Software Requirements Specification

for

COSC2299 Major Project 2023

Milestone 2 Version

Prepared by

James Czeredys – s3900781

Ned Pearson - s3844470

Rita Lam Cordeiros- 3947881

Raphael Bullen - s3897844

Yash Dhukate - s3898134

Yiqi (Gilbert) Du - s3665887

RMIT

September 2023

Table of Contents

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Introduction

1.1 Purpose

1.2 Intended Audience and Reading Suggestions

1.3 Product Scope

1.4 References

2. Overall Description

2.1 Product Perspective

2.2 Product Functions

2.3 User Classes and Characteristics

2.4 Design and Implementation Constraints

2.5 Assumptions and Dependencies

3. External Interface Requirements

3.1 User Interfaces

4. System Features

5. Other Nonfunctional Requirements

5.1 Performance Requirements

5.2 Security Requirements

5.3 Software Quality Attributes

6. Scrum Artifacts

Appendix A: Analysis Models

Appendix: Milestone 2

Appendix: Milestone 3

# Introduction

## Purpose

By giving customers a solid platform to quickly find the best offers for the things they want to buy, our solution, SuperPrice, is intended to create a website that combines product prices from numerous retail outlets, making it simple for consumers to compare prices and find the best deals. We also provide a delivery service, allowing our customers to have their chosen products delivered to them.

## Intended Audience and Reading Suggestions

This document is intended for developers as well as our product owner. This SRS (Software Requirements Specification) document contains all our wireframes, user stories and planning. It is organized based around the table of contents above. It is recommend reading this document in order or going through the relevant sections based on the table of contents above.

## Product Scope

SuperPrice provides a single platform which allows users to have a cohesive online shopping experience which combines searching, comparing, and purchasing into one service.

## References

Template of SRS:

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

# Overall Description

## Product Perspective

This product is an entirely new, self-contained system. It does not follow-on member of a product family, or a replacement for certain existing systems.

## Product Functions

* Specific product search, display price options for searched items
* View products in chosen category/categories
* Order delivery of products
* Notification/alert system

## User Classes and Characteristics

This software can be widely used by any group of people who have internet access, especially for the following group of people:

* Young people who do not have much time to do the physical shopping.
* Elderly who are not capable of physical shopping.
* People with physical impairments that may hinder their shopping ability.
* People who are trying to maximise the use of their money by shopping for the cheapest products.

## Design and Implementation Constraints

A diagram of software architecture

Description automatically generated

## Assumptions and Dependencies

This colour highlighted text means dependencies added in milestone 2.

This colour highlighted text means dependencies added in milestone 3.

The App is developed based on JDK version 17 with the dependency's versions listed below:

**REACT (FRONT-END) DEPENDINCES**

testing-library/jest-dom : “^5.17.0"

testing-library/react: "13.4.0"

testing-library/user-event: “^13.5.0"

react: “^18.2.0"

react-dom: “^18.2.0"

react-router-dom: “^6.15.0”

react-scripts: “5.0.1"

web-vitals: "^2.1.4"

"axios": "^1.5.0"

**SPRING BOOT (BACK-END) DEPENDENCES**

spring-boot-starter-parent

spring-boot-starter-web

spring-boot-starter-test

h2

spring-boot-starter-jdbc

spring-boot-maven-plugin

flyway-maven-plugin

flyway-mysql

mysql-connector-java

# External Interface Requirements

## User Interfaces

# A screenshot of a computer Description automatically generated

# System Features

|  |  |  |  |
| --- | --- | --- | --- |
| **Story 1:** | Search feature | **Priority** | 1 |
| **Effort** | 9 |
| As a | Online Shopper | | |
| I want | to be able to use a search feature | | |
| So that I can | view specific products | | |
| Acceptance criteria | **Criterion i:**  **Given that I am on the home page of the SuperPrice**  **When I enter a search query in the search bar and press enter**  **Then I should see a list of products matching the search on the results page in a clear and organized manner such that information such as the product name, image, prices, and description is visible.**  **Criterion ii:**  **Given that I have entered a search that does not return any results**  **When I preform the search**  **Then the system should display a message indicating that no matching products were found**  **Criterion iii:**  **Given that I am entering a search query in the search bar**  **When I start typing, auto-suggestions should appear**  **Then I can click on one of the autosuggestions so that I can quickly search for a product**  **Criterion iv:**  **Given that I am on the results page**  **When I click on a product**  **Then the system should present all items in a format that prominently displays the product name, location, and price, ensuring clear visibility and easy comprehension** | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Story 2:** | Category feature | **Priority** | 1 |
| **Effort** | 9 |
| As a | Online Shopper | | |
| I want | to be able to search by category | | |
| So that I can | efficiently browse and discover products that align with my interests and needs | | |
| Acceptance criteria | **Criterion i:**  **Given that I am on the home page of the SuperPrice**  **When I click on one of the available categories**  **Then the system should display all the items associated as well as the prices**  **Criterion ii:**  **Given that products of the category are displayed**  **When I click on one of the available products**  **Then the system should present all items in a format that prominently displays the product name, location, and price, ensuring clear visibility and easy comprehension** | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Story 3:** | Price Sort feature | **Priority** | 3 |
| **Effort** | 6 |
| As a | Online Shopper | | |
| I want | sort the results based on price | | |
| So that I can | efficiently browse and discover products that fit my price budget | | |
| Acceptance criteria | **Criterion i:**  **Given that I have obtained some result from my search**  **When I view the result page**  **Then the system sound presents all items in a format that prominently displays the product name, location, and price, ensuring clear visibility and easy comprehension**  **Criterion ii:**  **Given that I have obtained some result from my search**  **When I click on a sort by button**  **Then the system should display all the items sorted the way I want it either from high to low, or low to high allowing me to compare prices and find the cheapest store** | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Story 4:** | Shopping Cart Feature | **Priority** | 2 |
| **Effort** | 7 |
| As a | Online Shopper | | |
| I want | A shopping cart feature | | |
| So that I can | Add all the products I am searching for, making it convenient to manage and finalise my purchases | | |
| Acceptance criteria | **Criterion i:**  **Given that I am browsing products on the system**  **When I find a product, I want to purchase**  **Then there should be a clear and easily accessible "Add to Cart" button or icon displayed with the product.**  **Criterion ii:**  **Given that I have added products to my shopping cart,**  **When I click on the shopping cart icon or navigate to the cart page,**  **Then I should see a detailed list of all the products I have added, including their names, quantities, and prices.**  **Criterion iii:**  **Given that I am viewing the contents of my shopping cart,**  **When I want to adjust the quantity of a product or remove it,**  **Then the system should provide options to increase, decrease, or delete items, updating the cart total accordingly.**  **Criterion iv:**  **Given that I am viewing the contents of my shopping cart,**  **When I am satisfied with my selections,**  **Then I should have the ability to proceed to checkout, initiating order placement process.** | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Story 5:** | Delivery service feature | **Priority** | 2 |
| **Effort** | 7 |
| As a | Online Shopper | | |
| I want | SuperPrice to offer a delivery service | | |
| So that I can | Have my groceries delivered to my doorstep | | |
| Acceptance criteria | **Criterion i:**  **Given that I have finished adding all my products**  **When I am ready to place the delivery**  **Then I should see a delivery option that allows me to select having the groceries delivered to my doorstep.**  **Criterion ii:**  **Given that I have selected the delivery option for doorstep delivery**  **When I proceed with the checkout process and confirm my order**  **Then I should receive confirmation that my groceries will be delivered to my specified address.**  **Criterion iii:**  **Given that I have finished adding all my products to the cart**  **When I finished with entering my details**  **Then the system should also ask me if when I want the delivery to be made** | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Story 6:** | Notification feature | **Priority** | 3 |
| **Effort** | 6 |
| As a | Online Shopper | | |
| I want | SuperPrice to deliver notifications and alerts | | |
| So that I can | Be informed about price drops and special offers | | |
| Acceptance criteria | **Criterion i:**  **Given that I have selected a product**  **When there is a cheaper alternative available**  **Then the system should send a notification or alert on the platform's interface**  **Criterion ii:**  **Given that I the system has sent a notification**  **When I have viewed it**  **Then I should have the option of closing it** | | |

# Other Non-functional Requirements

## Performance Requirements

Pages should be loaded quickly. When a user clicks a button, the corresponding action should be executed with little delay to minimise the time waiting for the user.

Searches should be executed quickly, and results should be displayed in a timely manner. Efficient methods to access/sort the data, such as using specific SQL queries instead of creating algorithms in the backend/frontend, should be used to reduce strain on the server and to provide search results in a shorter time to the software user.

## Security Requirements

The software should be protected against SQL injection attacks. These pose a threat from users potentially using their own SQL queries or commands by using certain characters in a field that would be used to query the database (potential for deletion or illegal data access), such as the search bar. To prevent this, either a blacklist (block certain characters) or a whitelist (allow certain characters) should be used to ensure no unintended SQL queries can be executed.

## Software Quality Attributes

The software should provide a seamless experience for the end user. The software should be easy to navigate and understand. Colour choices should be made to ensure all elements are visible and clear for users.

Interface elements should be easy to identify and interact with. They should also be in intuitive locations to make the flow of the software easy to follow.

The software will be set up using maven, which can be used for automated testing when building the project. This will ensure any changes made do not break the existing codebase.

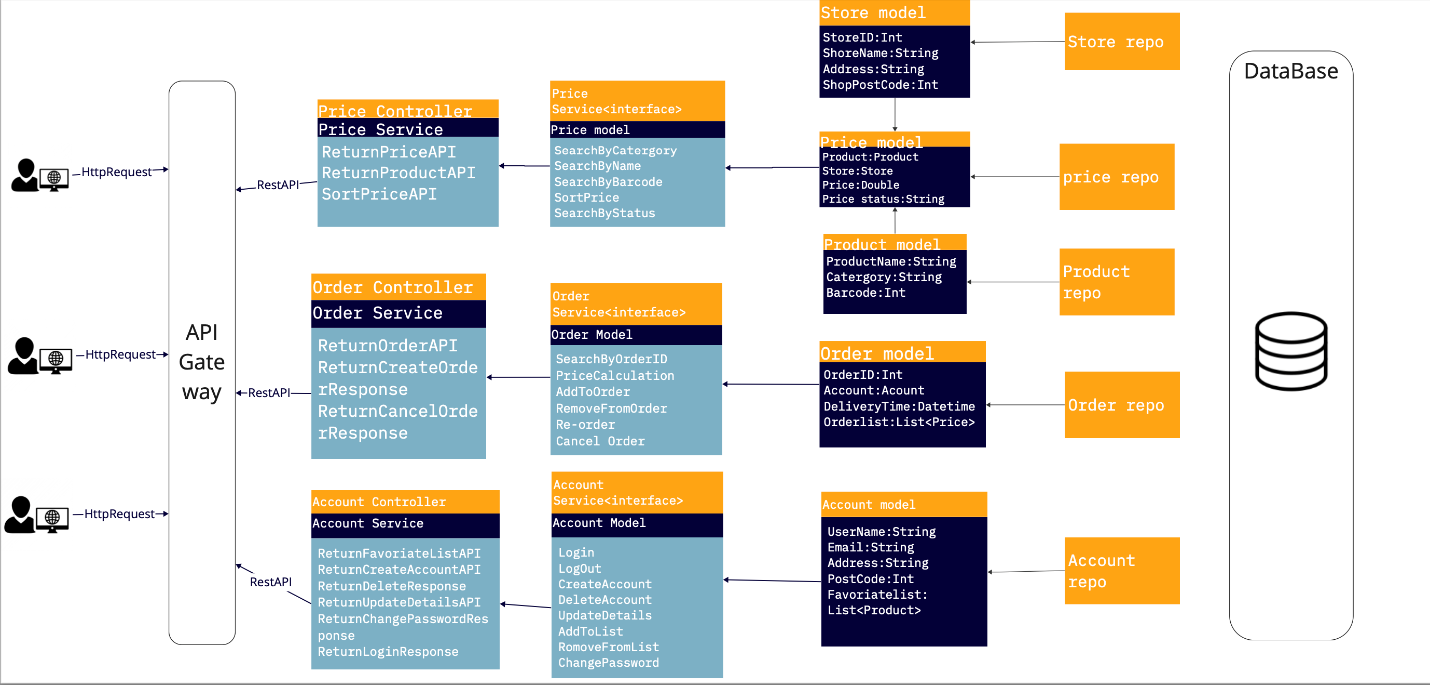
## 6. Scrum Artifacts

Product Backlog, Sprint 1 Backlog, and Sprint 2 Backlog viewable on GitHub board: <https://github.com/orgs/cosc2299-sept-2023/projects/62>

**Sprint 0 Retro**

|  |  |  |
| --- | --- | --- |
| What Went Well? | What didn’t Go Well? | What Can be Improved? |
| Team collaboration – first time most members have worked with each other | Database clarification received was not timely | Assign tasks more specifically ahead of time |
| Frequent team meeting and attendance | Client interaction – did not provide as much guidance as expected | Unify team members vision of product |
| Work finished timely and with high quality |  |  |

Appendix A: Analysis Models



Class Diagram

*A screenshot of a computer

Description automatically generated*

Entity-relationship diagram

Appendix: Milestone 2

**Contribution Statement**

|  |  |  |
| --- | --- | --- |
| **Name** | **Contribution %** | **Items contributed** |
| James | 16.6% | Front-end & back-end integration, continuous integration, documentation |
| Ned | 16.6% | Front-end |
| Rita | 16.6% | Front-end |
| Raphael | 16.6% | Back-end unit tests, back-end debugging |
| Yash | 16.6% | Front-end HTML, CSS and React |
| Gilbert | 16.6% | Back-end implementation, DB schema |

**Links to videos**

**James:** [James Czeredys - s3900781 - COSC2299-23s2 M2.mov](https://rmiteduau-my.sharepoint.com/:v:/r/personal/s3900781_student_rmit_edu_au/Documents/James%20Czeredys%20-%20s3900781%20-%20COSC2299-23s2%20M2.mov?csf=1&web=1&e=8LsHwQ&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZyIsInJlZmVycmFsQXBwUGxhdGZvcm0iOiJXZWIiLCJyZWZlcnJhbE1vZGUiOiJ2aWV3In19)

**Ned:** [EdwardPearsonS3844470 COSC2299-23s2 PTC1.mp4](https://rmiteduau-my.sharepoint.com/:v:/r/personal/s3844470_student_rmit_edu_au/Documents/EdwardPearsonS3844470%20COSC2299-23s2%20PTC1.mp4?csf=1&web=1&e=uLs18r&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZyIsInJlZmVycmFsQXBwUGxhdGZvcm0iOiJXZWIiLCJyZWZlcnJhbE1vZGUiOiJ2aWV3In19)

**Raphael:** [Raphael Bullen - s3897844 - COSC2299-23s2 MPM2.webm](https://rmiteduau-my.sharepoint.com/:v:/r/personal/s3897844_student_rmit_edu_au/Documents/Raphael%20Bullen%20-%20s3897844%20-%20COSC2299-23s2%20MPM2.webm?csf=1&web=1&e=SEtmUt&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZyIsInJlZmVycmFsQXBwUGxhdGZvcm0iOiJXZWIiLCJyZWZlcnJhbE1vZGUiOiJ2aWV3In19)

**Yash:** [Recording-20230917\_091051.webm](https://rmiteduau-my.sharepoint.com/:v:/r/personal/s3898134_student_rmit_edu_au/Documents/Recording-20230917_091051.webm?csf=1&web=1&e=HaW3Hm&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZyIsInJlZmVycmFsQXBwUGxhdGZvcm0iOiJXZWIiLCJyZWZlcnJhbE1vZGUiOiJ2aWV3In19)

**Gilbert:** [Recording-20230916\_213244.webm](https://rmiteduau-my.sharepoint.com/:v:/r/personal/s3665887_student_rmit_edu_au/Documents/Recording-20230916_213244.webm?csf=1&web=1&e=0r8nzl&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZyIsInJlZmVycmFsQXBwUGxhdGZvcm0iOiJXZWIiLCJyZWZlcnJhbE1vZGUiOiJ2aWV3In19)

**Rita:** [Rita-s3947881-COSC2299-23s2.mp4](https://rmiteduau-my.sharepoint.com/:v:/r/personal/s3947881_student_rmit_edu_au/Documents/Rita-s3947881-COSC2299-23s2.mp4?csf=1&web=1&e=QaDr5o&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZyIsInJlZmVycmFsQXBwUGxhdGZvcm0iOiJXZWIiLCJyZWZlcnJhbE1vZGUiOiJ2aWV3In19)

**Links to GitHub Repository and Board**

GitHub repo:

<https://github.com/cosc2299-sept-2023/team-project-group-p07-06>

Board:

<https://github.com/orgs/cosc2299-sept-2023/projects/62>

Continuous Integration

In */.github/workflows/* we have *superPrice-ci.yml* file which is used for our continuous integration. Every time *main* or a specified branch is pushed the following is executed:

* name: Build and test with Maven
* run: *./mvnw --batch-mode --update-snapshots test package*
* name: Install dependencies
* run: *npm install*
* name: Run tests
* run: *npm test*
* name: Build
* run: *npm run build*

We also use “act,” which is a GitHub Action used to run GitHub workflows locally before pushing changes to your repository, allowing us to test out continuous integration without risking pushing changes to our repository.

Screenshots

Building frontend using “act”

A screenshot of a computer

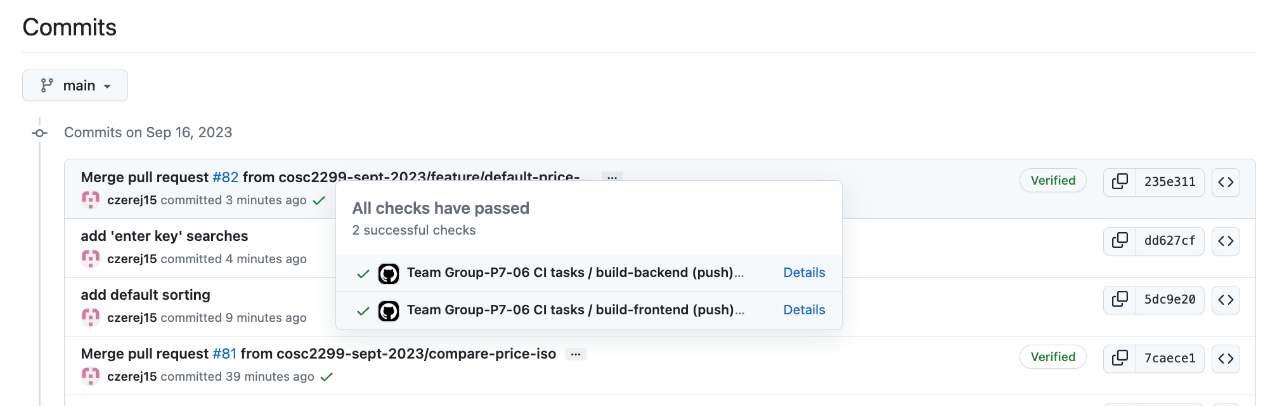
Description automatically generated

Building backend using “act”

A screenshot of a computer program

Description automatically generated

Automated CI in GitHub example



**Testing**

Test files located in: /backend/SuperPrice/src/test/java/superPrice

**Controller Tests:**

* /controllers
  + StorePriceControllerTests.java

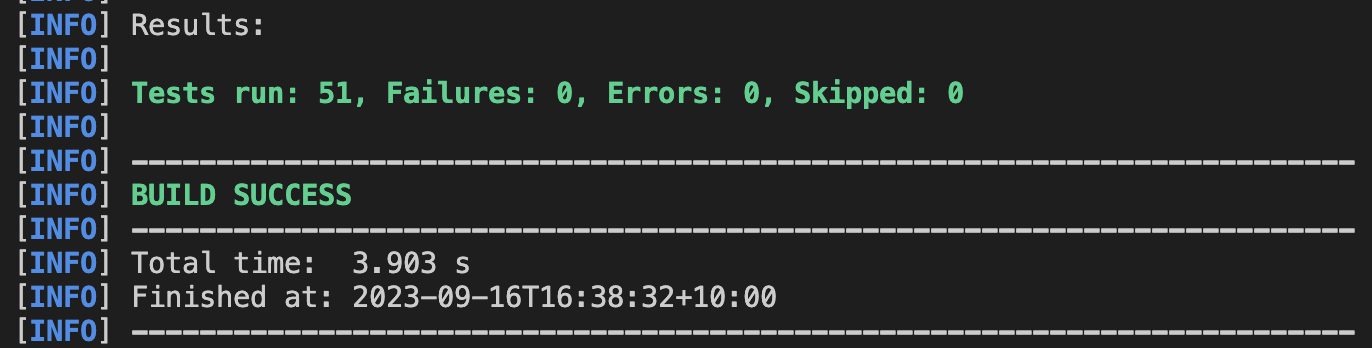
**Model Tests:**

* /models
  + PriceTests.java
  + ProductTests.java
  + StorePriceTests.java
  + StoreTests.java

**Repository Tests:**

* /repositories
  + PriceRepositoryImplTests.java
  + ProductRepositoryImplTests.java
  + StoreRepositoryImplTests.java

**Test execution and results**



Updated Dependencies

In section *2.5 Assumptions and Dependencies*, Highlighted text shows dependencies added in milestone 2.

New Features

* Search products
* Browse products by category
* Compare prices of a product

**Meeting Minutes**

***Sprint 1- Week 1 Notes***

* Assigned Areas:
  + Gilbert: back-end
  + Yash, Ned, Rita: front-end
  + James: back-end and front-end
  + Raphael: testing
* Learn to use react, html, CSS
* Start working on backend
* Become familiar with JUnit testing
* Populate database

***Sprint 1- Week 2 Notes***

* Start writing tests
* Continue working on back-end
* Agree on API endpoints needed by front-end
* Convert pure html and CSS of homepage into react
* Solve database migration issues
* End of week aim to finish features
  + Product search
  + Category search

***Sprint 1- Week 3 Notes***

* Implement continuous integration (refer to Web App Coding Notes)
* Implement GitHub actions to test CI
* Merge branches to main
* Implement default sorting based on price
* Make sure project board up to date
* End of week aim to finish features
  + Product price comparison
* Submission
  + Fill out contribution sheet and record videos
  + Receive clarification on video submission process
  + Upload docs to repo

**Sprint 1 Retro**

|  |  |  |
| --- | --- | --- |
| What Went Well? | What didn’t Go Well? | What Can be Improved? |
| Very little merge conflicts | Database migration was more difficult than expected | Creating more branches |
| Frequent and productive team communication |  | Branch naming |
| Work finished timely and with high quality |  | Task to sprint distribution |

Appendix: Milestone 3

**Contribution Statement**

|  |  |  |
| --- | --- | --- |
| **Name** | **Contribution %** | **Items contributed** |
| James | 20% | Front-end & back-end integration, front-end testing, continuous integration, documentation |
| Ned | 20% | Front-end UI and functionality |
| Rita | 0% | - |
| Raphael | 20% | Backend testing, backend functionality |
| Yash | 20% | Front–end UI and functionality |
| Gilbert | 20% | Backend, backend testing |

**Links to videos**

**James:** [James Czeredys - s3900781 - GROUP-P7-06 COSC2299-23s2 Milestone 2.mov](https://rmiteduau-my.sharepoint.com/:v:/r/personal/s3900781_student_rmit_edu_au/Documents/James%20Czeredys%20-%20s3900781%20-%20GROUP-P7-06%20COSC2299-23s2%20Milestone%202.mov?csf=1&web=1&e=88DpBZ&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZyIsInJlZmVycmFsQXBwUGxhdGZvcm0iOiJXZWIiLCJyZWZlcnJhbE1vZGUiOiJ2aWV3In19)

**Ned:** [EdwardPearsonS3844470 COSC2299-23s2 Milestone3.mp4](https://rmiteduau-my.sharepoint.com/:v:/g/personal/s3844470_student_rmit_edu_au/EVLotjhwhtJFlN-f6FWr54MBgjNZYMCkNY0SMb_KaYa1-Q?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZyIsInJlZmVycmFsQXBwUGxhdGZvcm0iOiJXZWIiLCJyZWZlcnJhbE1vZGUiOiJ2aWV3In19&e=xlo92D)

**Raphael:** [GROUP-P7-06 COSC2299-23s2 Milestone 3.webm](https://rmiteduau-my.sharepoint.com/:v:/g/personal/s3897844_student_rmit_edu_au/ETDBMXACyiVHu_IG7USjU3MBdqVzIW2mImbQL7-8HmBIDQ?nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZyIsInJlZmVycmFsQXBwUGxhdGZvcm0iOiJXZWIiLCJyZWZlcnJhbE1vZGUiOiJ2aWV3In19&e=2fdtFU)

**Yash:** [Recording-20231008\_175027.webm](https://rmiteduau-my.sharepoint.com/:v:/r/personal/s3898134_student_rmit_edu_au/Documents/Recording-20231008_175027.webm?csf=1&web=1&e=7nzVfd&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZyIsInJlZmVycmFsQXBwUGxhdGZvcm0iOiJXZWIiLCJyZWZlcnJhbE1vZGUiOiJ2aWV3In19)

**Links to GitHub Repository and Board**

GitHub repo:

<https://github.com/cosc2299-sept-2023/team-project-group-p07-06>

Board:

<https://github.com/orgs/cosc2299-sept-2023/projects/62>

**Meeting Minutes**

***Sprint 2- Week 1 Notes***

* Finish working on cart functionality
* Continue working on payment page
* Start working on and agreeing on notification system

***Sprint 2- Week 2 Notes***

* Add Input validation for payment page
* Make it so groups same items instead of individually listing each for cart
* Work on docker compose
* Go to lecture/consultation class to resolve docker issues
* Check test coverage
* Add front-end testing
* Change database to MySQL
* Work on view orders page

***Sprint 2- Week 3 Notes***

* Save order details to backend
* Add more data (specifically groceries) to database
* Fix bug - having to select date twice to get to checkout
* Complete refactoring report
* Do milestone 3 documentation such as project report
* Finalise project board
* Everyone needs to fill contribution sheet and upload video

**Sprint 2 Retro**

|  |  |  |
| --- | --- | --- |
| What Went Well? | What didn’t Go Well? | What Can be Improved? |
| Effective meetings and collaboration | Didn’t get time to do login/logout extra feature | Creating more branches |
| Frequent and productive team communication | Deploying to AWS | Branch naming conventions |
| Completed user stories | Setting up docker compose with MySQL | Attending more consultations |

Updated Dependencies

In section *2.5 Assumptions and Dependencies*, highlighted text shows dependencies added in milestone 3.

New Features

* Added shopping cart
* Added more items (specifically groceries) to database
* Popup notification for special offers
* Added payment checkout (Custom Feature)
* Can view orders (Custom Feature)

Highlighted text means extra features, features they were not originally required by the user but nevertheless features we went above and beyond to add.

**Completed Custom User story**

This is a feature that was not present in our initial stories, yet will still completed it in addition.

|  |  |  |  |
| --- | --- | --- | --- |
| **Story 3:** | View Order feature | **Priority** | 4 |
| **Effort** | 6 |
| As a | Online Shopper | | |
| I want | To view the details of my order | | |
| So that I can | I can double check information (such as delivery date) | | |
| Acceptance criteria | **Criterion i:**  **Given that I am on the view orders page**  **And entered my order id**  **When I view the search button**  **Then I should see the details of my corresponding order correctly listed.**  **Criterion ii:**  **Given that I am on the view orders page**  **And entered an incorrect order id**  **When I view the search button**  **Then I should see an error messaging informing me that order could not be found.** | | |

**Refactoring Report**

**1. Refactoring the code design from a package-based design pattern to a Micro-service MCV design pattern.**

**Before:** the code structure design was model package based: package -> repo, service, controller

**After:** the code structure design is Micro-service based: Micro-service -> model, controller, service, repo.

**2. Adding different error messages regarding different issues from the same endpoint.**

When the error is caused by the order item information provided, the backend will throw invalidAttributesException with the message “wrong order information provided”

When the error is caused by the order delivery address or delivery type, the message will be “invalid delivery information”

When the error is caused by the order delivery time the message will be “invalid delivery date”

**3. Refactoring logic in different layers.**

**Before:** Add order and Items is a method in the order repository that will add the order and all the items into the database. However, this method has two different tables (Order and order\_product\_store) involved and if any issue happens during adding the data, this method also needs to call delete order.

**After:** splitting this method to add order, add items and delete order, these three methods only related to the relevant table in the database respectively. Implementing the add order and its items and deleting the order if anything goes wrong when adding items in service layer, so the repo only works for the database communication, and the business logic is handled by service layer.

**4. Clean up some unnecessary code and add some small performance improvement.**

**Project Report**

Our vision is a world where grocery shopping is a convenient, cost-effective, and enjoyable experience for everyone. Our product, is born out of the belief that modern technology can transform the way people shop for groceries, save them valuable time and money, and simplify their lives.

Our application is valuable because it provides real-time access to an extensive database of products and prices, allowing users to instantly compare prices across various supermarkets. We enable our users to stretch their budgets further. We provide the ultimate convenience by delivering right to our customer’s doorstep. SuperPrice also keeps users in the loop about special offers, through timely notifications and alerts. SuperPrice prioritizes user-friendliness and accessibility. Our user-friendly interface caters to shoppers of all backgrounds, making grocery shopping a breeze for both tech-savvy individuals and those less familiar with digital platforms.

A diagram of software architecture

Description automatically generated

**Git commits**

In total we had around 190 commits in our git repository.

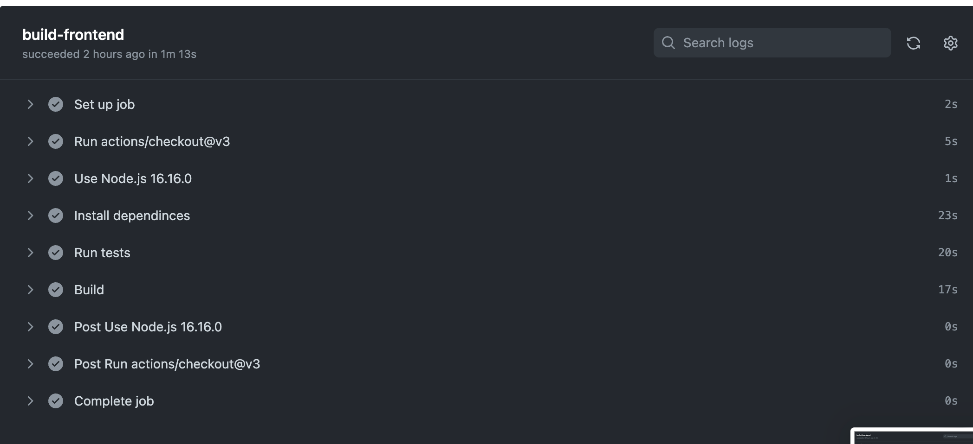
**SCRUM process**

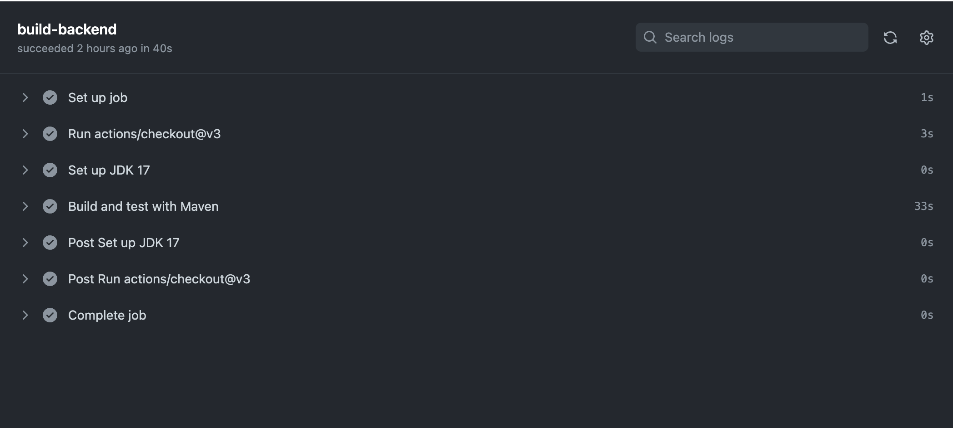
Scrum master: James Czeredys – s3900781@student.rmit.edu.au

We meet roughly twice a week. We often had calls, either as a team or between two people to work through issues and take part in pair programming.

**Continuous Integration Pipeline**

Our GitHub Actions workflow for frontend and backend:





**Amazon Web Services**

We pushed our images to the ECR (under student s3900781 – James Czeredys).

Frontend image Backend image

