Bilkent University

Department of Computer Engineering

Senior Design Project

Project short-name: **FOODSTER**

Project Specifications Report

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

12th October 2020

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 [1] which help the user reach their nutritional goals with the least possible effort.

1.1 Description

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[2], Carrefour[3], or Getir[4].

Alternatively, the user will be provided with the option to get food delivered directly from a restaurant, using yemeksepeti [5] 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 [6] 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.

1.2 Constraints

1.2.1 Economic Constraints

- Maintaining servers and releasing the software in the Google Play Store and IOS app store may have a one time or subscription costs [7].
- Servers may be required for machine learning models for user preference analysis and suggestion generation. These servers may incur variable costs.

1.2.2 Environmental Constraints

- The project does not affect any facet of the environment and hence its environmental footprint is zero.

1.2.3 Ethical Constraints

- All user's personal data, preferences, diet plans, financial data and all other such information will be encrypted and secured.
- Partners (such as Getir) will not be able to access more data than required for their service. In essence, the principle of least information will be followed.
- Sub-par and unhealthy suggestions for ingredients and health will not be made to users.

1.2.4 Implementation Constraints

- Flutter [8] will be used for cross-platform (Android and IOS) mobile app development
- Git will be used for version control [9].
- The backend will be implemented as a Node.js server to interface with client and database [10].
- Firebase [11] will be used as the database
- CI / CD principles [12] will be used hence services such as bitrise [13] may be used for cross platform testing and development

1.2.5 Reliability Constraints

- The application will build on both IOS and Android.
- The servers will have zero downtime after deployment.
- Only Updated and maintained third-party libraries for Flutter and Node.js will be used.

1.2.6 Social Constraints

- Health of users, will remain a priority and mission of the project throughout development.
- The application will be localized for Turkish and English speaking markets.
- Local businesses / shops that collaborate for ingredient / meal delivery will be considered equal partners and mutual growth will be encouraged.

1.3 Professional and Ethical Issues

- Privacy of any and all user data provided to the application may be an issue, however security of such data will be paramount throughout the development of the project. The user's will always have complete ownership and transparency of their entered data.
- Businesses such as Yemeksepeti, Getir, Migros and Carrefour may be hesitant to collaborate hence a mutually beneficial and profitable strategy would be required.
- Some users may be reserved with respect to meal-plan suggestions due to the personal nature of such suggestions. Hence care must be taken to make sure users are satisfied with the suggestions and recommendations.

2 Requirements

2.1 Functional Requirements

2.1.1 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.

2.1.2 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.

2.1.3 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.

2.1.4 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.

2.2 Non-Functional Requirements

2.2.1 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.

2.2.2 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.

2.2.3 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.

2.2.4 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.

2.2.5 Capacity

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

2.2.6 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.

2.2.7 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.

2.2.8 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.

2.2.9 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.

2.2.10 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.)

2.2.11 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.

2.2.12 Licensing

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

2.2.13 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.

2.2.14 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.

2.2.15 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.

2.2.16 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.

2.2.17 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.

2.2.18 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.

2.2.19 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.

References

[1] “Fitness starts with what you eat.,” myfitnesspal. [Online]. Available: https://www.myfitnesspal.com/. [Accessed: 11-Oct-2020].

[2] “Migros Sanal Market,” Migros Sanal Market: Online Market Alışverişi. [Online]. Available: https://www.migros.com.tr/. [Accessed: 11-Oct-2020].

[3] “Carrefour Online Market,” CarrefourSA. [Online]. Available: https://www.carrefoursa.com/tr/. [Accessed: 11-Oct-2020].

[4] Getir, “getir bi mutluluk,” *getir*. [Online]. Available: https://www.getir.com/. [Accessed: 11-Oct-2020].

[5] Online Food Order &amp; Delivery from Yemeksepeti.com. [Online]. Available: https://www.yemeksepeti.com/en. [Accessed: 11-Oct-2020].

[6] M. Serifovic, “Image-to-Recipe Translation with Deep Convolutional Neural Networks,” *Medium*, 19-Aug-2019. [Online]. Available: https://towardsdatascience.com/this-ai-is-hungry-b2a8655528be. [Accessed: 11-Oct-2020].

[7] N. Kharchenko, “The Hidden Costs of Developing A Mobile App You Should Be Aware of,” Dumb Little Man, 21-Mar-2018. [Online]. Available: https://www.dumblittleman.com/how-much-does-it-cost-to-create-an-app/. [Accessed: 11-Oct-2020].

[8] “Beautiful native apps in record time,” Flutter. [Online]. Available: https://flutter.dev/. [Accessed: 11-Oct-2020].

[9] “Version Control with Git,” *Git*. [Online]. Available: https://git-scm.com/. [Accessed: 11-Oct-2020].

[10] Node.js, “Node JS,” Node.js. [Online]. Available: https://nodejs.org/en/. [Accessed:

11-Oct-2020].

[11] “Firebase,” *Google*. [Online]. Available: https://firebase.google.com/. [Accessed: 11-Oct-2020].

[12] I. Sacolick, “What is CI/CD? Continuous integration and continuous delivery explained,” InfoWorld, 17-Jan-2020. [Online]. Available: https://www.infoworld.com/article/3271126/what-is-cicd-continuous-integration-and-continuous-delivery-explained.html. [Accessed: 11-Oct-2020].

[13] “Mobile Continuous Integration and Delivery,” Bitrise. [Online]. Available: https://www.bitrise.io/. [Accessed: 11-Oct-2020].