσματικότητα στην χρήση του διαθέσιμου προσωπικού μας.



*Μοντέλο SCRUM*

Αυτή ήταν μόνο μία από τις τροποποιήσεις που κάναμε. Για να μην χάνουμε περισσότερο χρόνο με καθημερινές συναντήσεις, οι συναντήσεις για κάθε Sprint αντί να γίνονται καθημερινά, θα γίνονται ανά 1 εβδομάδα. Στο μεταξύ κάθε μέλος ανά 2-3 ημέρες θα σημειώνει στον φάκελο του τα πρακτικά του daily SCRUM απαντώντας στις 3 ερωτήσεις (Τι έκανε τις προηγούμενες μέρες, Τι θα κάνει τις επόμενες, και τέλος τι προβλήματα αντιμετώπισε) ώστε να έχουμε μια συνολική εικόνα του τι συμβαίνει μεταξύ των εβδομάδων και να μπορούμε να βοηθάμε όλοι όπου μπορούμε για να αποφύγουμε καθυστερήσεις. Στο τέλος κάθε Sprint θα δανειζόμαστε ένα χρονικό πλαίσιο για να γίνουν τα επίσημα χαρτιά αποδοχής λειτουργικότητας προϊόντος από τους πελάτες. Μόλις ολοκληρωθεί αυτή η διαδικασία θα ξεκινάει το επόμενο Sprint. Στην συγκεκριμένη περίπτωση ο Project Manager θα αναλάβει και τον ρόλο του SCRUM master για να έχουμε καλύτερη οργανωτική διοίκηση. Λόγω της ολοκληρωμένης μορφής του project μας θα διαφοροποιήσουμε το μοντέλο SCRUM και στο αρχικό του πλαίσιο. Αντί να περιμένουμε «ιστορίες» του Product Owner, χωρίσαμε το λογισμικό σε υποτμήματα τα οποία θα υλοποιηθούν ξεχωριστά μέσα σε κάθε Sprint και θα συνενώνονται με το υπόλοιπο λογισμικό στο τέλος κάθε Sprint. Έτσι πετυχαίνουμε την ίδια λειτουργικότητα του μοντέλου SCRUM με τα δεδομένα που έχουμε.

## Στελέχωση

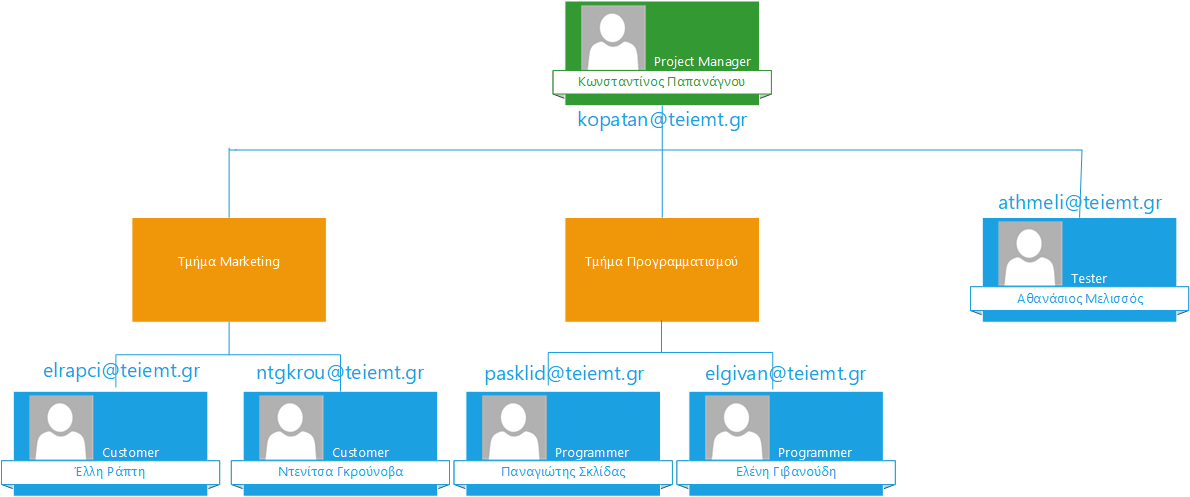
Η στελέχωση της ομάδας θα είναι ως εξής:

1 Project Manager – Κωνσταντίνος Παπανάγνου

2 Πελάτες – Έλλη Ράπτη & Ντενίτσα Γκρούνοβα

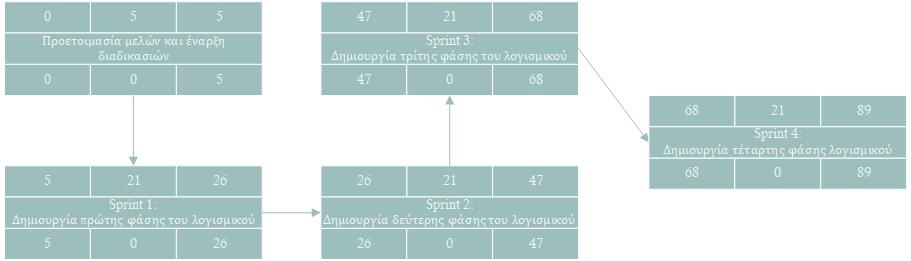
2 Προγραμματιστές – Παναγιώτης Σκλίδας & Ελένη Γιβανούδη

1 Ελεγκτής – Αθανάσιος Μελισσός

****

## Αρχικά Διαγράμματα

### Διάγραμμα PERT



Το έργο το χωρίζουμε σε 5 υποέργα τα οποία θα τα αναλύσουμε αμέσως. Πρώτα όμως να δούμε πως χωρίζεται το έργο σε υποέργο και γιατί. Καταρχάς, χωρίζουμε το πρόγραμμα σε λειτουργικότητες οι οποίες μπορούν να δουλέψουν ανεξάρτητα. Έτσι, μετά από κάθε Sprint μπορούμε να έχουμε ένα λειτουργικό κομμάτι το οποίο είναι έτοιμο να δουλέψει. Το πρώτο κομμάτι είναι η οθόνη σύνδεσης. Μόλις δημιουργηθεί αυτή θα μπορεί κάθε χρήστης να συνδέεται στην υπόλοιπη εφαρμογή. Ένα άλλο κομμάτι είναι η διαχείριση του αποθέματος. Ένα άλλο κομμάτι είναι η διαχείριση παραγγελιών και τέλος έχουμε την διαχείριση τιμολογίων παραγγελιών.

Τ1 Προετοιμασία μελών και έναρξη διαδικασιών. Σε αυτή τη φάση ξεκινάμε την επιλογή ρόλων από όλα τα μέλη, ανάθεση εργασιών σε κάθε μέλος και σχεδιασμό και οργάνωση του έργου. Αυτή η διαδικασία θα διαρκέσει 5 μέρες.

Τ2 Sprint 1: Δημιουργία πρώτης φάσης του λογισμικού. Κατά αυτήν την φάση θα σχεδιαστεί, θα υλοποιηθεί και θα ελεγχτεί η οθόνη σύνδεσης και η λειτουργικότητα της. Θα φτιαχτούν αρχικά μοντέλα διεπαφής χρήστη για παρουσίαση στους πελάτες και θα υπάρξει αναμονή έγκρισης. Στο τέλος του Sprint θα υπάρξει ένα meeting 45 λεπτών για παρουσίαση στους πελάτες το τελικό αποτέλεσμα του τρέχοντος Sprint για αποδοχή λειτουργικότητας. Αυτή η διαδικασία θα διαρκέσει 21 μέρες

Τ3 Sprint 2: Δημιουργία δεύτερης φάσης του λογισμικού. Εδώ θα σχεδιαστεί, θα υλοποιηθεί και θα ελεγχτεί η λειτουργικότητα της διαχείρισης του αποθέματος φαρμάκων. Θα φτιαχτούν αρχικά μοντέλα διεπαφής χρήστη για παρουσίαση στους πελάτες και θα υπάρξει αναμονή έγκρισης. Στο τέλος του Sprint θα υπάρξει ένα meeting 45 λεπτών για παρουσίαση στους πελάτες το τελικό αποτέλεσμα του τρέχοντος Sprint για αποδοχή λειτουργικότητας. Αφού περάσει τον έλεγχο προαπαιτούμενων θα συνενωθεί με το αποτέλεσμα του προηγούμενου Sprint. Αυτή η διαδικασία θα διαρκέσει 21 μέρες

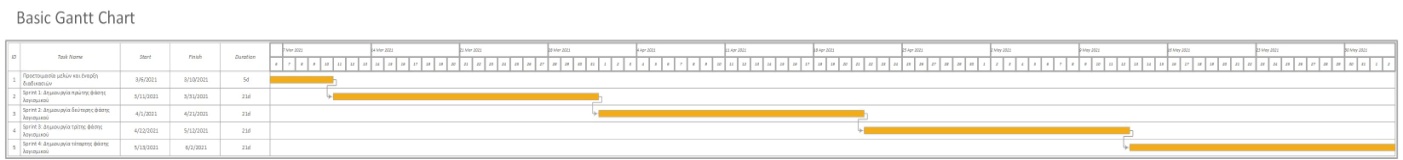
Τ4 Sprint 3: Δημιουργία τρίτης φάσης του λογισμικού. Εδώ θα σχεδιαστεί, θα υλοποιηθεί και θα ελεγχτεί η λειτουργικότητα της διαχείρισης παραγγελιών φαρμακοποιών. Θα φτιαχτούν αρχικά μοντέλα διεπαφής χρήστη για παρουσίαση στους πελάτες και θα υπάρξει αναμονή έγκρισης. Στο τέλος του Sprint θα υπάρξει ένα meeting 45 λεπτών για παρουσίαση στους πελάτες το τελικό αποτέλεσμα του τρέχοντος Sprint για αποδοχή λειτουργικότητας. Αφού περάσει τον έλεγχο προαπαιτούμενων θα συνενωθεί με το αποτέλεσμα του προηγούμενου Sprint. Αυτή η διαδικασία θα διαρκέσει 21 μέρες

Τ5 Sprint 4: Δημιουργία τέταρτης φάσης του λογισμικού. Εδώ θα σχεδιαστεί, θα υλοποιηθεί και θα ελεγχτεί η λειτουργικότητα της διαχείρισης τιμολογίων παραγγελιών. Θα φτιαχτούν αρχικά μοντέλα διεπαφής χρήστη για παρουσίαση στους πελάτες και θα υπάρξει αναμονή έγκρισης. Στο τέλος του Sprint θα υπάρξει ένα meeting 45 λεπτών για παρουσίαση στους πελάτες το τελικό αποτέλεσμα του τρέχοντος Sprint για αποδοχή λειτουργικότητας. Αφού περάσει τον έλεγχο προαπαιτούμενων θα συνενωθεί με το αποτέλεσμα του προηγούμενου Sprint. Αυτή η διαδικασία θα διαρκέσει 21 μέρες

Λόγω του μοντέλου SCRUM που επιλέξαμε και με τον τρόπο που χωρίσαμε τα υποέργα προκύπτει ένα γραμμικό διάγραμμα PERT με μόνο ένα μονοπάτι, το κρίσιμο μονοπάτι.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Υποέργο | Διάρκεια (Μέρες) | Εξάρτηση | ECT | LCT | Αδράνεια |
| Τ1 | 5 | - | 5 | 5 | 0 |
| Τ2 | 21 | Τ1 | 26 | 26 | 0 |
| Τ3 | 21 | Τ2 | 47 | 47 | 0 |
| Τ4 | 21 | Τ3 | 68 | 68 | 0 |
| Τ5 | 21 | Τ4 | 89 | 89 | 0 |

### Διάγραμμα Gantt

****

Το παραπάνω διάγραμμα Gantt προκύπτει από το διάγραμμα Pert και δεν διαφέρει καθόλου. Μας δείχνει πως θα προχωρήσει το project μας ανά μέρα. Στην πορεία θα ακολουθήσει ένα διάγραμμα PERT και ένα διάγραμμα Gantt για το πώς είναι σχεδιασμένα τα Sprints (Από τον SCRUM master).

## Αναθέσεις Προσωπικού

Το προσωπικό θα ξεκινήσει να λειτουργεί μετά την ολοκλήρωση του πρώτου υποέργου. Τα υπόλοιπα υποέργα είναι Sprints. Μέσα σε κάθε Sprint θα δουλεύει όλη η ομάδα με πλήρη συντονισμό για την ευκολότερη διεκπεραίωση του project.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Μέλος/Υποέργο | Τ1 | Τ2 | Τ3 | Τ4 | Τ5 |
| Κωνσταντίνος |  |  |  |  |  |
| Παναγιώτης |  |  |  |  |  |
| Αθανάσιος |  |  |  |  |  |
| Ντενίτσα |  |  |  |  |  |
| Ελένη |  |  |  |  |  |
| Έλλη |  |  |  |  |  |

## Μοντέλο Κύκλου Ζωής

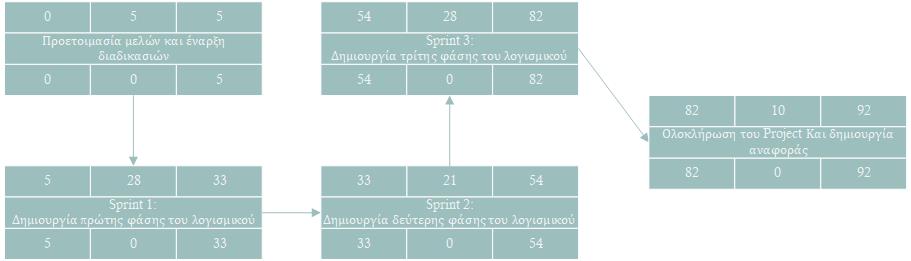
Παρότι αναφέραμε το μοντέλο το οποίο επιλέξαμε και παραθέσαμε τις προσωπικές μας διαφοροποιήσεις, θα αναλύσουμε τον τρόπο λειτουργίας του μοντέλου SCRUM στην αρχική του μορφή. Το μοντέλο SCRUM χαρακτηρίζεται ως ευέλικτο λόγω της φιλοσοφίας του. Είναι ένα μοντέλο στο οποίο η ομάδα είναι αυτή η οποία θα πρέπει να αποφασίσει πως θα λυθεί το πρόβλημα που τους ανατίθεται σε κάθε κύκλο (Sprint). Υπάρχουν 2 κύριοι ρόλοι. Ο SCRUM master ο οποίος έχει τον ρόλο του «προπονητή» της ομάδας με σκοπό την εμψύχωση και βοήθεια της ομάδας όπου χρειάζεται, και του Product Owner. Ο Product Owner φέρνει στο τραπέζι διάφορες «ιστορίες» οι οποίες περιγράφουν τι θέλει να κάνει το λογισμικό. Αυτή η διαδικασία δεν γίνεται μόνο μία φορά, αλλά πολλές φορές. Λόγω του ότι αυτή η διαδικασία καθυστερεί και το λογισμικό πρέπει να ξεκινήσει την στιγμή την οποία έχει κάτι να υλοποιήσει, η υλοποίηση του λογισμικού χωρίζεται σε κύκλους που ονομάζονται Sprints. Τα Sprints διαρκούν 2-4 εβδομάδες. Στην αρχή κάθε Sprint γίνεται μία συνάντηση όπου λαμβάνουν μέρος όλα τα μέλη της ομάδας, ο SCRUM master και ο Product Owner. Σε αυτή τη συζήτηση ο Product Owner δηλώνει τις «ιστορίες» στην ομάδα και σε συνεργασία με τον SCRUM master αποφασίζουν το χρονικό περιθώριο που απαιτείται για να υλοποιηθούν. Μόλις ολοκληρωθεί αυτή η συνάντηση ξεκινάει η διαδικασία από την ομάδα. Πρώτα πρέπει να εξαχθούν απαιτήσεις λογισμικού από τις «ιστορίες» που παρουσίασε ο Product Owner, στην πορεία να εξαχθούν προδιαγραφές, να γίνουν τα απαραίτητα έγγραφα και να ξεκινήσει η σχεδιαστική πορεία του συγκεκριμένου Sprint. Μόλις υλοποιηθεί και ελεγχτεί ο τελικός κώδικας, θα συνενωθεί με το υπόλοιπο πρόγραμμα αν υπάρχει, με αποτέλεσμα να έχουμε μία «νέα» λειτουργική έκδοση του λογισμικού. Να σημειωθεί πως κατά την διάρκεια του Sprint, τα μέλη της ομάδας πρέπει να συμμετέχουν στο daily scrum, το οποίο είναι καθημερινές συναντήσεις που δεν ξεπερνάνε τα 15 λεπτά με σκοπό την συζήτηση μεταξύ μελών της ομάδας στην πρόοδο που έχουν κάνει. Κατά το daily scrum κάθε μέλος της ομάδας πρέπει να απαντήσει 3 ερωτήσεις. 1) Τι έκανε χτες, 2) Τι θα κάνει σήμερα, 3) Που κολλήσανε. Στο τέλος κάθε Sprint συγκαλείται πάλι μία συνάντηση με όλα τα μέλη της ομάδας, τον SCRUM master, και τον Product Owner με σκοπό την παρουσίαση των νέων χαρακτηριστικών που προστέθηκαν στο λογισμικό.



Μοντέλο SCRUΜ

## Τελικά Διαγράμματα:

### Διάγραμμα PERT:



Τ1 Προετοιμασία μελών και έναρξη διαδικασιών. Σε αυτή τη φάση ξεκινάμε την επιλογή ρόλων από όλα τα μέλη, ανάθεση εργασιών σε κάθε μέλος και σχεδιασμό και οργάνωση του έργου. Αυτή η διαδικασία θα διήρκησε 5 μέρες.

Τ2 Sprint 1: Δημιουργία πρώτης φάσης του λογισμικού. Κατά αυτήν την φάση θα σχεδιαστεί, θα υλοποιηθεί και θα ελεγχτεί η οθόνη σύνδεσης και η λειτουργικότητα της. Θα φτιαχτούν αρχικά μοντέλα διεπαφής χρήστη για παρουσίαση στους πελάτες και θα υπάρξει αναμονή έγκρισης. Στο τέλος του Sprint θα υπάρξει ένα meeting 45 λεπτών για παρουσίαση στους πελάτες το τελικό αποτέλεσμα του τρέχοντος Sprint για αποδοχή λειτουργικότητας. Αυτή η διαδικασία διήρκησε 28 μέρες

Τ3 Sprint 2: Δημιουργία δεύτερης φάσης του λογισμικού. Εδώ θα σχεδιαστεί, θα υλοποιηθεί και θα ελεγχτεί η λειτουργικότητα της διαχείρισης του αποθέματος φαρμάκων. Θα φτιαχτούν αρχικά μοντέλα διεπαφής χρήστη για παρουσίαση στους πελάτες και θα υπάρξει αναμονή έγκρισης. Στο τέλος του Sprint θα υπάρξει ένα meeting 45 λεπτών για παρουσίαση στους πελάτες το τελικό αποτέλεσμα του τρέχοντος Sprint για αποδοχή λειτουργικότητας. Αφού περάσει τον έλεγχο προαπαιτούμενων θα συνενωθεί με το αποτέλεσμα του προηγούμενου Sprint. Αυτή η διαδικασία διήρκησε 21 μέρες

Τ4 Sprint 3: Δημιουργία τρίτης φάσης του λογισμικού. Εδώ θα σχεδιαστεί, θα υλοποιηθεί και θα ελεγχτεί η λειτουργικότητα της διαχείρισης παραγγελιών φαρμακοποιών και η λειτουργικότητα διαχείρισης τιμολογίων παραγγελιών. Θα φτιαχτούν αρχικά μοντέλα διεπαφής χρήστη για παρουσίαση στους πελάτες και θα υπάρξει αναμονή έγκρισης. Στο τέλος του Sprint θα υπάρξει ένα meeting 45 λεπτών για παρουσίαση στους πελάτες το τελικό αποτέλεσμα του τρέχοντος Sprint για αποδοχή λειτουργικότητας. Αφού περάσει τον έλεγχο προαπαιτούμενων θα συνενωθεί με το αποτέλεσμα του προηγούμενου Sprint. Αυτή η διαδικασία διήρκησε 28 μέρες

Τ5 Τέλος Έργου για 3 μήνες. Σε αυτή τη φάση όλα τα μέλη της ομάδας συνεργάζονται για να ολοκληρώσουν όλες τις απαραίτητες διαδικασίες ώστε να ολοκληρωθούν όλα τα απαραίτητα έγγραφα και να βγει το report του έργου καθώς και γίνονται διορθώσεις.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Υποέργο | Διάρκεια (Μέρες) | Εξάρτηση | ECT | LCT | Αδράνεια |
| Τ1 | 5 | - | 5 | 5 | 0 |
| Τ2 | 28 | Τ1 | 33 | 33 | 0 |
| Τ3 | 21 | Τ2 | 54 | 54 | 0 |
| Τ4 | 28 | Τ3 | 82 | 82 | 0 |
| Τ5 | 10 | Τ4 | 92 | 92 | 0 |

### Διάγραμμα Gantt

****

(Τα διαγράμματα μπορούν να βρεθούν και στον φάκελο Docs/Charts/ProjectManager σαν εικόνες)

## Εκτίμηση Κόστους

Κύριο μέλημα μας για το λογισμικό που θέλουμε να παραδώσουμε είναι η ποιότητα του και η αποδοτικότητα του. Θέλουμε να βασιστούμε στην ευχρηστία του λογισμικού αλλά και την ασφάλεια του, όπως και την αξιοπιστία του. Με βάση αυτά τα χαρακτηριστικά που θέλουμε να δώσουμε έμφαση θα αναλύσουμε το κόστος το οποίο έχουμε υπολογίσει.

Οι μήνες που εκτιμούνται για την παράδοση του έργου είναι 5 αλλά ο πελάτης το θέλει σε 3. Λόγω του μειωμένου χρόνου θα απαιτηθούν τουλάχιστον 5 εργάτες για την διεκπεραίωση του έργου στον χρόνο που εκτιμάται. Με βάση αυτό η ομάδα θα δημιουργηθεί ως εξής: 2 προγραμματιστές, 2 πελάτες, 1 ελεγκτής. Θα προσπαθήσουμε σε αυτούς τους 3 μήνες να ολοκληρωθεί όσο το δυνατόν μεγαλύτερο κομμάτι του λογισμικού όσο γίνεται με τον καλύτερο δυνατό τρόπο. Το λογισμικό θα είναι scalable αλλά και maintainable για τα features που θα προστεθούν μετά τους 3 μήνες.

Για την καλύτερη δημιουργία του λογισμικού και για να πετύχουμε την μέγιστη αξιοπιστία του όλα τα μέλη της ομάδας που θα δημιουργηθεί θα δουλεύουν παράλληλα και μαζί. Αυτό σημαίνει ότι η δουλειά θα μοιραστεί ισάξια και κάθε μέλος θα συνεισφέρει όσο το δυνατόν το ίδιο με κάθε άλλο. Επομένως και οι ανθρωπομήνα των εργατών θα είναι ίδιες για όλους και θα διαφοροποιηθεί μόνο η αξία που υπολογίζεται στον κάθε ρόλο.

Με βάση αυτό θα πραγματοποιηθεί η αρχική εκτίμηση του κόστους ανά ρόλο και η αρχική εκτίμηση κόστους για όλο το έργο

Προγραμματιστής: 1100€ τον ανθρωπομήνα

Πελάτης: 1000€ τον ανθρωπομήνα

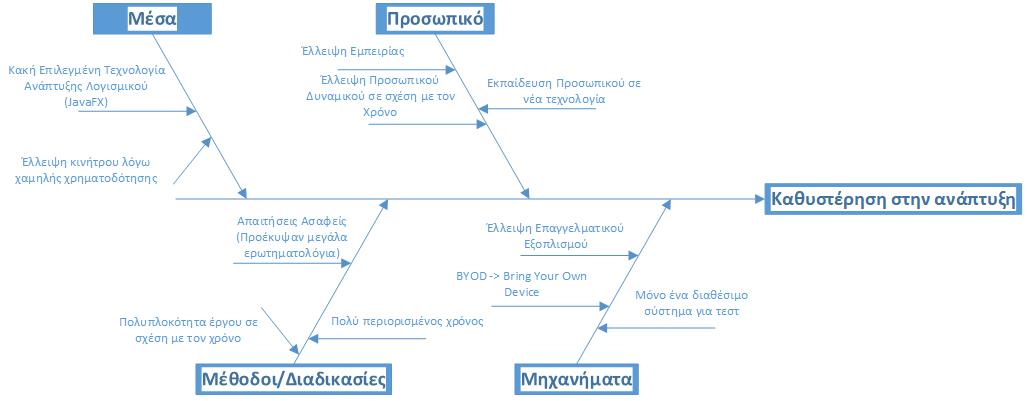
(Με τον όρο πελάτης εδώ αναφέρομαι στα άτομα τα οποία επικοινωνούν με τον κανονικό πελάτη, εξάγουν πληροφορίες σχετικά με το έργο και επίσης βγάζουν τις προδιαγραφές.)

Ελεγκτής: 1200€ τον ανθρωπομήνα

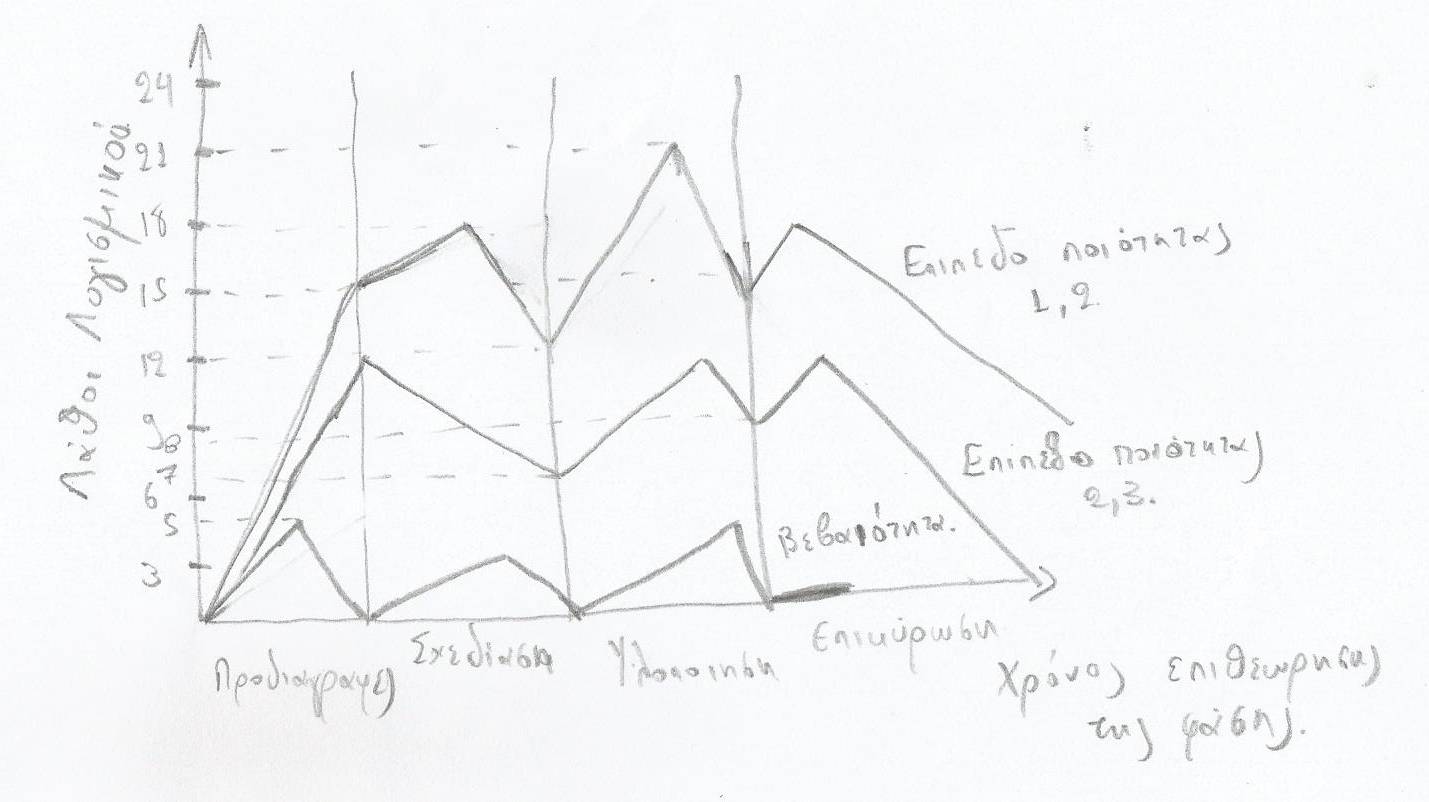
Οι προγραμματιστές είναι 2 άρα 2200€ για τους προγραμματιστές, αντίστοιχα 2000€ για τους πελάτες. Άρα έχουμε 5200€ για το προσωπικό τον ανθρωπομήνα με γνώμονα τον κόπο για το κάθε λειτουργικό μέρος.

Το συνολικό κόστος του έργου για τους 3 μήνες εκτιμάται κοντά στα 30,000€ .

## Αναδρομική Ανάλυση Έργου



Διάγραμμα Ψαροκόκκαλο

Διάγραμμα Επιπέδων Ποιότητας

# Last Part

# Test πρώτου sprint.

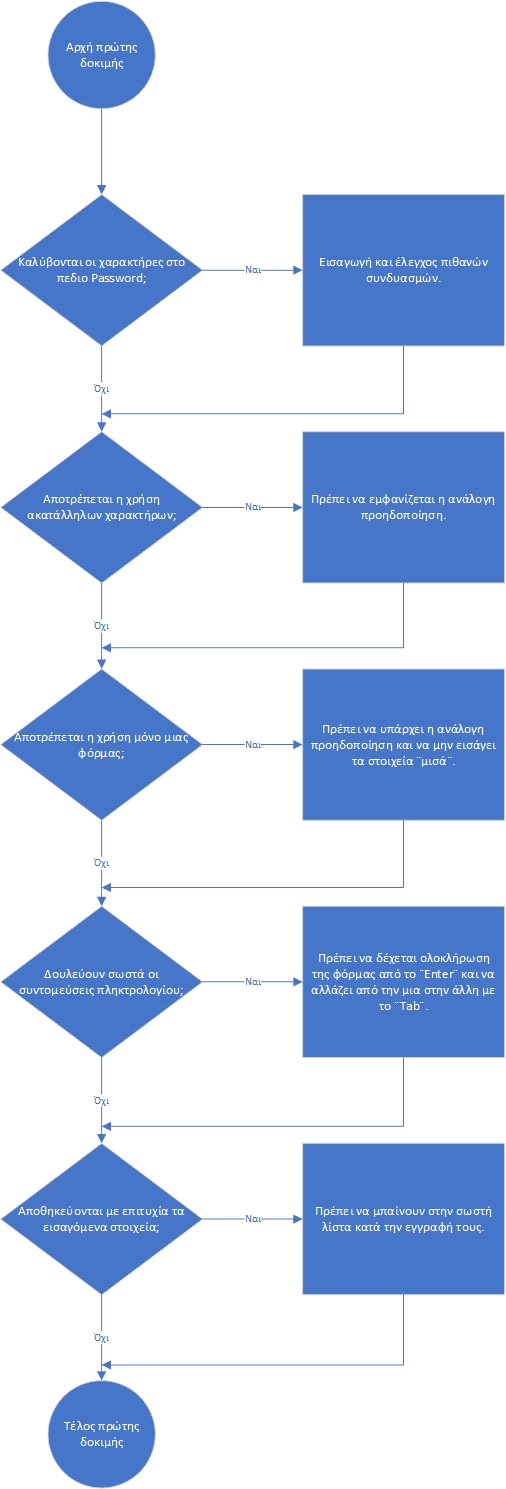
## Πρώτο μέρος test λειτουργικότητας εφαρμογής

Το πρώτο μέρος της εργασίας που θα μου δοθεί για έλεγχο λειτουργικότητας και ανίχνευση προβλημάτων θα είναι το log-in screen (οθόνη εισόδου χρήστη) και έχει ονομαστεί από τους προγραμματιστές ως ¨SSRI¨.

Σε αυτή την οθόνη ο χρήστης θα εισάγει τα στοιχεία του για να μπορεί να αποκτήσει πρόσβαση στην εφαρμογή μέσα από μία φόρμα η οποία θα του ζητάει τα εξής στοιχεία: Username και Password, όπου username θα είναι η φόρμα στην οποία θα εισάγεται το όνομα χρήστη και password η φόρμα η οποία θα συμπληρώνεται με τον κωδικό αυτού του χρήστη.

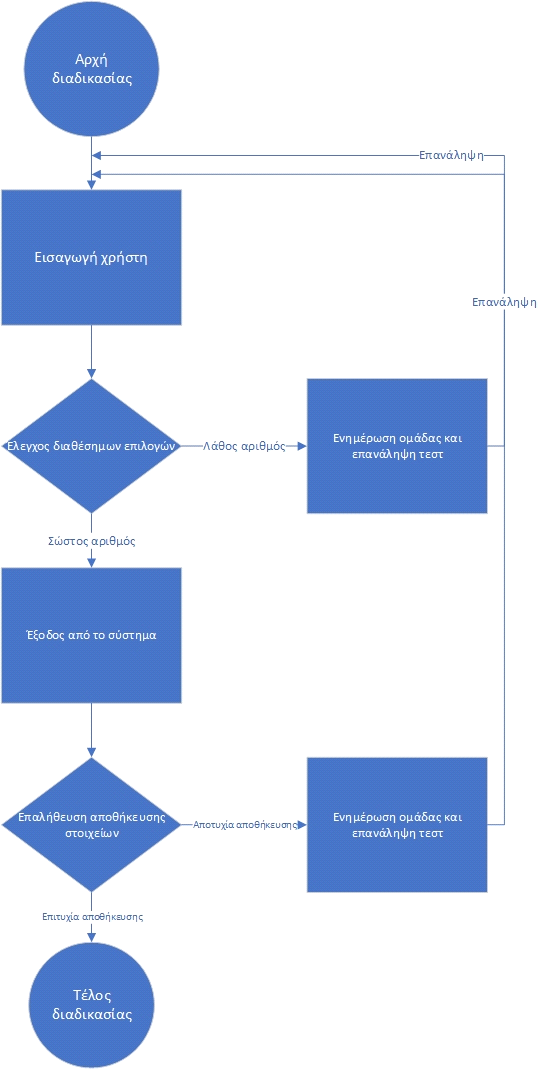
Οι περιπτώσεις τεστ θα ακολουθήσουν ως εξής:

* Έλεγχος της φόρμας password εάν τηρεί τις βασικές αρχές προστασίας του χρήστη κρύβοντας τον κωδικό με κάποιον τρόπο.
* Τι χαρακτήρες θεωρούνται αποδεκτοί από το σύστημα; Σε περίπτωση ακατάλληλου input από τον χρήστη υπάρχει τρόπος να τον ενημερώσει η να τον προτρέψει;
* Έλεγχος της οθόνης όσο αναφορά άδεια πεδία. Ο χρήστης πρέπει να προτρέπεται από το να χρησιμοποιεί μόνο μία φόρμα.
* Έλεγχος λειτουργικότητας κουμπιών και λειτουργίες πληκτρολογίου όπως η αλλαγή φόρμας με το πλήκτρο tab και αποδοχή εισόδου μέσω του enter.
* Έλεγχος αποθήκευσης των στοιχείων που εισάγονται από τον χρήστη, έτσι ώστε να καταγράφονται από το σύστημα.
* Έλεγχος αποθήκευσης των προαναφερόμενων στοιχείων στις σωστές λίστες. (Τα ονόματα στο username, τους κωδικούς στα password κ.λπ.)
* Έλεγχος ευχρηστίας. (Π.χ. εάν το παράθυρο είναι αρκετά ορατό, κατάλληλου μεγέθους και οι φόρμες δεν πέφτουν πολύ κοντά μεταξύ τους μπερδεύοντας τον χρήστη ή κάνοντας την εμπειρία δυσάρεστη).

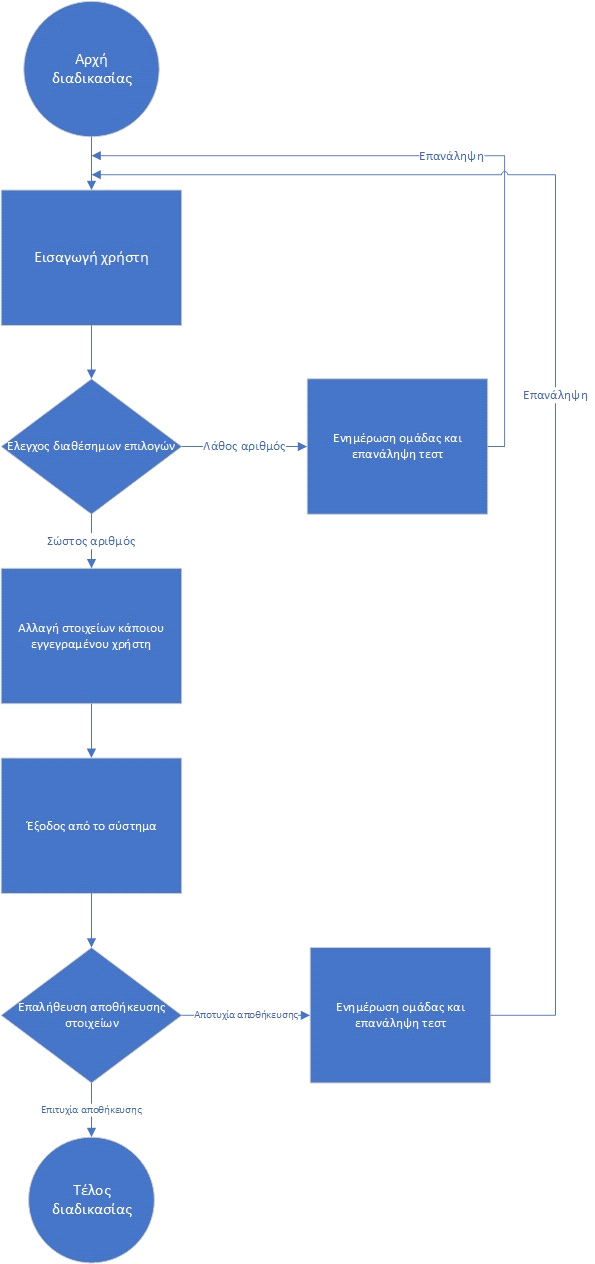
Πρώτο διάγραμμα γενικού πλάνου τεστ (Log-in screen):

Κάθε ένας χρήστης του συστήματος θα έχει τα δικά του επίπεδα πρόσβασης ανάλογα τον ρόλο του. Για να γίνει σίγουρο πως αυτό δουλεύει θα εισάγω δύο χρήστες στο σύστημα μέσω της βάσης δεδομένων, θα τους δοθεί διαφορετικός ρόλος στον καθένα και θα ελεγχθούνε αν οι δυνατότητες που δίνονται στους χρήστες αναλογούν στον τίτλο τους. Εάν το πρόγραμμα εμφανίσει τις σωστές οθόνες ,ανάλογα με τον εισαγόμενο χρήστη, αποθηκεύοντας αλλαγές και προσθήκες, η αποτρέποντάς τον ανάλογα, τότε το τεστ θα θεωρείται επιτυχία.

### Διάγραμμα τεστ εισαγωγής απλού χρήστη:

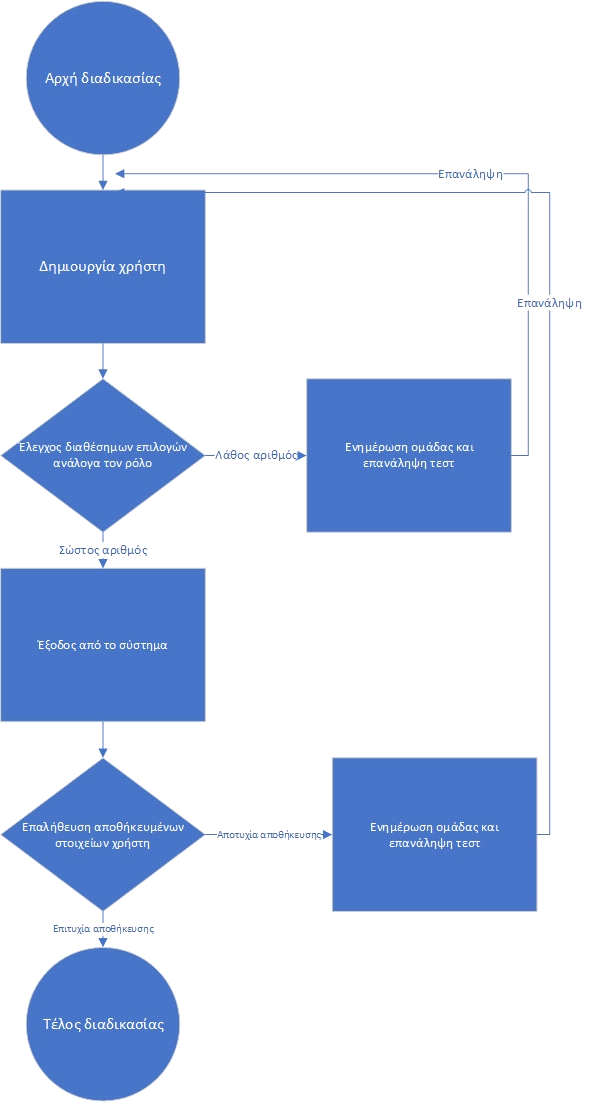


### Διάγραμμα τεστ εισαγωγής admin:

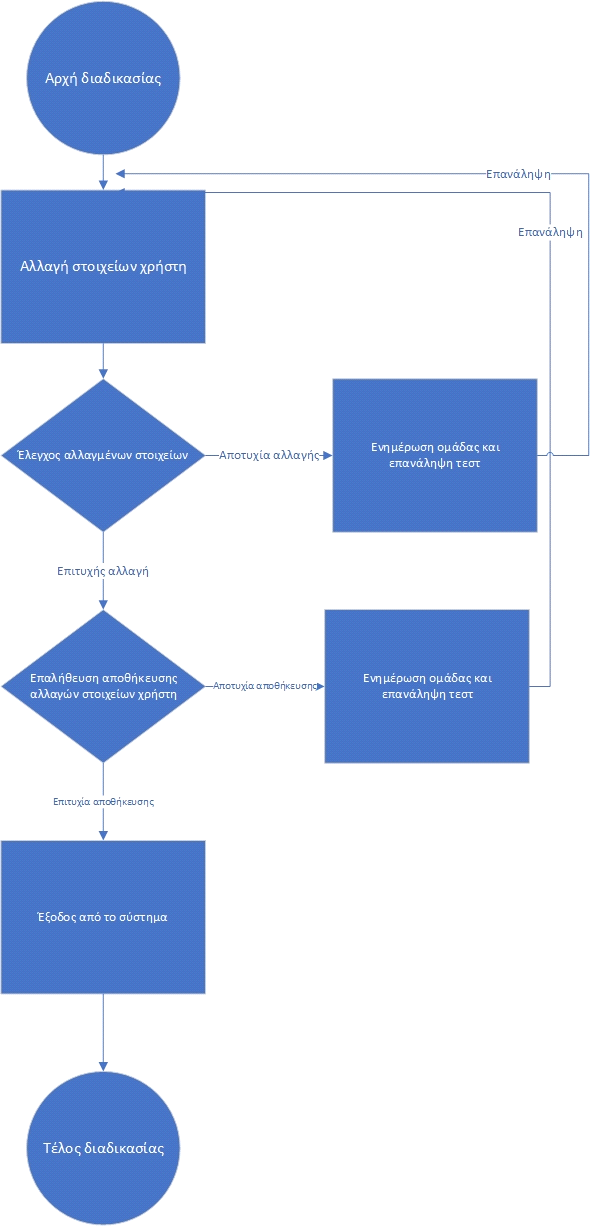


Ο διαχειριστής, καθώς και το ίδιο το πρόγραμμα, θα πρέπει να έχει την δυνατότητα απαραίτητων αλλαγών όπως τον εμπλουτισμό στοιχείων, πρόσθεση και διαγραφή αυτών από το σύστημα, καθώς και την ορθή εκτέλεση αυτών. Για παράδειγμα, οι διαγραφές δεν θα πρέπει να αφήνουν κενά μνήμης καθώς, οι προσθήκες θα πρέπει να γίνονται μέσα στα όρια του χρόνου απόκρισης κ.λπ. Το τεστ θα γίνει με τις εισαγωγές που αναφέρονται προηγουμένως. Εάν το πρόγραμμα προσθέσει με επιτυχία τα εισαγόμενα δεδομένα, τα αλλάξει και τέλος τα διαγράφει χωρίς υπολείμματα στοιχείων στην μνήμη, μέσα στα όρια απόκρισης που ορίστηκε και επιστραφεί η οθόνη, με όλες τις αλλαγές οι ποίες έγιναν σε κάθε μια από αυτές τις περιπτώσεις, τότε το τεστ θεωρείται επιτυχία.

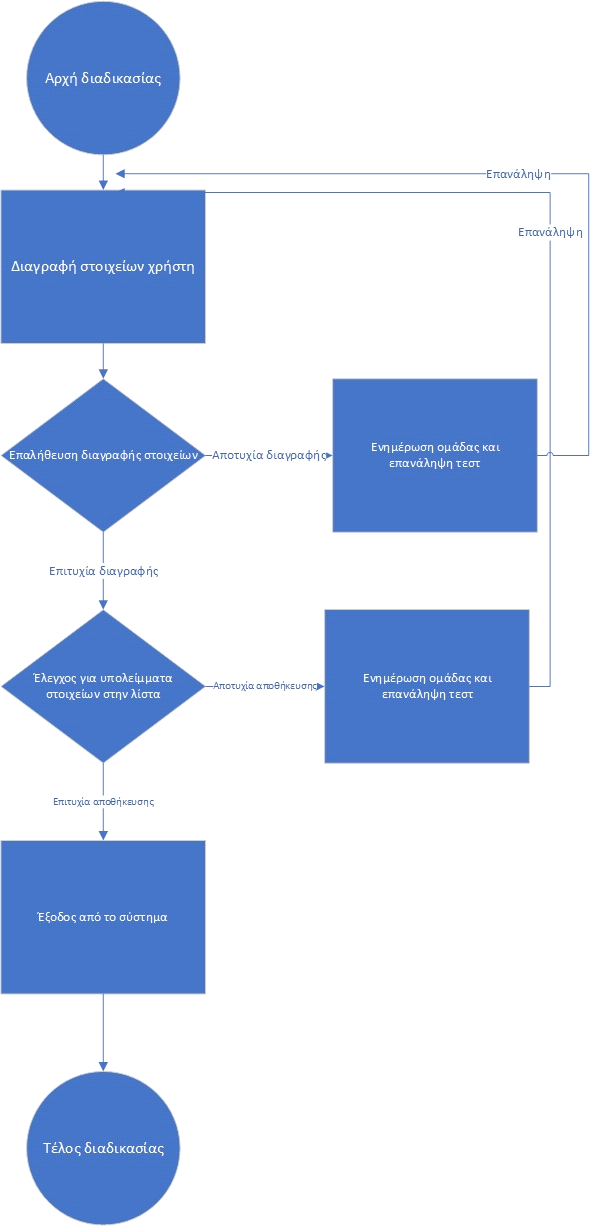
### Διάγραμμα τεστ δημιουργίας ενώς χρήστη:



### Διάγραμμα τεστ αλλαγής στοιχείων χρήστη:



### Διάγραμμα τεστ διαγραφής στοιχείων χρήστη:



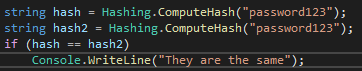
## Περιγραφή ανάλυση και αποτελέσματα Unit Testing του Log-In Screen

Σε αυτό το μέρος του εγγράφου γίνεται η ανάλυση των test που πραγματοποιήθηκαν προκειμένου να εξασφαλιστεί η σωστή λειτουργία της εφαρμογής. Σε αυτή περιέχονται οι μέθοδοι, ο αριθμός επαναλήψεων και τέλος είσοδοι και έξοδοι αυτών οι οποίοι έπαιξαν καθοριστικό ρόλο στην ημερομηνία κυκλοφορίας της πρώτης έκδοσης της εφαρμογής.

Η μέθοδος που χρησιμοποιήθηκε εμπεριείχε τα εξής test:

* Έλεγχος hash του κωδικού χρήστη.
* Username generator το οποίο καθορίζει το Username ανάλογα με την είσοδο του χρήστη.
* Συνθήκες bool που ελέγχουν την εγκυρότητα των εισόδων του χρήστη.
* Τον τρόπο με τον οποίο ελέγχθηκαν οι bool είσοδοι.

Ξεκινώντας με τον έλεγχο του hash οι είσοδοι δεν ήταν σημαντικοί ως προς το test όσο ήτανε το αποτέλεσμα που προέκυπτε. Προκειμένου αυτό το test να διεξαχθεί με επιτυχία οι όροι είχανε ως εξής: αφού ο tester εισήγαγε τον ίδιο κωδικό και στο hash και στο hash2, όταν έτρεχε ο κώδικας έλεγχε εάν το υπολογισμένο hash είναι ίδιο με το άλλο. Αφότου λοιπόν πραγματοποιούταν αυτός ο έλεγχος, εάν όλα πήγαιναν καλά, η έξοδος του προγράμματος έπρεπε να είναι ¨They are the same¨ επιβεβαιώνοντας ότι όντως το hash είναι ίδιο και ότι το test πραγματοποιήθηκε με επιτυχία. Ακολουθεί screenshot του κώδικα:



Οι είσοδοι που ακολούθησαν ήταν: ¨password123¨,¨kodikos345¨,¨simplepass¨. Οι τρείς κωδικοί χρησιμοποιήθηκαν από δύο φορές ο καθένας, συνολικά 6 test, και υπήρξε 100% επιτυχία.

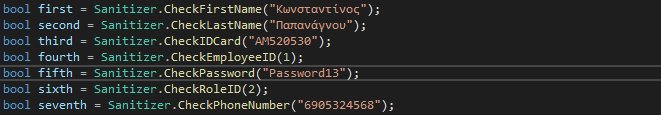
Συνεχίζοντας, το επόμενο test πραγματοποιήθηκε επάνω στο username generator όπου ο γίνεται είσοδος κάποιου ονόματος και επιθέτου, ως αποτέλεσμα λαμβάνεται ένα τυχαία δομημένο Username το οποίο αποτελείται από δύο (2) γράμματα του ονόματος και τέσσερα (4) γράμματα του επιθέτου. Οι είσοδοι και έξοδοι κάτω του screenshot αποτελούνε παραστατικό παράδειγμα. Ακολουθεί screenshot του κώδικα:

username_gen_nakanwparadigmata

|  |
| --- |
| Είσοδοι:   * (ΠΑΝΑΓΙΩΤΗΣ,ΣΚΛΙΔΑΣ) * (ΑΘΑΝΑΣΙΟΣ,ΜΕΛΙΣΣΟΣ) * (ΚΩΝΣΤΑΝΤΙΝΟΣ,ΠΑΠΑΝΑΓΝΟΥ)   Έξοδοι:   * paskli,paklid * a8meli,a8mell * kwpapa,kwpana |

Το test έγινε έξι (6) φορές και είχε απόλυτη επιτυχία. Οι μόνες παρατηρήσεις που έγιναν ήταν ότι λόγο σύνδεσης ελληνικού αλφαβήτου με αγγλικούς χαρακτήρες πολλές φορές το τυχαία παραγόμενο username είναι δυσνόητο από τον χρήστη ιδιαίτερα όταν υπάρχουν ονόματα που συμπεριλαμβάνουν τα γράμματα ¨ξ¨ και ¨θ¨ καθώς τα ανάλογα στο αγγλικό αλφάβητο αντικαταστήθηκαν με το ¨3¨,¨8¨ ανάλογα.

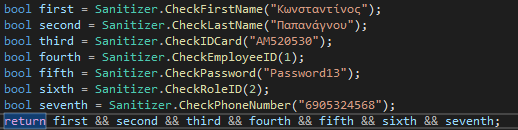
Κλείνοντας το πρώτο test στα τελευταία μέρη του κώδικα έγινε ένας γενικός έλεγχος όλων των πεδίων, αυτά είναι: Όνομα, Επίθετο, Id χρήστη, Επίπεδο πρόσβασης Κωδικός χρήστη, Ρόλος του χρήστη, Τηλέφωνο χρήστη. Με αυτόν τον έλεγχο τα πεδία προς συμπλήρωση ελέγχθηκαν κατάλληλα με διάφορους εισόδους για να εξασφαλισθεί η αξιοπιστία αυτών, έτσι ώστε να αποφευχθεί οποιαδήποτε λάθος χρήση ή είσοδος. Ακολουθεί screenshot του κώδικα:



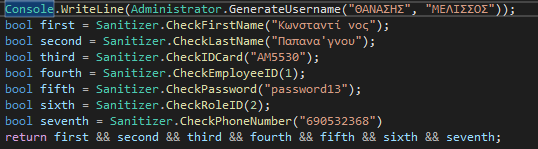
Οι έλεγχοι που ακολούθησαν ήτανε σε μορφή εισόδων και έχουν ως εξής: άδεια ή ελλείπει στοιχεία, λάθος στοιχεία και τυχαία κενά ανάμεσα στα στοιχεία. Ο έλεγχος έγινε με την λογική ανίχνευσης των λαθών για την ομαλή λειτουργία της εφαρμογής σε περίπτωση λάθους εισόδου. Ο έλεγχος έγινε με βάση breakpoints όπου το κάθε ένα από αυτά τα στοιχεία εάν θεωρούτανε αποδεκτά από το πρόγραμμα επέστρεφαν true αλλιώς το πρόγραμμα ενημέρωνε κατάλληλα επιστρέφοντας false. Ο έλεγχος έγινε με την βοήθεια της εξής γραμμής κώδικα:

apotelesmata_EKSIGISI

Σε αυτή, ελέγχονται όλες οι bool καταστάσεις πολύ πιο άμεσα εξυπηρετώντας ως ένα γρήγορο και αξιόπιστο εργαλείο ελέγχου, όπου εάν υπήρχε έστω και ένα λάθος, το breakpoint της γραμμής θα ενημέρωνε με true η false κατά την διάρκεια του debugging προειδοποιώντας ανάλογα. Ακολουθεί screenshot ολόκληρου του κώδικα με μέθοδο boole:



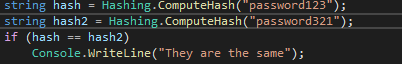
Παραδείγματα από tests στα οποία το πρόγραμμα έπρεπε να ενημερώσει τον ελεγκτή ότι υπάρχει λάθος και χρησιμοποιήθηκαν κατά την διάρκεια των test:



Εδώ το πρόγραμμα έχει ως έξοδο false καθώς υπάρχουνε προβληματικές είσοδοι σε όλα τα πεδία κειμένου, πιο αναλυτικά υπάρχει κενό στο όνομα του χρήστη, λάθος χαρακτήρας το επίθετο, ελλιπείς id κάρτας, αποκλειστική χρήση μη κεφαλαίων στον κωδικό και αριθμό τηλεφώνου που αποτελείται από 9 χαρακτήρες. Επομένως κάνοντας αναδρομή και χρησιμοποιώντας breakpoints για debugging μπορούμε να δούμε ότι:

|  |  |
| --- | --- |
| first | false |
| second | false |
| third | false |
| fourth | true |
| fifth | false |
| sixth | true |
| seventh | false |

Στην περίπτωση hash όταν γίνεται έλεγχος και ο κωδικός διαφέρει του άλλου αλλάζουν τα hash οπότε δεν καλείται ποτέ η γραμμή ¨They are the same¨ καθώς το hash ==hash2 μετατρέπεται σε false. Ακολουθεί παράδειγμα άνισης ισοδυναμίας:



### Πρώτο μέρος test διεπαφής

Αυτό είναι το test της οθόνης log-in. Σε αυτό θα δοκιμασθεί η απλότητα χρήσης και η λειτουργικότητα αυτής. Τα test που ακολούθησαν αναφέρονται αυστηρά και μόνο στην οθόνη εισόδου χρήστη. Ακολουθεί διαδικασία εξέτασης λειτουργιών στην πρώτη έκδοση της εφαρμογή με όνομα ¨SSRI¨:

Οι λειτουργίες πληκτρολογίου βοηθάνε στην μέγιστη ταχύτητα χρήσης της εφαρμογής τονίζοντας την ευχρηστία αυτής. Αυτές περιλαμβάνουν:

* Λειτουργία αλλαγής φόρμας με το πλήκτρο tab
* Επιλογή του περιεχομένου της φόρμας με την αλλαγή φόρμας μέσω tab
* Κύκλος μεταξύ των επιλογών μέσω tab
* Εισαγωγή των στοιχείων μέσω του Enter

Οι προαναφερόμενες λειτουργίες δοκιμάστηκαν από δέκα (10) φορές και πάνω για την μέγιστη εγκαθίδρυση της αξιοπιστίας.

Οι λειτουργία αλλαγής φόρμας δουλεύει άψογα, δίνοντας στον χρήστη την δυνατότητα γρήγορης εναλλαγής μεταξύ των δύο. Η λειτουργία δούλεψε 100% των test.

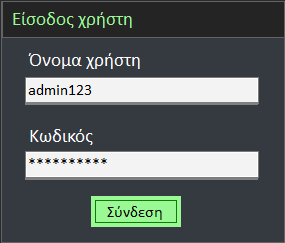


*(Απεικόνιση χρήσης tab)*

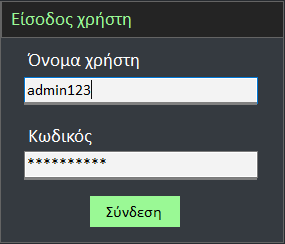
Έχοντας επιλέξει την πρώτη φόρμα του ονόματος χρήστη, με την χρήση του πλήκτρου tab, μπορούμε να πλοηγηθούμε στην αμέσως επόμενη πατώντας το μία φορά, με την δεύτερη να επιλέξουμε το πλήκτρο εισόδου ¨Σύνδεση¨ και τέλος με την τρίτη να κάνουμε έναν πλήρη ¨κύκλο¨ γυρνώντας πίσω στην αρχική φόρμα εισόδου ονόματος. Η λειτουργία δούλεψε 100% των test αλλά αξίζει να σημειωθεί ότι όταν γίνεται επιλογή του κουμπιού δεν υπάρχει αρκετά έντονο feedback από το outline που το περιβάλλει.



*(Πρώτη χρήση του πλήκτρου tab)*



*(Δεύτερη χρήση του πλήκτρου tab)*



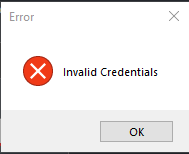
*(Τρίτη χρήση του πλήκτρου tab)*

Έχοντας λοιπόν καλύψει τις βασικές λειτουργίες χρήσεις, ας περάσουμε στις λειτουργίες και χαρακτηριστικά τα οποία ενημερώνουν και αποτρέπουν τον χρήστη σε περίπτωση που κάνει λάθος κατά την χρήση αυτής. Αυτές που ελέγχθηκαν είναι:

* Έλεγχος άδειας φόρμας/ελλιπή στοιχεία.
* Λάθος στοιχεία
* Ευχρηστία οθόνης εισόδου (Εάν η δομή είναι αρκετά κατανοητή)

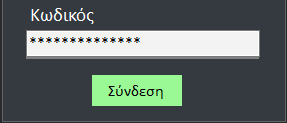
Οι παρακάτω λειτουργίες δοκιμάστηκαν 5 φορές καθώς το αποτέλεσμα ήταν πάντα σταθερά το ίδιο.

Όταν ο χρήστης δεν έχει συμπληρώσει την φόρμα σωστά τότε ενημερώνεται σχετικά με το σφάλμα του με ένα μήνυμα σφάλματος το οποίο αναγράφει ¨Invalid Credentials¨.



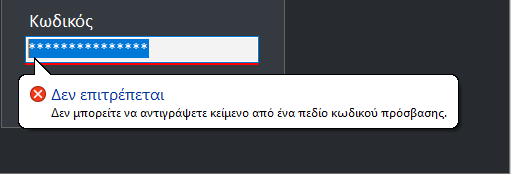
Το ίδιο μήνυμα εμφανίζεται όταν είτε το Password είτε το Username είναι άδεια παρόλο το γεγονός ότι ένα από αυτά έχει συμπληρωθεί. Επιπρόσθετα το Ui είναι κατανοητά δομημένο δίνοντας τον χρήστη 100% σαφήνεια επάνω στις λειτουργίες που του παρουσιάζονται μέσω των τίτλων κάθε φόρμας και μινιμαλιστικού σχεδιασμού.

Λήγοντας το Ui test της πρώτης φάσης ελέγχθηκε ένα από τα ποιο σημαντικά χαρακτηριστικά μίας εφαρμογής, η ασφάλεια. Σε αυτή την φάση των test πραγματοποιήθηκαν 2 είδη τεστ. Το πρώτο να είναι η κάλυψη του κωδικού του χρήστη από κάποιο χαρακτήρα αποτρέποντας την υποκλοπή στοιχείων, π.χ. από κάποιον άλλο συνεργάτη του που εργάζονται στον ίδιο χώρο. Στο δεύτερο test έγινε δοκιμή της υποκλοπής στοιχείων, συγκεκριμένα με το password όπου έγινε επιλογή, αντιγραφή και επικόλληση του κάπου αλλού.



*(Πρώτο test)*

Στο δεύτερο test όταν γίνεται επιλογή και μετά απόπειρα αντιγραφής με Ctrl-V τότε το πρόγραμμα ενημερώνει τον χρήστη ότι δεν επιτρέπεται να κάνει αντιγραφή και επικόλληση του πεδίου και όντως μετά από πολλαπλά test το σύστημα αυτό αποτρέπει τον χρήστη επιτυχώς από το να πράξει μια τέτοια ενέργεια. Το ίδιο ακριβώς γίνεται και με το δεξί κλικ όπου η επιλογή της αντιγραφής δεν είναι επιλέξιμη από τον χρήστη.



*(Δεύτερο test)*

Τέλος πρώτης φάσης test διεπαφής χρήστη (Log-In Screen).

# Test δεύτερου sprint.

### Δεύτερο μέρος test λειτουργικότητας της εφαρμογής (Μενού φαρμάκων)

Προχωρώντας στο δεύτερο μέρος της υλοποίησης της εφαρμογής σειρά έχει το σύστημα αποθήκευσης φαρμάκων και τα χαρακτηριστικά αυτών. Το κάθε φάρμακο θα χαρακτηρίζεται από τους κατασκευαστές του και τον σκοπό χρήσης του, δηλαδή το είδος φαρμάκου, ενώ το σύστημα θα πρέπει, με σωστό τρόπο, να ενημερώνει τον χρήστη με τις κατάλληλες πληροφορίες, οι οποίες και θα ελεγχθούν στην δεύτερη έκδοση της εφαρμογής ονόματι¨Mesulid¨:

* Απόθεμα φαρμάκων
* Κατάταξη φαρμάκων ανάλογα της επιλεγμένης ιδιότητας με αλφαβητική σειρά
* Συλλογή πληροφοριών απόδοσης φαρμάκων (πωλήσεις, κέρδη κ.λ.π.)
* Προειδοποίηση σε περίπτωση κινδύνου (έλλειψη, μη ανανεωμένα στοιχεία όπως το απόθεμα κ.λ.π.)
* Δεδομένα και χαρακτηριστικά κάθε φαρμάκου

Όπως και με το πρώτο μέρος που εξετάστηκε, σε αυτό εφαρμόζουν οι ίδιοι κανόνες πρόσβασης που πάει να πει ότι ανάλογα με την ιδιότητα του χρήστη καθορίζονται και τα στοιχεία τα οποία μπορεί να αλλάξει η να παρακολουθήσει. Επομένως αφότου γίνει ένας επανέλεγχος αυτού του κανόνα, μένει να εξετασθούνε οι οθόνες με τις οποίες θα περιηγείται ο χρήστης, σε αυτό το μέρος της εργασίας αυτές είναι οι οθόνες του αποθηκάριου του υπεύθυνου μάρκετινγκ και του πωλητή φαρμάκων. Σύμφωνα με τις προδιαγραφές και απαιτήσεις οι ιδιότητες που θα εξεταστούν είναι οι εξής:

**Αποθηκάριος:**

* Σωστή απεικόνιση των στοιχείων ανάλογα της επιλεγμένης κατηγορίας
* Απεικόνιση των σωστών αριθμών και χαρακτηριστικών φαρμάκων

**Μάρκετινγκ:**

* Έλεγχος διαθεσιμότητας όλων των τριών (3) οθονών
* Ακριβής απεικόνιση πληροφοριών κέρδους

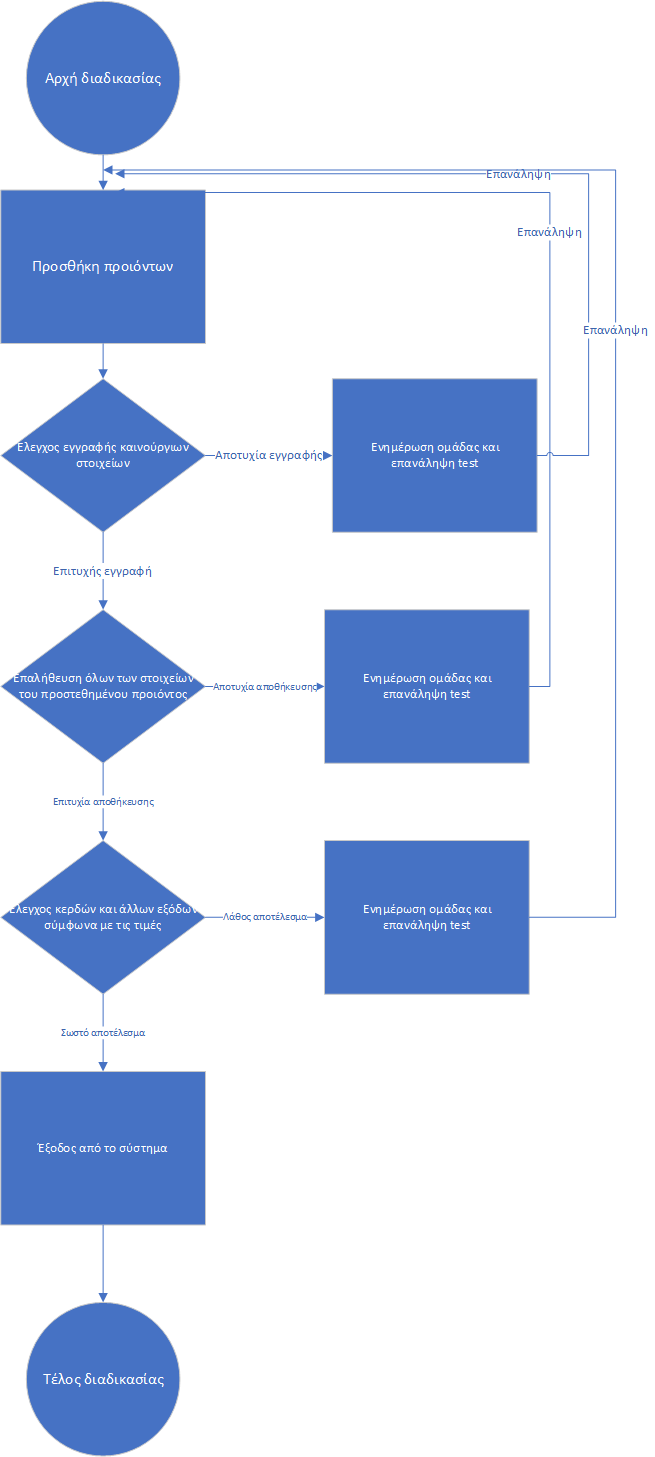
**Πωλητής φαρμάκων:**

* Έλεγχος των διαθέσιμων επεξεργάσιμων πεδίων
* Ακριβής απεικόνιση στοιχείων και άμεση επεξεργασία αυτών.

**Ανάλυση των σχεδίων test δεύτερης φάσης:**

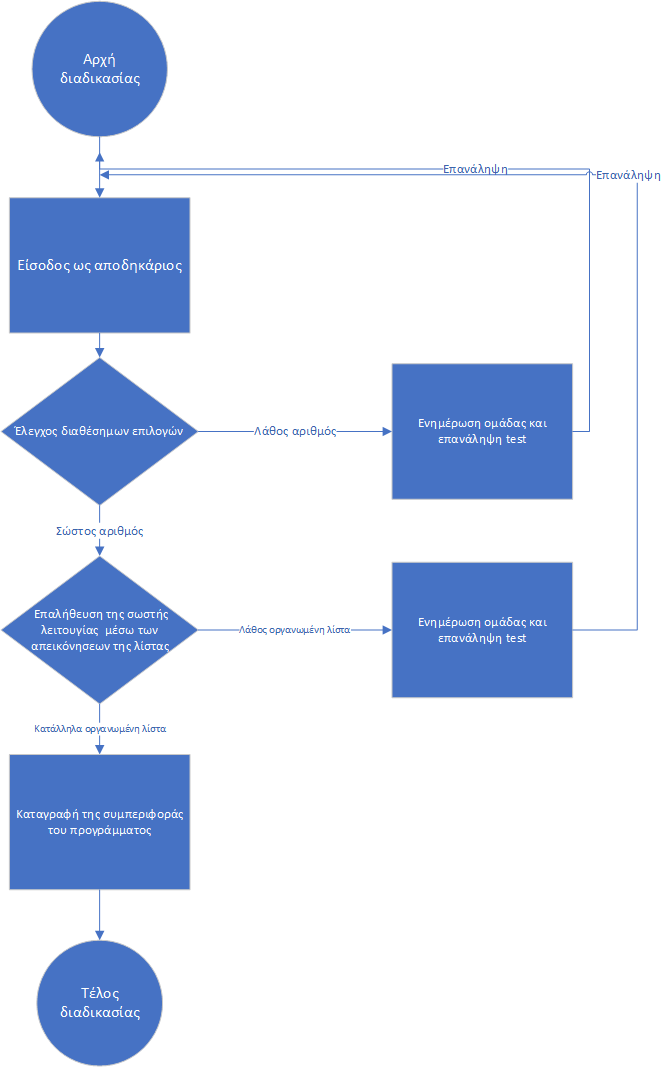
Η απεικόνιση των σωστών νούμερων καθώς και χαρακτηριστικών, αν και σημαντική, δεν εδραιώνει την απόλυτα ορθή λειτουργία της εφαρμογής. Προκειμένου να γίνει έλεγχος εις βάθος επάνω στην αξιοπιστία αυτής θα γίνουν δοκιμές ελέγχου τύπου κλειστού κουτιού με εισόδους-εξόδους των οποίων τα αποτελέσματα θα καταγράφονται με σκοπό την επαλήθευση της, σωστής η μη , λειτουργίας της εφαρμογής.

### Διάγραμμα test ελέγχου εισόδων εξόδων (2ου μέρους):

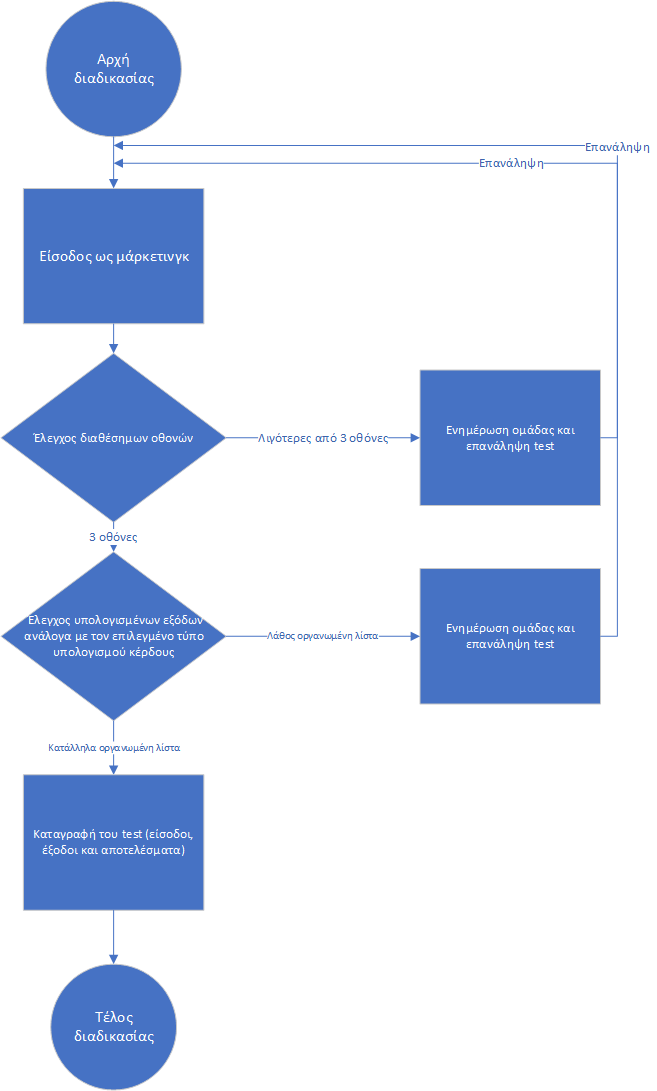


Επιπλέων θα δοκιμαστούν οι ιδιότητες που δίνονται στον χρήστη ανάλογα με τον ρόλο του. Στην περίπτωση του αποθηκάριου η λίστα των προϊόντων θα πρέπει να οργανώνονται κατάλληλα για την εύκολη περιήγηση και έλεγχο αυτής. Επιπλέων στην περίπτωση του μάρκετινγκ θα πρέπει να γίνεται ο κατάλληλος υπολογισμός των πληροφοριών με την βοήθεια και των τριών διαθέσιμων οθονών οι οποίες πρέπει να είναι διαθέσιμες σε όλες τις περιπτώσεις δοκιμών test.

### Διάγραμμα test ελέγχου αποθηκάριου:



### Διάγραμμα test ελέγχου μάρκετινγκ:



### Διάγραμμα test ελέγχου πωλητή φαρμάκων:

****

Η μοναδική παρατήρηση που έγινε κατά την διάρκεια δοκιμών είναι το όνομα χρήστη ,το οποίο πολλές φορές, αποτελούτανε από λιγότερο από 6 χαρακτήρες, όπως και θα έπρεπε. Το περιστατικό παρατηρήθηκε κατά την διάρκεια δημιουργίας χρηστών για τον κατάλληλο έλεγχο των λειτουργιών που αφορούν τον καθένα. H δυσλειτουργία παρατηρήθηκε κατά το δεύτερο sprint και αναλύθηκε λεπτομερώς στο αρχείο ¨Report 1¨ το οποίο στάλθηκε επιτόπου στους προγραμματιστές για περαιτέρω έρευνα αυτής, όταν αυτό επιλύθηκε το κόστος σε εργατομέρες υπολογίσθηκε σε μια εργατομέρα. Επιπλέων, ο ρόλος του αποθηκάριου ανανεώθηκε κατάλληλα με την δικιά του οθόνη η οποία μπορεί να ειδωθεί στις αναφορές που σχετίζονται με την ευχρηστία του μενού της εφαρμογής. Ολοκληρώνοντας την δεύτερη φάση ελέγχου της εφαρμογής έγινε εμφανές ένα ακόμα πρόβλημα όσο αναφορά την ασφάλεια χρήσης αυτής και αυτό περικλειόταν γύρο από την δημιουργία χρηστών όταν το πεδίο ρόλου παρέμενε ¨null¨, δηλαδή άδειο, έχοντας ως αποτέλεσμα να αναθέτει στον πρόσφατα δημιουργημένο χρήστη τον ρόλο του διαχειριστή.

Το πρόβλημα, το διάγραμμα και η ακολουθία βημάτων ανακάλυψης αυτού αναλύθηκε λεπτομερώς στο αρχείο ¨Report 2¨, η επιδιόρθωση της δυσλειτουργίας κόστισε 2 εργατοώρες. Ολοκληρώνοντας την δεύτερη φάση ελέγχου της εφαρμογής, μια πολύ σημαντική αλλαγή που ωφελεί την ασφάλεια της, άρα και του συστήματος που την περικλείει, είναι η εφαρμογή παύσης συνεχόμενων σφαλμένων προσπαθειών όπου όταν ο χρήστης δώσει λάθος στοιχεία παραπάνω από 3 φορές τότε μπαίνει σε αναμονή 30 δευτερολέπτων προτού ξαναπροσπαθήσει και ενημερώνεται κατάλληλα με ένα παράθυρο σφάλματος το οποίο αναλύθηκε λεπτομερώς στο test της διεπαφής (Ui).

Κατά την διάρκεια των αλλαγών της εφαρμογής ολοκληρώθηκαν κάποιοι έλεγχοι που αφορούσανε τα στοιχεία των χρηστών και συγκεκριμένα την αλλαγή κωδικού από το μενού αλλαγής στοιχείων το οποίο οδηγούσε στην δυσλειτουργία του συστήματος. Έχει γίνει κατάλληλη αναφορά. Η διόρθωση του σφάλματος κόστισε 1 εργατοώρα.

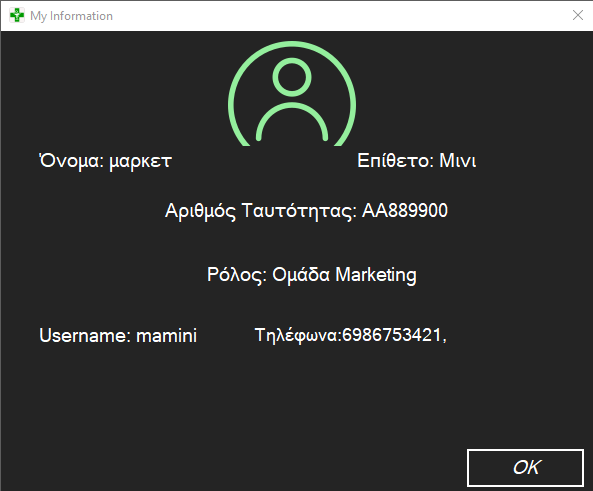
## Δεύτερο μέρος δοκιμών διεπαφής (Μενού φαρμάκων)

Στην δεύτερη φάση ελέγχου της διεπαφής χρήστη η οποία εξετάζει την δεύτερη έκδοση της εφαρμογής ¨Mesulid¨, η οποία επικεντρώνεται στην προσθήκη τριών ρόλων όπου ο καθένας θα έχει διαφορετικές ιδιότητες, δοκιμάζεται η ευχρηστία της κάθε μιας οθόνης ανάλογα με τον επιλεγμένο ρόλο. Οι τρείς ρόλοι είναι: η ομάδα μάρκετινγκ, ο πωλητής φαρμάκων και ο αποθηκάριος. Στον κάθε ρόλο προσφέρεται και ένα διαφορετικό μενού το οποίο βοηθάει στον υπολογισμό ή την αναζήτηση πληροφοριών επάνω στα διαθέσιμα προϊόντα.

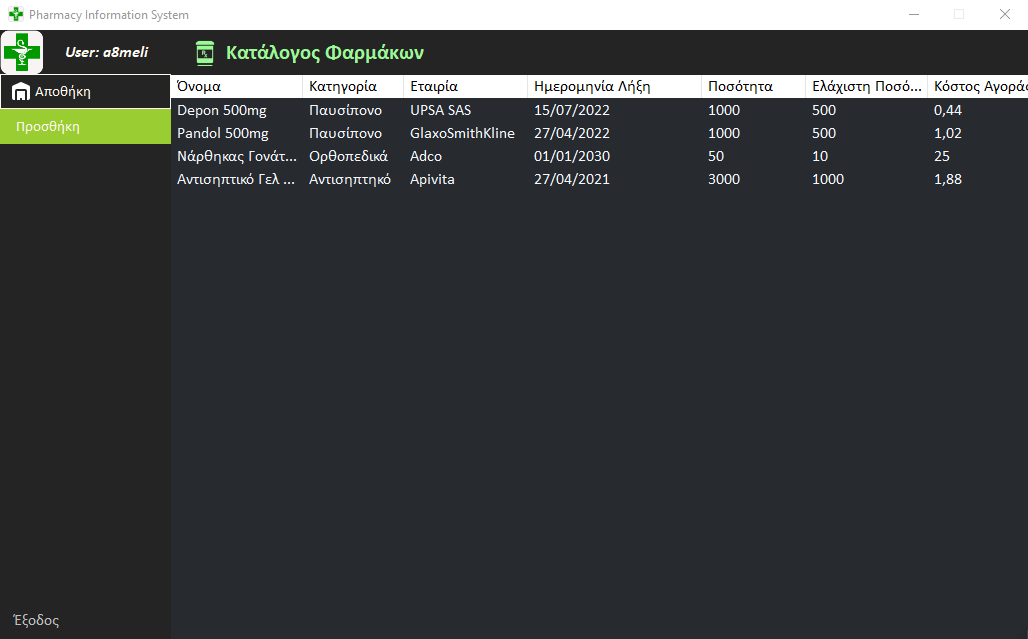
Αρχίζοντας με τα γενικά χαρακτηριστικά που μοιράζονται όλοι οι προαναφερόμενοι ρόλοι στις οθόνες τους, αυτά είναι:

* Κουμπί εξόδου από το σύστημα
* Μενού με της κατάλληλες λειτουργίες ανάλογες του ρόλου
* Κουμπί πληροφοριών χρήστη με τίτλο ¨User: (όνομα user)¨

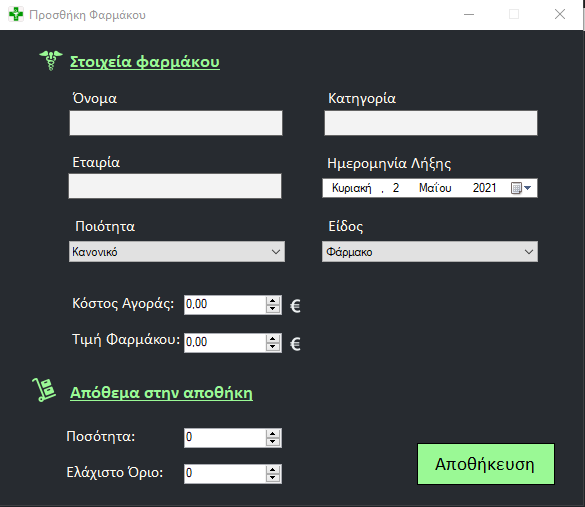
Τα αναφερόμενα χαρακτηριστικά είναι απόλυτα λειτουργικά. Το κουμπί εξόδου δίνει την ικανότητα τον χρήστη να αποσυνδεθεί γυρνώντας τον στην οθόνη εισόδου log-in, οι λειτουργίες προβάλλουν τα κατάλληλα μενού ανάλογα με το ποια έχει επιλεχθεί και τέλος το κουμπί ¨User: (όνομα user)¨ λειτουργεί όπως προοριζόταν προβάλλοντας τις κατάλληλες πληροφορίες ανάλογα με τον χρήστη, τα χαρακτηριστικά και τον ρόλο του στο σύστημα. Ακολουθεί παράδειγμα προβολής πληροφοριών χρήστη:



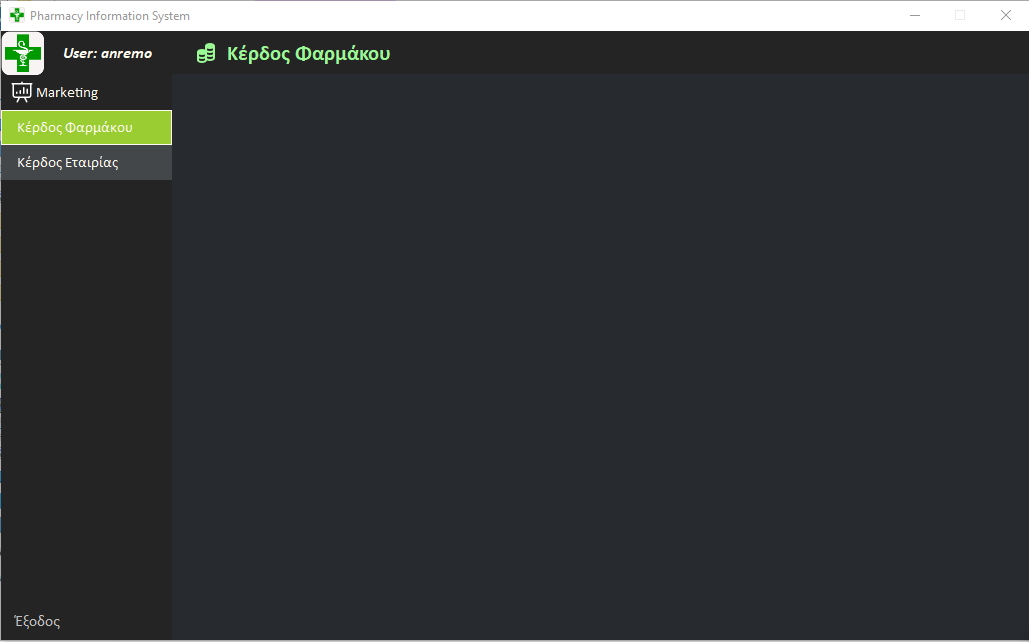
Έχοντας μιλήσει για τα γενικά χαρακτηριστικά και την λειτουργικότητα τους υπάρχουν δύο κύριοι παράγοντες που βοηθάνε τον χρήστη στην διαφοροποίηση των οθονών μεταξύ τους. Αυτοί οι παράγοντες είναι, πρώτων η προβολή του ρόλου πάνω από τις λειτουργίες όπου ο κάθε ένας έχει ένα μοναδικό εικονίδιο και δεύτερων οι ίδιες οι λειτουργίες οι οποίες διαφέρουν ανάλογα με τον συνδεδεμένο χρήστη. Ακολουθούνε οι οθόνες που εξετάστηκαν:



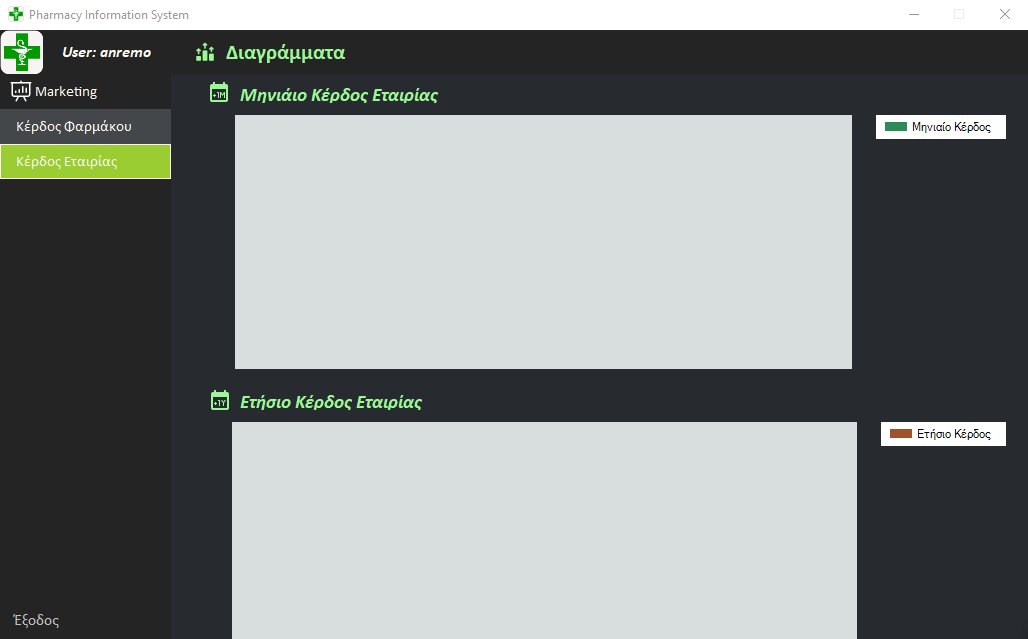
*(Οθόνη αποθηκάριου)*

**

*(Οθόνη αποθηκάριου για προσθήκη φαρμάκων και στοιχείων αυτών)*



*(Οθόνη ομάδας μάρκετινγκ κέρδος φαρμάκων)*

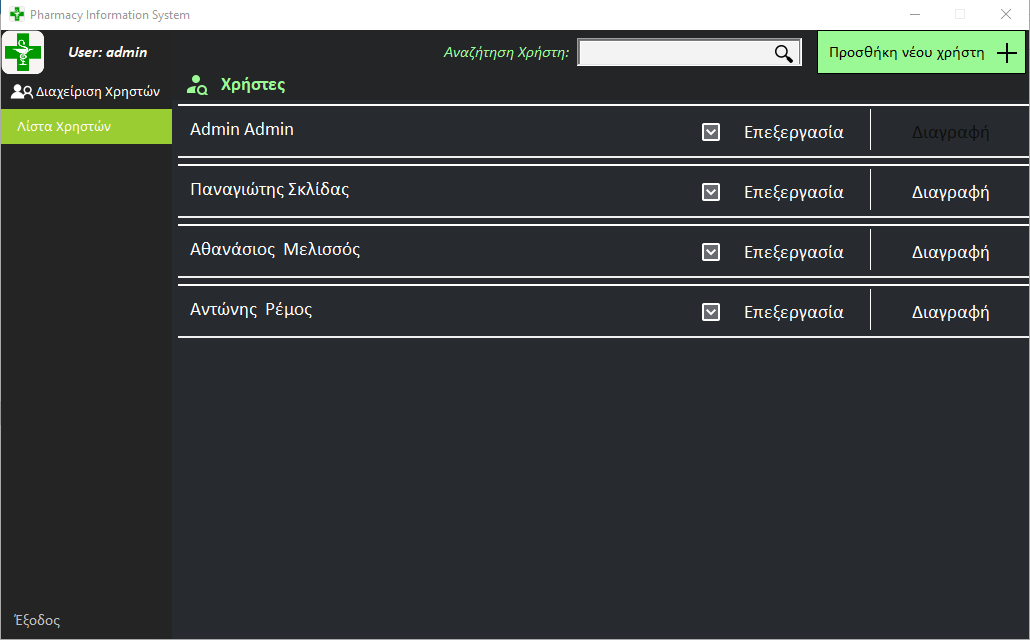
**

*(Οθόνη ομάδας μάρκετινγκ κέρδος εταιρίας)*



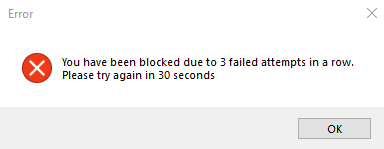
*(Οθόνη πωλητή φαρμάκων)*

Μαζί με τις αλλαγές που έγιναν για την καλύτερη πλοήγηση και κατανόηση της διεπαφής της εφαρμογής έγιναν και αλλαγές που επίλυσαν προβλήματα όπως το πάνω αριστερό κουμπί ¨User:¨ να μην εμπεριέχει το όνομα του χρήστη υπήρξε ανακαίνιση της οθόνης του Admin. Ακολουθεί εικόνα της νέας οθόνης:

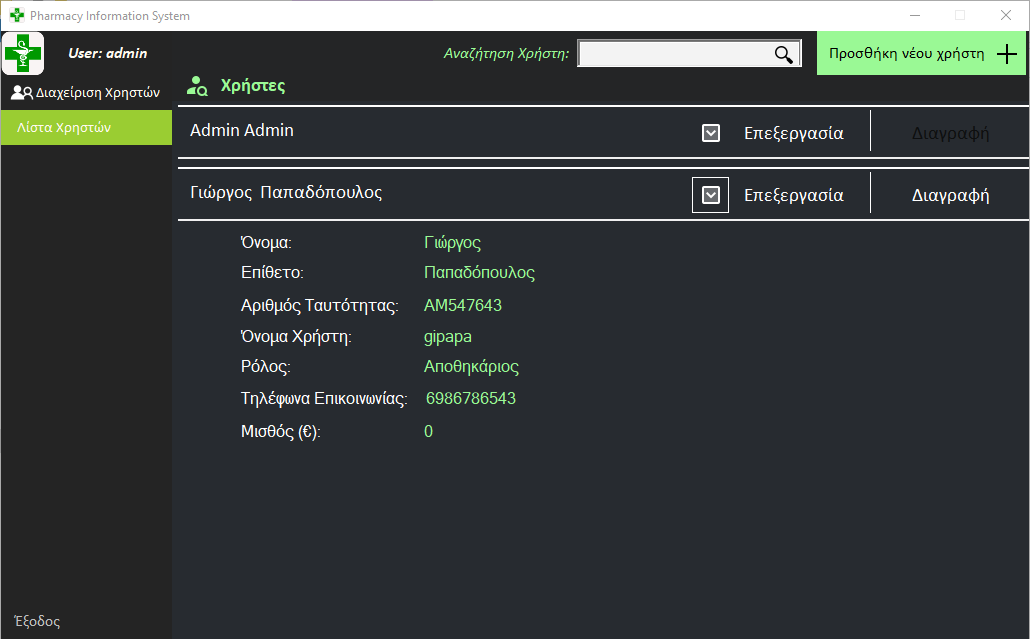


*(Νέα οθόνη admin)*

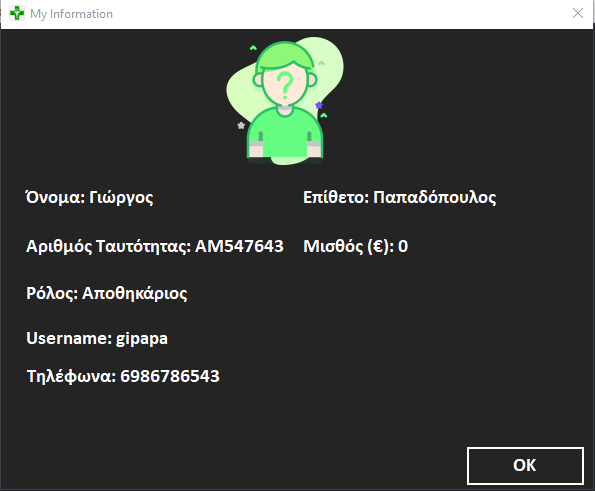
Λήγοντας την σύνοψη αλλαγών και επιπρόσθετων χαρακτηριστικών που περιβάλλουν την δεύτερη εκδοχή της εφαρμογής με όνομα ¨Mesulid¨ είναι αξιόλογη η πρόσθεση της προειδοποίησης πολλών αποτυχημένων προσπαθειών σύνδεσης στο σύστημα ενημερώνοντας τον χρήστη κατάλληλα με την ειδοποίηση που ακολουθεί:



Έπειτα της συνέχειας της ανάπτυξης της εφαρμογής στο δεύτερο sprint, μερικές οθόνες αλλάχτηκαν για την διευκόλυνση του χρήστη και το καλύτερο παρουσιαστικό της εφαρμογής. Πιο συγκεκριμένα, έγινε προσθήκη των μισθών για τον κάθε χρήστη ενημερώνοντας τον, έτσι, για το εισόδημα του, στοιχείο πολύ σημαντικό για την ευχρηστία της εφαρμογής. Ακολουθεί η αναφερόμενη οθόνη:



*(Όψη από μεριάς Admin)*

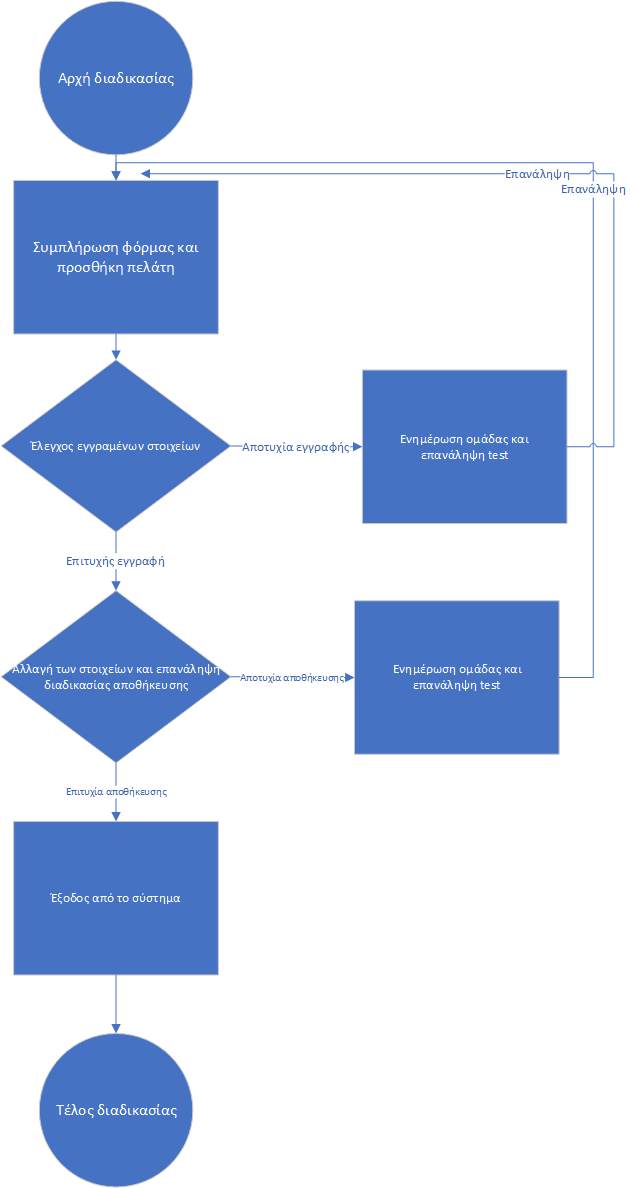
**

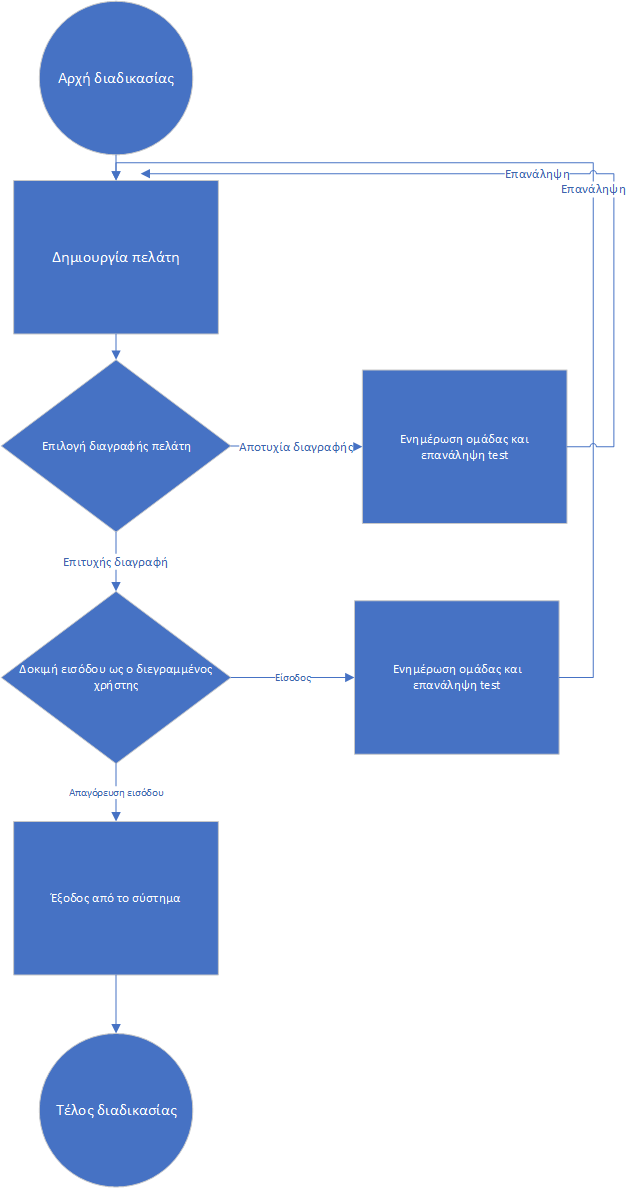
*(Ανανεωμένη οθόνη χρήστη)*

# Test τρίτου Sprint.

### Τρίτο μέρος test λειτουργικότητας της εφαρμογής (Ανανεωμένα μενού)

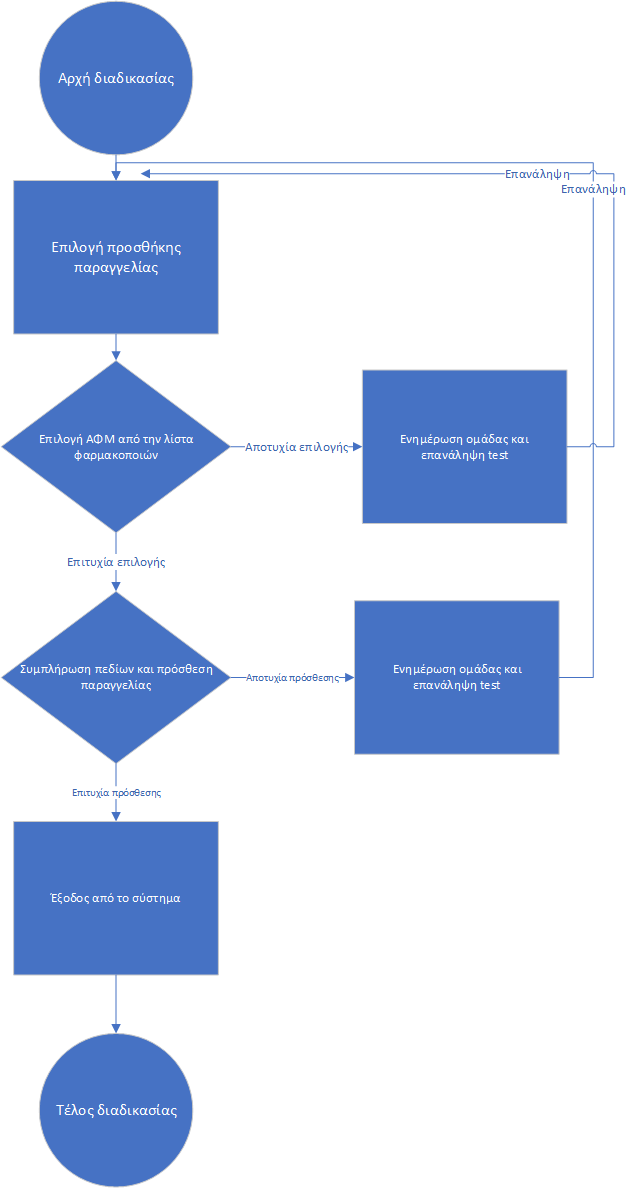
Σε αυτό το μέρος της εφαρμογής προστέθηκαν λειτουργικά μέρη στις οθόνες Μαρκετινγκ και Πωλητών. Τα test που ακολουθούν παρέμειναν πιστά στην ακολουθία βημάτων των διαγραμμάτων με μοναδικές διευκρινήσεις επάνω στις νέες λειτουργίες που προστέθηκαν. Επομένως τα διαγράμματα που ακολουθούν αφορούνε αυστηρά τα καινούργια χαρακτηριστικά των οθονών. Οι καινούργιες λειτουργίες της οθόνης πωλητή έχουνε ως εξής: ικανότητα προσθήκης πελατών, διαγραφή αυτών καθώς και επεξεργασία των στοιχείων τους και η ικανότητα διαχείρισης των παραγγελιών. Ακολουθούνε τα κατάλληλα διαγράμματα διεξαγωγής test των προαναφερόμενων λειτουργιών:

Διάγραμμα ελέγχου λειτουργίας προσθήκης και αλλαγής στοιχείων πελάτη

Διάγραμμα ελέγχου λειτουργίας διαγραφή πελάτη

Ακολουθεί διάγραμμα ελέγχου σχετιζόμενο με την οθόνη παραγγελιών όπου οι λειτουργίες έχουν να κάνουν με την επιλογή φαρμακοποιού, προσθήκη παραγγελιών και αποθήκευση αυτών στο σύστημα.

### Διάγραμμα ελέγχου προσθήκης παραγγελίας μέσω επιλογής φαρμακοποιού



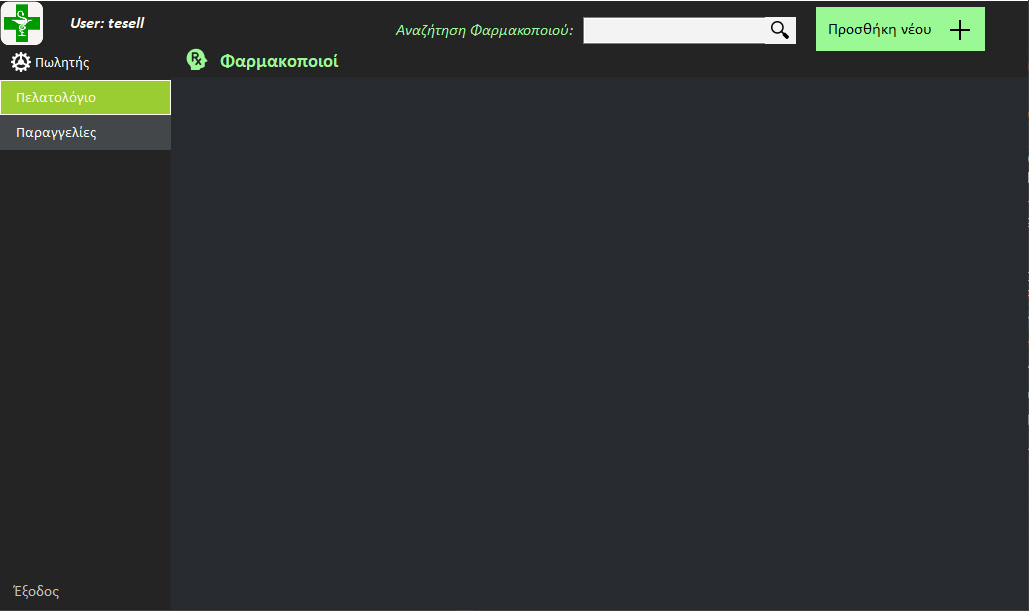
Κατά την διάρκεια των ελέγχων παρατηρήθηκε μια δυσλειτουργία όσο αναφορά το διάγραμμα παραγγελιών και είχε να κάνει με την έλλειψη των κατάλληλων check στον κώδικα, τα οποία οδηγούσαν στο crashing της εφαρμογής. Πιο συγκεκριμένα όταν γινότανε επιλογή ΑΦΜ (φαρμακοποιού) και τα υπόλοιπα πεδία της παραγγελίας ήτανε κενά οδηγούσε σε απότομό κλείσιμο της εφαρμογής, έχει γραφτεί η κατάλληλη αναφορά. Η λύση σε αυτή την προβληματική συμπεριφορά ήταν σχετικά εύκολη χωρίς χρονοβόρες διαδικασίες, με άλλα λόγια το σφάλμα απαλείφτηκε σε 1 εργατοώρα.

## Τρίτο μέρος test διεπαφής (Ανανεωμένα μενού)

Κοντεύοντας στην ολοκλήρωση της εφαρμογής η ομάδα συγκεντρώθηκε στην καλυτέρευση της διεπαφής χρήστη καθώς και την προσθήκη λειτουργικών σημείων που ήρθε με αυτή. Πιο συγκεκριμένα οι έλεγχοι που ακλούθησαν είναι στις ανανεωμένες οθόνες Μάρκετινγκ και Πωλητή όπου προστέθηκαν νέες λειτουργίες για καλύτερη ευχρηστία όπως αναλυτικά διαγράμματα και πληροφορίες παραγγελιών αντίστοιχα.

Οθόνη πωλητή

Στην οθόνη του πωλητή προστέθηκαν οι δύο βασικές λειτουργίες που ζητούνται από τις απαιτήσεις, αυτές είναι η λίστα πελατών όπου ο χρήστης μπορεί να αναζητεί να προσθέτει ή να διαγράφει κάποιον φαρμακοποιό εάν επιθυμεί καθώς και να μπορεί να διαχειριστεί παραγγελίες.

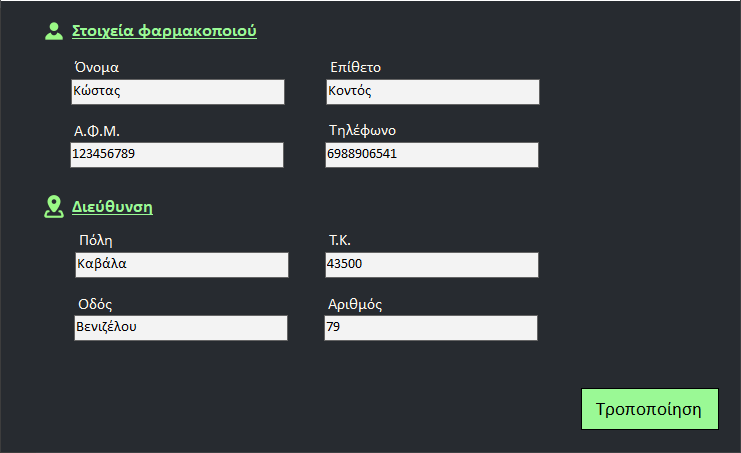


*(Νέα οθόνη πωλητή)*

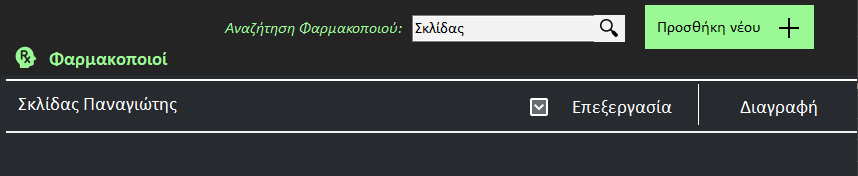
Στο πελατολόγιο εμφανίζονται τα ονόματα των πελατών του πωλητή σε μια λίστα και όπως προαναφέρθηκε, υπάρχουν οι κατάλληλες λειτουργίες για μέγιστη ευχρηστία. Πιο συγκεκριμένα η προσθήκη πελάτη, διαγραφή καθώς και επεξεργασία των στοιχείων τους. Για κάθε λειτουργία ελέγχθηκε εάν υπάρχει η κατάλληλη ανατροφοδότηση προς τον χρήστη ή εάν εμφανίζεται η κατάλληλη φόρμα ανάλογα της ενέργειας που ακολούθησε προηγουμένως. Ακολουθούνε ανάλογα αποσπάσματα:



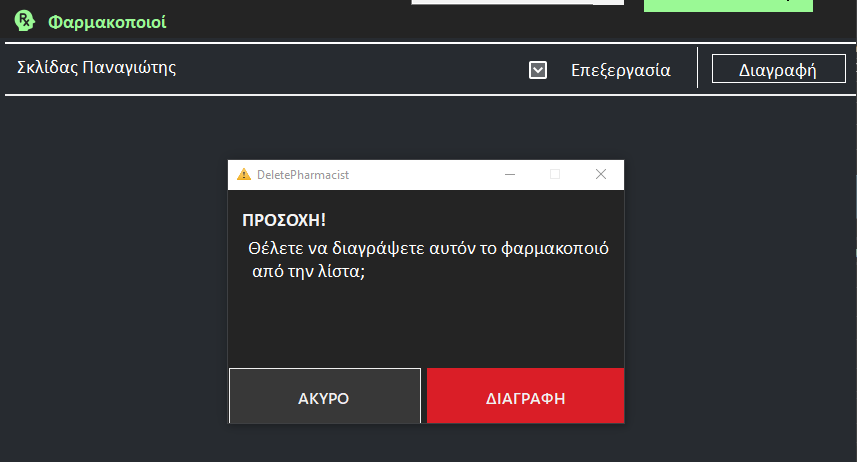
*(Προσθήκη φαρμακοποιού)*

**

*(Αλλαγή κουμπιού αποθήκευσης σε “Τροποποίηση” όταν γίνεται αλλαγή στοιχείων)*

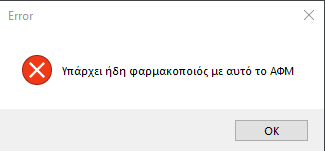


*(Αναζήτηση φαρμακοποιού)*

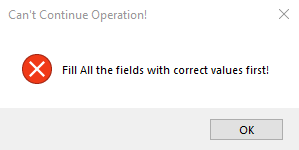


*(Διαγραφή φαρμακοποιού και το ανάλογο μήνυμα)*

Ακολουθούνε μηνύματα συσχετιζόμενα με σφάλματα τα οποία προέκυψαν κατά την είσοδο ακατάλληλων, ελλιπή ή διπλότυπων στοιχείων κατά την διάρκεια έλεγχων της οθόνης πωλητή:

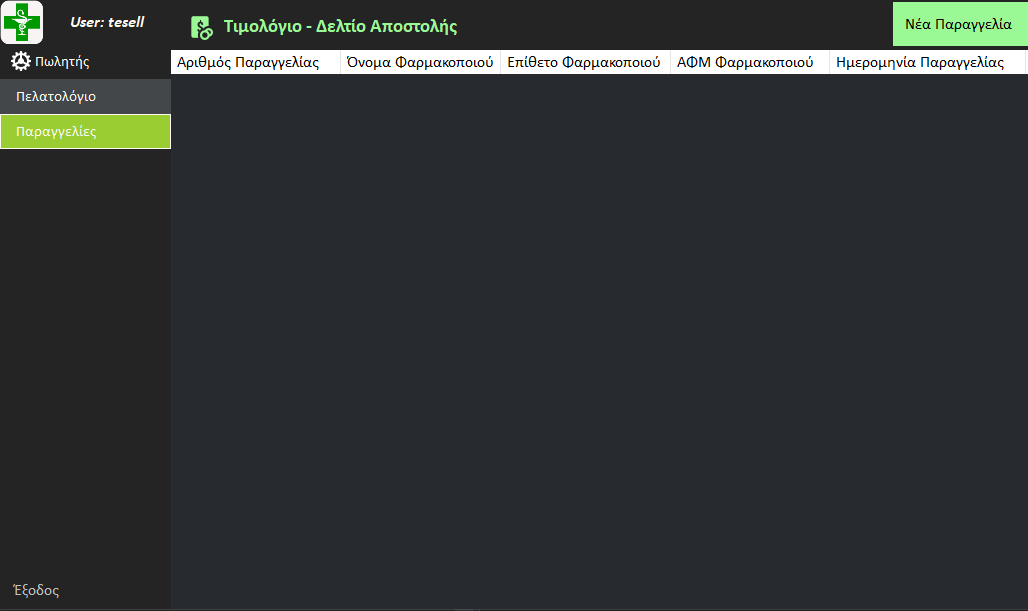


*(Μήνυμα προειδοποίησης διπλότυπου ΑΦΜ)*

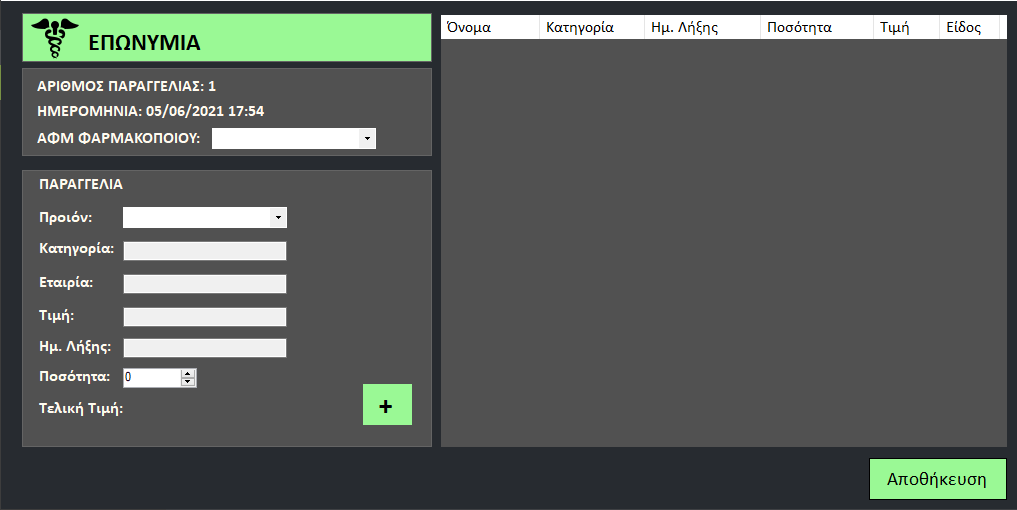
**

*(Μήνυμα προειδοποίησης σφαλμένης/ελλιπής φόρμας)*

Στην οθόνη παραγγελιών υπάρχει μια λίστα με τα στοιχεία της κάθε μίας καθώς και οι κατάλληλες φόρμες συμπλήρωσης οι οποίες δουλεύουνε άψογα. Ακολουθούν κατάλληλα screenshot:

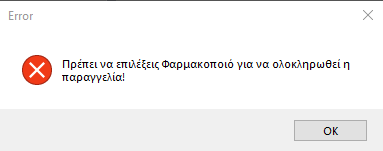


*(Οθόνη παραγγελίας)*

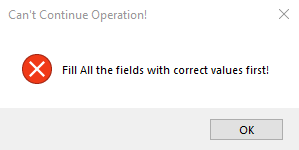
**

*(Οθόνη προσθήκης παραγγελίας)*

Ακολουθούνε μηνύματα σφάλματος:



*(Μήνυμα αποτυχίας επιλογής φαρμακοποιού)*

**

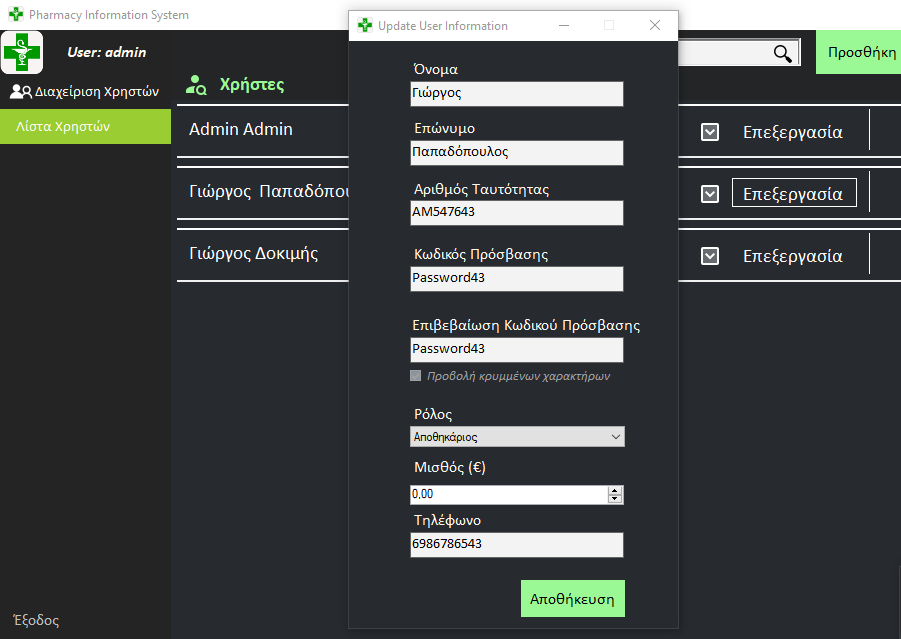
*(Μήνυμα προειδοποίησης σφαλμένης/ελλιπής φόρμας)*

# Αναφορές με χρονική σειρά.

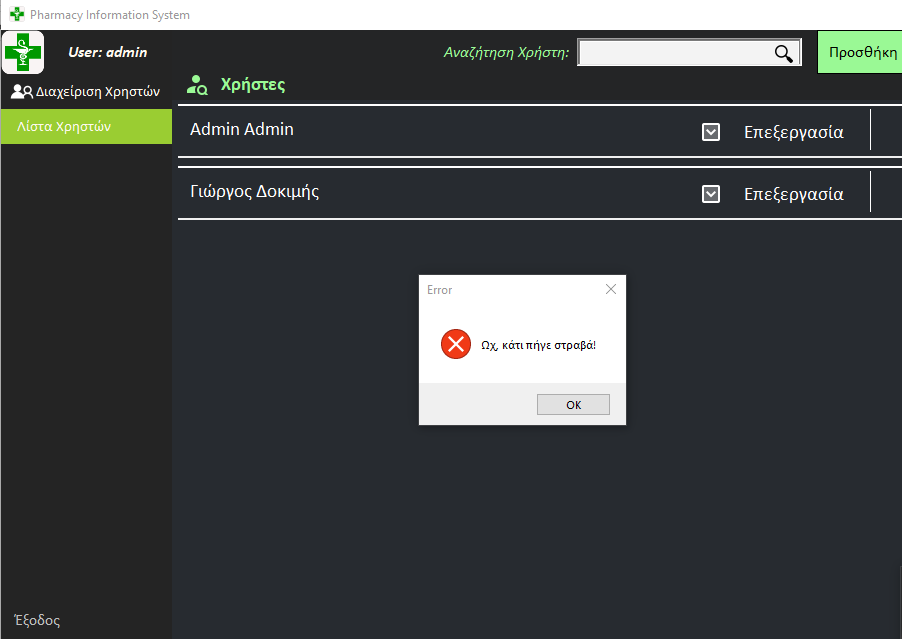
Κάθε αναφορά έχει τον κατάλληλο τίτλο ο οποίος αναφέρεται στο μέρος/σπρίντ η οποία αυτή υλοποιήθηκε καθώς και την σειρά της με την εξής μορφή: Εάν είχαμε μια αναφορά η οποία γράφτηκε κατά το τρίτο σπρίντ και είναι η δεύτερη η οποία γράφτηκε κατά την διάρκεια αυτού τότε ο τίτλος της εμπεριέχει το χαρακτηριστικό (3.2). Ακολουθούνε οι αναφορές:

## Αναφορά προβληματικής συμπεριφοράς εφαρμογής (1.1)

Η δυσλειτουργία αυτή έκανε την εμφάνιση της κατά την διάρκεια δοκιμών του πρώτου sprint και έχει να κάνει σχετικά με την αλλαγή στοιχείων ενός χρήστη. Πιο συγκεκριμένα, όταν ο admin δημιουργεί έναν χρήστη, έπειτα άλλαζε τα στοιχεία του τότε το πρόγραμμα αδυνατούσε να ολοκληρώσει την διαδικασία αλλαγής των στοιχείων εμφανίζοντας πάντα ένα μήνυμα σφάλματος. Ακολουθούν τα ανάλογα screenshot:

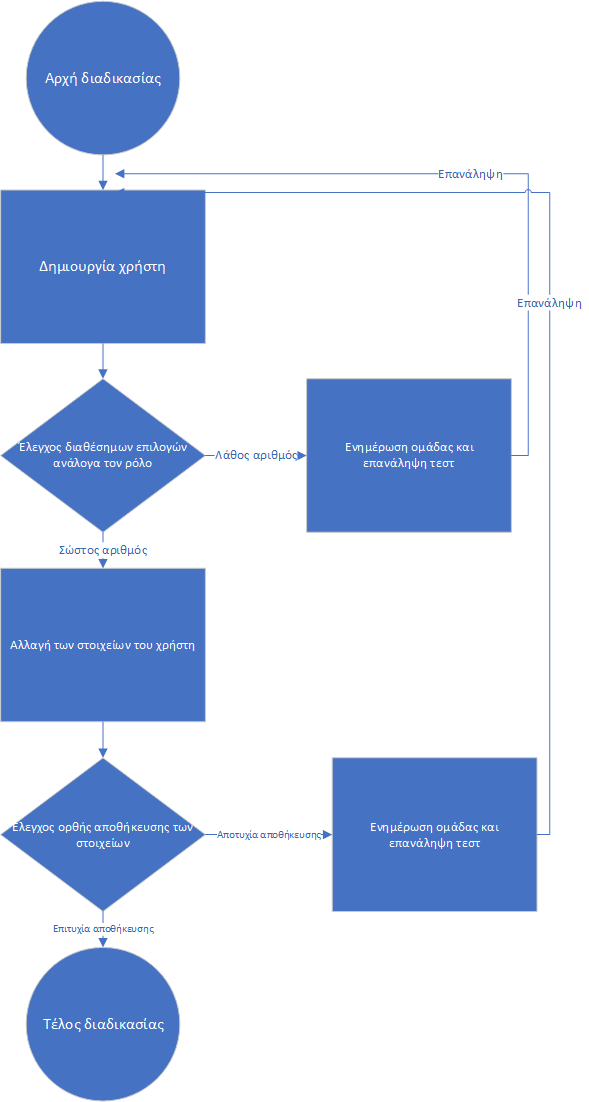


*(Στοιχεία χρήστη πριν γίνει αλλαγή των στοιχείων)*

**

*(Μήνυμα σφάλματος μετά την αλλαγή στοιχείων)*

Αξιοσημείωτο είναι το γεγονός ότι μετά την αποτυχία αλλαγής των στοιχείων και την εμφάνιση του μηνύματος σφάλματος ο χρήστης γίνεται αόρατος όπως γίνεται εμφανές στην δεύτερη εικόνα. Αυτό για να διορθωθεί πρέπει να γίνει έξοδος και επανασύνδεση του admin. Ακολουθεί αναλυτικό διάγραμμα:



*(Διάγραμμα εύρεσης σφάλματος)*

Η εύρεση του συγκεκριμένου λάθους έγινε στον έλεγχο της ορθής αλλαγής στοιχείων όσο αναφορά τους χρήστες που αποτελούνε το σύστημα της εφαρμογής. Ποιο συγκεκριμένα το σφάλμα βρέθηκε στο βήμα 1.6 του διαγράμματος. Ακολουθεί ανάλυση βημάτων:

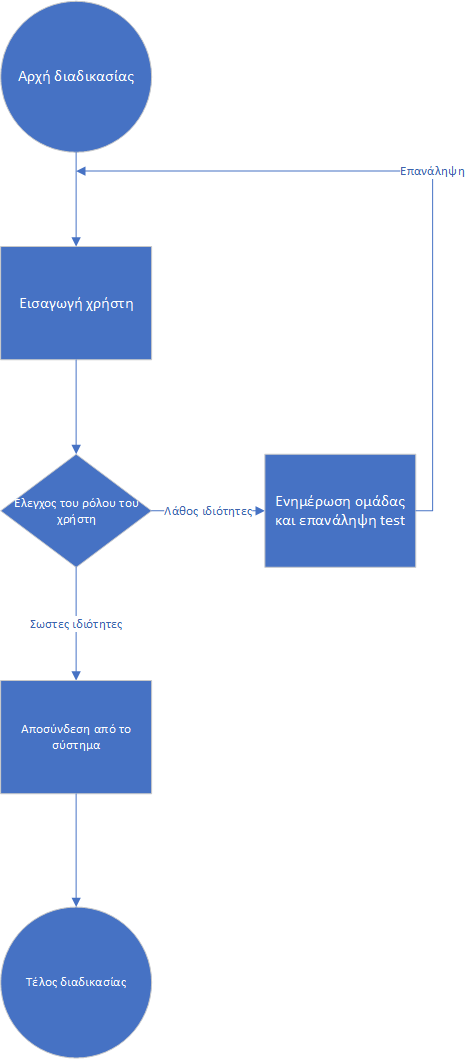
|  |
| --- |
| * 1. Αρχή διαδικασίας   2. Δημιουργία χρήστη   3. Έλεγχος διαθέσιμων επιλογών   1.3.1 Λάθος αριθμός και ενημέρωση 1.3.2 Σωστός αριθμός στοιχείων, συνέχιση  1.5 Αλλαγή των στοιχείων του χρήστη  1.6 Έλεγχος ορθής αποθήκευσης των στοιχείων  1.6.1 Λάθος στοιχεία και ενημέρωση 1.6.2 Σωστά στοιχεία, συνέχιση  1.7 Τέλος διαδικασίας |

**Λήξη αναφοράς.**

## Αναφορά προβληματικής συμπεριφοράς εφαρμογής (2.1)

Η δεύτερη δυσλειτουργία βρέθηκε κατά την διάρκεια δημιουργίας χρήστη στην οθόνη συμπλήρωσης στοιχείων αυτού. Συγκεκριμένα κατά την διάρκεια χρήσης της εφαρμογής με ιδιότητες admin εάν γινότανε η επιλογή δημιουργίας κάποιου χρήστη με κενή θέση στην επιλογή ιδιότητας ρόλου το σύστημα του έδινε την ιδιότητα του διαχειριστή έχοντας έτσι έναν μεγάλο αριθμό χρηστών με λάθος ρόλο. Η εμφάνιση της δυσλειτουργίας ήτανε σταθερή που πάει να πει ότι εμφανιζότανε 100% κατά την διάρκεια των test.

### Ανάλυση εύρεσης λάθους (διάγραμμα)

****

Η εύρεση του λάθους έγινε στον έλεγχο του ρόλου του χρήστη μετά την δημιουργία αυτού ενώ το συγκεκριμένο πεδίο παρέμεινε άδειο. Πιο συγκεκριμένα η εύρεση λάθους έγινε στο 1.3 και ακολουθήθηκε το μονοπάτι του 1.3.1. Ακολουθεί ανάλυση βημάτων του διαγράμματος:

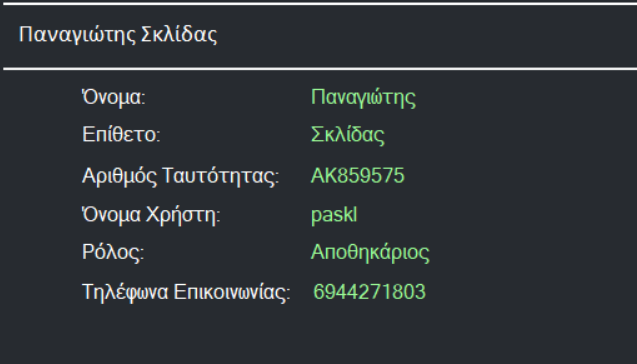
|  |
| --- |
| * 1. Αρχή διαδικασίας   2. Εισαγωγή χρήστη   3. Έλεγχος του ρόλου του χρήστη   1.3.1 Λάθος ιδιότητες και ενημέρωση 1.3.2 Σωστές ιδιότητες και συνέχιση  1.5 Αποσύνδεση από το σύστημα  1.6 Τέλος διαδικασίας |

Η δυσλειτουργία αυτή βρέθηκε κατά την διάρκεια δοκιμασιών στην δεύτερη φάση της ανάπτυξης της εφαρμογής και μπορεί να αποδειχθεί μεγάλη απειλή προς αυτή.

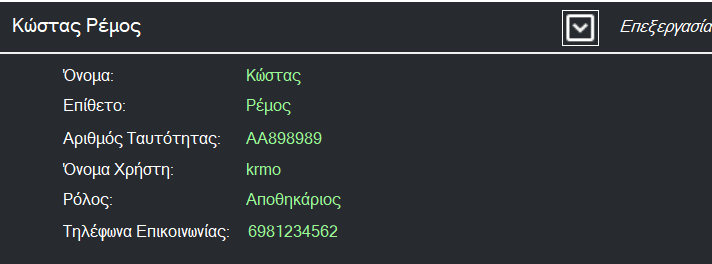
**Λήξη αναφοράς.**

## Αναφορά προβληματικής συμπεριφοράς εφαρμογής (2.2)

Η συγκεκριμένη δυσλειτουργία εμφανίστηκε κατά την διάρκεια δημιουργίας χρηστών με την ιδιότητα του admin στην εφαρμογή. Πιο συγκεκριμένα, η εφαρμογή κατόπιν της δημιουργίας του χρήστη, ανεξαρτήτως της ιδιότητας αυτού, όταν το πρόγραμμα υπολογίζει το τυχαία δημιουργημένο όνομα χρήστη υπολογίζει και τους τόνους του ονόματος ως χαρακτήρες, αυτό έχει ως αποτέλεσμα την δημιουργία ονομάτων χρήστη μικρότερα των 6 χαρακτήρων. Η δυσλειτουργία αυτή είχε μια 0,16% (2/14) πιθανότητα εμφάνισης κατά την διάρκεια χρήσης της εφαρμογής. Ακολουθούνε τα δύο παραδείγματα.



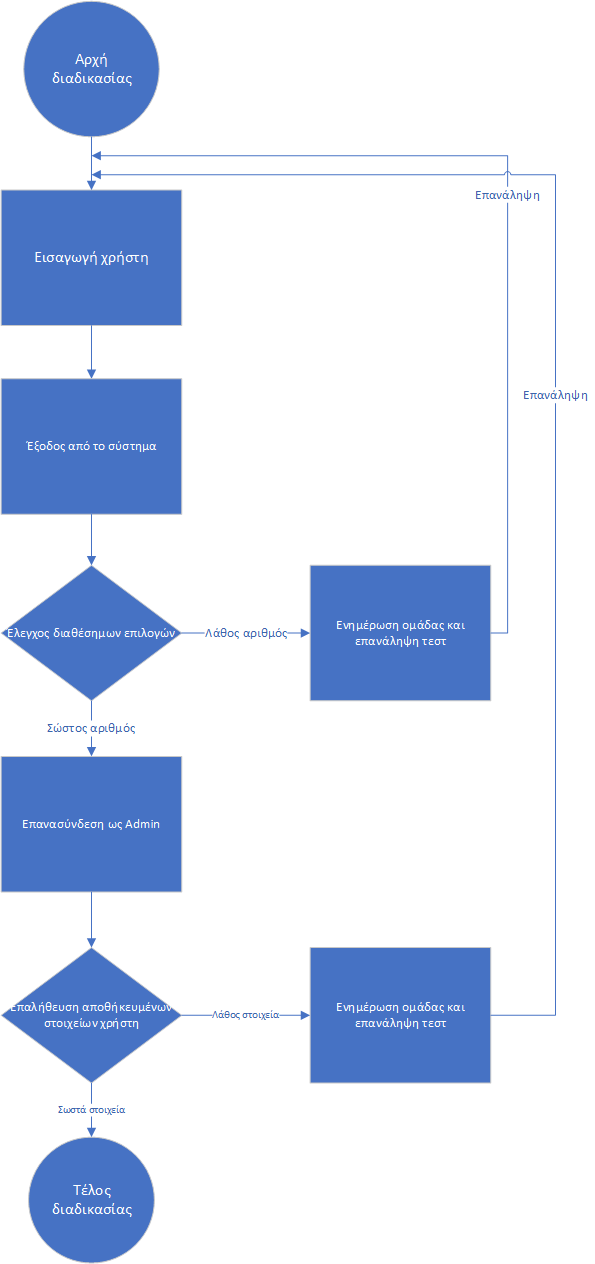
*(Πρώτο παράδειγμα)*



*(Δεύτερο παράδειγμα)*

Στα προαναφερόμενα παραδείγματα η έξοδος που προέκυψε ήτανε λάθος καθώς στην πρώτη περίπτωση ο χρήστης ονόματι ¨Παναγιώτης Σκλίδας¨ έχει δοθεί το όνομα χρήστη ¨paskl¨ των 5 γραμμάτων ενώ αντίστοιχα ο χρήστης ¨Κώστας Ρέμος¨ έχει δοθεί το όνομα χρήστη ¨krmo¨.

### Ανάλυση εύρεσης λάθους (διαγράμματα)



Η εύρεση του λάθους έγινε στην επαλήθευση στοιχείων του χρήστη όπου και γράφτηκε το report. Πιο συγκεκριμένα η εύρεση λάθους έγινε στο 1.6 και ακολουθήθηκε το μονοπάτι του 1.6.1. Ακολουθεί ανάλυση βημάτων του διαγράμματος:

|  |
| --- |
| * 1. Αρχή διαδικασίας   2. Εισαγωγή χρήστη   3. Έξοδος από το σύστημα   4. Έλεγχος διαθέσιμων επιλογών   1.4.1 Λάθος αριθμός και ενημέρωση 1.4.2 Σωστός αριθμός στοιχείων, συνέχιση  1.5 Επανασύνδεση ως admin  1.6 Επαλήθευση αποθηκευμένων στοιχείων χρήστη  1.6.1 Λάθος στοιχεία και ενημέρωση 1.6.2 Σωστά στοιχεία, συνέχιση  1.7 Τέλος διαδικασίας |

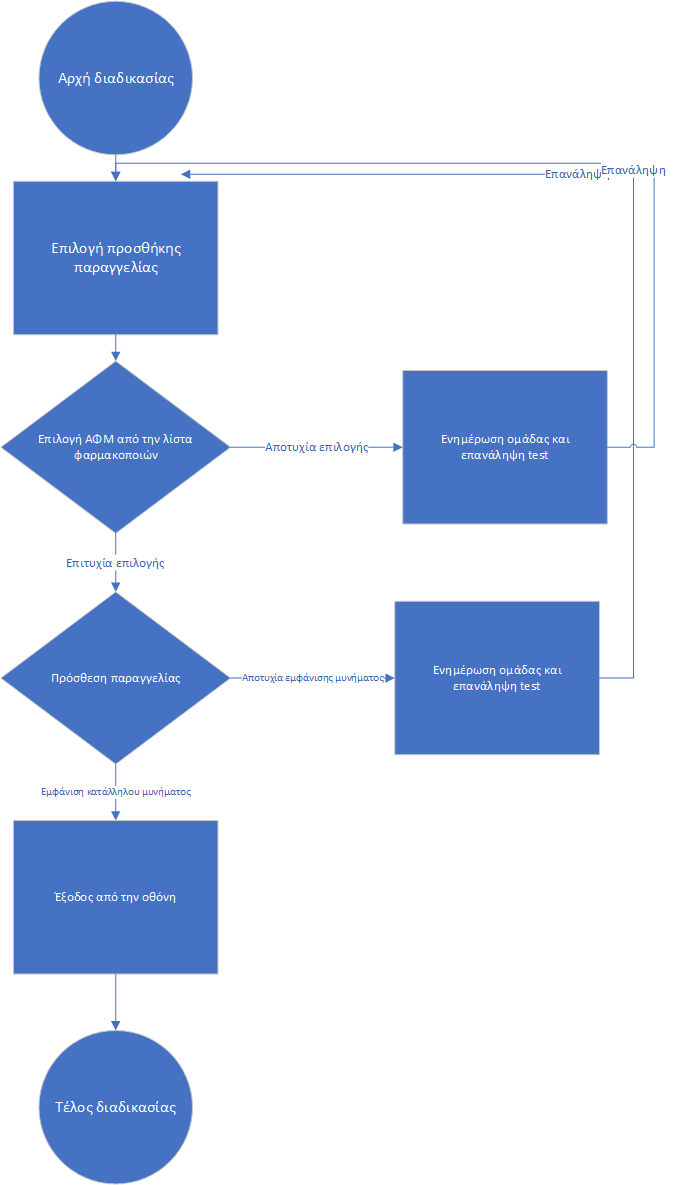
Η δυσλειτουργία αυτή αν και κυρίως άκακη και σπάνια με ένα ποσοστό εμφάνισης μικρότερο του 1%, είναι αξιοσημείωτη καθώς μπορεί να μπερδέψει τον μελλοντικό χρήστη και να οδηγήσει σε περεταίρω προβλήματα.

**Λήξη αναφοράς.**

## Αναφορά προβληματικής συμπεριφοράς εφαρμογής (3.1)

Κατά την διάρκεια εξαγωγής ελέγχων τρίτης φάσης και συγκεκριμένα στην οθόνη του πωλητή υπήρξε μια καταστροφική δυσλειτουργία η οποία crashαρε την εφαρμογή κάθε φορά που ο χρήστης ακολουθούσε συγκεκριμένα βήματα. Πιο αναλυτικά, όταν γινότανε επιλογή ΑΦΜ (φαρμακοποιού) καθώς τα υπόλοιπα πεδία της παραγγελίας ήτανε κενά και ο χρήστης πατούσε προσθήκη τότε αυτό οδηγούσε σε απότομό κλείσιμο της εφαρμογής 100% των δοκιμών. Ακολουθεί διάγραμμα και ανάλυση:

### Ανάλυση εύρεσης λάθους (διάγραμμα)



Η εύρεση του λάθους έγινε στην προσθήκη παραγγελίας όπου και γράφτηκε το report. Πιο συγκεκριμένα η εύρεση λάθους έγινε στο 1.4 και ακολουθήθηκε το μονοπάτι του 1.4.1. Ακολουθεί ανάλυση βημάτων του διαγράμματος:

|  |
| --- |
| * 1. Αρχή διαδικασίας   2. Επιλογή προσθήκης παραγγελίας   3. Επιλογή ΑΦΜ Φαρμακοποιού   1.3.1 Αποτυχία επιλογής και ενημέρωση 1.3.2 Επιτυχία επιλογής, συνέχιση   * 1. Πρόσθεση παραγγελίας   1.4.1 Αποτυχία εμφάνισης μηνύματος 1.4.2 Εμφάνιση μηνύματος, συνέχιση   * 1. Έξοδος από την οθόνη   2. Τέλος διαδικασίας |

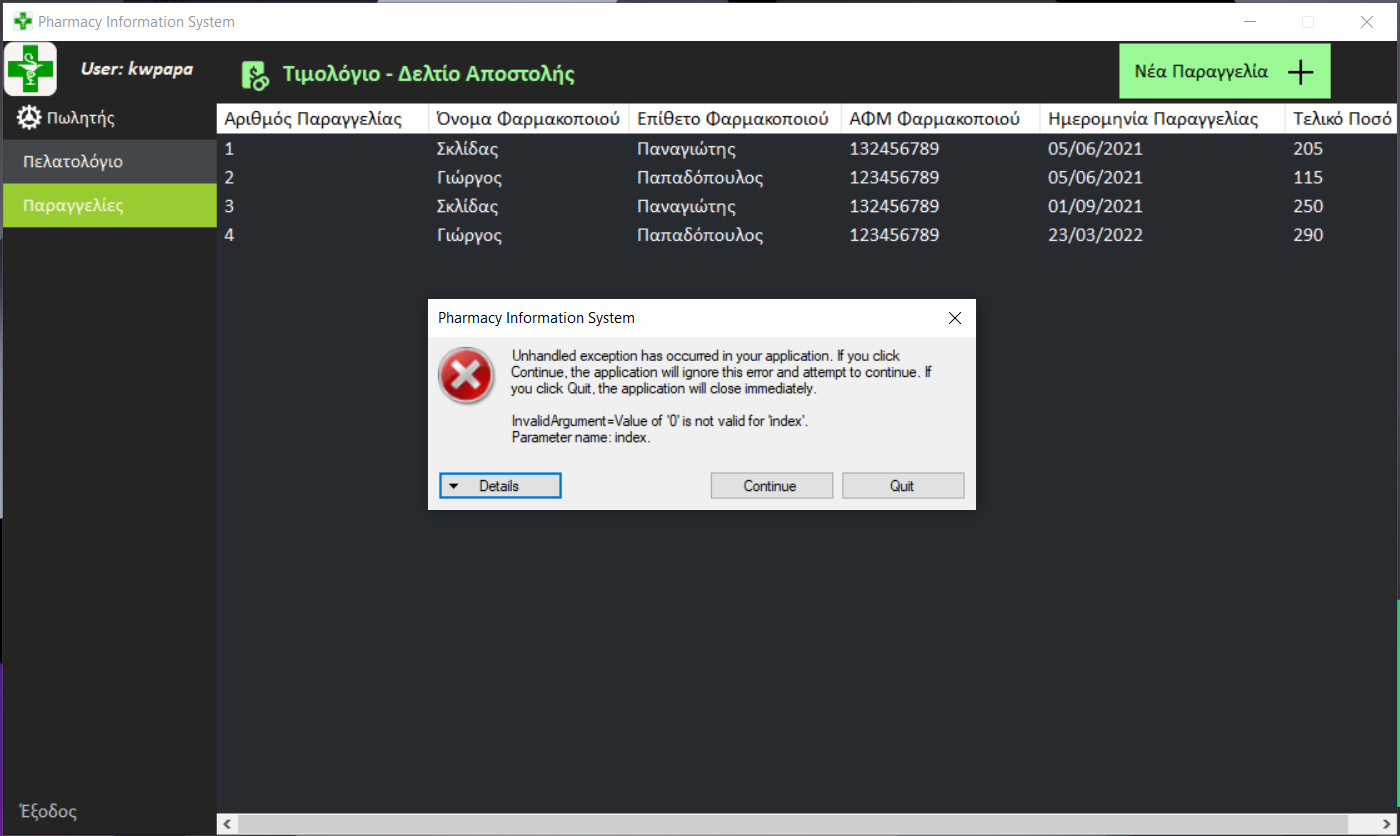
**Λήξη αναφοράς.**

## Αναφορά προβληματικής συμπεριφοράς εφαρμογής (3.2)

Κατά την διάρκεια δοκιμών επάνω στην οθόνη του πωλητή, συγκεκριμένα την οθόνη παραγγελιών, παρατηρήθηκαν διάφορες δυσλειτουργίες μια εκ των οποίων είχε να κάνει με την επεξεργασία ή διαγραφή κάποιας παραγγελίας χωρίς να έχει επιλεχθεί κάποια παραγγελία, κάτι που οδηγούσε σε crash/κλείσιμο της εφαρμογής καθώς δεν υπήρχε το κατάλληλο check στον κώδικα. Το σφάλμα αυτό οδηγούσε πάντα στο κλείσιμο της εφαρμογής επομένως ήτανε κρίσιμο να διορθωθεί προκειμένου η λειτουργικότητα της εφαρμογής να παραμείνει άθικτη. Ακλουθεί κατάλληλο διάγραμμα και ανάλυση:

### Ανάλυση εύρεσης λάθους (διάγραμμα)





*(Απόσπασμα δυσλειτουργίας της εφαρμογής)*

Η εύρεση του λάθους έγινε στην τροποποίηση παραγγελίας όπου και γράφτηκε το report. Πιο συγκεκριμένα η εύρεση λάθους έγινε στο 1.4 και ακολουθήθηκε το μονοπάτι του 1.4.1. Ακολουθεί ανάλυση βημάτων του διαγράμματος:

|  |
| --- |
| * 1. Αρχή διαδικασίας   2. Προσθήκη προϊόντων   3. Έλεγχος εγγραφής χαρακτηριστικών των προϊόντων   1.3.1 Αποτυχία εγγραφής και ενημέρωση 1.3.2 Επιτυχία εγγραφής , συνέχιση   * 1. Αλλαγή στοιχείων και επανάληψη αποθήκευσης   1.4.1 Αποτυχία αποθήκευσης και ενημέρωση 1.4.2 Επιτυχία αποθήκευσης, συνέχιση  1.5 Προσθήκη στοιχείων για έλεγχο του αποτελέσματος  1.5.1 Λάθος αποτέλεσμα και ενημέρωση 1.5.2 Σωστό αποτέλεσμα, συνέχιση   * 1. Έξοδος από το σύστημα   2. Τέλος διαδικασίας |

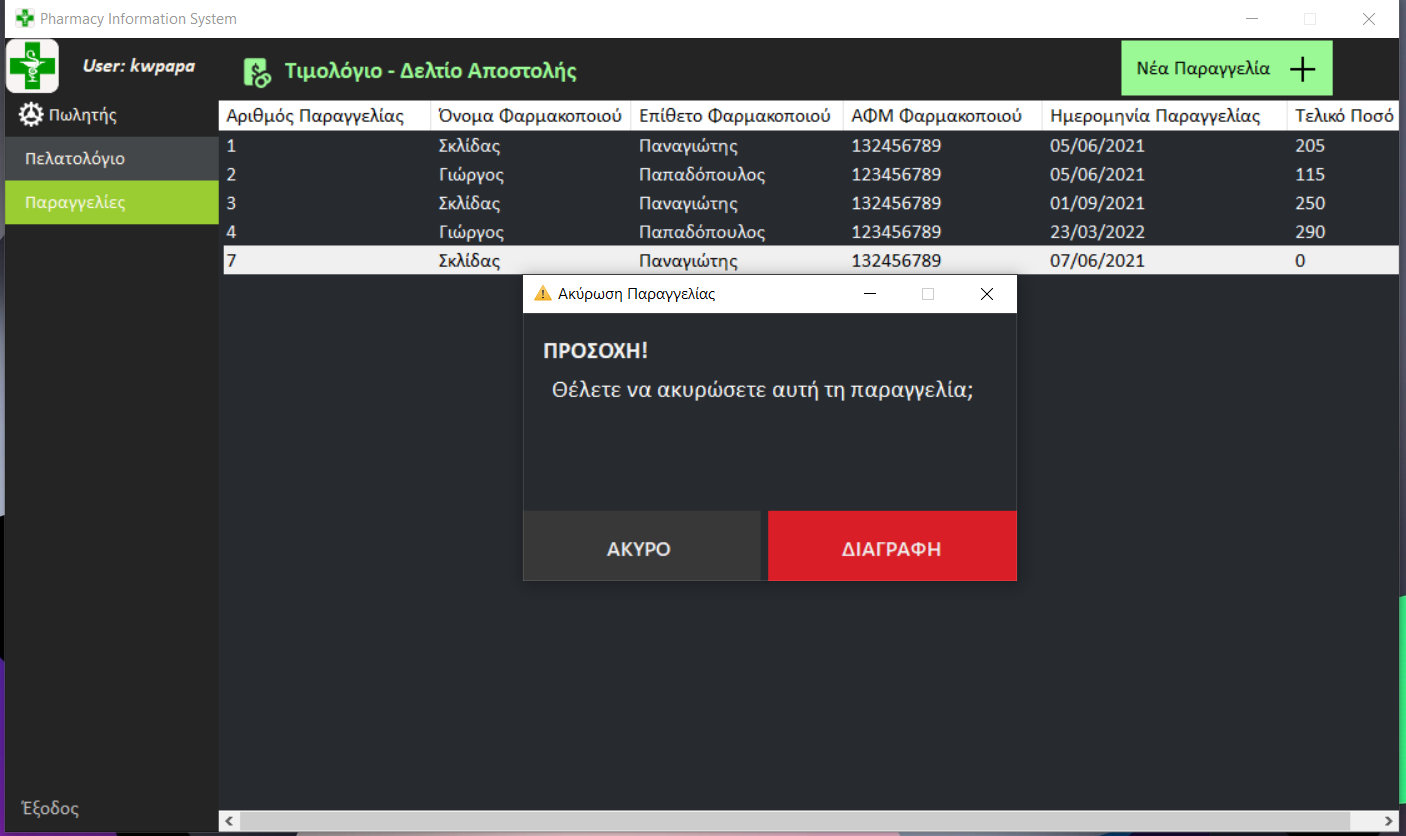
**Λήξη αναφοράς.**

## Αναφορά προβληματικής συμπεριφοράς εφαρμογής (3.3)

Κατά την διάρκεια ελέγχων παρατηρήθηκε άλλη μια δυσλειτουργία στην οθόνη παραγγελιών του πωλητή και συγκεκριμένα στην διαγραφή κάποιας παραγγελίας. Η δυσλειτουργία αυτή είχε να κάνει με τα κουμπιά του μηνύματος διαγραφής. Το σφάλμα αυτό καθιστούσε τα κουμπιά επιβεβαίωσης άχρηστα καθώς δεν έφερναν καμία αλλαγή επομένως ο χρήστης δεν μπορούσε να διαγράψει κάποια παραγγελία. Ακολουθεί κατάλληλο διάγραμμα και ανάλυση:

### Ανάλυση εύρεσης λάθους (διάγραμμα)





*(Το κουμπί επιβεβαίωσης διαγραφής δεν έχει λειτουργία)*

Η εύρεση του λάθους έγινε στην αλλαγή των στοιχείων της λίστας (διαγραφή παραγγελίας) όπου και γράφτηκε το report. Πιο συγκεκριμένα η εύρεση λάθους έγινε στο 1.4 και ακολουθήθηκε το μονοπάτι του 1.4.1. Ακολουθεί ανάλυση βημάτων του διαγράμματος:

|  |
| --- |
| * 1. Αρχή διαδικασίας   2. Προσθήκη προϊόντων   3. Έλεγχος εγγραφής χαρακτηριστικών των προϊόντων   1.3.1 Αποτυχία εγγραφής και ενημέρωση 1.3.2 Επιτυχία εγγραφής , συνέχιση   * 1. Αλλαγή στοιχείων και επανάληψη αποθήκευσης   1.4.1 Αποτυχία αποθήκευσης και ενημέρωση 1.4.2 Επιτυχία αποθήκευσης, συνέχιση  1.5 Προσθήκη στοιχείων για έλεγχο του αποτελέσματος  1.5.1 Λάθος αποτέλεσμα και ενημέρωση 1.5.2 Σωστό αποτέλεσμα, συνέχιση   * 1. Έξοδος από το σύστημα   2. Τέλος διαδικασίας |

## Αναφορά προβληματικής συμπεριφοράς εφαρμογής (3.4)

Κατά την διάρκεια ελέγχων της εφαρμογής, συγκεκριμένα στην οθόνη παραγγελίας του πωλητή, όταν ο χρήστης αποφάσιζε να τροποποιήσει την παραγγελία του αλλά απλά πατούσε έξοδο από την οθόνη της φόρμας αλλαγών τότε το φάρμακο αφαιρούταν εντελώς από την λίστα. Το πρόβλημα αυτό μάλιστα είχε εμφανιστεί και στην αναφορά (3.2) μόνο που αυτή την φορά δεν οδηγεί το σύστημα σε ολοκληρωτικό κλείσιμο. Αξιοσημείωτο είναι επίσης το γεγονός ότι στην προσπάθεια λύσης του ανάλογου προβλήματος του 3.2 αυτή η δυσλειτουργία βρέθηκε κατά την διάρκεια κλήσης με τους προγραμματιστές, οπότε λείπει φωτογραφικό υλικό.

### Ανάλυση εύρεσης λάθους (διάγραμμα)



Η εύρεση του λάθους έγινε στην τροποποίηση παραγγελίας όπου και γράφτηκε το report. Πιο συγκεκριμένα η εύρεση λάθους έγινε στο 1.4 και οδήγησε στο μονοπάτι του 1.4.1. Ακολουθεί ανάλυση βημάτων του διαγράμματος:

|  |
| --- |
| * 1. Αρχή διαδικασίας   2. Προσθήκη προϊόντων   3. Έλεγχος εγγραφής χαρακτηριστικών των προϊόντων   1.3.1 Αποτυχία εγγραφής και ενημέρωση 1.3.2 Επιτυχία εγγραφής , συνέχιση   * 1. Αλλαγή στοιχείων και επανάληψη αποθήκευσης   1.4.1 Αποτυχία αποθήκευσης και ενημέρωση 1.4.2 Επιτυχία αποθήκευσης, συνέχιση  1.5 Προσθήκη στοιχείων για έλεγχο του αποτελέσματος  1.5.1 Λάθος αποτέλεσμα και ενημέρωση 1.5.2 Σωστό αποτέλεσμα, συνέχιση   * 1. Έξοδος από το σύστημα   2. Τέλος διαδικασίας |

# Βιβλιογραφία

Cohn, M. (n.d.). *Scrum Methodology and Project Management*. Mountain Goat Software. https://www.mountaingoatsoftware.com/agile/scrum

ce.uth, C. E. U. T. H. (n.d.). *Εισαγωγή στην εκτίμηση κόστους Λογισμικού / Μέθοδος COCOMO*. Courses.e-Ce.Uth.Gr. https://courses.e-ce.uth.gr/CE420/Fall12/lectures/4-COCOMOModel.pdf

*Online Radar Chart Templates*. (n.d.). Online Radar Chart Generator. https://online.visual-paradigm.com/charts/templates/radar-charts/

C. (2014, August 5). *What is FURPS+?* Business Analyst Training in Hyderabad - COEPD. https://businessanalysttraininghyderabad.wordpress.com/2014/08/05/what-is-furps/

bugfender.com (Μέθοδοι test).

smartbear.com (Μεθοδολογίες και προσεγγίσεις).

Σημειώσεις και διαφάνειες καθηγητή.