FEED ME

Systems Requirements and Analysis Specification

Version 1.0

Document Number: SAS-001

**Project Team: **A23****

**Project Members: **Peter Holoubek** (ph1335), **Teddy Kim** (tk1931), **JinZhao Su** (js9202)**

# 

# REVIEW AND APPROVALS

|  |  |  |  |
| --- | --- | --- | --- |
| Printed Name | Function | Date | Signature |
| Theodore Kim | Author | April 27, 2019 |  |
| JinZhao Su | Author | April 27, 2019 | https://scontent-lga3-1.xx.fbcdn.net/v/t1.15752-9/52861846_2336825663028189_6860368647928414208_n.png?_nc_cat=107&_nc_ht=scontent-lga3-1.xx&oh=8bb7d8bce3fedf38668fe3f9d05d7a6b&oe=5D258DEB |
| Petr Holoubeck | Author | April 27, 2019 | https://scontent-lga3-1.xx.fbcdn.net/v/t1.15752-9/s2048x2048/52825818_155163132053922_2270601179052376064_n.png?_nc_cat=107&_nc_ht=scontent-lga3-1.xx&oh=4b9107141ccddee8f9ebf13e5428e56c&oe=5CDB1CAF |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# REVISION HISTORY

|  |  |  |
| --- | --- | --- |
| Date | Revision Number | Purpose |
| April 27, 2019 | Version 1.0 | Initial Document Release |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

TABLE OF COTENTS

[1. INTRODUCTION 1](#_Toc7381159)

[1.1 Purpose 1](#_Toc7381160)

[2. SCOPE 1](#_Toc7381161)

[2.1 Identification 1](#_Toc7381162)

[2.2 Bounds 1](#_Toc7381163)

[2.3 Objectives 2](#_Toc7381164)

[3. SYSTEM OVERVIEW 3](#_Toc7381165)

[3.1 System Overview 3](#_Toc7381166)

[4. DOCUMENT OVERVIEW 3](#_Toc7381167)

[4.1 Document Overview 3](#_Toc7381168)

[5. REFERENCE DOCUMENTS 3](#_Toc7381169)

[6. BUSINESS REQUIREMENTS 4](#_Toc7381170)

[6.1 Technology 4](#_Toc7381171)

[6.2 Economics 4](#_Toc7381172)

[6.3 Regulatory and Legal 4](#_Toc7381173)

[6.4 Market Consideration 4](#_Toc7381174)

[6.5 Risks and Alternatives 5](#_Toc7381175)

[6.6 Human Resources and Training 5](#_Toc7381176)

[7. LOGICAL AND ARCHITECTURAL SPECIFICATION 6](#_Toc7381177)

[7.1 Context Diagram 6](#_Toc7381178)

[7.2 System Capability 7](#_Toc7381179)

[7.3 User Interface Requirements 10](#_Toc7381180)

[7.4 Logical Component (Component/Package/Subsystem) Architecture 16](#_Toc7381181)

[7.5 Class Diagram 17](#_Toc7381182)

[7.6 Class Relationship / Interactions Diagrams 18](#_Toc7381183)

[7.7 Events 18](#_Toc7381184)

[7.8 Activity / State (Scenario) Section 21](#_Toc7381185)

[7.9 State logic 21](#_Toc7381186)

[7.10 Behavior 21](#_Toc7381187)

[7.11 Dictionaries 23](#_Toc7381188)

[8. NON-FUNCTIONAL / OPERATIONAL SPECIFICATION 23](#_Toc7381189)

[8.1 System External Interface Requirements 23](#_Toc7381190)

[8.2 Safety Requirements 23](#_Toc7381191)

[8.3 Security and Privacy Requirements 23](#_Toc7381192)

[8.4 System Environment Requirements 23](#_Toc7381193)

[8.5 Computer Resource Requirements 23](#_Toc7381194)

[8.6 System Quality Factors 24](#_Toc7381195)

[8.7 Design and Construction Constraints 24](#_Toc7381196)

[8.8 Personnel-Related Requirements 24](#_Toc7381197)

[8.9 Training-Related Requirements 24](#_Toc7381198)

[8.10 Logistics Related Requirements 24](#_Toc7381199)

[8.11 Packaging Requirements 24](#_Toc7381200)

[8.12 Precedence and Criticality Requirements 25](#_Toc7381201)

[8.13 Other Requirements 25](#_Toc7381202)

[9 SYSTEM TEST PLAN REQUIREMENTS 25](#_Toc7381203)

[10 QUALITIFICATION PROVISIONS 25](#_Toc7381204)

[11 REQUIREMENTS TRACEABILITY 26](#_Toc7381205)

[12 EVOLUTION OF THE RAS 26](#_Toc7381206)

[13 RATIONALE 26](#_Toc7381207)

[14 NOTES 26](#_Toc7381208)

[15 APPENDICES 27](#_Toc7381209)

[15.1 Dictionaries 27](#_Toc7381210)

[15.2 UML Diagrams 28](#_Toc7381211)

[15.3 Schedule Tracking 29](#_Toc7381212)

[15.4 Defect Tracking 30](#_Toc7381213)

[15.5 Updated Gantt Schedule 30](#_Toc7381214)

## 1. INTRODUCTION

### Purpose

This document provides analysis of functional specifications for the *Feed Me* cross-platform application. Specifically, the document addresses user’s expectation for the application and further analysis for functional specifications of the system. These specifications are used for the developers to reconcile outcomes of the project. The components of the document act as assurance of the quality, verification and validation of the system such that it fulfills the requirements. In other words, it serves also as directives for the Software Quality Assurance (SQA) team. Further it breaks down the requirements into smaller functional units which can be divided into its technical behavior and operation specification. As part of the specification, the quality and human interaction standards and software quality standards will be definitive. Furthermore, this document serves as an agreement between the project client and project team on the requirements for the final system. Hence, the document serves as the foundation for communication between the two parties such that the final product delivered conforms to the client’s expectations.

## 2. SCOPE

The system being produced is the Feed Me cross-platform application. The Feed Me system includes a front-end client application and a back-end task server. The system should be able to make informed and accurate decisions about where and when a user should eat with minimal input.

One main point which was identified among potential customers is that traditional online food recommendation services require some amount of vetting and research as services such as Yelp may have ingenuine reviews and ratings. Therefore, a service that performed quality control and screening is a need among said consumers. Hence, the customer’s expectations are that the product *make a decision* for the user rather than just offer a search functionality through a database of restaurant options. This decision should require minimal user input and should be representative of their individual preferences. Finally, the product should product a social interaction component to allow different users to coordinate food decisions collaboratively.

The product can be used not only in the daily lives of individuals looking to eat out, but also o travelers looking to become familiar with local cuisines and restaurants looking to advertise their business to potential customers.

The goal of the system as a whole is to provide a food choice as good as if an individual made the choice themselves (unassisted).

### 2.1 Identification

This document is the Software Requirement and Analysis Specification (SAS-001). It is meant to guide the development and release of the first version of the *Feed Me* application. It is being submitted by Team A23. Version 1 was submitted to the project authority on February 27, 2019, Version 2 was submitted to the project authority on March 28, 2019 and Version 3 was submitted to project authority on April 28, 2019.

### 2.2 Bounds

The proposed system includes a means of making decisions for users based upon their individual preferences (which they provide to the product) and restaurant data provided by 3rd party application programming interfaces (APIs). It also provides a means for businesses to advertise to the product’s users and target audiences that have demonstrated behaviors and preferences compatible to said businesses. It is important to distinguish that the system DOES NOT track information about individual restaurants (location, food type, hours of operation, etc.) but rather receives that information from the use of APIs provided by other services which do track that information. The system simply aggregates said data and matches individual’s preferences to said restaurant choices.

### 2.3 Objectives

This project is will be delivered as a single, completed product at the designated delivery date (currently, ***to be determined***, as of Version 1 of this document).

This project is not critical; however, it should be treated with high priority because competitor research has revealed several other projects currently under development that are seeking to accomplish the same goals as the proposed product, therefore, development of the project should not be delayed significantly in order to gain some advantage in the potential market. See Table 1 for the project’s initial document deliverables and their descriptions and deadlines.

TABLE 1

Project Document Deliverables

|  |  |  |
| --- | --- | --- |
| Deliverable | Description | Deadline |
| Project Proposal | An outline of the project’s motivation, description of its requirements and constraints, description of its concept, and list of deliverables | February 16, 2019 |
| Software Business Specification | An outline of the business needs of the project, as well as a development of the product requirements and proposed decomposition of the product. | February 28, 2019 |
| Software Requirements Specification | Specification of the final requirements of the product, traceability to customer expectations and constraints, and verification plans for said requirements | March 26, 2019 |
| Software Project Management Plan | Definition of the project organization and proposed management of the project including planning for workflow, risk management, and technical management. | April 4. 2019 |
| Software Analysis Specification | A breakdown of the proposed product and its requirements decomposed into smaller functional units. A technical specification for proposed behavior and operation. Also a definition of acceptable software quality and human / environment interaction standards. | April 23, 2019 |
| Software Design Document | Final document for scope of this project | March 9, 2019 |

## 3. SYSTEM OVERVIEW

### 3.1 System Overview

The proposed product is not a standalone system but is dependent on external products and services to achieve its full functionality. Specifically, it is dependent on 3rd party food review APIs and services (i.e. Google Reviews and Yelp) for its restaurant information and review dataset as well as user’s input to help shape the service to specifically match each person’s individual preferences. The interfaces with external parties are further explored in the context diagram (Section 7.1).

The primary users would be young, urban professionals and students whom, during customer research, were identified as being most likely to eat out regularly for their meals (rather than cooking). Furthermore, they are more likely to eat socially (i.e. with other individuals) and consult online services in determining where to eat. They are often technologically savvy and early adopters of new software products. A key factor in their use of technology is its incorporation into their daily routines without too much of their input. However, they are also reticent to share too much personal information on an internet based service (information privacy).

Another potential user of the product are businesses (restaurants). Restaurants often suffer from low lunch throughput, therefore being able to promote their businesses to a targeted audience which may result in new customer acquisition / sustained income would be useful to said users. Businesses are often represented by either an individual (or group) solely responsible for the marketing of said business or directly by their owners / operators. These individuals are often price conscious, but open to services which provide unique advantages for growth.

## 4. DOCUMENT OVERVIEW

### 4.1 Document Overview

This document is the first version of the Software Requirement and Analysis Specification. It is meant for the development and overview of functional specifications of the release of the first version of the Feed Me application. The remainder of this document includes fifteen sections and an appendix.

The next section describes the documents referenced during the creation of the SAS. Section 6 includes the requirements for the business development of the project (i.e. requirement no relating to the development of the software product). Section 7 outlines the functional and non-functional requirements for the product, and the use cases through which said requirements could be mapped onto. Section 8 outlines how the product will be tested and the requirements for that testing process. Section 9 outlines how this document will be reviewed to ensure adherence to quality standards. Section 10 traces the requirements specified in this document backwards and forwards in the project lifecycle. Section 11 describes how this document will develop as the system develops and the project is matured. Section 12 describes the rationale for the specified requirements. Finally, Section 13 includes any notes made by reviewers on the current document to take into consideration in future revisions. The last portion of the document are the appendices which include auxiliary tables related to the development of this document.

## 5. REFERENCE DOCUMENTS

Project Proposal for Feed Me, Version 1 (February 10, 2019): Theodore Kim, JinZhao Su, Petr Holoubek.

System Requirements Specification for Feed Me, Version 2 (April 2, 2019): Theodore Kim, JinZhao Su, Petr Holoubek.

Software Project Management Plan for Feed Me, Version 1 (April 9, 2019): Theodore Kim, JinZhao Su, Petr Holoubek.

## 6. BUSINESS REQUIREMENTS

### 6.1 Technology

Feed Me application is thick client-server application powered by machine learning paradigm. The application will continuously update itself through self-learning algorithms for the benefit of the most optimal food choice for the user.

### 6.2 Economics

Feed Me application deems itself as a direct competitor to the Google, Yelp and other small startups. Yelp is the most direct competitor. Aggregate valuations of food recommendation systems (FRS) is approximately $5B market. The market has low cost of entry and therefore is highly sensitive to competition. The main market for the FRS, but not limited to, is densely populated urban areas such as New York City, Chicago and Los Angeles, to name few. FRS thrives in these areas due to immersive amount of restaurants and limited capabilities of a user to make the most optimal choice.

### 6.3 Regulatory and Legal

FRSs are data-driven applications calculating past behavior of an user and average consensus among the designated bucket of users. Therefore, Feed Me application has to comply with general outline of regulatory framework for data privacy in the United States and rest of the world. Particular focus must be paid to General Data Protection Regulations erected in 2018 in the European Union. Furthermore, Republic of China requires that any stored data about its citizens must be stored at servers placed within the borders of the nation.

### 6.4 Market Consideration

FRS has low cost of entry into the market. It follows that FRS is prone to cycles where abundance of new entrants will lead to the lowered profit margins but eventually increase, ironically, due to the abundance. The stickiness of website to the user is from average to high as FRS becomes more tailored to the user as the time of use progresses. Further consideration must be aimed at the market itself. The major segment is typical highly populated urban area with many food and beverage places around. The second major segment is aimed at tourists who travel across nations and are prone to taste local food.

### 6.5 Risks and Alternatives

Table 2 represents the risks currently identified for the system.

TABLE 2

Identified System Risks

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Type | Risk | Proba-  bility | Discovery | Responsible Party | Mitigation Strategy | Impact | Status |
| **T** | System does not return an accurate decision (i.e. user does not like choice) | HIGH | Proposed during discussion of system concept | Software developers | The algorithm will learn from poor choices and improve itself | LOW | Identified |
| **O** | Developers do not have the knowledge to create the product | LOW | Proposed during discussion of project proposal | Software developers | The software team will divide up functional responsibilities early and research before development | HIGH | Being investi-gated |
| **Ec** | 3rd Party companies revoke use of their APIs because we are a business | LOW | Proposed during SRS Revision | Business Team | Prior to development, the project team will contact said companies and receive authorization to use their APIs (with credit) | HIGH | Identified |
| **Env** | The initial userbase does not utilize the application enough: not enough data is collected to make the app effective | HIGH | Proposed during revision of SRS | Business Team | Adequate customer research and canvasing is conducted to encourage early adoption of the product | HIGH | Identified |
| **T** | Assumptions made regarding existing technologies are incorrect (see 3.2.4) | LOW | Proposed during discussion of system concept | Software developers | Adequate research must be done to ensure that services exist to support application functionality | HIGH | Being investi-gated |
|  |  |  |  |  |  |  |  |

Types of risk are technological (T), environmental (Env), organizational (O), and economic (Ec).

### 6.6 Human Resources and Training

Feed Me application is made for the public at large and therefore must have highly intuitive user experience design. Human resources will be primarily composed from Technical Support and Client Support. Each staff must be trained on how to use the Feed Me application but also on auxiliary faucet software in order to gather more appropriate information on incurring client. Furthermore, the staff must be trained on data-privacy and procedures how to verify identity but also what kind of information can and cannot be exposed to the user.

## 7. LOGICAL AND ARCHITECTURAL SPECIFICATION

### 7.1 Context Diagram

Figure 1 is the context diagram for the Feed Me Application. Notice that the application interfaces with external APIs to get its restaurant information as well as users and businesses to provide the feed-back and decision parameters for the system. The project utilizes an Object-Oriented methodology.

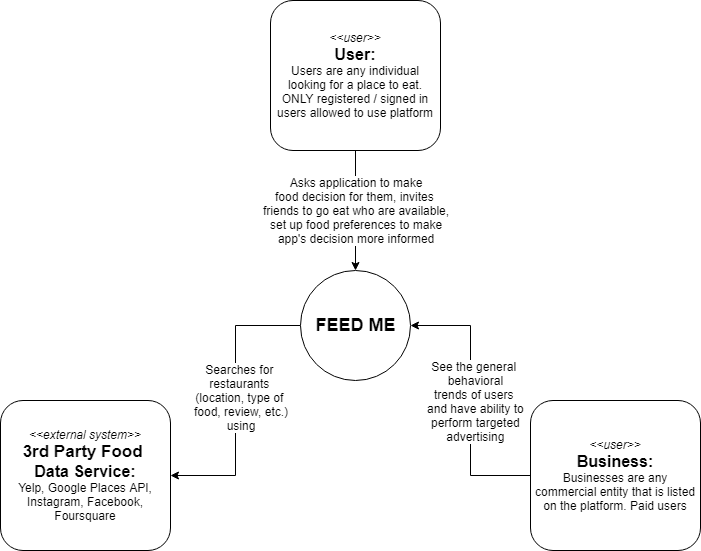


FIGURE 1: Context Diagram

### 

### 7.2 System Capability

#### 7.2.1 Capabilities

The following are the functional requirements and specification for the system as defined in the System Requirements Specification (identifiers precede the requirement with the prefix FSR, **F**unctional **S**ystem **R**equirement):

***FSR1) The system shall provide a means for users to define their individual food preferences.***

FSR1.1) The system shall give the user this option upon registration.

FSR1.2) The system shall give the user the option to change or update their preferences after initial registration.

FSR1.3) The system shall present this feature as a survey where each user will provide information as: a choice from a set of predefined answers, a quantitative value, or range of values.

FSR1.4) To ensure that the system makes accurate recommendations, users will be required to complete the survey when they register.

FSR1.5) The system shall use the user’s answers to further refine its decision-making process.

***FSR2) The system shall provide users with personalized recommendations for restaurants.***

FSR2.1) The system shall present this feature as a single action user interface element (i.e. a button).

FSR2.2) The system shall present this feature as the primary action within the product (i.e. presented to the user by default).

FSR2.3) The system shall provide five recommendations to the user.

FSR2.4) The system shall allow the user to decline or accept recommendations using a two action user interface element (i.e. two buttons, swiping, etc.).

FSR2.5) The system shall record the user’s action and use it to update its decision making algorithm.

FSR2.6) The system shall record the user’s location in order to make a more personalized decision based on nearby locations as well as track whether the user went to the restaurant.

FSR2.7) The system shall provide the same recommendations upon multiple uses of the feature (i.e. pressing the button multiple times within a small time period results in the same recommendations being generated).

***FSR3) The system shall interface with external APIs to gather data about restaurants compatible to each user.***

FSR3.1) The system shall automatically reference APIs when the users request a recommendation.

FSR3.2) The system shall utilize multiple APIs and accumulate the collected data into a single dataset.

***FSR4) The system shall allow users to maintain a public profile with the service.***

FSR4.1) The system shall allow users to create an account with the system.

FSR4.2) The system shall allow users to import account information from 3rd party platforms.

FSR4.3) The system shall enforce secure password policies when users create an account with the system.

FSR4.4) The system shall allow users to upload a profile picture.

FSR4.5) The system shall verify users with their email address or phone number.

FSR4.6) Users should be searchable by their profile’s username or email address.

FSR4.7) Users shall be allowed to log into their profile from multiple accounts.

***FSR5) The system shall allow users to add other users as friends.***

FSR5.1) The system shall allow users to search other users by their profile’s username or email address.

FSR5.2) The system shall allow users to send friend invitations to other users.

FSR5.3) The system shall allow users to accept or decline pending friend invitations.

FSR5.4) The system shall allow users to view the status of their friend invitations.

***FSR6) The system shall allow users to invite other users to eat out with them.***

FSR6.1) The system shall alert users with a notification when they are invited to eat out.

FSR6.2) The system shall check to see if other users are available to eat out by checking whether the invited user has already asked for a suggestion (implying that they had already eaten).

FSR6.3) The system shall recommend friends that are nearby to the user when the user asks for a food recommendations.

***FSR7) The system shall allow businesses to view generalized marketing data on the behavior of users using the product.***

FSR7.1) The system shall show businesses summary statistics on users that fit their business profile.

FSR7.2) The system shall allow businesses to create profiles for their business defining characteristics of the business (in the form of a survey).

FSR7.3) The system shall charge businesses to use its reporting services.

FSR7.4) The system shall not show data on individual users to businesses.

FSR7.5) The system shall only show relevant user data based on location, food preferences, budget, etc.

***FSR8) The system shall provide businesses a means of advertising on the platform to users within their target audience by offering special discounts and promotions to attract customers.***

FSR8.1) The system shall allow businesses to create custom advertisement listings to be shown to relevant users.

The following are the use cases of the proposed application:

* Login
* Registration
* Add Friend
* View Friends
* Invite Friends
* Take Initial Survey
* View Customer Trends
* Create Customer Advertisement

#### 7.2.2 Use Case Diagrams

Figure 2 is the Use-Case Diagram for the system.

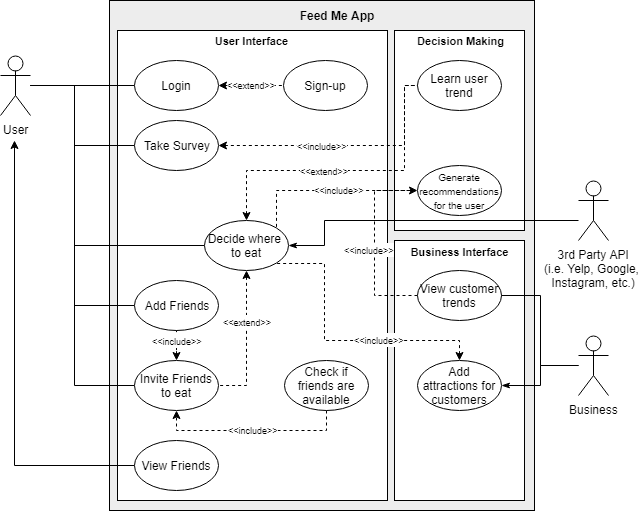
**

Figure 2: System Use Case Diagram

#### 7.2.3 Use Case Descriptions

#### 7.2.2 Use Case Descriptions

|  |  |  |
| --- | --- | --- |
| **User Registration** | | |
| **Description** | This use case provides a means for users new to the system to create a profile on the system (FSR4). | |
| **Pre-conditions** | The user performing this action does not already have an account on the service. | |
| **Flows** | Basic or Normal Flow | 1. User chooses option to create a profile on the system. 2. System redirects user to user registration screen 3. User provides email address, username, and password for new account. 4. System adds account information to database, sends verification email to provided email address. 5. User confirms email address. 6. System marks user account as verified. |
|  | Alternative Flows | 1. *Account exists at given email:*   If, in Step 3, the user provides an email address already associated with an account the system will prompt the user to use another email (repeat Step 3) or login with that email (User login Use Case)   1. *Password does not meet security policy*   If, in Step 3, the provided password does not meet the predefined security policy, the system will prompt the user to enter a new password (repeat Step 3)   1. *Missing Verification email address*   After Step 4, if the user does not receive the verification email, the system will provide the option to resend the email, or change the email address.   1. *Unverified Account*   If the user does not verify their email address, they will not be allowed to log into their account and the use case effectively ends. |
| **Post Conditions** | The user will be allowed to log into their account using the login use-case. | |
| **Special Requirements** | None | |
| **Extension Points** | None | |

|  |  |  |
| --- | --- | --- |
| **User Login** | | |
| **Description** | This use case provides a means for existing users on system to log into their saved profile (FSR4). | |
| **Pre-conditions** | The user performing this action already has an account on the service. | |
| **Flows** | Basic or Normal Flow | 1. Upon opening the app, the user is prompted to login 2. User provides email address and password for existing account. 3. System checks provided information against information in the database. 4. System informs user of successful login and redirects the user to the main system page. |
|  | Alternative Flows | 1. *User forgot their password:*   If, at Step 2, the user does not remember their password, the system will provide an option for the user to indicate such. An email will be sent to the provided email with a link to reset their password. If a account exists at that email, the user may click the link and set a new password for their profile.   1. *Account does not exist with given credentials.*   If, in Step 3, the provided email and password does not match an existing profile within the system, the system shall return an error code and require the user to repeat Step 2.   1. *First time login.*   After Step 4, if the user has never logged into the system before (i.e. it is a new account), the system will redirect the user to the Survey Use Case rather than the main landing page. |
| **Post Conditions** | After login, the user will have access to the full functionality of the system. | |
| **Special Requirements** | None | |
| **Extension Points** | At the first login prompt, the user can instead choose to sign up for a new account | |

|  |  |  |
| --- | --- | --- |
| **User Survey** | | |
| **Description** | This use case provides a means for users to complete a survey in which they can define their food preferences in order to personalize their recommendations (FSR1). | |
| **Pre-conditions** | The user performing this action must be logged into the system via the login use case. Furthermore, the user is automatically entered into this use case if they are new to the system or can manually enter into this use case later via an option within the system. | |
| **Flows** | Basic or Normal Flow | 1. The user is presented the survey page 2. The system generates questions and answer choices for the user based upon the factors which it uses to make its decisions. 3. The user answers each question (the user must provide an answer, questions cannot be skipped). 4. Upon completion of the survey, the user will be returned to the main landing page. |
|  | Alternative Flows | 1. *The user has already completed the survey:*   If, in Step 3, while the user is answering the questions. If the user had already answered the survey (i.e. the user had entered into the use case manually), the user will be presented the option to skip questions that they have already answered.   1. *The user does not complete the survey*   If, in Step 3, the user exits the system without completing the survey and there are unanswered questions left, the user will be prompted to enter said answers the next time they enter the system (i.e. they will re-enter the use case). |
| **Post Conditions** | The system will have updated the user’s preferences. | |
| **Special Requirements** | None | |
| **Extension Points** | None | |

|  |  |  |
| --- | --- | --- |
| **User Decision Making** | | |
| **Description** | This use case provides a means for users to query the system to generate restaurant recommendations for the user. | |
| **Pre-conditions** | The user performing this action must be logged into the system via the login use case. Furthermore, the user must have completed the survey use case at least one as well. | |
| **Flows** | Basic or Normal Flow | 1. The user is selected the option to initiate the decision-making process. 2. The system utilizes information gathered on the user from the survey and previous decisions, to generate a set of parameters for desirable recommendations. 3. The system queries external APIs with said parameters to generate five recommendations for the user. 4. The user is presented the recommendations and chooses whether to accept or reject each recommendation. 5. The system records the user’s actions and use it to redefine better decision making for the user for the next recommendation. |
|  | Alternative Flows | 1. *External APIs are not available:*   If, in Step 3, a portion of APIs utilized by the system are not available, the system will utilize the active APIs and inform the user that they decision may not be optimal. If no external APIs are available, the system returns an error to the user and ends the use case.   1. *No recommendation can be made*   If, in Step 3, the system cannot generate any recommendations for the user, random restaurants the search parameters will be relaxed until some restaurants are found.   1. *The user rejects all recommendations.*   After Step 4, if the user rejects all the recommendations, they are not allowed to ask for more recommendations, a sufficient amount of time (defined later) must pass between recommendation queries.   1. *The user is matched with an advertising business.*   If one of the user’s recommendations is a business that is advertising on the platform, the user will be shown that it is an advertisement and shown the text for the ad. |
| **Post Conditions** | The system will have | |
| **Special Requirements** | None | |
| **Extension Points** | When the user is shown the option to initiate the decision making feature, they will have the option to invite recommended friends to eat with them (thus altering the decision making process) | |

|  |  |  |
| --- | --- | --- |
| **User Adding Friends** | | |
| **Description** | This use case provides a means for users to add other users on the system as friends. | |
| **Pre-conditions** | The user performing this action must be logged into the system via the login use case. The user enters into the use case via a manual option within system. | |
| **Flows** | Basic or Normal Flow | 1. The user enters a user name or email address for the desired user into the search query box. 2. The system searches for users that match those parameters. 3. The user is presented a list of matching users. 4. The user chooses one of the users to request as a friend. |
|  | Alternative Flows | 1. *Location based search:*   If, in step 1, the user chooses to, they can search for users near them who have enabled location tracking for their account rather than providing a search string.   1. *No users are found that match the search query.*   If, in Step 2, the system cannot match any users within the system with the information provided, the system will return an empty list and allow the user to readjust they search query such that users are found.   1. *The invited user has already sent the current user a friend request.*   If, in Step 4, the user chooses to send a friend request with a user that has already sent them a friend request (i.e. if User A sends a request to User B, but User B already sent a request to User A) the two users are automatically made friends. |
| **Post Conditions** | The system will record the friend request and notify the other user of the pending invitation. | |
| **Special Requirements** | None | |
| **Extension Points** | None | |

|  |  |  |
| --- | --- | --- |
| **User Managing Friend Requests** | | |
| **Description** | This use case provides a means for users to view pending user friend requests and either decline or accept friend requests. | |
| **Pre-conditions** | The user performing this action must be logged into the system via the login use case. The user must have received a friend request from another user to be able to use this use case. The user can enter into this use case via an option within the system. | |
| **Flows** | Basic or Normal Flow | 1. The user managing their friend requests is resented with a list of pending invitations for the system. 2. The user can choose to either accept or reject each request. 3. The system will update the friends list of both the current user and the user that sent the request, after the current user chooses an action. 4. The user that sent the request will be notified of the current user’s decision. |
|  | Alternative Flows | 1. *No pending requests:*   If, in step 1, the current user has no pending requests, the user is presented an empty list. |
| **Post Conditions** | The system will have updated the user’s friend lists | |
| **Special Requirements** | None | |
| **Extension Points** | None | |

|  |  |  |
| --- | --- | --- |
| **User Inviting Friends to Eat** | | |
| **Description** | This use case provides a means for users to invite other users (that are accepted friends of the current user) to eat with the current user. | |
| **Pre-conditions** | The user performing this action must be logged into the system via the login use case. Furthermore, the user may only perform this action with another user that is a friend of the current user. This action can be entered manually as an extension of the decision making process. | |
| **Flows** | Basic or Normal Flow | 1. The user selects the option to invite friends 2. The system will generate a list of recommended friends based on location, if they already ate, etc. 3. The user selects a user from that list to invite to each 4. The target user is sent a notification. 5. The current user is returned to the list of users 6. The current user exits from the use case when it selects the UI option to exit the current screen. |
|  | Alternative Flows | 1. *User searches for non-recommended friends:*   If, in Step 2, the user wants to select a friend not on the recommended list, the app will allow the user to search for specific friends by username.   1. *The target user rejects the invitation*   If, in Step 4, the target user rejects the invitation, the current user is notified and the decision algorithm is altered. Otherwise, by default, the decision is made with the assumption that the invited user will attend. |
| **Post Conditions** | The system will use the invited friends to alter the decision maker algorithm. Following exit from this use case, the user is returned to the decision making use case. | |
| **Special Requirements** | None | |
| **Extension Points** | None | |

|  |  |  |
| --- | --- | --- |
| **Business Viewing User Summary Information** | | |
| **Description** | This use case provides a means for subscribed businesses to view summary statistics on relevant users. | |
| **Pre-conditions** | The business using this feature must have a normal, verified user account and be logged in. They can select this option under the business features within the system. | |
| **Flows** | Basic or Normal Flow | 1. The business is presented with the base dataset with filtering options. 2. The business can adjust the base dataset with the filtering options. 3. After applying the filtering options, the system prompts the business for payment of a microtransaction. Applying different filters require different amounts of microtransactions. 4. The system shows the dataset with the applied filters. |
|  | Alternative Flows | 1. *Business has already paid microtransactions:*   If, in Step 3, the business has already paid a microtransaction for the given set of filtering options, they will be shown the dataset with the given filters without needed to pay an additional fee.   1. *Microtransaction fails*   If, in Step 3, the microtransaction fails, the business must reenter payment details, otherwise the use case exits. |
| **Post Conditions** | The system will have updated the user’s preferences. | |
| **Special Requirements** | Businesses will need to pay a small fee to see information. Fees are scaling based upon the resolution of data (with the base dataset being free). | |
| **Extension Points** | This use case is extended by the adding advertisement use case. Upon filtering data, an option will be presented to create an advertisement towards that filtered audience, leading into the next use case | |

|  |  |  |
| --- | --- | --- |
| **Business Adding Advertisements** | | |
| **Description** | This use case provides a means for businesses to create advertisements towards specific user audience groups. | |
| **Pre-conditions** | The business must have first made a normal, verified user account and be logged in. Next, they must have paid for a filtered user summary from the previous use case and selected the option to create an advertisement for that audience. | |
| **Flows** | Basic or Normal Flow | 1. The business is presented the summary statistics for the user audience they are targeting as well as a place to enter their business details and advertisement text. 2. The business enters their information. 3. The system presents the business with a summary / preview of their advertisements. 4. The business enters billing information to be billed a small fee every time the ad is shown to a user. 5. Upon transaction completed the business is sent a receipt and the system adds the ad to the invoice. |
|  | Alternative Flows | 1. *Billing information fails:*   If, in Step 4, the provided billing information is incorrect, the business is prompted to reenter their information, or the use case exits.   1. *Cancels the transaction*   At any point, the business can cancel the transaction and return to the previous use case screen.   1. *Existing ad*   If, at Step 2, the business information is already associated with an ad given the selected audience profile, the business will be prompted to “change” the currently running ad rather than create a new ad, or cancel the current ad. |
| **Post Conditions** | The system will contain the new ad to be shown to users | |
| **Special Requirements** | None | |
| **Extension Points** | None | |

### 7.3 User Interface Requirements

The most important requirement of the user interface is for it to be as simple as possible. Users of the system must be able to access the primary functionality of the product in a single button click (FSR2.1). The user interface must be clean and feature a consistent color scheme and layout theme. The simple user interface ensures that users do not get confused by the application and would be more likely to use it again.

### 7.4 Logical Component (Component/Package/Subsystem) Architecture

#### 7.4.1 Component Descriptions

The following components are a part of the Feed Me Application system:

* The User Interface: an application hosted on the user’s device, which communicates with the server to accept user input and display information back to the user.
  + The user interface is components of three primary views: login / registration page (where users can login into their account or create a new account), the main page (where the primary app action occurs as well as where the users can manage their friends) and the settings page, where users can alter the application’s behavior and their own preferences.
  + The application uses a thin-client architecture.
* The Application server: the central component of the application, the application server collects the outputs of the application components and feeds it into the other components as needed. For example, for the login use case, it receives the credentials from the user interface, and compares them to the contents of the database, and indicates to the user success or failure. The application server also interfaces with the 3rd party API (according to the context diagrams).
* The database: the database contains information about the users on the application. The database hold users’ login credentials and food preferences to be fed into the machine learning model during the decision-making process.
* The machine learning model: the machine learning model is responsible for taking the information provided about the user during registration and selecting good restaurant recommendations for said users. This component is also responsible for improving itself based upon the information collected about the user’s usage of the application.

#### 7.4.2 Component Architecture Diagram

See Figure 3 for the Component Architecture Diagram.

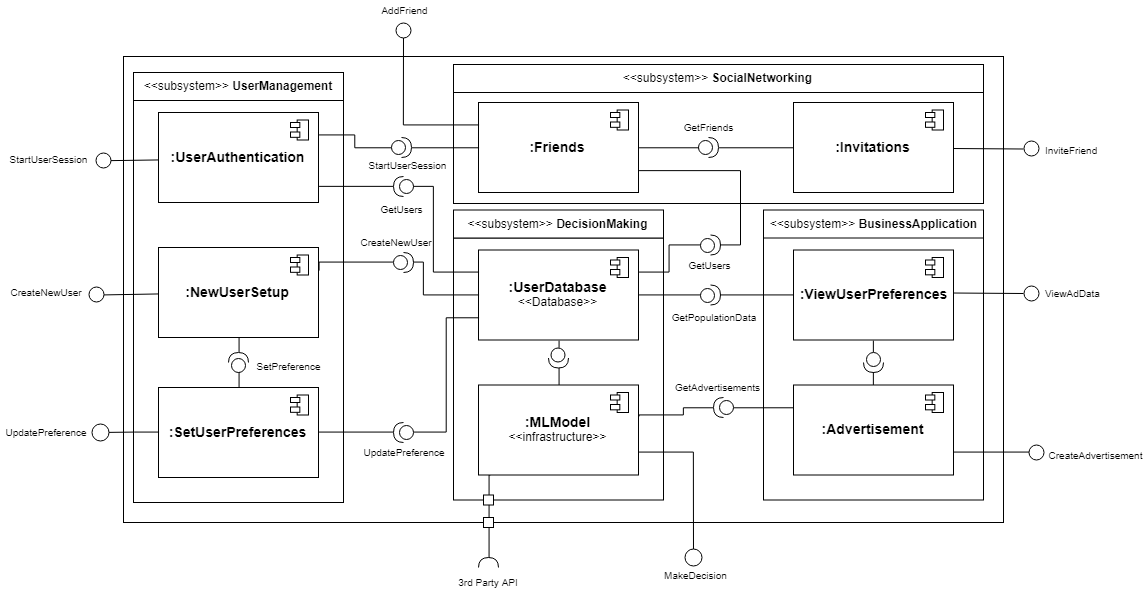


Figure 3: Component Diagram

### 7.5 Class Diagram

See Figure 4 for the UML Class Diagram.

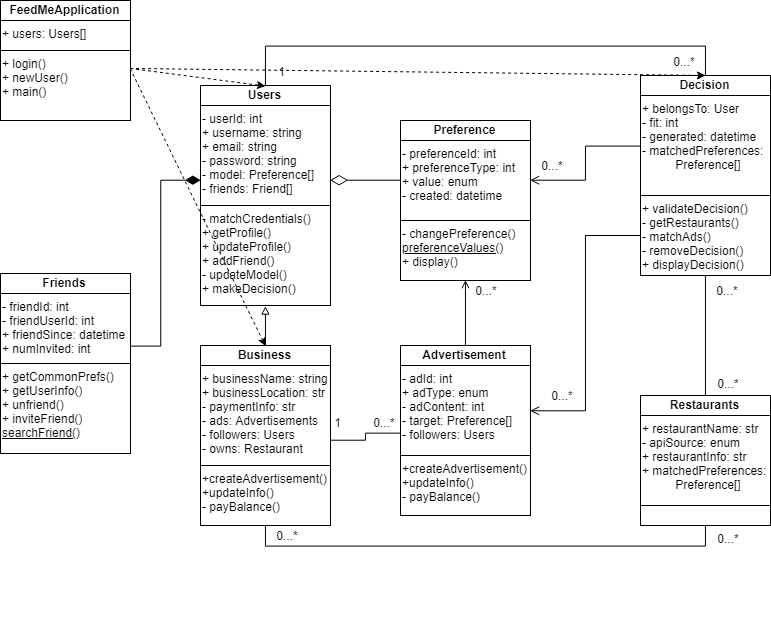


Figure 4: UML Class Diagram

### 7.6 Class Relationship / Interactions Diagrams

See Figure 4 for the class interactions and relationships.

### 7.7 Events

The following events have been identified in the proposed product:

* User starts new session
* User creates a new account
* User updates their profile with new preferences
* User links account with business
* Business takes ownership of restaurants
* User initiates a new decision making process
* User rejects or accepts decisions
* User searches for another user to friend
* User adds a friend
* User accepts friend request
* User invites friend to eat
* Business views preferences
* Business creates advertisement to match preferences

#### 7.7.1 Motives

* *User starts new session*: a user logs onto the system and establishes a session with the server. This would be done when an existing user wishes to use a new device
* *User creates a new account:* a user would want to create a new account if they do not already have an account with the system.
* *User updates their profile with new preferences*: users would update their profile if their tastes change or they are finding their decisions unsatisfactory
* *User links account with business*: The user can upgrade their account to a business account, allowing them access to the enterprise (advertisement) features. Users wishing to use the platform for promotion would initiate this event.
* *Business takes ownership of restaurants*: once a user upgrade their account, they can take ownership of their business if it is listed on the service in order to link their advertisements.
* *User initiates a new decision-making process*: the primary purpose of the proposed application. The user can initiate this event in order to receive a restaurant recommendation based upon their preferences.
* *User rejects or accepts decisions*: The user can choose to reject or accept a decision in order to improve future suggestions.
* *User searches for another user to friend*: users can search through other users by username or email to add them as a friend.
* *User adds a friend:* users can add friends based upon their searches. Added users still need to be accepted by the other party.
* *User accepts friend request*: users can accept other friend requests from other users. Accepted friends can then invite each other to eat.
* *User invites friend to eat:* users can invite other users to eat and doing so will alter the decision making process.
* *User remove friend:* Some people fall out of friendship. So it is natural they would unfriend them on the application too.
* *Business views preferences:* businesses can view the users’ preferences that patronize their restaurants.
* *Business creates advertisement to match preferences:* businesses can use the preferences that they view before in order to create a targeted advertisement.

#### 7.7.2 Event Diagrams

Table 3 has the event table for the proposed application.

TABLE 3

Event Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Event** | **Initiator** | **Input** | **Use Case** | **Output** | **Recipient** |
| New Session | User | Email and password | Login | User profile | User |
| New Account | User | Email, password, confirmation | Registration | New user profile. | User |
| Update Profile | User | Profiles | Take survey | Updated preferences | User |
| Link Account | User | Business and payment info | Adding Advertisement | Upgraded user account | Business |
| Take ownership | Business | Business information and proof of ownership | Adding Advertisement | Business association | Business |
| Initiate decision | User | User preferences, advertisements, state data (i.e. time of day and location) | Decision making | 5 decisions for the user | User |
| Reject or accept decision | User | Decisions | Decision Making | Feedback for the decision backend | Decision (backend) |
| Search for users to friend | User | Search query (by username or email) | User managing friend requests | List of users matching query | User |
| Send friend request | User | User identifier (username or email) | User managing friend requests | Friend request | Friend |
| Accept friend request | Friend | Friend request identifier | User managing friend requests | Accepted friend | User |
| Remove friend | User | Friend identifier | User managing friend requests | Removed friend | Friend |
| Invite friends to eat | User | Friend list | Inviting friends to eat | Invitation | Friend |
| View user preferences trends | Business | Query information (what preferences, locations, times) | Use summary information | Summary of user preferences | Business |
| Create advertisement | Business | Matching preferences, ad type, ad content | Adding advertisement | Advertisement added to the database | Decision (backend) |

### 7.8 Activity / State (Scenario) Section

Figure 5 is the activity and state diagram for the proposed application.

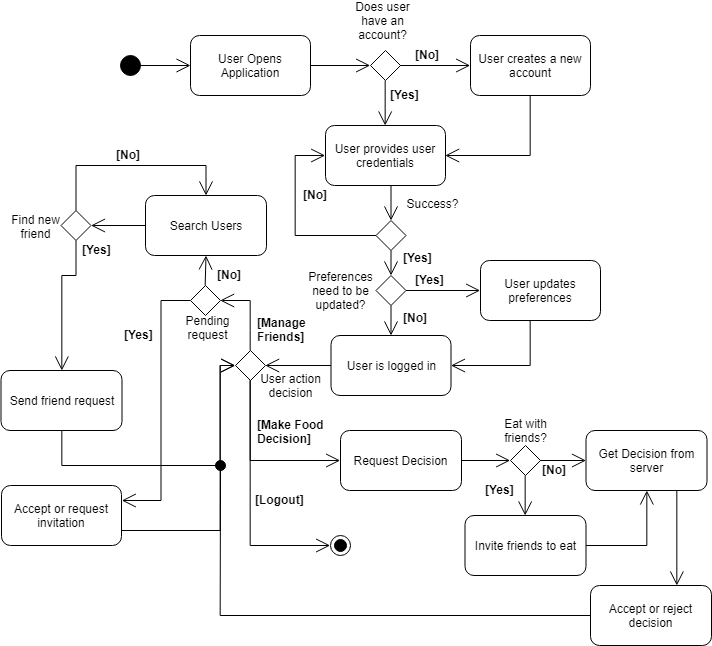


Figure 5: Activity Diagram

### 7.9 State logic

See activity diagram (Figure 5) for a description of the states and their transitions. This is further refined in later documents and developments of the project and proposed product.

### 7.10 Behavior

#### 7.10.1 Sequence Diagrams

See Figure 6 for the UML Sequence Diagram from each use case.

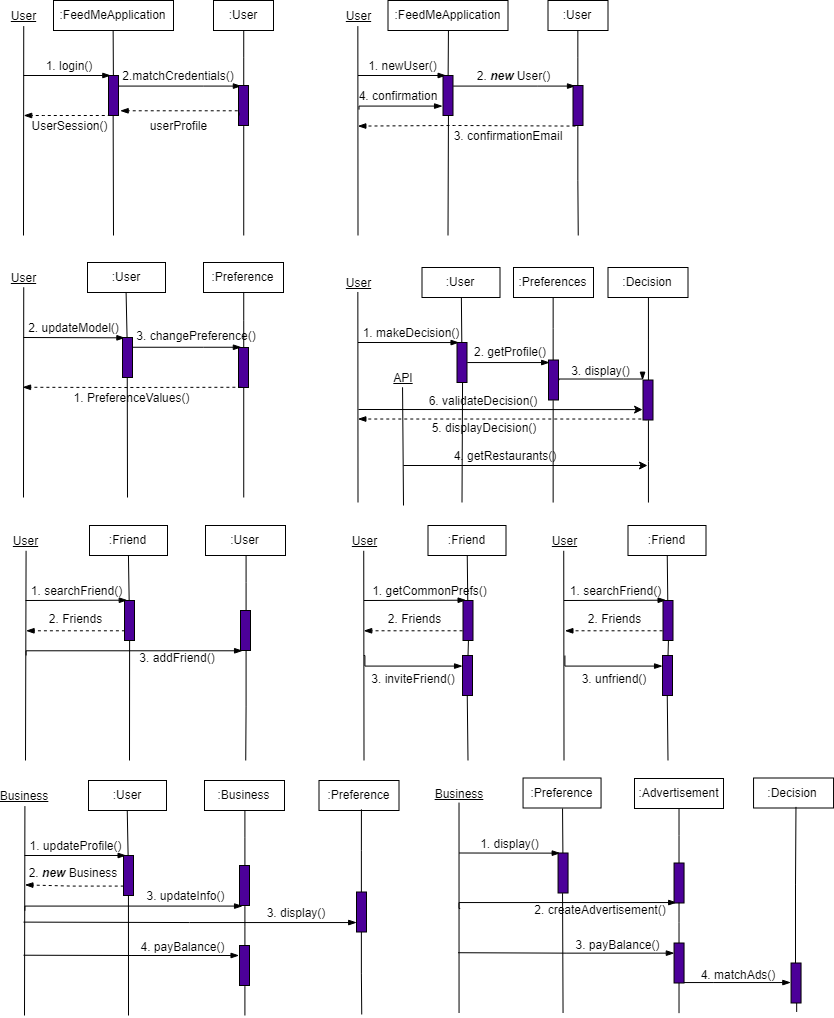


Figure 6: Sequence Diagrams for Use Cases (Left to Right, Top to Bottom): login, registration, take survey, Make decision, Add friend, Invite Friend, Manage Friend Request, View User Summary Statistics, and Create Advertisement

#### 7.10.2 Collaboration Diagrams

Nothing as of Version 1 of this document.

### 7.11 Dictionaries

See the appendix (section 15.1) for the dictionaries defined for the classes of the proposed product.

## 

## 8. NON-FUNCTIONAL / OPERATIONAL SPECIFICATION

### 8.1 USER Interface Requirements

The application will be built with Progressive Application Paradigm (PAP) in mind. As opposed to the standard procedure of PAP development, the Feed Me application will be based off desktop design first and progressed into smartphone layout second. Lastly, due to the PAP, Feed Me won’t have screen resolution constraint.

The interface development will follow BEM CSS style guide. Furthermore, Feed Me will use CSS components as opposed to inline styling. For example, each CSS style sheet must be separated from GUI component and imported at the top of GUI file.

### 8.2 Performance Requirements

The Machine Learning system shall be design for parallel computing for a training phase. The system must process at least 100 recommendation per second in a peak load of HTTP requests. Further, production of 5 recommendations shall not take more than 2 – 3 seconds for 95% of the requests.

### 8.3 Security and Privacy Requirements

* Every input data must be bleached for unwanted adversary code.
* At least 99% intrusions shall be detected within 10 seconds.
* The database for user data shall be fully encrypted via Symmetric Encryption of at least AES-255.
* Storage of sample data for machine learning phases must be containerized from the rest of production data.
* Developers working with sample data for machine learning system shall be logged into remote logger.

### 8.4 System Environment Requirements

The system environment must be encapsulated within special purpose file denominated as .env file. Such file shall not be ever committed into remote and shared codebase repository. This implies that the .env file must reside at all time on developer’s machine.

### 8.5 Computer Resource Requirements

#### 8.5.1 Computer Hardware Requirements

The computers running the components of the proposed system much have access to persistent storage capabilities in order to maintain a database record of the users.

#### 8.5.2 Computer Hardware Resource Requirements

Given the thin client architecture employed by the application, the client device does not have to be rich in powerful resources and capabilities to run the application. Rather the server must have the ability to serve multiple clients at once and run pre-built machine learning algorithms, which are often resource intensive.

#### 8.5.3 Computer Software Requirements

Nothing as of Version 1 of this document.

#### 8.5.4 Computer Communications Requirements

Given the server client architecture utilized by the proposed product, it is required that the computers running the components of the system be connected to the public facing internet such that the clients may communicate remotely with the servers as well as 3rd party APIs.

### 8.6 System Quality Factors

To ascertain the quality of the system, the team shall use number of requests processed per 1 second. To ascertain the quality of the machine learning system, the team shall use accuracy on training sample data. As an additional improvement, the training sample data must be sampled at random every training phase.

### 8.7 Design and Construction Constraints

The only design and construction constraint as of Version 1 of this document is that the project be completable within the given deadline of the project by the project authority.

### 8.8 Personnel-Related Requirements

Nothing as of Version 1 of this document

### 8.9 Training-Related Requirements

* Each developer shall be fully acquainted with privacy protocol when handling user data.
* Each developer shall be fully trained on sampling protocol of data for machine learning training phase.
* Each developer shall understand machine learning metrics used by the system in full

### 8.10 Logistics Related Requirements

Not included as of Version 1 of this document.

### 8.11 Packaging Requirements

The system should be self-contained and distributed on publicly available channels (i.e. the Apple App Store, the Google Play Store, etc.). The size of the application should be kept minimal (below 100 MB) to encourage uses to not uninstall the application.

### 8.12 Precedence and Criticality Requirements

Nothing as of Version 1 of this document

### 8.13 Other Requirements

Nothing as of Version 1 of this document.

## 9 SYSTEM TEST PLAN REQUIREMENTS

The test plan’s purpose is to ensure that the final system conforms to the requirements previously defined so that it meets the needs and expectations of the customers and stakeholders. The most effective way of doing so would be to test the product within a closed group of testers who belong to the target audience / user-base. Therefore, the requirement for the testing of the final system should include the following components:

* The testers must belong to the identified user groups from and exhibit the characteristics described in Section 3.2.2. This is to ensure that the testing closely resembles the wild usage of the product after release.
* The testers must be allowed full use of the platform in its production state and the developers / testers should not interfere with their usage of the platform. This requirement includes the prohibition of developers from being able to explain the platform to the testers as they would not have said interaction available after the system is deployed to actual users.

The requirements will be delivered to SQA who will create a test plan including test scenarios and expected output, then they will execute the tests and report defects.

## 10 QUALITIFICATION PROVISIONS

Four types of testing will be performed: a desk check, a peer review, a walkthrough and an inspection. The first method of ensuring the quality of this artifact, a desk check, essentially involves the writers of the document proofreading their work such that they identify errors they made when drafting the artifact such as typos, obscurities and confused wording. A peer review involves team members reviewing each other’s work to identify possible defects and errors made during the drafting process. A peer review allows for more individuals to review the work and helps to improve the clarity of the artifact as individuals evaluating their own work tend not to see their own ambiguities as they are familiar with their own thought process.

Walkthroughs involve a group of reviewers having the artifact author “walkthrough” their work in person in front of the reviewers (in this case the rest of the project team and the SQA) so that the reviewers can check that the artifact adequately conveys the author’s intent. Finally, inspections involve submitting the artifact to an independent review panel for review. In the case of this document, it is submitted to the SQA and project authority (Professor Fred Strauss) for review.

## 11 REQUIREMENTS TRACEABILITY

The requirements specified in this document are derived from the functional and nonfunctional requirements defined in the System Requirements Specification, which in turn are based upon the expectations and preliminary requirements outlined in the Project Proposal document.

The requirements are each numbered with a unique identifier such that, later developments of the project baseline, can reference specific requirements. These identifiers are maintained in this document as well to allow developers to track the development of the project from earlier iterations. This traceability ensures that when a defect is found, it can be traced back to where it was injected so that identifying what changes need to be made is easier.

## 12 EVOLUTION OF THE SAS

This document is released as a development of the Systems Requirements Specification. It is based upon the previous document deliverables such as the SRS and SPMP. Aside from information included in previous documents, this document contains more information on the high level architecture and stricture of the application, meant to be taken into account in the creation of later artifacts, such as the final design document, as well as the eventual implementation of the product. Before the submission of the document, the authors of the document will review it for thoroughness and correctness of its portrayal of the system as per the process described in the Qualification Provisions section of this document. The initial state of the document will then be submitted for review to the project authority in order to identify defects missed by the project team. The returned criticisms will be incorporated into the document. Upon return from the project authority, the document will be revised by the project team to correct any identified defects and committed to the configuration and document management system as described in the Configuration Management Plan in the SPMP.

## 13 RATIONALE

Not included as of Version 1 of this document.

## 14 NOTES

Not included as of Version 1 of this document.

## 15 APPENDICES

### 15.1 Dictionaries

TABLE 15.1-1

Class Dictionaries

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Description** | **Attributes** | **Methods** |
| FeedMeApplication | The main interface | +users: Users[] | + login()  + newUser()  + main() |
| Friends | Allows friends to be invite to events | - friendId: int  - friendUserId: int  + friendSince: datetime  + numInvited: int | + getCommonPrefs()  + getUserInfo()  + unfriend()  + inviteFriend()  searchFriend() |
| Users | Authentication of the user | - userId: int  + username: string  + email: string  - password: string  - model: Preference[]  - friends: Friend[] | - matchCredentials()  + getProfile()  + updateProfile()  + addFriend()  - updateModel() |
| Preference | The history logs of the user that shows, which choice was chosen | - preferenceId: int  + preferenceType: int  + value: enum  - created: datetime | - changePreference()  preferenceValues()  + display() |
| Restaurants | The restaurant food categories that is used as input. | + restaurantName: str  - apiSource: enum  + restaurantInfo: str  + matchedPreferences:            Preference[] |  |
| Advertisement | The preference of what the user would like to view in restaurants. | - adId: int  + adType: enum  - adContent: int  - target: Preference[]  - followers: Users | +updateInfo()  - payBalance() |
| Business | The choices that are given to the user. | + businessName: string  + businessLocation: str  - paymentInfo: str  - ads: Advertisements  - followers: Users  - owns: Restaurant | +createAdvertisement()  +updateInfo()  - payBalance() |
| Decisions | The Machine Learning interface that makes the decision | + belongsTo: User  - fit: int  - generated: datetime  - matchedPreferences:            Preference[] | + validateDecision()  - getRestaurants()  - matchAds()  - removeDecision() |

TABLE 15.1-2

Method Dictionaries

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Description** | **Class** | **Arguments** |
| login() | Authenticate the user | FeedMeApplication | username, password, appsecret |
| newUser() | Create a new user | FeedMeApplication | username, email, password, appsecret |
| main() | Main application function | FeedMeApplication | **None** |
| getCommonPrefs() | Get common preferences between friend and given user | Friend | **None** |
| getUserInfo() | Get the associated user information (email / username) for the friend | Friend | **None** |
| unfriend() | Accept or reject friend request (false for accept) or unfriend current friend | Friend | Yes or no (bool) |
| inviteFriend() | Invite friend to eat | Friend | **None** |
| searchFriend() | Static function, search for friend by identifier (email, username, location) | Friend | Identifier |
| matchCredentials() | Match credentials given with the given user (authentication) | User | username, password |
| getProfile() | Get information (including preferences) for the given user | User | **None** |
| updateProfile() | Update the user information (not the profile) for the given user | User | username, email, password |
| addFriend() | Add a new friend to the given user | User | newFriend (Friend) |
| updateModel() | Change the user preferences | User | newPreferences (Preference[]) |
| makeDecision() | Generate 5 decisions for the given user | User | **None** |
| changePreference() |  | Preference | newValue (enum) |
| preferenceValue() | Static function, get all potential preferences given the preferences type | Preference | preferenceType |
| display() | Retrieve preference | Preference | **None** |
| createAdvertisement() | Create a new advertisement for the business | Business | Text (string), matched (Preference[]) |
| updateInfo() | Change information for the business | Business | Infotype (enum), Information (string) |
| payBalance() | Pay off the balance that the business owes | Business | Balance (number) |
| validateDecision() | User can reject or accept a decision | Decision | Reject / accept (bool) |
| getRestaurants() | Get restaurants that match a query | Decision | Query (string) |
| matchAds() | Get all ads related to the decision | Decision | **None** |
| removeDecision() | Stop showing given decision within the user model | Decision | **None** |
| updateInfo() | Change advertisement information | Advertisement | newContent (string) |
| payBalance() | Business can pay balance on specific advertisement | Advertisement | business (Business) |

### 15.2 UML Diagrams

All diagrams are included in the body of this document.

### 15.3 Schedule Tracking

TABLE 15.3-1

Individual Artifact Schedule Tracking

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (individual or team) | Estimated | Actual | Difference |
| SRS – Business Definition | Theodore Kim | 5 hours | 6 hours | +1 hour |
| JinZhao Su | 3 hours | 2 hours | -1 hour |
| Petr Holoubeck | 3 hours | 3.5 hours | +0.5 hours |
| Entire Team | 11 hours | 11.5 hours | +0.5 hours |
| SRS - Requirements | Theodore Kim | 5 hours | 4 hours | -1 hour |
|  | JinZhao Su | 2 hours | 2 hours | 0 hours |
|  | Petr Holoubeck | 2 hours | 4 hours | +2 hours |
|  | Entire Team | 9 hours | 10 hours | +1 hour |
| SPMP | Theodore Kim | 8 hours | 12 hours | +4 hours |
|  | JinZhao Su | 5 hours | 6 hours | +1 hour |
|  | Petr Holoubeck | 6 hours | 8 hours | +2 hours |
|  | Entire Team | 19 hours | 26 hours | +7 hour |
|  | Theodore Kim | 6 hours |  |  |
|  | JinZhao Su | 3 hours |  |  |
|  | Petr Holoubeck | 4 hours |  |  |
|  | Entire Team | 13 hours | 26 hours | +7 hour |

TABLE 15.3-2

Cumulative Schedule Tracking

|  |  |  |  |
| --- | --- | --- | --- |
| Who (individual or team) | Estimated | Actual | Difference |
| Theodore Kim | 24 hours | 22 hours | +4 hours |
| JinZhao Su | 13 hours | 13.5 hours | +3.5 hours |
| Petr Holoubeck | 15 hours | 12 hours | +1 hour |
| Entire Team | 52 hours | 47.5 hours | +8.5 hours |

### 15.4 Defect Tracking

TABLE 15.4-1

Individual Artifact Defect Tracking

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (individual or team) | Estimated | Actual | Difference |
| SRS – Business Definition | Theodore Kim | 3 defects | 5 defects | +2 defects |
| JinZhao Su | 4 defects | 3 defects | -1 defects |
| Petr Holoubeck | 3 defects | 2 defects | -1 defects |
|  | Entire Team | 10 defects | 10 defects | 0 defects |
| SRS – Requirements | Theodore Kim | 5 defects | 3 defects | -2 defects |
|  | JinZhao Su | 3 defects | 0 defects | -3 defects |
|  | Petr Holoubeck | 6 defects | 2 defects | -4 defects |
|  | Entire Team | 14 defects | 5 defects | -9 defects |
| SPMP | Theodore Kim | 3 defects | 2 defects | -1 defect |
|  | JinZhao Su | 2 defects | 1 defect | -1 defects |
|  | Petr Holoubeck | 3 defects | 1 defects | -2 defects |
|  | Entire Team | 8 defects | 4 defects | -4 defects |
| SAS | Theodore Kim | 3 defects |  |  |
|  | JinZhao Su | 3 defects |  |  |
|  | Petr Holoubeck | 2 defects |  |  |
|  | Entire Team | 8 defects |  |  |

TABLE 15.4-2

Cumulative Schedule Defect Tracking

|  |  |  |  |
| --- | --- | --- | --- |
| Who (individual or team) | Estimated | Actual | Difference |
| Theodore Kim | 11 defects | 10 defects | -1 defects |
| JinZhao Su | 9 defects | 4 defects | -5 defect |
| Petr Holoubeck | 12 defects | 5 defects | -6 defect |
| Entire Team | 32 defects | 19 defects | -13 defects |

### 15.5 Updated Gantt Schedule

