



Software Requirements Specification for the Noverty App

Matthew Tang

Ethan Lee

Tim Gu

Chanchev Mahendran

Kriti Ravindran

Table of Contents

1 Introduction	5
1.1 Purpose of Software Requirements Specifications	5
1.2 Scope of Product	5
1.3 Acronyms, Abbreviations, Definitions, Notational Conventions	6
2 Overall Description	7
2.1 Product Perspective	8
2.1.1 Use Case Diagram	8
2.1.2 Use Case Description	8
2.1.3 Actor Descriptions	9
2.2 Product Features	9
2.2.1 Activity Diagram for Use Case #1	10
2.2.2 Activity Diagram for Use Case #2	12
2.2.3 Activity Diagram for Use Case #3	14
2.2.4 User Stories	15
2.3 User Characteristics	16
2.4 Sources of Constraints on Requirements	18
2.5 Assumptions and Dependencies for the Novertly App	19
3 Specific Requirements	20
3.1 Domain Model	21
3.1.1 Domain Model Entity Descriptions	21
3.1.2 Domain Model Relation Descriptions	22
3.1.3 UI Sketches	23
3.1.3.1 UI Sketch #1 - Main Menu	23
3.1.3.2 UI Sketch #2 - Browse Items	24
3.1.3.3 UI Sketch #3 - Shopping List	25
3.1.3.4 UI Sketch #4 - Optimization Prompt	27
3.1.3.5 UI Sketch #5 - Optimization Results	28
3.1.3.6 UI Sketch #6 - Google Maps Routes	29
3.1.3.7 UI Sketch #7 - Store Main Menu	30
3.1.3.8 UI Sketch #8 - New Discount	31
3.1.3.9 UI Sketch #9 - Manage Inventory	32
3.1.3.10 UI Sketch #10 - Manage Stores	33
3.2 Scenarios	34
Introduction to Scenarios:	34
3.3 State Machine Models	43
3.3.1 Introduction to Navigation Map	43

	3
3.3.2 Navigation Map	44
3.3.3 UI State Labels	44
3.3.4 Transition Labels	45
3.3.5 UI Conditions/Events	47
3.4 Quality Requirements	48
3.4.1 Highest Priority Attributes	48
3.4.2 Rich Fit Criteria	50
References	51

Table of Figures

Figure 1: Use Case Diagram	8
Figure 2: UML Activity Diagram 1	10
Figure 3: UML Activity Diagram 2	12
Figure 4: UML Activity Diagram 3	14
Figure 5: Persona - Karen	17
Figure 6: Persona - Kevin	18
Figure 7: Specification Domain Model Diagram	21
Figure 8: UI Sketch - Main Menu	23
Figure 9: UI Sketch - Browse Items	24
Figure 10: UI Sketch - Shopping List	25
Figure 11: UI Sketch - Optimization Prompt	27
Figure 12: UI Sketch - Optimization Results	28
Figure 13: UI Sketch - Google Maps Routes	29
Figure 14: UI Sketch - Store Main Menu	30
Figure 15: UI Sketch - New Discount	31
Figure 16: UI Sketch - Manage Inventory	32
Figure 17: UI Sketch - Manage Stores	33
Figure 18: Navigation Diagram	44

1 Introduction

1.1 Purpose of Software Requirements Specifications

This document is intended to define the requirements and specifications of a grocery shopping route optimization mobile application. This document seeks to utilize the critical use cases of the design solution and explore the process of satisfying the three orthogonal use cases: to optimize grocery shopping list for time and discounts, to manage a grocery list, and for grocery stores to provide coupon and discount information about their products.

This document's intended audience consists of customers and the developers of the system. This includes technical staff such as product designers, testers, and software engineers. It is assumed that the reader would have prior knowledge of the grocery shopping process and the use of route navigation. Readers are expected to have a basic understanding of software requirements tools such as UML diagrams, use case diagrams, and navigation maps. After approval by primary stakeholders, this document can serve as a technical guide for the application's functionalities. This will provide software engineers a reference during implementation and for stakeholders to document and understand important functions.

1.2 Scope of Product

The software application described in this document is called the Noverty App. It is a mobile application and it seeks to aid customers in improving their grocery shopping experience by optimizing their shopping trips based on their preferences for cost and time savings. The Noverty App allows users to browse grocery items and add them to their shopping list. It then has users provide their preferences for cost savings, shopping time, and preferred mode of transportation. The routing algorithm then combines the user parameters, the location of the user, and their grocery list to output a list of possible routes that are optimized based on a set of user provided preferences listed above.

The inconveniences of trip planning using flyers and comparing route options without access to aggregated discount information are either reduced or eliminated by using this application. The Noverty App will provide users with a streamlined process to add items to their grocery list and transparently view discounts and prices of items at stores. Most importantly, the application will save the user time and money. Grocery stores will also benefit from this application; by uploading items and discount information to a centralized platform, customers will be able to easily explore their inventory. This will help increase the number of customers that visit their stores and increase revenue.

This document's scope does not describe the minutiae of the optimization algorithm nor the way in which routes are calculated and planned using the Google Maps API. Thus, the Novert App has limitations imposed by the quality and accuracy of the information it receives from the Google Maps API [1] and the resulting route produced by the optimization algorithm.

1.3 Acronyms, Abbreviations, Definitions, Notational Conventions

Definitions:

- **Novert App:** The mobile grocery route optimization application
- **Application:** The application (can be used interchangeably with Novert App)
- **Routing service:** Google Maps routing services
- **Users:** End users of the product. This group can be divided into two main groups of users who use the app for different purposes:
 - Customers: In the scope of this document, “customers” are defined as those who are using the Novert App to create grocery lists, browse discount information, and plan optimized grocery trips based on their user preferences.
 - Grocery Stores: In the scope of this document, “grocery stores” are defined as grocery store owners who use the Novert App to input the items they have in stock and any discount information related to their products.
- **Shopping constraints:** These are the preferences that a user (customer) inputs into the Novert App which are inputs to the routing algorithm (shopping constraints can be used interchangeably with user preferences). There are 3 types of shopping constraints:
 - Cost savings: This is the minimum amount of savings a customer is looking for
 - Shopping Time: This will be the maximum amount of a time a customer wants to shop for
 - Transportation: This will be the preferred mode of transportation
- **Routing Algorithm:** This combines the shopping constraints, the customer grocery list, and their current location to provide the Novert App with a list of routes that can be taken (can be used interchangeably with optimization algorithm).
- **Route:** A route is an output from the routing algorithm and provides the user (customer) with an optimized grocery trip run. It shows step by step how to navigate to each grocery store.
- **Software Requirements Specification:** A document that includes descriptions of functional and non-functional requirements of the system.

Acronyms:

- **UML:** Unified Modeling Language
- **SRS:** Software Requirements Specification

- **UI:** User Interface
- **API:** Application Programming Interface
- **GUI:** Graphical User Interface

Abbreviations:

- **App:** An abbreviation for the aforementioned application definition

2 Overall Description

Introduction to the Noverty App

In the following paragraphs, the primary use cases of the Noverty App are outlined. The section will walk through the Noverty App from a product perspective and dive into how the app will be used. The descriptions of primary actors and secondary actors will also be given. The following is a guideline of how the Noverty App will be explained from a product perspective:

1. Use Case Diagrams
2. Use Case Descriptions
3. Actor Descriptions

Furthermore, the product's features are then explained clearly with a corresponding use case description and workflow models for each core use case. The following is a guideline of how the features of the Noverty App will be explained.

1. Workflow Models
2. User Stories

After the product's features have been identified and explained thoroughly, the characteristics of users who intended to use the Noverty App will be explained. Any assumptions made about the users abilities will be discussed as well. There will also be a discussion on sources of constraints on requirements.

Lastly, the section will conclude with any assumptions and dependencies that were taken during the creation of the SRS of the Noverty App.

2.1 Product Perspective

By conducting several interviews and performing a stakeholder analysis, the three core use cases of the Noverty App are:

1. Optimize grocery shopping list for time and discounts
2. Manage a grocery list
3. Grocery stores provide coupon and discount information about their products

The rest of the report will use these 3 use cases as a pillar for describing the software requirements and specifications.

2.1.1 Use Case Diagram

The following is a use case diagram that illustrates how the Noverty App interacts with its actors.

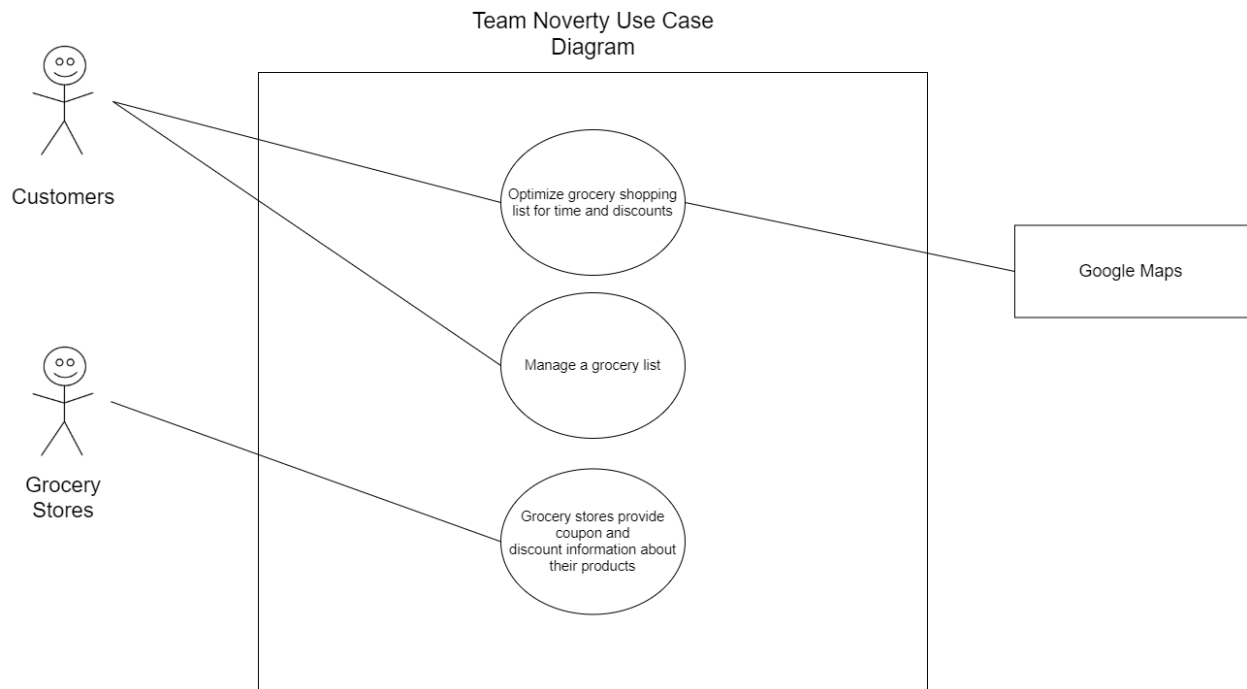


Figure 1: Use Case Diagram

2.1.2 Use Case Description

Optimize grocery shopping list for time and discounts (Primary Use Case #1)

The application will provide a platform that will allow customers to plan their optimized shopping plan based on their time, cost, and transportation preferences. Google Maps will be used to provide locations of the grocery stores and notify customers of possible routes based on distance and traffic conditions [1].

Manage a grocery list (Primary Use Case #2)

Customers will be able to add items to their grocery list and analyze the various discounts/coupons offered by grocery stores. They will be able to input their preferences in terms of time, cost, and transportation preferences.

Grocery stores provide coupon and discount information about their products (Primary Use Case #3)

This is the primary use case for grocery stores. They provide information of their weekly discounts and coupons that would ordinarily be distributed through flyers. This data allows the platform to recommend shopping plans for customers using the optimization algorithm.

2.1.3 Actor Descriptions

Customers (Primary)

These are the primary human actors using this application. They will be the ones using the app to plan trips using the grocery trip optimization algorithm.

Grocery Stores (Primary)

The grocery stores are the ones providing discount information and flyers so that the application can aggregate them into a single platform for the customers.

Google Maps (Secondary)

This is a non-human actor. The system will use Google Maps to map out transportation to and from grocery stores and determine the duration of the shopping trip.

2.2 Product Features

In this section, the features of the Novert App are described. Sample user stories are given and there will be a corresponding workflow model for each feature described.

The workflow model is a graphical representation of a series of tasks that are performed to accomplish each use case. The following are the UML activity diagrams and descriptions of the three use cases:

2.2.1 Activity Diagram for Use Case #1

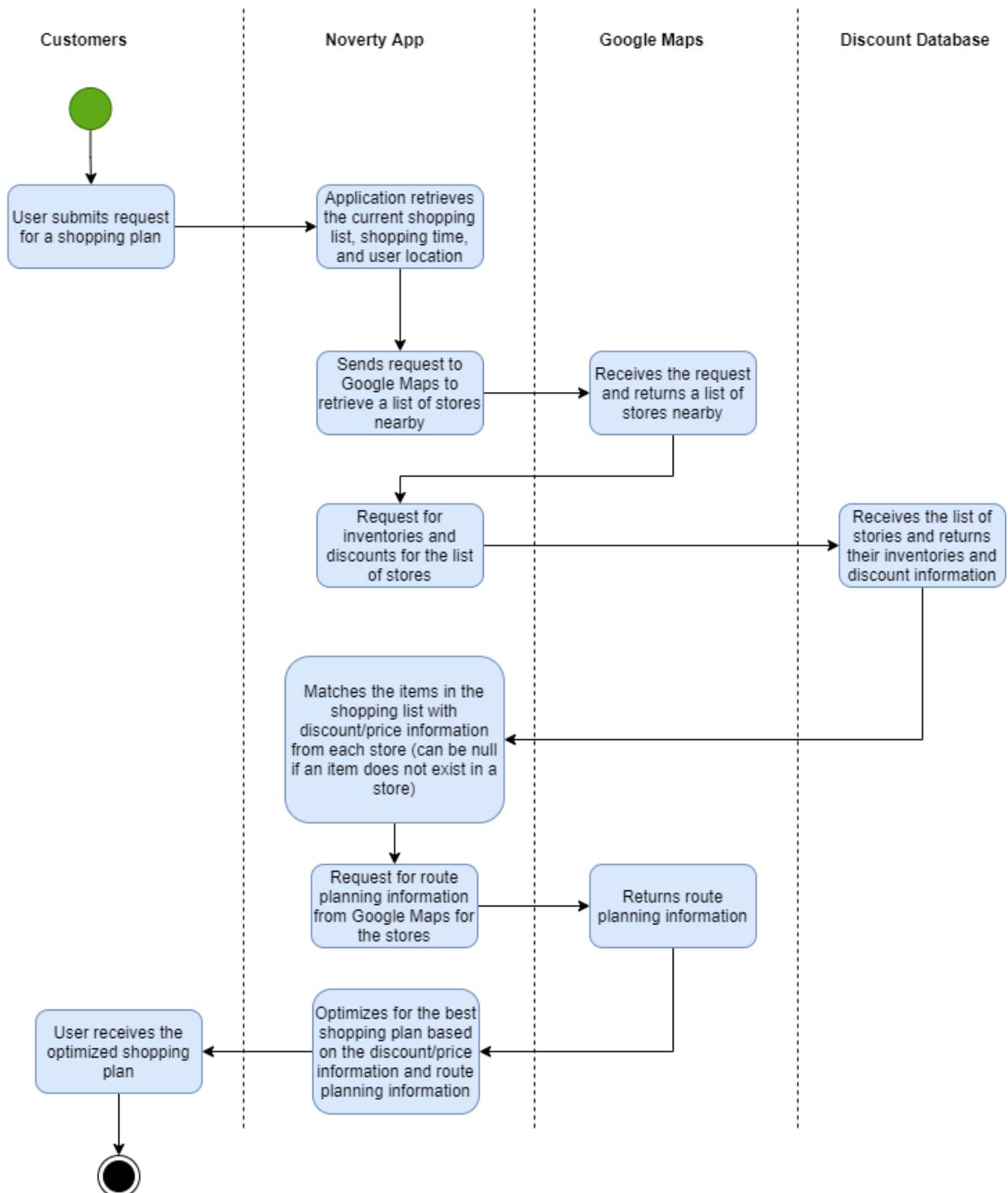


Figure 2: UML Activity Diagram 1 - Optimize grocery shopping list for time and discounts

Description of activity: To generate a shopping route, the user will have to submit a request with multiple shopping constraints. The application will then retrieve a list of discount information from the database and location information from Google Maps [1] based on the user's shopping list and location information. Finally, the application will use an algorithm to produce a shopping plan optimized for their user preferences and show it to the user (customer).

Justification of using an Activity Diagram: There is no complex data flow involved in this use case. All that is being exchanged are the location data and discount data. The complexity of the use case lies with the sequence in which the application interacts with the user, Google Maps, and the discount database to produce an optimized shopping plan. Hence, an activity diagram here is more appropriate than a data flow diagram.

Elicitation Techniques:

- Interview: Early adopters expressed this use case as the top use case
- Stakeholder Analysis: Figured out what pain points that the user has when grocery shopping and implemented solutions to those pain points in the application.

2.2.2 Activity Diagram for Use Case #2

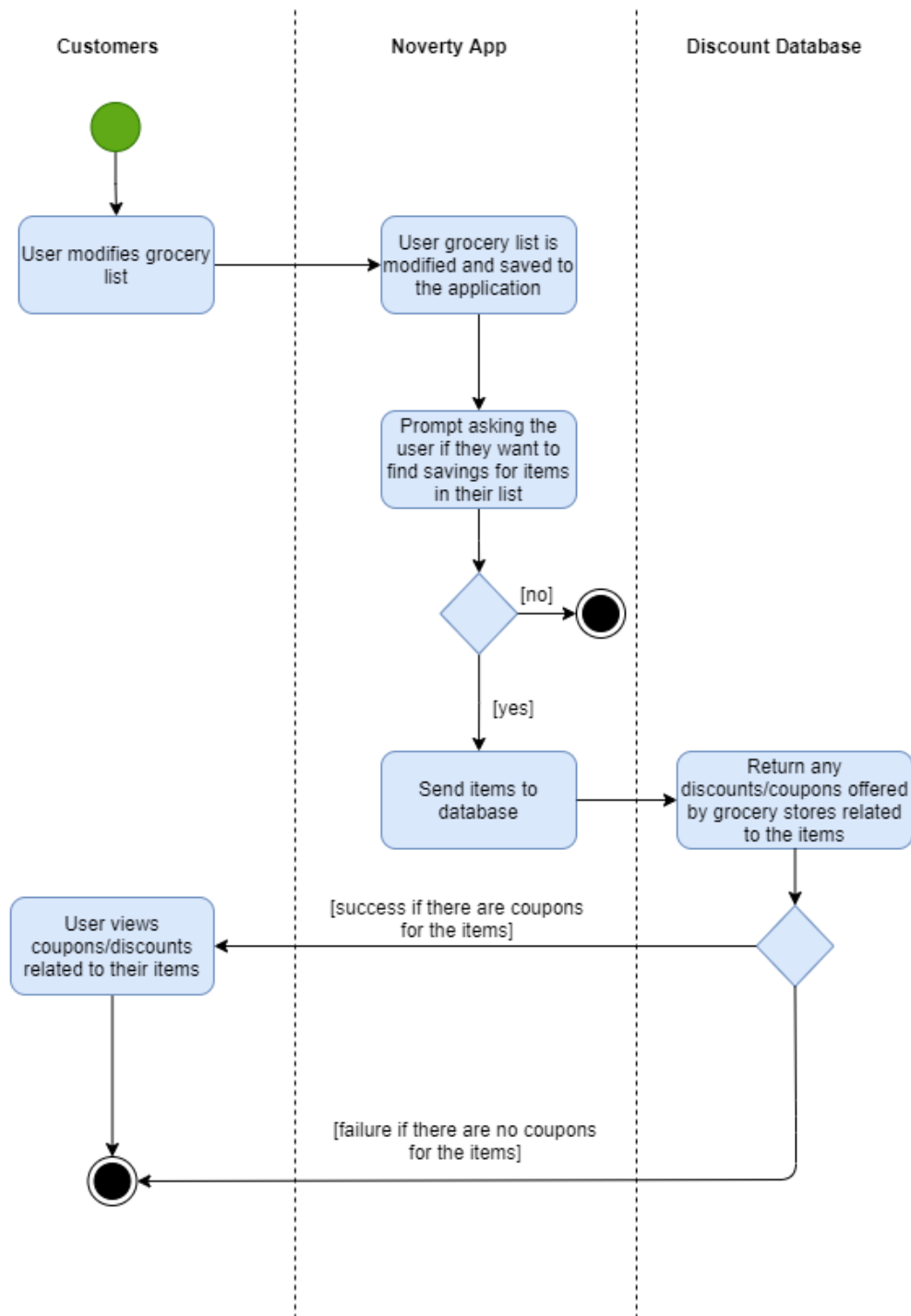


Figure 3: UML Activity Diagram 2 - Manage a grocery list

Description of activity: Users (customers) will be able to modify their grocery list. During modification, they can add or remove items to their grocery list and that will be saved on the Novert App. The app will then prompt the user if they want to find savings/discounts for the

items in their grocery list. If the user selects “no”, that is the end of the flow. However, if the response is “yes”, the app will send the items to the database and the database will return any discounts/coupons related to the items. If there are coupons/savings found, the user is then able to view them. If there aren’t any, that is the end of the flow.

Justification of using an Activity Diagram: The diagram is driven more by how the user (customer) interacts with the app. This is because the flow of data throughout the system is relatively straight forward. For example, the user modifies an item on their grocery list. This is a straightforward operation and the flow of data is not complex. Additionally the decision to look for savings related to the items in their grocery list is a binary yes or no. It would be more appropriate to see the activity flow of this decision instead of the data flow. Therefore, an activity diagram would be more suitable than a data flow diagram.

Elicitation Techniques:

- Brainstorming session: A group discussion within the Novert team and a collaborated ideation session with another CS445 team was used to generate this idea
- Interviews: In the interviews closed-ended questions and open-ended questions were utilized to gather opinions and suggestions from target customers about their expectations regarding managing a grocery list.

2.2.3 Activity Diagram for Use Case #3

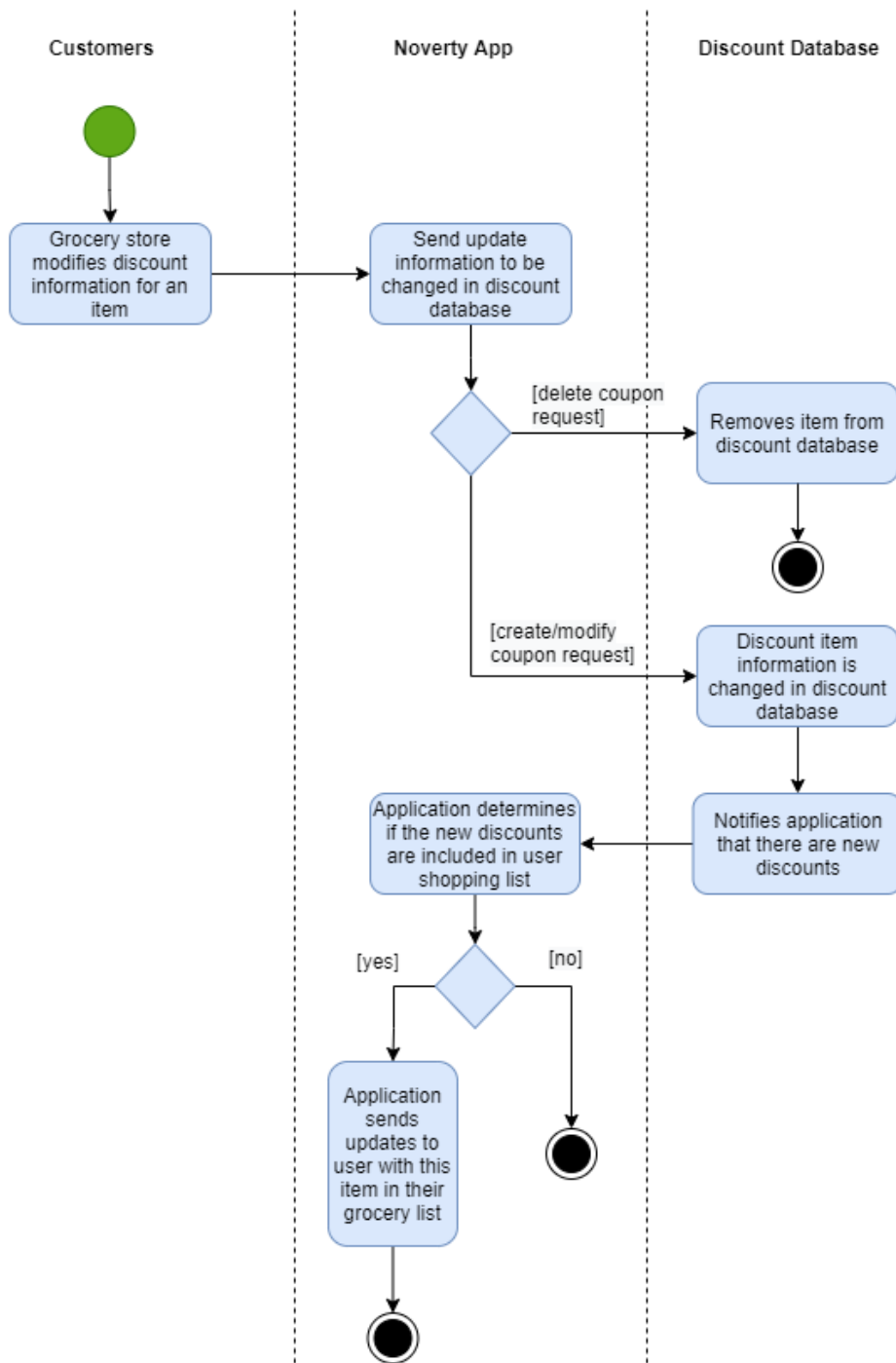


Figure 4: UML Activity Diagram 3 - Grocery stores provide coupon and discount information about their products

Description of activity: In this workflow diagram, the grocery store provides updates to the app with the newest discount information. If the grocery store removes discount information on an item, it is then subsequently deleted in the discount database. If the grocery store creates/modifies discount information on an item, in addition to updating the database with the new information, all the user (customer) applications will be notified with the update. If the updated item is in the user's grocery list, then a notification is sent notifying the user about the new discount information.

Justification of using an Activity Diagram: Similar to the two diagrams above, this workflow does not have a complex data flow. This is more suitable for an activity diagram because there are only two main operations: 1) CRUD operations on a database 2) If there are new discounts, check if any of the customers have items related to those discounts and if they do, send them a notification. These two operations are sequential and there is no complicated data flow. There is greater emphasis on the activity flow rather than the data flow which explains the choice of an activity diagram.

Elicitation Techniques:

- Stakeholder analysis: This technique was used to determine the functionalities and features needed for a grocery store to spend time and resources to provide information to the Novert App.
- Interview: Most grocery stores express this as the top use case.

2.2.4 User Stories

For each use case, the associated user stories are written from the point of view of the different user classes. User stories help with managing and expressing requirements. These are short statements that are written from the point of view of the users about any functionality or feature.

They are written with the following syntax: As a **user/role**, I want **something**, so that **benefit is achieved**.

Optimize grocery shopping list for time and discounts (Primary Use Case #1)

1. As a customer, I want to be able to balance between my time spent grocery shopping and saving money so that I can finish my grocery shopping within the time I've allocated while minimizing my costs.
2. As a customer, I want to take into consideration my travel time between multiple stores so that I can get the best prices for the items I want in the time I allocated for shopping.

Manage a grocery list (Primary Use Case #2)

1. As a customer, I want to be able to add items I want into a grocery list so that I can have all the items I want in a central page.
2. As a customer, I want to be able to remove items and modify my grocery list so that I can accommodate for any changes I have in my groceries.

Grocery stores provide coupon and discount information about their products (Primary Use Case #3)

1. As an owner of a small local convenience store, I want a platform to easily advertise for ongoing discount information so that I can attract more customers.
2. As an owner of a local grocery store, I want a unified platform to post my store inventory so that I can attract more customers.

2.3 User Characteristics

Users of the Noverty App are required to have a valid email and sufficient knowledge about using a mobile phone. It is assumed that they can create an account on the Noverty App with a valid email during registration. The following section explains further user characteristics:

Education Level:

There are no requirements for education to be able to use the Noverty App. Anyone that owns a device that is able to connect to the internet and has basic familiarity with using mobile apps would be able to use the app.

Disabilities:

Those who are visually impaired may find the app difficult to use as they would not be able to interact with the app well. The ability to interact with the app using voice may be added as a feature in the future to make the app more inclusive.

Experience and Technical Expertise:

Users who have experience with mobile apps and familiarity with specific interfaces that allow you to navigate from screen to screen would be able to use the app. In addition, if they are familiar with existing mapping services (ex: Google Maps or Apple Maps), they would find the app intuitive to use and understand.

Target Users:

- **Students**, especially post-secondary students would be one of the intended user groups. These students are familiar with using mobile apps and would often have time

and budget constraints for grocery shopping. This makes them one of the earliest adopters of the Noverty App.

- **Low income workers and low income families** are in a similar situation as students. They would likely want to optimize their shopping trips as much as possible to fit their time and budget constraints. They would also be an early user of the Noverty App.
- **Grocery stores** would be on the other end of the spectrum for the intended groups of users. They would want to advertise their deals from the stores on the Noverty App and this would increase the number of customers and business overall.

Below are two personas provided. Personas are useful when real users are not reachable or are too numerous to interview them all. The following are two personas for the Noverty App:


	<h3>Bio</h3>	<h3>Technical Background</h3>
	<p>Karen is a single mother of 1 infant and 1 young child. She attended community college for digital arts in the past but dropped out to work full time as a waitress to support her family. Her interests are: Reading, Travelling, and Photography. After work, she tutors her neighbours children and on the weekends she volunteers at the local food bank. She has a 2003 Honda Civic, however the car is approaching the end of it's lifespan.</p>	<p>Basic technological background. Has a cell phone but rarely uses it. She likes to use Flipp and RedFlagDeals.</p>
	<h3>Goals</h3>	<h3>Experiences</h3>
	<ul style="list-style-type: none"> • Save money towards her children's tuition • Spend more time with her children and not worry about spending time with other tasks such as grocery shopping 	<p>Collects flyers in the mail and searches online for coupons to print. She goes grocery shopping once a week but has to bring her children with her.</p>
<p>KAREN SMITH Waterloo, Ontario</p> <p>Gender: Female Age: 42 Occupation: Waitress Education: High School</p>		<h3>Pain Points</h3>
		<ul style="list-style-type: none"> • No good unified platform for flyers, promotions, and coupons. • Needs to travel between different grocery stores because her local grocery store does not stock baby formula and diapers.
<p><i>"My children are my number 1 priority. I never want them to ever worry about money."</i></p>		

Figure 5: Persona - Karen

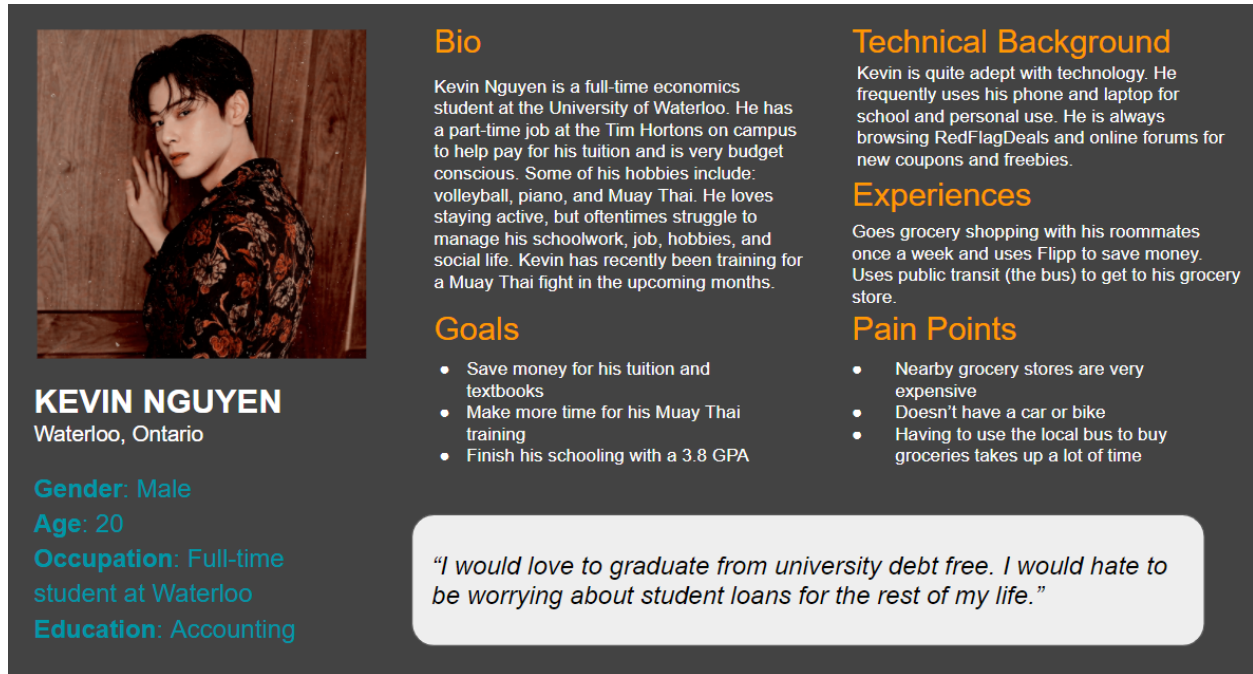


Figure 6: Persona - Kevin

2.4 Sources of Constraints on Requirements

Constraints restrict the design solution and the requirements engineering process. The following are a series of constraints for the Noverty App.

Existing System Limitations:

- The Noverty App is constrained by the Google Maps API [1]. It must conform to the format of data transmitted via the API. It must also adhere to the data transfer protocols used by the API.
- The machine learning algorithm used to generate an optimized route will improve the more it's trained and run. As a result, the algorithm may not provide the most accurate optimization results until a large number of grocery trips have been planned. Furthermore, these algorithms are dependent on sets of data provided as inputs. As a result, the app may not be requiring enough inputs from the user preventing the algorithm from creating an optimal route.

Safety and Security Considerations:

- The personal information of users, their grocery lists and shopping constraints should not be leaked or stored anywhere in the application
- Sensitive store information such as inventory should be kept secured and encrypted
- Administrators must be authenticated using a username and password

Laws:

- Laws concerning privacy of information must be abided by. In Canada, users need to be informed about the information that is collected, used and disclosed about them under PIPEDA [2].

2.5 Assumptions and Dependencies for the Noverty App

Assumptions address fidelity between interface phenomena and related environment phenomena as well as dependent adjacent systems. The following assumptions for the Noverty App are split up amongst the three use cases. These are elaborated on below:

1. Optimize a grocery list for time and discounts
 - Users provide their honest shopping constraints to the app when submitting their grocery route to the optimization algorithm.
 - Customers using the Noverty App will allow the app access to their location information
 - A customer's cellular device will provide accurate location information and have internet access
 - Since the application relies on Google Maps to provide location information to the routing algorithm, there is an assumption that Google Maps services are always available and accurate. Google Maps will provide up to date data concerning traffic conditions and local route maps [1].
2. Manage a grocery list
 - The computer systems that maintain the server are always online since the server functionalities will be outsourced to a third party provider.
3. Grocery stores provide coupon and discount informations about their products
 - The Grocery Store provides sufficient item inventory and discounts so that there will be enough selection for the customers to choose from.
 - The Grocery Store is responsible for providing accurate information about items and discounts.
 - Grocery Store has organization's correct edit access permissions.

The following dependencies that the Noverty App depends on are:

- Google Maps. The Noverty App relies on the services of the Google Maps API to provide data concerning traffic conditions, local routes maps and trip routing information [1].
- Routing Algorithm. The Noverty App relies on the accuracy and quality of the machine learning algorithm developed. This may rely on third party research and open sourced public machine learning code.
- Grocery Store Information. The entire app depends on grocery stores providing their item information as well as discount/savings.

3 Specific Requirements

Introduction to Domain Model and UI Sketches

Interface specifications provide information about the proposed system. It allows for the connections between the external environment and the system to be mapped in an interpretable manner. This section will contain a domain model and annotated user interface diagrams. The domain model uses UML techniques to represent interactions and data flow through the app. The user interface sketches show the top ten primary states users will come across while navigating through the app. Furthermore, these sketches will be annotated with features and functions explaining what a user will be able to do on the screen.

3.1 Domain Model

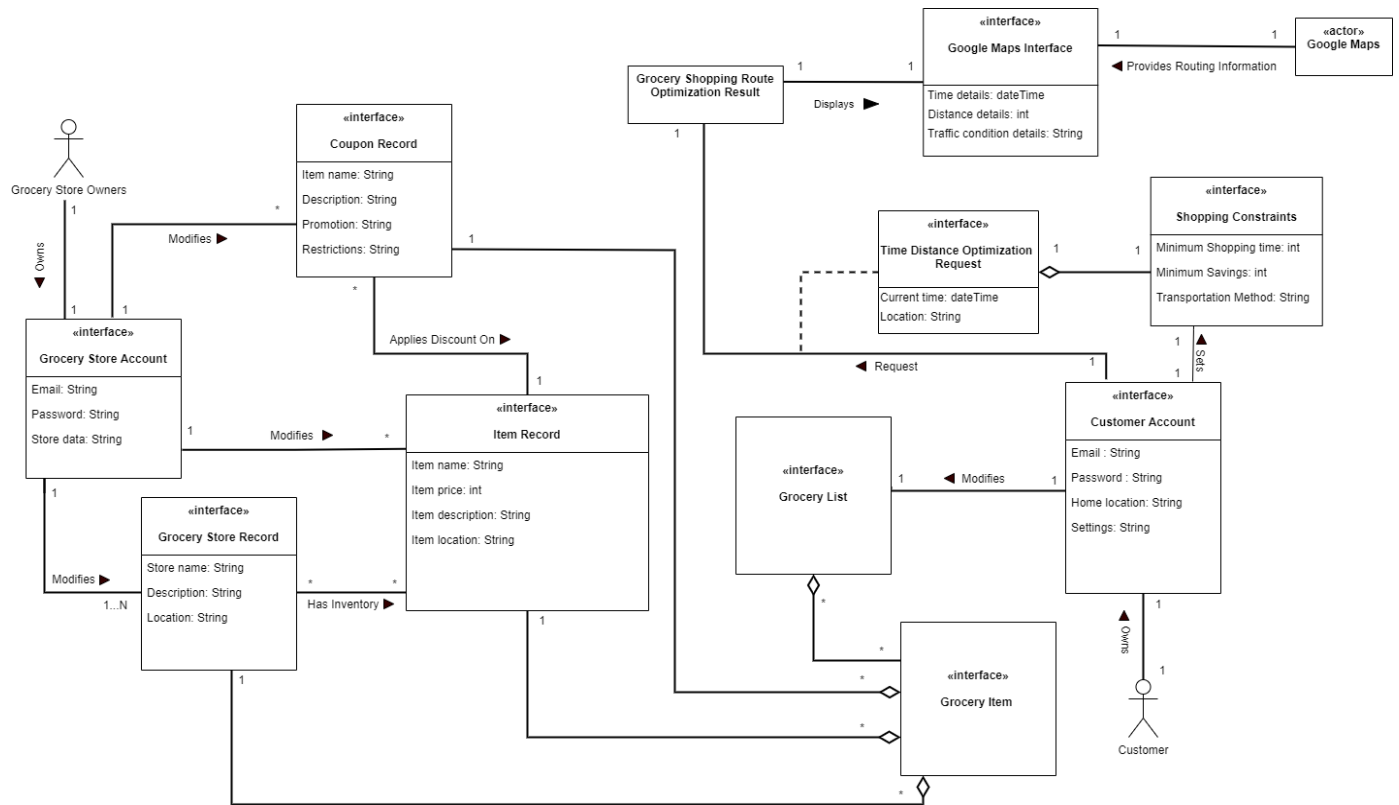


Figure 7: Specification Domain Model Diagram

3.1.1 Domain Model Entity Descriptions

Customer (actor):

Represents an individual using the app to optimize their grocery list

Grocery Store (actor):

Represents a grocery store that provides data on grocery items for the app

Google Maps (actor):

Represents the mapping service used to provide the path they have to take to visit stores given time details and distance details from the app

Customer Account:

Represent the customer actor which has information such as their email, password, and location

Grocery List:

Represents the grocery list that a customer has made

Item Record:

Represents the item that a grocery store sells

Coupon Record:

Represents a coupon that can be applied to an item

Grocery Store Record:

Represents a singular grocery store location in the world

Grocery Item:

Represents a grocery item that a user is adding or deleting to their grocery list.

Grocery Store Account:

Represents a grocery store or a franchise if there are multiple locations

Time Distance Optimization Request:

Represents a request to use the time distance optimization

Google Maps Interface:

Represents the map of the route provided for the grocery trip

Grocery Shopping Route Optimization Result:

Represents the optimized grocery shopping route returned to the user. There may be more than one, up to three routes returned

Shopping Constraints:

Represents the constraints the user wants to apply to the time distance optimization request

3.1.2 Domain Model Relation Descriptions

Customer Account:

A customer account is used to modify a grocery item aggregate to and from a grocery list.

Grocery Item:

A grocery item is made of an aggregation of grocery store record, item record, and coupon record.

Grocery Store Account:

A grocery store account will have a multiplicity of 1..N grocery store records where each grocery store record can be created, modified, or deleted. For example, there will be a general Walmart account and it will create a different grocery store record for each individual Walmart location. A grocery store account will also have a collection of item records that the entire franchise supports. Each individual item record has details such as item name, item price, item description, and item location which the grocery store account can modify. Additionally, a grocery store account will have a collection of coupon records that the entire franchise

supports. Similar to item records, every coupon record has details such as item name, description, promotion, and restrictions which the grocery store account can modify.

Shopping Constraints:

A customer account will set shopping constraints for their minimum shopping time, minimum savings, and their preferred transportation method.

Grocery Shopping Route Optimization Result:

A customer account will request an optimization route. During this request, the time optimization algorithm reads the grocery list and user preferences as input. Upon completion of the request the response returned is the Grocery Shopping Route Optimization Result and it will be displayed on the Google Maps Interface. The Google Maps Interface will request route information from Google Maps services.

3.1.3 UI Sketches

3.1.3.1 UI Sketch #1 - Main Menu



Figure 8: UI Sketch - Main Menu

This is the landing page when a customer first logs in.

GUI Components:

- Profile Button: Click to browse user settings
- Home Button: Click to return to home and back to Sketch 1 (Main menu)
 - Event Triggered: *clickHomeButton*
- Gallery of Coupons: Swipe left or right to see coupons
- Shopping List: Click pencil to modify list
 - Event Triggered: *clickShoppingList*
- New Item Button: Click to add new item
 - Event Triggered: *clickNewItemButton*
- Browse Button: Click to find items
 - Event Triggered: *clickBrowse*
- Plan Trip Button: Click to plan grocery trip
 - Event Triggered: *clickPlanTrip(ShoppingList)*

3.1.3.2 UI Sketch #2 - Browse Items

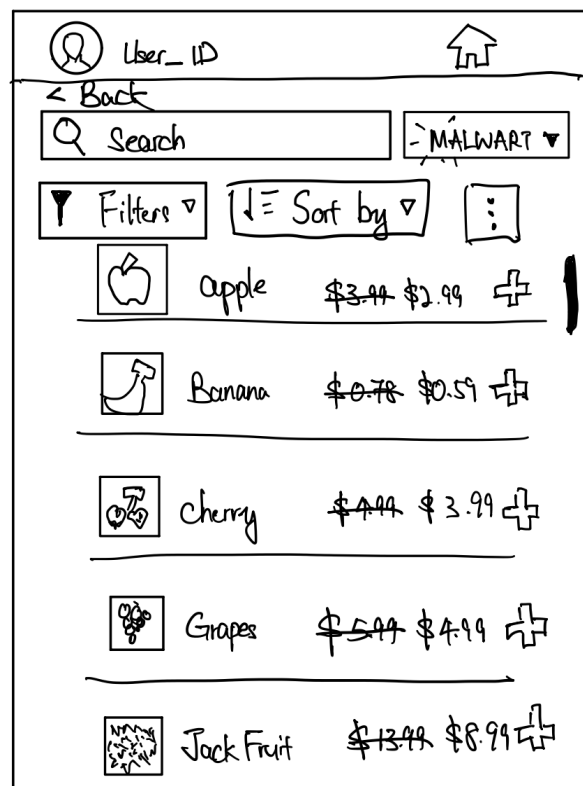


Figure 9: UI Sketch - Browse Items

This is the landing page for a customer that wants to browse items.

GUI Components:

- Search Input Field: Customer can input item they want to search
- Filter Dropdown: Customer can filter items by price or type
- Sort By Dropdown: Customer can sort by price or name
- Store Dropdown: Customer can select which store they want to browse items from
- Home Button: Click to return to home and back to Sketch 1 (Main menu)
 - Event Triggered: *clickHomeButton*
- Back Button: Click to return back to Sketch 1 (Main menu)
 - Event Triggered: *clickBack*
- Item Row: Inside an item row is the picture of the item, the name of the item, the original price, the discounted price, and a button for the customer to add the item into their grocery list
- Plus Button: Customer can click the plus button on each item row to add it to the shopping list
 - Event Triggered: *clickPlusButton*

3.1.3.3 UI Sketch #3 - Shopping List

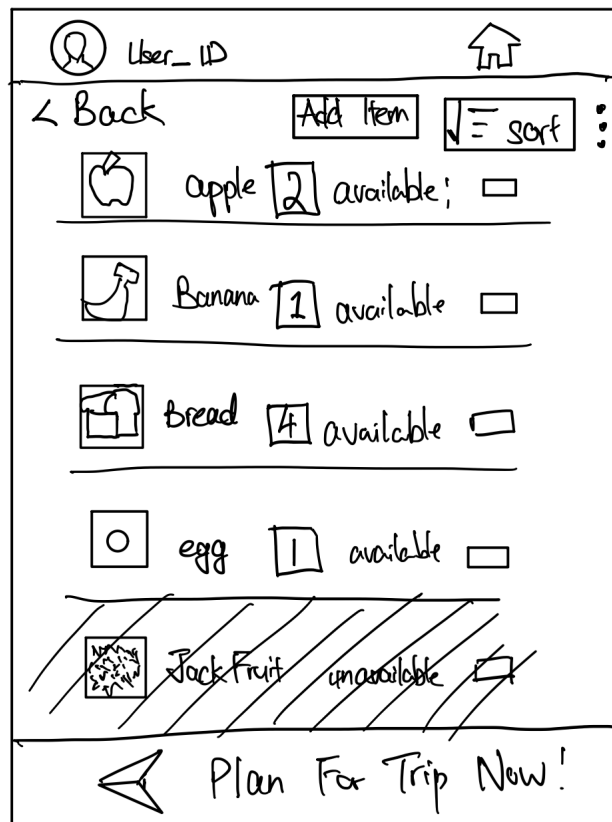


Figure 10: UI Sketch - Shopping List

This is the landing page for a customer that views their shopping list.

GUI Components:

- Item Row: Inside an item row is the picture of the item, the name of the item, the quantity of the item which can be modified, and it's availability across all the stores. If an item is unavailable, it is greyed out.
- Sort By Dropdown: Customer can sort by item name or availability
- Add Item Button: Click to add a new item to grocery list
- Home Button: Click to return to home and back to Sketch 1 (Main menu)
 - Event Triggered: *clickHomeButton*
- Back Button: Click to return back to Sketch 1 (Main menu)
 - Event Triggered: *clickBack*
- Plan Trip Button: Click to plan grocery trip
 - Event Triggered: *clickPlanTrip(ShoppingList)*
- Profile Button: Click to browse user settings
- Minus Button: User can click the minus button to remove the item from the shopping list.
 - Event Triggered: *clickMinusButton*
- Tools button (three dots): Contains advanced functionalities such as saving the shopping list as a template, load preset shopping list, etc.

3.1.3.4 UI Sketch #4 - Optimization Prompt

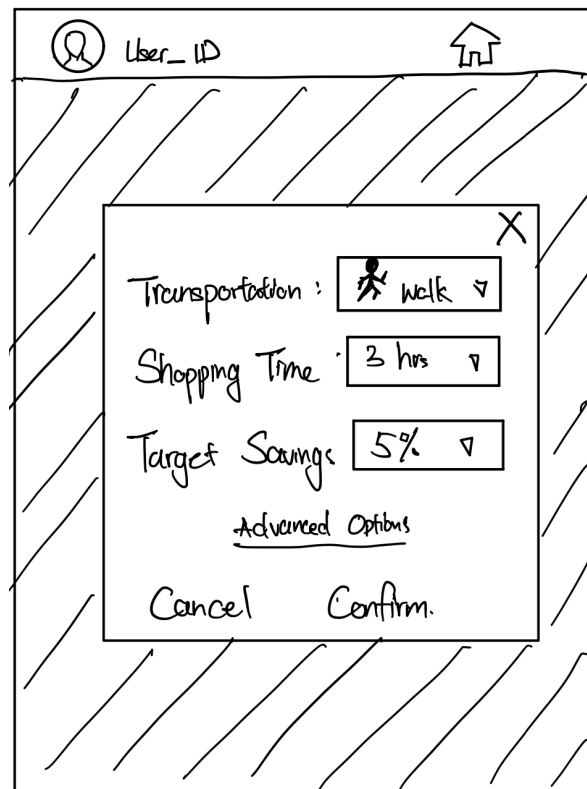


Figure 11: UI Sketch - Optimization Prompt

This is the landing page for a customer that has a grocery list and is ready to plan a trip. It is a prompt that appears in the centre of the device screen.

GUI Components:

- Profile Button: Click to browse user settings
- Home Button: Click to return back to Sketch 1 (Main menu)
 - Event Triggered: *clickHomeButton*
- Transportation Dropdown: Customer can select their preferred mode of transportation
- Shopping Time Dropdown: Customer can select their time preferences
- Target Savings Dropdown: Customer can select their savings preferences
- Advance Options: Click to select additional advanced trip options. This includes options such as membership to certain stores or ability to utilize loyalty points at certain stores.
- Confirm: Only after filling the 3 preference settings in the optimization prompt, the customer is then allowed to click confirm
 - Event Triggered: *clickConfirm*
- Cancel: Cancel Plan Trip. The user will be redirected back to the shopping list (Sketch 3)
 - Event Triggered *clickCancel*

3.1.3.5 UI Sketch #5 - Optimization Results

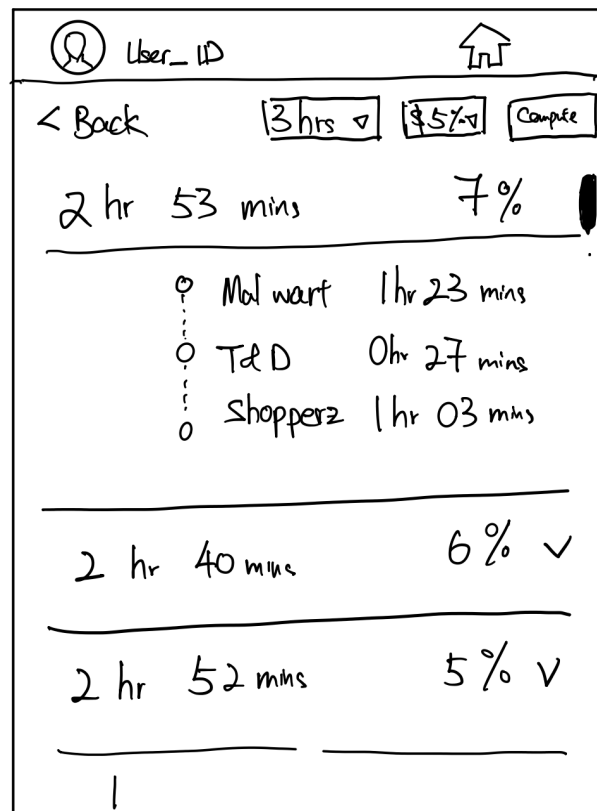


Figure 12: UI Sketch - Optimization Results

This is the landing page for the results of the trips found by the optimization algorithm.

GUI Components:

- Profile Button: Click to browse user settings
- Back Button: Click to navigate to previous screen
 - Event Triggered: *clickBack*
- Home Button: Click to return to home and back to Sketch 1 (Main menu)
 - Event Triggered: *clickHomeButton*
- Shopping Time Dropdown: Customer can select their time preferences
- Target Savings Dropdown: Customer can select their savings preferences
- Trip Row: Containing the time and savings information for planned trips. Click on the 'v' to expand trip details. Click anywhere else on the row to select the route and begin navigation
 - Event Triggered: *selectRoute(route)*

3.1.3.6 UI Sketch #6 - Google Maps Routes

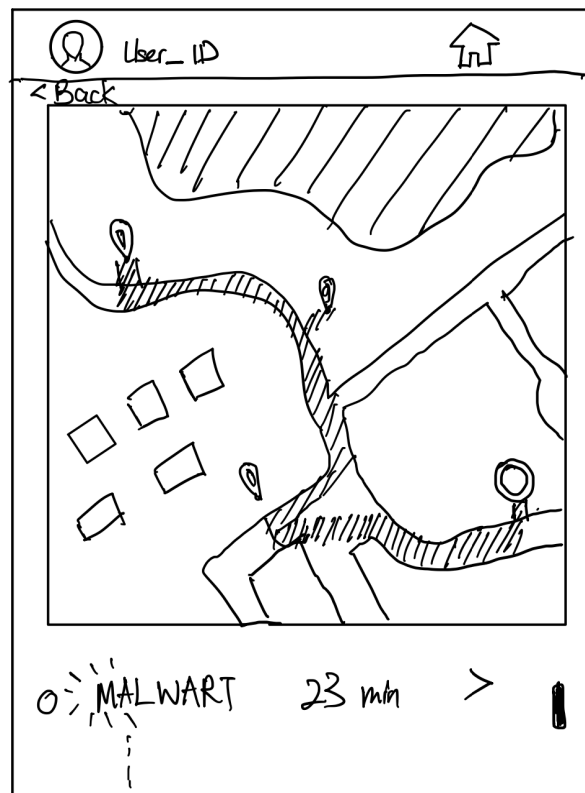


Figure 13: UI Sketch - Google Maps Routes

This page enables the customer to navigate to stores on their chosen shopping route.

GUI Components:

- Profile Button: Click to browse user settings
- Back Button: Click to navigate to previous screen
 - Event Triggered: *clickBack*
- Home Button: Click to navigate to home screen
 - Event Triggered: *clickHomeButton*
- Optimization Selection Dropdown Ribbon: Click to select different optimization preferences from the dropdown selectors
- Optimization Results List Item Card:
 - Time: Displays estimated time to complete trip
 - Discount Amount: Displays percentage saved on trip
 - Grocery Store Stops: Displays list of stores on trip and estimated time spent at each location
- Google Maps Output Widget: Google Maps widget contains the optimized output result.

3.1.3.7 UI Sketch #7 - Store Main Menu

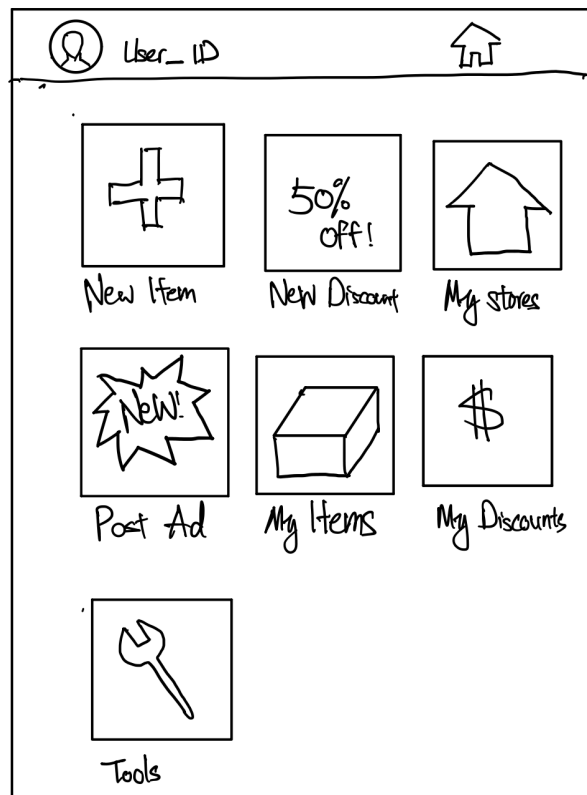


Figure 14: UI Sketch - Store Main Menu

This is the main landing page for a grocery store user when they first login.

GUI Components:

- New Item Button: Click to add a new item supported by the grocery store
- New Discount Button: Click to add a new discount/coupon
 - Event Triggered: *clickNewDiscount*
- My Stores Button: Click to browse the grocery stores under a user's account
 - Event Triggered: *clickMyStores*
- Post Ad Button: Click to post a new ad
- My Discounts Button: Click to browse the current discounts/coupons listed under your account
- My Items Button: Click to browse the current items listed under a user's (grocery store) account (Sketch 9)
 - Event Triggered: *clickMyItems*
- Tools Button: Update store preferences and verify stores
- Profile Button: Click to browse grocery store settings
- Home Button: Click to return to home and back to Sketch 7 (Store main menu)
 - Event Triggered: *clickHomeButton*

3.1.3.8 UI Sketch #8 - New Discount

Figure 15: UI Sketch - New Discount

This is the landing page when a grocery store adds a new discount/coupon,

GUI Components:

- Back Button: Click to navigate to previous screen
 - Event Triggered: *clickBack*
- Home Button: Click to return to home and back to Sketch 7 (Store main menu)
 - Event Triggered: *clickHomeButton*
- Discount Card: Provides information on the discount for an item
- Discount type badge: Specifies the type of discounts such as a price deduction, buy one get one free and other options
- New Price: Can manually enter new discount price
- Duration dropdowns: Two drop downs provide a start and end date for a discount
- Condition dropdowns: Apply to specific accounts
- Profile Button: Click to browse grocery store settings
- Confirm: Click to send the data entered to the database.
 - Event Triggered: *clickConfirm*

- Tools button (three dots): Provides additional regarding new discounts such as adding multiple items at once

3.1.3.9 UI Sketch #9 - Manage Inventory

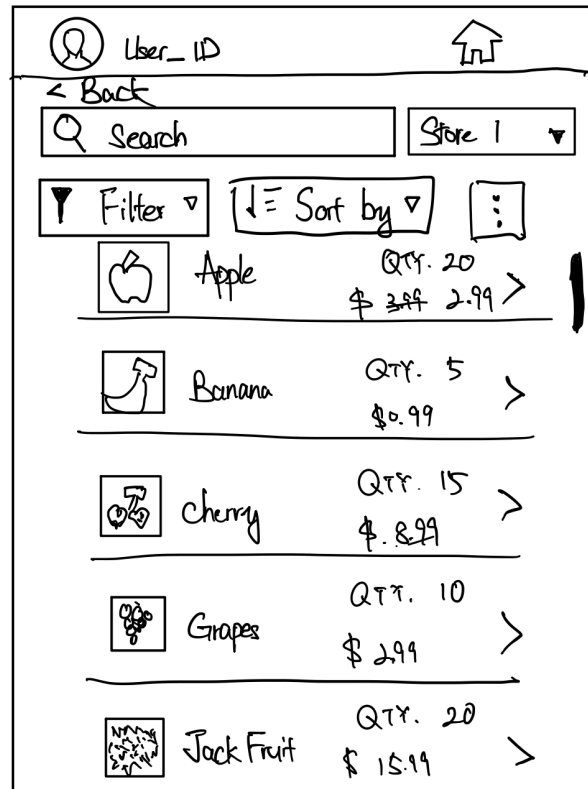


Figure 16: UI Sketch - Manage Inventory

This is the landing page when a grocery store clicks “My items” from UI Sketch 7.

GUI Components:

- Back Button: Click to navigate to previous screen
 - Event Triggered: *clickBack*
- Home Button: Click to return to home and back to Sketch 7 (Store main menu)
 - Event Triggered: *clickHomeButton*
- Store dropdown: Pick which store's inventory needs to be modified
- Filter button: Filter based on various properties like category of food
- Sort button: Sort based on various properties like characters
- Tools button (three dots): Additional information button such as the store it belongs to and an identification number. This button contains options to save and clear items from the grocery list

- List item: This is a row and it displays a grocery store item with an image, price, description and any applied discounts
- Profile Button: Click to browse grocery store settings

3.1.3.10 UI Sketch #10 - Manage Stores

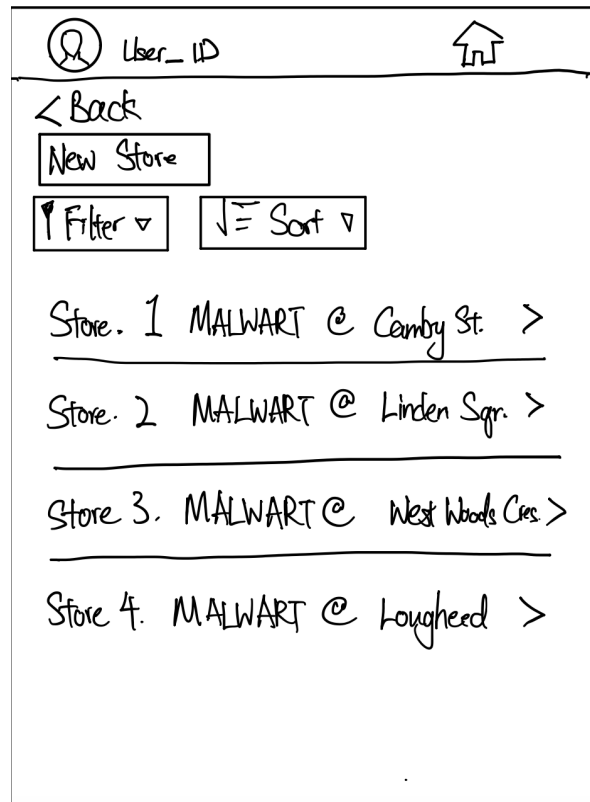


Figure 17: UI Sketch - Manage Stores

This is the landing page when a grocery store clicks “My Stores” from Sketch 7.

GUI Components:

- Back Button: Click to navigate to previous screen
 - Event Triggered: *clickBack*
- Home Button: Click to return to home and back to Sketch 7 (Store main menu)
 - Event Triggered: *clickHomeButton*
- New Store Button: Click to create a new store
- Filter button: Filter based on various properties like category of food
- Sort button: Sort based on various properties like characters
- Store List Items: Provides information on the stores, such as name and location. It can be clicked to create a modal with additional information.
- Profile Button: Click to browse grocery store settings

3.2 Scenarios

Introduction to Scenarios:

The three scenarios in this section address the three core use cases of the application. A scenario demonstrates actions within an execution path for a use case. This also comprises interactions between the system and external actors. Each use case includes the steps required to accomplish the normal use case, 4 notable alternatives, and 2 notable exceptions. The exceptions include instances where the scenario could not be completed.

Note that alternative and exception steps are numbered “x.y” where “x” denotes the step number in the sequence and “y” denotes the alternative/exception number.

Scenario 1:

Use Case Name: Optimize a grocery shopping list for time and discounts

Scenario Name and Number: Optimize a grocery shopping list (SFQ1)

Trigger: User presses the plan route button

Preconditions: User is logged in, has their location specified, and a grocery list with 1 or more items.

Stakeholders: Customers

Postconditions: A list of optimal grocery shopping routes are presented to the user for them to pick from.

Normal Case Steps: User wants to optimize their shopping route with their grocery shopping list for time and discounts

User	Noverty App	External Database Server
1. A user wants to optimize their shopping route. User selects the plan route from the app home page.		
	2. Presents questions to the user to complete regarding their preferences (time they want to spend shopping, % money they want to save, and mode of transportation)	
3. User completes series of questions about their shopping		

constraints and selects plan route button		
		4. Application retrieves grocery items and using user preferences calculates the top 3 optimal routes for the user
	5. Displays the top 3 calculated routes to the user with a clear emphasis on the first route	
Alternative 1 - User indicates specific grocery store they want to visit		
A1.1 User is not satisfied with route provided and wants to add a grocery store to visit and clicks the edit route button		
	A1.2 Receives previous plan route data with the data for store the user wants to include and presents questions to the user to complete regarding their route preferences with information previously entered filled for user to edit	
A1.3 User enters grocery store that they want to visit on their route and selects plan route button		
		A1.4 Go to 4
Alternative 2 - User indicates specific grocery stores they want to avoid		
A2.1 User is not satisfied with route provided, wants to avoid a specific grocery store and clicks the edit route button		
	A2.2 Receives previous plan route data with the data for the	

	store the user wants to avoid and presents questions to the user to complete regarding their route preferences with information previously entered filled for user to edit	
A2.3 User enters grocery store that they want to avoid and selects plan route button		
		A2.4 Go to 4
Alternative 3 - User is not satisfied with the total time spent, and changes the time parameter in order to save more time		
A3.1 User is not satisfied with the total time spent in the route provided and wants to avoid a specific grocery store. User clicks the edit route button		
	A3.2 Receives previous plan route data and discards the time constraint data. Presents questions to the user to set a new time constraint	
A3.3 User re enters the time they want to spend grocery shopping		
		A3.4 Go to step 4
Alternative 4 - User wants to visit less grocery stores than in the route provided, and changes the maximum grocery store input		
A4.1 User is not satisfied with the number of stores visited in the route provided. User clicks the edit route button.		
	A4.2 Receives previous plan route (user input) and presents questions to the user to complete regarding their route preferences with information previously entered for user to	

	edit	
A4.3 User enters the maximum number of grocery stores they want to visit		
		A4.4 Go to step 4
Exception 1 - Grocery store user wants to visit is closed		
	E1.1 Notify the user that selected grocery store is closed and display alternative route without selected grocery store	
Exception 2 - An optimal route is unable to be created that matches all the criteria specified by the user		
	E1.1 Notify user that an optimal route that matches all user criteria does not exist and suggest alternative routes	

Scenario 2:**Use Case Name:** Manage a grocery list**Scenario Name and Number:** Manage a grocery list (SFQ2)**Trigger:** User press add button in the items inventory view**Preconditions:** User is logged in and has their location specified.**Stakeholders:** Customers**Postconditions:** User's shopping list is modified.**Normal Case Steps:** User wants to add a single item to their local shopping list.

User	Noverty App	External Database Server
1. User wants to add an item to the shopping list. Opens the store inventory list and select the item		
	2. Receives the selected grocery item information and retrieves the information	

	associated (discount information, price, associated grocery store, etc.) with the selected item	
		3. Receive the grocery item and associated information and add it to the local current shopping list.
	4. Update the status on the grocery item in the inventory list to indicate that it has been added to the current shopping list	
Alternative 1 - User selects an item to be removed from the shopping list.		
A1.1 User selects the item to be removed from the shopping list		
	A1.2 Receives the user selected grocery item and prompt the user with a confirmation dialogue	
A1.3 User confirms they want to delete the item.		
		A1.4 Remove the item from the current shopping list
	A1.5 Updated the shopping list view to show that the item has been deleted	
Alternative 2 - User clears everything from the existing shopping list		
A2.1 User taps the tools button and then selects the deletion button to clear everything from the shopping list		
	A2.1 Receives input that the deletion button was selected,	

	and prompt the user with a confirmation dialogue	
A2.2 User confirms that they want to clear the shopping list		
		A2.4 Remove the item from the shopping list
	A2.5 Update the shopping list view to show an empty list	
Alternative 3 - User saves the current shopping list as a template shopping list.		
A3.1 User selects the tools , then selects the button on the shopping list view that saves the current list as a template		
	A3.2 Receives input that the saved grocery list button was selected. Prompt the user with a dialogue to input name for the template	
A3.3 User inputs the name for the template		
		A3.4 Saves the current shopping list as a template with the user-defined name.
	A4.5 Prompts the user with a success message.	
Alternative 4 - User loads a preset shopping list template to the current shopping list.		
A4.1 User selects the tools button on the shopping list view and then selects the button that loads a preset shopping list		
		A4.2 Retrieves a list of available preset templates

	A4.3 Prompt user with a list of available templates	
A4.4 Select the template the user wants to load		
	A4.5 Prompt the user a confirmation message	
A4.6 User confirms		
		A4.7 Retrieves the content of the selected preset shopping list
		A4.8 Add the retrieved items to the current shopping list
	A4.9 Update the current shopping list UI to show that items from the preset shopping list have been loaded.	
Exception 1 - The name for the preset shopping list the user wants to save already exists		
	E3.4 Notify the user that a preset shopping list with the name already exists.	
	E3.5 Go to A3.3	
Exception 2 - User loads a preset shopping list with items that are no longer in stock		
	E4.5 Warns the user that the unavailable items are no longer available	
	E2.2 Remove them from the list of items to be added to the current shopping list.	
E2.3 Go to 1		

Scenario 3:

Use Case Name: Grocery stores provide item and discount information about their products

Scenario Name and Number: Modify Store Repository (SFQ3)

Trigger: User (grocery store) presses the clickMyItems or clickNewDiscount buttons

Preconditions: User is logged in and has the correct edit permission in their user settings

Stakeholders: Grocery Store users

Postconditions: User's product information is modified

Normal Case Steps: User wants to add a single item to their collection of offered items

User	Mobile Application	External Database Server
1. User inputs item information to be uploaded		
	2. Receives item name, image, price, discounts information	
3. Confirm item information and commit changes.		
	4. Send information to file database	
		5. Receive item information and store in database
	6. Notify user (grocery store) of successful operation	
Alternative 1 - User adds multiple items using a csv		
A1.1 User selects the multiple items button under the advanced options		
A1.2 User uploads the correct csv file		
	A1.3 Go to step 2 and continue	
Alternative 2 - User modifies information about an item from the database		
A2.1 User selects the item they want to edit from the existing repository		

A2.2 User presses the edit button on the item		
	A2.3 A modal pops up with the modifiable information	
A2.4 Go to step 3 and continue		
Alternative 3 - User deletes an item from the database		
A3.1 User selects the item they want to edit from the existing repository		
A3.2 User presses the delete button on the item		
	A3.3 A confirmation button pops up on the screen	
A4.4 Confirm the deletion		
	A4.5 Send information to the database	
		A4.6 Finds item and deletes it from the database
	A4.6 Notify the user of a successful deletion	
Alternative 4 - User wants to add a limited time discount to an item		
A4.1 User selects existing item to add a limited time discount		
A4.2 User inputs limited time discount information regarding time duration and discount price		
	A4.3 Receives limited time discount information.	
A4.4 Go to step 3 and continue		

Exception 1 - The user does not have the correct edit permissions from the organization		
	E1.3 Permission error displayed if the user does have edit access. Stop action from going through	
Exception 2 - Database error that prevents an item from being saved		
		E2.1 Internal server/DB error and notifies the user of the error.
	E2.2 Notify user of failed operation due to error	

3.3 State Machine Models

3.3.1 Introduction to Navigation Map

The navigation map is a state machine model that illustrates the UI navigation process for the “Optimize grocery shopping list for time and discounts”, “Manage a grocery list”, and “Grocery stores provide coupon and discount information about their products” use cases described in section 3.2. The navigation map contains 10 UI screen states labeled 1 to 10. A brief description for each UI state is found in section 3.3.3, with more details in section 3.1.3 for all screens except the login screen. Each state transition is labeled with the corresponding UI event and actions. Descriptions for the transition labels can be found in section 3.3.4.

3.3.2 Navigation Map

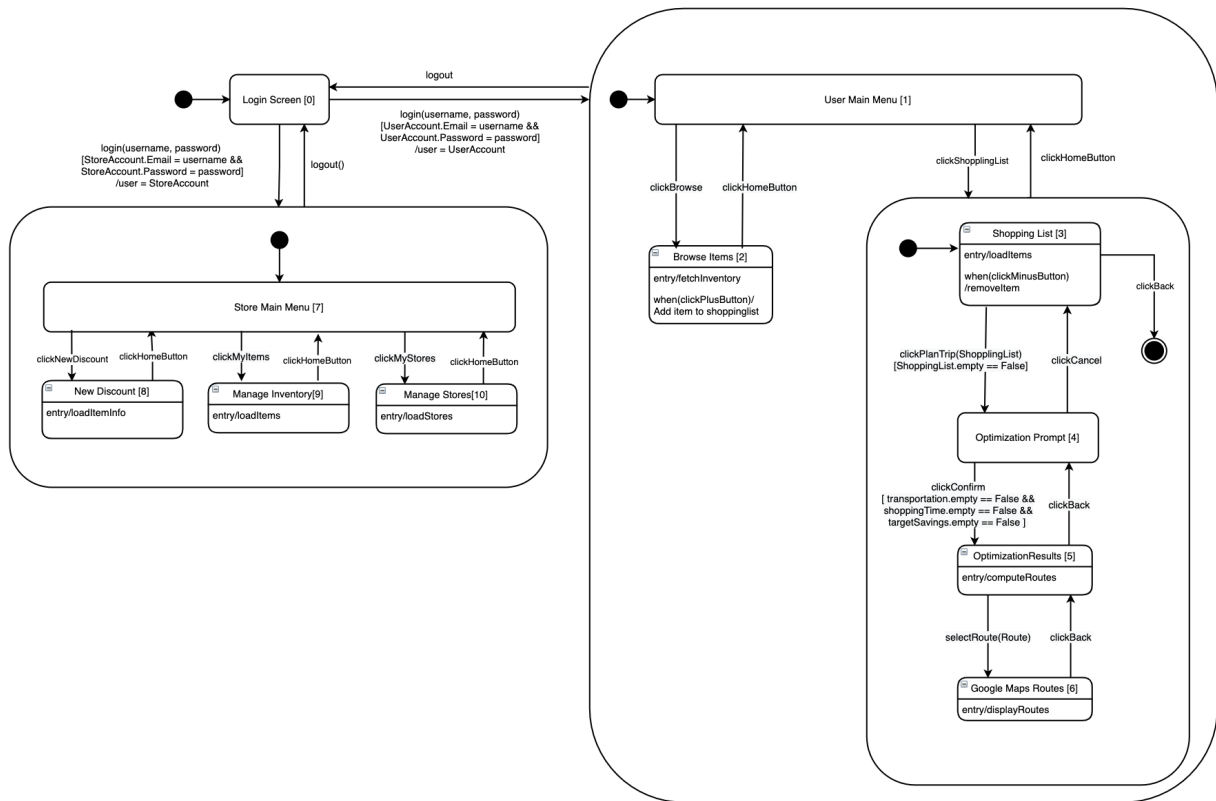


Figure 18: Navigation Diagram

3.3.3 UI State Labels

States	Description
[0] Login Screen	The screen that prompts the user to enter a user id and a password, which will be checked against the database to grant access to corresponding user account/store account
[1] User Main Menu Screen	Main landing screen for a user (customer)
[2] Browse Items Screen	This screen allows users (customers) to browse items
[3] Shopping List Screen	This screen displays a user's (customer) shopping list with the items it contains
[4] Optimization Prompt Screen	This screen is intended for a user (customer)

	that has a shopping list and is ready to plan a trip. It is a prompt that appears in the centre of the device screen
[5] Optimization Results Screen	This screen displays the results of the trips produced by the optimization algorithm
[6] Google Maps Routes Screen	This screen enables the user (customer) to navigate to stores on their chosen shopping route
[7] Store Main Menu Screen	Main landing screen for the stores
[8] New Discount Screen	Screen creates a form in which a information can be entered about a discount
[9] Manage Inventory Screen	Screen shows the items that belong to a store, allowing filtering, searching, choosing which store location and obtaining information about a selected item
[10] Manage Stores Screen	Screen shows a dropdown of the various store locations that belong to store account

3.3.4 Transition Labels

UI	Function/Transitions	Description
[#1 Main Menu]	clickShoppingList	User (customer) presses this icon to to access their grocery list
[#1 MainMenu]	clickBrowse	User (customer) presses this icon to be sent to a list on searchable items from various stores
[#2 - Browse Items]	clickPlusButton	User (customer) clicks this button to add an item to the grocery list
[#3 - Shopping List]	clickMinusButton	User (customer) clicks this button to remove an item from the grocery list

[#3 - Shopping List]	clickPlanTrip(ShoppingList)	User (customer) presses this icon to send the grocery list to the optimization algorithm to plan out a route
[#4 - Optimization Prompt]	clickCancel	User (customer) clicks this button to cancel an optimization request
[#4- Optimization Prompt]	clickConfirm	User (customer) clicks this button to process the optimization request
[#5 - Optimization Results]	selectRoute(route)	User (customer) clicks this button to send the planned route to be displayed on the Google Maps interface
[#7 - Store Main Menu]	clickNewDiscount	User (store owner) clicks this button to load a form to create a new discount
[#7 - Store Main Menu]	clickMyItems	User (store owner) clicks this button to access a landing page, displaying a page containing information about items, and GUI components to modify these items
[#8 - Store Main Menu]	clickMyStores	User (store owner) clicks this button to access a landing page, displaying a page enabling individual stores to be managed
[Login Screen]	login(username, password)	<p>User clicks login after entering credentials. The credentials will be checked against the server backend before granting access to the account.</p> <p>If the user is authenticated then go to the user main page (ex: user main menu, or store main menu)</p>

[Login Screen]	logout	User clicks the logout button to log out of the current account.
[Top Navigation bar]	clickHomeButton	User clicks button to return to home screen
[Top Navigation bar]	clickBack	User clicks button to return to previous screen

3.3.5 UI Conditions/Events

Conditions/Events	Description
login(username, password) [UserAccount.Email = username && UserAccount.Password = password] /user = UserAccount	When a user (customer) attempts to login to the application, they will enter their username and password which will be checked against the records in a server backend before granting authentication and access to the account.
login(username, password) [StoreAccount.Email = username && StoreAccount.Password = password] /user = StoreAccount	When a user (store) attempts to login to the application, they will enter their username and password which will be checked against the records in a server backend before granting authentication and access to the account.
entry/loadItemInfo	The entry action of accessing the “New Discount” screen. It will load item info every time that it enters this screen.
entry/loadItems	The entry action of accessing the “Manage Inventory” screen. It will load item items every time that it enters this screen.
entry/loadStores	The entry action of accessing the “Manage Stores” creen. It will load store information every time that it enters this screen.
entry/computeRoutes	The entry action of accessing the “Optimization Results” screen. It will compute a list of routes every time that it enters this screen.

entry/displayRoutes	The entry action of accessing the “Google Maps Routes” screen. It will display the selected route on Google Maps.
entry/fetchInventory	The entry action of accessing the “Browse Items” screen. It will load all the items that a store contains.
when(clickPlusButton)/ Add item to shopping list	When the plus button is clicked an item is added to the shopping list.
entry/loadItems	The entry action of accessing the “Shopping List” screen. It will load item items every time that it enters this screen.
when(clickMinusButton)/ removeItem	When the plus button is clicked an item is removed from the shopping list.
ShoppingList.empty() == False	A user(customer) can only plan a trip, if their shopping list is not empty.
transportation.empty() == False && shoppingTime.empty() == False && targetSavings.empty() == False	A user(customer) can only confirm the optimization request, if they have filled in all of the user constraints.

3.4 Quality Requirements

3.4.1 Highest Priority Attributes

The top three quality requirements determined were: Reliability, Usability, and Efficiency. Below is the result of the 100-dollar prioritization chart used during the quality requirements interviews:

	Requirements					
Scores	Efficiency	Usability	Scalability	Reliability	Performance	Security
Student 1	30	20	5	30	10	5

Student 2	10	20	5	40	20	5
Student 3	25	20	3	30	20	12
Student 4	20	20	5	35	15	5
Adult 1	15	30	0	15	15	15
Sum	100	110	18	150	80	42

The 100 dollar test was utilized in which each interviewee was able to allocate points to each quality attribute summing to 100. The top three quality attributes were chosen to be the focus of the rich fit criteria discussed in section 3.4.2.

From the 100 dollar prioritization chart, the top 3 priorities were reliability, usability, and efficiency. Reliability is the top priority of the 3, as a total of 150 dollars were allocated to it and the majority of interviewees value this requirement. Usability is the second highest priority of the 3, with a value of 110 and efficiency is the third highest priority with a value of 100.

Below are the descriptions of the top three highest-priority attributes and their corresponding objective metrics:

- 1) **Reliability:** For reliability the 2 major criteria are: Discount information being accurate and trips planned by the algorithm find an optimization between time and money.
 - Measurements: Percentage cost deviation between the predicted amount time and money spent versus the actual amount time and money spent in a trip.
- 2) **Usability:** Usability refers to the ease of accessing features of the app including the shopping list and discount information.
 - Measurements: Number of clicks to access most frequently used features
- 3) **Efficiency:** Efficiency in that context of this project refers to finding a path that minimizes the amount of money a user spends on groceries within the time they want to spend grocery shopping.
 - Measurements: Percentage of money saved with the same amount of time saved.

3.4.2 Rich Fit Criteria

Below is the rich fit criteria table elicited from the quality requirement interviews:

	Outstanding	Target	Minimum
Reliability	mean = 0.4% std = 0.49	mean = 4.2% std = 0.98	mean = 11% std = 3.74
Usability	mean = 1.8 clicks std = 0.98	mean = 4.2 clicks std = 1.47	mean = 6.8 clicks std = 1.72
Efficiency	mean = 25% std = 4.47	mean = 13% std = 2.45	mean = 6% std = 2

The rich fit criteria table contains the mean and standard deviation for the top three quality requirements. The top three quality requirements were determined by summing up the results from the 100 dollar prioritization chart in section 3.4.1. For the rich fit criteria, there were 1 student and 4 adults (who have full-time jobs) interviewed regarding the top three priority quality attributes.

Based on the rich fit table, it is evident that more participants are required in order to obtain more accurate mean values for reliability, usability, and efficiency. The standard deviation for almost all of the criteria are very high, meaning that the data is not clustered around the mean, but rather more spread out. This means that the numbers may contain outliers, and with such a small sample size (5 interviewees), the data could be skewed.

References

[1] “Geo-location APIs | Google Maps Platform | Google Cloud,” *Google*. [Online]. Available: <https://cloud.google.com/maps-platform/>. [Accessed: 17-Apr-2021].

[2] Office of the Privacy Commissioner of Canada, “PIPEDA in brief,” Office of the Privacy Commissioner of Canada, 07-Jun-2019. [Online]. Available: https://www.priv.gc.ca/en/privacy-topics/privacy-laws-in-canada/the-personal-information-protection-and-electronic-documents-act-pipeda/pipeda_brief/. [Accessed: 17-Apr-2021].