Software Requirements Specification

for

Grocery List Maker

**Version 1.0 approved**

**Prepared by Group No.2**

**Section No. 3**

**14/10/21**

**Table of Contents**

1. [**Introduction**](#_1fob9te)1
   1. [Purpose](#_3znysh7) 1
   2. [Document Conventions](#_2et92p0) 1
   3. [Intended Audience and Reading Suggestions](#_tyjcwt) 1
   4. [Product Scope](#_3dy6vkm) 1
   5. [Reference](#_1t3h5sf)2
2. [**Overall Description**](#_4d34og8)3
   1. [Product Perspective](#_2s8eyo1) 3
   2. [Product Functions](#_17dp8vu) 3
   3. [User Classes and Characteristics](#_3rdcrjn) 4
   4. [Operating Environment](#_26in1rg) 5
   5. [Design and Implementation Constraints](#_lnxbz9) 5
   6. [User Documentation](#_35nkun2) 6
   7. [Assumptions and Dependencies](#_1ksv4uv)6
3. [**External Interface Requirements (To be filled later)**](#_44sinio)7
   1. [User Interfaces](#_2jxsxqh) 7
   2. [Hardware Interfaces](#_z337ya) 7
   3. [Software Interfaces](#_3j2qqm3) 7
   4. [Communications Interfaces](#_1y810tw)7
4. [**System Features**](#_4i7ojhp) **8**
   1. [Account Management](#_2xcytpi) 8
   2. [Grocery List Prediction](#_y1ll75dcvzi6) 15
   3. [Data Management](#_cfgp34ij8uj3) 19
   4. [Help and Support](#_5ie901dhztl0)24
5. [**Other Nonfunctional Requirements**](#_3whwml4)27
   1. [Performance Requirements](#_2bn6wsx) 27
   2. [Safety Requirements](#_qsh70q) 27
   3. [Security Requirements](#_3as4poj) 27
   4. [Software Quality Attributes](#_1pxezwc) 27
   5. [Business Rules](#_49x2ik5)28
6. [**Other Requirements**](#_2p2csry) **28**

**Appendix A: Glossary 29**

**Appendix B: Analysis Model 29**

**Appendix C: To Be Determined List 29**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

Our product for Software Engineering Group 2 is a Grocery List Maker apk app developed on Flutter limiting the app to play store users only. This app aims to assist daily grocery shoppers in managing and purchasing their groceries. It’ll provide them with different options of grocery lists based on their previous purchases and guide them to the store based on their convenience. It targets a wide group of people who have trouble managing their groceries and lack the knowledge required for effective shopping. It also provides the functionality to create a family list shared across all members to cut the communication gap between family members required for grocery shopping.

## Document Conventions

By default, all text is written in Arial font size 11 with headings in size 14. This document is divided into 6 sections that increase in complexity from 1 to 4, one being the simplest to 4 being the most relevant to the software developer. All subheadings are made **bold** ,while there is no use of *italics* till this version.

## Intended Audience and Reading Suggestions

This document is intended for developers, project managers, clients and testers. This SRS contains six sections starting from Introduction in section 1, Overall Description in section 2, External Interface Requirements in section 3, System Features in section 4, Non-Functional Requirements in section 5 and Other Requirements in section 6. Project Managers will begin from section 1 all the way to section 6 since it is important for them to know every single thing. Developers should skip section 1 as it doesn't really matter if they go through it or not. Testers should begin from section 3 because any information before that is irrelevant to testing. Clients should read all the sections with assistance if required.

## Product Scope

The software that is being used is a Smart Grocery List, providing optimization benefits to the users of the apps. The main objectives include the benefits for the product. The User can benefit from the automatic Grocery List and work better in their personal lives. Users can facilitate and make their preference of grocery list manually,

Primarily, the scope pertains to the E-Store product features for making Marvel Electronics and Home Entertainment projects live. It focuses on the company, the stakeholders and applications, which allow for online sales, distribution and marketing of electronics.

This SRS is also aimed at specifying requirements of software to be developed but it can also be applied to assist in the selection of in-house and commercial software products. The standard can be used to create software requirements specifications directly or can be used as a model for defining an organization or project specific standard. It does not identify any specific method, nomenclature or tool for preparing an SRS.

## Reference

The summary of the project was made on Trello and can be found [here](https://trello.com/invite/b/ABXL2Dxv/be2b95c0e234245bd9b9e426aeb61756/se-project)

https://trello.com/invite/b/ABXL2Dxv/be2b95c0e234245bd9b9e426aeb61756/se-project

# Overall Description

## Product Perspective

The product perspective for this system is simple. It provides an optimization benefit to predict a list based on previous buying patterns. There are no comparison-oriented applications for this. The main benefit goes to the user who can choose the items as he wants them to be. The items can be categorized on the basis of cost as well as ease/fuel cost. This product is totally new for our users to use and test. Usually, a User may go ahead with a Simple Shopping Application or a To-Do List. While that is the norm, it definitely is not the ideal way to Shop. It’s much better to check shops easily from the Application to know how cost control will be done and time can be saved.

The product's perspective for our users is that they benefit from their items as well as the Stores benefit through the sales generated by the Application. The Stores are general, like Imtiaz, Carrefour who will be provided with a dedicated merchant portal and this will be used to manage their end of the rope. The Perspective from the Merchants point of view will be that they will get sales and traffic from sales items(future requirement, low priority). On the other end, the Application will predict the items to be used for a user based on their specific preferences and then link the Application and Item list to the desired store, hence connecting the User with the merchant store. The combined picture will be the User benefiting from the Application and the Merchant benefiting from the Sales, although we’re not going to implement the merchant point of view but can be considered in future.

The application will be standalone, and work on Android Systems. It does not depend on any other systems and works with comfort and ease on its own.

## Product Functions

***Objective: Ease Grocery buying and instead of buying and adding items each time, having an Application by your Side that handles the complex work for you.***

* User will have a dedicated portal to manage and connect with the admin.
* User trends will be picked up by the Application and a Model will be trained, providing ease to the user.
* Items will be generated as per user’s preferences and needs. We call this feature **‘Predicted List’**.
* Items to be bought will be selected by the user to generate the final list
* A final list will be generated and then sorted as per the preference of the user. This can be either a timing basis or a monetary basis.
* The user can modify the predicted list according to their convenience.
* The user will then be provided with an opportunity to buy the items from a preferable store training the AI more efficiently.
* Complete FAQ and help to be provided by our Software, to absolve any possible issues that may exist.

## User Classes and Characteristics

Our general end-user is a very simple person. This application can be used by any person that needs to shop often, or from time to time. However, we have decided to categorize the user on the following basis.

* **Frequent User:**

This is a type of user who very frequently uses our application. This customer has a high frequency of usage, as well as high dependency on the application. For all our customer classes, the available product functions will be the same. This means, simply adapting to their pattern of choice and working out a feasible, viable and correct list for themselves. Same goes for the Security consideration. It is our highest priority to predict an accurate list for this customer. This means that to perceive the goodness of this app, one must understand some basics behind its functionality. This user is an important type for us as he will be our main facilitator for the Application.

* **General User:**

This is the user that does not depend much on our application. Basically, this is a generic user who might occasionally use our application. As evident, these will be people opting to use the Application for generic purposes. It is important to note that however, a general user will be less influenced by the Application and its technical details. A certain subclass of these users may also perceive the application as another To-Shop List and use the App as such. It is important to note that such users will always exist, however we have to understand that those people who understand and appreciate the value of the existence of our application are most important to us and our application.

* **Merchant Store Maintainer (to be implemented in the future):**

This is the User Class that will be using the Merchant Application to maintain the merchant's end of system. The education level for this user class needs to be high enough to maintain the Merchant’s end of System. Merchants need to maintain their information using this app for their own benefits. Our whole connection system depends on this system. This user is an important type for us as he will be our main facilitator for the Application behind the scenes functionality.

## Operating Environment

Our App is a simple mobile application. Our System will run on a Server and our App will be used by Android Application.

Both User and Merchant need an android account to connect and operate in the system. Most of the development features are supported in Android through its development methods, so implementation of the Application is scalable and durable.

**Hardware Requirements:** 40GB Hard Disk (for the developer), 40MB Storage(for the user) for Storage and 2 GB Ram.

Software wise it is a simple application so external dependencies are minimalistic.

## Design and Implementation Constraints

Our regulation will be done as per the SECP for the Pakistani market and for other countries, as per the legal requirements that might be required. Our Corporate requirements will be as per the region as well. However, we have to note that there are hardware limitations as well. Our system should not be overloaded with data so most of the work will be done on Client’s end. Our Memory needs to be optimized and so do the timing requirements. The App needs to respond faster to queries and responses required by the user.

**Technologies:** Flutter for Android App development

**Main Language:** Dart

**Prediction language:** Python 3

**Database:** Oracle Database 19c

**Team Communications**

Our whole communication will take place on Trello, GitHub and WhatsApp.

**App Communications**

Using Http protocols.

**Security**

Access limitation, data integrity and security to be our top concern. Only allow senior most members of the Company to manage and run the App for security reasons and employ strong passwords for fool-proof security of the Application.

**Programming Standards**

Camel casing methodologies and proper naming standards will be followed. As well as to respect all the logical constraints in development and object-oriented class modelling.

**Input output and App management:**

Done on users end on his Phone.

## User Documentation

We have taken certain steps to ensure the usage of our application becomes easy for our users. Our System contains an app tutorial which is designed and developed using special considerations in mind.

We plan to release first and foremost, a list of FAQs Sections that engages the user in the top-most easy questions. These questions will be general and ensure that the user does not face any problem at all levels of the program. The FAQs will contain general usability questions and answers, in order to facilitate the user. Further there will be dedicated backup management for our App Users that allows ease of storage and efficient functionality.

## Assumptions and Dependencies

* We assume that the costing of Development will be low in Flutter.
* We also assume all stores have interfacing facilities for shopping online.
* We assume that our design, implementation and development will be user friendly and smoothness oriented.
* We assume the Application to run without facing bugs since we plan on testing the application beforehand, and to be functional 24/7 whenever requested or opened by the Application.
* Our users have access to the internet and nearby stores that we designate as merchants.
* Our Merchants are responsive and active over the development, and maintenance of their portals. And also, they are dedicated members protecting the trust of our company and our customers.(future requirement)
* The Application, when run, has the ability to run without errors and in case errors surface, a proper debug and error finding team is available to resolve the issue

# External Interface Requirements (To be filled later)

## User Interfaces

*<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>*

## Hardware Interfaces

*<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>*

## Software Interfaces

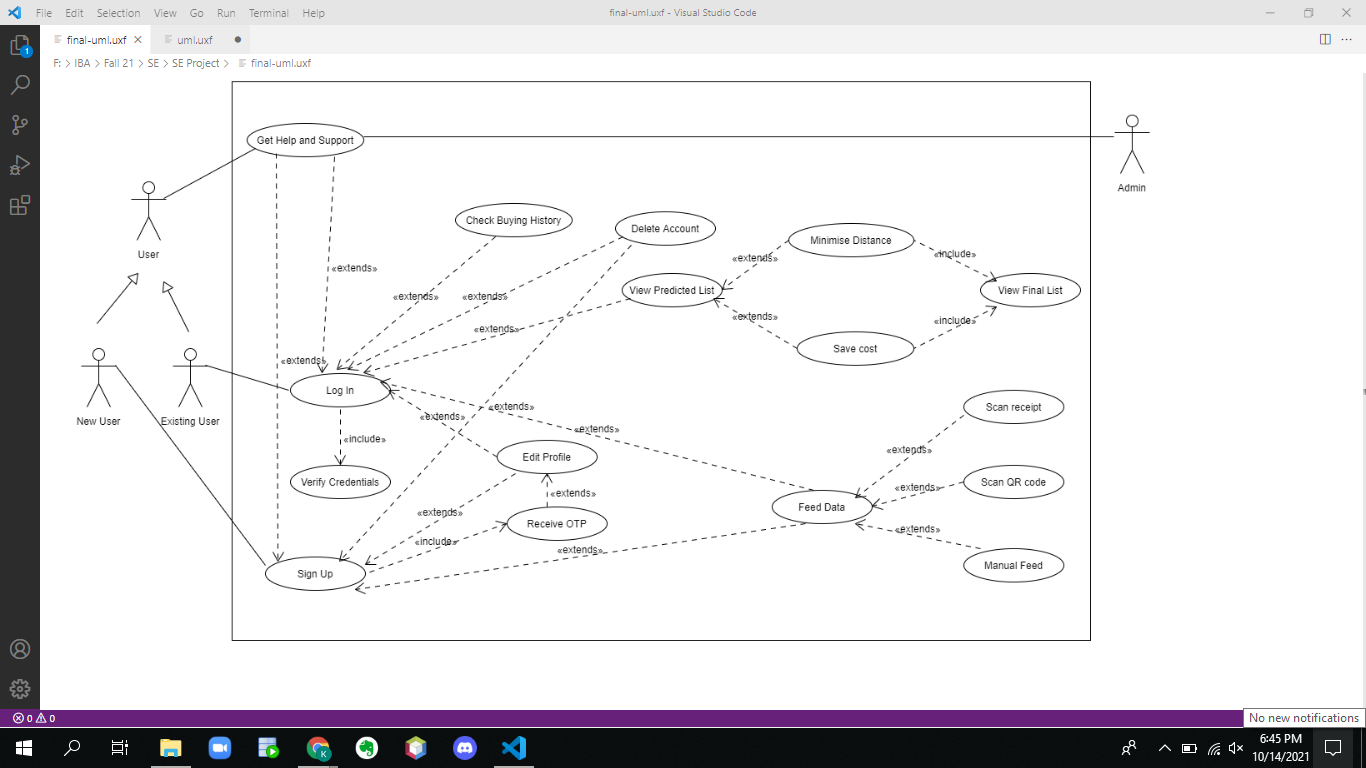
*<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>*

## Communications Interfaces

*<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>*

# System Features

### Use Case Diagram:

****

## Account Management

**4.1.1 Description and Priority**

This feature will allow the user to create an account so that they have access to the application. After this they will be able to log in and use the application. Through this feature, the user will be able to edit their profile information. They can also delete their account if they wish to stop using the application permanently. Their credentials will be verified when they log in every time and receive a one-time passcode when he makes a new account. This is a high priority feature.

**4.1.2 Stimulus/Response Sequences**

**Use Cases:**

* **Sign Up**

The application will ask the user if they are an existing user, or a new user. If a new user, they will proceed to the sign-up screen where they will add all information and a new account will be created.

* **Log In**

The application will ask the user if they are an existing user, or a new user. If an existing user, they will proceed to the Login screen where they will add their credentials and if correct, they will be given access to the application.

* **Delete Account**

The user will have a ‘Delete Account’ button from account settings, which they can click to delete their account permanently after which they will be prompted for confirmation.

* **Edit Profile**

The user will select the Edit Profile Icon and make the necessary changes in the data. The application will make the changes in the database as required.

* **Verify Credentials**

This feature will be used to authenticate a user's ID and password from the database so he can have access to his profile and the application.

* **Receive OTP**

A new user will receive a one-time passcode on either his email or contact number to verify that it is actually him signing up to the account. He will then enter this one-time passcode to finish signing up.

**4.1.3 Functional Requirements**

* **AM1**: Users should be able to create an account.
* **AM2**: User will receive a one-time password(OTP) on email provided which will be entered by the user.
* **AM3**: User should be connected to the internet while signing up.
* **AM4**: User should be able to add their details in respective fields.
* **AM5**: User should be able to log in to their account when they enter correct credentials.
* **AM6**: User can reset his password if forgotten.
* **AM7**: Access should be denied access if login fails.
* **AM8**: User should be connected to the internet while logging in.
* **AM9**: User should be able to change his details.
* **AM10**:The changes made by the user should be saved.
* **AM11**:If the user wishes to change his email, the email will be verified again.
* **AM12**:If a user changes his password, an authentication code will be received on their email.
* **AM13**: User can delete their account
* **AM14**: User will be given a warning before deleting their account
* **AM15**: User is asked the reason for deletion
* **AM16**: User’s email should be valid so that the OTP email can be sent.
* **AM17:** User must enter the OTP within 10 seconds
* **AM18**: User can choose resend code option to get a new code
* **AM19**: User must have an existing account.

|  |  |
| --- | --- |
| **Use case name** | **Sign Up** |
| Related Requirements | AM1, AM2, AM3, AM4 |
| Goal in context | New user is able to make a new account |
| Pre conditions | The user has appropriate proof of identity |
| Successful End conditions | The user manages to create an account |
| Failed end conditions | The user was not able to sign up |
| Primary Actors | New User |
| Secondary Actors | None |
| Trigger | When sign up button is clicked |
| Included cases | Receive OTP, Feed Data, Delete Account, Get Help and Support, edit profile |
| Main Flow | |  |  | | --- | --- | | Step | Action | | 1 | User will be shown sign up/login window | | 2 | User will click sign up | | 3 | Sign up User Interface will become visible | | 4 | User will enter their details | | 5 | User will receive an email with OTP which will be entered by the user. | | 6 | Username and password will be added to the database | | 7 | User will choose their account type (Joint/ individual) | | 8 | The user will be redirected to the home page of the app | |

## 

|  |  |
| --- | --- |
| **Use case name** | **Log in** |
| Related Requirements | AM3, AM5, AM6, AM7, AM8 |
| Goal in context | User has added all the requirements |
| Pre conditions | User already has an account |
| Successful End conditions | User has able to add all the requirements |
| Failed end conditions | User wasn't able to add all the requirements |
| Primary Actors | Existing user |
| Secondary Actors | None |
| Trigger | When log in button is pressed |
| Included cases | Feed Data, Get help and Support, Delete Account, View Predicted list, Check Buying History, Edit Profile, Verify Credentials |
| Main workflow | |  |  | | --- | --- | | Step | Action | | 1 | User will be shown sign up/log-in button | | 2 | User will click log-in | | 3 | Login User Interface will become visible | | 4 | User will enter their username and password | | 5 | If user has forgotten their password, they can use the option of forgot password to reset their password | | 6 | Username and password will be verified | | 7 | User will select their account type (Joint/ individual) | | 8 | The user will be redirected to the home page of the app | |

|  |  |
| --- | --- |
| **Use case name** | **Delete Account** |
| Related Requirements | AM13, AM14, AM15 |
| Goal in context | User is able to delete their account |
| Pre conditions | User has an account |
| Successful End conditions | User’s account is deleted |
| Failed end conditions | User’s account is not deleted |
| Primary Actors | Existing User |
| Secondary Actors | none |
| Trigger | User wishes to delete their existing account |
| Included cases | None |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | User chooses the option to delete the account | | 2 | User is displayed a prompt confirming if he is sure about deleting the account | | 3 | User confirms by clicking at yes or no | | 4 | User is asked for the reason for deleting the account | | 5 | User states the reason | | 6 | Account is deleted after reason is submitted | |

|  |  |
| --- | --- |
| **Use case name** | **Edit Profile** |
| Related Requirements | AM9, AM10, AM11, AM12 |
| Goal in context | User can change his profile details |
| Pre conditions | Account already exists |
| Successful End conditions | User is able to edit profile |
| Failed end conditions | User is not able to edit profile |
| Primary Actors | Existing User |
| Secondary Actors | none |
| Trigger | When the user clicks the Edit Profile icon. |
| Included cases | receive OTP, edit profile |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | User will click the edit profile button from home page | | 2 | User will click the Edit Profile button | | 3 | User will make the necessary changes | | 4 | If the user wishes to change his email, he will receive an email verification. | | 5 | User can change their password through authentication code | | 6 | Changes made will be updated in the database | | 7 | A prompt that changes are made will be shown on screen | |

|  |  |
| --- | --- |
| **Use case name** | **Receive OTP** |
| Related Requirements | AM16, AM17, AM18 |
| Goal in context | User’s email is verified using OTP |
| Pre conditions | User has provided an email address |
| Successful End conditions | User has a valid email address and it was verified |
| Failed end conditions | User’s email wasn’t verified |
| Primary Actors | User |
| Secondary Actors | None |
| Trigger | When user clicks sign up button after entering all the sign-up details |
| Included cases | none |
| Main Flow | |  |  | | --- | --- | | Step | Action | | 1 | User will receive an email containing the OTP | | 2 | User will enter the code received. | |

## 

|  |  |
| --- | --- |
| **Use case name** | **Verify Credentials** |
| Related Requirements | AM19 |
| Goal in context | To verify user credentials from database |
| Pre conditions | user has entered username and password |
| Successful End conditions | Verifies the credentials correctly |
| Failed end conditions | Fails to verify the credentials correctly. |
| Primary Actors | User |
| Secondary Actors | None |
| Trigger | The login button is pressed |
| Included cases | None |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | User’s username and password will be checked from the database | | 2 | If the username and password are correct then the user proceeds to the next screen | | 3 | If the username and password are incorrect then the user will be sent a warning to enter the correct password | |

## Grocery List Prediction

**4.2.1 Description and Priority**

This feature will predict future grocery lists, based on the user's shopping history. The user will then have the option to finalize the predicted list based on minimum shopping cost or minimum distance between his location and the store. The final list will then be viewed to him based on his preferences which he can then export to his device. He can later view his shopping history by date from the main menu. This is a high priority.

**4.2.2 Stimulus/Response Sequences**

**Use Cases:**

* **View Predicted List**

The user will be asked if he wants to manually feed data, or if he wants a predicted list. If the user chooses the predicted list option, the predicted list will be displayed which can be further modified by the user

* **Minimize Distance**

After the user is shown a predicted list, he is given the option to modify the list according to minimum distance from his location to the suggested stores nearby. If he selects this option, this will be the final list.

* **Save Cost**

After the user is shown a predicted list, he is given the option to modify the list according to minimum shopping cost for all suggested items. If he selects this option, this will be the final list.

* **View Final List**

After he selects his preference, the list shown to him will be the final list.

**4.2.3 Functional Requirements**

* **GLP1**:User should be able to choose his mode of convenience to buy items in the list.
* **GLP2**:User should be able to see the final list based on distance.
* **GLP3**:User should be able to see the final list based on prices.
* **GLP4**:Users should be able view the final grocery list after selecting a mode of convenience.
* **GLP5**:Users should be able to export the final list in pdf or doc format.
* **GLP6**:A predicted list is generated according to the user's buying pattern.
* **GLP7**:Users can add items that are not already predicted by the algorithm.
* **GLP8**:User can remove items from the predicted list that he feels are not required.

|  |  |
| --- | --- |
| **Use case name** | **View Final List** |
| Related Requirements | GLP4, GLP5 |
| Goal in context | Final list is approved by the user is displayed |
| Pre conditions | User has edited and/or approved the predicted list |
| Successful End conditions | Final list is displayed |
| Failed end conditions | Final list is not displayed |
| Primary Actors | Existing User |
| Secondary Actors | None |
| Trigger | When the user has selected the mode of convenience and wishes to view the final list |
| Included cases | none |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | User will choose their mode of convenience for the predicted list | | 2 | Final list will be displayed based on distance, or price. | | 3 | A checklist will be generated that checks against the data items that were purchased by the user. | | 4 | Users can also choose to download the final list in pdf or doc format. | |

|  |  |
| --- | --- |
| **Use case name** | **View Predicted List** |
| Related Requirements | GLP1, GLP6, GLP7, GLP8 |
| Goal in context | Show predicted list according to previous data patterns |
| Pre conditions | some user buying history exists |
| Successful End conditions | Predicted list is generated |
| Failed end conditions | Enough data does not exist to predict a list |
| Primary Actors | Existing user |
| Secondary Actors | none |
| Trigger | When user selects the option to view predicted list |
| Included cases | Minimize Distance, Save Cost |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | User will select the option to view predicted list from home page | | 2 | Predicted list will be displayed | | 3 | User can then edit the list according to their preferences (add/remove) | |

|  |  |
| --- | --- |
| **Use case name** | **Minimize Distance** |
| Related Requirements | GLP2 |
| Goal in context | User should be able to find predicted items at the nearest store |
| Pre conditions | Predicted list has been created at previous stage |
| Successful End conditions | System suggests the nearest store with predicted items |
| Failed end conditions | System fails to suggest the nearest store |
| Primary Actors | Existing User |
| Secondary Actors | none |
| Trigger | User has selected the minimized distance convenience |
| Included cases | View Final List |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | user will opt for the minimized distance predicted list | | 2 | user will proceed to the final list according to the minimized distance predicted list | |

|  |  |
| --- | --- |
| **Use case name** | **Save Cost** |
| Related Requirements | GLP3 |
| Goal in context | User should be able to find predicted items with the least cost |
| Pre conditions | Predicted list has been created at previous stage |
| Successful End conditions | System predicts the least cost items |
| Failed end conditions | System fails to suggest |
| Primary Actors | Existing User |
| Secondary Actors | none |
| Trigger | User has selected the save cost convenience |
| Included cases | View Final List |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | user will opt for the least cost predicted list | | 2 | the user will proceed to the final list according to the least cost predicted list. | |

## Data Management

**4.3.1 Description and Priority**

Through this feature the user has the option to enter his own data through three different modes offered to him: manual feed, scan QR code and image processing. .His data is entered and stored through his preferred mode. He can check his buying history from the main menu whenever he wishes and filter the history by date. This is a high priority.

**4.3.2 Stimulus/Response Sequences**

**Use Cases:**

* **Feed Data**

The user has the option to view a predicted list or feed his own data. He is given three modes to feed data when he clicks on this option.

* **Scan QR Code**

The user can use this feature to scan the QR code behind his old shopping receipts and this data will then be stored as his shopping history.

* **Image Processing**

The user can use this feature to take a picture of his old shopping receipts and this data will then be stored as his shopping history.

* **Manual Feed**

The user will use this feature to manually enter data to make his own shopping list or add items in a predicted list that are not present.

* **Check Buying History**

The user will have a ‘Check Buying History’ button at all times in the menu, which he can click to view his shopping history whenever he wants which will direct him to a page for shopping history.

**4.3.3 Functional Requirements**

* **DM1**:Users should be able to manually add purchased item details.
* **DM2**:Users can search from a list of items that can be bought, and select the one that they have purchased to feed data into the system.
* **DM3**:Users should be able to feed purchased item details using QR code.
* **DM4**:Users can check items from the predicted list that are bought.
* **DM5**:Users should be able to scan the receipt. Images of receipt should be processed, and data should be extracted and saved.
* **DM6**:Users can view previous purchases and keep track of money spent.
* **DM7**:Users can delete their purchasing history

|  |  |
| --- | --- |
| **Use case name** | **Feed Data** |
| Related Requirements | DM4 |
| Goal in context | User feeds purchased item into the list |
| Pre conditions | User exists |
| Successful End conditions | User was able to choose the mode of feeding data |
| Failed end conditions | User was not able to choose the mode of feeding data |
| Primary Actors | Existing User |
| Secondary Actors | none |
| Trigger | User has selected the option to feed data |
| Included cases | scan QR code, Image processing, manual feed |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | User will select the Feed Data option | | 2 | They will choose an option to feed data(QR code, manual feed, Image processing, check list from predicted data) | | 3 | According to the chosen option, the user feeds data to the system | |

|  |  |
| --- | --- |
| **Use case name** | **Scan QR code** |
| Related Requirements | DM3 |
| Goal in context | User successfully scans the QR code from the receipt, which displays the list of items bought. |
| Pre conditions | User should have given access to the camera |
| Successful End conditions | QR code is scanned |
| Failed end conditions | QR code fails to scan |
| Primary Actors | User |
| Secondary Actors | None |
| Trigger | The user opens the camera to scan |
| Included cases | None |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | QR code button is pressed | | 2 | Picture of the QR code is taken from the user | | 3 | QR code will be processed | |

## 

|  |  |
| --- | --- |
| **Use case name** | **Image Processing** |
| Related Requirements | DM5 |
| Goal in context | Receipt is processed and converted into text |
| Pre conditions | Users should have a functioning camera; they should have given permission to the app to use the camera. |
| Successful End conditions | Grocery items from the receipt are converted to text and stored in the buying history database |
| Failed end conditions | Grocery items from the receipt are not converted to text or not stored in the buying history database |
| Primary Actors | Existing User |
| Secondary Actors | none |
| Trigger | User opens the camera to scan the receipt |
| Included cases | none |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | User opens the camera | | 2 | User takes picture of the receipt | | 3 | The receipt is converted to text data through image processing | | 4 | Processed data is stored in the buying history database | |

|  |  |
| --- | --- |
| **Use case name** | **Manual Feed** |
| Related Requirements | DM1, DM2 |
| Goal in context | User wants to enter the grocery items bought in the database |
| Pre conditions | The user has bought grocery and wants to enter data |
| Successful End conditions | Data is successfully entered in the database |
| Failed end conditions | Data is not entered in the database |
| Primary Actors | Existing User |
| Secondary Actors | none |
| Trigger | User has selected the option to manually feed data |
| Included cases | none |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | User chooses to feed data manually | | 2 | User types in the grocery items bought | | 3 | Grocery items are saved in the buying history database | |

## 

|  |  |
| --- | --- |
| **Use case name** | **Check Buying History** |
| Related Requirements | DM6, DM7 |
| Goal in context | User is able to see all previously purchased items |
| Pre conditions | Existing user |
| Successful End conditions | User can view all purchased item history |
| Failed end conditions | User can't view all purchased item history |
| Primary Actors | Existing user |
| Secondary Actors | none |
| Trigger | When the user chooses to view buying history. |
| Included cases | none |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | User chooses the option to check buying history | | 2 | Buying history is displayed to user | | 3 | User also has the option to delete buying history | |

## Help and Support

**4.4.1 Description and Priority**

This feature will be used when the user has difficulty understanding the application. He can either watch a tutorial guide on how to navigate through the application, rate the application or contact the admin for any particular queries. This is a medium priority feature.

**4.4.2 Stimulus/Response Sequences**

The user will have a help button at all times in the menu, which he can click whenever he wants help which will direct him to the help and support page. This will give him three options: watch a tutorial, rate the application or contact the admin and direct him to the respective page of the option he chooses.

**4.4.3 Functional Requirements**

* **HS1**:User should be able to view app tutorial.
* **HS2**:User can contact admin.
* **HS3**:User can check frequently asked questions.
* **HS4**:User can give their feedback regarding their app experience.
* **HS5**:Admin can check feedback, and reply to queries.

|  |  |
| --- | --- |
| **Use case name** | **Help and Support** |
| Related Requirements | HS1, HS2, HS3, HS4, HS5 |
| Goal in context | User can choose an option to give feedback, ask questions, |
| Pre conditions | User is using the app |
| Successful End conditions | User gets the required help |
| Failed end conditions | User wasn't able to get the required help |
| Primary Actors | User |
| Secondary Actors | Admin |
| Trigger | User clicks the help icon |
| Included cases | none |
| Main Flow | |  |  | | --- | --- | | **Step** | **Action** | | 1 | User chooses the help button from the main menu. | | 2 | User is displayed 3 options   * App tutorial * Give rating * Contact admin | | 3 | User chooses their desired option | | 4 | Admin checks ratings and replies to queries. | |

# Other Nonfunctional Requirements

## Performance Requirements

* The frames of the application should switch from one to another within 1 second.
* Respective buttons should only do what they’re supposed to do.
* Viewing shopping history should allow the user to filter the history according to the date of shopping and the filtering should happen within a second.
* The user should be given the option to export the final list in either pdf or docx format.

## Safety Requirements

* In case there is data loss from the servers, backup should be made on cloud storage that the company can use to retrieve user data (for future implementations).
* A confirmation prompt should be given to the user at the end of important decisions like deleting an account or a list.

## Security Requirements

* User credentials for an already existing user will be checked from our database, and if the credentials pass authentication only then a user can log in and continue using the app.
* The database should be protected with a strong password and only the admin has access to it.
* The application cannot use user data to study usage trends without their consent. Users should be able to turn this off at any time.
* The application cannot sell user data to third party applications
* The user must know the statement of the purpose for which the data is being collected.
* Where applicable, the user must be directed to a link to agree to specific terms and conditions associated with the stated purpose adhering to IEEE’s data collection, access and use policy.

## Software Quality Attributes

* **AVAILABILITY:** The application should be available 24x7 to provide predicted lists even when the internet is unavailable. Unavailability of the internet should only impact location features.
* **CORRECTNESS:** The predicted lists should be in accordance to the respective user’s shopping habits and not assigned randomly.
* **MAINTAINABILITY:** The administrator should maintain user data and accounts.
* **USABILITY:** The application should satisfy the maximum number of user needs and run on all android devices.

## Business Rules

* Allows admin to maintain the application and user data.
* Allow multiple users to use the application concurrently.
* Ensure user data protection
* Enable admin to reply to user queries and view their feedback to ensure customer satisfaction.

# Other Requirements

**Database requirements**

All database tables need to be coherent, and unified. Complete modelling needs to be observed and data redundancy needs to be avoided in all cases. For all Database interactions, the Database needs to be tested and running well. For that purpose, we plan on using a Network DBMS or a Relational Database for efficient query processing and avoid impedance mismatch.

**Internationalization requirements**

With all the places we are looking to operate, we will provide language support for the betterment of the people, the locality and nation as well as adhering to their preference and specific needs.

**Reuse requirements:**

The Application has been made such that it can accommodate both iOS and Android users. The application is friendly as well as maintainable in the long run. The database can be erased and used with any other software or application. Predictor algorithms can be used with a variety of different applications and the interface can be altered for any other application as per the client’s requirements.

**Legal Requirements:**

The app needs permission to store and use the client’s data. It needs the grocery store’s permission to show its catalog and name within the app. It further needs location permission from the user in order to access it. We need to make sure the database being used adheres to the conditions set by the database’s creator for e.g., Oracle. Furthermore, the app needs to make sure no data is shared amongst different members in a joint family account otherwise breach of data could trigger legal issues. Security of the application should be fool proof such that no data is leaked to advertising companies.

**Appendix A: Glossary**

1. **UML**: Unified Modelling Language
2. **SECP**: Securities and Exchange Commission of Pakistan
3. **APK**: Android Package
4. **SRS**: Software Requirement Specification
5. **FAQ**: Frequently Asked Questions
6. **Use** **cases**: A use case is a methodology used in system analysis to identify, clarify and organize system requirements
7. **Functional** **Requirements**: Defines a function of a system or its component , where a function is described as a specification of behavior between inputs and outputs
8. **IEEE**: Institute of Electrical and Electronics Engineers
9. **OTP**: One time passcode
10. **DBMS**: Data Base Management System- Software package that allows user to maintain, create, delete, and update databases.
11. **Database**: Organized collection of data stored and accessed electronically from a computer system
12. **Object** **Oriented** **Programing**: Object-oriented programming (OOP) is a computer programming model that organizes software design around data, or objects, rather than functions and logic.

**Appendix B: Analysis Models**

Will be furnished in the next revision .

**Appendix C: To Be Determined List**

* Section 3 will be completed in the next revision.
* The merchant feature(mentioned in section 2) will be incorporated after the client’s approval.