Bilkent University

Department of Computer Engineering

Senior Design Project

Foodster: Make maintaining your diet easier

Analysis Report

Khasmamad Shabanovi, Gledis Zeneli, Balaj Saleem, Ibrahim Elmas, Perman Atayev

Supervisor: Ozcan Ozturk
Jury Members: Dr. Çigdem Gündüz-Demir, Dr. Can Alkan

Contents

Introduction **3**

Proposed System **3**

Overview 3

Functional Requirements 4

Payment methods 4

Users should be able to pay using their credit or debit cards for the premium features of the app, ordering ingredients from the grocery list or ordering meals from the delivery services. 4

Non-functional Requirements 5

Accessibility 5

Pseudo Requirements 7

System Models 8

Scenarios 8

Use-Case Model 12

Object and Class Model 13

Dynamic Models 14

User Interface 17

Other Analysis Elements **24**

Consideration of Various Factors in Engineering Design 24

Risks and Alternatives 25

Project Plan 26

Ensuring Proper Teamwork 30

Ethics and Professional Responsibilities 30

Planning for New Knowledge and Learning Strategies 30

References **30**

Analysis Report

Foodster: Make maintaining your diet easier

1 Introduction

Food is the first of all fundamental human needs, yet how many times today have you, personally, consciously thought of what you have and will be eating today, what its nutritional value is and how it fits into your greater nutritional, fitness, and lifestyle goals? With the ease of falling into a routine and the abundance of staple food, keeping track of all these variables, and making sure that you plan your meals such that your priorities are heeded, becomes increasingly mundane, monotonous, and just unnecessary extra work.

With the fast-paced lifestyle of the 21st century, monotony and unnecessary work are the last things an individual needs in his/her busy life. Figuring out what to eat, to order or to cook, how to cook it, where to get the ingredients from, and how beneficial this meal would be for your body are questions that would take precious time and energy that could be better employed elsewhere.

Furthermore, the complications of searching for the nutritional data, planning a healthy diet according to one's needs, planning the budget for such a diet, and finding recipes to support this time is truly a cumbersome endeavor.

This is where Foodster comes in. The goal of the project is to simplify the complicated process of eating healthy and take the thinking out of the simple and fundamental process of having food. This means the complete integration of planning meals with one's constraints, listing the ingredients and requirements for such meals, automated ordering of these ingredients or automated ordering of these meals. This draws inspiration from apps such as myfitnesspal 0 which help the user reach their nutritional goals with the least possible effort.

2 Proposed System

2.1 Overview

Foodster is meant to be a complete meal management solution that will provide everything from having a specific nutritional goal to having the meal that makes that goal a reality on the table.

At its core will be a cross-platform(android and IOS) mobile application that will have a simple set of stages: planning, detailing, and ordering.

The planning stage will get the nutritional preferences and constraints of the user which may include but are not limited to:

- Amounts of meals per day
- Frequency of meal repetition per week
- Amount of times and times of day expected to eat out
- Budgetary constraints
- Meal plan goal (e.g. lose weight, gain muscle, etc)
- Metabolism relevant user data (age, sex, weight, amount of weekly exercise)
- Regional constraints (related to location and hometown).

Using this information foodster will generate a meal-plan. This meal-plan will then be used as the cornerstone of the user's diet in the future. Furthermore, factors such as diet type,

possible budget constraints, location, and other such factors that affect the nutritional habit of the user, would be implicitly taken into account.

The recipe's generated from the meal plan and their ingredients can then be ordered online through the ordering service of the app. The grocery list will be automatically generated based on the available recipe. Any ingredients unavailable will be highlighted and the others will be delivered after the user's approval using services such as Migros Sanal Market 1, Carrefour 2, or Getir 3.

Alternatively, the user will be provided with the option to get food delivered directly from a restaurant, using yemeksepeti 4 as a service. Alternatives may be suggested with nutritional values close to the user's meal plan and then automated order can be placed on behalf of the user.

To refine the Foodster platform and make data-entry as least cumbersome as possible, food recognition based on machine learning methods will be used to recognize the recipe and get the overall nutritional value. Hence, accurate logging may be done for the user even if they deviate from the suggested meal plan.

Foodster will also allow the user to have an estimate of the price based on their preferences and trends. Such as, how often they prefer to eat out, what is the current price of the ingredients they need for their meals, and how much they have spent previously on such meals. This will allow for a more transparent road-map for users given their financial situation.

Finally, the platform will allow user preference estimation by looking for patterns of ingredients, calories, sugar levels, and other factors from the available data for the user. This data can be used for more personalized and beneficial suggestions.

2.2 Functional Requirements

2.2.1 Payment methods

Users should be able to pay using their credit or debit cards for the premium features of the app, ordering ingredients from the grocery list or ordering meals from the delivery services.

2.2.2 Grocery List

- Users should be able to compile their grocery lists for one time, one week and month period of time.
- Users should be able to order everything from the grocery list and pay online.
- Users should be set an automatic order request in the app for one week or one month, so that the latest compiled grocery list is automatically ordered for them.

2.2.3 Meal Planning

- Users should be able to plan meals for one time, one week and one month consumption using the preferences stored in their account.
- Users should be able to order the closest meal to what was planned for them from the food delivery services such as Yemeksepeti and Bilkent Yemek.

2.2.4 Data Logging

- Users should be able to add their preferences such as diet type, calorie amount, protein amount, minerals etc to their accounts, so that meal planning and grocery compilation uses that data.

- Users should be able to update the preferences detected from either pictures they have uploaded or entered manually.
- Users should be able to update their data regarding their physical characteristics such as height, weight, age etc.
- (Premium feature) Paid Users should be able to add pictures of the meals they have eaten so that the system can detect their preferences of meal types, budget per meal, amount of calories in the meal and other criterias.

2.3 Non-functional Requirements

2.3.1 Accessibility

- The system will require Android Jelly Bean 4.1.x or newer and iOS 8 or newer because we will be using Flutter library to build applications for both Android and iOS. We will use Node.js for building the web-server of the application that will be easy to send requests to from Flutter applications.

2.3.2 Accuracy

- If the preferences of users are available for a user, they should be satisfied as much as possible. Especially the preferences that could affect the health condition of a user such as allergies must be considered.

2.3.3 Availability

- Our initial audience will be Turkey; therefore, any down-time for updating databases, fixing some critical bugs should be happening at night in Turkey, so that the least amount of users are affected by the down-time of the application that should not exceed more than 1 hour.
- Since the system is highly modular, the whole system should not be down due to an update to a particular submodule.

2.3.4 Backup and Recovery

- Firebase is going to take care of Backup and Recovery of data, since that is the database we are going to use.
- The preferences of users should be stored both locally and globally, so that no data is lost.

2.3.5 Capacity

- The server should have satisfactory computation power and storage for each user.
- The server should handle at least 10000 registered users from Turkey.

2.3.6 Compatibility

- Libraries used in Flutter should not conflict with any Android or iOS phones that support installation of the application.
- The users should have Android or iOS phone devices to support the application.

2.3.7 Concurrency

- We will use AWS to deploy our back end service which will be available 24/7. AWS will support 5-10K requests per month, which will be more than enough to keep

the response time of a user under 1s.

2.3.8 Configurability

- The user should be able to update any data that has to do with his / her payment methods, preferences, body characteristics such height, weight and any other information that user has an access to.

2.3.9 Exception Handling

- In case of exception/error, error must be realized as much as possible. The error must be clearly explained to the user clearly and also required steps must be explained, so that the user knows what to do and how to do.

- In case of unexpected errors, the users should be able to share this error with us via crash log or core dump.

2.3.10 Extensibility

- It must be easy to develop new features and add new functionality into the application for future business needs. So, logical separation of the application will be maintained so that application will be divided into different tiers(e.g. client, presentation, business logic, etc.)

2.3.11 Legal and Regulatory Requirements

- The users will be warned that they are responsible for law-violating actions (e.g. copying licensed content, etc.) that are taken by them.

2.3.12 Licensing

- Required licenses for the libraries, services and modules used during development will be arranged

2.3.13 Maintainability

- Subsystems will be loosely coupled by means of the logical separation we will maintain. So, a modification or integration to a subsystem/module will not affect others.

2.3.14 Performance

- Meal recommendations should not take more than 1 seconds.

- Redirecting users to food delivery services should not take more than 2 seconds.

- Compiling grocery lists should not take more than 4 seconds.

- Data logging/updating should not take more than 500 ms.

2.3.15 Reliability

- The application should not crash at any time due to software domained error.

- The server must be running the whole time other than the maintenance times that is 1-hour once in a month during nights in Turkey.

- Crash logs will be stored in Firebase to be inspected and analyzed to avoid further crashes.

2.3.16 Scalability

- The system will be designed so that scaling the system will be easy by choosing right technologies for web-server, database type, network etc.
- The database must be able to an annual growth rate of 20%, with no decrease in database performance
- The web-servers must be able to support an annual growth of 10% of new customers.

2.3.17 Security

- The users must log in with their private credentials.
- The web-server must be available and behave reliably even under DOS attacks..
- The application must provide the integrity of the customer's private information.

2.3.18 Testing

- The web-server, database and mobile application will be tested regularly. Some of test types will be made:
    - Reliability tests will be made for web-server to function without failure,
    - Load tests will be made for web-server to measure performance based on actual customer behavior.

2.3.19 Usability

- The GUI will be intuitive and user friendly such that it will not restrict any functionality and user interactions will be easy.
- The users will not need to spend more than 10 seconds to learn the functionalities of the screens.
- The users will be able to submit their feedback to developers.

2.4 Pseudo Requirements

- Git will be used for version control
- Dart will be used for Mobile development and Javascript for backend development.
- Node.js and associated libraries and node packages will be used to implement the REST API that client can contact with
- External APIs will be used, (where available) for nutritional info, grocery ordering, meal ordering
- The system will be implemented in a modular fashion for easier testing and maintenance, with different modules to handle database operations, client-server communication.
- The server architecture will be developed to be scalable and robust.
- Continuous integration / Continuous Development principles(CI/CD) will be used. Bitrise will be used for this.
- Firebase will be used to manage data persistence and other database operations.

- Before they are used in the project, licenses for APIs and external frameworks will be reviewed and necessary authorizations will be obtained.
- Feedback will be elicited from the users regarding the product and its services
- Modular testing will take place before integration
- Amazon web services will be used for hosting Node.js server.
- Jira will be used for issue tracking.
- Data Scraping using Selenium will be used in cases where APIs are not available.
- The product will abide by legal obligations of App Store(iOS) and Play Store(Android).
- SendGrid will be used as an email delivery service for sending authentication and invoicing related emails.

2.5 System Models

2.5.1 Scenarios

Scenario 1: Sign Up

Actors: Customer

Entry Conditions: The user opens the app

Exit Conditions: The user is navigated to the login screen.

The flow of Events:

1. The user clicks the "Sign Up" button on the Login page.
2. The registration page is open.
3. The user enters the required information.
4. The user clicks the "Sign Up" button.
5. An authentication mail is sent to the provided email address.
6. The user enters the code given in the mail.
7. The user clicks the "Verify the Code" button.
8. The system verifies the code and creates the desired account.

Scenario 2: Login

Actors: Customer

Entry Conditions: The user opens the app

Exit Conditions: The user is navigated to their home screen or to the initial detail entry screen.

The flow of Events:

1. If the user logged in from the device previously and chose to stay logged in for 30 days, within the last 30 days, the last time she logged in: cached login credentials on the User's phone are taken and sent to the back-end service to verify identity. If the

identity is verified, the user is redirected to the homepage.

2. Else if the user logins the credentials and clicks the "Login" button. The credentials are cached and sent to the back end service to check credentials.

3. The application redirected to the home page of the application by loading the user information from either cached information or the database.

Scenario 3: Enter Initial Details

Actors: Customer

Entry Conditions: The user has logged in for the first-time.

Exit Conditions: The user is brought to the homescreen.

The flow of Events:

1. The user enters Height, Weight, Gender, Age, Body Fat (Low, Medium, High), Activity level (1 no exercising-5 frequent exercising) and diet type if any such as Ketogenic, Paleo, Vegan, Vegetarian etc.

2. The user agrees to terms and conditions regarding usage of the data.

Scenario 4: Generate a meal plan

Actors: Customer

Entry Conditions: The user is on her homepage.

Exit Conditions: The user gets her generated meal plan details.

The flow of Events:

1. The user enters if she has not entered before the following information:
    - Number of meals per day
    - Timespan
    - Budget constraints
    - Diet type
    - Goals (lose / maintain / gain weight)
    - Allergies / disliked food (implicit)
    - Number of eat-outs per day / per week
    - Meal variety
2. The following are retrieved using available data:
    - Hometown
    - Location information
    - Season
    - Physical characteristics such as gender, height, weight etc.
    - All other preferences that user entered before

Scenario 5: Order a meal

Actors: Customer

Entry Conditions: The user is at the homescreen

Exit Conditions: The user clicks "Order a meal" button and is navigated to the meal ordering automated platform

The flow of Events:

1. The user chooses from a recommended or general list of available meal vendors.

2. The user selects a meal.

3. All nutritional information available regarding the meal is shown to the User.

4. If addresses are saved the user selects an address for delivery, or the user enters a new address that will be saved for the future reference.

5. If the User had saved cards connected to her account, she will use one of them to pay for the meal; otherwise, she will enter details of a new card that will be saved for the future reference.

Scenario 6: Breakdown Ingredients(Premium Feature)

Actors: Customer

Entry Conditions: The user is at the homescreen

Exit Conditions: The user views the ingredient list

The flow of Events:

1. The user opens the ingredient scanner (camera)

2. The user takes several pictures of the meal and uploads them to the app.

3. The user selects a name of the meal from a list of possible matches (if any)

Scenario 7: Generate a Meal Recipe

Actors: Customer

Entry Conditions: The user is on the Home Screen.

Exit Conditions: The user clicks "Generate Meal" button

The flow of Events:

1. The user enters the required information for the meal they plan to have or chooses to go with preference she saved previously.

2. The user views recipe pertaining to the constraints provided.

Scenario 8: Log Meal

Actors: Customer

Entry Conditions: The user is on the Home Screen.

Exit Conditions: The user clicks "Log Meal" button on the screen

The flow of Events:

1. The user selects a meal by searching for it by name or by using breakdown ingredients option(if she has a premium account) and waits for recognition.

2. The user modifies the nutritional details of the food if required.

3. The user presses save to add the meal to the log.

Scenario 9: Order Grocery

Actors: Customer

Entry Conditions: The user is on Meal Plan Details

Exit Conditions: The user is returned to the home screen and receives order confirmation via email / phone.

The flow of Events:

1. The user clicks the order grocery button.

2. The user increases or decreases the quantity of each item required for the meal plan.

3. The user chooses the vendor for groceries, if there is no default one saved for her.

4. The user chooses the delivery service for groceries (if required).

5. The user chooses a payment method.

6. The user chooses a saved address or enters one.

7. User orders groceries if payment was successful and gets an email notification regarding the order.

Scenario 10: Edit Meal Plan

Actors: Customer

Entry Conditions: The user is on Meal Plan Details

Exit Conditions: The Meal Plan Details are updated

The flow of Events:

1. The user adds / removes / modifies meals suggested by the meal plan.

2. The user selects alternatives based on preferences.

Scenario 11: Like Dislike Food

Actors: Customer

Entry Conditions: The user is on Meal Plan Details

Exit Conditions: The Food is liked / disliked

The flow of Events:

1. The user clicks the thumbs up / thumbs down button next to the meal.

2. The like/unlike button is highlighted.

Scenario 12: View Inventory

Actors: Customer

Entry Conditions: The user is on Meal Plan Details / Homescreen

Exit Conditions: The Inventory items are listed

The flow of Events:

1. The user clicks the inventory button. .

Scenario 13: Edit Grocery Inventory

Actors: Customer

Entry Conditions: The user is on Inventory Screen

Exit Conditions: The inventory items are updated

The flow of Events:

1. The user clicks the item which is to be updated.
2. The user changes the quantity available.

2.5.2 Use-Case Model

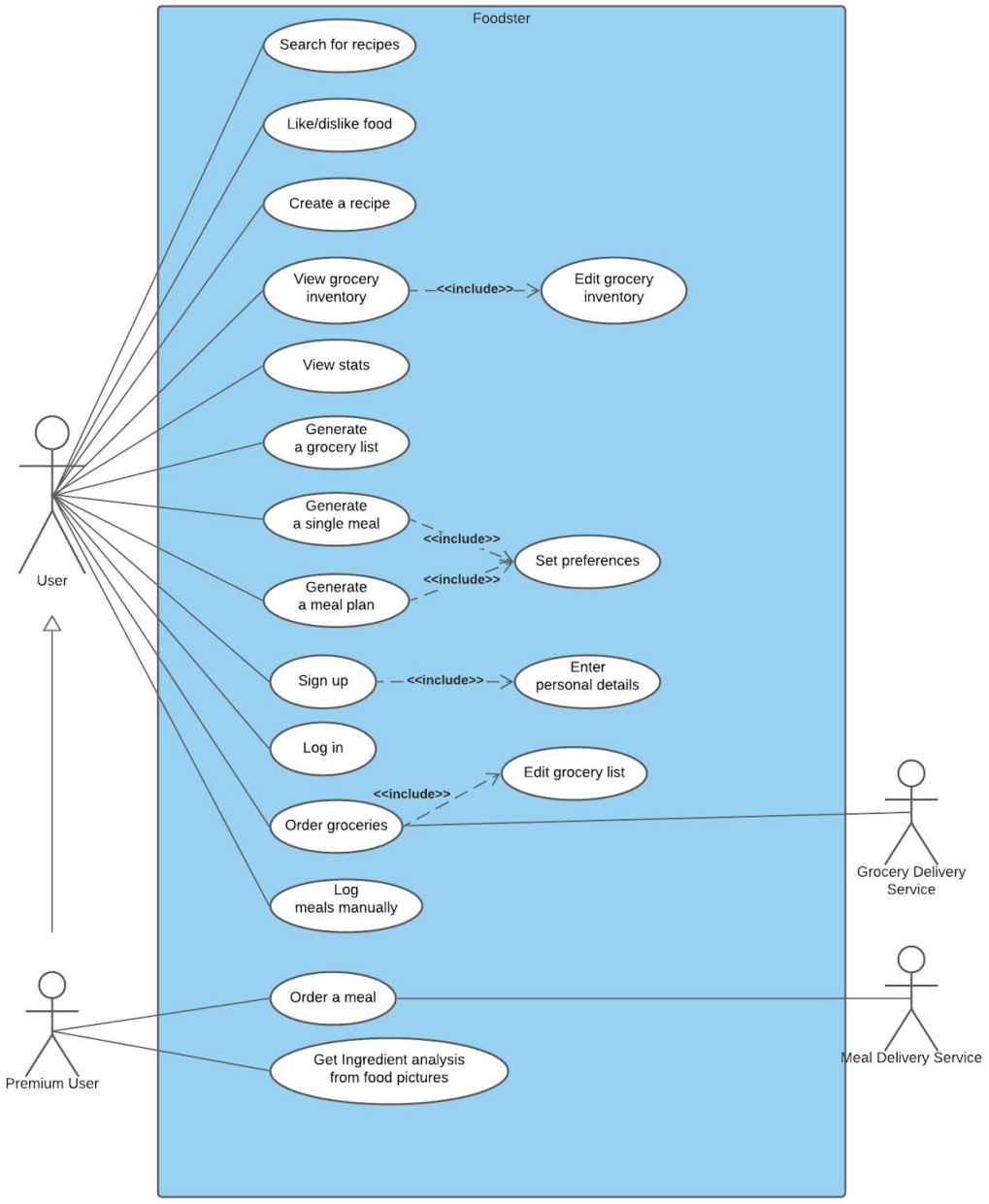

Figure 1. Use Case Diagram

2.5.3 Object and Class Model

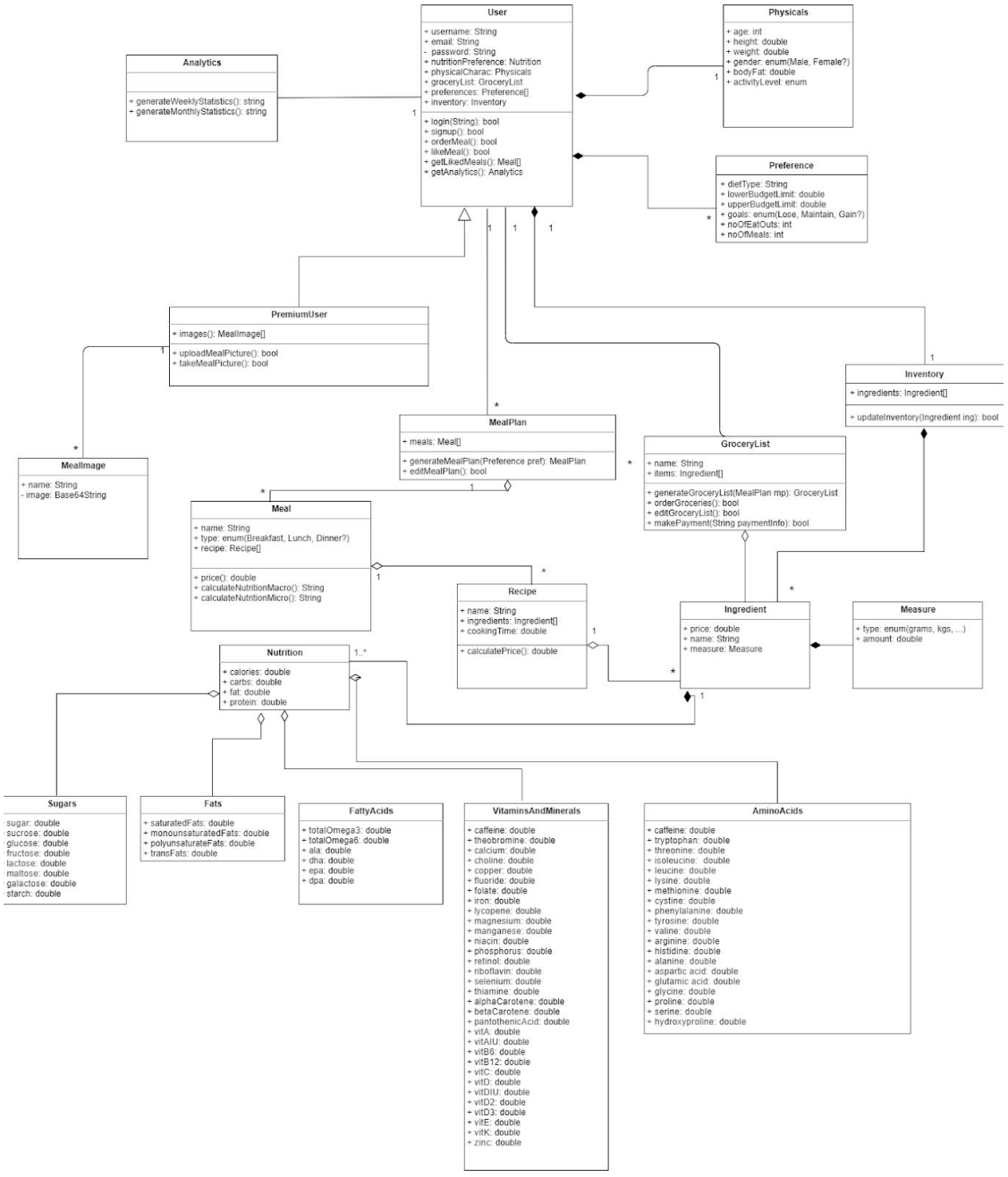

Figure 2. Object and Class Model

2.5.4 Dynamic Models

The below diagram depicts the sequence of operations for generating a meal plan.

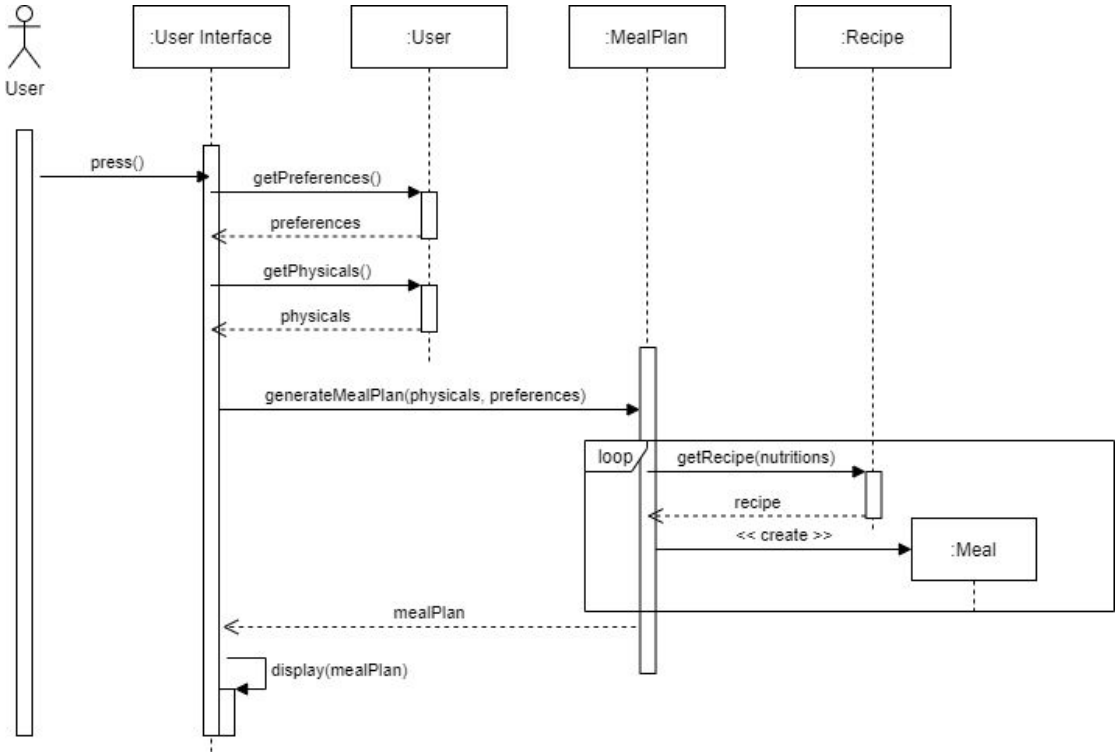

Figure 3. Sequence of operations when generating meal plan

You can see the sequence of operations for ordering a grocery list in the below picture.

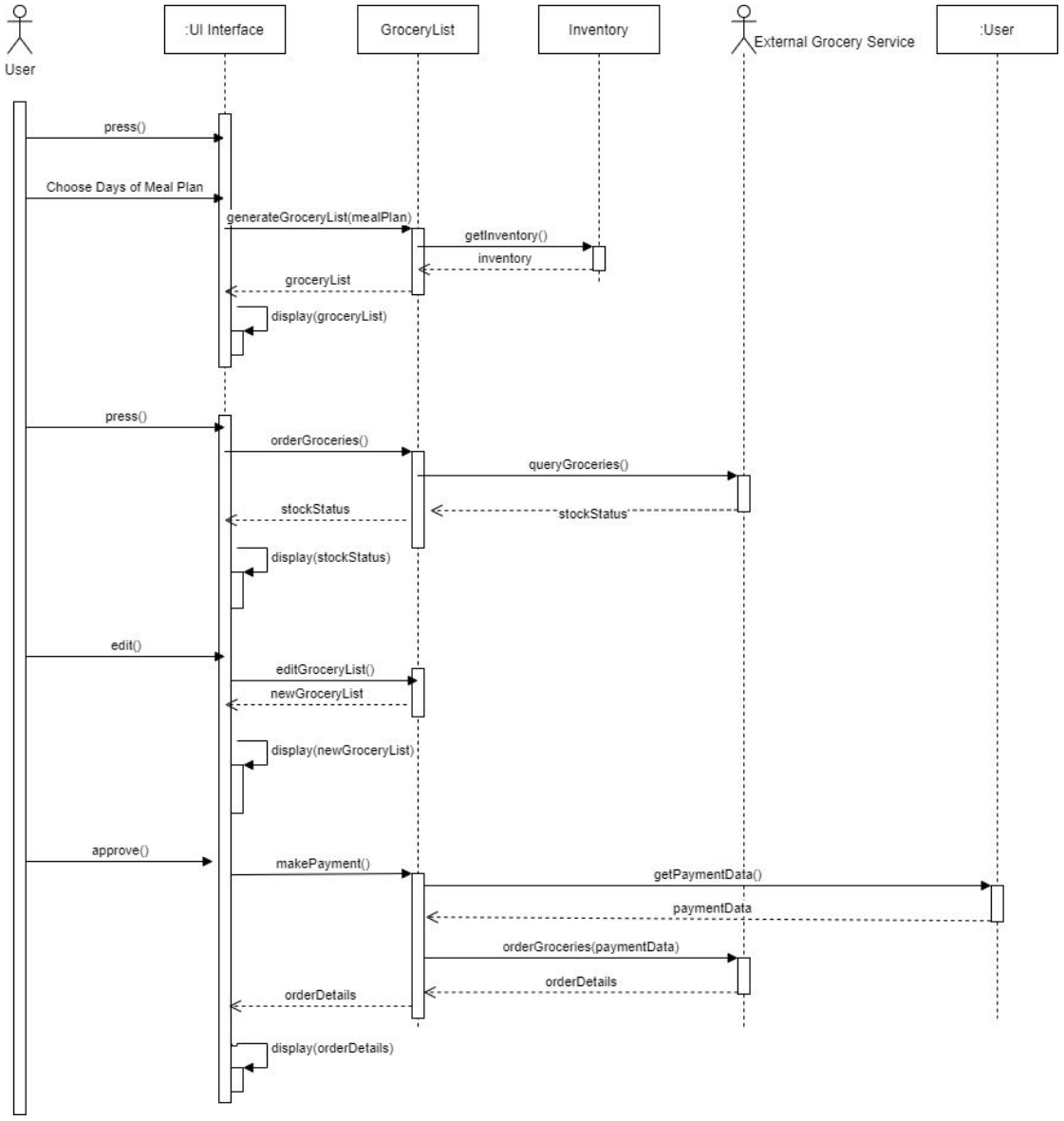

Figure 4. Sequence of operations when ordering grocery list

Below is the activity diagram of ordering a grocery list.

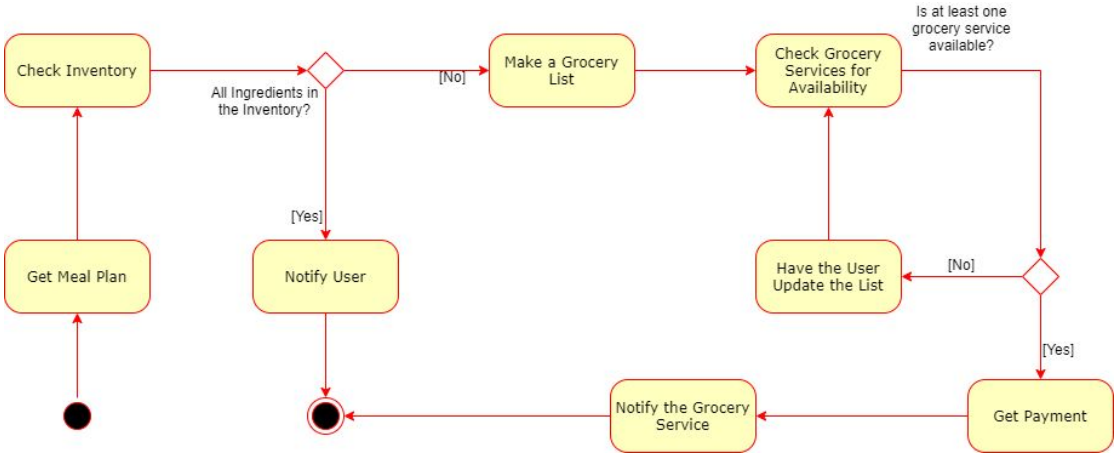

Figure 5. Activity Diagram of Ordering Grocery List

2.5.5 User Interface

Our application is intended to primarily be used in mobile phones, and the UI diagrams have been designed with this in mind.

The users will be asked to sign in or sign up if they don't have an account. An option to sign up using typical third party accounts like Google, Apple, and Facebook, will be provided to the users.

Figure 6. Login screen

During Sign Up the users will be asked to fill in a minimal yet crucial amount of information through a variety of different form input widgets. If the users sign up with one of the third party accounts, available information will be filled in for them.

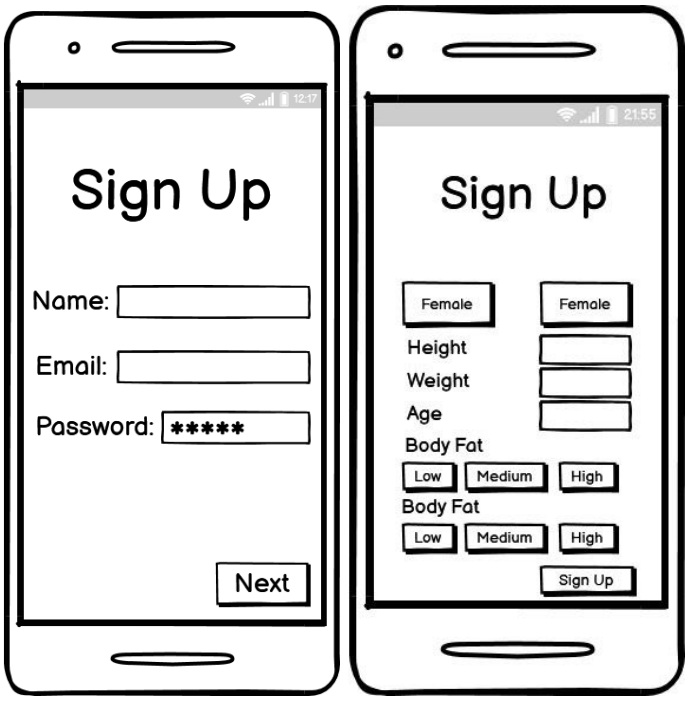

Figure 7. Sign up information extraction

The 5 major functionalities of the application are exposed in the form of a sliding menu at the bottom of the screen. This menu will be visible on all screens of the application. (throughout the rest of the diagrams the menu will not be shown anymore to avoid redundancy)

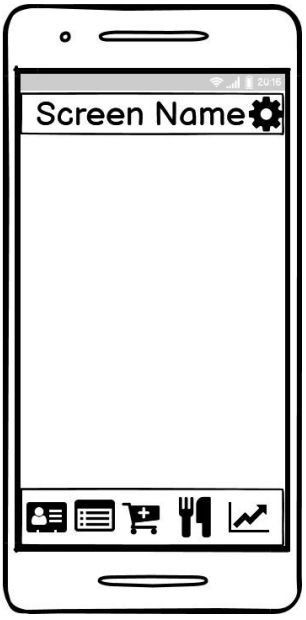

Figure 8. The sliding navigation menu UI

The first of the menu options is the Statistics screen which displays the users nutrition consumption statistics. These statistics will be provided for multiple indicators and will be presented in the form of graphs. These graphs will be interpreted in short paragraphs to give insight on the users' health.

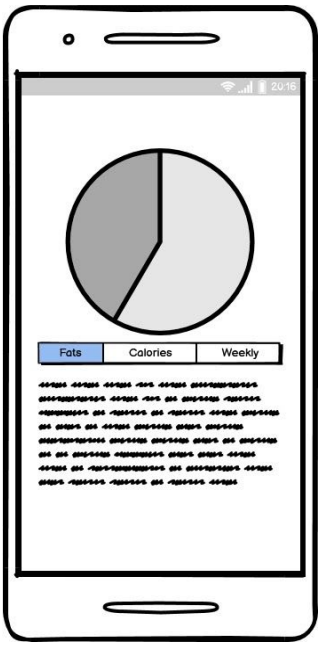

Figure 9. The statistics screen

The second menu option is the Groceries tab. Here the user can see the items in their inventory, and can access their shopping cart.

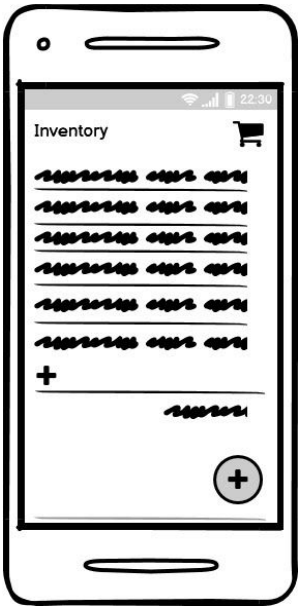

Figure 10. Inventory in grocery tab screen

If the user already created and saved a grocery list, it will be shown in the shopping cart. If the shopping cart is empty, the user will be able to generate a shopping list for some arbitrary timespan in their diet. That will generate an ingredient list with their estimated prices. The user can then choose to order the groceries from several presented options.

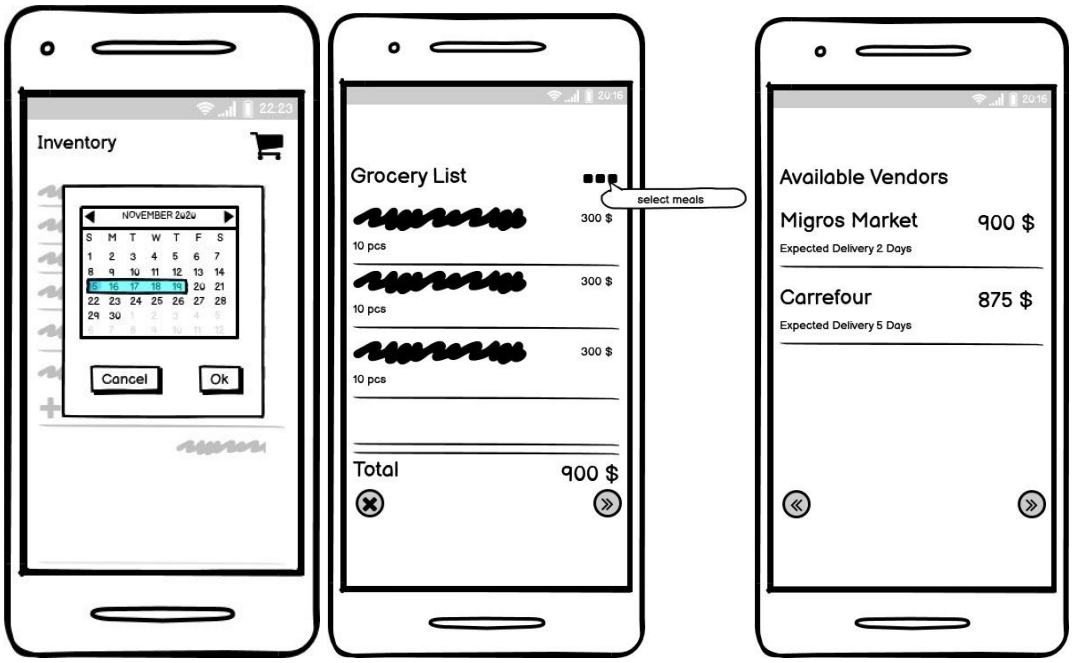

Figure 11. UI for generating a grocery list and selecting an ordering option

The user will be able to save credit/debit cards in the app, or add a new card. They will be able to use these cards to pay through our app for the groceries which will be delivered to their address.

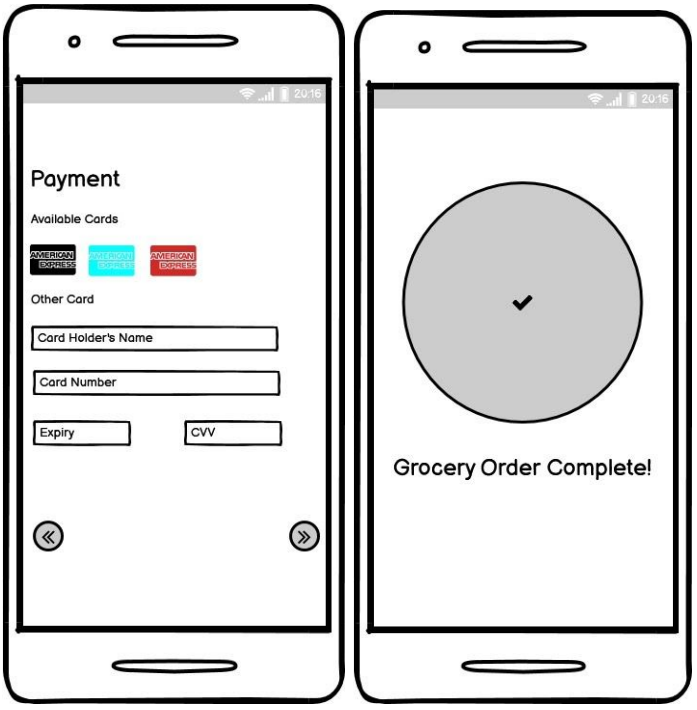

Figure 12. Payment and order confirmation screen

The middle menu option is the Meal tab. On entry the user will be presented with the scheduled meals for the day. A sliding calendar is accessible at the top of the screen. The user can use this to look at the scheduled meals for the other days. Tapping on the current day, will provide a menu with commonly used functionalities.

The meals can have custom or default names. They will contain the names, calories, pictures, and the size of the servings for the specific meal.

At the bottom of the page, a list of unplanned meals will be displayed that the user might have decided to generate for their own reasons. These will be displayed in a similar to the scheduled meals.

Figure 13. The main Meal page the menu available by clicking the current day

Tapping one of the servings will open an information tab on the food recipe will be presented. The tab will include information on preparation time, cooking time, calories, servings; a pie chart on the macros distribution for the recipe; vitamins, minerals, a list of required ingredients presented with pictures and required amounts, a description of the recipe, and an estimated price.

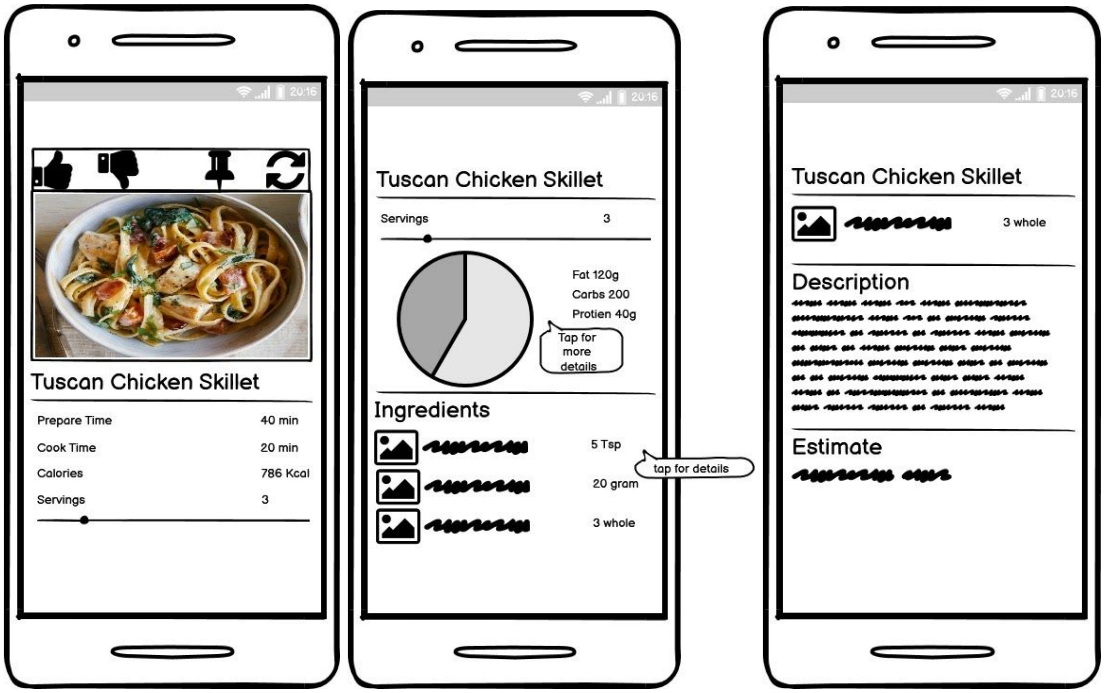

Figure 14. The recipe information tab

Taping one of the ingredients displayed in the recipe will lead to an ingredient tag displaying information about the macro-s, vitamins, and minerals present in the ingredient.

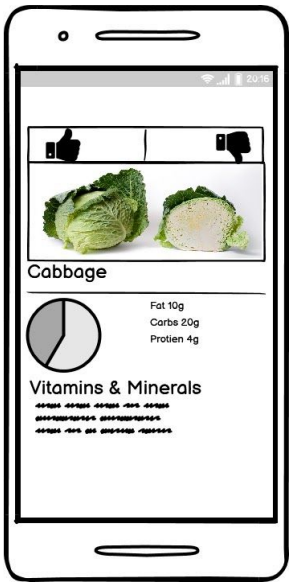

Figure 15. Ingredient information tab

If no scheduled meals are available for the day, the user will be shown a single + button. Tapping that will generate a meal plan based on the users' saved preferences (more on these later). The users then will be required to go through everyday's planned meals and approve or change them as they see fit. At the end of the approval stage the user will have the option to order groceries for the generated meal plan which will lead to the exact same UI presented in Figure 11.

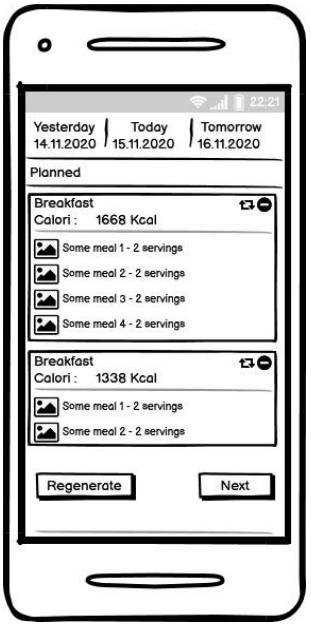

Figure 16. Approval screen for each day after the meal plan is generated

The fourth menu option, the Recipes screen, will show a list of recipes categorized in some arbitrary way we will decide. It allows for recipe search which can be filtered more intricately via the advanced tab presented under the search bar. A + button is available if the user wants to add their own recipe. The UI of adding your own recipe is identical to Figure 14, but all the entries are editable.

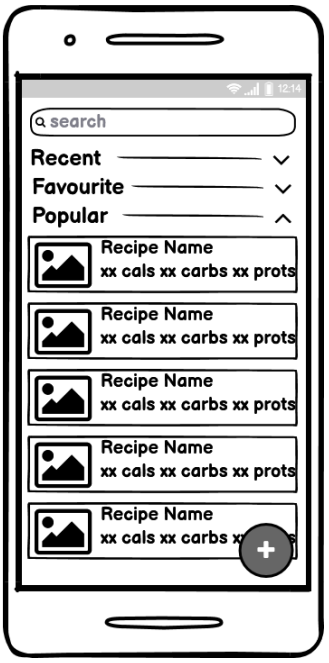

Figure 17. Recipe tab

The advanced search options will allow for a large array of applicable filters which we log using diverse UI input logging widgets.

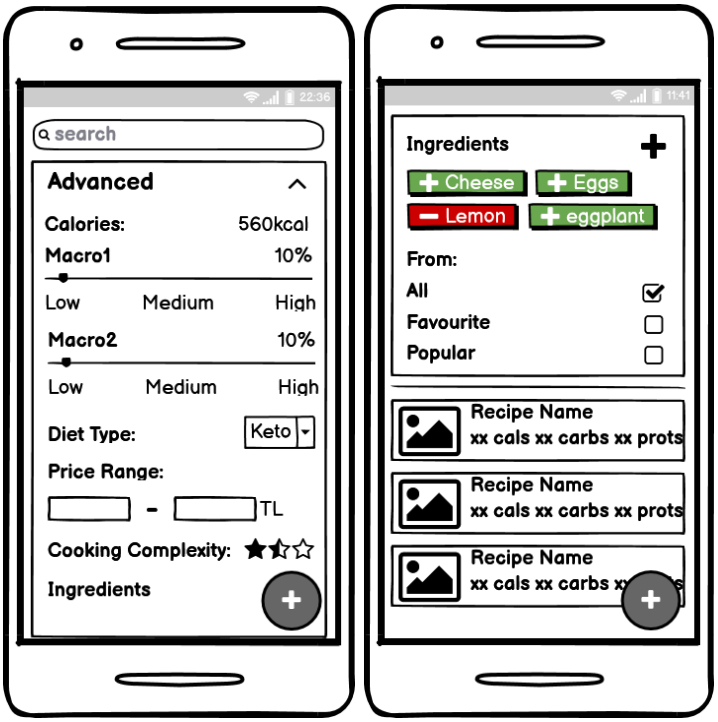

Figure 18. Different UI widgets we could use for logging the different filters

The last menu option, Profile & Preferences, will show a list of personal user information, the majority of which are collected at sign up, which can be edited. Below that, a list of created preferences will be shown.

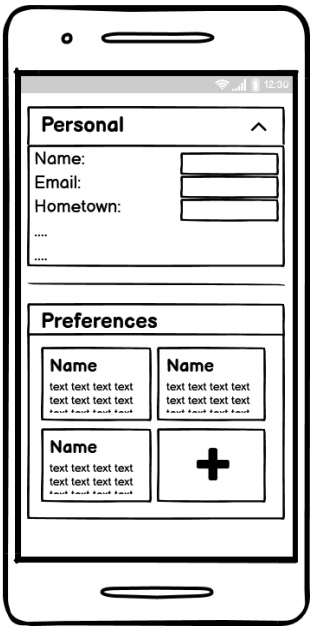

Figure 19. Profile & preferences screen

Preferences can be opened for edit by tapping on the specific preference, or a new preference can be created by tapping the + button. The UI for editing and creating preferences is the same.

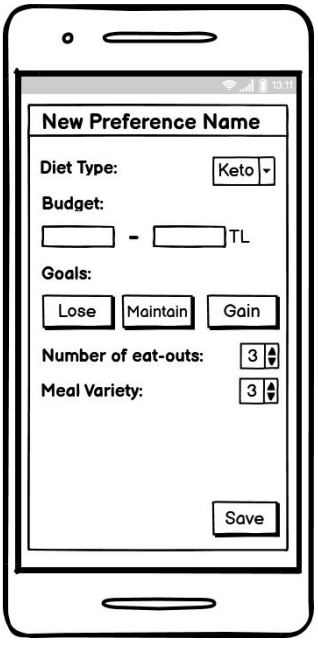

Figure 20. Preferences add/edit form

3 Other Analysis Elements

3.1 Consideration of Various Factors in Engineering Design

Our product revolves around optimising the convenience and health factors of cooking. Therefore, a big part of our design process will be food and health related considerations ranging from more serious conditions like allergies and diabetes to milder concerns like making sure the users receive a well balanced variety of nutrients. These will dictate how we configure our recommendation algorithms. Depending on how one looks at it, these

may also be considered as safety concerns especially when certain foods might risk the well being of the users.

One of the features of our app is financial optimization of cooking. We will focus on designing a system which achieves this while maintaining quality of product recommendations.

In order to make the experience most tailored to the user, we will design our recommendations as such to take into account the food culture of the region where the user lives and has grown up with. We believe this will have a great impact on the user experience with the app.

Table 1. Factors that can affect analysis and design.

	Effect level	Effect
Public health	10	Improving the health of the users is a primary concern. We will have to configure our recommendations so that they provide the greatest benefits to our users.
Public safety	4	We will have to put reliable failsafes to protect users who can have adverse health effects from specific foods.
Public welfare	4	Designing our tool so we both collect optimal data and give proper recommendations to help our users make cheaper and qualitative choices.
Global factors	0	Our app is quite personal in the experience it provides and therefore its design is not really reliant on global factors.
Cultural factors	8	In order to provide the best experience possible our recommendations will have to be tweaked such that they account for the cultural elements of local cuisine.
Social factors	0	We did not find any social factors which may be critical to consider during our design process.

3.2 Risks and Alternatives

Our project has an abundance of features and some ambitious goals which are not displayed in competitor products operating in Turkey. Therefore, we will be engaging with challenges new to the Turkish markets, and there will expectedly be some risks attached to this.

Interfacing with ordering services is a big selling point of our application. However, there doesn't seem to be any other service in the Turkish market to have done this successfully. Ideally, we would be able to get an API from the service provider which would allow us to scan their catalogue and make orders programmatically. Unfortunately, it appears that this will not be the case and we will have to resort to scrapping these services to be able to provide said functionalities to our users.

Another big selling point of our software is the ability to give location aware food recommendations. In order to achieve this, we would need an abundance of high quality Turkish food recipes. We have been able to find some international recipes nicely formatted into databases online, but the same cannot be said about Turkish recipes. These recipes are available online, but not necessarily nicely formatted. If we are not able to acquire access to such databases, we will have to scrape and format the data ourselves from the cooking websites.

Our software employs picture based food logging to make the process of the users recording their diet seamless. We intend to do this via machine learning (ML). Adapting the machine learning model to our potential users' needs might be very fine tuned, meaning that the ML models produce only partially accurate responses. This would lead to an inferior user experience, and if the results are bad enough, it would lead us to annulling this feature. The repercussions of removing this feature would not be too severe as our platform provides other ways of food logging.

Table 2. Risks

Risk	Likelihood	Effect on the project	B Plan Summary
Not being able to acquire an API for food and grocery delivery services	Very Likely (90%+). Even if the companies have such a service, it is hard to imagine they would make that available only to us.	It would greatly add to the complexity of our work, and would take up more development hours to complete as a feature.	Scrapping the services with tools like Selenium. This plan is not ideal, but it is the only option if we are not able to acquire an API from such services.
Not being able to find databases with recipes, and more specifically, with Turkish recipes	Not so likely (30%). There seem to be many recipe databases online. It is only a matter of getting some Turkey relevant recipes as well.	We would have to dedicate time to build our own database and process data from many smaller sources. Would significantly increase development hours.	Scrapping Turkish cooking sites, and processing that data ourselves to make it compatible with our selected data storing format.
Not be able to set up an accurate ML model to do food recipe classification through pictures.	Not so likely (20%). We already did some research and there appears to be models which cover this functionality.	We would have to drop the feature as there is not really an alternative to ML food detection.	The alternatives are already built in as we will provide other features which allow for meal logging regardless of the success of this feature.

3.3 Project Plan

Table 3. List of work packages

WP#	Work package title	Leader	Members involved
WP1	Analysis Report	Ibrahim	Everyone
WP2	High-Level Design Report	Perman	Everyone
WP3	Mobile Application Development	Balaj	Gledis Ibrahim
WP4	Food Recommendation System	Khasmamad	Perman Gledis
WP5	Database Management System	Ibrahim	Khasmamad Balaj
WP6	Third Party Applications	Perman	Ibrahim Balaj

WP 1: *Analysis Report*			
Start date: *12/10/2020* **End date:** *09/11/2020*			
Leader:	*Ibrahim Elmas*	**Members involved:**	*Everyone else*
Objectives: *The analysis report contains a detailed analysis of the system to be developed. The report describes the functional and nonfunctional requirements as well as the use case scenarios. Object and Class Model and Dynamic Models are included together with screen mock-ups. Additionally, the report includes a discussion of consideration of various factors in engineering design, risks and alternatives, and project development plan.*			
Tasks: ***Task 1.1 Task planning*:** *The report is divided into chunks and each chunk is assigned to one or more team members. The deadlines are set for completion of the subtasks.* ***Task 1.2 First draft*:** *First draft of the report is produced.* ***Task 1.3 First draft peer review*:** *First draft is reviewed and changes for the final draft are discussed.* ***Task 1.4 Final draft I*:** *Final draft of the report is produced and sent to our supervisor for more feedback before submission* ***Task 1.5 Final draft II*:** *Final draft is polished further based on the feedback, submitted, and uploaded to the project website..*			
Deliverables ***D1.1:*** *Analysis report first draft* ***D1.2:*** *Analysis report final draft*			
WP 2: *High-Level Design Report*			
Start date: *November 16, 2020* **End date:** *December 21, 2020*			
Leader:	*Perman Atayev*	**Members involved:**	*Everyone else*
Objectives: *High-level system design achieves the transportation of the analysis model into system design model. The report includes the discussion of design goals, strategies, the proposed software architecture, and subsystem decomposition. Moreover, the report includes the discussion of consideration of various factors in engineering design such as public health, welfare, and social and economic factors.*			
Tasks: ***Task 2.1 Task planning* :** *The report is divided into chunks and each chunk is assigned to one or more team members.* ***Task 2.2 First draft*:** *First draft of the report is produced.* ***Task 2.3 First draft peer review*:** *First draft is reviewed and changes for the final draft are discussed.* ***Task 2.4 Final draft I*:** *Final draft of the report is produced and sent to our supervisor for more feedback before submission* ***Task 2.5 Final draft II*:** *Final draft is polished further based on the feedback, submitted, and uploaded to the project website..*			
Deliverables ***D2.1:*** *High-level design report first draft* ***D2.1:*** *High-level design report final draft*			
WP 3: *Mobile Application Development*			
Start date: *November 16, 2020* **End date:** *30 April, 2020*			
Leader:	*Balaj Saleem*	**Members involved:**	*Gledis Zeneli* *Ibrahim Elmas* *Perman Atayev*
Objectives: *This work package includes the development of the mobile application. The development process involves design and development of the user interface, backend development, integration with cloud servers and database system as well testing and deployment. QA (quality assurance) and testing are also part of the development process.*			

Tasks:
***Task 3.1 Strategy and planning*:** *Development strategy is determined and the work is divided into smaller chunks and assigned to members. Development tools are determined. Knowledge and skills gaps are identified and tackled.*
***Task 3.2 Backend development*:** *The logical workflow of the application is implemented. The communication with the backend servers and the database system is established.*
***Task 3.3 UI/UX development*:** *User experience priorities are determined. Screen mockups finalized together with navigational paths, palette and color story and complete typography. The design is transported into code.*
Task 3.4 Testing: *Test cases are prepared. The tests are performed and the working quality is evaluated. The bugs are identified and fixed and necessary improvements are made. The changes are tracked for the purpose of retesting.*

Deliverables
D3.1: *Screen mockups and navigational paths*
D3.2: *Iteration I - with limited functionalities*
D3.3: *Iteration II - with all the functionalities*
D3.4 *Final build - QA'ed, with all the functionalities*

WP 4: *Food Recommendation System*

Start date: *November 16, 2020* **End date:** *1 April, 2020*

Leader:	*Ibrahim Elmas*	**Members involved:**	*Khasmamad Shabanovi* *Balaj Saleem*

Objectives: The objective is to create a highly-personalized food recommendation system that serves as the backbone of our software. This is achieved by gathering a large volume of data and utilizing Machine Learning to process and categorize it.

Tasks:
Task 4.1 Strategy and planning *Development strategy is determined and the work is divided into smaller chunks and assigned to members. Development tools are determined. Knowledge and skills gaps are identified and tackled.*
***Task 4.2 Data collection and preprocessing*:** *Recipe and ingredients data are collected and prepared for further processing.*
***Task 4.3 Data processing with ML*:** *Collected data is processed and categorized by means of ML methods.*
***Task 4.4 API development*:** *APIs are developed and tested.*
***Task 4.5 Cloud deployment*:** *The system is deployed on a cloud server.*

Deliverables
D4.1: *Data*
D4.2: *APIs*

WP 5: *Database Management System*

Start date: *February 1, 2020* **End date:** *1 April, 2020*

Leader:	*Ibrahim Elmas*	**Members involved:**	*Khasmamad Shabanovi* *Balaj Saleem*

Objectives: *The objective is to create a database management system to store data in a way that supports improved information retrieval and modification and enhanced security. This is achieved by database design, implementation, and testing.*

Tasks:
Task 5.1 Strategy and planning *Development strategy is determined and the work is divided into smaller chunks and assigned to members. Development tools are determined. Knowledge and skills gaps are identified and tackled.*
***Task 5.2 Database design*:** *Database entities and the relationships among the entities are designed and modeled with an ER diagram.*
***Task 5.3 Implementation*:** *The design model is transported into code. The database management system is implemented and the data is loaded to the system,*
***Task 5.4 Testing*:** *The developed system is tested thoroughly.*

Deliverables
D5.1: *ER diagram*
D5.2: *Database management system*

WP 6: *Third Party Applications*

Start date: *February 1, 2020* **End date:** *1 April, 2020*

Leader:	*Perman Atayev*	**Members involved:**	*İbrahim Elmas* *Balaj Saleem*

Objectives: *The objective is to ensure the integration of third party food and grocery delivery applications.*

Tasks:
***Task 6.1 Strategy and planning*:** *Development strategy is determined and the work is divided into smaller chunks and assigned to members. Tools and data sources are determined. Knowledge and skills gaps are identified and tackled.*

Task 6.2 Survey of delivery applications**:** *A list of food and delivery applications is collected. The means of integration is determined.* ***Task 6.3 Integration*****:** *The integration of the proposed applications with our software is achieved.*
Deliverables ***D6.1:*** *Survey of delivery applications*

We also present the Gantt chart for our project plan below.

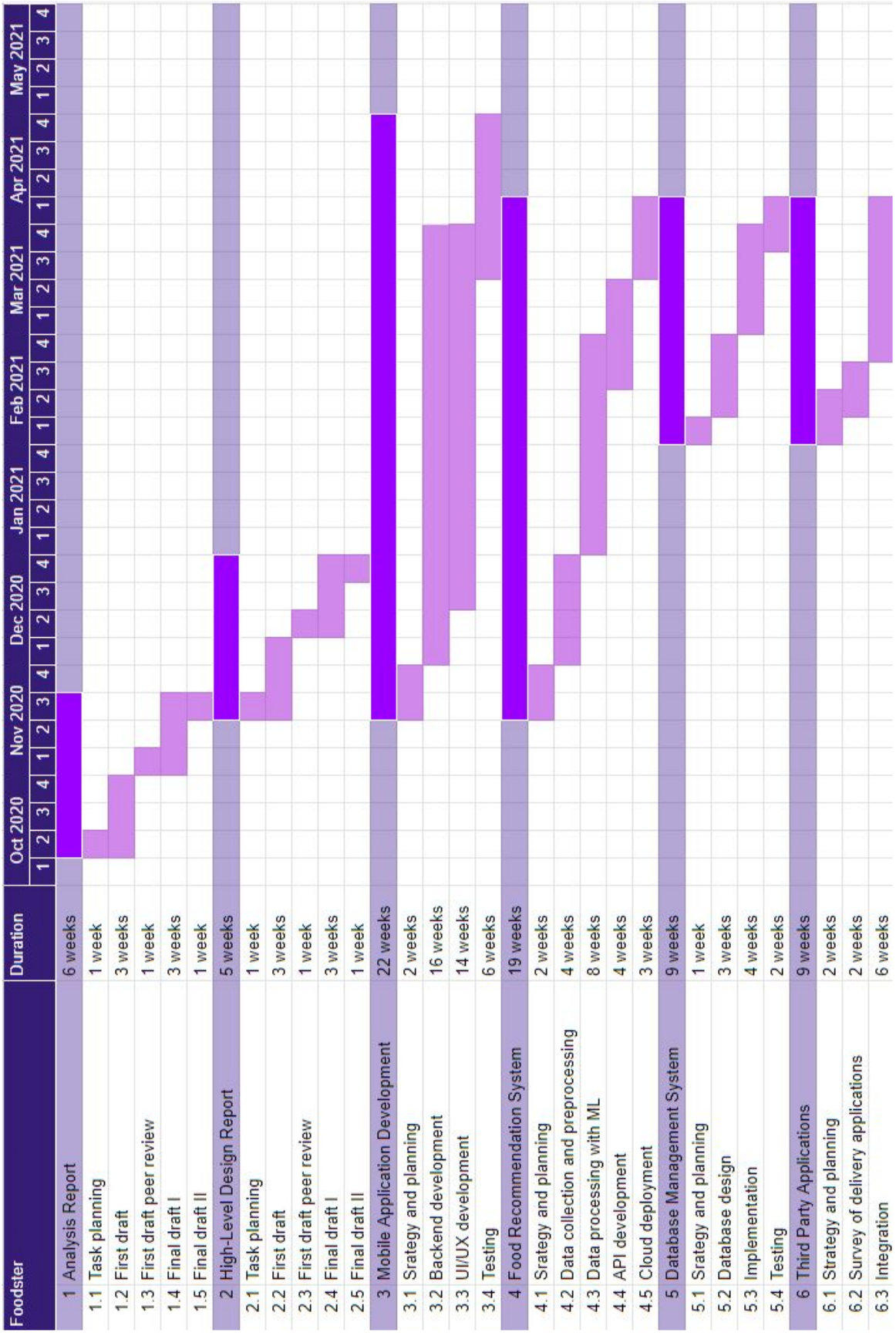

Figure 20. Gantt Chart

3.4 Ensuring Proper Teamwork

Modulation of tasks will be important to ensure proper teamwork. Having everyone deal with everything will add unnecessary complexity and will require everyone to learn many new technologies. On the other hand, splitting the work into self-sustained modules handled by groups of no more than 3 people will allow us to be specialized while providing adequate support to each other. Everyone will be assigned to lead one sub-team each.

Another key element will be our communication tools. We are looking into Jira to use as a task tracking tool. A private repository in Github will be used to coordinate our code. These tools will be properly set up and our mentor will be given access so they can be up to date with our progress and contributions. The majority of the work will be done asynchronously, but we will have joint discussion sessions to make sure everyone working on the same modules is on the same page.

3.5 Ethics and Professional Responsibilities

We will have an ethical and professional responsibility to safeguard the data of our users. Special care will have to go to ensuring that our meal recommendations are safe and do not risk the health of people who use our platform.

Professionally, we have a responsibility to provide a robust software platform which gives accurate tailored recommendations, in contrast to simply generating arbitrary recommendations which are loosely relevant to our users.

We might have to rely on data scraping to gather data and interface with the food/grocery delivery services. Data scraping could be ethically gray depending on whether the scraped entities want their data to be scrapped. To avoid any form of legal retaliation or ethical concerns we will try to limit ourselves to publicly available data, and not scrape from entities who would be harmed by having their data scraped.

3.6 Planning for New Knowledge and Learning Strategies

We will be required to acquire a great array of new technical skills including machine learning, Flutter development, data scraping with Selenium, and a large body of data processing techniques. However, modulation of our tasks will allow us to focus on only a subset of these skills. In addition, we will be assigned to our sub-teams so that to minimize the average new knowledge acquisition requirement per member. More experienced members with some of the technologies will be paired with others less experienced members to aid them in their learning.

After assigning the responsibilities, everyone will be given at least one week to familiarize themselves with the technology they will be working on through learning channels they are most comfortable with. This we believe will facilitate on the job learning. The majority of the learning will be done while doing the work to avoid unnecessary acquisition of knowledge which is outside the scope of our work.

4 References

MyFitnessPal, myfitnesspal.com.

Migros Sanal Market, migros.com.tr

Carrefour, carrefoursa.com/tr.

Getir, getir.com

YemekSepeti, yemeksepeti.com