Addis Ababa University
Addis Ababa Institute of Technology
School of Electrical and Computer Engineering

Restaurant e-menu with recommendation system
Software Requirement Specification (SRS)

Group members

- Alemayehu Ebissa
- Habtamu Bogale

Table of Contents

CHAPTER ONE

INTRODUCTION

1. Introduction

This section gives a scope description and overview of everything included in this SRS document. Also, the purpose for this document is described and a list of abbreviations and definitions is provided.

1.1 Purpose

The purpose of this document is to give a detailed description of the requirements for the restaurant e-menu and recommendation software. It will illustrate the purpose and complete declaration for the development of system. It will also explain system constraints, interface and interactions with other external applications.

1.2 Scope

The restaurant e-menu with recommendation (RER) software helps people by replacing the paper menu and lets manager to post only currently available items on a web. This allows users to view only currently available items and orders online. User (customer) is recommended based on his past experience.

Restaurant managers can provide their restaurant information using the web-portal. This information will act as the bases for the search results displayed to the user. An administrator also uses the web-portal in order to administer the system and keep the information accurate. The administrator can, for instance, verify restaurant owners and manage user information.

The software needs internet and any device that can connect to internet and browse to let customers order online. The system contains a database which is providing lists of

users, recipes, category of recipes, recipes' ingredients. This database enables to keep the information of recipes and users.

1.3 Definitions, Acronyms, and Abbreviations

User – someone who interacts with the system.

Customer – individual who get his/her meal from restaurant.

Client – hotel and restaurants that use this system.

Admin/administrator – system administrator who is given specific permission for managing and controlling the system.

1.4 Overview

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter. The third chapter, Requirements Specification section, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product. Both sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different language

CHAPTER TWO

OVERALL DISCRIPTION

2. Overall Description

2.1 Product perspective

Restaurant e-menu with recommendation system is in need of internet to improve the customers dining experience. Our system is web-based system implementing client server model. The system can able the restaurant for a better usage interaction with their customer on their product and unnecessary cost. This project will be completed in span of 9 weeks. The project constitutes of web development and generating machine learning model.

2.2 User classes and characteristics

Customer

- Login if he has an account or register if he wants to have an account
- Get recommendation based on different parameters
- View meal and its descriptions
- Place order and get notification
- Give feedback

Manager

- Login to his account
- Create product categories and descriptions
- View orders, approve orders and notify customers
- Activate and deactivate customer account

Meal deliverer

- Login
- View customer order and customer details

- Notify that the meal is delivered

Recommender class

- Collect customer preference data
- Analyze customer previous feedback
- Recommend meals to customer
- Predict customer behavior to restaurant

2.3 General Constraints

Since the software is a web-based application, one of the major constraining factors is an internet connection. Users should be able to connect to the internet in order to enjoy any of the services available by the system.

2.4 Assumptions and Dependencies

We have assumed that users, both customer and client, are familiar enough with basic computer operation and have an internet access.

We assumed hotel and restaurant will provide free Wi-Fi connection.

CHAPTER THREE

DETAILED REQUIREMENTS

3. Detailed Requirements

3.1 External Interface

3.1.1 User Interfaces

The user interface for the software shall be compatible to any browser such as Internet Explorer, Mozilla or Netscape Navigator by which user can access to the system. Customer, optionally, will have an account to get access using this system or s/he can get service anonymously. Every time a user requires to access its account a login page appears to authorize the user. The restaurant web portal will have three different pages based on the role as customer, manager or meal deliverer. Manager and meal deliverer start at log in page opens as shown in fig below. Customer can be redirected to login page based on demand to login into its account. After account is authorized different page displayed for the user based on their role.

3.1.2 Hardware interfaces

Since neither the mobile application nor the web portal has any designated hardware, it does not have any direct hardware interfaces. The connection to the database server is managed by the underlying operating system on the mobile phone, web browser and the web server.

3.1.3 Software interfaces

The web portal is a web application, and it run in every operating system using in any browser. The mobile app is an android application and it only run on android OS. There are also software interfaces with third party software to provide complete system. The basic third-party service provider used in this system are TensorFlow and Firebase. TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It

is a symbolic math library, and is also used for machine learning applications such as neural networks. For this project TensorFlow is specifically used for the recommendation system. We need to train a model for each registered customer based on their preferences and previous dining experience.

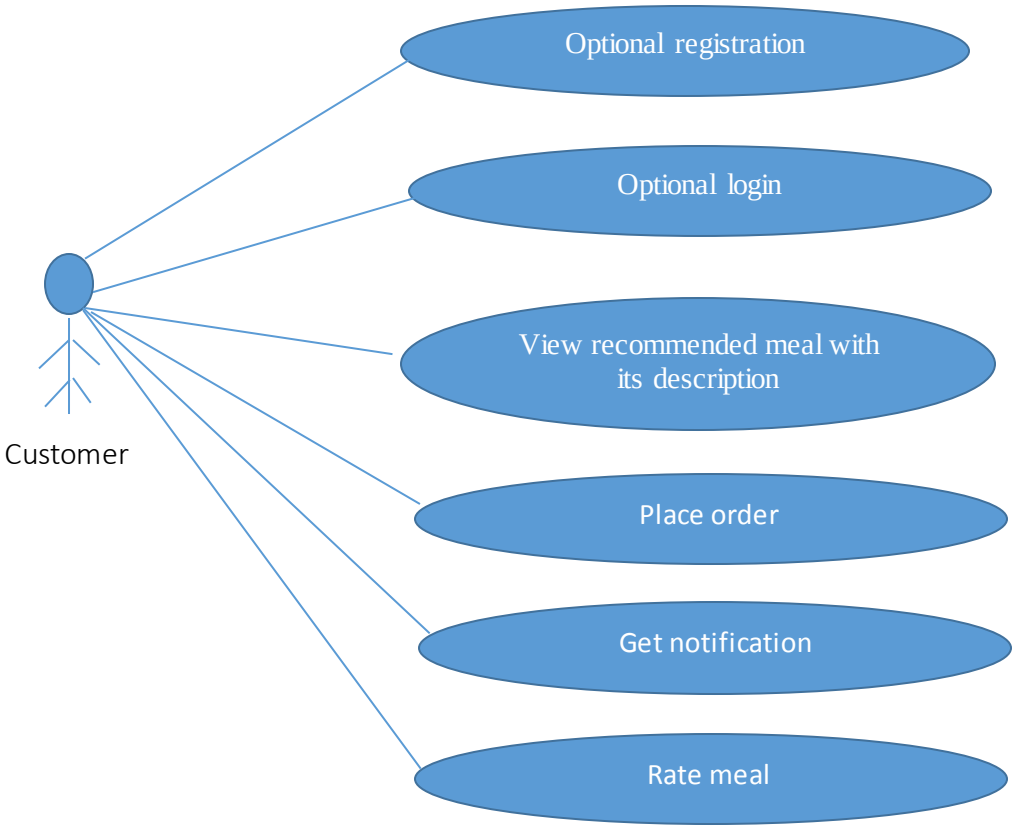
3.1.4 Communications interfaces

The communication between the different parts of the system is important since they depend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the underlying operating systems for both the mobile application and the web portal. Since the system uses some sort of communication with the third-party software's the communication interface is clearly affected by the efficiency of their service. The communication between the firebase database and the web portal consists of operation concerning both reading and modifying the data, while the communication between the database and the mobile application consists of only modifying operations.

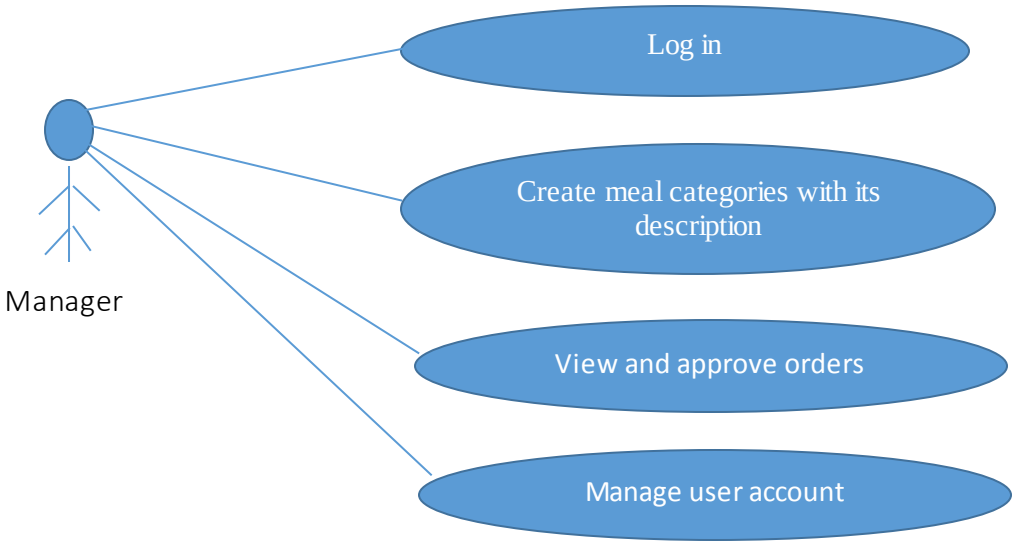
3.2 Functional requirements

3.2.1 Use Case Diagram

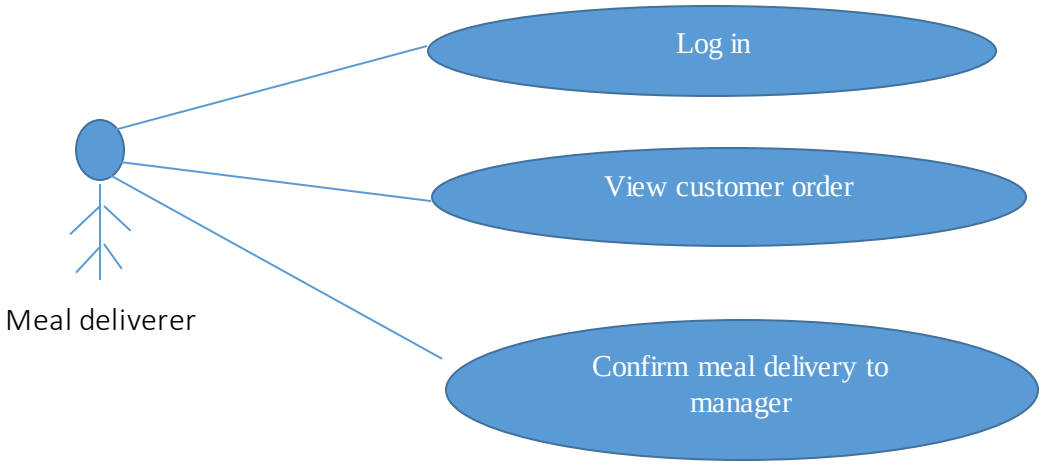
- Customer use case



- Manager use case



- Meal deliverer



3.2.2 Use Cases for functionality

UC 1: Register User

Use case name: Register (customer, client).

Primary Actor: User.

Preconditions: User has no account.

Main success scenarios:

1. System sends registration form.
2. User fills the form with required information and submits.
3. System checks the validity of the information submitted.
4. System registers user in the database and grants access to its account based on its role assigned.

Exception scenarios:

2a. Invalid (illegal) information submitted

- System asks user to enter legal information.

2b. User with same information exists.

- System notifies that a user with same credentials exist and asks to change his credentials.

UC 2: Login

Use case name: Login

Primary Actor: User

Preconditions: User has created an account.

Main Success Scenario:

1. System sends log form.
2. User: - Enters his/her credential.

Exception scenarios: System: Can't log in.

- System: - Notifies user it is incorrect password (forget password).

UC 3: Forgot password

Use case name: Forgot password.

Primary Actor: User.

Preconditions: User has created an account.

Main Success Scenario:

- User clicks forgot password link.
- System prompts the user to enter account recovery information.
- User enters information.
- System checks if the entered information is valid.
- System asks to enter a new password.
- System log in user and updates user information.

Exception scenarios:

- Invalid information submitted.
- System asks user to enter valid credentials.

UC 4: Add Food

Use case name: Add new food.

Primary Actor: Manager.

Preconditions: Food is not in a menu yet.

Main success scenarios:

1. System sends add new food form.
2. User fills the form with required information and submits.
3. System checks the validity of the information submitted.
4. System add food in the menu to the database.

Exception scenarios:

2a. Invalid (illegal) information submitted

- System asks user to enter legal information.

2b. Food with same information exist in menu.

- System notifies that a food with same information exist and asks to appropriate action.

UC 5: Edit Menu

Use case name: Edit Menu.

Primary Actor: Manager.

Preconditions: Menu already created.

Main success scenarios:

1. System sends edit menu form.

2. User edits the required information and submits.
3. System checks the validity of the information submitted.
4. System update menu data in the database.

Exception scenarios:

2a. Invalid (illegal) information submitted

- System asks user to enter legal information.

UC 6: Place Order

Use case name: Place new order.

Primary Actor: Customer.

Preconditions: Undefined.

Main success scenarios:

1. System sends place order form with menu and recommendation.
2. User fills the form with required information and submits.
3. System checks the validity of the information submitted.
4. System sends to manager for verification and confirmation.

Exception scenarios:

2a. Invalid (illegal) information submitted

- System asks user to enter legal information.

UC 7: Confirm Order

Use case name: confirm order.

Primary Actor: Manager.

Preconditions: New order received.

Main success scenarios:

1. System sends pending order.

2. Manager confirm and forward it to cooker.
3. System sends requested user the order is placed.

Exception scenarios:

2a. Manager rejected the order

- System notifies the requested user.

UC 8: Notification:

Primary Actor: System

Precondition: None

Main Success Scenario:

1. The system send notification to the manager at the time of sending the letter for getting the service and show how many employee has sent the letter

Exception Scenario:

2. The notification has sent is out of the date
3. Unable to reach the manager

3.3 Performance Requirements

The performance requirement clearly specifies the static and dynamic performance of the system. Static requirement such as capacity is listed and on the other hand dynamic requirements like response time, database request and size of the webpage is described below.

Capacity of the System

Title: Capacity

Description: The terminals and simultaneous users should be supported in a maximum number.

Rationale: In order for users to enter and visit a site easily.

Must: it's should allow more than 1000000 customers to register and manageable by the system and enjoy the service.

Wish: All terminals and users should be supported during testing

The size of Webpage

Title: The size of Webpage

Description: The size of the page should be minimized.

Rationale: Make the response time fast enough.

Must: The webpage is expected to be less than 600kB during testing.

Wish: Expected to be under 200kB during testing.

Database Requests

Title: Database requests

Description: Data request should be accessed easily and quickly.

Rationale: to access the car's profile and driver's profile immediately throughout the website.

Must: Data caching and replication methods should be used whenever possible

Design constraints

Constraints, as the dictionary definition indicates, are a limiting factor and severely restrict options for making design decisions. They are also fixed design decisions that cannot be changed and must be satisfied. You could think of constraints as the ultimate nonnegotiable, "must have" requirement. There are number of factors that may limit our design choices. Our cars fleet management system come in many constraints but cars tracking is the main constraints.

Recommendation problem

A recommender system is a software that exploits user's preferences to suggests items (movies, products, songs, events, etc….) to users. It helps users to find what they are looking for and it allows users to discover new interesting never seen items. Two different approaches may be adopted to solve the recommendation problem:

Content-based filtering methods are based on a description of the item and a profile of the user's preferences. In a content-based recommender system, keywords are used to describe the items and a user profile is built to indicate the type of item this user likes. In other words, these algorithms try to recommend items that are similar to those that a user liked in the past (or is examining in the present). In particular, various candidate items are compared with items previously rated by the user and the best-matching items are recommended. This approach has its roots in information retrieval and information filtering research.

Collaborative filtering is a method of making automatic predictions (filtering) about the interests of a user by collecting preferences or taste information from many users (collaborating). The underlying assumption of the collaborative filtering approach is that if a person A has the same opinion as a person B on an issue, A is more likely to have B's opinion on a different issue than that of a randomly chosen person. In the more general sense, collaborative filtering is the process of filtering for information or patterns using techniques involving collaboration among multiple agents, viewpoints, data sources, etc. Applications of collaborative filtering typically involve very large data sets.

Reliability and Fault Tolerance

If we cannot avoid a failure, then we must constrain the software design so that the system can recover in an orderly way. Each software process or object class should provide special code that recovers when triggered. A software fault tolerant library with a watchdog daemon can be built into the system.

Hardware Limitations

There are no specific requirements on the expected hardware. We expect that all users (manager and meal deliverer) use the core platform in a restaurant or on a notebook/ tablet PC. And Customer use its own phone or notebook/tablet PC.

Security

During login time, there will be numerous invalid login attempts. Accordingly, the system may not work correctly, so in those cases securities is the main issue. These cases include those listed above in addition to forcing users to use “strong” passwords. A strong password is a password that meets a number of conditions that are set in place

so that user's passwords cannot be easily guessed by an attacker. Generally, these rules include ensuring that the password contains a sufficient number of characters and contains not only lowercase letters but also capitals, numbers, and in some cases, symbols.

Attributes

Software system attributes

Quality attribute requirements are part of an application's nonfunctional requirements which capture the many facets of how the functional requirements of an application are achieved. All but the most trivial application will have nonfunctional requirements that can be expressed in terms of quality attribute requirements. The requirements in this section specify the required reliability, availability, usability, security and maintainability of the software system.

Reliability: The capability to provide failure-free service.

Maintainability: The capability to be modified for purposes of making corrections, improvements, or adaptation.

Availability: means the presence of the system during accessing from different users.

Portability: The capability to be adapted for different specified environments without applying actions or means other than those provided for this purpose in the product

Reliability

OVERVEIW: The reliability of the system.

SCOPE: The reliability that the system gives the available car in the search time

TEST: Measurements obtained from 500 searches during testing.

REQUIERED: More than 98% of the searches.

TARGATE: More than 99% of the searches.

DESIRE: 100% of the searches.

Availability

OVERVEIW: The ease of access of the system when it is accessed.

SCOPE: The average system availability (not considering network failing).

TEST: Measurements obtained from 1000 hours of usage during testing.

REQUIERED: More than 98% of the time.

TARGATE: More than 99% of the time.

DESIRE: 100% of the time.

TITLE: Internet Connection

DESCRPTION: The application should be connected to the Internet.

RATIONAL: In order for the application to communicate with the database.

DEPENDENCY: none

Security

ID: Communication Security

OVERVEIW: Security of the communication between the system and server.

SCOPE: The messages should be encrypted for log-in communications, so others cannot get user-name and password from those messages.

TEST: Attempts to get user-name and password through obtained messages on 1000 log-in session during testing

REQUIERED: 100% of the Communication Messages in the communication of a log-in session should be encrypted.

Communication Messages: Defined every exchanged of information between client and server.

ID: Customer Login Account Security

OVERVEIW: Security of accounts.

SCOPE: If the system owner tries to log in to the web portal with a non-existing account then the system owner should not be logged in. The system owner should be notified about log-in failure.

TEST: 1000 attempts to log-in with a non-existing user account during testing.

REQUIERED: 100% of the time.

ID: Client Login Account Security

OVERVEIW: Security of accounts.

SCOPE: If a client tries to log in to the web portal with a non-existing account then the client should not be logged in. The client should be notified about log-in failure.

TEST: 1000 attempts to log-in with a non-existing user account during testing.

REQUIERED: 100% of the time.

ID: System User Account Security

OVERVEIW: Security of user accounts.

SCOPE: A user account and IP address should not be able to log-in for a certain time period after three times of failed log-in attempts.

TEST: 1000 attempts to log-in during the lock period after user account has been locked because of failed log-in attempts of three times.

REQUIERED: The locking period should be half an hour, and during that period the log-in function is disabled.

ID: Admin Account Security

OVERVEIW: Security of admin accounts.

SCOPE: An admin and IP address should not be able to log-in to the web portal for a certain time period after three times of failed log-in attempts.

TEST: 1000 attempts to log-in during the lock period after user account has been locked because of failed log-in attempts of three times.

REQUIERED: The locking period should be half an hour, and during that period the log-in function is disabled.

ID: User Create Account Security

OVERVEIW: The security of creating account for users of the system.

SCOPE: If a user wants to create an account and the desired user name is occupied, the user should be asked to choose a different user name.

TEST: Measurements obtained on 1000 hours of usage during testing.

REQUIERED: 100% of the time.

Maintainability

TITLE: Application extendibility

DESCRPTION: The application should be easy to extend. The code should be written in a way that it favors implementation of new functions and adaptive to new requirements.

RATIONAL: In order for future functions to be implemented easily to the application.

DEPENDENCY: none

TITLE: Application testability

DESCRPTION: Test environments should be built for the application to allow testing of the application's different functions.

RATIONAL: In order to test the application.

DEPENDENCY: none

Portability

TITLE: Application portability

DESCRPTION: The application should be portable with windows, Linux and Android.

RATIONAL: The adaptable platform for the application to run on.

DEPENDENCY: none

Usability

TITLE: Application usability

DESCRPTION: the application should be easy to understand and no need to take training to use it.

DEPENDENCY: none