Software Requirements Specification

for

P1-Group 5

Royal Melbourne Institute of Technology

21/08/23

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

Documentation

P01 - Group 05

Name	Student ID	Contribution
Shreyas Venadan	s3944652	25%
William Truong	s3945703	25%
Brendon Rodrigues	s3945693	25%
Sandrup Sasikumar	s3943775	25%

Contribution Statement

Shreyas Venadan - 1.2 Stakeholder, 2.1 Product Perspective, 2.3 User Classes and Characteristics, 5.2 Safety Requirements, User Story #5

William Truong - 2.4 User Stories (User Story #1 and User Story #2), 3.1 User Interface, 5.3 Security Requirements

Brendon Rodrigues - 2.5 System Architecture, 2.6 Data Model, 3.2 Software Interfaces, 5.5 Testing and Acceptance Criteria, Product Backlog, Sprint 1, Sprint 2.

Sandrup Sasikumar - 1.1 Purpose, 2.2 Product Functions, 2.7 Design and Implementation Constraints and Assumption, 4.1 System Feature 1, 4.2 System Feature 2, 5.1 Performance Requirements, User Story #

Meeting Minutes: 192

GitHub Link

https://github.com/cosc2299-sept-2023/team-project-p01-05

Video Contributions:
https://drive.google.com/drive/folders/18u-rSkXtVOrG4HMenV-UHFRe1r4PghVv?usp=sharing

1.　Introduction

1.1　Purpose

The objective of this Software Requirements Specification (SRS) is to provide a detailed description of all features and requirements for the initial release (version 1.0) of the SuperPrice Price Matching and Delivery Application. This document intends to serve as a comprehensive technical reference for developers building the system.

The SuperPrice application aims to deliver an integrated platform enabling consumers to search for groceries, compare pricing across retailers, and schedule deliveries. The core goals are:

- *Enable users to efficiently locate products and identify the lowest available pricing through a database.*
- *Provide accurate visibility into stock levels and pricing across supported supermarkets.*
- *Allow users to select preferred delivery dates, locations, and options.*
- *Supply users with relevant notifications about price changes and promotional offers.*
- *Facilitate account signup and management for streamlined re-ordering.*
- *Implement an accessible, user-friendly interface design.*
- *Incorporate effective search, browse, and comparison capabilities.*
- *Support secured payment processing and order management functions.*

This SRS document defines the full set of features, system components, interfaces, dependencies, and performance parameters that must be satisfied based on analysis of user needs and project goals. Please advise if any part requires elaboration or clarification.

1.2　Stakeholders

A stakeholder is either an individual, group, or organisation that's impacted by the outcome of a project or business venture. This project involves a diverse group of stakeholders, each playing a crucial role in the development of it.

The first stakeholder is the Product Owner. They are the primary entity for whom the SuperPrice application is being developed for. Not only are they the project sponsor, but they oversee the development process to ensure it aligns with their business objectives. They determine what needs to be done and set the priorities to deliver the highest quality product.

Next is the scrum master, who protects the scrum process and prevents distractions. They are a crucial facilitator within the scrum framework, responsible for guiding the team in pursuit of agile excellence.

The next stakeholder is the development team which will consist of several key members who are instrumental in bringing the project to life. They take on and determine how to deliver chunks of work in frequent increments.

Another crucial stakeholder is the supermarkets and retailers. A key aspect of the SuperPrice's functionality is its integration with various supermarkets and retailers. They will need to actively collaborate with the website by providing real-time product information and pricing data, allowing customers to get their desired product for the cheapest price. This participation from the supermarkets and retailers is essential for the website to deliver accurate and up-to-date results.

2.   Overall Description

2.1   Product Perspective

The SuperPrice website is a new product aimed at revolutionising grocery shopping for money conscious users. It is not a replacement for existing systems, rather a standalone innovation that changes the way all customers shop. This website does not replace existing systems as no major website that compares grocery prices across multiple stores as well as handles the delivery. Although it does not replace existing websites, it will collaborate with various supermarkets and retailers to provide users with real-time pricing data.

2.2   Product Functions

The SuperPrice website needs to enable users to:

- *Easily search for specific groceries or browse aisles and categories just like in a store.*
- *Compare apples to apples (and bananas to bananas!) by pulling up real-time prices for the same products across different supermarkets.*
- *Choose delivery options that fit their schedule - whether it's ASAP, next day, or weekly.*
- *Get alerts about price drops and specials on their favourite buys.*
- *Create an account to save preferred stores, lists, delivery spots, payment info, and other details to make reordering a cinch.*
- *Benefit from a super intuitive interface that makes the experience smooth as butter.*
- *Access accurate and current info on inventory so they know exactly what's in stock.*
- *Get order confirmations and have full visibility into the status of their cart and delivery.*
- *Feel secure entering payment details and know transactions are safely handled.*
- *Sync seamlessly with real-time databases from participating stores.*

The goal is for all these capabilities to come together to create a frictionless, value-driven grocery shopping and delivery experience - something users can't wait to use over and over again. That's what I envision for the app. Let me know if any part needs more clarification!

2.3   User Classes and Characteristics

The SuperPrice website will only work efficiently and effectively if there is a diverse range of user classes. Each of these classes will have distinct characteristics and requirements, allowing them to do the tasks they need with ease.

The first user class is the shoppers. This class will indefinitely have the most users out of all the user classes as they are the core customers of the website. The shoppers will use the website to

search for groceries, compare prices across retailers, and schedule deliveries. Their technical expertise will be varied as there could be senior citizens who want their groceries delivered all the way to young adults who are well versed with online shopping. Depending on their grocery shopping needs, they will frequently use this website, especially if they are money conscious.
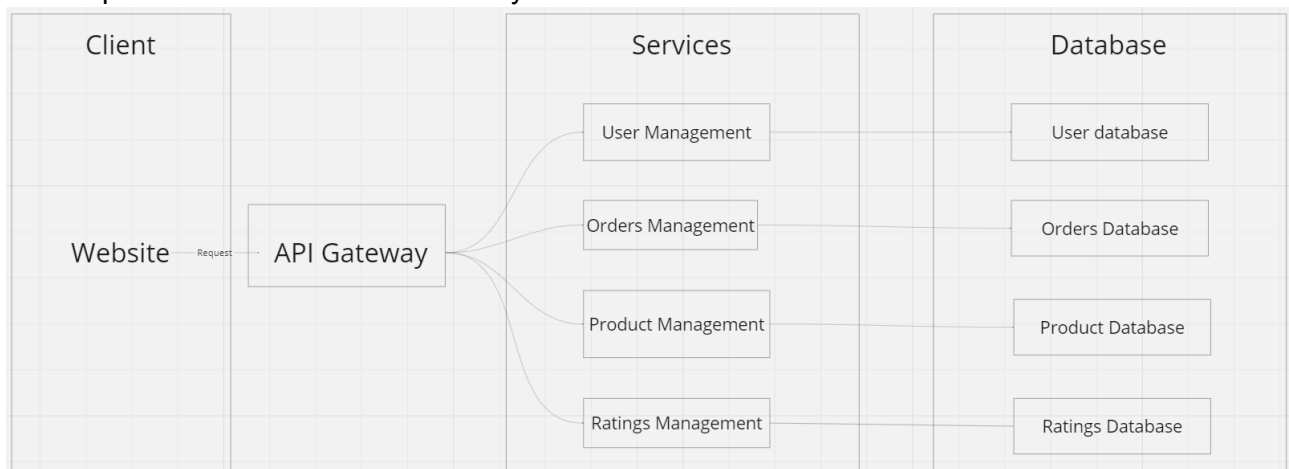
The next user class is the administrators themselves. These users are responsible for managing and maintaining the SuperPrice website which includes handling user accounts, system configuration and ensuring data accuracy. Unlike the shoppers, all of the administrators will have high technical proficiency in order to carry out system administration tasks. Admins will have a higher security/privilege level, meaning they will have more control in order to maintain the website in case something goes wrong. These users will only occasionally use the website in order to fix or update some functionality, other than that, the website should be mostly self-run.

2.4   User Stories

https://github.com/orgs/cosc2299-sept-2023/projects/192/views/1
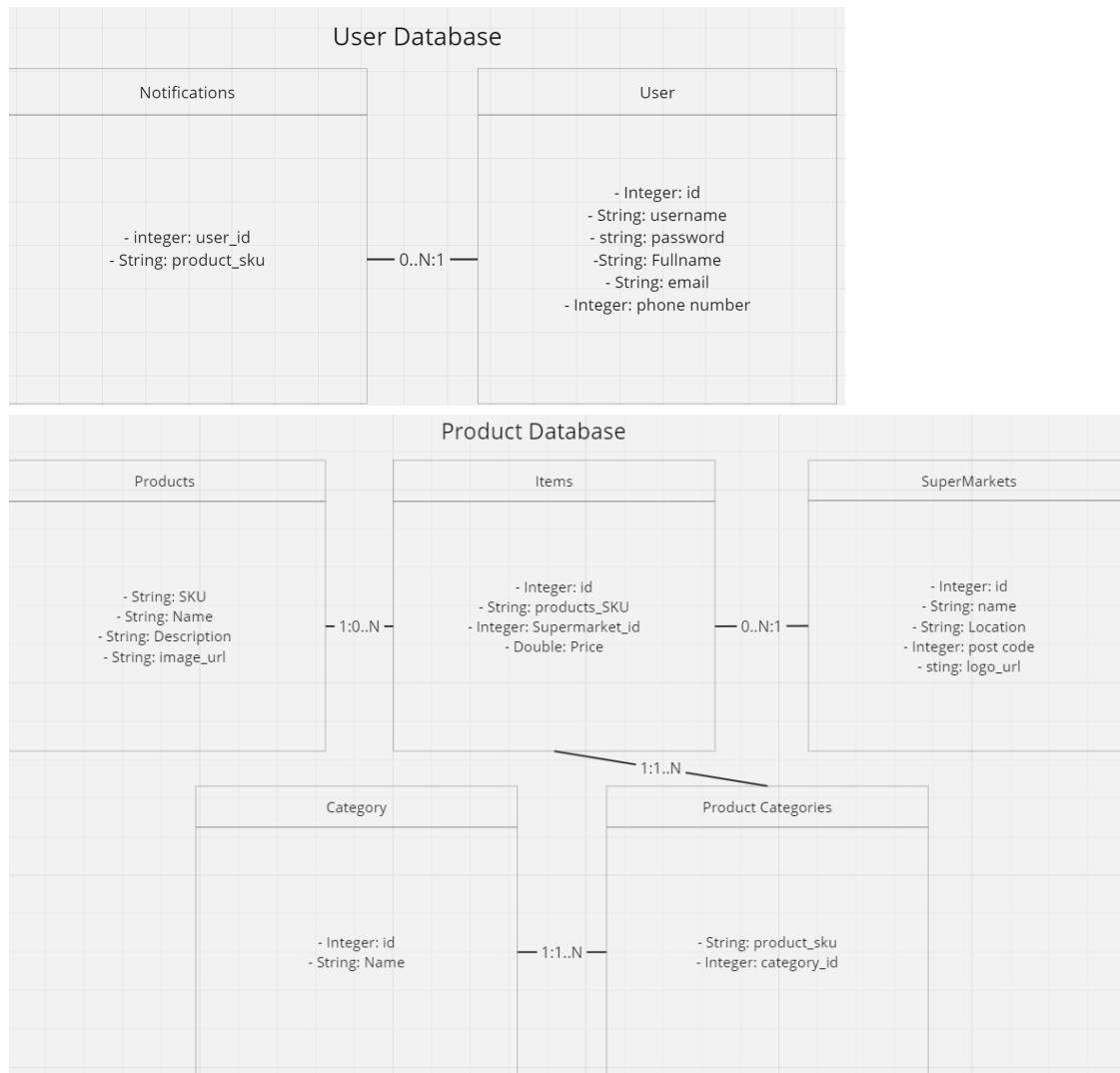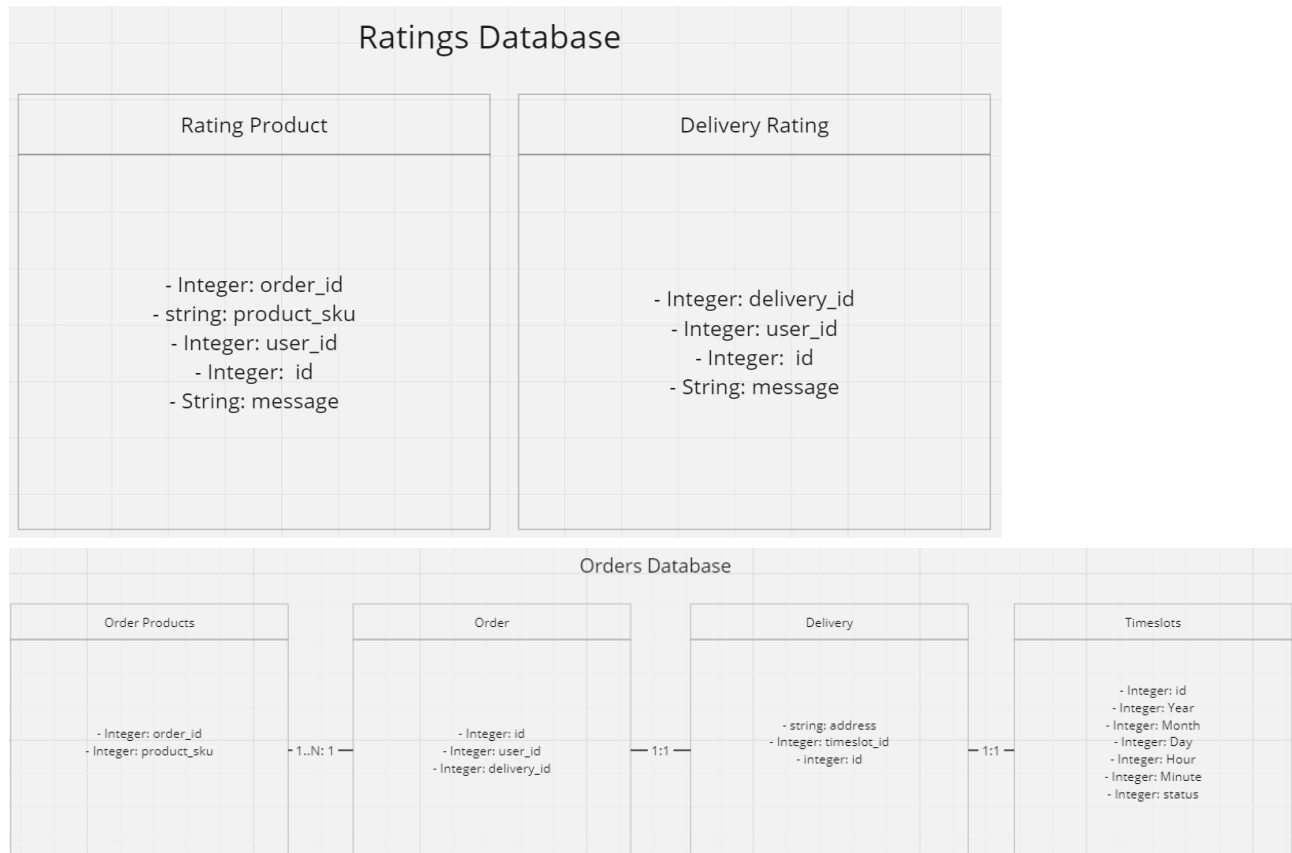
2.5   System Architecture

The system architecture is based on microservices, with the microservices being user management, orders management, ratings management and product management. Each service fulfils a part of the website's functionality

2.6   Data Model

User Database

Notifications		User
- integer: user_id - String: product_sku	— 0..N:1 —	- Integer: id - String: username - string: password -String: Fullname - String: email - Integer: phone number

Product Database

Products		Items		SuperMarkets
- String: SKU - String: Name - String: Description - String: image_url	— 1:0..N —	- Integer: id - String: products_SKU - Integer: Supermarket_id - Double: Price	— 0..N:1 —	- Integer: id - String: name - String: Location - Integer: post code - sting: logo_url

— 1:1..N —

Category		Product Categories
- Integer: id - String: Name	— 1:1..N —	- String: product_sku - Integer: category_id

2.7   Design and Implementation Constraints and Assumption

For this project, there are a few key things we're taking for granted and some dependencies we need to keep in mind:

- *On the data front, we're assuming supermarket databases and pricing APIs will be available for us to tap into. Our real-time info and price matching totally rely on having that data feed.*
- *Users will need a solid internet connection to access real-time data and get timely alerts from the website. We'll have to think through how to handle spotty connections gracefully.*
- *We'll want to lock down partnerships with local stores so we can nail down accurate delivery availability, scheduling, and pricing. The delivery piece hinges on those relationships.*
- *User feedback will be crucial. We'll need people to share their experiences so we can keep making the app better and address any issues that come up.*
- *Adhering to security and privacy regulations is a must. We have to make sure people's data is safeguarded and handled ethically. Trust is key.*
- *Supporting a range of device sizes, specs, and resolutions will be important for accessibility and performance. We'll need to optimise the interface and experience across devices.*
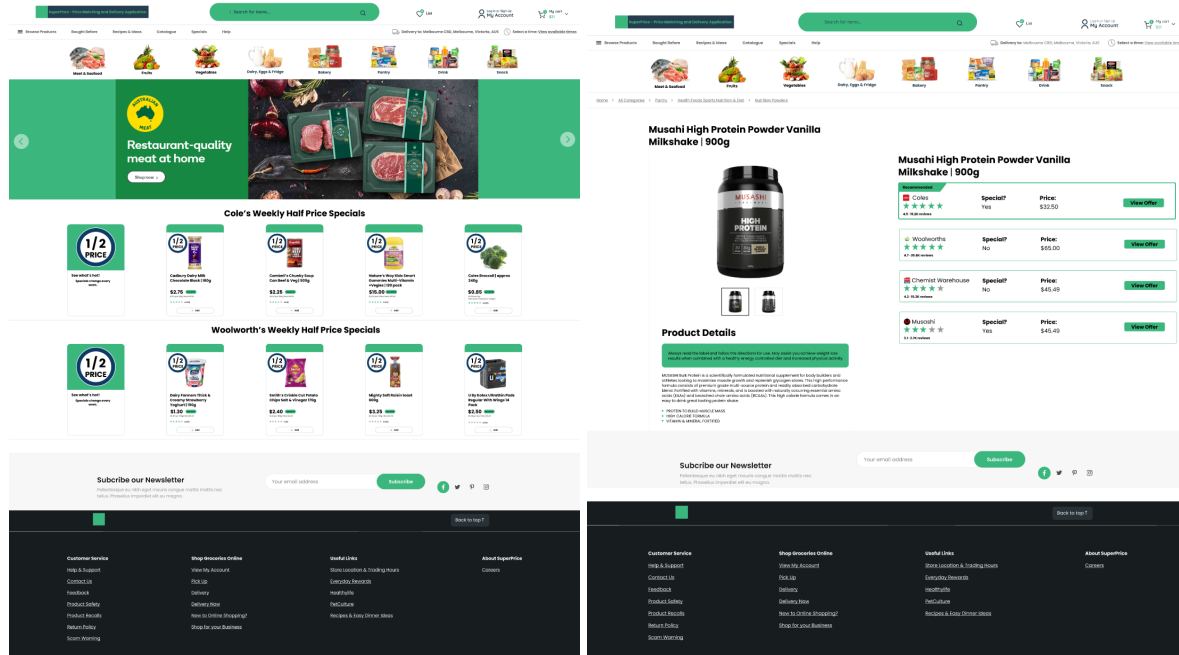
As operating systems are updated, we'll need to watch for changes that could impact features and compatibility. Being proactive will help avoid surprises.

I want us to go into this eyes wide open about potential risks, trade-offs, and dependencies. Please share any others you think of! Addressing this stuff early will set us up for smooth sailing.

3. External Interface Requirements

3.1 User Interfaces (Wireframes)

Wireframes:
https://www.figma.com/file/gsk2j4gWUMgu42oRvx1nID/SEPT?type=design&node-id=63%3A41&mode=design&t=95H7QZsWKn9mWFZY-1

- Keep the colours to a minimum (only choosing 3 main colours) in order to keep the website clean.
- Landing page contains clickable icons to make finding items easier.
- Similar to websites like Coles and Woolworths, the special is shown at the top. In addition, when price checking, the most recommended one is at the top to make users make a quicker decision.
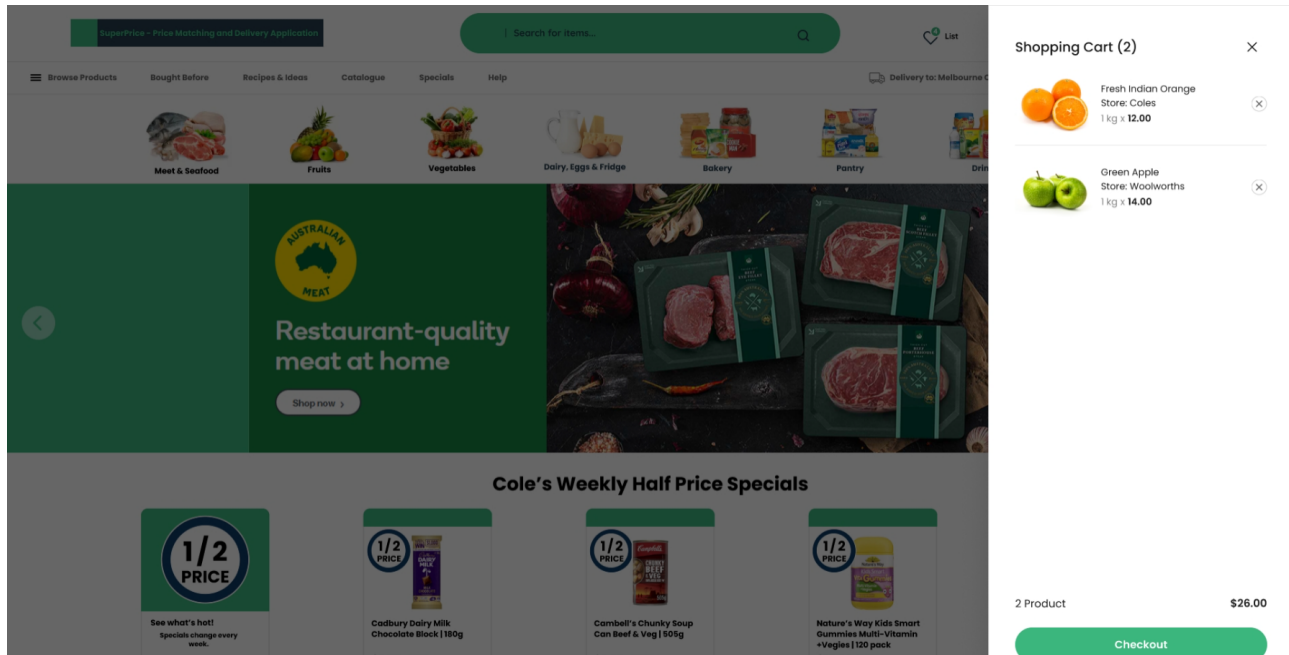
- When pressing on 'My Cart' Button, a side menu bar comes out and fades everything in the back for easier usability.

3.2 Software Interfaces (Dependencies)

- We will be using spring boot for the backend, and react for the front end.
- Since the application is a website, we do not need to worry about which operating system the user is on.
- RESTful Api's will be used to allow communication between the frontend and backend.
- Incoming messages would include data from user registration, product search queries, and completing an order.
- Outgoing messages would include search results from queries, order confirmation/details, and notifications or alerts for the user.

4. Functional Reqs Features

4.1 System Feature 1 Price Comparison

4.1.1 Description and Priority

This feature enables users to compare prices of specific products across different supermarkets in the area. Priority: 9

4.1.2 Stimulus/Response Sequences

- *User selects a product for price comparison.*

- *System retrieves and displays a list of supermarkets with their corresponding prices for the selected product.*
- *User chooses a specific supermarket.*
- *System displays a detailed comparison of prices for the selected product across all available supermarkets.*

4.1.3 Functional Requirements

- REQ-1: The system shall allow users to select a specific product for price comparison.
- REQ-2: The system shall retrieve and display a list of supermarkets along with their prices for the selected product.
- REQ-3: The system shall provide filters to sort the list of supermarkets based on price, location, or user rating.
- REQ-4: The system shall allow users to choose a specific supermarket from the list.
- REQ-5: The system shall display a detailed comparison table showing the prices of the selected product across all available supermarkets.
- REQ-6: The system shall highlight the lowest price for the selected product among the displayed supermarkets.
- REQ-7: If the selected product is not available at a particular supermarket, the system shall indicate "N/A" for that supermarket's price in the comparison table.
- REQ-8: If a user selects a supermarket, the system shall provide additional details about that supermarket, such as its location, user ratings, and delivery options.

4.2 System Feature 2: Delivery Organisation

4.2.1 Description and Priority

This feature allows users to organise deliveries for their selected groceries. Priority: 9

4.2.2 Stimulus/Response Sequences

- *User adds items to their cart and proceeds to checkout.*
- *User selects a delivery option and preferred delivery time slot.*
- *System confirms the selected delivery time slot and displays an estimated delivery window.*
- *User confirms the delivery details and submits the order.*
- *System generates an order confirmation with delivery information.*

4.2.3 Functional Requirements

- REQ-1: The system shall allow users to add items to their cart and proceed to the checkout process.
- REQ-2: The system shall provide multiple delivery options for users to choose from during the checkout process.
- REQ-3: The system shall display available delivery time slots for the chosen delivery date.

- REQ-4: The system shall prevent users from selecting delivery time slots that are already fully booked.
- REQ-5: The system shall provide an estimated delivery window based on the chosen time slot.
- REQ-6: The system shall allow users to review and confirm the selected delivery details before submitting the order.
- REQ-7: If a user encounters an error while selecting a delivery time slot, the system shall provide an error message with guidance.
- REQ-8: After the user submits the order, the system shall generate an order confirmation that includes the chosen delivery time and estimated delivery window.
- REQ-9: The system shall send a notification to the user confirming the order and providing delivery details.
- REQ-10: In case of changes to delivery schedules (e.g., due to unforeseen circumstances), the system shall send timely notifications to users and offer alternative options.

5.   Other Nonfunctional Requirements

5.1   Performance Requirements

Error Handling and Recovery:

- *Requirement: The application shall display error messages to users within 3 seconds of encountering an error condition.*
- *Rationale: Quick error feedback helps users understand and address issues promptly, enhancing their overall experience.*

Notification Delivery Time:

- *Requirement: Notifications for price drops or special offers shall be delivered to users within 1 minute of the event occurrence.*
- *Rationale: Timely notifications are critical for users to take advantage of time-sensitive deals and offers.*

Loading Time:

- *Requirement: The application shall load and display its main interface within 3 seconds of app launch on average.*
- *Rationale: Fast loading times are crucial for user retention. Users are more likely to engage with an application that doesn't keep them waiting.*

5.2   Safety Requirements

The main safety requirement for this project is to ensure there is a secure payment process. The website needs to adhere to the latest security measures in order to safeguard the users from potential financial loss. This requirement could be solved with the integration of certified payment gateway providers such as PayPal known for their industry-leading security standards. The website can prevent fraudulent activities, unauthorised charges, and data breaches during payment transactions by replying on these established providers.

5.3   Security Requirements

User information such as their name, number and email address, need to be stored securely to ensure that the information does not fall prey to malicious actors Users should be prompted to use strong passwords, this would be done by ensuring minimum length for passwords, inclusion of special characters. We will need to ensure that the website is not exposed to malicious attacks, such as SQL injections, user inputs will also need to be validated to ensure that the queries are safe.

5.4   Software Quality Attributes

In addition to meeting the functional requirements, the solution should also adhere to a set of software quality standards that are vital at retaining customers. The website should ensure that both customers and developers have a robust and user-friendly interface.

One of the most important attributes is usability. Customers should be able to complete the most common tasks in a relative time. This website should achieve a user satisfaction rating of at least 85% from doing user surveys.

Reliability is also a crucial software quality attribute. Shoppers won't become regulars if the website is always down or needs fixing. The development team should aim to achieve a system uptime of 99.5% or higher. Scheduled downtime should not exceed 1 hour every month for maintenance. The website should also easily be able to handle concurrent users. A minimum of 1000 simultaneous users without crashing or degrading performance would be optimal.

Portability is another vital attribute which would make ordering groceries much easier for shoppers. The website should be accessible and fully functional on all the latest versions of major web browsers such as Chrome, Safari, and Edge as well as mobile platforms like IOS and Andriod.

Lastly, the website should support the addition of new supermarkets and retailers with minimal code changes and configurations. This would make everything very seamless and allow for rapid expansion.

These qualities reflect a high-performing, reliable and user-friendly website.

5.5   Testing and Acceptance Criteria

Testing

- We will perform unit testing on required functions/components, to ensure that they work as expected, and are able to handle various errors, such as invalid input for search functions.
- Integration testing will also be performed to ensure that components across the website work as expected with each other. Tests include a user checking their order history. This test would check whether there are any errors that arise when a user is viewing their order history.

Acceptance Criteria

- Product Search And Comparison: users should be able to search products by name/category. The results should display relevant products and their information. Users should be able to select and compare the product across different supermarkets.
- User Authentication: users should be able to login and reset their passwords. New users should also be able to create an account
- Delivery: users should be able to select a delivery destination, along with the time they would like their delivery to occur.
- Notifications: if a user has signed up for notifications for a product. They should be messaged price drops for that product on a real time basis. Notifications should include supermarket name, product information and the current discounted price.
- User Reviews/Ratings: users should be able to leave a review/rating for a product from various supermarkets. Users can also be able to rate/review their delivery experience.
- Order Process: users should be able to add items to a cart, and check them out. This process should ask the user for delivery and payment information.