Software Requirements Specification

for

Grocery List Maker

Version 1.0 approved

Prepared by Group 2

Section 1

October 13th, 2021

Copyright © 1999 by Karl E. Wiegers. Permission is granted to use, modify, and distribute this document.

Table of Contents

Table of Contents ii
Revision History ii
1. Introduction 1
1.1 Purpose 1
1.2 Document Conventions 1
1.3 Intended Audience and Reading Suggestions 1
1.4 Product Scope 1
1.5 References 1
2. Overall Description 2
2.1 Product Perspective 2
2.2 Product Functions 2
2.3 User Classes and Characteristics 2
2.4 Operating Environment 2
2.5 Design and Implementation Constraints 3
2.6 User Documentation 3
2.7 Assumptions and Dependencies 3
3. External Interface Requirements 3
3.1 User Interfaces 3
3.2 Hardware Interfaces 3
3.3 Software Interfaces 3
3.4 Communications Interfaces 3
4. System Features 4
4.1 System Feature 1 4
4.2 System Feature 2 (and so on) 4
5. Other Nonfunctional Requirements 4
5.1 Performance Requirements 4
5.2 Safety Requirements 5
5.3 Security Requirements 5
5.4 Software Quality Attributes 5
5.5 Business Rules 5
6. Other Requirements 5
Appendix A: Glossary 5
Appendix B: Analysis Models 5
Appendix C: To Be Determined List 6

Revision History

Name	Date	Reason For Changes	Version
Group 8	October 12, 2021	First Draft	1.0

1. Introduction

This section of the Software Requirements Specification (SRS) provides an overview of the entire SRS with purpose, scope, definitions, acronyms, abbreviations and references of the SRS. The aim of this document is to gather, analyze and give an in-depth insight of the complete Grocery List Maker software system by defining the problem statement in detail.

1.1 Purpose

This document describes the features, interface, underlying system, and constraints for software release 1.0 of the Grocery List Maker Application. The purpose of the document is to establish the complete functional and quality requirements of the software application. It also describes non-functional requirements and other factors necessary to provide a complete and comprehensive description of the requirements for the software.

1.2 Document Conventions

This Document was created based on the IEEE template for System Requirement Specification Documents. The term ‘Grocery List Maker’ and ‘Grocery List Application’ are used interchangeably throughout the document.

1.3 Intended Audience and Reading Suggestions

This document is primarily intended to be used by both the developers and stakeholders of the system and is used to outline the system.

1.4 Product Scope

The Grocery List Maker application will be an Android application which will predict the grocery items a user would be willing to buy based on previously collected data. The user will feed in data about daily purchases which the application will keep track of. Based on these purchases, this app will notify the users what to buy, when to buy and from where to buy that particular item.

The user will be able to choose a grocery list that saves money or a list that helps him buy stuff conveniently. It will also mark the best route to make the purchases. Unless otherwise noted, all requirements specified here are committed for release 1.0.

1.5 References

[1] IEEE Software Engineering Standards Committee, “IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications”, October 20, 1998.

2. Overall Description

2.1 Product Perspective

Grocery List maker is a totally independent application which is neither related to any other system nor a component of a larger system. This application has only one type of user, and thus there are no functionality differences between users. Thus, it has one type of user interface.

The objective of the project is to provide an efficient grocery management whose main functionality is predicting the products for the next grocery purchase, also suggesting the best price for the grocery items based on the user's buying habits.
The product also aims to learn to keep track of the grocery items. The app learns the interval between purchases of a product to learn when a product gets finished. The app aims to notify the user to buy a product before it gets finished.

2.2 Product Functions

- Ability of the application to sync data across devices
- Ability of the application to scan the barcodes of products and redirect the user to the information pages pertaining to the scanner
- Ability of the application to add consumer products and compare prices
- Ability of the application to predict buying patterns
- Ability of the application to show the best route to make the purchases
- Ability of the application to have an updated database
- Ability of the application to allow signup using third party social networks
- Ability of the application to recover account loss

2.3 User Classes and Characteristics

The user class is that of a regular smartphone. The app is designed to be user friendly. No technical expertise is required to use the app. The user is also expected to understand English as the application will be shipped in the English Language to ensure that a large portion of the global audience is able to use the app.

2.4 Operating Environment

The hardware platform is that of a regular android phone. Although the version of the operating system cannot be specified, it is confirmed that you will need at least Android 5.0 or higher to run our product. We do not offer our application on other platforms.

On the hardware side, your phone should have internet connectivity to fetch data and location services enabled to provide accurate geographical data.

There may be google maps integration.

Minimum System Requirements as follows :

- At least 64MB of free space
- Android 5.0 or higher
- 2GB RAM or Higher
- An Active Internet Connection

2.5 Design and Implementation Constraints

As the app is lightweight, it will not be resource-hungry. As mentioned earlier you may need an internet connection and location services enabled to get accurate geographical data from our app.

As the app only employs a native login system and the purpose of the app is only to manage the user's grocery, therefore there are not many security considerations that need to be accounted for. Corporate policies disallow the reverse engineering of the app and public access to our databases.

2.6 User Documentation

- A User Manual available in PDF.
- Youtube Shorts in both English and Urdu to help install, navigate and use the app.
- An in-app help page which mainly answers FAQs.

2.7 Assumptions and Dependencies

- A Nutrition tab for the consumer product may be integrated
- A barcode scanner will be integrated assuming there are resources to spare.
- Signup through third party social media accounts will be integrated assuming there are resources to spare.
- Memory may be a constraint if a user puts in an enormous amount of grocery lists.
- Devices may lag if one list contains an enormous amount of consumer products.
- A currency converter may be integrated for users seeking the convenience.
- The application may support other languages assuming there is community support.

3. External Interface Requirements

3.1 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.>

3.2 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.>

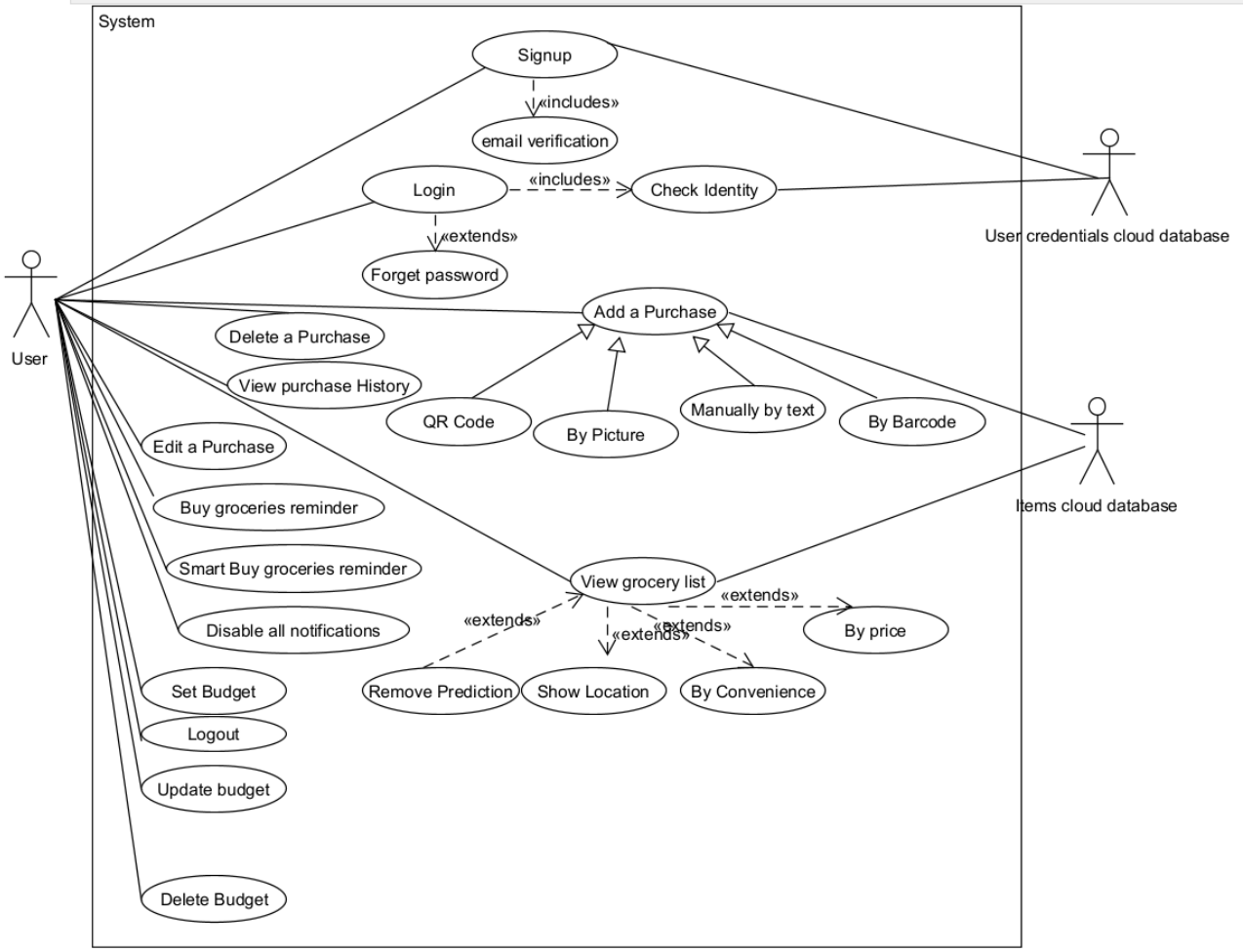
3.3 Software Interfaces

<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.>

3.4 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.>

4. System Features



4.1 System Feature 1

Creating and Accessing Accounts

4.1.1 Description and Priority

This feature allows the users to create and handle accounts as per their requirements. It includes signing up, verifying the email, logging in, checking the identity, resetting the password, and logging out options respectfully.

This is definitely a “High” priority feature due to several reasons. We cannot access the main features of the app if we do not create an account using the sign-up process. The login process is extremely important to

protect your account from unauthorized access. This prevents intruders from reading and manipulating your important information. The log out process is also important for data protection in the case some other person gets hold of your device. The forgot password option is critical as this is a human error which cannot be completely prevented.

4.1.2 Stimulus/Response Sequences

When the user clicks on the grocery list maker's icon, it will launch the mobile application. Firstly, a welcome page will be displayed on the user's screen. Then the user will get two options in the form of two buttons. The first button will read "Sign Up". This option is for new users who currently do not have an account on this app. Clicking on this button will take the users to another screen where they will be given three further choices.

The first button would contain the title "Continue with Facebook" . This will connect your app's account to your Facebook account, and it will obtain necessary details required from your Facebook profile. However, this app would not post anything on your behalf to Facebook without your approval and would keep your details private. The connection will be made when you enter your Facebook email id and password. This feature is very important as social media has taken over the world of communication and most of the people already have accounts on major social media platforms like Facebook. This will make the signing up process easier and faster for them. The second button will read "Continue with google". The option will connect your app with your google/Gmail account. The third option will be to "Continue with email". This choice will be for those people who either do not have a Facebook/google account or do not want to use it to connect with this app. They will be required to enter an active email id and create a password for this app. They may also be asked to enter a few additional details such as name or date of birth.

The users will have to verify their accounts by clicking on a verification link which will be emailed to them. Multiple sign-up options are given in order to facilitate a wide array of users who have different preferences. Several verification and validation checks are applied to different fields and would generate an error message if inconsistent data is entered. The Second option on the welcome page was to log in. This option enables existing users to access their accounts. The log in option also contains sub options of logging in using Facebook, logging in using google account, or entering the account credentials that you have previously created for this app. The system will verify your entered credentials against the User Credentials Database and then allow access.

Furthermore, there will be a "Forgot Password" option for users to reclaim their account in case they have forgotten their password . They will have to enter the answer for the security question that they were asked during the sign-up process. There will also be a log out option available at different stages of the application which will enable the user to click on it whenever they wish to leave the app.

4.1.3 Functional Requirements

Fr 1.1 Application should display a sign-up button on the welcome page.

Fr 1.2 Application must display three other buttons when the user clicks on the sign-up button.

Fr 1.3 The first button must allow users to sign-up using their google account.

Fr 1.4 The second button must allow users to sign-up using their Facebook account.

Fr 1.5 The third button must open up a form and allow users to sign-up using any email account.

Fr 1.6 The verification link must be sent to the user's email and access should only be provided after the user clicks on it.

Fr 1.7 Application should also display a login button on the welcome page.

Fr 1.8 The login button should lead to three other buttons when clicked.

Fr 1.9 The first button must allow users to log in using their google account.

Fr 1.10 The second button must allow users to log in using their Facebook account.

Fr 1.11 The third button must allow users to log in using their email account.

Fr 1.12 The system must compare the user entered credentials with the stored credentials and only give access if they match.

Fr 1.13 The system should also give a forgot password option.

Fr 1.14 The system should ask the user the security question in case he chooses the forgot password option.

Fr 1.15 The system should compare this answer with the one previously stored and only give users an option to set a new password if they match.

Fr 1.16 System should also give the user a log out option at multiple instances.

Fr 1.17 The system should save the user’s progress and log them out of their account when they click on the log out button.

Use case name	Signing up for the mobile application of grocery list maker.	
Related Requirements	Fr 1.1 , 1.2, 1.3, 1.4, 1.5	
Goal In Context	To register a new user in the app	
Preconditions	The user has installed and launched the application.	
Successful end condition	The user is registered, and his login credentials are stored	
Failed end condition	The user registration is rejected	
Primary Actor	User	
Secondary Actor	User Credentials Database	
Trigger	User asks the program to register him	
Included Cases	Email Verification	
Main flow	Step	Action
	1	The user asks to sign up
	2	The user enters his/her details
	3	The user's email id is verified using the “Email Verification” use case.
	4	Users are registered upon successful verification and their credentials are stored.

	5	Users are shown a prompt upon successful registration.

Use Case Name	Verifying the email account which the user has used to register themselves.	
Related Requirements	Fr 1.6	
Goal in context	To make sure that the email account belongs to the user who is trying to use it for registration.	
Preconditions	User has launched the app and filled up the sign-up form.	
Successful end condition	The email is verified, and the system is given a go ahead for account creation.	
Failed end condition	The verification fails and no account is created.	
Primary Actor	User	
Secondary Actor	None	
Trigger	User enters details in the sign-up form and asks to proceed.	
Included Cases	None	
Main flow	Step	Action
	1	User enters their email address.

	2	A verification link is sent to that email id.
	3	Users must log in to that email account and click on the link within a certain period of time.
	4	Account verified and access given as soon as the link is clicked.

Use Case Name	Logging into your user account.
Related Requirements	Fr 1.7, 1.8, 1.9, 1.10, 1.11
Goal in context	To uniquely identify the users and protect the data from unauthorized access.
Preconditions	The user must be registered
Successful end condition	The user successfully logs in
Failed end condition	The user’s login attempt fails
Primary Actor	User
Secondary Actor	User Credentials Database
Trigger	User requests to log in

Included Cases	Check Identity	
Main flow	Step	Action
	1	User requests to login
	2	The user enters his credentials
	3	Credentials are checked using Check Identity use case
	4	Access given upon successful authentication
	5	User is shown a prompt upon successful log in.

Use Case Name	Checking the identity of the user.
Related Requirements	Fr 1.12
Goal in context	Check the credentials entered by the user against prestored values to verify them.
Preconditions	The user must be registered.
Successful end condition	The entered credentials match the stored ones, and the user is authorized.

Failed end condition	The entered credentials fail to match with the original ones and access is denied.	
Primary Actor	User	
Secondary Actor	User Credentials Database	
Trigger	User enters detail in the log in form and asks to proceed.	
Included Cases	None	
Main flow	Step	Action
	1	User opens the login form.
	2	The user enters their credentials and asks to proceed.
	3	Entered credentials are compared with the accurate ones which are previously stored in the system.
	4	Access given upon successful authentication .

Use Case Name	Changing the password in case a certain user forgets it.	
Related Requirements	Fr 1.13, 1.14, 1.15	
Goal in context	To replace the previous password which the user forgot with a new one.	
Preconditions	The user is logged in.	
Successful end condition	The password is changed	
Failed end condition	The request for password change is rejected	
Primary Actor	User	
Secondary Actor	User credentials database	
Trigger	The user requests to use the forgot password option.	
Included Cases	Check Identity	
Main flow	Step	Action
	1	The user clicks on the forgot password button.
	2	The user is asked to answer the security question.
	3	This answer is compared with the stored one which the user entered at the time of account creation.
	4	On successful comparison, the user is asked to enter a new password and it is saved to the system.

	5	Users are given access to use the account.

Use Case Name	Logging out of the user account.	
Related Requirements	Fr 1.16, 1.17	
Goal in context	This use case allows an account holder to log out of his account if he wants to.	
Preconditions	The user must have an account and he must already be logged into it at that time.	
Successful end condition	The user is successfully logged out of their account.	
Failed end condition	The user is still logged into their account.	
Primary Actor	User	
Secondary Actor	None	
Trigger	The user asks the system to log them out.	
Included Cases	None	
Main flow	Step	Action
	1	User asks to logout.

	2	Their progress during this session is saved.
	3	The information/features previously displayed on the screen are hidden
	4	The initial Sign Up/ Login page of the application is displayed.

4.2 System Feature 2

Managing the Grocery List

4.2.1 Description and Priority

This feature defines the main purpose of the grocery list maker application. It basically deals with accessing and manipulating the predicted grocery lists. This guides the users regarding which items they should buy and from where they should buy it. This further allows the user to view the stores with the lowest cost of items and the stores which are the closest to their location. The user is also given the privilege to remove items from future predictions, following a change in their lifestyle.

This is a “High” priority feature as it performs the task which the name of our project “Grocery List Maker” requires doing. Our app does not fulfill its integral goal if it does not predict grocery lists as this is the operation for which the user will download our app. It is crucial to point out the required items and best possible stores to buy them from.

4.2.2 Stimulus/Response Sequences

When a user wishes to manage the grocery list, they are provided with a series of options with each corresponding to a different feature. The main tool is the “View Grocery List” tool which creates a list of several items that a user is suggested to buy at that time. It’s working is based on the user’s buying habits. This applies the “By Convenience” and “Saves Money” methods in order to produce the best path in turn to buy all the items present on the list.

The first process helps the users by applying the principle of convenience . This is accessed by pressing the button which reads “By Convenience” . This will display the details of the store which has the shortest distance from the user’s location. This option is integrated with google maps and compares the routes to reach different stores which supply this item and selects the one with the best route.

The second button is of the “Saves Money” option which applies the principle of cost effectiveness. This compares the price that a certain item has in different stores and releases the details of the store which provide the lowest rates. The user can also obtain the maps containing the directions to stores from where they would want to purchase items.

The user also has the authority to prevent a certain item from appearing in future predictions. This is a case where a user does not want to continue buying a certain item. For example, a user has quitted alcohol consumption, or another user has stopped the use of a certain medicine. They could do this by using the remove item option.

4.2.3 Functional Requirements

Fr 2.1 The application must display a view grocery list button.

Fr 2.2 The application must further display two different filter options.

Fr 2.3 Users can select the “By Convenience” tool to view stores with the shortest distance from their location.

Fr 2.4 Users can select the “Saves Money” tool to view the stores which provide the least cost for items.

Fr 2.5 The System should also provide the “Show Location” option.

Fr 2.6 The system should display the maps on the user's screen which lead to the concerned stores when this option is selected.

Fr 2.7 The system must allow the users to remove unwanted items from the predicted list using the “Remove Items” feature.

Fr 2.8 The system must also automatically generate the grocery list after the timer set by the user expires.

Use Case Name	Viewing the grocery list predicted by the application.

Related Requirements	Fr 2.1, 2.2, 2.8	
Goal in context	Show users the predicted grocery list based on their buying habits from the past and applying different cost and distance related filters.	
Preconditions	The predicted grocery list exists, and the user is logged in	
Successful end condition	The prediction grocery list is shown	
Failed end condition	The predicted grocery list is not shown	
Primary Actor	User	
Secondary Actor	Items database	
Trigger	1)The user asks the application to display the new grocery list based on current circumstances. 2) An automatic timer is set by the system which creates a grocery list after every two weeks.	
Extended Cases	1) Remove Prediction 2) Saves Money 3) By convenience 4) Show location	
	Log in	
Main flow	Step	Action
	1	The user requests to view the prediction.

	2	The predicted list is fetched .
	3	The predicted list is shown to the user.
	4	The user can scroll through the list and view required items.

Use Case Name	Predicting the most convenient list(Distance Wise).
Related Requirements	Fr 2.3
Goal in context	Show the shortest route prediction list.
Preconditions	The predicted list exists, and the user is logged in
Successful end condition	The shortest route prediction list is shown.
Failed end condition	The program fails to show the shortest route prediction list
Primary Actor	User
Secondary Actor	Google Map
Trigger	The user asks to view the predicted list by convenience
Included Cases	None

Main flow	Step	Action
	1	The user asks to view predicted list by convenience
	2	The predicted list is fetched from Create Prediction and shortest route predictions are shortlisted
	3	The list is shown to the user

Use Case Name	Predicting the most economically efficient list(Cost Wise).
Related Requirements	Fr 2.4
Goal in context	Show the lowest price prediction list
Preconditions	The predicted list exists, and the user is logged in
Successful end condition	The lowest price prediction list is shown
Failed end condition	The program fails to show the lowest price prediction list
Primary Actor	User
Secondary Actor	Items Database
Trigger	The user asks to view the predicted list which saves money

Included Cases	None	
Main flow	Step	Action
	1	The user asks to view the predicted list by convenience.
	2	The predicted list is fetched from Create Prediction and lowest price predictions are shortlisted.
	3	The list is shown to the user.

Use Case Name	Obtaining the locations of suitable grocery marts.
Related Requirements	Fr 2.5, 2.6
Goal in context	To get the locations of the stores from where they should buy the needed products.
Preconditions	The user is logged in and a predicted grocery list exists.
Successful end condition	A map leading to the location is shown to the customer.
Failed end condition	The location is not displayed on the customer’s screen.

Primary Actor	User	
Secondary Actor	Google Maps	
Trigger	The user asks the system to show them the location.	
Included Cases	None	
Main flow	Step	Action
	1	The user requests the system for the location.
	2	The system fetches the details related to the store’s location from the cloud.
	3	The system displays the map leading to the location on the user's screen.
	4	The user can hover over the map however they want to and can then use the start option to activate it when they begin their journey.

Use Case Name	Removing an unwanted item from the predicted list.	
Related Requirements	Fr 2.7	
Goal in context	To remove a product from predicted grocery list	
Preconditions	The user is logged in and a predicted grocery list exists	
Successful end condition	The prediction is removed	
Failed end condition	The prediction is not removed	
Primary Actor	User	
Secondary Actor	Items database	
Trigger	The user asks to remove a prediction	
Included Cases	None	
Main flow	Step	Action

	1	The user asks to remove a prediction
	2	The user selects the prediction to remove
	3	The prediction is removed
	4	The user is shown a success prompt

4.3 System Feature 3

Managing the Cloud

4.3.1 Description and Priority

Our application is linked with cloud computing which enables us to get on demand access to information technology resources over the internet. Examples of cloud platforms include i cloud, one drive, google cloud, drop box. This is rated as a medium priority feature as we already have an alternate option of hosting servers available. Using the cloud provides a lot of advantages as mentioned above but it is not something which is compulsory for the deployment of every application. It also has some disadvantages such as limited control, internet issues, and security issues.

4.3.2 Stimulus/Response Sequences

The main purpose of a cloud is to provide easy, efficient, and scalable access to services. The alternate option of hosting brings along several issues related to buying, maintaining, and fixing your own server. By using the cloud, our business can instead focus more on other things such as facilitating users. The cloud we are connected to also provides a high level of security to safeguard data related to our company and customers. Our customers can access their share of data stored in the cloud from any place around the world where the internet is available. The data can be backed up and restored in case of problem occurrence. The data can be stored, accessed, and maintained at extremely low costs . The pay per use service means that we only have to pay for the number of services which we have used during a particular time instead of high fixed costs.

Our company has stored various kinds of information on the cloud. This contains the personal details of users including their login credentials for verification. This also contains the data related to all the items purchased by all the users. Along with this, the details about the stores and the routes to those stores are also saved. The company also stores all the previously predicted grocery lists on the cloud which help in the future analysis. Not only is our data stored on the cloud, but the development, running and management of our application also takes place on the cloud.

4.3.3 Functional Requirements

Fr 3.1 The system must provide purchase history related data to the user from the cloud if they request for it.

Fr 3.2 The system must save the edits made by the user to purchase history on the cloud.

Fr 3.3 The system should generate and provide grocery related data to the user from the cloud if they request for it.

Fr 3.4 The system must delete the removed grocery list items from the cloud.

Fr 3.5 The system must obtain the mapping details stored in the cloud and display it to the user.

Fr 3.6 The system must verify user credentials from the ones stored in the cloud whenever a user requests for access.

Fr 3.7 The application should be running on the cloud and should allow the admin to manage itself using the cloud.

Fr 3.8 The system should keep a backup of all the data stored in the cloud.

FR 3.9 The system should restore all the data after a hardware or software problem occurs.

Use Case Name	Using the cloud's services for data related operations.
Related Requirements	Fr 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9
Goal in context	Getting access to services and resources provided by the cloud.
Preconditions	Data is stored in the cloud and the users have access to their share of data.
Successful end condition	The data is displayed on the user's screen.
Failed end condition	The system fails to give the user the access to cloud data.
Primary Actor	Cloud

Secondary Actor	User	
Trigger	User requests to get the access of data which is stored inside the cloud.	
Included Cases	1)Log in 2)View Purchase History 3)View grocery list	
Main flow	Step	Action
	1	The user requests a system to show them the data related to purchase history or grocery lists.
	2	Data is fetched by the system from the cloud.
	3	The user can scroll through the data and make changes to it as per the requirement.
	4	The system updates the clouds and saves the changes made by the user.
	5	Users are shown a prompt about successful changes.

4.4 Managing Purchases

4.4.1 Description and Priority

This feature enables the user to add new purchases, view purchases, edit existing purchases and delete previous purchases. Having a record of purchases is necessary for the app to achieve its purpose of predicting a new grocery list. Adding and Deleting a purchase have high priority

because the app needs these to have an accurate record of purchases. Viewing purchases also has a high priority because the user needs to view purchases in order to delete them.

Editing a purchase has low priority because its effect can also be achieved by deleting the wrong purchase and then adding the correct purchase.

4.4.2 Stimulus/Response Sequences

ADDING NEW PURCHASE

There will be a button to add new purchases. After pressing the button, the user will be given two options, Single Product and Multiple Products. There is only one way to add purchase of a Single Product, Manual way.

If the Multiple Products option is selected then the user will be given four ways to add the purchase namely, Manual, QR Code, Barcode and Picture. The Multiple Products option should be chosen if the products have been purchased on the same day and from the same shop. If the products have been purchased from different shops, then the purchases must be added separately.

The Multiple Products option is only for ease of use while adding purchases. Each product would still be added as a separate purchase. A purchase record can have only one product, albeit in any quantity.

SINGLE PRODUCT

- If the user chooses a Single Product, then he will be shown a form to enter information about the purchase. The form will require Date of Purchase, Product name, Quantity, Price, Shop Name, Shop Location and Total amount.

- The Date of Purchase field will have the current date as the default value. If the user wishes to change the date, he can touch the date field upon which the user will be shown a calendar view to select a date. The user will not be able to select a date which has not yet arrived. After selecting the date, the user will press OK. The calendar would no longer be visible and the form will be shown again now with the selected date.

- When the user will start typing the Product Name, he will be shown predictions using the previous purchase records. If one of the records is selected, then all the empty fields will also be copied from that record. The user would be able to change the values of the auto filled fields if needed.

- When the user starts typing the Shop Name, the app will try to predict the name from previous records and data from Google Maps. If a predicted name is selected, then the Shop Location field would also be copied from the selected record.

- When the user attempts to add the Shop Location, he will be shown a map with a search bar at the top. There will be a back button on the top left to go back to the Add a Purchase form without adding the Shop Location. The user can search the name of the shop and select the correct shop from the search results. Another way is to drop a pin at the correct location. After selecting a location, the user will press the Confirm Location button, the map would no longer be visible. The form would be shown again now with the Shop Location added.

- Price, Quantity and Total Amount fields will only show numerical input. The values of these three fields are related. Total Amount will have the mathematical product of Price and Quantity as its value. By default, quantity will have the value '1'. If price is changed then the Total Amount would also change automatically and vice versa.

- After filling all the fields, the user would press the Add Purchase button, and the purchase would be added to the records. If any of the fields are empty, the user will be asked to fill them before adding the purchase.

MULTIPLE PRODUCTS

MANUAL METHOD

- If the user has chosen to manually add purchase of multiple products, then the flow would be similar to manually adding purchase of a single product. There would be some differences in the form displayed.

- The four fields, Product Name, Price, Quantity and Amount will represent the information of a single product. The field's behavior shall be the same as described in the Single Product case.

- There will be a button with a 'plus' on it below the four fields. On pressing this button, a new instance of the four fields would appear to represent the information of the next product.

- The Date, Shop Name and Shop Location fields will only appear once in the form. These fields represent the information common between all the purchased products.

- When the user presses the Add Purchase button, each product would be added as an individual purchase to the records.

- If any of the fields were empty, then the purchases would not be added and the user would be notified which fields still need to be filled.

QR CODE/BARCODE METHOD

- If the user chooses the QR code/Barcode method then a QR Code/Barcode scanner will open.
- After the QR Code/Barcode is scanned successfully, the same form as in the previous case will be shown but this time the form would be filled automatically using the information obtained from the QR Code/Barcode.
- The user will be requested to verify that the information filled is correct and make changes if required.
- When the user presses the Add Purchase button, the purchase would be added to the records.

PICTURE METHOD

- If the user chooses the Picture method, the camera would open.
- After taking a picture of the receipt, OCR will be used to extract relevant information to fill the fields.
- If the OCR fails to extract information from the picture, the user would be notified and given the option to take a new picture or cancel.
- If the OCR works successfully then the form with filled information would be shown.
- The user will be requested to verify that the information filled is correct and make changes if required.
- When the user presses the Add Purchase button, the purchase would be added to the records.

VIEWING PURCHASES

- The purchases shall be shown using a scrollable list.
- Each record in the list shall represent a purchase. Each record shall show the Date of purchase, Product Name, Price and Shop Name.
- A purchase can be opened by touching on it. When the purchase is opened, a form same as in the Adding Single Product case is shown. From this form the user can view all the available information of the purchase.
- A purchase can be selected by touching and holding on it. After a purchase has been selected, touching on other purchases selects multiple purchases.
- All the selected purchases can be deselected by touching a back button on the top left corner.

EDITING A PURCHASE

There are two ways to edit a purchase. In the first way a single purchase can be edited at one time. In the second way, multiple purchases can be edited at once. In this method, only the three fields, Date, Shop Name and Shop Location can be edited as these three fields are more frequently common among different purchases.

EDITING SINGLE PURCHASE

- A purchase can be edited by opening a purchase from the Viewing Purchases screen.
- When the purchase is opened, a form same as in the Adding Purchase of Single Product case is shown.
- The fields shown are editable. So, the purchase can be edited by selecting a field and modifying its value.
- When the user has made all the required changes, he can press the Save Changes button which will show a confirmation dialog and proceed accordingly.
- If the user has made some changes and presses the back button without saving them, then he will be asked if wants to save the changes or discard them. The app will proceed according to the option selected.

EDITING MULTIPLE PURCHASES

- The second way is by selecting at least one purchase from the Viewing Purchases screen.
- If purchases are selected, there will be a calendar and a shop icon shown at the bottom.

- By pressing the calendar, a date picker will appear. The selected date will be the new date of purchase of the selected purchases.
- Upon pressing the shop icon, a form will appear with two fields, Shop Name and Shop Location. The values in these fields will be filled in the way described in the Add Purchases section.
- After filling the fields, the user will press Save Changes, a confirmation dialog shall appear.
- If the user confirms the changes, then the Shop Name and Shop Location of all the selected purchases shall be updated to the new values.

DELETING PURCHASES

Purchases can be deleted in two ways.

- In the first way, a purchase has to be opened first. When the purchase has been opened, on the screen there will be a trash can button. When this button is pressed, a confirmation dialog will appear. If the user cancels the operation, then the purchase would not be deleted. If the user confirms the operation, then the purchase would be deleted.
- The second way is by selecting at least one purchase from the Viewing Purchases screen. There would be a trash can icon in the bottom. When the trashcan is pressed, a confirmation dialog appears. If the user presses No, then the selected purchases would not be deleted. If the user confirms the operation, then the selected purchases would be deleted.

4.4.3 Functional Requirements

FR 4.1 There will be a button to add new purchases.

FR 4.2 When the add purchases button is pressed, two options are shown to the user, Single Product and Multiple Products

FR 4.3 There will be a form with relevant fields to add purchase of a single product.

FR 4.4 There will be a date picker to select the date.

FR 4.5 There will be a prediction system to predict the product name from previous purchases.

FR 4.6 There will be a map for the user to select the Shop Location.

FR 4.7 There will be a form with relevant fields to add purchase of multiple products.

FR 4.8 There will be a QR code/Barcode scanner.

FR 4.9 There will be integration with the camera.

FR 4.10 There will be Optical Character Recognition.

FR 4.11 If any fields are empty when the add purchase button is pressed, the purchase will not be added. The user will be asked to fill the empty fields.

FR 4.12 There will be a scrollable list.

Use Case Name	Adding Purchase by scanning the QR code.	
Related Requirements	FR 4.4 - FR 4.8	
Goal in context	To add product details by scanning QR code	
Preconditions	The user is logged in	
Successful end condition	The purchase is added	
Failed end condition	The purchase is not added	
Primary Actor	User	
Secondary Actor	Items Database	
Trigger	The user requests to scan a QR code	
Included Cases	Log in	
Main flow	Step	Action
	1	The user requests to scan a QR code

	2	The QR code is scanned
	3	The details are fetched
	4	The details are saved
	5	User is shown a prompt about successful addition

Use Case Name	Adding Purchase by taking images of the receipt
Related Requirements	FR 4.4 - FR 4.7, FR 4.9
Goal in context	To add a purchase by picture
Preconditions	The user is logged in
Successful end condition	The purchase is added
Failed end condition	The purchase is not added
Primary Actor	User
Secondary Actor	Items database
Trigger	The user asks to add a picture
Included Cases	Log in

Main flow	Step	Action
	1	The user asks to add a picture
	2	The user submits the picture
	3	The image is scanned and converted to text
	4	The text is organized and saved
	5	User is shown a prompt about successful addition

Use Case Name	Adding purchase data manually by typing
Related Requirements	The user will enter the purchase details by text to add a purchase
Goal in context	To add a purchase by text
Preconditions	The user is logged in
Successful end condition	The purchase is added
Failed end condition	The purchase is not added
Primary Actor	User

Secondary Actor	Items database	
Trigger	The user asks to add a purchase by text	
Included Cases	Log in	
Main flow	Step	Action
	1	The user asks to add a purchase manually by text
	2	The user enters the details in text form
	3	The details are saved
	4	User is shown a prompt about successful addition

Use Case Name	Deleting a previously entered purchase.
Related Requirements	FR 4.12
Goal in context	To delete a previous purchase
Preconditions	The user is logged in and at least one purchase is already added

Successful end condition	The purchase is deleted	
Failed end condition	The purchase is not deleted	
Primary Actor	User	
Secondary Actor	Items database.	
Trigger	The user asks to delete a purchase	
Included Cases	Log in	
Main flow	Step	Action
	1	The user asks to delete a purchase
	2	The user is shown a list of all purchases
	3	The user selects a purchase to delete
	4	The purchase is deleted
	5	The user is shown a prompt about successful deletion

Use Case Name	Editing a purchase.	
Related Requirements	FR 4.3 - FR 4.6, FR 4.12	
Goal in context	To edit details of the purchase	
Preconditions	The user is logged in and at least one purchase is already added	
Successful end condition	The details are modified	
Failed end condition	The changes are not made	
Primary Actor	User	
Secondary Actor	Items database	
Trigger	The user asks to edit a purchase	
Included Cases	Log in	
Main flow	Step	Action
	1	The user asks to edit a purchase
	2	The user is shown a list of all purchases
	3	The user selects a purchase to edit
	4	The user is shown the table of detail where he makes changes

	5	The user clicks submit
	6	The details are modified
	7	User is shown a prompt about successful modification

Use Case Name	Removing an unwanted item from the predicted list.	
Related Requirements	The user can remove a prediction if he no longer uses that product. This product will be removed from current list of prediction and will not be added to the prediction in future	
Goal in context	To remove a product from predicted grocery list	
Preconditions	The user is logged in and a predicted grocery list exists	
Successful end condition	The prediction is removed	
Failed end condition	The prediction is not removed	
Primary Actor	User	
Secondary Actor	Items database	
Trigger	The user asks to remove a prediction	
Included Cases	Log in	
Main flow	Step	Action
	1	The user asks to remove a prediction
	2	The user selects the prediction to remove
	3	The prediction is removed
	4	The user is shown a success prompt

4.5 System Feature: Managing notifications

4.5.1 Description and Priority

This feature enables the user to manage the settings of notifications. It enables the user to personalize notification settings according to his preferences. This feature has a low priority because the app can still achieve its purpose without this feature.

The app sends only two types of notifications, Add Purchases Reminder and Buy Groceries Reminder. There are only two types of notifications so that the user does not get annoyed by frequent notifications. Both types of notifications have a Smart Reminder option. If this option is enabled, the app automatically decides the best day and time to send the notification. Otherwise, the user has to specify when he wants to receive the notification.

The Add Purchases Reminder reminds the user to add new purchases to the app in case the user might have forgotten to do so. The Add Purchases Reminder will be especially useful for new users. A new user can easily forget to add purchases to the app because he is not used to using the app. This notification will help the user to develop a habit of adding purchases to the app. This is extremely important because if the user does not add purchases to the app, the app would not be able to learn the shopping habits and thus would not be able to make useful predictions. If the app would not make useful predictions, the user may find the app useless and even uninstall the application. So, this notification plays a very important role in the application.

The Buy Groceries Reminder reminds the user to purchase groceries. This notification is important so that the user purchases groceries timely and does not run out of any essential items.

By default, The Add Purchases Reminder is enabled and its Smart Reminder option is also enabled. The Smart Reminder option learns the shopping habits of the user using his Purchase history. It attempts to detect the case when the user might have made some purchases but forgot to add them in the app. If such a case is detected, the app sends a notification reminding the user to add purchases to the app. If the Smart Reminder option is disabled, the user will be requested to select the days and time on which he should be reminded to add purchases to the app. The user can choose to disable the Add Purchases Reminder.

The Buy Groceries Reminder is also enabled by default and its Smart Reminder option is also enabled. When Smart Reminder is enabled, the app will automatically learn the day on which the user does grocery shopping. The app will learn this using the purchase history of the user. The app will automatically decide the best day and time to remind the user to buy groceries. If the user chooses to disable the Smart Reminder, he will be requested to set a specific day and time on which the user will be reminded to buy groceries. The user can also choose to disable the Buy Groceries Reminder.

The user also has the option to disable all notifications.

4.5.2 Stimulus/Response Sequences

Add Purchases Reminder

The user will go to the notifications section of the app. The two types of notifications will be shown in a list. Each type of notification will have a toggle button in front of it.

The Add Purchases Reminder can be enabled and disabled by pressing the toggle button in front of it. If the Add Purchases Reminder is enabled, Smart Reminder will be written below it with its own toggle button. The Smart Reminder option can be enabled and disabled by pressing the toggle button.

When the user disables the Smart Reminder option, he will be requested to select the day and time on which he wants to be reminded. The selected day and time will be shown in the list below the smart reminder option.

Buy Groceries Reminder

The Buy Groceries Reminder can be enabled and disabled by pressing the toggle button in front of it. If the Buy Groceries Reminder is enabled, Smart Reminder will be written below it with its own toggle button. The Smart Reminder option can be enabled and disabled by pressing its toggle button.

When the user disables the Smart Reminder option, he will be requested to select the day and time on which he wants to be reminded. The selected day and time will be shown in the list below the smart reminder option.

Disable all notifications

On the notifications section of the app, there will be a button to disable all notifications. On pressing this button, the user will be shown a confirmation dialog box. All notifications would be disabled if the user confirms his actions. The settings would remain unchanged if the user cancels his action.

4.5.3 Functional Requirements

FR- 1.1 The user must be logged in to change notification settings

FR 1.2 There must be a toggle button to enable or disable ‘Buy Groceries reminder’ button

FR 1.3 If the user enables ‘Buy Groceries reminder’, a new form will open to ask the day or date on which to show reminder

FR 1.4 If the user disables ‘Buy Groceries reminder’ the reminder will be disabled

FR 1.5 There must be a toggle button to enable or disable ‘Smart Buy Groceries reminder’ button

FR 1.6 If the user enables ‘Smart Buy Groceries reminder’ the system will learn his buying pattern show reminder accordingly

FR 1.7 If the user disables ‘Smart Buy Groceries reminder’ the reminder will be disabled

FR 1.8 There must be a button to disable all reminders

FR 1.9 If the user disables all notifications, all types of notifications will be disabled

FR 1.10 There must be an Add purchases reminder button

FR 1.11 The user can enable or disable add purchases reminder by toggling that button

Use Case Name	Enable or disable Buy Grocery reminder notification
Related Requirements	FR 1.1, FR 1.2, FR 1.3, FR 1.4
Goal in context	To Enable or disable the reminder
Preconditions	The user is logged in
Successful end condition	The reminder notification setting is toggled
Failed end condition	The notification reminder setting is not toggled
Primary Actor	User

Secondary Actor	None	
Trigger	The user will ask to toggle reminder notification settings	
Included Cases	check identity	
Main flow	Step	Action
	1	The user asks to toggle notifications settings
	2	The user enables or disables the button
	3	If enabled, the user selects the time of the month when he wants to get the notification If disabled, the disabled setting will be saved
	4	The user is shown a success prompt

Use Case Name	Enable or disable Smart Buy Grocery reminder notification
Related Requirements	FR 1.1 FR 1.5 FR 1.6

	FR 1.7	
Goal in context	To Enable or disable the Smart reminder	
Preconditions	The user is logged in	
Successful end condition	The Smart reminder notification setting is toggled	
Failed end condition	The Smart reminder notification setting is not toggled	
Primary Actor	User	
Secondary Actor	None	
Trigger	The user will ask to toggle Smart reminder notification settings	
Included Cases	check identity	
Main flow	Step	Action
	1	The user asks to toggle notifications settings
	2	The user enables or disables the button
	3	If enabled, the system would show reminder using user's buying pattern If disabled, the disabled setting will be saved
	4	The user is shown a success prompt

Use Case Name	Enable or disable add purchases reminder notification	
Related Requirements	FR 1.1 FR 1.10 FR 1.11	
Goal in context	To Enable or disable the reminder	
Preconditions	The user is logged in	
Successful end condition	The reminder notification setting is toggled	
Failed end condition	The notification reminder setting is not toggled	
Primary Actor	User	
Secondary Actor	None	
Trigger	The user will ask to toggle reminder notification settings	
Included Cases	check identity	
Main flow	Step	Action

	1	The user asks to toggle notifications settings
	2	The user enables or disables the button
	3	If enabled the system will show purchase reminder periodically If disabled, the system will not show the reminder
	4	The user is shown a success prompt

Use Case Name	Disable All notifications
Related Requirements	FR 1.1 FR 1.8 FR 1.9
Goal in context	To disable all reminders
Preconditions	The user is logged in
Successful end condition	The reminder notification setting is disabled

Failed end condition	The notification reminder setting is not disabled	
Primary Actor	User	
Secondary Actor	None	
Trigger	The user will ask to disable reminder notifications	
Included Cases	check identity	
Main flow	Step	Action
	1	The user asks to disable notifications
	2	The setting is saved
	3	The user is shown a success prompt

4.6 System Feature: Managing the budget

4.6.1 Description and Priority

This feature will allow the user to get a predicted list according to his budget and if the total amount exceeds his budget he will get a warning notification. Similarly, as the user will add purchases, the system would record how much he has spent and show him the remaining budget the user is left with. To record this there would be a progress bar which will show how much of the budget has been spent.

The priority of this feature is low because it is not a necessary feature and the app would still perform its core functions even if this feature is not added.

4.6.2 Stimulus/Response Sequences

First, the user would set a budget. For which:

1. He'll request to set a budget.
2. Then he will enter the budget in the text field and click the set button
3. The system would save this budget and show the prompt about successful setup

Secondly, The user, after sometime might want to change the budget, either to lower or increase it. For this:

1. He would request to edit the budget
2. Then he'll enter the new budget in the text field and click on update button
3. The system would update the budget amount and show the prompt about successful updation

It is also possible that after some time the user no longer cares about the budget. To delete the budget he would:

1. The user would request to delete the budget
2. The budget would be deleted
3. The user will be shown a prompt about successful deletion

4.6.3 Functional Requirements

FR 1.1 There must exist a SET Budget button

FR 1.2 User would click the button to set the budget

FR 1.3 The system would show a form with a text field to enter budget amount

FR 1.4 The system would check the character type and then set the budget

FR 1.5 If the character type is not valid an error would popup

FR 1.6 There must exist a button to update the budget

FR 1.7 The user would ask to update the budget by clicking Update Budget button

FR 1.8 The system would show a form with a text field to enter the new budget amount

FR 1.9 The system would check the character type and then update the budget

FR 1.10 If the character type is not valid an error would popup showing updation error

FR 1.11 There must exist a button to delete the budget

FR 1.12 The user would ask to delete the budget by clicking Delete Budget button

FR 1.13 The system would show a confirmation prompt

FR 1.14 If the user confirms, the existing budget would be deleted

FR 1.15 The user will be shown a prompt about successful deletion

.

Use Case Name	Set a monthly budget for the prediction of grocery list

Related Requirements	FR 1.1 FR 1.2 FR 1.3 FR 1.4 FR 1.5	
Goal in context	To restrict prediction list from exceeding the budget	
Preconditions	The user is logged in.	
Successful end condition	The budget is set successfully	
Failed end condition	The budget is not set	
Primary Actor	User	
Secondary Actor	None	
Trigger	The user requests to set a budget.	
Included Cases	Log in	
Main flow	Step	Action
	1	The user requests to add a budget
	2	The user enters the budget
	3	The budget is saved

	4	User is shown a prompt about successful budget setup

Use Case Name	Update the monthly budget
Related Requirements	FR 1.6 FR 1.7 FR 1.8 FR 1.9 FR 1.10
Goal in context	To edit an existing budget
Preconditions	The user is logged in and there exists a budget
Successful end condition	The budget is edited successfully
Failed end condition	The budget is not edited
Primary Actor	User
Secondary Actor	None
Trigger	The user requests to edit a budget.

Included Cases	Log in	
Main flow	Step	Action
	1	The user requests to edit the budget
	2	He/she enters the new budget
	3	The new budget is saved
	4	User is shown a prompt about successful budget edition

Use Case Name	Delete the monthly budget
Related Requirements	FR 1.11 FR 1.12 FR 1.13 FR 1.14 FR 1.15
Goal in context	To delete an existing budget

Preconditions	The user is logged in and there exists a budget	
Successful end condition	The budget is deleted successfully	
Failed end condition	The budget is not deleted	
Primary Actor	User	
Secondary Actor	None	
Trigger	The user requests to delete a budget.	
Included Cases	Log in	
Main flow	Step	Action
	1	The user requests to delete the budget
	2	The budget is deleted
	3	User is shown a prompt about successful budget deletion

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Performance requirements determine the time utilized by a specific user command to run. It comprises of 3 main requirements which are listed along with their work as follows:

- Reliability:

Our software would be designed so that it performs well and consistently without failure. It would work efficiently without utilizing too much resources and the system would be scalable.

- Availability:

Our software would be optimized so that it takes the minimum amount of time to cater any user command without crashing.

- Maintainability:

Our software would rapidly respond to any command given to it and in case of a failure; it would restore without any hassle.

5.2 Safety Requirements

Safety requirements consist of the after effects of any damage or possible loss that can culminate because of the usage of the product itself. Our software would have the ability to work normally in case of any discrepancy that may be caused by any user command. We will have the following set of solutions when there is a safety hazard:

- The system would detect any anomaly or exception in the application.
- It would immediately notify the User to correct the command or warn the User of an impending error.
- Any sort of hazard would be dealt with by the system at all costs.
- Any attempt to invade the database system would lead the application to restart to prevent harm.
- In case of any accident, the data would be autosaved so that it would prevent any sort of data loss in the system.

5.3 Security Requirements

This requirement focuses on preventing any kind of data loss and provides protection against harmful viruses and unauthorized access. Our system would be properly encrypted and only admins would have access to the database system of our application which is tentative. Users will only be granted permission to view the grocery list to ensure protection of the data. Our system will have a definitive system separate for admins and users. Following points reinforce our security requirements:

- Secured database which would work in a predictable way.
- Specific constraints for User and Admin accessibilities.
- Software must ensure the integrity of the client's data
- Protection against external environments which could hamper progress.

5.4 Software Quality Attributes

Features that facilitate the measurement of performance of a software product by Software Testing professionals are known as Software Quality Attributes. These are unique for every software system depending on the kind of software you are creating. They help us to measure how well the software is performing and can help us to improve the performance of the software. Usability, reliability, functionality, portability, maintainability, and efficiency are those attributes of the application which can be used to measure its performance.

- The app will be easily downloaded and installed by the user on their smartphone.
- Every account holder on the app will be able to view ,create and edit their own grocery list and not that of any other account holder.
- The app will be flexible in the sense that changes in the code can be made to it easily after the software has been created.
- Only those people who download the app and have an account on it will be able to use the app.
- The app will be able to add grocery items and predict the grocery list for customers without having an internet connection. Only the location of the user will not be accessible without internet connection.
- Only the admin will be able to answer customer queries.
- Different users will not be able to interact with each other through the app.

5.5 Business Rules

A business rule is a constraint of the business itself that may guide system development. It is a rule that must be followed, no matter what else is happening. It often involves very specific criteria or conditions for compliance. These are intended to assert the business structure and describe the rules, operations and constraints that apply to a particular business. These will be different for the various businesses as each business has its own specifications that are followed.

Business rules for our grocery list application are as follows:

- ➢ Without a valid email address the user will not be able to sign in or sign up.
- ➢ Admin will be able to see and manipulate user data in order to assist them.
- ➢ Every grocery list should have a corresponding account name.
- ➢ Predictions for the grocery list will be different for every account that has been created.

6. Other Requirements

The application will also show a routine feedback window for the user to inform the software how well it is predicting the grocery items. And depending on the user feedback, changes are made to the parameters of the algorithm.

Appendix A: Glossary

Term	**Definition**
SRS	Software Requirements Specification
GUI	Graphical User Interface
HTTP	Hyper Text Transfer Protocol
FTP	File Transfer Protocol
QR Code	A machine-readable code consisting of an array of black and white squares.

Cloud	Storage server accessible over the internet

Appendix B: Analysis Models

No Analysis Models available in this draft.

Appendix C: To Be Determined List

1. **Nutrition Tab** : A feature displaying calories, sugar and other nutritional values of the various items.
2. **Currency Converter:** To convert the product prices to the user's local currency.