SOFTWARE REQUIREMENTS SPECIFICATION

VERSION: FINAL

OCTOBER 1, 2019, CHANGES: OCTOBER 2019

AUTHORS

Valentinos Parizza
Marios Pafitis
Demetris Shimitras
Leonidas Achilleos
Stephanos Pantziaros

REVISION CHART

Version	Primary Author(s)	Description of Version	Date Completed
Draft	Valentinos Parizza Marios Pafitis	Initial draft for separating the jobs giving some ideas and recommendations to the rest of the team members as Product Managers.	2/10/2019
Preliminary	Valentinos Parizza Marios Pafitis Leonidas Achilleos Demetris Shimitras Stephanos Pantziaros	Research and analysis of all the parts from each member of the team. Use of examples for some cases and also graphs or images. Analysis of the 90% of the document, and research for hardware and software requirements. User Interface basic design and Database basic structure.	8/10/2019
Final	Valentinos Parizza Marios Pafitis	Final review of the content. Corrections and improvements for certain parts. Final touches and furthermore analysis in some cases.	10/10/2019

Contents

1. INTRODUCTION

1.1 Purpose

This specification document's purpose is to specify the requirements of the new system "Food Recommendation Agent" which is going to be an expansion of Foody, to provide better services to its users. Some of the requirements have been given to us from the Head Leader of the Software Team in Foody and some others have been derived by requirement analysis. This procedure was done, by using some techniques of requirements derivation, like meeting with the Head leader of the Software team at Foody (Panayiotis Pavlides) and by examining related systems like the recommender system of Amazon.

This document is intended for the General Director of Foody Company in Cyprus, Argyris Argyrou, to inform him about the system and its requirements. It is also intended for the software engineers, Database Administrators and Web Developers of Foody to help them understand the requirements of the new part of the system and prepare for the combination of the two systems as they will adopt the system after its completion.

1.2 Scope

The "Food Recommendation Agent" is a software system, which is going to be used by Foody company, to recommend targeted foods to its customers associated with their designated or/and possible preferences. The system is going to be built upon three main operations:

1. Collect data from Foody's database and feed them into the Recommendation System.
2. The Recommender System will calculate the recommendation attributes for each user and store them in our database.
3. The everyday collection of essential data from users like order-history, rankings and other information taken from websites which will let the users to show their preferences by checking what foods they like or dislike and by creating groups of their preferred combinations of foods and store them in a database.
4. The production of recommendation attributes through the Recommender System for the users of Foody, using the information gathered for them and store them in the Database.
5. The use of the recommendation attributes produced and stored in the Database to recommend targeted foods and drinks to users of Foody.

1.3 Definitions, Acronyms, and Abbreviations

- Food Recommendation Agent (FRA): The system that we are implementing which is consisted of the Recommender System, Likes/Dislikes and Journeys features.
- Foody: A company in Cyprus in which you can order online from different restaurants,
- Recommender System: A system that is going to calculate the recommendation attributes of a user.
- Recommendation attributes: Values that define the preferences of a user.
- Likes: Foods that a user like.
- Dislikes: Foods that a user doesn't like.
- Journey: A collection of liked foods for a user.
- My Journeys: A collection of Journeys for a user.
- Food: Any kind of food or beverage. Products that Foody's restaurants sell.
- Database: A place to store the data calculated from our system.
- Foody Online Orders system: The existing system of Foody which is used for orders and with which the new system is trying to connect.

1.4 References

- Material.io for User Interface Design: https://material.io/
- Foody's Website: https://foody.com.cy/gr/2202?filters=1
- Amazon's Website: https://www.amazon.com/
- Constraints Information: http://agilemodeling.com/artifacts/constraint.htm
- User Interface Design: https://en.wikipedia.org/wiki/User_interface_design
- Responsive User Interface: https://en.wikipedia.org/wiki/Responsive_web_design
- GDPR Law: https://gdpr-info.eu/
- Waterfall Model: https://en.wikipedia.org/wiki/Waterfall_model
- Requirements Specification: https://en.wikipedia.org/wiki/Software_requirements_specification
- Software Lifecycle Processes: https://standards.ieee.org/standard/12207-2017.html
- System Interfaces: https://deseng.ryerson.ca/dokuwiki/design:system_interface?fbclid=IwAR3M2mEcuOhjhTSC2gJjJcVaPzkBB5RgHTwCoSosNiRAbJ3VEiuoUCrhhhg
- Software Requirements: https://belitsoft.com/blog/software-requirements-specification-document-example-international-standard?fbclid=IwAR2nFX0f5kFs6hB7EvZBi3DEb83fhqXTMv8kW78q8JxqADt7gZtINejeq8Y
- Software Engineering Ian Sommerville: https://www.academia.edu/31140292/Ian_Sommerville_Software_Engineering_9th_Edition_2011_1_

2. OVERALL DESCRIPTION

2.1 Product Perspective

The software product is intended to become an element/subsystem of an existing system and more specifically shall merge with the preexisted software system of Foody for orders, which will provide recommendations of items to the users that should coincide with their preferences on Foody's content. This system depends merely on the existing system of Foody for orders, as it uses data from its existing Database and also it provides the Foody system with the appropriate user's attributes, used for recommending foods in Foody's webpages. This system runs on a single server of Foody as it can satisfy the needs of the population of Cyprus using one server.

2.1.1 System Interfaces

The "Food Recommendation Agent" System is going to be a subsystem of an existing system of Foody which interacts with the users using its website for helping in creating orders of food and/or drinks as well as with the different restaurants from which clients/users want to order. So, it combines two types of clients, by providing different types of services to both of them as well as create a unique connection

between them, and behave as a communicator of orders and offers (orders of users to restaurants, and offers from restaurants to users).

Now with the existence of "Food Recommendation Agent" subsystem, the previous part of the system, the Foody Order Online system is going to have a new input coming from the "Food Recommendation Agent" system and a new output going to the "Food Recommendation Agent" system. So, the input to the "Food Recommendation Agent" system will be some data from the existing Database of Foody Online Orders system, which are going to represent information associated with each user (That is each user's Foody id, user's N previous orders' history (More specifically 0<=N<=100), user's ratings).

The input to the "Food Recommendation Agent" is output of the existing Foody online orders system. Following this, the "Food Recommendation Agent" system gives its output to the Foody online orders system, which for it is supposed to be input. This output includes the user's attributes which are then used by Foody Online orders system to provide recommendations to its users. Another input to the **"**Food Recommendation Agent" system coming from the Foody Online Orders system (output for this system) is the account details of each user, which are used for authentication of a user.

2.1.2 User Interfaces

2.1.2.1 General Page User Interface

Some of the inputs to the "Food Recommendation Agent" system is coming from some web pages associated with food likes/dislikes actions and creation of groups of combinations of preferred foods. These webpages are going to be included in the site of "Food Recommendation Agent" system as they will be used only for producing the necessary recommendation attributes of the users for the better recommendation of foods and drinks.

So, all the users of Foody, are going to have an initial similar log-in page, from where they can sign-in to their accounts using the sign-in web page of Foody online orders system. After that, there will be a redirection link where user can use it to proceed to the formal web page of the "Food Recommendation Agent". This web page is going to be used by users to like/dislike different foods and drinks.

The "Food Recommendation Agent" that we are going to implement it has to be responsive with attractive transitions, especially for the pop-up dialogs and for the expansion mechanism which is going to be explained in the section 2.1.2.2.3. The User Interface is going to use Material.io components by Google as the rest of Foody uses the same design. Moreover, the website will be fully responsive for Desktops, Laptops, Tables and Smartphones.

At the web page "Food Recommendation Agent" you will have five different options in the top bar. You can select between the tabs "Foody", "Likes", "Dislikes", "My Journeys" and "Explore". Initially the user will be in the "My Journeys" tab. The user can return in Foody's web page by clicking the "Foody" button on the left of the top bar.

2.1.2.2 Specific Page User Interface

2.1.2.2.1 PAGE USER INTERFACE: MY JOURNEYS

The user can select the option "My Journeys" from the top bar which is going to display all the Journeys that has already created, if any, otherwise a big button will be displayed saying "New Journey". By clicking the "New Journey" button the user will be redirected to the "Explore" tab to pass through the procedure of creating a new Journey.

In the case that the user has already some Journeys, they are going to be displayed as big boxes containing a number of pictures representing the foods that the user has selected from a previous exploration. Also, the name of the Journey is going to be displayed as well with three buttons, "View", "Edit", "Delete" the Journey.

By pressing the “View” button the user will be able to see the full content of the Journey. If the user presses the “Edit” button, he/she will be redirected to the “Explore” page and will be able to add or remove food from the Journey. If the user presses the “Delete” button a pop-up dialog will show up asking the user “Are you sure you would like to delete this Journey?” and if the user presses the button “Yes”, the Journey is going to be deleted.

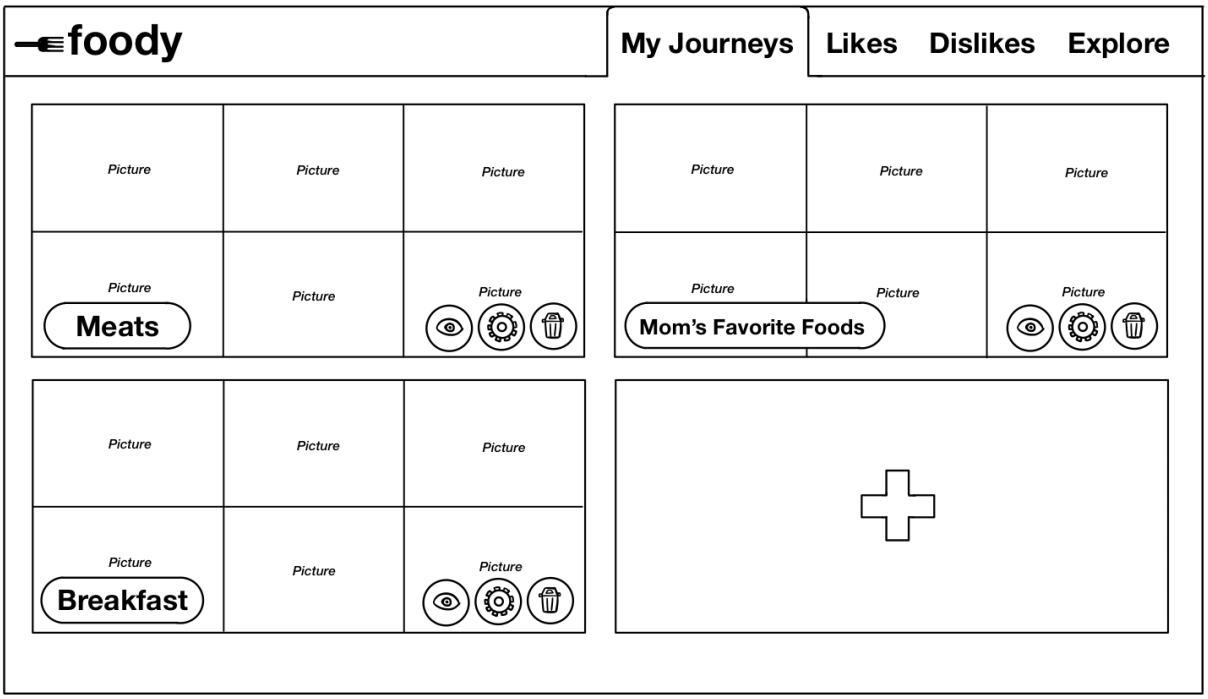


Figure 1: My Journeys Tab

2.1.2.2.2 PAGE USER INTERFACE: LIKES/DISLIKES

The user can select one of the two options, the “Likes” and “Dislikes” from the top bar. The user is going to see the foods that has already been liked or disliked respectively, if any, represented as pictures in a grid with the title of each food.

The user will have the choice to add or remove foods from that group. When the user hovers above a picture, a trash can will be displayed to remove the specific element. If the user presses the trash can, a verification dialog will pop-up asking “Are you sure you would like to delete this food?” and if the user presses the button “Yes”, the food is going to be deleted.

At the end of the list, a button is located in the shape of the pictures, having the plus icon inside indicating to the user that by pressing this button the user can add more foods in each group. In that case the user will be redirected in the “Explore” tab, to find more food for either the “Likes” or “Dislikes” group.

The user can select the “Edit” icon in the bottom right corner of the screen, and a menu with functionalities will appear such as “Multiple Deletes”, ”Rearrange Foods”, “Sort A-Z”, etc. will appear giving some extra functionality to the user, to manage the content of the web pages.

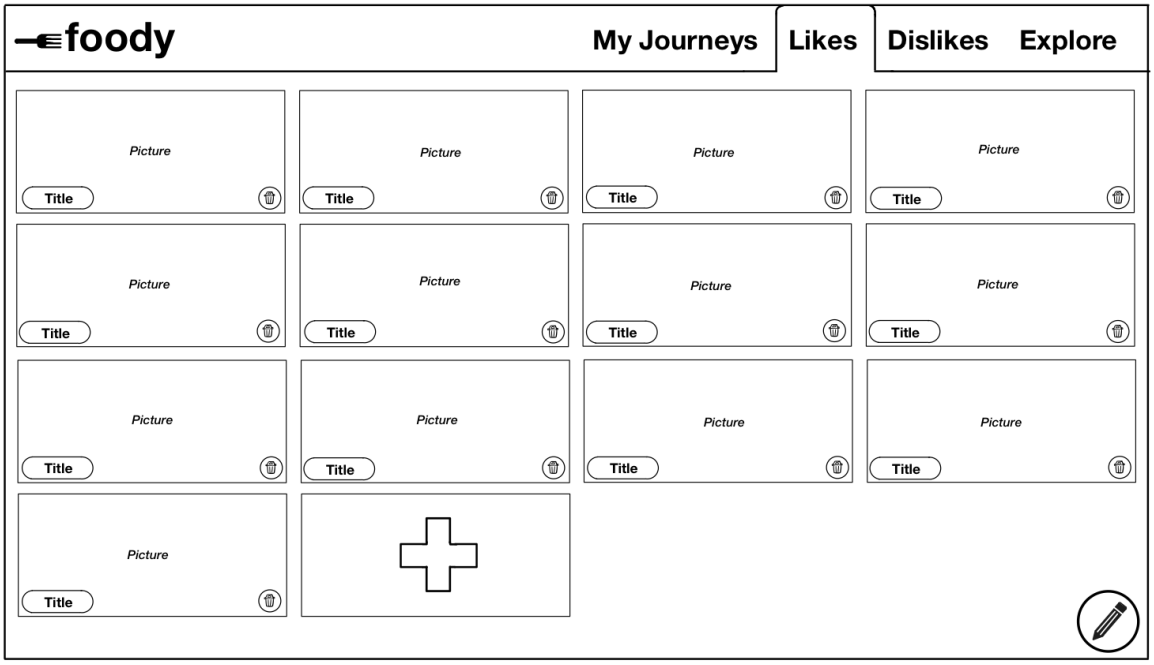


Figure 2: Likes Tab

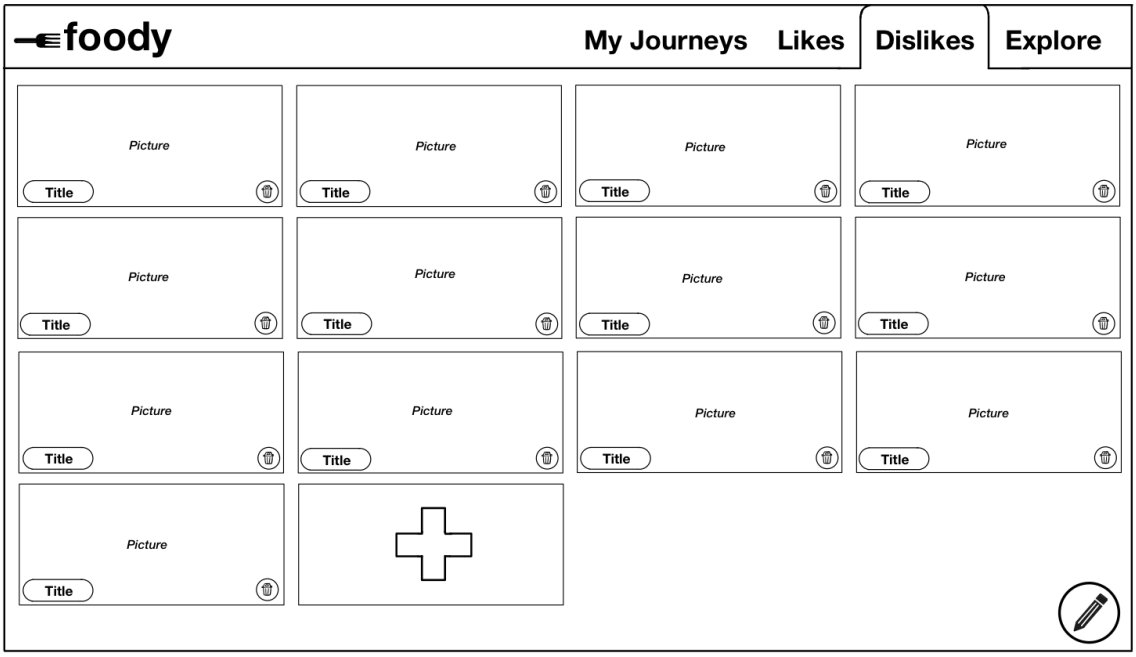


Figure 3: Dislikes Tab

2.1.2.2.3 PAGE USER INTERFACE: EXPLORE

The user can select the option “Explore” from the top bar. The explorations help the user create new Journeys or Like and Dislike foods. The expansion mechanism will be able to help the user indicate to us the specific foods that desires. A grid with a small number of pictures representing the categories will be displayed. with a title describing the content of the picture.

When a user hovers over a picture will have three choices, “Like”, “Dislike” or “Add to Journey”. If the user selects the “Like” or “Dislike” button the food is going to be added in the “Likes” or “Dislikes”

group respectively. In the case that the user selects the “Add to Journey” button, the food is going to be added in the Journey. The user would be able to see the content of the current Journey on the left part of the screen, and set the name of the Journey as well.

When a user selects the “Add to Journey” button the expansion mechanism is going to update the content of the grid. For example, some titles could be “Drinks”, “Salads”, “Meat”, “Traditional” etc. with the appropriate picture. The user can select any of the categories and the grid is going to expand by adding types of foods in the specific category.

If the user selects the category “Meat”, the types of foo added can be “Barbeque”, “Oven”, “Chinese”, etc. These pictures of new types of food will be placed after the “Meat” picture, shifting the rest of the pictures and the picture “Meat” will be highlighted as selected. Immediately the category “Meat” is going to be added in the Journey.

If the user presses on a picture of a type of food such as the “Chinese” for example we are going to expand the grid by adding specific consumable products, such as “Chicken Teriyaki”, “Sweet & Sour”, etc.

In the case that the user would like to undo an action, he/she can press the ‘x’ button on the top corner of each picture to reverse an expansion. Finally, the user can “Save” the Journey and give it any preferable name according to the content of the Journey such as “Lunch Journey”, “Moms Favorite Foods”, “Eat Healthy” etc. The Journey is going to be saved in our Database in order for the user to view, edit or delete it in the future.

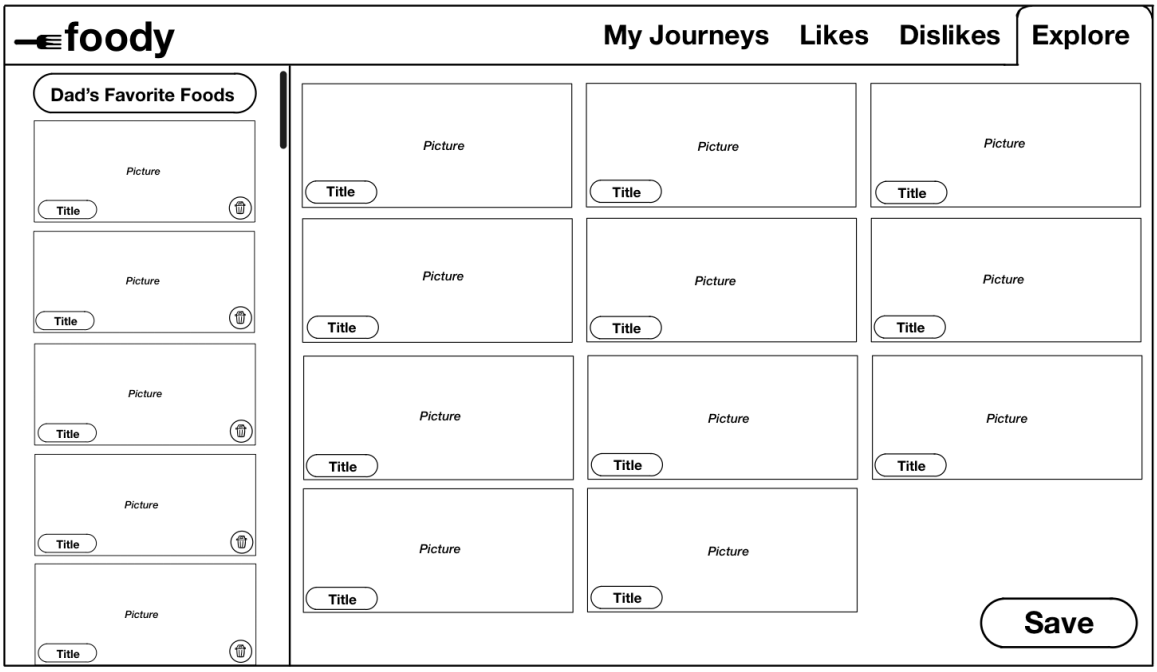


Figure 4: Explore Tab

2.1.3 Hardware Interfaces

The server on which the software system will run is the following:

Server name	SYS-6049GP-TRT - Super Server 6049GP-TR
Dimensions	• Height 7.0" (178mm) • Width 17.2" (437mm) • Depth 32.1" (815mm) • Weight Net Weight: 65.5 lbs. (29.7 kg) • Gross Weight: 100 lbs. (45.3 kg)
Color	Black
Motherboard	Super X11DPG-OT-CPU
CPU	• Dual Socket P (LGA 3647) • 2nd Gen. Intel® Xeon® Scalable Processors (Cascade Lake/Skylake) ‡, 3 UPI up to 10.4GT/s • Support CPU TDP 70-205W
Cores	28 Cores
Memory Capacity	• 24 DIMM slots • Up to 6TB 3DS ECC DDR4-2933MHz† RDIMM/LRDIMM • Supports Intel® Optane™ DCPMM†
Memory Type	2933†/2666/2400/2133MHz ECC DDR4 RDIMM/LRDIMM
INPUT/OUTPUT	• SATA 10 SATA3 (6Gbps) ports • LAN 2 RJ45 10GBase-T LAN ports 1 RJ45 Dedicated IPMI LAN port • USB 4 USB 3.0 ports (rear) • Video 1 VGA Connector • COM Port 1 COM port (header)
Power Supply	2000W Redundant Power Supplies with PMBus
Fans	8x 92mm RPM Hot-Swappable Cooling Fans

For more information about the server see

https://www.supermicro.com/en/products/system/4U/6049/SYS-6049GP-TRT.cfm

2.1.4 Software Interfaces

Database (DB) has	Name: MySQL (for Ubuntu Linux) Version: 8.0.17 Source: https://dev.mysql.com/downloads/mysql/
Server Operating System:	Name: Ubuntu Version: Ubuntu Server 18.04 LTS Source: https://ubuntu.com/#download
Server extra software	● SFT-OOB-LIC: OOB Management Package ● SFT-DCMS-Single: Data Center Management Package
Web Browser of the end devices/ hosts of Customers/Users	It will be supported for the following web Browsers: ● Google Chrome ● Microsoft Edge ● Mozilla ● Safari ● Internet explorer ● Opera ● Any other web browser which is non-restrictive—> Offers the same basic functionalities as the browsers above.
Web development Platform	LAMP stack→ Combining Linux, Apache HTTP Server, MariaDB / MYSQL, PHP / Python / Perl

Actually, exactly the same software interfaces apply for the existing system in Foody: the Foody Online Orders System. So, the fact that it was adopted for the Food Recommendation Agent system, it isn't coincidence and actually is a software requirement defined for compatibility, accessibility and maintainability reasons occurring at the combination of the entire final system.

2.1.5 Communications Interfaces

For this system there are going to be used two types of networks for communication. One communication is going to exist with the internet (Wide Area Network - WWW) and one with the Foody Online Orders system locally (Local Area Network - LAN → which is going to be private). The communication with the Internet is going to exist for providing the services of the server to the clients/users of the Foody and for receiving the different requests from the users/clients (orders). The private communication with the Foody Online Orders System is used for the necessary exchange of data between that and "Food Recommendation Agent" system.

2.1.6 Memory Constraints

The Server should have enough space to store the records for approximately 200,000 users, as the maximum population of Cyprus that can have a Foody Account. For each user we have to store data about its Journeys, Likes/Dislikes and recommendation attributes which had been generated from the Recommender System. Approximately, 250GB of HDD storage should be enough to keep all the data for our clients.

In terms of RAM, the machine learning algorithm is not so demanding for a logical number of arguments and for only 200,000 users. So, a normal modern server should be capable of calculating the recommendation attributes within an hour. Because we are not going to use the Recommender System live mode, we will generate the recommendation attributes during night hours (11:00 pm - 11:00 am), when no orders are performed as the restaurants are closed. Though, no extreme hardware is needed for our system.

2.1.7 Site Adaptation Requirements

Since Foody is an already developed system, the new recommendation system, that will be an extension of Foody, will need to adapt to it. Thus, in the beginning of its functionality a particular amount of time might be necessary so that it can process the preexisted data of the clients/users in Foody's system and eventually lead to the results expected.

As mentioned before in *section 2.1.1*, as an input to the "Foody Recommendation Agent" will be the users' previous data, such as their orders and ratings. Therefore, these data which have been collected since Foody has been active, must be processed altogether for the recommendation system to conclude to results for every user. After the startup and adaption of the "Food Recommendation Agent" system to Foody's preexisted system, each night this process is planned to be made overnight, when traffic is minimal to none, so that new data collected from the interaction of the users with Foody will update the information already obtained for each one.

2.2 User Characteristics

For the system "Foody Recommendation Agent" there are some users that are going to be using and interact with the system in different ways. These users are the ones that are going to receive the system after its completion. Therefore, an important aspect is to define exactly the audience who will use the system in order to better understand how the system as a whole will eventually be of significant help to them and consequently leading to more qualitative and accurate system requirements. The users and their characteristics are the following:

Software Engineers of Foody:

- Experienced Software Engineers with strong industrial background
- Abilities to understand quick quantitative and qualitative results of a system and figure out ways to maintain and expand the system
- Capable To observe the new system and its interactions with the existing, and try to see extensions of the system.

Front-End developers of Foody:

- Experienced front-end designers and developers
- Have nice and elegant way of expressing aesthetics
- Capable for the web design of the front-end part of the webpages

Database Administrator of Foody

- Advanced knowledge on Databases and the specific database which the server of the new system will be running
- Abilities to understand problems occurring in the databases
- Knowledge of how to help databases, increase their efficiency.
- Capable of managing a database, and figuring out solutions to many problems that might occur.

Administrator of Foody's servers

- Administrator of the Operating System running on the server
- Advanced Knowledge of Operating Systems and Systems' Programming
- Experienced background from the industry
- Capable of managing the Servers of Foody and of the new system

Foody's Customers/Users

- People living in Cyprus
- Of any age in which they are capable of creating orders
- No prerequisite technical knowledge in order to use the website
- Capable of using the web and carry out a transaction or order

2.3 Constraints

There are some hardware limitations for the Food Recommendation Agent system. These are:

- One server must be adequate for running the Software of the FRA system. More specifically the work that the software part of the Food Recommendation Agent system will be doing must be satisfied by only one server.
- Server must have the necessary capacity of memory, CPU capabilities, number of cores, GPUs, RAM capacity and other computer components to store and manage data for 1.2 times the population of Cyprus users.
- Users using the Food Recommendation Agent system from via websites should be using the appropriate hardware which supports web non-restrictive Browsers.

The actual existing system of Foody, Foody Online Orders system placed the following constraints to the abstract format of the messages that are going to be exchanged between the two systems. These are:

- The FRA system will be receiving the basic data - messages associated to each user requested based on his/her Foody ID and with the format (user's Foody ID, user's ratings, User's N last orders [History orders and N is an Integer number → 0=<N<=100])
- Also, the FRA system will be sending the basic data-messages (to the Foody Online Orders system) with the format (user's Foody ID, User's recommendation attributes).

The FRA system has the following parallel operation constraints:

- The system must be able to service two or more users using the FRA's website at the same time.
- Operations/Transactions on the database must be able to be made in parallel, keeping the state of the database consistent after any combination of parallel operations on the database.

The FRA System, due to audit functions constraints is going to follow GDPR rules, for keeping the data of the user private and only accessible by Foody. Will follow Foody's privacy policies, the one that each user agrees when he/she creates an account in Foody. The System is going to be used only for providing more accurate food recommendations to the customers of Foody.

For higher-order language requirements the FRA system has the following constraints:

- The back-end code of the Food Recommendation Agent system must be done using the most convenient and rich programming language, in terms of tools for Artificial Intelligence and Machine Learning. For that purpose, the Python Programming language has been decided to be the base on which the back-end of the system will rely on.
- The Front-End part of the web pages must be structured and built upon HTML 5.0, CSS and JavaScript.
- For the design frame and structure of the webpages Foody Company demands the use of Material.io from Google, in order to follow the same principle as the Foody Online Orders System.

For the private LAN network which will be used by Food Recommendation Agent system and Foody Online Orders system to exchange messages, a handshake network protocol (three-way-handshake like) will be applied. Therefore, each subsystem will be establishing a connection to either send data or receive data through an encrypted and reliable way of communication.

The FRA system, has also constraints associated with reliability. These are:

- The system must be able to recover from unexpected errors without affecting significantly the users of the system. This means that the system, and more specifically the software running on the server must be able to recover from most of the errors immediately without having the customers and users waiting for more than an hour.
- The system must be able to guarantee that any transaction made by simultaneous users, doesn't have any undesired and bad side effects, like charging two times a user for one order.
- The system must be able to guarantee that any transaction made and completed by any user's device, actually was applied and registered by the software system and the server, no matter the situation.

There is only one constraint relating to the criticality of the application of the FRA system and is about the critical commitment that the system ensures that the personal data stored in the FRA system's database are only for recommendation use and are not going to be collected by any non-authorized by the users person or group of people.

This places the constraint that the information collected by the FRA system must be strictly protected and secured by any non-authorized activity and also the communication between the two subsystems must be encrypted in a way to ensure protection from unauthorized activities. So, this leads to the criticality of the application which is that the system stores sensitive personal data that are not for the use of any other purposes than those the user accepts. Despite this the Foody Online Orders system doesn't rely on the Foody Recommendation System in order to operate normally and satisfy the basic needs of its customers, but it's critical that the information stored in FRA system, remain secured.

From the previous constraint about criticality of application, another constraint type: safety and security considerations has appeared. These constraints are coming just to ensure that the information used by the FRA system will remain as much as possible secured from unauthorized activities on them. These constraints and considerations are the following:

- The information of each user stored in the FRA's system database must be distinguished from the other users' information by a single number, the Foody ID for each user. Therefore, if some of the information in the FRA's server was stolen, the data stolen wouldn't be adequate to distinguish who actual users are related to the specific data, because it would need the mapping of the Foody ID's of users to the actual Users' information in the Foody Online Order system's database.
- Also, between the Foody Online Orders system and Food Recommendation Agent system there must be some type of encrypted communication so that ideally no one would be able to decode the messages and use them.

2.4 Assumptions and Dependencies

The FRA would take the users' orders and ratings in order to present to them other items on Foody's system. Therefore, at the moment no factors considered that would need to change the requirements on this matter.

The Like/Dislike function is a simple service for the user to tell their preferences to the website, in order for the website to help the user with their decisions in future use of Foody.

The “Journey” function is merely a specification that would help each user, if they would like to, organize their Journeys however they’d like, so they would have their preferences categorized in different Journeys or even have them scattered in just one. However, their Journeys would again contribute to the FRA to specify their likings and make suggestions in future times.

3. SPECIFIC REQUIREMENTS

3.1 Software Product Features

3.1.1 Feature 1: Like/Dislike

3.1.1.1 Purpose

The problem that we are trying to solve through the Like/Dislike feature is to directly ask a user what kind of food he/she likes and dislikes. By using those data, Foody could make more appropriate recommendations to a user based on his/her real preference. Also, the likes and dislikes will be an essential input for the Recommender System which we are analyzing at section 3.1.3. These direct declarations of food preferences are going to help the Recommender System find indirectly food preferences for a user.

Clarification: The foods in the sections below referred to any kind of product that Foody’s restaurants may sell, for example instead of foods could be also beverages such as coffee, beer, wine, juices, smoothies etc.

3.1.1.2 Associated Functional Requirements

3.1.1.2.1 FUNCTIONAL REQUIREMENT 1: LIKE A PRODUCT

3.1.1.2.1.1 Introduction

The purpose of the **Like** feature is to let the user directly categorize foods based on his/her preferences. Through the exploration process, a user will have the opportunity to see foods and decide whether he/she has a desire for them. This functionality is going to help Foody understand what each different user prefers. Then, Foody will know what kind of food should recommend for each client based on their own direct preference.

3.1.1.2.1.2 Inputs

The user is going to press the Like button if he/she likes the food that is being represented in the picture.

3.1.1.2.1.3 Processing

All the pictures of food are going to be represented in the form of a grid. At the beginning the pictures will show more general food categories such as meat, vegetables, salad, fish, etc. As soon as the user selects and Likes a food, the grid is going to expand itself around the specific food. It will show dynamically relevant types of foods as the one that have been selected.

For example, let's say a user has selected and liked the category salad, then 5 different kinds of salads (Chef salad, Caesar salad, Lettuce etc.) are going to be placed around the picture of the salad. Through this process, the user can select even more specific details about the kind of salad he/she prefers.

Also, at the moment that a user Dislikes an object, he/she will have the ability for a few seconds to undo the action if he/she changed his/her mind.

3.1.1.2.1.4 Outputs

User’s Likes and are going to be stored in our database. The user can view his/her Likes and edit them in a different tab of our website. Also, Foody can access those data to use them for recommending restaurants based on the foods that the user had chosen.

3.1.1.2.2 FUNCTIONAL REQUIREMENT 1: DISLIKE A PRODUCT

3.1.1.2.2.1 Introduction

The purpose of the **Dislike** feature is to let the user directly categorize foods that he/she doesn't like. Through the exploration process, a user will have the opportunity to see foods and decide whether he/she doesn't feel a desire for them. This functionality is going to help Foody understand what each different user doesn't prefer in order to not recommend those food to the client.

3.1.1.2.2.2 Inputs

The user is going to press the Dislike button if he/she doesn't like the food that is being represented in the picture.

3.1.1.2.2.3 Processing

All the pictures of food are going to be represented in the form of a grid. At the beginning the pictures will show more general food categories such as meat, vegetables, salad, fish, etc. As soon as the user selects and Dislikes a food, the grid is going to reorganize the content by removing the specific food. In the place of the specific food is going to place a food not related with the one being Disliked. .

For example, let's say a user has selected and disliked the category salad, then the salad picture is going to be replaced by a picture of a fish, meat, etc. Through this process, the user can continue exploring to find foods that admires.

Also, at the moment that a user Dislikes an object, he/she will have the ability for a few seconds to undo the action if he/she changed his/her mind.

3.1.1.2.2.4 Outputs

User's Dislikes are going to be stored in our database. The user can view his/her Dislikes and edit them in a different tab of our website. Also, Foody can access those data to use them for not recommending specific restaurants based on the foods that the user had chosen to dislike.

3.1.2 Feature 2: Journeys

3.1.2.1 Purpose

The problem that we are trying to solve through the Journeys feature is to let the user create groups of specifying kind of products. In general, to let the user categorize things based on his/her preferences and needs. We can imagine a Journey as a collection of foods for a specific purpose.

For example, a user can create a Journey to separate the foods that he likes to eat for breakfast, for lunch and for dinner. So, when he is going to order for lunch, he will have the option to select the "Lunch Journey" that he has already created based on the foods that he likes for lunch. The Foody is going to recommend him/her restaurants based on his/her preferences in the "Lunch Journey".

Another example is to create Journeys based on the categories of foods that the user likes. For example, you can have a journey "Fast Food Journey", or "Eat Healthy Journey" and based on his/her mood can select a Journey and let the Foody recommend him the appropriate restaurants based on the Journey.

Moreover, a common situation of Foody's user is that all the family has only one Foody account. You can separate the preferences of your mother, father, sister, etc. into different Journeys. So, when a family member except from the account owner would like to order something can select his/her Journey, to let the user recommend him/her restaurants.

3.1.2.2 Associated Functional Requirements

3.1.2.2.1 FUNCTIONAL REQUIREMENT 1: CREATE A JOURNEY

3.1.2.2.1.1 Introduction

Each user will have the ability to create a certain number of Journeys based on the way he/she would like to use them. Each Journey will help the user to select what he/she likes and also explore other different foods that he/she might be interested in.

3.1.2.2.1.2 Inputs

The user will create a new Journey and name it the way he/she likes. The user will be able to choose foods that are being displayed in a grid of pictures and add them to the Journey. It is the same procedure as in Likes/Dislikes feature section 3.1.1, but now instead of Like or Dislike button, the user is going to use the Add to Journey button.

3.1.2.2.1.3 Processing

When a user adds a food to the Journey, the expansion mechanism is going to add new foods in the grid based on the selected food. The new foods will have the same or similar category as the food that had been added to the Journey. The purpose of this is to help the user specify the meals that he/she prefers most.

3.1.2.2.1.4 Outputs

At the end of this procedure the user will have the new created Journey which can use for helping him in orders. The Journey is going to be stored in our Database and the user will have access to view, edit or delete the Journey from the My Journeys tab.

3.1.2.2.2 FUNCTIONAL REQUIREMENT 2: EDIT A JOURNEY

3.1.2.2.2.1 Introduction

The user will be able to edit a Journey in the future. The user will have the ability to add new foods in the Journey or just delete certain foods that he/she had already added.

3.1.2.2.2.2 Inputs

The exploration system will be generated again and the user can add more foods in the Journey. The foods that are already in the Journey will be displayed in the grid among the other pictures of foods and the user will have the ability to remove them.

3.1.2.2.2.3 Processing

The already selected foods will be displayed at the very end of the grid, and the user can Remove them from the Journey. If he/she selects and adds a food in the Journey that doesn't exist already in the Journey, the expansion mechanism is going to be generate new related foods. This procedure will use the same expansion mechanism as described in section 3.1.1.

3.1.2.2.2.4 Outputs

The Journey will keep all the changes if the user presses the Save button, otherwise if he/she press the Cancel button all the changes will be discarded. The new updated Journey will be stored in the Database and also in the tab My Journeys to view, edit or delete it in the future.

3.1.2.2.3 FUNCTIONAL REQUIREMENT 3: DELETE A JOURNEY

3.1.2.2.3.1 Introduction

The user will have the ability to delete a Journey that he/she might not like anymore, Also, a limit will be set as the number of Journeys a user can create. The need of deletion might show up in the case of a user having the maximum number of Journeys.

3.1.2.2.3.2 Inputs

The user will select a Journey from My Journeys tab that he/she might like to delete. The user can choose to view the Journey before delete it or just delete it immediately.

3.1.2.2.3.3 Processing

If the user selected to view the Journey, he/she will have the ability to see the content of the Journey. Then, if the user still desires to delete the Journey, he/she can press the Delete button, and Accept the deletion in the verification dialog. The user will have the ability to delete immediately the Journey without viewing it from the My Journeys tab.

3.1.2.2.3.4 Outputs

After deleting a Journey, the Journey is not going to be presented in the My Journeys tab anymore and it will be deleted from the Database too.

3.1.3 Feature 3: Recommender System

3.1.3.1 Purpose

The problem that we are trying to solve through the Recommender System is to make indirect recommendations to the user based on what the system thinks he/she might like. Through a learning process with different kinds of arguments, the Recommender System will be able to predict with precision what a user likes, or he/she would be interested to try.

3.1.3.2 Associated Functional Requirements

3.1.3.2.1 FUNCTIONAL REQUIREMENT 1: USE RATINGS

3.1.3.2.1.1 Introduction

This method of defining the recommendation attributes will use the ratings for the restaurants for each user which we are going to collect from Foody's Database. This method even if it is very powerful it is not enough because a user more frequently may rate a restaurant based on a group order and not individually for his/her meal. This could end up bot making very precise predictions for all the cases. So, the other two methods that the Recommender System is going to use the History of Orders and Like/Dislikes will help the system predict exactly the preferences of a user.

3.1.3.2.1.2 Inputs

From Foody's Database will get the ratings for the restaurants of each user. In the case of no ratings exist for a user, random generate values are going to be generated and through the machine learning procedure the system can find some basic recommendation attributes.

3.1.3.2.1.3 Processing

The machine learning algorithm will use the ratings to generate the recommendation attributes for the user. If the recommendation attributes have already been generated in the past, the system will use them among the new data of the user to generate new updated recommendation attributes.

3.1.3.2.1.4 Outputs

The recommendation attributes based on the ratings will be stored in our Database which the Foody will have access if they desire to perform recommendations to the users.

3.1.3.2.2 FUNCTIONAL REQUIREMENT 2: USE HISTORY OF ORDERS

3.1.3.2.2.1 Introduction

The user's order history is a key asset for calculating the recommendation attributes for a user. Though it is not correct to take only the orders as inputs for the Recommender System, because an order for a

specific restaurant might have been performed by a group of people and not the user himself/herself. Also, the user might didn't like the specific restaurant, so the ratings will give us this feedback

3.1.3.2.2.2 Inputs

We will collect a number of orders from the history of each user form Foody's Database and use those data to feed the Recommender System. If a user hasn't made any orders yet we will use just the random values as ratings in order to find some general recommendations for the user.

3.1.3.2.2.3 Processing

The machine learning algorithm is going to use the most recent orders of the user to generate the recommendation attributes. If the orders of the user haven't changed since the last time that the recommendation attributes had been generated, the system will not perform the algorithm for the specific user.

3.1.3.2.2.4 Outputs

The recommendation attributes based on user's order history will be stored in our Database which the Foody will have access if they desire to perform recommendations to the user.

3.1.3.2.3 FUNCTIONAL REQUIREMENT 3: USE LIKES/DISLIKES

3.1.3.2.3.1 Introduction

Through the feature of our system Likes/Dislikes (section 3.1.1) the user can directly define his/her preferences for certain foods. The Recommender System will use those data to create indirect preferences for the user based on the machine learning algorithm of the Recommender System. In general, is going to guess what the user might also like based on the foods that also prefers.

3.1.3.2.3.2 Inputs

The Likes/Dislikes from our Database for each user. In the case that no Likes/Dislikes have been set for a user, random values will be generated to create more general recommendation attributes.

3.1.3.2.3.3 Processing

The machine learning algorithm will use the Likes and Dislikes from the user to eliminate food choices based on the relevant foods that user Dislikes. Also, is going to use the Likes to find similar foods that the user would like to eat. Altogether, the foods that user may like and the foods that doesn't like create the recommendation attributes, the indirect preferences, for each user,

3.1.3.2.3.4 Outputs

The recommendation attributes based on the Likes/Dislikes will be stored in our Database and Foody will have access to them, in order to perform some recommendations for each user based on his/her indirect preferences.

3.2 Performance Requirements

3.2.1 Static numerical Requirements

Title: Time for Update of all Users' recommender attributes

Description: The time needed for the recommender system algorithm, to update all the attributes of the users, and replace them with new attributes derived from new information taken for each user (new orders, new ratings, new likes/dislikes, new groups of foods created by him/her).

Must: The time for updating all Users' recommender attributes to be one or less than a day.

Goal: The time for updating all Users' recommender attributes to be less than 12 hours.

Justification: Because Cyprus is a small island, having no more than 2 million residents, it is clear that the system can update the attributes of the users of Foody in less than a day. More specifically, our goal is to have the system updating the attributes during the hours of 11:00 p.m. to 11:00 a.m., so that the traffic in our system would be small rather than having it to run in other parts of the day when many people order and ask for services. From a test that we have made we have estimated that for approximately 200.000 people, it takes less than an hour to calculate their attributes.

Title: Maximum capacity allocated for each user's information in the database

Description: The maximum amount of memory space that can be allocated in total for user's information in the Database. The information includes the user's recommender attributes, the likes/dislikes of the user's and the groups of foods.

Must: Maximum capacity allocated for each user's information in the database must be less than 300KB.

Goal: To reduce redundancies of data for each user and store more compact forms of data for users in the database so that accomplishing less than or equal to 200KB Maximum capacity allocated for each user's information.

Justification: The information needed for each user is not going to be huge. The number of food groups and the number of foods in each group are going to be limited to a finite and relatively small number. Also, the attributes of each user are going to need constant amount of memory space, independently of the exact preferences and history of users.

3.3 Software System Attributes

3.3.1 Reliability

The users of our system have to be logged in at Foody before having access to our content. Each Journey, Likes/Dislikes of a user can only be accessed by the user himself/herself. So, the login part will be controlled by Foody.

The Database is going to be stored at Foody's Servers among the other System that Foody controls. The only person who has access to the Database is Foody. Even more, if Foody doesn't have the API defining the encoding and decoding for the data can't identify their users in our Database, because the user ID is encoded in our database.

The Recommender System which is the most demanding in Database usage, of all the features in our System, is going to perform without interrupting other systems of Foody during night hours (11:00 pm - 11:00 am).

3.3.2 Availability

The system should be available any time of the day. The users should have the opportunity to create/edit/delete at any time their Journeys. Also, the Database should be accessible from Foody 24/7 as it is always up and running for constantly keeping the most updated data for the users.

3.3.3 Security

3.3.3.1 Cryptography

Our system is going to store user's sensitive content in terms of their preferences of food. We will not keep in our database any information that can identify who the user is in real life. We are going to convert the Foody's ID for each user through an encoding method to a new ID and this is going to be the primary key for the specific user in our Database. When Foody would like to access the data in our database they have to use the decoding method to generate the original ID of the user and relate it to their Database.

3.3.3.2 Activity Logging

In order for a user to use our Like/Dislike and Journey features the user has to be logged in to Foody. So, a person has to know the user's credentials in order to access those two features of the system with the preferences of the specific user. In terms of the recommender system, no user has access to it as it is going to run on the background by collecting data from Foody's database and generating recommendations attributes through its algorithm.

3.3.3.3 Restrictions on Inter Module Communications

We collect our inputs only from Foody's Database, and only Foody has access in our Database. Our communication for reading and sending critical information of the user are going to pass through the encoding and decoding algorithms explained in part 3.3.3.1.

3.3.3.4 Data Integrity Checks

Each user has to have only one instance in the database. Each time we receive an ID from Foody we perform a check whether the ID exists already or nor. In case it is not we create a new record, otherwise we update the current data of the user.

A product can't exist at the same time in the Like and Dislike list, because we will have problem whether we should recommend the product or not to the user.

3.3.4 Maintainability

The system will be designed using the principles of components and objects; something that will help to maintain the FRA system, isolate problems at components and objects and also apply changes to the different components without affecting significantly the other components.

Moreover, Good documentation of the code will be applied in order to keep the code as much as possible readable for any type of programmers.

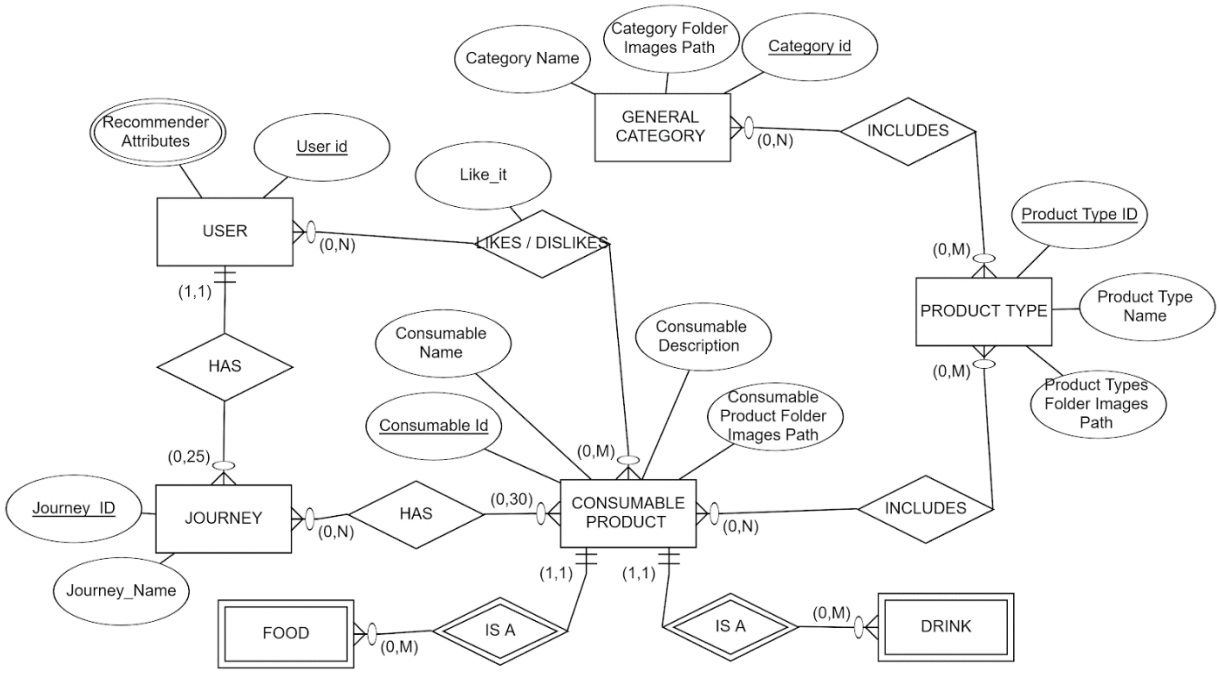
The way in which the Food Recommendation Agent system and Foody Online Orders system communicate operates under some level of abstraction. What this means, is that the FRA system doesn't know and doesn't need to know how the Foody Online Orders system works. It only needs some data. So even the Foody Online Orders system changes significantly the FRA system doesn't have to change (if we suppose that the data returned have the same structure). The same applies for the Foody Online Orders system. It won't know and it won't be necessary to know about the FRA system. The only think that it will have to know is the exact structure of the data coming from the FRA system. Any change in the FRA system should not and won't be affecting the Foody Online Orders system, in terms of design and implementation

3.3.5 Portability

The software can easily be split into two individual components. First the Recommender System and second the Likes/Dislikes with the Journeys feature. So, we can run those two subsystems into two different servers. The only communication that those two systems have among them are through our Database. The communication between our systems can be established only between Foody's Database and our Database. These factors can let us execute the Recommender System in a much more powerful Server in the future if Foody decides to expand itself outside of Cyprus.

3.4 Logical Database Requirements

The Entity-Relationship Diagram shown below is made using the ERDPlus software, free available online at ***https://erdplus.com*** .



Here a desired level of abstraction is kept in order not to make any action that can limit the functionalities of the Database later. Therefore, a lot of things are kept in a specific layer of abstraction as it can be seen from the diagram above and from the explanations above.

- In general, the main type of data that are going to be used are:
  - different kinds of foods (with their characteristics and images)
  - user information (what he likes/dislikes, the input to his journeys, his credentials)
- The number of Foods/ Drinks users like, or dislike can be as many as all the Foods/Drinks exist. That's because all the Foods and Drinks aren't as many as possible to lead storing more than 300KB memory space for each of the users. Despite this possible restriction may be applied in functional phase of the Database. This has a general purpose not to limit from early the Database relationships on entities.
- Since, a user of Foody can log in to his account, it has to be possible to access the database (only the part that he is allowed to and only through the graphical part of the webpages) at any time.
- The Recommender Attributes is defined here as a multivalued attribute, but later it can be transformed into a table with entries for each user or a constant number of attributes for each user, after a specific number of attributes will be defined.
- The attributes which contain the "Folder Images Path" substring is related to the path in the server where the images associated with the meaning of the entity (on which the attributes refer) exist.

3.5 Other Requirements

There are some factors needed to guarantee a defined level of availability for the System. Foody has to constantly collect the orders history for each client. This is necessary for the Recommender System in order to calculate the recommendation attributes per user. In addition, the users should rate the restaurants to increase the quality of our results.

The users that are using the Likes/Dislikes feature are able to get more precise results such as the exact kind of food to be recommended and not just the restaurant. For example, the system could be able to recommend restaurants that make Caesar Salads (it could be any restaurant) and not just restaurants that only make types of Salads.

For the Journeys exploration, grid is going to be used for keeping the food's pictures in it. This will help for the arrangement of the pictures by keeping the same padding. Also, by accessing the grid, you will have access to all the pictures than just traversing all the pictures one by one. Imagine a normal case scenario having 20 pictures at the beginning and after 10 expansions of 5 pictures each we will end up with 70 pictures in total. So, it is more functional to keep one ID for the grid and have access to all 70 pictures through it, than keeping 70 IDs for each picture.

The Foody will have administrator access to change the content of the Journeys in order to add, edit or remove a food. The pictures and the names of each food will be stored in a table of Foody's Database in which we will have access to them to generate the Journeys, this process will be performed in an abstract way in order to give to the content some maintainability. So, Foody is going just to change the content of its table and our system will generate the related foods with description in the Journey's grid.

4. APPENDICES

4.1 Meeting Program

Time	Subject
0 - 3:30	The welcoming from the client/Introduction
3:30 - 9:15	Informing the client about our project and discussion about the potential use
9:15 - 12:21	Analysis of Mr. Panagiotis requirements and how they can be merged with out project
12:21 - 14:45	Discussion about the technologies that will be used in the development of the software
14:45 - 30:45	More discussion about the requirements
30:45 - 32:50	Discussion about the management of data that we will receive from Foody and the implementation of the back-end part
32:50 - 43:05	Discussion about the interface
43:05 - 44:15	Discussion about the programming languages that will be used
44:15 - 50:00	General discussion about the company's goals and future/Epilogue