# **Health Partner**

Ξάνθης Ασημάκης Γεωργιάδης Κωνσταντίνος

ΑΜ : 133978 ΑΜ :144164

[It133978@it.teithe.gr](mailto:It133978@it.teithe.gr) it144164@it.teithe.gr

# **Εισαγωγή**

Η εργασία αυτή πραγματοποιήθηκε στα πλαίσια του μαθήματος Ηλεκτρονικές Υπηρεσίες Προστιθέμενης Αξίας και έχει ως σκοπό την δημιουργία και παρουσίαση μίας εφαρμογής διατροφολογίας. Μέσα στην καθημερινότητά του ο άνθρωπος έχει πολλές ασχολίες, μία από αυτές που καταναλώνει όλο και περισσότερο χρόνο αποτελεί η χρήση ενός υπολογιστή, όμως κατά τον χρόνο αυτό ο χρήστης δεν έχει την δυνατότητα να πραγματοποιήσει και άλλες εργασίες ταυτόχρονα. Αυτή η εφαρμογή σχεδιάστηκε ώστε να μπορεί οποιοσδήποτε χρησιμοποιεί υπολογιστή να χρησιμοποιήσει και αυτήν, ταυτόχρονα με οποιαδήποτε άλλη εργασία εξοικονομώντας έτσι χρόνο ενώ παράλληλα βοηθάει στη διατήρηση ενός σταθερού παράγοντα έχοντας ήδη προγραμματίσει τα μελλοντικά του γεύματα.

Η εφαρμογή που αναπτύχθηκε χρησιμοποιεί μία διαδικτυακή υπηρεσία από την οποία έχει την δυνατότητα να δέχεται δεδομένα μέσα από ένα σύστημα αιτήσεων, οπότε και αποτελεί πολύ σημαντικό κομμάτι της εφαρμογής. Λόγω της σημαντικότητας της υπηρεσίας αυτής είναι καλό να εξηγηθεί γιατί επιλέχθηκε η συγκεκριμένη υπηρεσία. Αρχικά θα γίνει μία ανάλυση των κύριων τεχνολογιών που ακολουθούν οι υπηρεσίες και στην συνέχεια.

Τις τελευταίες δεκαετίες με την ανάπτυξη του διαδικτύου και την εκθετική αύξηση των δεδομένων που έχουν γίνει διαθέσιμα, έχουν αναπτυχθεί πολλά μοντέλα για την ανταλλαγή πληροφοριών μεταξύ απομακρυσμένων τερματικών. Οι δύο τεχνολογίες που έχουν επικρατήσει και είναι πιο ευρεία διαδεδομένες από τις υπόλοιπες είναι οι SOAP και οι REST. Οι δύο αυτές προσεγγίσεις αναλαμβάνουν να λύσουν το πρόβλημα της αποστολής δεδομένων ανάμεσα σε μία υπηρεσία και ένα χρήστη. Σκοπός της υπηρεσίας είναι να έχει δεδομένα τα οποία μπορεί να αποκτήσει ένας χρήστης ο οποίος, χρησιμοποιώντας τα κατάλληλα εργαλεία μπορεί να την ζητήσει και στην συνέχεια η υπηρεσία να του τα παρέχει.

**Συγκρίνοντας**

Αρχικά καλό είναι να γίνει κατανοητό πως οι και οι δύο τεχνολογίες αναπτύχθηκαν για να λύσουν ακριβώς το ίδιο πρόβλημα, δηλαδή πώς δύο διαφορετικές μεταξύ τους εφαρμογές, προγράμματα ή συσκευές ανταλλάσσουν και μοιράζονται μεταξύ τους πληροφορίες, με έναν επεκτάσιμο και εύκολα κατανοητό τρόπο.

# **SOAP**

Το SOAP ( Simple Object Access Protocol ) ξεκίνησε στα τέλη της δεκαετίας του 90 με σκοπό να ικανοποιήσει την ανάγκη για την μεταφορά δεδομένων μέσα από το ίντερνετ. Για την επικοινωνία των δύο πλευρών το SOAP είναι ανεξάρτητο από πρωτόκολλα μεταφοράς δεδομένων ωστόσο το πλέον συνηθισμένο μέσο αποτελεί το HTTP, μέσα στο οποίο ενθυλακώνεται η πληροφορία βάση της XML. Ο τρόπος που χρησιμοποιείται το HTTP και οποιοδήποτε άλλο πρωτόκολλο μεταφοράς όμως δεν είναι παρόμοιος με τον τρόπο που δημιουργήθηκε για να χρησιμοποιηθεί, καθώς το SOAP δεν κάνει χρήση δυνατοτήτων του όπως τις μεθόδους αιτήσεων (GET,POST,HEAD...), τους κωδικούς κατάστασης, την κατάσταση σύνδεσης κτλ., αντίθετα όλος ο όγκος της πληροφορίας βρίσκεται ενθυλακωμένος με αποτέλεσμα ο αποδέκτης να μην έχει καμία γνώση για την εισερχόμενη πληροφορία μέχρι την αποκωδικοποίηση του μηνύματος. Η προσέγγιση αυτή έχει αρκετά θετικά όσο και αρνητικά. Ένα από τα θετικά είναι πως μη χρησιμοποιώντας τα χαρακτηριστικά κάποιου ξεχωριστού πρωτοκόλλου το SOAP είναι ανεξάρτητο και μπορεί να χρησιμοποιηθεί πάνω από πολλές εφαρμογές, ενώ ταυτόχρονα με την χρήση μίας καλά τυποποιημένης γλώσσας όπως η XML γίνεται και ανεξάρτητο πλατφόρμας και γλώσσας προγραμματισμού. Επίσης ανάπτυξη των XML μηνυμάτων ακολουθεί πρότυπα ως προς την δομή και την σημασιολογία των κεφαλίδων, χαρακτηριστικό που έχει οδηγήσει στην τυποποίηση του, και ταυτόχρονα με την μακροχρόνια ανάπτυξή του στα πρότυπα του WS\* έχει εδραιωθεί ως ξεχωριστό ανεξάρτητο πρωτόκολλο μετάδοσης πληροφορίας. Τέλος λόγω της ανεξαρτησίας από τα πρωτόκολλα επικοινωνίας κάθε εφαρμογή που χρησιμοποιεί το πρωτόκολλο SOAP έχει την δυνατότητα για την ανάπτυξη ξεχωριστού ελέγχου σφάλματος στο επίπεδο της εφαρμογής, κάτι που κάνει πιο αποδοτική την χρήση του μέσα σε ελεγχόμενο περιβάλλον αν και ιδιαίτερα πιο πολύπλοκο στην ανάπτυξή του.

**REST**

Από την άλλη το REST δημιουργήθηκε με σκοπό να αναιρέσει τα αρνητικά του SOAP. Το μεγαλύτερο μειονέκτημα του SOAP αποτελεί το γεγονός ότι χρειάζεται πλήρης γνώση της υπηρεσίας που με την οποία γίνεται η ανταλλαγή δεδομένων λόγω του ότι μπορεί να είναι τροποποιημένη και λόγω του προσαρμόσιμου ελέγχου σφαλμάτων. Ο τρόπος με τον οποίο η αρχιτεκτονική REST επιτυγχάνει την επικοινωνία των δύο πλευρών είναι μέσω του HTTP πρωτοκόλλου από το οποίο είναι πλήρως εξαρτημένη καθώς έχει σχεδιαστεί να δουλεύει πάνω σε αυτό. Εξαιτίας λοιπόν αυτής της εξάρτησης γίνεται και πλήρης χρήση των επιπλέον χαρακτηριστικών που προσφέρει το HTTP, δηλαδή μία προσέγγιση τελείως διαφορετική από του SOAP. Μέσω του HTTP ο αποδέκτης έχει την δυνατότητα να αντιλαμβάνεται την ταυτότητα του χρήστη, το ιστορικό της επικοινωνίας τους, την λειτουργία που θα επιτελεστεί με τα δεδομένα και γενικότερα όλες τις ωφέλειες της χρήσης του HTTP. Επίσης με τον τρόπο αυτό το REST γίνεται πιο απλό στην χρήση του αφού δεν έχει ξεχωριστό έλεγχο σφαλμάτων από το HTTP. Όπως γίνεται εύκολα κατανοητό η αρχιτεκτονική REST χρησιμοποιεί το πρωτόκολλο επικοινωνίας όπως μία ιστοσελίδα, οπότε θα μπορούσε κάποιος να το παρομοιάσει εύκολα με μία ιστοσελίδα ενώ το πρωτόκολλο REST ως ολοκληρωτικά ξεχωριστό πρωτόκολλο. Τέλος η REST αρχιτεκτονική δεν έχει κάποιο τυποποιημένο τρόπο αποστολής των μηνυμάτων, όπως το REST έχει την XML, αντίθετα είναι πιο ευέλικτο έχοντας την δυνατότητα να χρησιμοποιήσει διαφορετικές μορφές μορφοποίησης της πληροφορίας όπως το JSON, XML,CSV κτλ. Με αυτόν τον τρόπο η αρχιτεκτονική REST μπορεί να χαρακτηριστεί από μεγαλύτερη ευελιξία από το REST, χαρακτηριστικό το οποίο θα μπορούσε να προσδώσει μία πολυπλοκότητα ωστόσο η χρήση καλά τυποποιημένων μορφοποιήσεων συνηθίζεται οπότε δεν δημιουργεί πρόβλημα στην πραγματικότητα.

Έχοντας επεξηγήσει πλέον τα σημαντικότερα χαρακτηριστικά και διαφορές των δύο αυτών τεχνολογιών μπορεί να εξηγηθεί καλύτερα γιατί επιλέχτηκε μία υπηρεσία που χρησιμοποιεί την προσέγγιση της REST. Αρχικά να αναφερθεί ότι επειδή η REST χρησιμοποιεί το πρωτόκολλο HTTP κάνει πολύ πιο εύκολη την αυθεντικοποίηση, των χρηστών, κάτι που μπορεί να γίνει ιδιαίτερα πολύπλοκο με την SOAP ειδικά όσον αφορά την επικοινωνία ανάμεσα σε διαφορετικές πλατφόρμες. Επιπλέον το μόνο πρότυπο που ακολουθείτε από την SOAP αφορά τις κεφαλίδες των XML μηνυμάτων καθώς πέρα από αυτό τα δεδομένα εξακολουθούν να προέρχονται από κάπου αλλού οπότε και να είναι ομαδοποιημένα ως πολύπλοκα αντικείμενα, αντίθετα η REST ακολουθεί το πρότυπο του HTTP. Όπως γίνεται εμφανές τα πλεονεκτήματα της REST προέρχονται από την χρήση του HTTP πρωτοκόλλου και όπως μπορεί να παρατηρηθεί η ιδεολογία της είναι παρόμοια με μίας ιστοσελίδας, ενώ αντίθετα η SOAP με μία πλήρως προσαρμόσιμη εφαρμογή.

# **Η υπηρεσία**

Η υπηρεσία που χρησιμοποιήθηκε για την υλοποίηση της εφαρμογής αναπτύχθηκε από την Spoonacular με την ονομασία food API. Η υπηρεσία αυτή αναπτύχθηκε βάση της REST προσέγγισης οπότε για την επικοινωνία με μεταξύ πελάτη - διακομιστή χρησιμοποιείται το HTTP βάση του οποίου γίνεται και η αυθεντικοποίηση. Για την απόκτηση πρόσβασης στα δεδομένα της υπηρεσίας χρειάζεται αρχικά η δημιουργία κάποιου λογαριασμού στην σελίδα της υπηρεσίας ή η αυθεντικοποίηση μέσω ενός τρίτου έμπιστου φορέα με τις επιλογές ανάμεσα σε Google, Facebook και GitHub. Με την δημιουργία του λογαριασμού η υπηρεσία παρέχει στους προγραμματιστές 2 κλειδιά, ένα που λειτουργεί ως κωδικός και ένα που λειτουργεί ως δρομολογητής και κατευθύνει το αίτημα στον κατάλληλο διακομιστή. Τα δύο κλειδιά χρησιμοποιούνται με κάθε αίτημα προς την υπηρεσία και αποτελούν μέρος του HTTP αιτήματος ως 2 κεφαλίδες. Το domain στο οποίο γίνονται τα αιτήματα αποτελεί το request URL και είναι διαφορετικό για κάθε πληροφορία είδος πληροφορίας. Τα μηνύματα που επιστρέφονται ακολουθούν την μορφοποίηση JSON και όπως κάθε request URL είναι διαφορετικό έτσι είναι και η απάντηση που επιστέφεται. Για καλύτερη κατανόηση μπορεί το αίτημα με την υπηρεσία να χαρακτηριστεί ως μία μέθοδο get() σε μία κλάση αντικειμένου ενώ η απάντηση του διακομιστή με το αντικείμενο που επιστρέφεται.

## **Δυνατότητες**

Η υπηρεσία περιστρέφεται γύρω από την διατροφολογία όπως αναφέρθηκε οπότε θα γίνει μία γρήγορη αναφορά στις αναζητήσεις που μπορούν να γίνουν και στα δεδομένα που επιστρέφονται από αυτές. Οι μέθοδοι της υπηρεσίας χωρίζονται σε 5 κύριες κατηγορίες.

• Στην πρώτη κατηγορία κυριαρχούν μέθοδοι με τις οποίες ένας χρήστης μπορεί να εισάγει δεδομένα στην υπηρεσία και μέθοδοι που η υπηρεσία παράγει κάποιο αποτέλεσμα που δεν είναι πάντα το ίδιο όπως η δημιουργία ενός εβδομαδιαίου προγράμματος. Στις μεθόδους για εισαγωγή δεδομένων γίνεται χρήση της μεθόδου POST του HTTP και είναι σε αυτήν την κατηγορία που συναντιούνται στο API καθώς ο κύριος όγκος αιτημάτων γίνονται με την μέθοδο GET.

• Στην δεύτερη κατηγορία υπάρχουν μέθοδοι οι οποίες έχουν την δυνατότητα να εξάγουν πληροφορία από την αποστελλόμενη αίτηση όπως για παράδειγμα η αναζήτηση μίας συνταγής μέσα από μια πρόταση( π.χ βάφλες χωρίς σοκολάτα) ή αναζήτηση συνταγής σε μία ιστοσελίδα παρέχοντας μόνο το URL της.

• Η τρίτη κατηγορία έχει και την μεγαλύτερη μερίδα της λειτουργικότητας της υπηρεσίας αφού εδώ βρίσκονται οι περισσότερες μέθοδοι που επιτρέπουν πρόσβαση στα πληροφορίες και αποτελούνται κυρίως από αναζητήσεις. Εδώ υπάρχουν η υπηρεσία παρέχει την δυνατότητα να γίνουν αναζητήσεις για συνταγές, φαγητά που βρίσκονται στο σούπερ-μάρκετ, πιάτα από καταλόγους εστιατορίων και όλες αυτές οι αναζητήσεις μπορούν να γίνουν ανάλογα χρησιμοποιώντας φίλτρα που θέλει ο χρήστης όπως θερμίδες, διατροφική αξία, κουζίνα(μεσογειακή, ανατολική), δίαιτα( χορτοφαγική , δυσανεξία στην λακτόζη) και άλλα διαφορετικά και συγκεκριμένα σε κάθε αναζήτηση. Επίσης προσφέρει την δυνατότητα της αυτόματης συμπλήρωσης στις αναζητήσεις και την επιλογή κρασιού ανάλογα με το γεύμα.

• Στην τέταρτη κατηγορία υπάρχουν μέθοδοι ανάλυσης των αποτελεσμάτων της τρίτης κατηγορίας, δηλαδή παρέχεται η δυνατότητα για επιπλέον πληροφορίες όσον αφορά τη διατροφική αξία των φαγητών. Εδώ να αναφερθεί πως για στις αναζητήσεις επιστρέφονται κάποιες βασικές πληροφορίες για κάθε φαγητό οπότε χρειάζεται ένα ξεχωριστό αίτημα για την άντληση περισσότερων πληροφοριών και αυτή είναι η ανάγκη που καλύπτει η τέταρτη κατηγορία.

• Τέλος η πέμπτη κατηγορία αποτελείται από 2 μεθόδους που αντιστοιχούν με αιτήματα για ομιλία με ένα Chat Bot. Στην πρώτη ο χρήστης μπορεί να συζητήσει για θέματα γενικού περιεχομένου ενώ μπορεί να πάρει σαν όρισμα μία ελεύθερη έκφραση και λειτουργεί ως κατηγοριοποιητής, δηλαδή καλεί με την σειρά της κάποια από τις παραπάνω μεθόδους που είναι πιο πιθανό να αντιστοιχεί στο αίτημα του της ελεύθερης έκφρασης.

‘Ολες οι πληροφορίες σχετικά με τις μεθόδους όπως το URL, τις προαιρετικές παραμέτρους, τις κεφαλίδες, τις μεθόδους του HTTP και τις τιμές που επιστρέφει είναι δημόσια διαθέσιμες από το Web Site της εφαρμογής. Τέλος να αναφερθεί πως η υπηρεσία υποστηρίζει Android, Java, Javascript, C#, Go, PHP, Node.js, και Ruby, δηλαδή μία ευρεία γκάμα από γλώσσες και πλατφόρμες προγραμματισμού συμπεριλαμβανομένων των περισσότερο δημοφιλών μεθόδων για την ανάπτυξη εφαρμογών με βάση τα Web Services.

**Περιβάλλον**

Το περιβάλλον που αναπτύχθηκε η εφαρμογή είναι το VIsual Studio Community Edition και ως γλώσσα προγραμματισμού η C#.

Η έκδοση που χρησιμοποιήθηκε αποτελεί την δωρεάν διανομή της πλατφόρμας με λειτουργικότητα παρόμοια με αυτήν της Professional έκδοσης του 2014.Υποστηρίζει πολλές γλώσσες και την χρήση πακέτων. Το Visual Studio αποτελεί μία πλατφόρμα ανάπτυξης λογισμικού που δημιουργήθηκε από την MIcrosoft και πλέον υποστηρίζει τις περισσότερες γλώσσες προγραμματισμού και έχει ένα επεξεργαστή κώδικα, ενσωματωμένο Debugger και βιβλιοθήκες για ανάπτυξη εφαρμογών για Desktop, Web, γραφικό σχεδιασμό περιβάλλοντος, σχεδιασμό κλάσεων, σχήματος βάσης δεδομένων. Πολύ σημαντικό είναι το γεγονός πως υποστηρίζει την δυνατότητα για δημιουργία και εισαγωγή πακέτων έχοντας μάλιστα και δικό του διαχειριστή. Μία βιβλιοθήκη χρησιμοποιήθηκε από το διαχειριστή πακέτων NuGet,η Json.NET από την Newtonsoft για την επεξεργασία των Json αντικειμένων που επιστρέφονται από το API.

Η C# είναι η γλώσσα προγραμματισμού που αναπτύχθηκε από την MIcrosoft στις αρχές του 2000. Βλέποντας μειονεκτήματα στον σχεδιασμό των πιο δημοφιλών γλωσσών προγραμματισμού όπως Java, C++ η Microsoft δημιούργησε το πρότυπο CLI ( Common Language Runtime ) το οποίο στην συνέχεια οδήγησε στον σχεδιασμό της C#. Υπάρχουν πολλά κοινά ανάμεσα στις 2 αυτές προ προΰπάρχουσες γλώσσες τα οποία περνώντας τα χρόνια αποτελούσαν όλο και μικρότερο ποσοστό έχοντας διαφορετικές σκοπούς, έτσι και στα αρχικά στάδια της η C# εμφάνιζε πολλές ομοιότητες και στην λειτουργία και στην ανάπτυξη με την Java και τη C++, όμως με την συνεχή ανάπτυξη της γλώσσας εδραιώθηκε ώς μία ξεχωριστή προσέγγιση.

# **Η εφαρμογή**

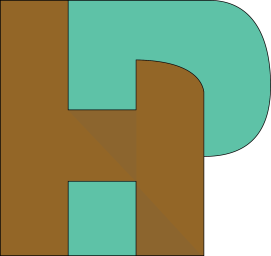
Ανοίγοντας την εφαρμογή ο χρήστης καλωσορίζεται από μία εισαγωγική φόρμα στην οποία μπορεί να κάνει εγγραφή ή να συνδεθεί αν έχει ήδη λογαριασμό. Για την σύνδεση με την βάση απαιτείται ένας κωδικός και ένα όνομα χρήστη, ενώ για την δημιουργία λογαριασμού μία το ονοματεπώνυμο, το βάρος, το ύψος και η ηλικία. Η εφαρμογή αποτελείται από τέσσερις καρτέλες οι οποίες φαίνονται όταν ο χρήστης συνδεθεί. Η πρώτη είναι μία καρτέλα με τα στοιχεία του χρήστη οι στην οποία έχει την δυνατότητα να αλλάξει τα στοιχεία του αν το επιθυμεί. Τα στοιχεία δεν είναι επεξεργάσιμα και γίνονται όταν ο χρήστης πατήσει την επιλογή Edit, οπότε και έχει το δικαίωμα να τα επεξεργαστεί και να μεταβάλει ότι επιθυμεί. Στην συνέχεια μπορεί να τα σώσει πατώντας το κουμπί Save όπου και τα στοιχεία αποθηκεύονται στην Βάση στην θέση των παλιών. Επίσης στην καρτέλα αυτή υπάρχει και το κουμπί About το οποίο έχει μερικά στοιχεία των δημιουργών της εφαρμογής μαζί με ένα e-mail.

Η επόμενη καρτέλα ονομάζεται Food και σε αντίθεση με την προηγούμενη εδώ είναι η πρώτη καρτέλα που γίνεται χρήση του API. Στην καρτέλα αυτή παρέχεται στον χρήστη η δυνατότητα για την αναζήτηση κάποιου φαγητού που είναι διαθέσιμο σε καταστήματα, σημαντικό είναι να γίνει ξεκάθαρο πως δεν γίνεται αναζήτηση κάποιας ολοκληρωμένης συνταγής αλλά των επιμέρους χαρακτηριστικών της. Η αναζήτηση γίνεται βάση του ονόματος της τροφής και αυτό αποτελεί υποχρεωτικό πεδίο όμως μαζί με αυτό υπάρχει και ένα σύνολο προαιρετικών διατροφικών χαρακτηριστικών που λειτουργούν σαν φίλτρα και κάνουν πιο αποτελεσματική την αναζήτηση του χρήστη. Τα πεδία αυτά είναι οι ελάχιστες και μέγιστες θερμίδες, πρωτεΐνη, υδατάνθρακες και λίπος. Η αναζήτηση γίνεται πατώντας το κουμπί Search και τότε καλείται η μέθοδος για το On-Click. Σε αυτήν την μέθοδο δημιουργείται ένα αντικείμενο της κλάσης RestClient η οποία είναι η κλάση που γίνονται τα αιτήματα προς το API. Για την δημιουργία ενός αντικειμένου RestClient χρειάζεται το URL το οποίο δίνεται από την μέθοδο του κουμπιού και ποια μέθοδος του HTTP θα χρησιμοποιηθεί, ως προκαθορισμένη είναι η μέθοδος GET μέσα από τον δομητή όμως είναι δηλωμένα και τα υπόλοιπα HTTP verbs (POST, PUT, DELETE). Η μέθοδος λοιπόν που καλείται ονομάζεται makeFoodRequest και επιστρέφει ως τιμή ένα πίνακα με αντικείμενα της κλάσης Food, η κλάση αυτή δημιουργήθηκε για τον καλύτερο χεισρισμό των δεδομένων που επιστρέφονται από την από το API. Η μέθοδος makeFoodRequest καλεί την μέθοδο getResponse() η οποία είναι και η μέθοδος που κάνει όλα τα αιτήματα. Δέχεται ως παράμετρο το URL και δημιουργεί ένα αντικείμενο HttpWebRequest βάση του URL, στη συνέχεια προστίθενται οι κεφαλίδες αυθεντικοποίησης και μέσω της μεθόδου GetResponse της HttpWebRequest επιστρέφεται η απάντηση σε ένα αντικείμενο HttpWebResponse. Πλέον αφού τα δεδομένα είναι στην εφαρμογή σκοπός είναι να μετατραπούν σε βασικούς τύπους. Για να γίνει αυτό πρώτα μετατρέπονται σε μία ακολουθία από bytes δηλαδή σε αντικείμενο της κλάσης Stream μέσω της κλάσης GetResponseStream του HttpWebResponse και στην συνέχεια σε String μέσω της StreamReader. Το String αυτό είναι που επιστρέφεται στην μέθοδο makeFoodRequest, όμως είναι το Json αρχείο που επιστράφηκε απλά σε μορφή κειμένου που σημαίνει πως δεν μπορεί να χρησιμοποιηθεί ακόμα. Για αυτό χρησιμοποιείτε το επιπλέον πακέτο Json.Net με την μέθοδο DeserializeObject της κλάσης JsonConvert. Η μέθοδος αυτή δέχεται ως όρισμα το String και αυτό που κάνει είναι να εντοπίζει τις διαβαθμίσεις που υπάρχουν, δηλαδή ποίες μεταβλητές είναι υποκατηγορίες άλλων μεταβλητών, ποιες δεν έχουν άλλη υποκατηγορία αλλά μόνο τιμή και μετατρέπει τις τιμές αυτές σε αντικείμενα συνδεδεμένα μεταξύ τους. Η τιμή που επιστρέφει είναι δυναμική οπότε και για αυτό η μεταβλητή που δέχεται το αποτέλεσμα είναι τύπου var που σημαίνει ότι ο τύπος της καθορίζεται κατά την διάρκεια της χρήσης του προγράμματος. Τώρα πλέον οι ζητούμενες τιμές είναι διαθέσιμες από μεταβλητές κάτω από την ιεραρχία των κλάσεων και τα ονόματα των κλάσεων καθορίζονται από τις επιστρεφόμενες τιμές του API. Για την αποθήκευση και καλύτερη διαχείριση των δεδομένων δημιουργήθηκε στην εφαρμογή η κλάση Food με τα πιο σημαντικά πεδία που επιστρέφονται. Η επιστρεφόμενη τιμή είναι λοιπόν ένας πίνακας με 10 αντικείμενα Food τα οποία και εμφανίζονται μέσω ενός DataGridView, το οποίο παρέχει την δυνατότητα εμφάνισης εικόνων εκτός από τους βασικούς τύπους. Οι στήλες του πίνακα αυτού είναι τρείς και αποτελούνται από το ID, το όνομα, μία εικόνα η οποία είναι τύπου System.Drawing.Image, καθώς αυτά είναι να στοιχεία που επιστρέφει η υπηρεσία κατά την αναζήτηση, για επιπλέον πληροφορίες χρειάζεται να γίνει ξεχωριστό αίτημα. Το DataGridView έχει την δυνατότητα για δημιουργία γεγονότος πατώντας πάνω στο κελί μιας γραμμής, οπότε για κάθε επιστρεφόμενη τιμή ο χρήστης μπορεί να αναζητήσει επιπλέον πληροφορίες σχετικά με το προϊόν αυτό οι οποίες παρουσιάζονται σε ένα ξεχωριστό παράθυρο που δημιουργείται.

Δύο καρτέλες έχει ακόμα η εφαρμογή και η μία από αυτές είναι το Schedule στην οποία ο χρήστης μπορεί να επιλέξει για την δημιουργία ενός εβδομαδιαίου προγράμματος με τρία γεύματα την ημέρα και προαιρετικά μπορεί να επιλέξει και ένα ανώτατο όριο ημερήσιων θερμίδων. Η διαδικασία για δημιουργία του αιτήματος είναι παρόμοια με του φαγητού που επεξηγήθηκε στην προηγούμενη παράγραφο, εδώ όμως χρησιμοποιείται ένα διαφορετικό URL και μία διαφορετική μέθοδος της κλάσης RestClient. Πάλι για αποτελεσματικότερη χρήση των δεδομένων δημιουργήθηκε η κλάση WeekMeal στην οποία υπάρχουν ως μεταβλητές η μέρα του γεύματος, το ID και ο τίτλος του γεύματος. Ένας πίνακας με 21 εγγραφές επιστρέφεται από την μέθοδο makeMealRequest(), που εξυπηρετεί την ίδια λειτουργικότητα με την makeFoodRequest(). Τα στοιχεία των γευμάτων εμφανίζονται μέσω ενός DataGridView στο οποίο υπάρχει η δυνατότητα με την επιλογή του πεδίου ID να γίνει εμφάνιση των συστατικών και των βημάτων για την ολοκλήρωση του γεύματος. Τέλος υπάρχει και η καρτέλα Recipe στην οποία προσφέρεται η δυνατότητα αναζήτησης συνταγών. Οι προαιρετικές επιλογές εδώ είναι τέσσερις, εξαιρούμενα υλικά, δυσανεξία σε τροφές, είδος πιάτου και τύπος δίαιτας και μαζί με το όνομα της συνταγής γίνεται το αίτημα στην υπηρεσία μέσω της μεθόδου makeRecipeRequest(). Ενδιαφέρον παρουσιάζει το γεγονός πως οι πληροφορίες που επιστρέφονται είναι διαφορετικές από αυτές για ένα γεύμα από την δημιουργία εβδομαδιαίου προγράμματος, ενώ τα αντικείμενα φαίνεται να είναι αποθηκευμένα στην υπηρεσία κάτω από το ίδιο ID. Λόγω αυτού υπάρχει ακόμα μία κλάση για τον χειρισμό των συνταγών, η Recipe, η οποία έχει ως πεδία το ID, το όνομα, τα λεπτά για την ετοιμασία και τον αριθμό των μερίδων. Η makeRecipeRequest() επιστρέφει λοιπόν έναν πίνακα με τις περισσότερο ταιριαστές συνταγές και εμφανίζονται σε ένα DataGridView στο οποίο ο χρήστης έχει πάλι την δυνατότητα με την επιλογή του πεδίου ID να αναζητήσει περισσότερες πληροφορίες για το γεύμα και αντο κάνει, θα εμφανιστούν σε ένα ξεχωριστό παράθυρο τα υλικά και τα βήματα της συνταγής, όπως ακριβώς και με τα γεύματα του εβδομαδιαίου προγράμματος. Για την εμφάνιση γίνεται χρήση του γεγονότος Click για τα κελιά του DataGridView και καλείται η μέθοδος makeRecipeInformationRequest() η οποία δέχεται το URL βάση του ID και επιστρέφει τις αντίστοιχες πληροφορίες.

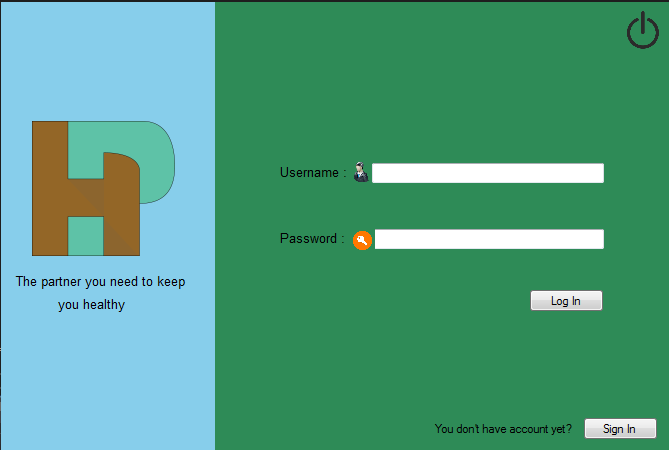
# **UI**

Λόγω του θέματος της εφαρμοφής που επιλέχθηκε , θεωρήσαμε σημαντικό να υπάρχουν χρώματα που να μην το κάνουν υπερβολικό , κουμπιά που να είναι κοντά μεταξύ τους , textboxes με κατάλληλο μέγεθος .Όλα αυτά σε θέση και στοίχηση που διευκολύνει τον χρήστη και του δίνει μια ευχάριστη σαφώς εμπειρία. Συνήθως, σχεδόν όλα τα προγράμματα έχουν ένα συγκεκριμένο λογότυπο το οποίο τα διακρίνει , έτσι κι έγινε στην περίπτωση της εφαρμογής .

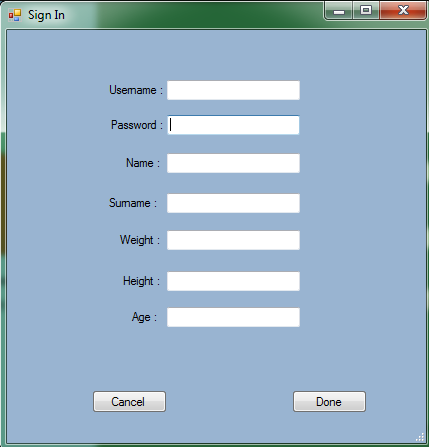


Για το Main πρόγραμμα εφαρμόσαμε το μπλέ ανοικτό που είναι χρώμα φιλικό στα μάτια αλλά και εξίσου άνετο. Θα παρουσιαστούν παρακάτω οι φόρμες που φτιάχτηκαν .

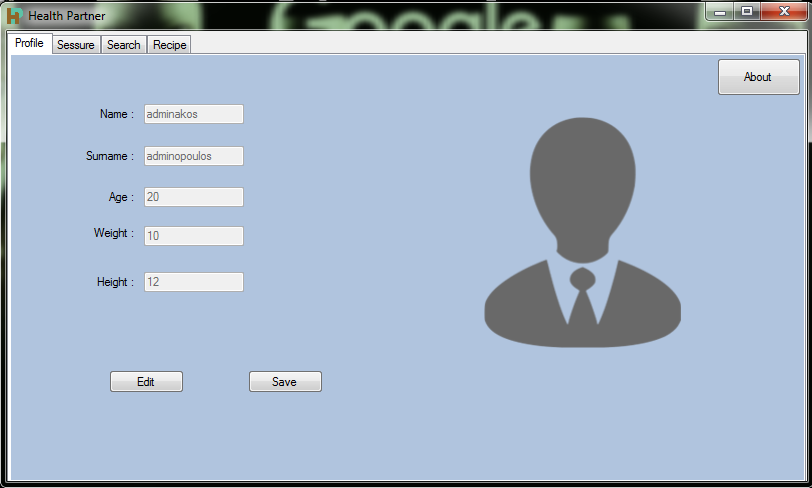
Login Form



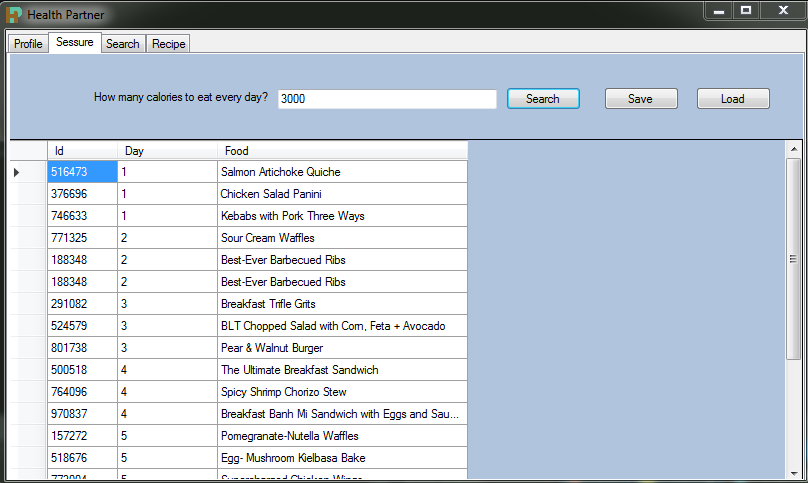
Sign In Form



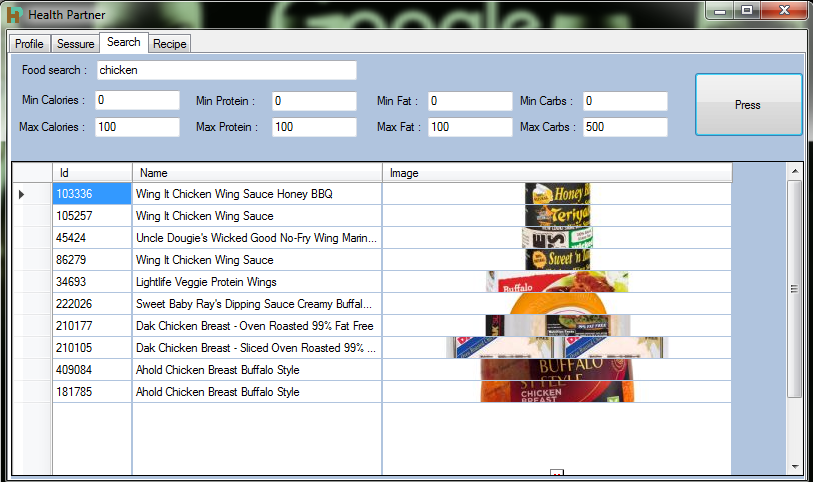
Profile Καρτέλα



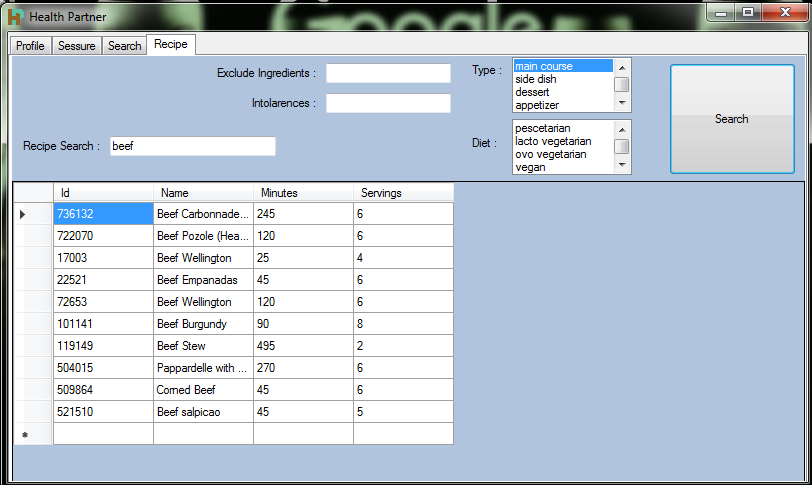
Sessure Καρτέλα



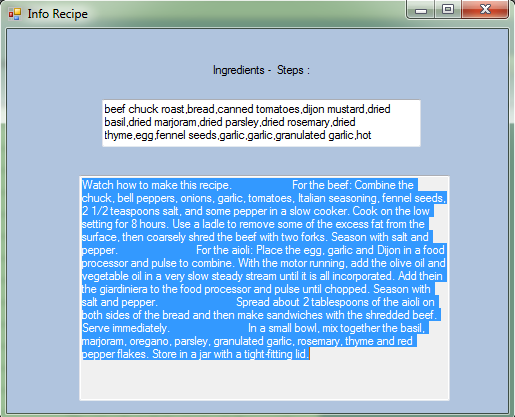
Search Καρτέλα



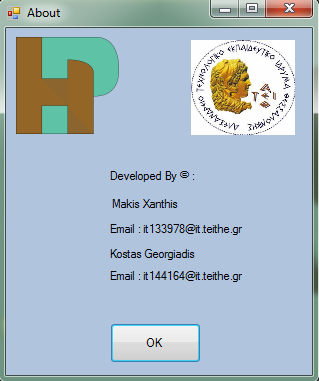
Recipe Καρτέλα



Info Recipe Form



About Form



# **Βάση Δεδομένων**

Μια βάση δεδομένων είναι απαραίτητη σε κάθε εφαρμογή τέτοιου επιπέδου . Αλλά θα πρέπει να υπάρχει λόγος που την φτιάχνεις . Ο λόγος λοιπόν που στήσαμε την βάση είναι γιατί θέλαμε να κρατάει τα στοιχεία των χρηστών , με βασικό στόχο το πρόγραμμα τους και την σύνδεση τους στην εφαρμογή με δικά τους στοιχεία όπως φαίνεται στην φόρμα Login . Έτσι έχουμε 2 tables το ένα για τους χρήστες και το άλλο για το πρόγραμμα που σώζουν με το κουμπί Save που υπάρχει στην καρτέλα Sessure. Υπάρχει φυσικά ένα κοινό πεδίο ώστε να γίνεται το ταίριασμα χρήστη με το δικό του πρόγραμμα . Προσφέρουμε την δυνατότητα στο χρήστη να αλλάξει τα στοιχεία του στην καρτέλα Profile και φυσικά να σώσει τις αλλαγές αυτές. Έτσι , με αυτόν τον τρόπο καλύπτουμε τις ανάγκες της εφαρμογής και φυσικά του χρήστη.

**Μελλοντική Επέκταση**

Στην εφαρμογή που αναπτύχθηκε χρησιμοποιήθηκαν οι πιο σημαντικές μέθοδοι τις υπηρεσίας ωστόσο υπάρχουν ακόμα πολλές που θα μπορούσαν να προσδώσουν λειτουργικότητα και δεν έχουν υιοθετηθεί. Όπως αναφέρθηκε οι χρήστες οι ίδιοι μπορούν να εισάγουν πληροφορίες στην εφαρμογή και να επεκταθεί το σύνολο των δεδομένων της, ένας από αυτούς τους τρόπους είναι οι χρήστες να εισάγουν δικές τους συνταγές. Στη συγκεκριμένη επιλογή η εφαρμογή θα χρησιμοποιούσε την μέθοδο POST του HTTP και κάνοντας ένα παρόμοιο αίτημα στην υπηρεσία με κάποιες βασικές πληροφορίες της συνταγής, θα γινόταν μία καινούργια συνταγή στην υπηρεσία. Με αυτόν τον τρόπο άλλοι χρήστες θα μπορούσαν να έχουν πρόσβαση σε αυτήν ή ακόμα και ο ίδιος χρήστης σε μία διαφορετική τοποθεσία, έτσι εξαλείφεται η ανάγκη να αποθηκεύει κάπου αλλού τις συνταγές του και να τις έχει όλες μαζί οργανωμένες σε μία σταθερή μορφή. Επίσης η υπηρεσία παρέχει αναζήτηση συνταγών που υπάρχουν στα εστιατόρια, με αυτό τον τρόπο οι χρήστες μπορούν να ξέρουν ακριβώς τα θρεπτικά συστατικά και τις θερμίδες που λαμβάνουν όταν τρώνε έξω χωρίς να χρειάζεται να υπολογίζουν το κάθε συστατικό μίας συνταγής μεμονωμένα, ενώ ταυτόχρονα παρέχεται και αυτόματη συμπλήρωση στις αναζητήσεις αυτές αλλά και των συνταγών και τροφών. Ακόμα μία χρήσιμη μέθοδος που παρέχεται αφορά την αντικατάσταση υλικών σε συνταγές, με αυτόν τον τρόπο γίνεται εφικτό κάποιος να μπορέσει να ολοκληρώσει μία συνταγή με μερικά διαφορετικά υλικά που μπορεί να μην είχε διαθέσιμα ή να αλλάξει κάποια που δεν είναι της προτίμησής του.

Μία διαφορετική ιδέα για την ανάπτυξη της υπηρεσίας αποτελεί η ιδέα να γίνει και χρήση μίας δεύτερης υπηρεσίας με σκοπό την πιο ολοκληρωμένη προσέγγιση της υπηρεσίας όσο αφορά το κομμάτι της υγείας. Είναι πολύ πιθανό πως όποιος ενδιαφέρεται για την διατροφή του θα ενδιαφέρεται και για την φυσική του κατάσταση οπότε μία ακόμα υπηρεσία που θα μπορούσε να προστεθεί θα ήταν σχετική με την γυμναστική, ώστε να μην υπάρχει ανάγκη για χρήση δεύτερης εφαρμογής, αντίθετα θα ήταν όλα διαθέσιμα με έναν ομογενοποιημένο τρόπο μέσω της εφαρμογής. Ένα ακόμα γνωστικό πεδίο που θα μπορούσε να υιοθετηθεί θα ήταν αυτό της ιατρικής. Σε αυτήν την περίπτωση θα μπορούσε να υπάρχει η δυνατότητα να γνωρίζει ο κάθε χρήστης αν το πρόγραμμά της διατροφής του ανταποκρίνεται στις ανάγκες που έχει ο οργανισμός του όσον αφορά τις βιταμίνες και τα θρεπτικά συστατικά που πρέπει να λαμβάνει, και η επικοινωνία των δύο υπηρεσιών θα γινόταν με έναν απρόσκοπτο τρόπο μέσω της εφαρμογής.

**Βιβλιογραφία**

<https://en.wikipedia.org/wiki/Application_programming_interface>

<https://medium.freecodecamp.org/what-is-an-api-in-english-please-b880a3214a82>

<https://smartbear.com/blog/test-and-monitor/understanding-soap-and-rest-basics/>

<https://spf13.com/post/soap-vs-rest/>

<https://www.youtube.com/watch?v=11f5KzVNQ90&t=526s>

<https://www.youtube.com/watch?v=XX5pn4pJ4b0&t=1207s>

<https://en.wikipedia.org/wiki/Representational_state_transfer>

<https://www.infoworld.com/article/3269878/apis/what-is-an-api-application-programming-interfaces-explained.html>