Τεχνολογία Λογισμικού
Τομέας Ηλεκτρονικής και Υπολογιστών
Τμήμα ΗΜΜΥ
Α.Π.Θ.

8ο Εξάμηνο
Άνοιξη 2016

Uni-Food

Because food matters

Σχεδίαση Συστήματος

Del.3.1

Έκδοση 1.0.3

Μαυροδής Κωνσταντίνος (kmavrodis@outlook.com)
Νήρας Δημήτρης (niras.94@gmail.com)
Σχινάς Γιώργος (schinas.georgios@outlook.com)
Χατζηθωμά Ανδρέας (andreas_h92@hotmail.com)

09 / 06 / 2016

Ιστορικό Αλλαγών

Όνομα	Ημ/νία	Περιγραφή Αλλαγής	Εκδ.
Α. Συμεωνίδης	29/05/2009	Δημιουργία Εγγράφου Προσαρμογή του ESA software engineering standards guidelines (1991) και του εγγράφου SDD document, από τους Bruegge και Dutoit (2004).	0.1
Uni-Food	06/05/2016	Συγγραφή 1ου και 2ου Κεφαλαίου	0.2.0
Uni-Food	08/05/2016	Μικρές αλλαγές στο 2ο Κεφάλαιο	0.2.1
Uni-Food	09/05/2016	Συγγραφή 3ου Κεφαλαίου	0.3.0
Uni-Food	10/05/2016	Συγγραφή 4ου Κεφαλαίου	0.4.0
Uni-Food	11/05/2016	Αλλαγές στο 3ο και 4ο Κεφάλαιο	0.4.1
Uni-Food	11/05/2016	Συμπλήρωση Πίνακα Ιχνηλασιμότητας	0.5.0
Uni-Food	12/05/2016	Συμπλήρωση Ανοιχτών Θεμάτων	0.6.0
Uni-Food	15/05/2016	Διόρθωση Αρχιτεκτονικής Συστήματος	0.6.0
Uni-Food	19/05/2016	Ολοκλήρωση και σύνθεση του εγγράφου	1.0.0
Uni-Food	04/06/2016	Διόρθωση προτύπων συμπεριφοράς	1.0.1
Uni-Food	06/06/2016	Διόρθωση Στόχων Σχεδίασης και Αρχιτεκτονικής 3 επιπέδων	1.0.2
Uni-Food	08/06/2016	Προσθήκη Σεναρίων Χρήσης για τις Οριακές Συνθήκες	1.0.3

Μέλη Ομάδας Ανάπτυξης

Όνομα	ΟΑ	Email
Α. Συμεωνίδης	*	asymeon@issel.ee.auth.gr
Χ. Ζολώτας	*	christopherzoltas@issel.ee.auth.gr
Μαυροδής Κωνσταντίνος	Uni-Food	kmavrodis@outlook.com
Νήρας Δημήτρης	Uni-Food	niras.94@gmail.com
Σχινάς Γιώργος	Uni-Food	schinas.georgios@outlook.com
Χατζηθωμά Ανδρέας	Uni-Food	andreas_h92@hotmail.com

Πίνακας Περιεχομένων

Λίστα Σχημάτων

1. Εισαγωγή

Η εφαρμογή Uni-Food αποτελεί ένα έργο λογισμικού που στοχεύει στη διευκόλυνση των μελών της πανεπιστημιακής κοινότητας, ενσωματώνοντας σύγχρονες τεχνολογίες και πρόσθετες υπηρεσίες. Για την επίτευξη του σκοπού αυτού, καθορίστηκαν αρχικά οι απαιτήσεις των χρηστών, καθώς και οι απαιτήσεις λογισμικού, που αναλύονται στο "Έγγραφο Απαιτήσεων Χρηστών" και στο "Έγγραφο Απαιτήσεων Λογισμικού" αντίστοιχα. Μετά τον ακριβή καθορισμό των απαιτήσεων αυτών, σειρά έχει ο αρχιτεκτονικός σχεδιασμός του συστήματος.

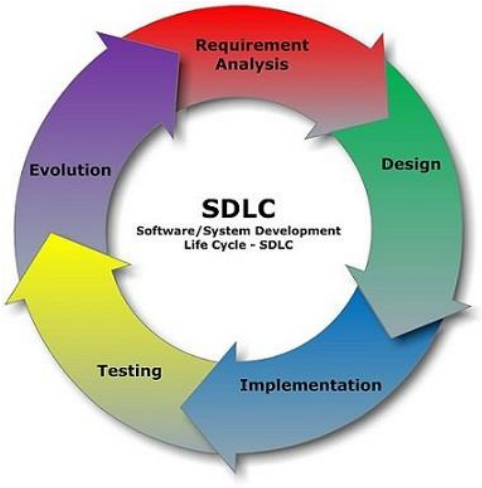
1.1 Στόχος του εγγράφου

Ο στόχος του παρόντος εγγράφου είναι η περιγραφή της σχεδίασης του συστήματος Uni-Food. Το επίπεδο της σχεδίασης αναφέρεται στη λογική αρχιτεκτονική τμημάτων του συστήματος. Η ιεραρχία λειτουργιών αναλύεται σε συνθετικά αλληλεπιδρώντα μέρη.

Περιγράφεται ο σκοπός και η λειτουργικότητα του έργου, ούτως ώστε να διαφανούν επακριβώς οι στόχοι της σχεδίασης. Στη συνέχεια περιγράφεται λεπτομερώς το μοντέλο σχεδίασης του συστήματος, με αναφορές, όπου χρειάζεται, σε προηγούμενα έγγραφα. Γίνεται λεπτομερής περιγραφή των πακέτων υποσυστημάτων και των προτύπων σχεδίασης που χρησιμοποιούνται.

Μέσω της αλληλεπίδρασης των υποσυστημάτων που περιγράφονται σε αυτό το έγγραφο, βλέπουμε να υλοποιείται η παρατηρηθείσα στο χρήστη συμπεριφορά που περιγράφηκε κατά την ανάλυση στο επίπεδο λειτουργικότητας. Στόχοι στην παραγωγή λογισμικού είναι η δόμηση και η αφαίρεση. Η δόμηση έχει επιτευχθεί με το διαχωρισμό της παραγωγής σε επίπεδα και με τη σύνταξη των εγγράφων απαιτήσεων που περιγράφουν αυτά τα επίπεδα. Αφαίρεση επιτυγχάνεται με χρήση κατάλληλων μοντέλων. Μία προσέγγιση με μοντέλα απαιτείται, η οποία να είναι εκφραστική αρκετά, ούτως ώστε να αντιμετωπίζονται όλες οι αναφερθείσες όψεις της αρχιτεκτονικής. Η περιγραφή ενός επίσημου μοντέλου με κατάλληλη θεωρία και υποθέσεις είναι χρήσιμη στη γραπτή τεκμηρίωση και περιγραφή και στην επισημοποίηση και συγκεκριμενοποίηση ανεπίσημων απαιτήσεων. Τα μοντέλα συλλαμβάνουν γνώση για όσα αφορούν την ανάπτυξη, τεχνογνωσία για το προϊόν και δίνουν κατευθυντήριες γραμμές για την ανάπτυξη του συστήματος.

Μετά, λοιπόν, από τον προσδιορισμό της αρχιτεκτονικής, σύμφωνα με τις μη λειτουργικές απαιτήσεις και την περιγραφή των εξαρτήσεων και της ιεραρχίας λειτουργιών, βρισκόμαστε στο σημείο της αποδόμησης του συστήματος. Εδώ πρέπει να καθοριστεί λεπτομερώς η λογική αρχιτεκτονική και οι αλληλεπιδράσεις μεταξύ των τμημάτων της. Ουσιαστικά, σε αυτό το βήμα μπορούμε να εξακριβώσουμε ότι από άποψη εφαρμογής, η αρχιτεκτονική που σχεδιάστηκε είναι ορθή. Τέλος, σε αυτό το σημείο, είναι πλέον εφικτή η λήψη αποφάσεων για τα είδη λογισμικού που θα χρησιμοποιηθούν, τη σχεδίαση του υλικού και για την εισαγωγή στο στάδιο της ανάπτυξης.



Σχήμα 1. Κύκλος Ανάπτυξης Λογισμικού

1.2 Αντικείμενο του λογισμικού

Το λογισμικό του συστήματος Uni-Food έχει ως αντικείμενο τη βελτίωση της ποιότητας της Πανεπιστημιακής Λέσχης, δίνοντας την δυνατότητα στους φοιτητές για την επιλογή του μενού της επόμενης εβδομάδας, αλλά και προσφέροντας τους χρήσιμες πληροφορίες για την προτεινόμενη ώρα προσέλευσης και τον μέσο χρόνο αναμονής. Δίνει επιπλέον τη δυνατότητα στου Διαχειριστές της Πανεπιστημιακής Λέσχης για τοποθέτηση ψηφοφοριών και προγραμμάτων σίτισης.

Το λογισμικό δέχεται ως εισόδους τις επιλογές των φοιτητών από την εκάστοτε ψηφοφορία που έχει θέσει ο Διαχειριστής του συστήματος, καθώς και τον αριθμό των φοιτητών που εισέρχονται στην Λέσχη την εκάστοτε χρονική στιγμή, συνεργαζόμενο με το API για αναγνώριση των Ακαδημαϊκών Ταυτοτήτων που προσφέρεται από το ΚΗΔ.

Έπειτα διαχειρίζεται τις επιλογές των φοιτητών από την εκάστοτε ψηφοφορία, οργανώνοντάς τις σε μια Βάση Δεδομένων και δίνει τη δυνατότητα στον Διαχειριστή του συστήματος να επιλέξει το πρόγραμμα σίτισης την επόμενης εβδομάδας. Επιπλέον βάσει ειδικού αλγορίθμου υπολογίζει τον μέσο χρόνο αναμονής και την προτεινόμενη ώρα προσέλευσης βασιζόμενο στην πληροφορία που δέχεται από το API του ΚΗΔ.

Με βάση επομένως αυτές τις πληροφορίες το σύστημα είναι σε θέση να ικανοποιεί τα αιτήματα των χρηστών για επιλογή του μενού της Πανεπιστημιακής Λέσχης, καθώς και για προβολή των χρόνων αναμονής και προσέλευσης. Ικανοποιεί επίσης και τα αιτήματα των Διαχειριστών για υποβολή προγραμμάτων σίτισης αλλά και ψηφοφοριών.

1.3 Ορισμοί, Ακρωνύμια, Συντομεύσεις

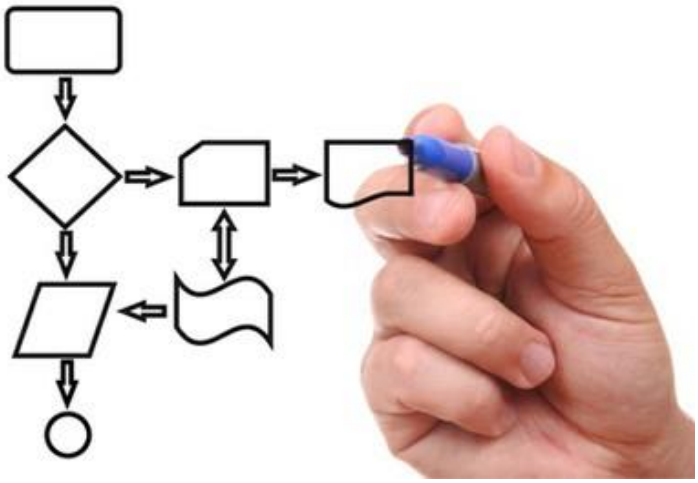
ΟΑ : Ομάδα Ανάπτυξης
Client – Server : Πελάτης - Διακομιστής
ΜΛΑ : Μη Λειτουργική Απαίτηση
GUI : Graphical User Interface
UML : Unified Modelling Language
API : Application Programming Interface
DB : Database (Βάση Δεδομένων)
ΚΗΔ : Κέντρο Ηλεκτρονικής Διακυβέρνησης
ΠΦΛ : Πανεπιστημιακή Φοιτητική Λέσχη

1.4 Στόχοι Σχεδίασης

Στα πλαίσια του λογισμικού, η σχεδίαση είναι μια διαδικασία βελτιστοποίησης, της οποίας στόχος είναι η περιγραφή του τρόπου υλοποίησης των λειτουργικών απαιτήσεων, υπό τους περιορισμούς που θέτουν οι μη λειτουργικές απαιτήσεις και η οποία συμμορφώνεται με συγκεκριμένες αρχές καλής ποιότητας και συμβιβασμού των απαιτήσεων όλων των συμβαλλόμενων μερών, του πελάτη, του τελικού χρήστη και του προγραμματιστή.

Κοινή απαίτηση και των τριών ομάδων είναι η αξιοπιστία του συστήματος. Κάποιες από τις απαιτήσεις επικαλύπτονται με αυτές κάποιας άλλης ομάδας, πράγμα που διευκολύνει την εργασία της ομάδας ανάπτυξης. Κάποιες άλλες, όμως, είναι ανεξάρτητες ή και συγκρουόμενες μεταξύ τους και η ικανοποίησή τους απαιτεί πλήθος διαφορετικών διαδικασιών μέχρι την τελική εκπλήρωσή τους.

Κατά την αρχιτεκτονική ανάλυση του συστήματος και για την ικανοποίηση όλων των αναγκών, είναι βασική η ελαχιστοποίηση της πρόσθετης πολυπλοκότητας κατά τη συντήρηση και την επέκτασή του. Απαιτείται, λοιπόν, η εύρεση της χρυσής τομής μεταξύ της συνεκτικότητας και της σύζευξης των υποσυστημάτων του. Η επιθυμητή ισορροπία απορρέει από τον συνδυασμό υψηλής συνεκτικότητας και χαμηλής σύζευξης. Υψηλή συνεκτικότητα σημαίνει ότι οι κλάσεις ενός υποσυστήματος εκτελούν παρόμοια έργα και είναι συσχετισμένες μεταξύ τους, ενώ η χαμηλή σύζευξη υποδηλώνει ότι μια αλλαγή σε ένα υποσύστημα δεν θα επηρεάσει κάποιο άλλο.



Σχήμα 2. Σχεδίαση Λογισμικού σε UML

1.5 Αναγνωστικό Κοινό. Τρόπος Ανάγνωσης εγγράφου.

Το έγγραφο σχεδίασης συστήματος απευθύνεται στο παρακάτω αναγνωστικό κοινό:

- Αρμόδιοι μηχανικοί του Κέντρου Ηλεκτρονικής Διακυβέρνησης (ΚΗΔ) του Αριστοτελείου Πανεπιστημίου Θεσσαλονίκης, το οποίο έχει προκηρύξει την ανάθεση του έργου αυτού.
- Μηχανικοί λογισμικού που θα αναλάβουν την υλοποίηση του συστήματος και τη συγγραφή του κώδικα του λογισμικού.
- Μηχανικοί λογισμικού που θα αναλάβουν τη συντήρηση και εύρυθμη λειτουργία του συστήματος.

Καταρχάς, είναι σημαντικό να έχει προηγηθεί η ανάγνωση των δύο προηγούμενων εγγράφων («Έγγραφο Απαιτήσεων Χρηστών» και «Έγγραφο Απαιτήσεων Λογισμικού»), ώστε να υπάρχει συνοχή στην κατανόηση των λειτουργιών και των απαιτήσεων του έργου. Επίσης, η ανάγνωση των κεφαλαίων θα πρέπει να γίνει με την σειρά που υπάρχουν στο έγγραφο για να μη δημιουργηθούν προβλήματα στην κατανόησή του.

Οι αρμόδιοι μηχανικοί του ΚΗΔ καλό θα ήταν να δώσουν ιδιαίτερη προσοχή στα δύο πρώτα κεφάλαια, έτσι ώστε να κατανοήσουν την αρχιτεκτονική και τον τρόπο της αποδόμησης του συστήματος σε υποσυστήματα, αλλά και για να πιστοποιήσουν αν πληρούνται οι στόχοι σχεδίασης.

Αντίστοιχα, οι μηχανικοί λογισμικού θα πρέπει να δώσουν έμφαση στην κατανόηση των κεφαλαίων 3 και 4, αφού αυτά περιγράφουν διεξοδικά τον τρόπο λειτουργίας των υποσυστημάτων.

1.6 Επισκόπηση Εγγράφου

Το πρώτο κεφάλαιο του εγγράφου είναι εισαγωγικό και περιλαμβάνει γενικές πληροφορίες για το περιεχόμενο του εγγράφου αλλά και για τα κεφάλαια που ακολουθούν.

Στο δεύτερο κεφάλαιο αναλύονται και συγκρίνονται δυο αρχιτεκτονικές, η client-server και η αρχιτεκτονική τριών επιπέδων που θα μπορούσανε να χρησιμοποιηθούν στο σύστημα. Επιπλέον παρουσιάζεται η αρχιτεκτονική του συστήματος Uni-Food.

Το τρίτο κεφάλαιο περιγράφει την αποδόμηση του συστήματος. Το συνολικό σύστημα διασπάται σε υποσυστήματα και επιμέρους τμήματα, σύμφωνα με την αρχιτεκτονική που επιλέχθηκε στο δεύτερο κεφάλαιο και ταυτόχρονα παρουσιάζονται διαγράμματα συνδέσεων μεταξύ των υποσυστημάτων. Στο τέλος, αναλύονται θέματα γενικού ελέγχου, ασφάλειας, πρόσβασης και οριακών συνθηκών στα οποία πρέπει να δοθεί ιδιαίτερη προσοχή.

Στο τέταρτο κεφάλαιο περιγράφονται πρότυπα σχεδίασης που θα μπορούσαν να χρησιμοποιηθούν στο σύστημα. Τα πρότυπα αυτά προσδιορίζουν μια γενική σχεδιαστική λύση για ένα επαναλαμβανόμενο κοινό πρόβλημα, που εμφανίζεται στο σύστημα.

Το πέμπτο κεφάλαιο παρουσιάζει μια λίστα με ανοιχτά θέματα, τα οποία μπορούν να συμπεριληφθούν στο σύστημα μελλοντικά, σε σχέση με τις υπάρχουσες ή καινούργιες απαιτήσεις του πελάτη.

2. Τρέχουσα Αρχιτεκτονική Λογισμικού

2.1 Αρχιτεκτονική Client - Server (Πελάτη – Διακομιστή)

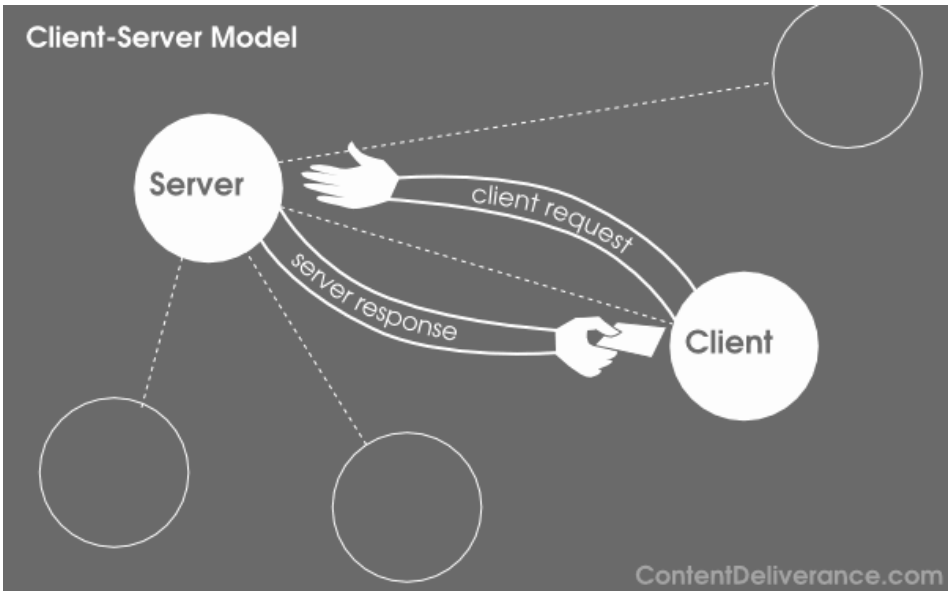
Ένα σύστημα το οποίο βασίζεται στην αρχιτεκτονική πελάτη - διακομιστή είναι ένα σύστημα στο οποίο το δίκτυο ενώνει διάφορους υπολογιστικούς πόρους, ώστε οι clients , ή αλλιώς front end, να μπορούν να ζητούν υπηρεσίες από έναν server, ή αλλιώς back end, ο οποίος προσφέρει πληροφορίες ή επιπρόσθετη υπολογιστική ισχύ.

Με άλλα λόγια, στο client-server μοντέλο, ο client θέτει μια αίτηση και ο server επιστρέφει μια ανταπόκριση ή κάνει μια σειρά από ενέργειες. Ο server μπορεί να ενεργοποιείται άμεσα για την αίτηση αυτή ή να προσθέτει την αίτηση σε μια ουρά. Η άμεση ενεργοποίηση για την αίτηση μπορεί, για παράδειγμα, να σημαίνει ότι ο server υπολογίζει έναν αριθμό και τον επιστρέφει αμέσως στον client. Η τοποθέτηση της αίτησης σε μια ουρά μπορεί να σημαίνει ότι η αίτηση πρέπει να τεθεί σε αναμονή για να εξυπηρετηθεί.

Τα πλεονεκτήματα του client-server computing είναι τα εξής:

- Αποτελεσματική χρήση της υπολογιστικής ισχύος.
- Μείωση του κόστους συντήρησης, δημιουργώντας συστήματα client-server που απαιτούν λιγότερη συντήρηση και κοστίζουν λιγότερο στην αναβάθμιση.
- Αύξηση της παραγωγικότητας, προσφέροντας στους χρήστες ξεκάθαρη πρόσβαση στις αναγκαίες πληροφορίες μέσω σταθερών και εύκολων στη χρήση διασυνδέσεων.
- Αύξηση της ευελιξίας και της δυνατότητας δημιουργίας συστημάτων που υποστηρίζουν πολλά περιβάλλοντα.

Ένας τρόπος δόμησης είναι η διαίρεση του συστήματος σε τρία επίπεδα, η οποία αναλύεται στην επόμενη ενότητα. Στα συγκεντρωτικά συστήματα, τα επίπεδα αυτά δεν χρειάζεται να είναι σαφώς διαχωρισμένα. Ωστόσο, κατά τον σχεδιασμό ενός κατανεμημένου συστήματος θα πρέπει να γίνει σαφής διάκριση μεταξύ τους, επειδή με αυτόν τον τρόπο κάθε επίπεδο μπορεί να κατανεμηθεί σε διαφορετικούς υπολογιστικούς κόμβους.



Σχήμα 3. Αρχιτεκτονική Πελάτη - Διακομιστή

2.2 Αρχιτεκτονική τριών επιπέδων (3 -Tier)

Στην αρχιτεκτονική τριών επιπέδων το γραφικό περιβάλλον της διεπαφής, το εκτελέσιμο κομμάτι του λογισμικού και ο χώρος που χρησιμοποιείται για την αποθήκευση των δεδομένων αναπτύσσονται και συντηρούνται ως ανεξάρτητες οντότητες σε ξεχωριστές πλατφόρμες. Το γραφικό περιβάλλον βρίσκεται στην πλευρά του πελάτη ενώ το εκτελέσιμο τμήμα του κώδικα και ο χώρος αποθήκευσης αναφέρονται στην πλευρά του διακομιστή.

Εκτός από τα συνήθη πλεονεκτήματα του μοντέλου πελάτη - διακομιστή, η αρχιτεκτονική πολλών επιπέδων επιτρέπει την αναβάθμιση οποιουδήποτε από τα επίπεδα, χωρίς να επηρεαστούν τα υπόλοιπα, ώστε να συμβαδίζουν με τις αλλαγές στην τεχνολογία.

Στις περισσότερες περιπτώσεις, το γραφικό περιβάλλον διεπαφής χρήστη εκτελείται σε ένα ή περισσότερα τερματικά τα οποία χρησιμοποιούν κάποια από τα συνήθη γραφικά περιβάλλοντα για την απεικόνιση. Το βασικό κομμάτι του λογισμικού που αποτελείται από μία ή περισσότερες οντότητες βρίσκεται σε κάποιο workstation ή application server, ενώ τα δεδομένα αποθηκεύονται σε ένα database server.

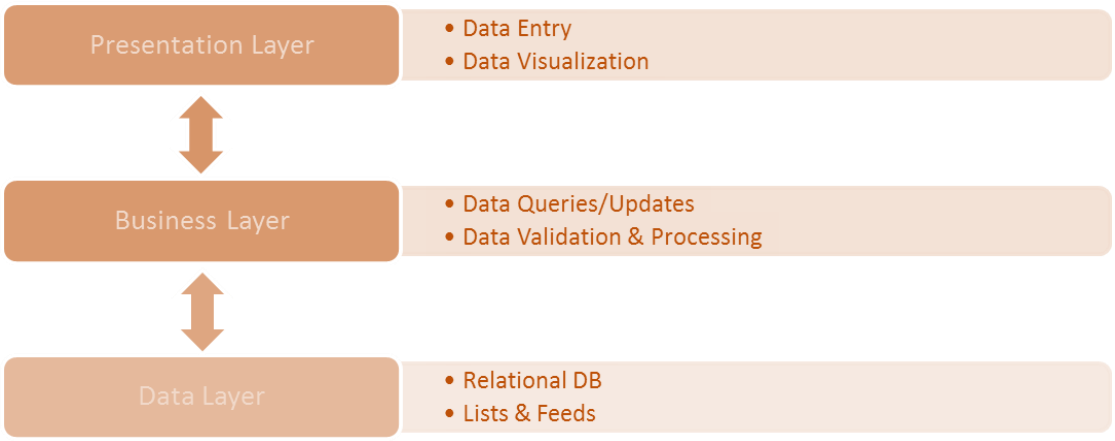
Αναλυτικά, τα επίπεδα της αρχιτεκτονικής τριών επιπέδων είναι:

- Παρουσίαση (Presentation Tier): Το επίπεδο της εφαρμογής που προβάλλεται στο χρήστη. Είναι υπεύθυνο για την απεικόνιση των πληροφοριών στην εκάστοτε πλατφόρμα του χρήστη μέσω γραφικών διεπαφών. Επικοινωνεί με τα υπόλοιπα επίπεδα τόσο αποστέλλοντας τα αιτήματα στο διακομιστή εφαρμογής του δεύτερου επιπέδου, όσο και λαμβάνοντας τα επιστρεφόμενα δεδομένα.
- Επεξεργασία Εφαρμογής (Application Tier): Σε αυτό το επίπεδο βρίσκεται το κύριο κομμάτι του λογισμικού. Ελέγχει τις λειτουργίες του λογισμικού μέσω των εισόδων από το πρώτο επίπεδο και επεξεργάζεται τα δεδομένα που βρίσκονται στο τρίτο επίπεδο. Ένα λεπτό σημείο του επιπέδου αυτού είναι το γεγονός ότι μπορεί να αποτελεί από μόνο του μια πολυεπίπεδη αρχιτεκτονική, με αποτέλεσμα να έχουμε τις γενικότερες αρχιτεκτονικές πολλών επιπέδων (n-tier architecture).
- Διαχείριση Δεδομένων (Data Tier): Το τελευταίο επίπεδο της αρχιτεκτονικής περιλαμβάνει τους διακομιστές δεδομένων. Είναι το επίπεδο στο οποίο αποθηκεύονται και από το οποίο ανακτώνται τα δεδομένα, ενώ η επεξεργασία τους γίνεται αποκλειστικά στο δεύτερο επίπεδο. Με αυτό τον τρόπο, τα δεδομένα διατηρούνται ουδέτερα και ανεξάρτητα από τους χρησιμοποιούμενους διακομιστές του δεύτερου επιπέδου. Με την δημιουργία αυτού του επιπέδου επιτυγχάνεται αποδοτικότητα και επεκτασιμότητα του συστήματος.

Πρόσθετα πλεονεκτήματα της αρχιτεκτονικής τριών επιπέδων είναι:

- Η επεξεργασία των δεδομένων γίνεται σε τοπικό επίπεδο, μεταξύ 2ου και 3ου επιπέδου, μεταφέροντας στο χρήστη μόνο το επιθυμητό αποτέλεσμα, ελαχιστοποιώντας έτσι την χρήση του δικτύου (για online εφαρμογές).
- Μέσω της χρήσης διαφορετικών διακομιστών για δεδομένα και εφαρμογές επιτυγχάνεται κατανομή του φόρτου εργασίας.
- Λόγω της ανεξαρτησίας μεταξύ των επιπέδων είναι εφικτή η χρήση διαφορετικής πλατφόρμας σε κάθε επίπεδο, ώστε να μεγιστοποιηθεί η απόδοση του συστήματος.

Η αρχιτεκτονική αυτή είναι κατάλληλη για εφαρμογές μεγάλης κλίμακας με εκατοντάδες ή χιλιάδες πελάτες, εφαρμογές που τόσο τα δεδομένα όσο και η εφαρμογή είναι ευμετάβλητες, καθώς και σε εφαρμογές στις οποίες ενοποιούνται δεδομένα από πολλές πηγές.



Σχήμα 4. Μοντέλο Αρχιτεκτονικής 3 Επιπέδων

2.3 Αρχιτεκτονική του συστήματος Uni-Food

Σύμφωνα με την ανάλυση που προηγήθηκε το λογισμικό Uni-Food θα ακολουθήσει ανάπτυξη αρχιτεκτονικής τριών επιπέδων.

Το πρώτο επίπεδο (επίπεδο παρουσίασης) θα αποτελείται από το σύνολο των γραφικών διεπαφών που θα προβάλλονται κάθε φορά στο χρήστη, ανάλογα με τον τρόπο σύνδεσης του χρήστη (Χρήστης ή Διαχειριστής). Για να εκτελεσθεί μια λειτουργία του προγράμματος οι χρήστες θα αλληλεπιδρούν με το γραφικό περιβάλλον, είτε έχουν συνδεθεί στο σύστημα ως Χρήστες είτε ως Διαχειριστές.

Το δεύτερο επίπεδο (επίπεδο εφαρμογής) θα περιλαμβάνει τους διακομιστές στους οποίους είναι εγκατεστημένο το σύστημα. Οι απομακρυσμένοι αυτοί διακομιστές θα είναι υπεύθυνοι για την επεξεργασία των δεδομένων, την εκτέλεση των αιτούμενων λειτουργιών από τα συστήματα ελέγχου και την αποστολή των αποτελεσμάτων στη συσκευή από την οποία έχει πρόσβαση ο χρήστης. Στο ίδιο επίπεδο βρίσκεται και ο εφεδρικός διακομιστής, ο οποίος τίθεται εν ενεργεία όταν σταματάει η λειτουργία του κανονικού ή βρίσκεται σε φάση ενημέρωσης.

Τέλος, το τρίτο επίπεδο (επίπεδο δεδομένων) θα περιλαμβάνει έναν ή περισσότερους ήδη υπάρχοντες διακομιστές, στους οποίους είναι αποθηκευμένες οι βάσεις δεδομένων.

Η αρχιτεκτονική αυτή προτιμήθηκε καθώς μπορεί να παρέχει ανεξαρτησία μεταξύ των επιπέδων, εύκολη συντήρηση και αναβάθμιση του συστήματος, αλλά και καλύτερη απόδοση του αναπτυσσόμενου λογισμικού σε σχέση με την απλή προσέγγιση πελάτη - διακομιστή.

3. Προτεινόμενη Αρχιτεκτονική Λογισμικού

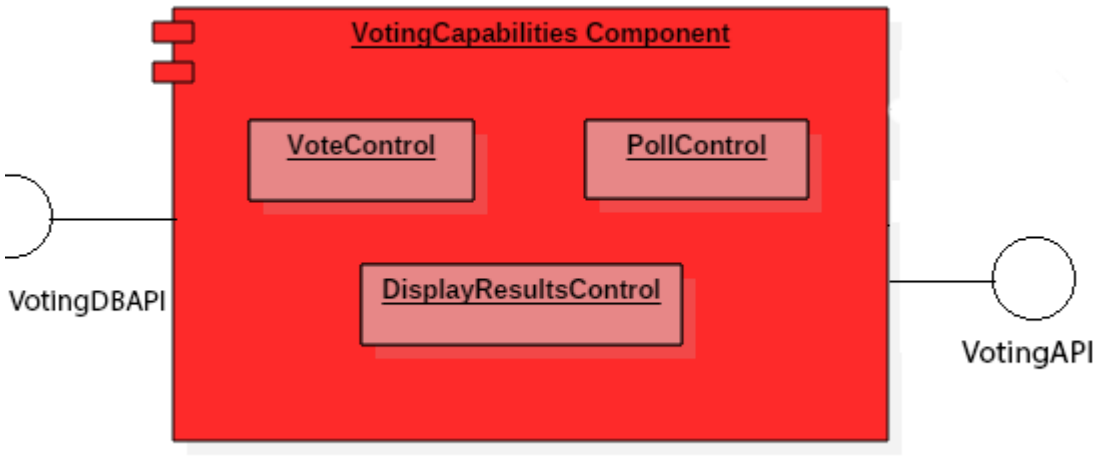
3.1 Αποδόμηση Συστήματος

Σε αυτό το μέρος περιγράφεται η αποδόμηση του συστήματος σε τμήματα (components), τα οποία παρέχουν μια προκαθορισμένη υπηρεσία και είναι επίσης ικανά να επικοινωνούν με άλλα τμήματα. Η επικοινωνία είναι δυνατή μέσω των διεπαφών (interfaces). Η αποδόμηση του συστήματος σε τμήματα προέκυψε από τα πακέτα λεξιλογίου σεναρίων και τα διαγράμματα κλάσεων, λαμβάνοντας υπ' όψιν τη μείωση της πολυπλοκότητας.

Πιο συγκεκριμένα ενοποιήθηκαν ξεχωριστά όλες οι διαφορετικές λειτουργίες του project αλλά και τα βασικά στοιχεία GUI, όπως και η επικοινωνία με την εξωτερική βάση δεδομένων.

3.1.1 VotingCapabilities Component

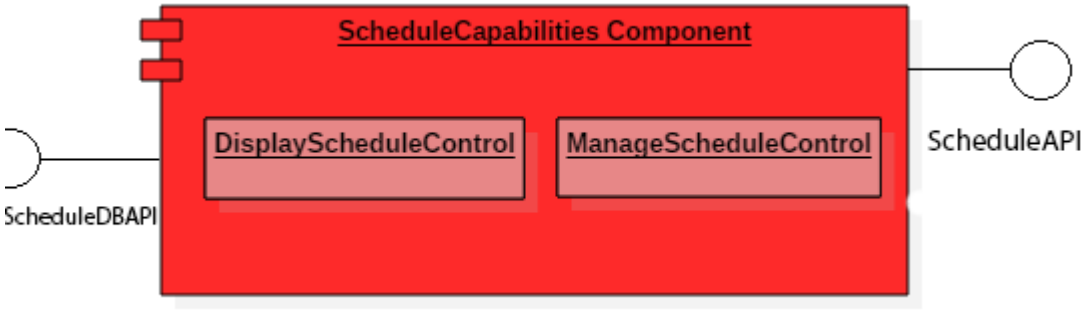
Το υποσύστημα αυτό περιλαμβάνει τη λειτουργικότητα που απαιτείται για καταχώρηση ψηφοφορίας από τον διαχειριστή, εμφάνιση ιστορικού ψηφοφοριών και καταχώρηση ψήφου από τον χρήστη. Επικοινωνεί με την εξωτερική βάση δεδομένων στην οποία στέλνει δεδομένα αλλά και από την οποία λαμβάνει δεδομένα. Επίσης, επικοινωνεί με το GUI Component, στο οποίο παρέχει πληροφορίες εμφάνισης περιερχομένου.



Σχήμα 5. VotingCapabilities Component

3.1.2 ScheduleCapabilities Component

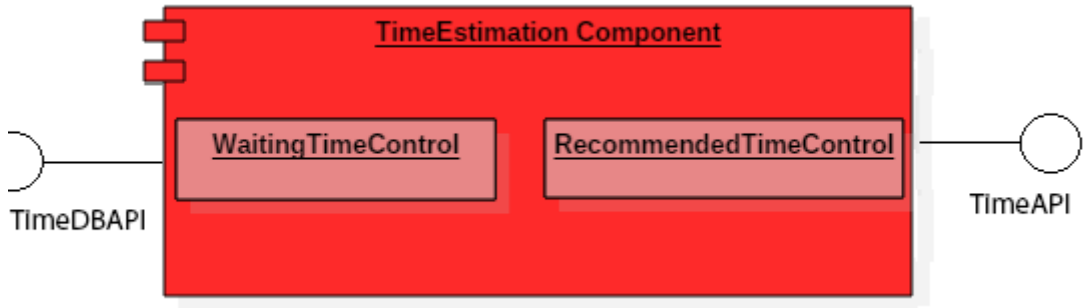
Το υποσύστημα αυτό περιλαμβάνει τη λειτουργικότητα που απαιτείται για την καταχώρηση και επεξεργασία του προγράμματος σίτισης από τον διαχειριστή καθώς και την εμφάνιση του εβδομαδιαίου προγράμματος σίτισης. Επικοινωνεί με την εξωτερική βάση δεδομένων στην οποία στέλνει δεδομένα αλλά και από την οποία λαμβάνει δεδομένα. Επίσης, επικοινωνεί με το GUI Component, στο οποίο παρέχει πληροφορίες εμφάνισης περιερχομένου.



Σχήμα 6. ScheduleCapabilities Component

3.1.3 TimeEstimation Component

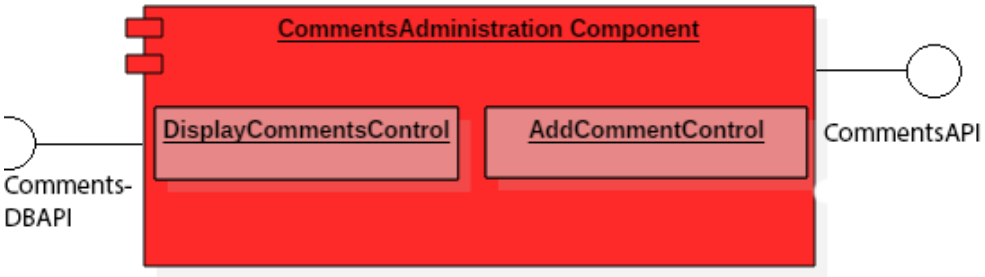
Το υποσύστημα αυτό περιλαμβάνει τη λειτουργικότητα που είναι απαραίτητη για την εμφάνιση του εκτιμωμένου χρόνου αναμονής στη λέσχη και της καλύτερης ώρας για να επισκεφτεί κανείς τη λέσχη. Επικοινωνεί με την εξωτερική βάση δεδομένων από την οποία λαμβάνει δεδομένα. Επίσης, επικοινωνεί με το GUI Component, στο οποίο παρέχει πληροφορίες εμφάνισης περιερχομένου.



Σχήμα 7. TimeEstimation Component

3.1.4 CommentsAdministration Component

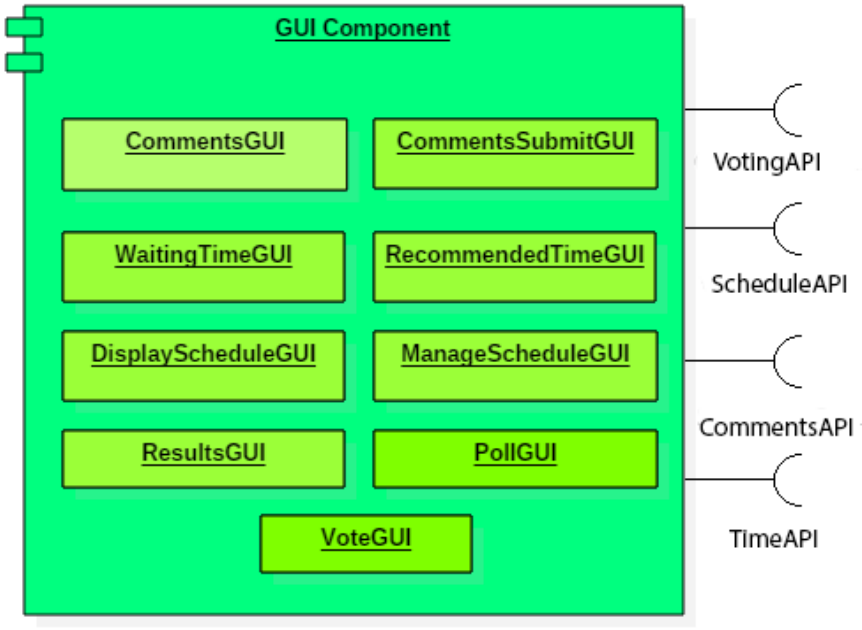
Το υποσύστημα αυτό περιλαμβάνει τη λειτουργικότητα που είναι απαραίτητη για την καταχώρηση ενός σχολίου από κάποιον χρήστη αλλά και την προβολή των σχολίων. Επικοινωνεί με την εξωτερική βάση δεδομένων στην οποία στέλνει δεδομένα αλλά και από την οποία λαμβάνει δεδομένα. Επίσης, επικοινωνεί με το GUI Component, στο οποίο παρέχει πληροφορίες εμφάνισης περιερχομένου



Σχήμα 8. CommentsAdministration Component

3.1.5 GUI Component

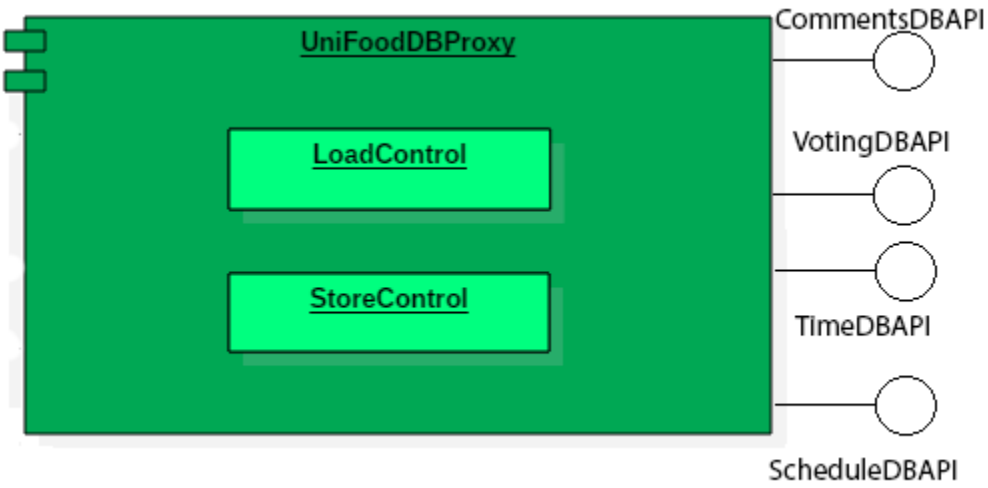
Αυτό το υποσύστημα είναι υπεύθυνο για την εμφάνιση της πληροφορίας στον χρήστη και στον διαχειριστή στα πλαίσια ενός γραφικού περιβάλλοντος. Επικοινωνεί με τα τέσσερα προηγούμενα υποσυστήματα ώστε να συλλέξει δεδομένα για εμφάνιση.



Σχήμα 9. GUI Component

3.1.6 UniFoodDBProxy Component

Το υποσύστημα αυτό αποτελεί την εξωτερική βάση με την οποία επικοινωνούν τα προηγούμενα υποσυστήματα.

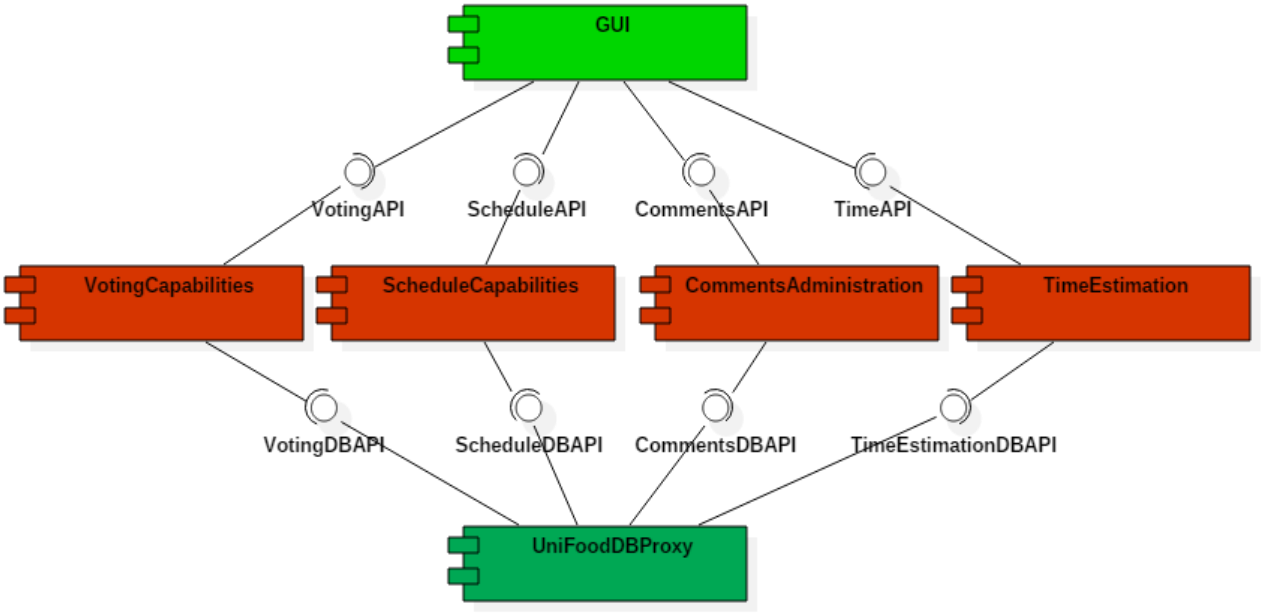


Σχήμα 10. UniFoodDBProxy Component

3.1.7 Σύνολο Υποσυστημάτων

Έχοντας αναλύσει τα τμήματα στα οποία χωρίσαμε το σύστημα και έχοντας αναφέρει το περιεχόμενο καθενός από αυτά, μπορεί να κατασκευαστεί το διάγραμμα τμημάτων (component diagram), που περιλαμβάνει πληροφορίες σχετικά με την επικοινωνία μεταξύ των τμημάτων, καθώς και τις μεταξύ τους εξαρτήσεις.

Η επικοινωνία μεταξύ δυο υποσυστημάτων επιτυγχάνεται μέσω των διεπαφών (API). Όταν ένα υποσύστημα προσφέρει μια διεπαφή τότε χρησιμοποιείται το σύμβολο του κύκλου, και όταν ένα υποσύστημα απαιτεί την παροχή χρησιμοποιείται το ημικύκλιο για τη δήλωση αυτή.



Σχήμα 11. Component Diagram

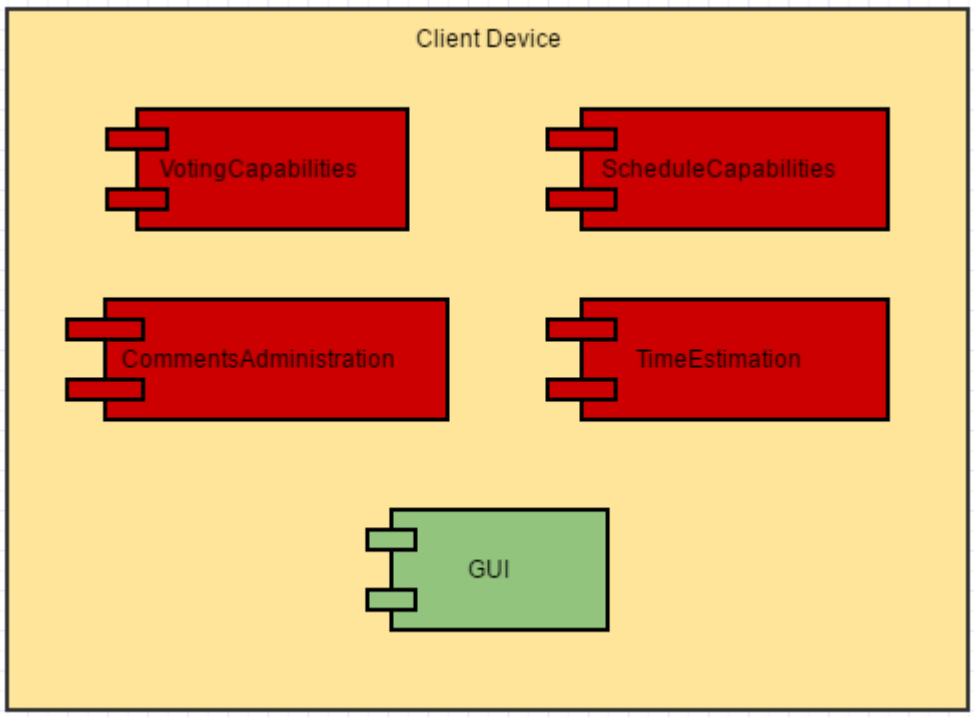
3.2 Απεικόνιση Υλικού/Λογισμικού

Για την πλήρη μοντελοποίηση του προς ανάπτυξη συστήματος είναι απαραίτητη η περιγραφή της υλοποίησης των υποσυστημάτων, δηλαδή ως λογισμικό ή υλικό, και η αντιστοίχιση του μοντέλου αντικειμένων στο αντίστοιχο υλικό και λογισμικό.

Συγκεκριμένα, η υλοποίηση του συστήματος, όπως παρουσιάζεται και από τη μοντελοποίηση των αντικειμένων, θα γίνει σε επίπεδο λογισμικού. Πρέπει, όμως, να δοθεί έμφαση στη σωστή υλοποίηση των διεπαφών που θα χρησιμοποιηθούν για την επικοινωνία του συστήματος με το εξωτερικό του περιβάλλον.

3.2.1 Client Device

Ο Client Device είναι ο κόμβος που επιτελεί τη λειτουργία του τερματικού για τον εκάστοτε χρήστη. Ουσιαστικά είναι ένα smartphone ή tablet, στο οποίο τρέχει η εφαρμογή. Η εφαρμογή είναι σε θέση να δώσει πληροφορίες στον χρήστη/διαχειριστή και να καταγράψει δεδομένα που αυτός εισάγει.

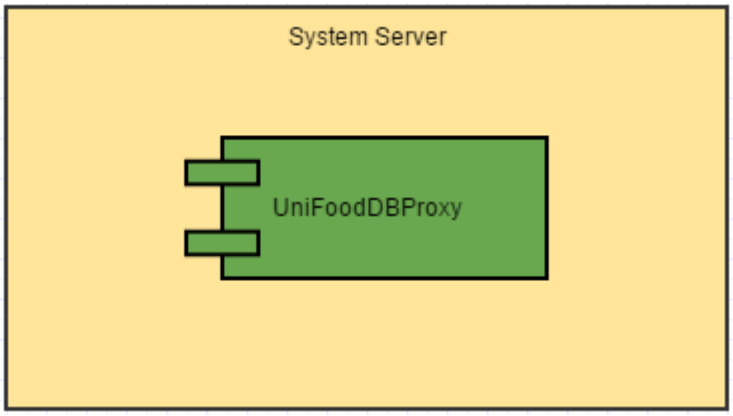


Σχήμα 12. Client Device

Ο κόμβος αυτός είναι υπεύθυνος για όλες τις βασικές λειτουργίες του Uni-Food και απλά επικοινωνεί όποτε είναι απαραίτητο με τον server για να αποστείλει ή να συλλέξει δεδομένα.

3.2.2 System Server

Ο κόμβος αυτός αποτελεί ουσιαστικά μία βάση δεδομένων υπεύθυνη για την αποθήκευση όλων των απαραίτητων για το σύστημα πληροφοριών. Βρίσκεται σε μια Cloud Computing πλατφόρμα (Microsoft Azure/Amazon WS) και είναι διαθέσιμη σε συνεχή βάση.

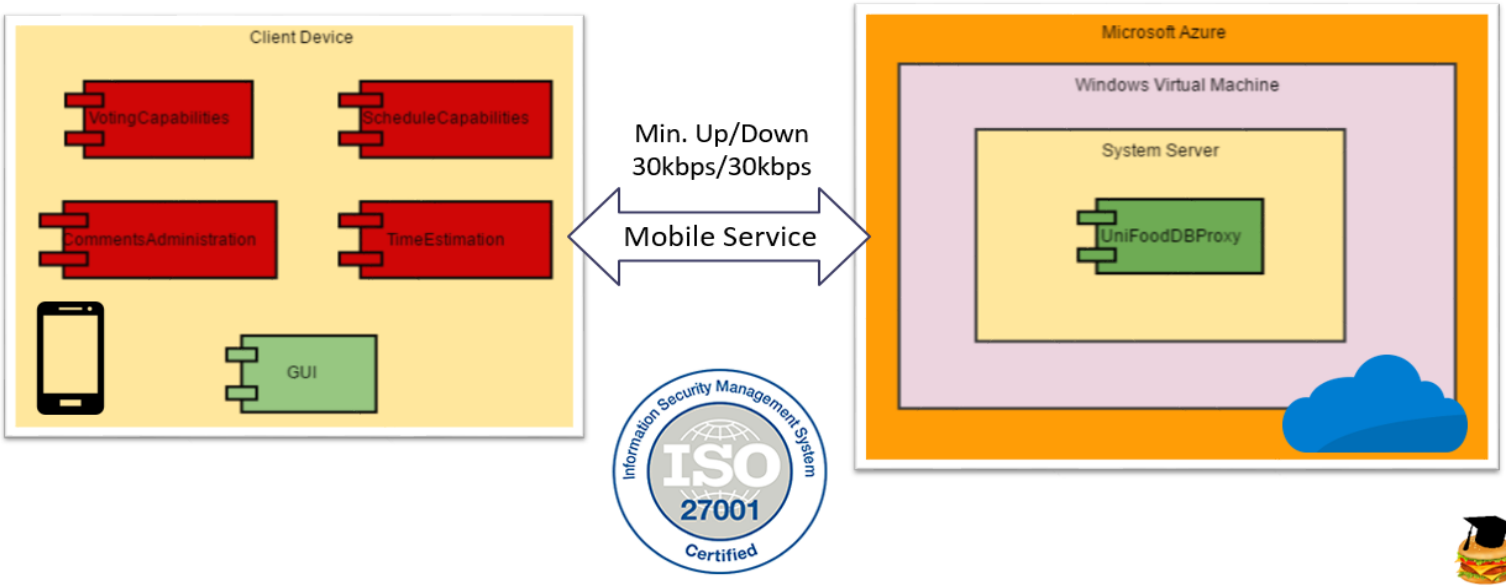


Σχήμα 13. System Server

3.2.3 Συνολικό Διάγραμμα Ανάπτυξης

Παρακάτω παρουσιάζεται ο τρόπος με τον οποίο είναι συνδεδεμένοι οι δύο κόμβοι του συστήματος και άρα η αρχιτεκτονική με την οποία δομείται όλο το σύστημα.

Σημαντικό είναι να τονιστεί ότι για την επικοινωνία Server-Client χρησιμοποιήθηκαν τα Mobile Services, μια τεχνολογικά που προσφέρει προσέγγιση υψηλότερου επιπέδου στον προγραμματιστή, μιας και προσφέρει ασφάλεια, ταχύτητα και σταθερότητα από την πρώτη στιγμή. Αυτό προϋποθέτει η βάση δεδομένων να βρίσκεται σε μια εικονική μηχανή Windows στο Microsoft Azure, το οποίο με τη σειρά του σημαίνει 99.9% up-time, ασφάλεια σύμφωνη με τα πρωτόκολλα που καθορίζονται από τους διεθνείς οργανισμούς διαδικτυακής ασφαλείας και τελευταία λέξη της τεχνολογίας όσον αφορά την Cloud Computing λύση που προσφέρουμε.



Σχήμα 14. Deployment Diagram

3.2.4 Υπολογιστικοί Πόροι

Όσον αφορά την mobile συσκευή ή tablet στην οποία θα τρέχει η εφαρμογή, οι υπολογιστικές απαιτήσεις δεν είναι ιδιαίτερα υψηλές καθώς οι αλγόριθμοι που εκτελούνται είναι βελτιστοποιημένοι. Μία mid προς low end σύγχρονη συσκευή mobile θα είναι περισσότερο από αρκετή. Μετά από δοκιμές στην εφαρμογή που ήδη αναπτύσσεται θέτουμε τα Minimum Requirements σε ένα κινητό/tablet με διπύρηνο επεξεργαστή με τουλάχιστον 1GB Ram και 20MB Rom διαθέσιμη.

Από την μεριά του server τώρα, μιας και αναφερόμαστε σε Mobile Services και Microsoft Azure δεν χρειάζεται να προκαθορίσουμε τις απαιτήσεις του συστήματος μιας και αυτό γίνεται αυτόματα scale ανάλογα με την κίνηση που αντιμετωπίζει. Υπολογίζουμε ότι μία συνδρομή της τάξης των 50 ευρώ τον μήνα θα είναι αρκετή.

3.3 Διαχείριση Μόνιμων Δεδομένων

Το παρόν σύστημα δομείται με βάση τα μοντέλα πελάτη - διακομιστή και τριών - επιπέδων. Πρέπει, πρακτικά να αποθηκεύει και να διαχειρίζεται τα δεδομένα του χρήστη, αλλά και δεδομένα που αφορούν ρυθμίσεις του συστήματος.

Λόγω του μικρού όγκου των δεδομένων που διαχειρίζεται η εφαρμογή, σε σχέση με τη σημερινή τεχνολογία, και για απλότητα στη σχεδίαση επιλέχθηκε η αποθήκευση των δεδομένων σε αρχεία. Τόσο το αρχείο των δεδομένων των μετρήσεων, όσο και το αρχείο των ρυθμίσεων έχουν δομημένη μορφή με σκοπό να είναι εύκολη η υλοποίηση της ανάγνωσης και της εγγραφής από και προς τη “μνήμη”.

3.4 Έλεγχος Πρόσβασης και Ασφάλεια

Όπως έχει προδιαγραφεί στο έγγραφο Απαιτήσεων Χρηστών, το σύστημα Uni-Food έχει δύο κατηγορίες χρηστών. Η μια κατηγορία είναι οι απλοί χρήστες που είναι οι φοιτητές και γενικά όλοι όσοι έχουν το δικαίωμα σίτισης στην ΠΦΛ. Η άλλη κατηγορία είναι ο διαχειριστής ο οποίος ορίζεται από την ΠΦΛ. Κάθε μια από τις παραπάνω κατηγορίες έχει διαφορετικά δικαιώματα και μπορεί να εκτελέσει διαφορετικές λειτουργίες. Αυτή η κατηγοριοποίηση είναι απαραίτητη, έτσι ώστε να παρέχεται το δικαίωμα εκτέλεσης συγκεκριμένων ενεργειών από κάθε τύπο χρήστη. Με αυτό τον τρόπο το σύστημα διασφαλίζει πως κανένας χρήστης δεν θα κάνει κάποια ενέργεια ενώ δεν έχει δικαίωμα να το κάνει.

Για την επίτευξη της αυθεντικότητας των χρηστών, πραγματοποιείται σύνδεση με τη βάση δεδομένων της ΠΦΛ, η οποία προσδιορίζει την κατηγορία και τα δικαιώματα του χρήστη. Έπειτα, το λογισμικό εμφανίζει το γραφικό περιβάλλον που αντιστοιχεί στην κατηγορία του χρήστη. Εάν ο χρήστης επιθυμεί να πραγματοποιήσει κάποια ενέργεια στο σύστημα, την επιλέγει μέσω των συντομεύσεων του γραφικού περιβάλλοντος. Κάθε κίνηση που γίνεται στο σύστημα καταγράφεται σε αρχεία τύπου log files.

Ένα ακόμα βασικό χαρακτηριστικό του συστήματος είναι η ασφάλεια και η προστασία που παρέχεται στους χρήστες. Με τον όρο ασφάλεια αναφερόμαστε σε τρία κυρίως σημεία:

- Εχεμύθεια (confidentiality): Καθιστά ασφαλείς τις προσωπικές πληροφορίες των χρηστών. Είτε αυτές αναφέρονται σε προσωπικά δεδομένα, είτε αναφέρονται σε ψηφοφορίες που γίνονται εντός του συστήματος, εξασφαλίζοντας με βεβαιότητα ότι πρόσβαση έχουν μόνο εξουσιοδοτημένα άτομα.
- Αξιοπιστία (integrity): Εξασφαλίζει ότι το πρόγραμμα και οι πληροφορίες τροποποιούνται μόνο με συγκεκριμένους και εξουσιοδοτημένους τρόπους.
- Διαθεσιμότητα (availability): Εξασφαλίζει ότι τα εξουσιοδοτημένα άτομα θα έχουν πρόσβαση στο σύστημα οποιαδήποτε στιγμή.

Προκειμένου να επιτευχθούν αυτοί οι στόχοι, πρέπει πρώτα να καθοριστεί πρέπει να καθοριστεί σε ποιες διεργασίες του συστήματος έχει πρόσβαση ο κάθε χρήστης. Για καλύτερη εποπτεία των δικαιωμάτων του κάθε χρήστη, δημιουργείται ο πίνακας πρόσβασης του συστήματος, οποίος παρουσιάζεται παρακάτω.

User	UserHomePageGUI	DisplayScheduleGUI	VoteGUI	WaitingTimeGUI
	clickOnDisplaySchedule()	dayClick()	clickOnSubmit()	clickOnOk()
	clickOnVote()		clickOnCancel()	
	clickOnWaitingTime()		setChoices()	
	clickOnRecommendedTime()			
	clickOnCommentsSubmit()			
	clickOnDisplayComments()			
	RecommendedTimeGUI	CommentsSubmitGUI	CommentsGUI	
	clickOnOk()	clickOnSubmit()	showComment()	
		clickOnCancel()	showRating()	
			chooseDate()	

Admin	AdminHomePageGUI	ManageScheduleGUI	PollGUI	ResultsGUI
	clickOnManageSchedule()	clickOnSubmit()	clickOnSubmit()	dayClick()
	clickOnPoll()	clickOnCancel()	clickOnCancel()	
	clickOnResults()	setMenu()		
	clickOnDisplayComments()			
	CommentsGUI			
	showComment()			
	showRating()			
	chooseDate()			

Σχήμα 15. Πίνακας πρόσβασης συστήματος

3.5 Γενικός έλεγχος λογισμικού

Το σύστημα Uni-Food χρησιμοποιείται από 2 κατηγορίες χρηστών, τους απλούς χρήστες (Users) και τον διαχειριστή (Admin), οι οποίοι μέσω των ενεργειών τους μπορούν να επιφέρουν μεταβολές σε περισσότερα από ένα στοιχεία του συστήματός. Οι αλλαγές αυτές πραγματοποιούνται από αντικείμενα ελέγχου, τα οποία είναι περισσότερα του ενός, γεγονός που προσδιορίζει ότι ο έλεγχος είναι κατανεμημένος. Ταυτόχρονα, τα ακολουθιακά διαγράμματα τα οποία φέρουν τη μορφή σκάλας και όχι πιρουνιού, έχει ως αποτέλεσμα την επιλογή του αποκεντρωμένου μοντέλου σχεδίασης για τον γενικό έλεγχο του συστήματος.

Η ακολουθία εκτέλεσης της κάθε λειτουργίας στο σύστημα έχει ως εξής:

1. Ο χρήστης μέσω του γραφικού περιβάλλοντος ενεργοποιεί μια λειτουργία.
2. Η λειτουργία εκκινεί μέσω οριακού αντικειμένου και μπορεί να συνεχίσει σε άλλο οριακό αντικείμενο έως ότου καταλήξει σε ένα αντικείμενο ελέγχου.
3. Το αντικείμενο ελέγχου αναγνωρίζει το αίτημα και εκκινεί τη διαδικασία εκτέλεσης του αιτήματος μέσω των entities ή προωθεί το αίτημα σε άλλο αντικείμενο ελέγχου.
4. Κατά τη διάρκεια της εκτέλεσης συχνά υπάρχει επικοινωνία με τη βάση δεδομένων (UniFoodDBProxy) για την ανάκτηση των απαραίτητων δεδομένων.

Παρόλο που και οι χρήστες αλλά και ο διαχειριστής προκαλούν αλλαγές σε διάφορα δεδομένα του συστήματος, παρόλα αυτά ο διαχειριστής δεν έχει δικαιώματα τροποποίησης σε δεδομένα στα οποία προκαλούν αλλαγές οι χρήστες, και το αντίθετο. Έτσι, δεν τίθεται απολύτως κανένα ζήτημα όσον αφορά το πρόβλημα του ταυτοχρονισμού δεδομένων μεταξύ διαχειριστή και χρηστών.

Όσον αφορά τον διαχειριστή, το σύστημα είναι σχεδιασμένο για πρόσβαση ενός μόνο διαχειριστή, κάτι που σημαίνει πως ούτε εδώ υπάρχει πρόβλημα ταυτοχρονισμού.

Τέλος, όσον αφορά τους χρήστες, το σύστημα είναι σχεδιασμένο για πρόσβαση πολλών χρηστών ταυτόχρονα, παρόλα αυτά τα δεδομένα που τροποποιούν οι χρήστες δεν επηρεάζονται από τα δεδομένα των υπόλοιπων χρηστών, οπότε ούτε και εδώ υπάρχει κάποιο πρόβλημα ταυτοχρονισμού.

Πιο αναλυτικά όλες οι διαδικασίες και η δυναμική τους συμπεριφορά, περιγράφονται στα διαγράμματα ακολουθιών που υπάρχουν στο έγγραφο των Απαιτήσεων Λογισμικού.

3.6 Οριακές Συνθήκες

Το σύστημα Uni-Food έχει ως κύριο σκοπό την εξασφάλιση της αξιοπιστίας των υπηρεσιών σε όλες του τις φάσεις και κατ' επέκταση στις οριακές του συνθήκες. Παρακάτω παρουσιάζονται οι βασικότερες περιπτώσεις των οριακών συνθηκών του συστήματος.

3.6.1 Εκκίνηση Εφαρμογής (start-up)

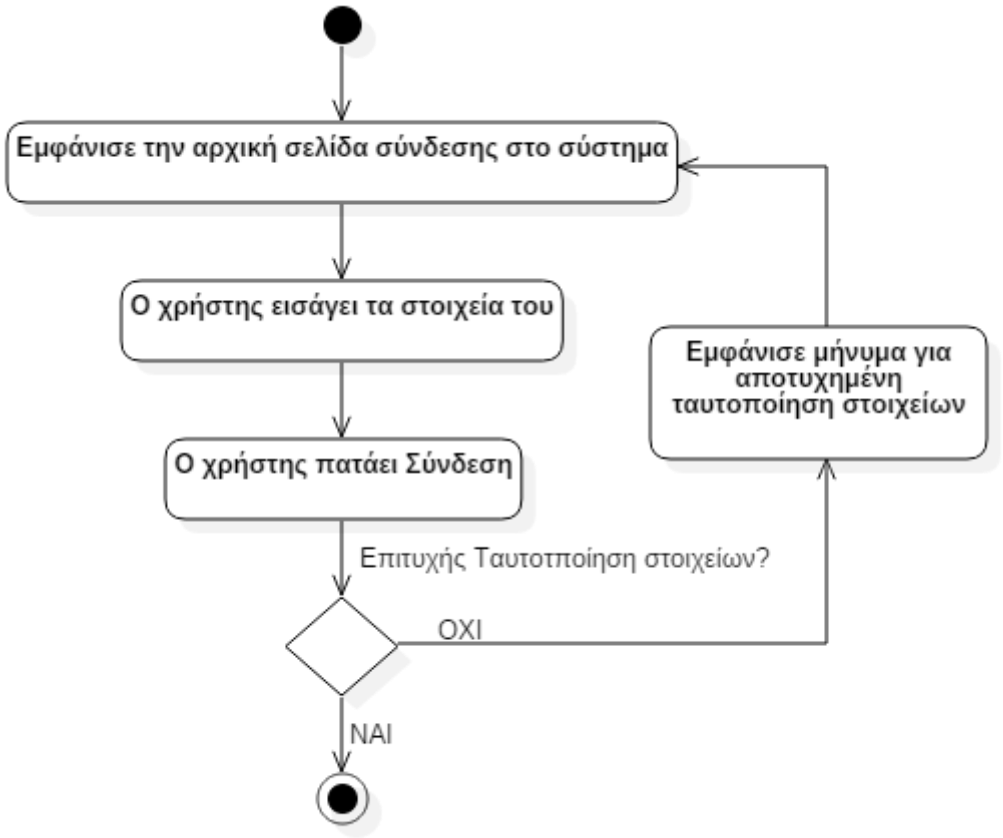
Η εφαρμογή Uni-Food θα εγκατασταθεί σε κάποια κινητή συσκευή smartphone από το αντίστοιχο application market (google play, windows store, apple store) ή μέσω του windows store από κάποιον υπολογιστή με λειτουργικό windows 10. Κατά την εκκίνηση ο χρήστης πρέπει να κάνει log in και ακολούθως γίνεται ταυτοποίηση των στοιχείων του μέσω της βάσης δεδομένων της ΠΦΛ. Για να λειτουργήσει σωστά η εφαρμογή και η επικοινωνία με τη βάση δεδομένων της ΠΦΛ, θα πρέπει να υπάρχει ενεργή σύνδεση στο διαδίκτυο, είτε αυτή είναι μέσω Wi-Fi είτε είναι μέσω κάποιου δικτύου κινητής τηλεφωνίας (π.χ. 4G). Σε περίπτωση που τα στοιχεία ταυτοποίησης είναι λάθος, τότε ο χρήστης καλείται να εισαγάγει ξανά τα στοιχεία του.

Σύντομη Περιγραφή:	Σε αυτό το Σενάριο Χρήσης περιγράφεται η διαδικασία κατά την οποία γίνεται εκκίνηση της εφαρμογής και σύνδεση στο σύστημα.
Πυροδότηση Δραστηριότητας:	Άνοιγμα της εφαρμογής από το γραφικό περιβάλλον της συσκευής.
Προϋπόθεση:	Η εφαρμογή να είναι εγκαταστημένη και η συσκευή να έχει ενεργή σύνδεση στο διαδίκτυο.

Βασική Ροή:		
Γραμμή	**Ενέργεια χρήστη συστήματος**	**Απάντηση Συστήματος**
1		Το σύστημα εμφανίζει το γραφικό περιβάλλον LogInGUI στο οποίο ο χρήστης ή ο διαχειριστής καλείται να εισαγάγει τα στοιχεία πρόσβασης του.
2	Ο χρήστης ή ο διαχειριστής γράφει τα στοιχεία του (username, password) και πατάει σύνδεση.	Το σύστημα επικοινωνεί με τη βάση δεδομένων για ταυτοποίηση των στοιχείων
Μετέπειτα κατάσταση:	Το σύστημα εμφανίζει την αρχική σελίδα του χρήστη (UserHomePageGUI) ή του διαχειριστή (AdminHomePageGUI) αντίστοιχα.	

Εναλλακτική Ροή (ΕΡ-1): Η ταυτοποίηση των στοιχείων δεν είναι επιτυχής.		
Εάν στη γραμμή 2 στη Βασική Ροή τα στοιχεία του χρήστη/διαχειριστή είναι λάθος με αποτέλεσμα η ταυτοποίηση των στοιχείων του να μην είναι επιτυχής.		
Γραμμή	**Ενέργεια χρήστη συστήματος**	**Απάντηση Συστήματος**
1		Το σύστημα εμφανίζει το μήνυμα που καλεί τον χρήστη/διαχειριστή να προσπαθήσει ξανά.
Το σενάριο συνεχίζει από γραμμή 1 της βασικής ροής.		

Διάγραμμα Δραστηριοτήτων:



Σχήμα 16. Ο χρήστης/διαχειριστής συνδέεται στο σύστημα

3.6.2 Τερματισμός Εφαρμογής (Shutdown)

Η εφαρμογή Uni-Food τερματίζει εάν πιεστεί το πλήκτρο «Πίσω» (αναλόγως τη συσκευή) από την αρχική οθόνη της εφαρμογής. Με τον τερματισμό της η εφαρμογή αποδεσμεύει όλους τους πόρους που είχε δεσμεύσει κατά την λειτουργία της και αποσυνδέει τον χρήστη από το σύστημα.

Σύντομη Περιγραφή:	Σε αυτό το Σενάριο Χρήσης περιγράφεται η διαδικασία κατά την οποία τερματίζεται η εφαρμογή.
Πυροδότηση Δραστηριότητας:	Πάτημα του αντίστοιχου πλήκτρου «Πίσω» (αναλόγως της συσκευής).
Προϋπόθεση:	Ο χρήστης/διαχειριστής να βρίσκεται στην αντίστοιχη αρχική σελίδα της εφαρμογής (UserHomePageGUI/AdminHomePageGUI).

Βασική Ροή:		
Γραμμή	**Ενέργεια χρήστη συστήματος**	**Απάντηση Συστήματος**
1	Ο χρήστης ή ο διαχειριστής πατάει το πλήκτρο «πίσω» της συσκευής.	Το σύστημα αποδεσμεύει όλους τους πόρους που είχε δεσμεύσει κατά την λειτουργία της.
		Το σύστημα αποσυνδέει τον χρήστη.
2		Το σύστημα τερματίζει την εφαρμογή.
Μετέπειτα κατάσταση:	Η εφαρμογή τερματίζεται και ο χρήστης/διαχειριστής μεταβαίνουν στο γραφικό περιβάλλον του λειτουργικού συστήματος της συσκευής.	

3.6.3 Διακοπή Σύνδεσης της Εφαρμογής

Πολλές φορές υπάρχει η περίπτωση διακοπής της σύνδεσης του χρήστη, για τεχνικούς ή άλλους λόγους. Εάν αυτό συμβεί, τότε αρχικά το σύστημα ενημερώνει τον χρήστη για την διακοπή σύνδεσης, με ένα μήνυμα pop-up. Εάν η διακοπή συμβεί κατά την εκτέλεση κάποιας ενέργειας η οποία απαιτεί την επικοινωνία με τη βάση δεδομένων, τότε το σύστημα αποθηκεύει την μέχρι τότε πρόοδο του χρήστη, και λόγω των αλλοιωμένων πακέτων που καταφθάνουν στους servers της βάσης δεδομένων, η ενέργεια αυτή απορρίπτεται ενημερώνοντας τον χρήστη ότι η ενέργεια του έχει ακυρωθεί. Μόλις επανέλθει η σύνδεση ο χρήστης συνεχίζει κανονικά από εκεί που έμεινε. Στην περίπτωση που η διακοπή σύνδεσης συμβεί κατά τη διάρκεια που δεν εκτελείται κάποια ενέργεια, τότε ο χρήστης συνεχίζει από το γραφικό περιβάλλον που έμεινε μέχρι να επανέλθει η σύνδεση.

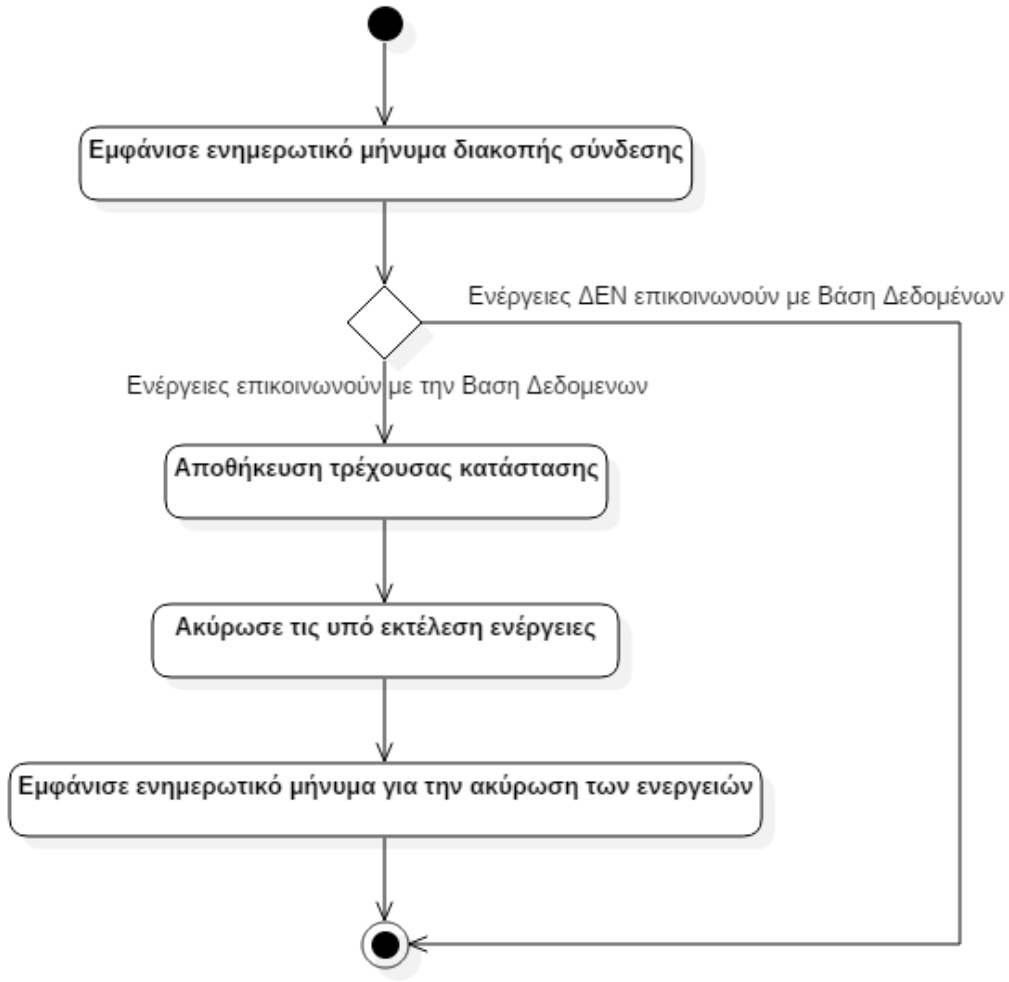
Σύντομη Περιγραφή:	Σε αυτό το Σενάριο Χρήσης περιγράφεται η διαδικασία κατά την οποία συμβαίνει μια διακοπή της σύνδεσης της Εφαρμογής.
Πυροδότηση Δραστηριότητας:	Διακοπή Σύνδεσης της Εφαρμογής.
Προϋπόθεση:	Ομαλή Λειτουργία του Συστήματος με ενεργή σύνδεση.

Βασική Ροή: Κατά τη διάρκεια της ομαλής λειτουργίας του συστήματος, συμβαίνει διακοπή σύνδεσης της Εφαρμογής.		
Γραμμή	**Ενέργεια χρήστη συστήματος**	**Απάντηση Συστήματος**
1		Το σύστημα ενημερώνει τον χρήστη για την διακοπή σύνδεσης της εφαρμογής.
2		Το σύστημα αποθηκεύει την τρέχουσα κατάσταση της εφαρμογής.
3		Το σύστημα ακυρώνει τις ενέργειες του χρήστη που απαιτούν επικοινωνία με τη βάση δεδομένων και βρίσκονται υπό εκτέλεση.
4		Το σύστημα ενημερώνει τον χρήστη ότι οι ενέργειες του έχουν ακυρωθεί.

Μετέπειτα κατάσταση:	Μετά την επαναφορά της σύνδεσης της εφαρμογής, ο χρήστης συνεχίζει από το γραφικό περιβάλλον στο οποίο ενεργούσε αμέσως πριν συμβεί η διακοπή σύνδεσης.

Εναλλακτική Ροή (ΕΡ-1): Δεν εκτελείται καμία ενέργεια με την Βάση Δεδομένων.		
Εάν στη γραμμή 2 στη Βασική Ροή δεν εκτελείται καμία ενέργεια που απαιτεί επικοινωνία με τη Βάση Δεδομένων.		
Γραμμή	**Ενέργεια χρήστη συστήματος**	**Απάντηση Συστήματος**
1		Το σύστημα παραμένει στο ίδιο Γραφικό περιβάλλον.
Το σενάριο χρήσης τερματίζει.		

Διάγραμμα Δραστηριοτήτων:



Σχήμα 17. Διακοπή σύνδεσης της Εφαρμογής

3.6.4 Διακοπή Σύνδεσης της Βάσης Δεδομένων

Όπως και στην εφαρμογή, έτσι και στη βάση δεδομένων πολλές φορές υπάρχει η πιθανότητα διακοπής της σύνδεσης της. Αυτό μπορεί να συμβεί για διάφορους λόγους, όπως π.χ. η διακοπή τροφοδοσίας στις εγκαταστάσεις του server. Εάν αυτό συμβεί κατά την εκτέλεση κάποιας ενέργειας η οποία απαιτεί την επικοινωνία με τη βάση δεδομένων, τότε λόγω των αλλοιωμένων πακέτων που καταφθάνουν στους servers της βάσης δεδομένων, η ενέργεια αυτή θα απορρίπτεται και θα του εμφανίζεται το αντίστοιχο μήνυμα. Ωστόσο, η μέχρι τότε πρόοδος του χρήστη θα αποθηκεύεται στην εφαρμογή ούτως ώστε ο χρήστης να μπορεί να συνεχίζει κανονικά από εκεί που έμεινε όταν η σύνδεση με τη βάση δεδομένων επανέλθει. Αν η διακοπή σύνδεσης συμβεί κατά την διάρκεια που δεν εκτελείται καμία ενέργεια, τότε ο χρήστης συνεχίζει κανονικά.

Παρόλα αυτά, για την αποφυγή τέτοιων προβλημάτων, οι βάσεις δεδομένων θα βρίσκονται σε δύο ξεχωριστούς αλλά πανομοιότυπους servers, και ο ένας θα κρατιέται ως εφεδρικός (backup). Όταν συμβεί μια διακοπή σύνδεσης στον τρέχον server τότε θα εκκινεί αυτόματα ο εφεδρικός. Επίσης, οι servers θα βρίσκονται σε διαφορετικές κτηριακές εγκαταστάσεις ούτως ώστε σε περίπτωση διακοπής τροφοδοσίας, να μην επηρεάζονται και οι δύο servers. Έτσι, με αυτούς τους τρόπους μειώνονται δραματικά οι πιθανότητες διακοπής της σύνδεσης της βάσης δεδομένων.

Σύντομη Περιγραφή:	Σε αυτό το Σενάριο Χρήσης περιγράφεται η διαδικασία κατά την οποία συμβαίνει μια διακοπή της σύνδεσης της Βάσης Δεδομένων.
Πυροδότηση Δραστηριότητας:	Διακοπή Σύνδεσης της Βάσης Δεδομένων.
Προϋπόθεση:	Ομαλή Λειτουργία του Συστήματος.

Βασική Ροή: Κατά τη διάρκεια της ομαλής λειτουργίας του συστήματος, συμβαίνει διακοπή σύνδεσης της Βάσης Δεδομένων.

Γραμμή	**Ενέργεια χρήστη συστήματος**	**Απάντηση Συστήματος**
1		Το σύστημα αποθηκεύει την τρέχουσα κατάσταση της εφαρμογής.
2		Το σύστημα ακυρώνει τις ενέργειες του χρήστη που βρίσκονται υπό εκτέλεση και απαιτούν επικοινωνία με τη βάση δεδομένων.
3		Το σύστημα ενημερώνει τον χρήστη για την διακοπή σύνδεσης της Βάσης Δεδομένων.
4		Το σύστημα ενημερώνει τον χρήστη ότι οι ενέργειες του έχουν ακυρωθεί.
Μετέπειτα κατάσταση:	Μετά την επαναφορά της σύνδεσης της εφαρμογής, ο χρήστης συνεχίζει από το γραφικό περιβάλλον στο οποίο ενεργούσε αμέσως πριν συμβεί η διακοπή σύνδεσης.	

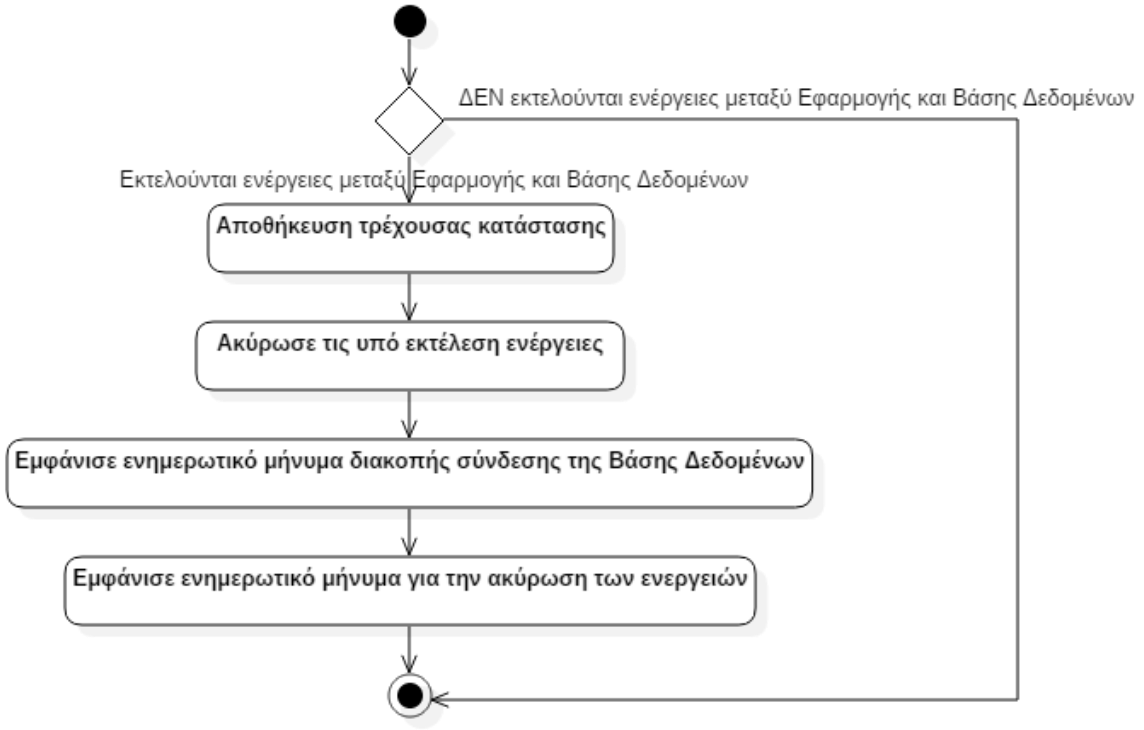
Εναλλακτική Ροή (ΕΡ-1): Δεν εκτελείται καμία ενέργεια με την Βάση Δεδομένων.

Εάν στη γραμμή 1 στη Βασική Ροή δεν εκτελείται καμία ενέργεια που απαιτεί επικοινωνία με τη Βάση Δεδομένων.

Γραμμή	**Ενέργεια χρήστη συστήματος**	**Απάντηση Συστήματος**
1		Το σύστημα παραμένει στο ίδιο Γραφικό περιβάλλον.

Το σενάριο χρήσης τερματίζει.

Διάγραμμα Δραστηριοτήτων:



Σχήμα 18. Διακοπή της σύνδεσης της Βάσης Δεδομένων

3.6.5 Σφάλματα Λογισμικού

Σφάλματα λογισμικού μπορούν να συμβούν για δύο λόγους, είτε στην περίπτωση που εμφανιστεί ένα σφάλμα στην εφαρμογή είτε συμβεί ένα σφάλμα κατά την σύνδεση της εφαρμογής και την επικοινωνία της με τη βάση δεδομένων.

Σε κάθε περίπτωση σφάλματος, το σύστημα κάνει τρείς βασικές λειτουργίες. Αρχικά αποθηκεύεται η τρέχουσα κατάσταση της εφαρμογής και γίνεται μια απλή αναφορά στο στοιχείο που προκάλεσε το σφάλμα έτσι ώστε να εξαλειφθεί ο λόγος για τον οποίο εμφανίστηκε. Ακολούθως ενημερώνει τον χρήστη για το σφάλμα που συνέβη εμφανίζοντας το ανάλογο μήνυμα. Στην συνέχεια γίνεται αυτόματη επανεκκίνηση του συστήματος. Τέλος η εφαρμογή επιστρέφει εκ νέου στην κατάσταση που βρισκόταν πριν εμφανιστεί το σφάλμα, προσπαθώντας να βρει έναν εναλλακτικό τρόπο εκτέλεσης της ενέργεια που προκάλεσε το σφάλμα.

Σύντομη Περιγραφή:	Σε αυτό το Σενάριο Χρήσης περιγράφεται η διαδικασία κατά την οποία συμβαίνει ένα σφάλμα Λογισμικού.
Πυροδότηση Δραστηριότητας:	Εμφάνιση Σφάλματος.
Προϋπόθεση:	Ομαλή Λειτουργία του Συστήματος

Βασική Ροή: Κατά τη διάρκεια της ομαλής λειτουργίας του συστήματος, συμβαίνει κάποιο σφάλμα λογισμικού.		
Γραμμή	**Ενέργεια χρήστη συστήματος**	**Απάντηση Συστήματος**
1		Το σύστημα αποθηκεύει την τρέχουσα κατάσταση της εφαρμογής.
3		Το σύστημα ενημερώνει τον χρήστη για την εμφάνιση σφάλματος.
4		Το σύστημα επανεκκινεί την εφαρμογή.
Μετέπειτα κατάσταση:	Μετά την επανεκκίνηση της εφαρμογής, ο χρήστης συνεχίζει από το γραφικό περιβάλλον στο οποίο ενεργούσε αμέσως πριν συμβεί το σφάλμα λογισμικού.	

3.6.6 Ενημέρωση Λογισμικού

Κατά καιρούς η ομάδα ανάπτυξης και συντήρησης της εφαρμογής θα προβαίνει σε βελτιώσεις, με αποτέλεσμα την ύπαρξη καινούριων εκδόσεων με ανανεωμένα ή ακόμα και καινούρια χαρακτηριστικά. Παρουσιάζεται λοιπόν, η ανάγκη να ενημερώνεται το λογισμικό του συστήματος για αυτές τις αλλαγές. Είναι προφανές ότι για όσο χρονικά διάστημα διαρκεί η κάθε ενημέρωση, η σύνδεση των χρηστών στο σύστημα θα είναι αδύνατη. Παρόλα αυτά, όλη η ανάπτυξη και η συντήρηση της εφαρμογής θα ελέγχεται σε emulators, ούτως ώστε ο χρόνος στον οποίο θα είναι αδύνατη η σύνδεση των χρηστών στο σύστημα, να είναι μηδαμινός.

3.6.7 Υπερφόρτωση Συστήματος

Σε περιπτώσεις όπου συνδέονται πάρα πολλοί χρήστες στο σύστημα, ή όταν πολλοί χρήστες προσπαθούν να εκτελέσουν ταυτόχρονα κάποιες ενέργειες, υπάρχει η πιθανότητα υπερφόρτωσης του συστήματος. Σε μια τέτοια περίπτωση, για να αποφευχθεί η κατάρρευσης του συστήματος αλλά και για να επιτευχθεί μια λύση με τις λιγότερες δυνατές επιπτώσεις, το σύστημα θα μειώνει αυτόματα την ποιότητα των παρεχόμενων υπηρεσιών προσφέροντας μόνο τα απαραίτητα. Παρόλα αυτά η υπερφόρτωση συστήματος είναι πιθανόν να μην συμβεί, εφόσον ο αριθμός των δικαιούχων σίτισης είναι γνωστός και η ομάδα ανάπτυξης θα έχει φροντίσει από πριν ούτως ώστε το σύστημα να είναι σε θέση να υποδεχτεί τον μέγιστο δυνατό αριθμό χρηστών.

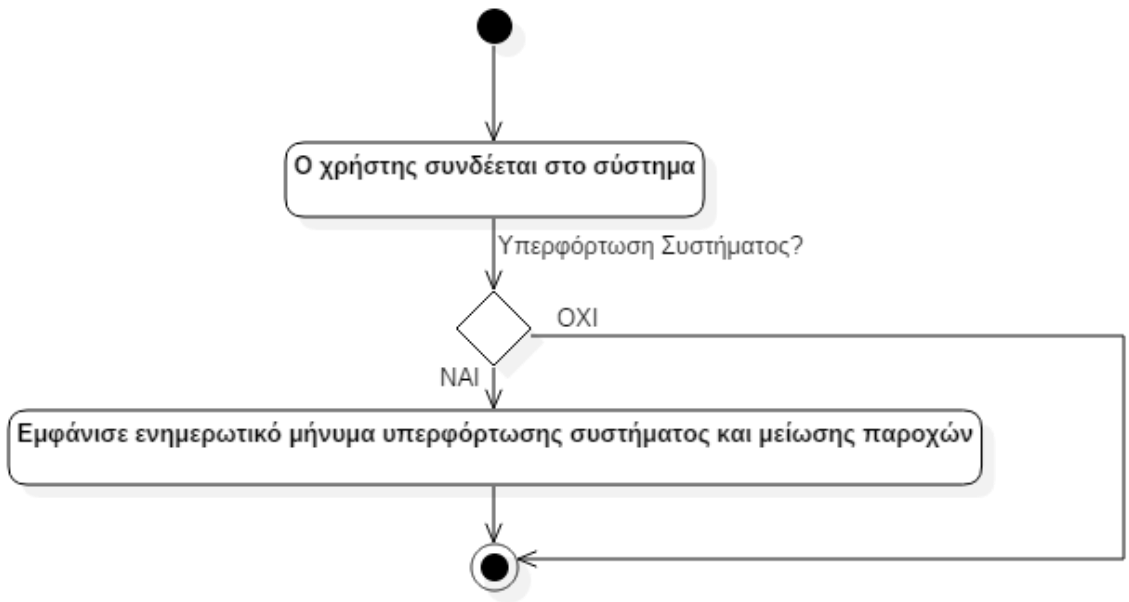
Σύντομη Περιγραφή:	Σε αυτό το Σενάριο Χρήσης περιγράφεται η διαδικασία κατά την οποία συμβαίνει Υπερφόρτωση του Συστήματος.
Πυροδότηση Δραστηριότητας:	Σύνδεση πολλών χρηστών ταυτόχρονα.
Προϋπόθεση:	Ομαλή Λειτουργία του Συστήματος.

Βασική Ροή: Κατά τη διάρκεια της ομαλής λειτουργίας του συστήματος, συνδέονται πολλοί χρήστες με αποτέλεσμα την υπερφόρτωση του συστήματος.		
Γραμμή	**Ενέργεια χρήστη συστήματος**	**Απάντηση Συστήματος**

1	Ο χρήστης συνδέεται στο σύστημα.	Το σύστημα ενημερώνει τον χρήστη για την υπερφόρτωση του συστήματος και ότι θα μειώσει τις παρεχόμενες υπηρεσίες.
3		Το σύστημα μειώνει τις παρεχόμενες υπηρεσίες.
Μετέπειτα κατάσταση:	Ο χρήστης συνεχίζει από το γραφικό περιβάλλον UserHomePageGUI, με τις διάφορες επιλογές που του παρέχει το σύστημα.	

Εναλλακτική Ροή (ΕΡ-1): Δεν εκτελείται καμία ενέργεια με την Βάση Δεδομένων.		
Εάν στη γραμμή 1 στη Βασική Ροή το σύστημα δεν υπερφορτωθεί.		
Γραμμή	**Ενέργεια χρήστη συστήματος**	**Απάντηση Συστήματος**
1		Το σύστημα παραμένει στο ίδιο Γραφικό περιβάλλον.
Το σενάριο χρήσης τερματίζει.		

Διάγραμμα Δραστηριοτήτων:



Σχήμα 19. Υπερφόρτωση του Συστήματος

3.6.8 SQL Injections

Τα SQL Injections θεωρούνται ως μια από τις σοβαρότερες απειλές για τις εφαρμογές ιστού ή γενικότερα τα συστήματα που επικοινωνούν με βάσεις δεδομένων. Για να εκμεταλλευτεί ο επιτιθέμενος μια αδυναμία του SQL κώδικα πρέπει να βρει μια παράμετρο που η εφαρμογή περνά σε μια βάση δεδομένων. Ενσωματώνοντας προσεκτικά κακόβουλες SQL εντολές στο περιεχόμενο της παραμέτρου, ο επιτιθέμενος μπορεί να εξαπατήσει την εφαρμογή, ώστε να αποστείλει μια κακόβουλη δήλωση, αντί της πρωτότυπης, στις βάσεις δεδομένων. Αυτές οι επιθέσεις δεν είναι δύσκολο να κατασκευαστούν και οι συνέπειες είναι ιδιαίτερα καταστροφικές, δεδομένου ότι ένας επιτιθέμενος μπορεί να εξάγει, να αλλοιώσει ή και να καταστρέψει το περιεχόμενο των βάσεων δεδομένων. Η πρωταρχική

αιτία των SQL Injections είναι ο ανεπαρκής έλεγχος της εισόδου. Επομένως, μια απλή λύση για την εξάλειψη αυτών των ευπαθειών είναι να εφαρμοστούν κατάλληλες αμυντικές πρακτικές κωδικοποίησης.

Τα Injections του SQL κώδικα μπορούν να εκτελεστούν με την έγχυση εντολών σε αλφαριθμητικές ή αριθμητικές παραμέτρους. Ακόμα και ένας απλός έλεγχος τέτοιων εισόδων μπορεί να αποτρέψει πολλές επιθέσεις. Για παράδειγμα, στην περίπτωση των αριθμητικών παραμέτρων μπορούν απλά να απορριφτούν όλες οι είσοδοι που περιέχουν χαρακτήρες αντί για αριθμούς. Συχνά πολλοί προγραμματιστές παραλείπουν αυτού του είδους ελέγχους, επειδή η είσοδος των χρηστών είναι σχεδόν πάντα εκφρασμένη υπό μορφή αλφαριθμητικής ακολουθίας χαρακτήρων, ανεξάρτητα από το περιεχόμενό της ή την προτεινόμενη χρήση.

4. Προδιαγραφές Τμηματικού Σχεδιασμού (Component Design Specifications)

Σε αυτό το τμήμα του εγγράφου αναφέρονται το πρότυπα σχεδίασης που χρησιμοποιήθηκαν. Πιο συγκεκριμένα, στις επόμενες παραγράφους περιγράφονται προβλήματα που εντοπίστηκαν κατά το σχεδιασμό και πώς χρησιμοποιήθηκαν δοκιμασμένες λύσεις από τη βιβλιογραφία, για την επίλυση τους.

4.1 Πρότυπα Σχεδιασμού που υιοθετήθηκαν

Η λειτουργία του συστήματος προκύπτει από την συνεργασία αντικειμένων που έχουν προκύψει από τα αντικειμενοστραφή περιβάλλοντα ανάπτυξης που χρησιμοποιούνται. Κατά την ανάπτυξη νέων λειτουργιών η εμφάνιση νεών προβλημάτων είναι αναπόφευκτη τα οποία ωστόσο σε πολλές περιπτώσεις έχουν ήδη προκύψει πολλαπλώς σε παρόμοια προβλήματα και έχουν επιλυθεί για τα παρόμοια προβλήματα. Αυτές ακριβώς οι λύσεις των συχνά εμφανιζόμενων προβλημάτων ονομάζονται πρότυπα σχεδίασης.

Τα πρότυπα σχεδίασης αποτελούν ουσιαστικά έξυπνες, σφαιρικές και παραμετροποιήσιμες λύσεις που έχουν σκεφτεί κάποιοι και μπορούν να επιλύσουν προβλήματα πολλών περισσοτέρων. Το πρότυπο σχεδίασης είναι συνήθως η πληρέστερη λύση από το σύνολο των υπαρχουσών λύσεων ενώ όπως είναι λογικό η τεκμηρίωσή του είναι τόσο απαραίτητη όσο και δύσκολη. Ένα καλό πρότυπο σίγουρα έχει τα παρακάτω χαρακτηριστικά:

- ***Δίνει λύση σε ένα υπάρχον πρόβλημα****:* Η λύση αυτή θα πρέπει να είναι δοκιμασμένη, για να θεωρείται αποτελεσματική.
- ***Είναι αφηρημένο****:* Δε θα πρέπει να περιγράφει μια συγκεκριμένη λύση, αλλά ένα σύνολο παρόμοιων προβλημάτων, χωρίς όμως να καταλήγει να παρουσιάζει απλώς μια γενικότερη στρατηγική για την επίλυση του προβλήματος.
- ***Έχει οργανωτική δομή****:* Με άλλα λόγια, κάνει το σύστημα περισσότερο δομημένο, κάνοντας πιο ξεκάθαρη τη λογική σχεδίασης του συστήματος.

4.1.1 Δομικά Πρότυπα

Τα δομικά πρότυπα μειώνουν την σύζευξη ανάμεσα σε δύο ή περισσότερες κλάσεις ενώ κάνουν και ευκολότερη την πιθανή μελλοντική επέκταση του έργου.

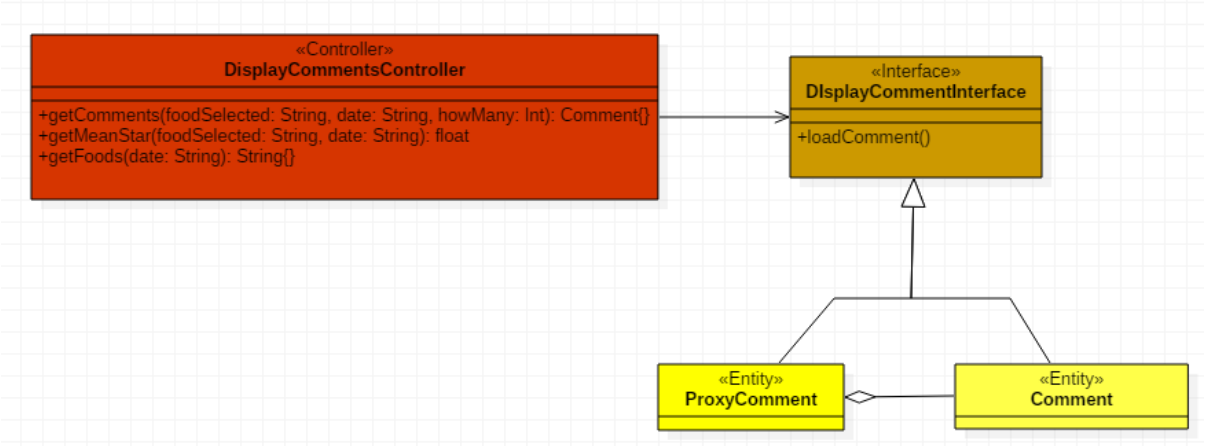
Όνομα προτύπου: Proxy

Κατηγορία: Δομικά πρότυπα

Πρόβλημα που αντιμετωπίστηκε: όταν ο χρήστης επιλέγει να δει παλαιότερα σχόλια ζητείται από τον server να επιστρέψει μία λίστα/ομάδα από σχόλια χρηστών. Η λίστα αυτή περιέχει ένα πλήθος από σχόλια τα οποία δεν γίνονται εμφανή στην οθόνη χωρίς ο χρήστης να κάνει "scroll down" και μπορεί ποτέ να μην φτάσει να διαβάσει όλα τα σχόλια. Για τον λόγο αυτό χρησιμοποιούμε το πρότυπο proxy ώστε να μειώσουμε τον όγκο των δεδομένων που πρέπει να λάβει ο client βελτιώνοντας έτσι την απόδοση του συστήματός μας.

Υποσυστήματα που αναδιοργανώθηκαν: Για την επίλυση του προβλήματος αναδιοργανώθηκε το πακέτο CommentsAdministration.

Σημείωση*: Στο σχήμα περιλαμβάνονται μόνο οι αλλαγές που έγιναν στα υποσυστήματα και όχι ολόκληρα τα υποσυστήματα μετά την αλλαγή.*



Σχήμα 20. Πρότυπο Proxy

4.1.2 Πρότυπα Συμπεριφοράς

Τα πρότυπα συμπεριφοράς χρησιμοποιούνται με σκοπό να χαρακτηρίσουν σύνθετες ροές ελέγχου και πληροφοριών οι οποίες είναι δύσκολο να παρακολουθηθούν κατά την εκτέλεση.

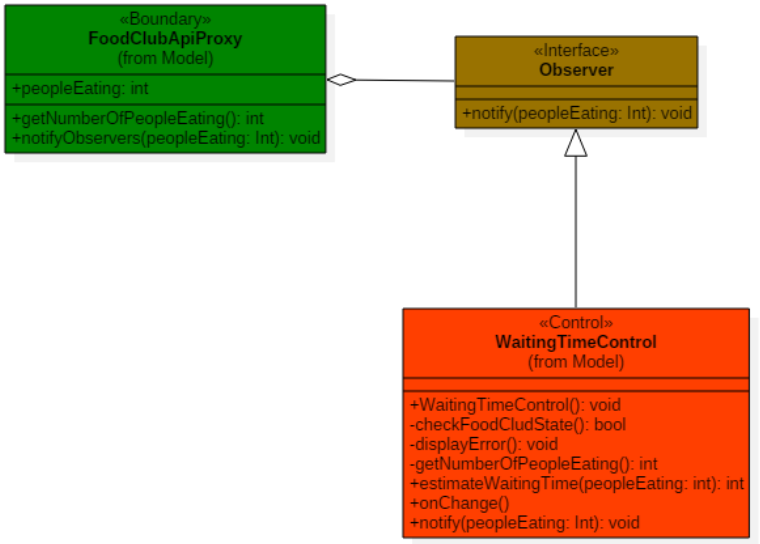
Όνομα προτύπου: Observer

Κατηγορία: Πρότυπα συμπεριφοράς

Πρόβλημα που αντιμετωπίστηκε: Κατά την παρακολούθηση από τον χρήστη του εκτιμώμενου χρόνου αναμονής στην λέσχη, ήταν πολύ πιθανό ο χρόνος αυτός να άλλαζε εάν ο χρήστης κρατούσε την εφαρμογή ανοικτή για κάποιο χρονικό διάστημα με αποτέλεσμα η μη ανανέωση του χρόνου αυτού να οδηγούσε σε λανθασμένα συμπεράσματα από τον χρήστη και να οδηγούσε και στην μη ικανοποίησή του από την χρήση του συστήματός μας.

Υποσυστήματα που αναδιοργανώθηκαν: Για την επίλυση του προβλήματος αναδιοργανώθηκε το πακέτο TimeEstimation.

***Σημείωση**: Στο σχήμα περιλαμβάνονται μόνο οι αλλαγές που έγιναν στα υποσυστήματα και όχι ολόκληρα τα υποσυστήματα μετά την αλλαγή.*



Σχήμα 21. Πρότυπο Observer

5. Πίνακας ιχνηλασιμότητας εγγράφων Σχεδίασης και Απαιτήσεων Λογισμικού

- Προστέθηκε η συνάρτηση dayClick() στην Boundary κλάση ResultsGUI του 2ου Παραδοτέου.

6. Παράρτημα Ι – Ανοιχτά Θέματα

Το θέμα που πρέπει να συζητηθεί με τους Διαχειριστές την ΠΦΛ πριν την οριστικοποίηση Εγγράφου Σχεδίασης Συστήματος είναι η τιμολογιακή πολιτική του συστήματος. Η τιμολόγηση του λογισμικού αναμφισβήτητα θα είναι συνδεδεμένη με το μέγεθος του έργου. Η τελική απόφαση μπορεί να βασιστεί είτε σε μοντέλο τιμολόγησης άλλων αντίστοιχων λογισμικών, είτε μέσα από γνώμη οικονομολόγων σε ιστοσελίδες διαδικτύου αντίστοιχου εύρους ανάπτυξης.

Επιπλέον είναι σημαντικό να γίνει σωστή κοστολόγηση του υπολογιστή server, ανάλογα με τον πληθυσμό των χρηστών.