Software Requirements Specification for Grocery Spending Tracker: subtitle describing software

Team 1, JARS
Jason Nam
Allan Fang
Ryan Yeh
Sawyer Tang

October 5, 2023

Contents

Revision History

Date	Version	Notes
Date 1	1.0	Notes
Date 2	1.1	Notes

1 Purpose of the Project

1.1 User Business

With the world currently facing record high inflation, cost-of-living is at the highest it has ever been. This affects all daily necessities but is especially true for food and groceries. As a whole, all households are affected but there is particular financial strain on those with lower-incomes. As a result, interest in personal finance has grown and become more important in peoples' everyday lives. To assist these individuals, we are developing an application that can help users better understand their spending habits and make smarter financial decisions. This application will allow users to take photos of grocery receipts and track their overall spending, analyze spending trends, and receive suggestions on cheaper alternatives for purchased grocery items. Overall, we believe this application will help users stay more informed and reduce grocery spending in the long-term.

1.2 Goals of the Project

- The created application will help users save money on groceries over time.
- The application will provide accurate spending data and suggestions to end users.

2 Stakeholders

2.1 Client

Both the client and customer will be the users of the system. These primary stakeholders are made up of low-income households of Hamilton and university students from McMaster who want to save money on groceries. The stakeholders will make decisions that influence the system through surveys, studies, and focus groups.

2.2 Customer

Outlined in the above Client section.

2.3 Other Stakeholders

- Grocery stores in the area
- Household members (people who the groceies are being purchased for)
- Mobile application store (ios app store, google play store)
- Any invidividual who wants to save on groceries

2.4 Hands-On Users of the Project

Low income households and individuals who are budget constrained on their groceries. The archetypal customer will have access to a device with a camera and have access to the internet. This entails adults or young adults who are responsible for making decisions on grocery items. Customers want to reduce the cost of their grocery items whilst maximizing the convenience of their grocery shopping trip.

- Low or single income households in Hamilton
  - Shops for household groceries.
  - Moderate experience with navigation mobile applications.
  - 30-70 years of age.
  - Has vehicle or some means to travel.
- Students of McMaster University
  - Shops for individual groceries
  - Adept experience with navigation mobile applications.
  - 20-30 years of age.
  - Has no vehicle or limited means to travel.

2.5 Personas

- John Bertuzzi
  - Age: 22
  - Favorite food: Chicken and Pasta

  - Nursing Student at McMaster University
  - Living in Ainslie Wood East.
  - Athlete.
  - Missing teeth.

- Alice Woll
  - Age: 51
  - Favorite food: Grilled Cheese
  - Mother of 2, Spouse of William Woll.
  - Accountant for tech company.
  - Living in Westdale.
  - Likes to watch Hockey.

2.6 Priorities Assigned to Users

All users will have equal priority within the system.

2.7 User Participation

Everyday users will provide their shopping data to the community database. This will be of no additional time commitment than they are already required to use the system at a base level.

From an elicitation perspective, a sample of users will be responsible for providing usability, look, and feel requirements feedback. This will be done at various stages in development.

2.8 Maintenance Users and Service Technicians

*** N/A

3 Mandated Constraints

3.1 Solution Constraints

Insert your content here.

3.2 Implementation Environment of the Current System

Insert your content here.

3.3 Partner or Collaborative Applications

Insert your content here.

3.4 Off-the-Shelf Software

Insert your content here.

3.5 Anticipated Workplace Environment

Insert your content here.

3.6 Schedule Constraints

Insert your content here.

3.7 Budget Constraints

Insert your content here.

3.8 Enterprise Constraints

Insert your content here.

4 Naming Conventions and Terminology

4.1 Glossary of All Terms, Including Acronyms, Used by Stakeholders involved in the Project

Insert your content here.

5 Relevant Facts And Assumptions

5.1 Relevant Facts

Insert your content here.

5.2 Business Rules

Insert your content here.

5.3 Assumptions

Insert your content here.

6 The Scope of the Work

6.1 The Current Situation

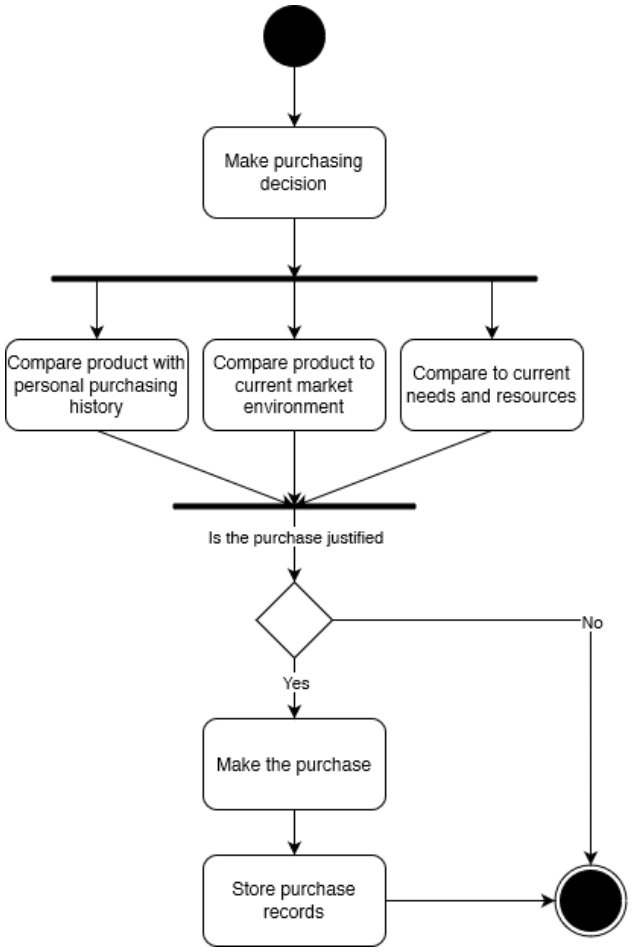

Figure 1: Current Situation Activity Diagram

6.2 The Context of the Work

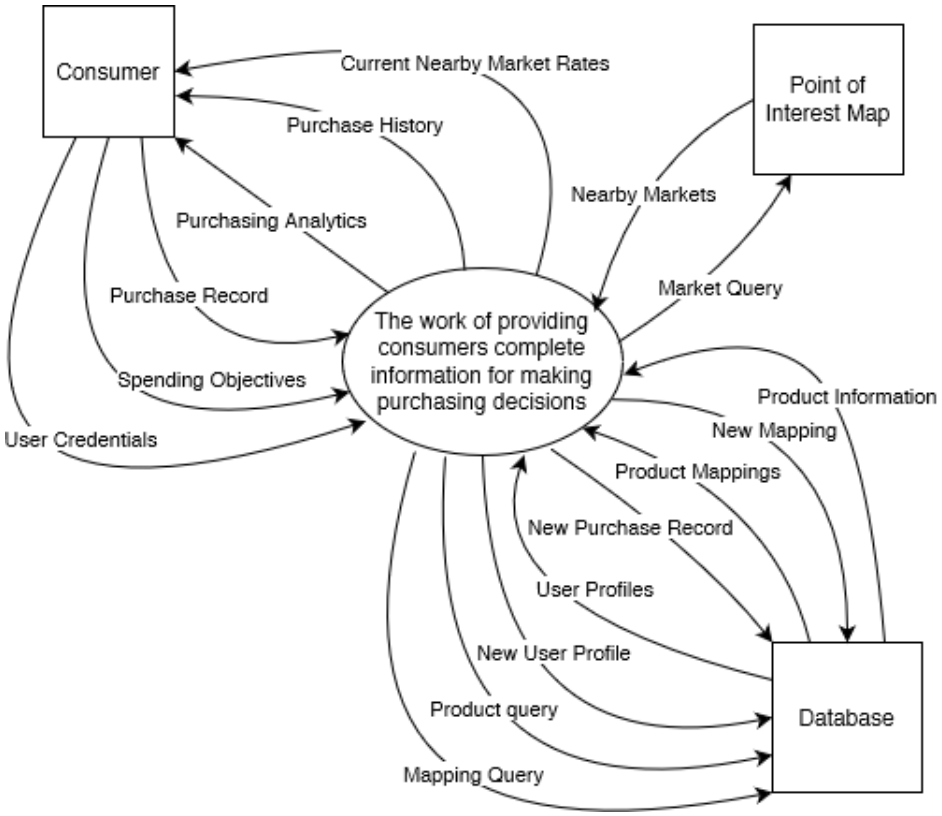

Figure 2: Work Context Model

6.3 Work Partitioning

Event Name	**Input/output**	**Summary of BUC**
User inputs user profile credentials	User Credentials (in)	Initialize user profile from received credentials
Consumer inputs spending objectives	Spending objectives (in)	Records the objectives and the owner

Consumer inputs purchase record	Purchase record (in)	Records the purchase and the owner of the purchase
User requests purchasing analytics	Purchasing analytics (out)	Report compiled and computed information from available data for the user profile
User requests personal purchase history	Purchase history (out)	Report compiled user purchasing history
User requests information on a product	Current nearby market rates (out)	Report market rates for a product that is accessible to the user
Compiling market information relevant to a user	Nearby markets (in)	Record information for accessible markets based on criteria
Request information on markets	Market query (out)	Send request for market information under a set of specified parameters
Database transmits product information	Product information (out)	Record information on product
New mapping received	Mapping Data (out)	Send new good/service mapping to database
Database transmits requested mapping data	Product mappings (in)	Record the product mappings
Send purchase record to database	New purchase record (out)	Send purchase record to database
Database transmits requested user profile data	User profiles (in)	Record the user profile data
Send user profile data	New user profile (out)	Send a user profile to database
Send product query to database	Product query (out)	Request relevant data based on product query from database

Send mapping query to database	Mapping query (out)	Request relevant data based on mapping query from database

Table 1: Business Event List

6.4 Specifying a Business Use Case (BUC)

Title: Create a new profile
Trigger: User creates a new profile
Pre-condition: User has application running
Outcome:

1. User creates an profile identified by a username and a password
2. System creates new profile entry in database

Title: Log in to user profile
Trigger: User submits profile credentials
Pre-condition: Profile exists in database
Outcome:

1. System checks if profile credentials exists
2. If exists, system allows loads user profile
3. If not exists, systems shows error message

Title: Record user objectives
Trigger: User submits personal objectives
Pre-condition: User is logged into profile
Outcome:

1. System stores user budgeting objectives of budget with associated product or product groups

Title: Input purchase records
Trigger: User submits image or manual entry of purchase record
Pre-condition: User is logged into profile
Outcome:

1. System reads and translates records to Strings
2. System maps input records to database objects
3. If mapping does not exist, make guesses and prompt user for feedback
4. Save records to database

Title: Create new mapping feedback
Trigger: Product mapping is not found in database
Pre-condition: Product record has been submitted
Outcome:

1. Create mapping guesses using language model
2. Prompt user with guesses for feedback
3. User inputs feedback
4. Process and screen user feedback
5. If mapping is eligible, create new mapping entry in database

Title: Create user profile analytics
Trigger: User requests purchasing analytics
Pre-condition: User is logged into profile
Outcome:

1. Retrieve user data from database
2. Transform user profile data into graphics
3. Display graphics in application

Title: Retrieve profile purchase history
Trigger: User requests purchase history
Pre-condition: User is logged into profile
Outcome:

1. Retrieve user purchase history from database
2. Transform user purchase history into graphics
3. Display graphics in application

Title: Retrieve product information
Trigger: User requests product information
Pre-condition: User is logged into profile and has current location information set
Outcome:

1. Retrieve user purchase history from database relevant to the product
2. Retrieve markets accessible to the user from map service
3. Retrieve product information belonging to the accessible markets from database
4. Transform data into graphics
5. Display graphics in application

Title: Set location information
Trigger: User sets location information
Pre-condition: User is logged into profile and location information is available
Outcome:

1. User sets initial location
2. User sets accessibility radius
3. Information is stored locally on device

7 Business Data Model and Data Dictionary

7.1 Business Data Model

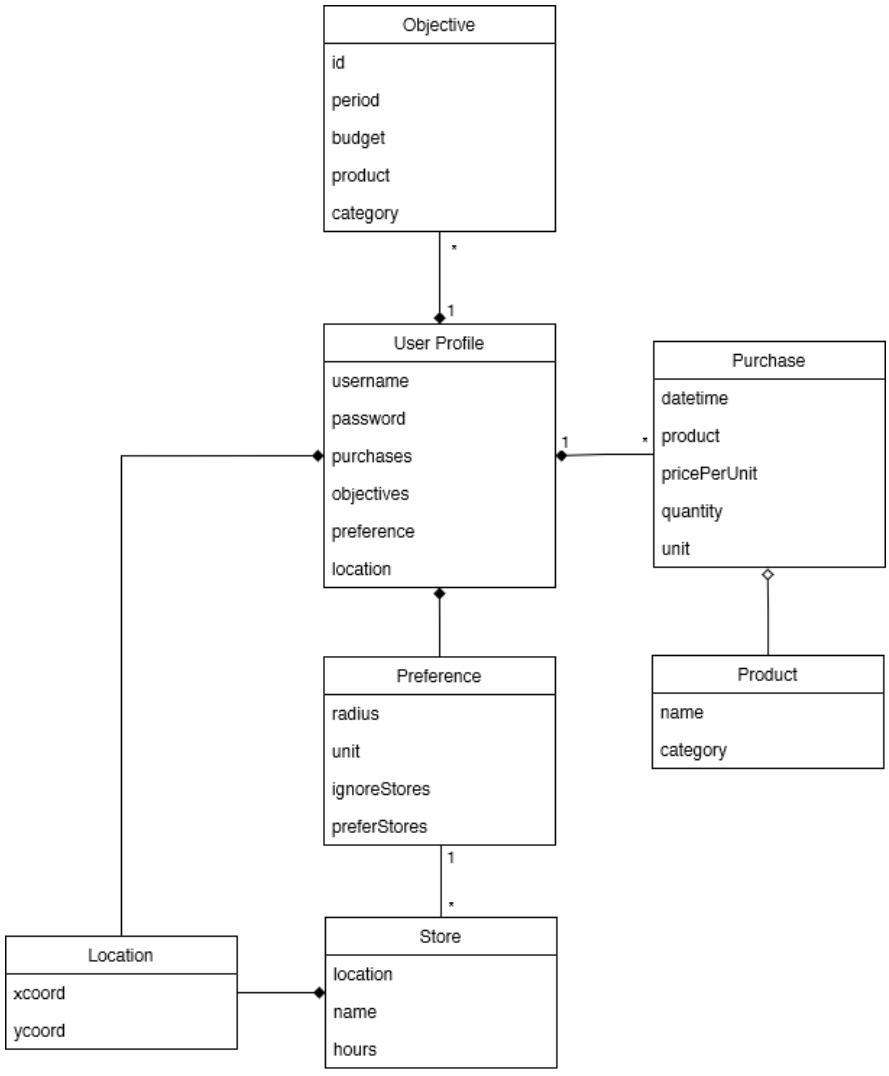

Figure 3: Class Diagram

7.2 Data Dictionary

Name	Content	Type
User Profile	Username + Password + Purchases + Objectives + User Preference + User Location	Class
Objective	Objective Id + Period + Budget + Target Product + Target Category	Class
Purchase	Datetime + Purchased Product + Price Per Unit + Quantity + Unit	Class
Product	Product Name + Category	Class
Preference	Accessible Radius + Ignore Stores + Prefer Stores + Prefer Units	Class
Store	Store Location + Store Name	Class
Location	X-Coordinate + Y-Coordinate	Class
Username	*Unique String*	Attribute
Password	*String*	Attribute
Purchases	*List of Purchase class*	Attribute
Objectives	*List of Objective class*	Attribute
User Preference	*Preference class*	Attribute
User Location	*Location class*	Attribute
Objective Id	*String*	Attribute
Period	*String: day/week/month/year*	Attribute
Budget	*Currency amount measured by Double to 2 decimals*	Attribute
Target Product	*String product name*	Attribute
Target Category	*String category name*	Attribute
Datetime	*YYYY-MM-DD HH:MM:SS*	Attribute
Purchased Product	*String Product name*	Attribute
Price Per Unit	*Int price per unit of product*	Attribute
Unit	*String*	Attribute
Quantity	*Double*	Attribute
Product Name	*String*	Attribute
Category	*String*	Attribute

Accessible Radius	*Integer*	Attribute
Ignore Stores	*List of Stores to ignore*	Attribute
Prefer Stores	*List of Stores to prioritise*	Attribute
Prefer Units	*String: metric/imperial*	Attribute
Store Location	*Location class*	Attribute
Store Name	*String*	Attribute
X Coordinate	*Double*	Attribute
Y Coordinate	*Double*	Attribute

Table 2: Business Event List

8 The Scope of the Product

8.1 Product Boundary

Figure 4: Use Case Diagram

8.2 Product Use Case Table

PUC No	PUC Name	Actor/s	Input/Output
1	Register for an account	User, System	User account data (in), Created user account (out)
2	Sign into application	User, System	User account data (in), Sign in authentication (out)
3	Enter spending goals	User	User spending preferences (in)
4	Enter location preferences	User	User location preferences (in)
5	Enter budgeting preferences	User	User budgeting preferences (in)
6	Enter receipt data	User	Photo of receipt (in)
7	Record user inputs	System	Inputs stored in database (out)
8	Process receipt	System	Receipt data stored in database (out)
9	Map receipt data to database	System	Individual product data properly stored in database (out)
10	Display analyzed spending data	System	Visualized spending data (out)
11	Provide budgeting suggestion	System	Budgeting suggestion (out)
12	Suggest cheaper grocery alternatives	System	Item, price, and location of cheaper alternative (out)

Table 3: Product Use Case Table

8.3 Individual Product Use Cases (PUC's)

Title: Register for an account
Trigger: User wants to create an account for the application
Pre-condition: User has necessary information requested by the system
Outcome:

1. User enters their email and password when prompted and submits the data.
2. System validates the data.
3. If valid, system stores user data in the database and creates the account.
4. If invalid, system prompts user with an error.

Title: Sign into application
Trigger: User wants to sign into their account
Pre-condition: User has an account for the application
Outcome:

1. User enters their email and password when prompted and submits the data.
2. System looks for any accounts with matching credentials.
3. If a match is found, system authentication is successful and user is granted application access.
4. If no match is found, system authentication fails and user is prompted with an error.

Title: Enter spending goals
Trigger: User wants to enter their spending goals for the application
Pre-condition: User is signed into an account
Outcome:

1. User enters information regarding their spending objectives and outcomes for application.
2. System records the entered spending goals into the database.

Title: Enter location preferences
Trigger: User wants to enter their location preferences for the application to consider
Pre-condition: User is signed into an account
Outcome:

1. User enters information regarding their current location and preferred area of coverage for the app.
2. User selects stores they prefer and stores they want to ignore.
3. System records the entered location preferences into the database.

Title: Enter budgeting preferences
Trigger: User wants to enter their budgeting preferences for the application to consider
Pre-condition: User is signed into an account
Outcome:

1. User enters price range and budgeting preferences for the application to consider.
2. System records the entered budgeting preferences into the database.

Title: Enter receipt data
Trigger: User wants to enter a receipt to add new spending data
Pre-condition: User is signed into an account and has a receipt to use
Outcome:

1. User takes a photo of the receipt.
2. System processes data from the receipt *(see Individual Product Use Case for "Process receipt")* and stores data in the database *(see Individual Product Use Case for "Map receipt data to database")*.

Title: Process receipt
Trigger: User enters a photo of a receipt to be analyzed by the application
Pre-condition: The receipt photo has already been taken and submitted to the application
Outcome:

1. System searches the receipt looking for purchase date, time, and location information.
2. System searches for products purchased and their prices.
3. Products are mapped *(see Individual Product Use Case for "Map receipt data to database")* and purchase data is recorded in the database.

Title: Map receipt data to database
Trigger: User enters a photo of a receipt to be analyzed by the application
Pre-condition: The receipt has already been processed by the system
Outcome:

1. System goes through each found product on the receipt and checks if it exists in the database.
2. If an entry in the database exists for the product, the date, time, location, and purchase price are recorded in that entry.
3. If no entry for the product exists in the database, an entry is created and the date, time, location, and purchase price are recorded.

Title: Display analyzed spending data
Trigger: User wants to check their spending data
Pre-condition: The user is logged in and has spending data already entered into the application
Outcome:

1. The system checks the user's purchases in the database.
2. The system displays a visual representation of the user's purchase history and data is provided on what was purchased and when.

Title: Provide budgeting suggestion
Trigger: User exceeds budgeting preferences submitted
Pre-condition: The user is logged in and has entered their budgeting preferences into the application
Outcome:

1. The system analyzes purchase history and compares it to the user's budgeting preferences
2. The system suggests items to stop purchasing and/or cheaper grocery alternatives if they exist *(see Individual Product Use Case for "Suggest cheaper grocery alternatives").*

Title: Suggest cheaper grocery alternatives
Trigger: A recently purchased item by the user is purchased at a cheaper price
Pre-condition: A database entry for the recently purchased item and spending data exist
Outcome:

1. The system analyzes recent purchase history of the item and purchase location from different users.
2. If the location is within the current user's location preferences, the item, price and location are prompted to the user.
3. If the location is not within the current user's location preferences, no prompt is created.

9 Functional Requirements

9.1 Functional Requirements

Insert your content here.

10 Look and Feel Requirements

10.1 Appearance Requirements

- **AR1**: The app should have a consistent colour scheme throughout.

- **AR2**: The app should have self-descriptive icons to convey to users what the related feature is.
- **AR3**: The user interface should avoid clutter and balance features with available space.

10.2 Style Requirements

- **SR1**: All app features and options should be in consistent locations.
- **SR2**: Navigation options should always be available regardless of the application page.
- **SR3**: Features and options should be grouped based on relation to each other to improve intuitiveness.
- **SR4**: The number of clicks to access different features should be minimized.
- **SR5**: The app should utilize indicators and messages to communicate information to users.

11 Usability and Humanity Requirements

11.1 Ease of Use Requirements

- **EUR1**: The app should save states between use in order to minimize user input.
- **EUR2**: The app should allow users to undo changes made.

11.2 Personalization and Internationalization Requirements

- **PIR1**: English will be supported.
- **PIR2**: The app should have dark and light mode colour schemes.
- **PIR3**: The app should allow users to enable/disable external notifications.

11.3 Learning Requirements

- **LR1**: The app should have an FAQ or Help page to assist users.
- **LR2**: The app should have a tutorial to assist users in learning the available features.

11.4 Understandability and Politeness Requirements

- **UPR1**: The app should use common language found in the English dictionary.

11.5 Accessibility Requirements

- **ACR1**: The app should allow users to change text size.
- **ACR2**: The app should have multiple colour schemes to help those with visual impairment.

12 Performance Requirements

12.1 Speed and Latency Requirements

- **SLR1**: The app should take on average 1 second to complete backend tasks and at worst should never exceed 12 seconds.
- **SLR2**: The navigation of the app should be smooth, taking no more than 1 second render and handle page changes.

12.2 Safety-Critical Requirements

- N/A

12.3 Precision or Accuracy Requirements

- **PAR1**: The app should be able to correctly read user inputs with 99% accuracy.
- **PAR2**: The app should return data to the user with correctness within 99% of the intended value.

12.4 Robustness or Fault-Tolerance Requirements

- **RFR1**: The app should have proper error-handling against improper user input.

12.5 Capacity Requirements

- **CR1**: The app should take up no more than 1GB of space.

12.6 Scalability or Extensibility Requirements

- N/A

12.7 Longevity Requirements

- N/A

13 Operational and Environmental Requirements

13.1 Expected Physical Environment

Insert your content here.

13.2 Wider Environment Requirements

Insert your content here.

13.3 Requirements for Interfacing with Adjacent Systems

Insert your content here.

13.4 Productization Requirements

Insert your content here.

13.5 Release Requirements

Insert your content here.

14 Maintainability and Support Requirements

14.1 Maintenance Requirements

Insert your content here.

14.2 Supportability Requirements

Insert your content here.

14.3 Adaptability Requirements

Insert your content here.

15 Security Requirements

15.1 Access Requirements

Insert your content here.

15.2 Integrity Requirements

Insert your content here.

15.3 Privacy Requirements

Insert your content here.

15.4 Audit Requirements

Insert your content here.

15.5 Immunity Requirements

Insert your content here.

16 Cultural Requirements

16.1 Cultural Requirements

Insert your content here.

17 Compliance Requirements

17.1 Legal Requirements

Insert your content here.

17.2 Standards Compliance Requirements

Insert your content here.

18 Open Issues

Insert your content here.

19 Off-the-Shelf Solutions

19.1 Ready-Made Products

There exists some existing products that address the problem we are also addressing.

flashfood matches users with food that is nearing their expiration and are marked down to prevent waste and provide cheaper prices.

ibotta helps users track their spending and offer coupons to those wanting to save money.

19.2 Reusable Components

- **node-tesseract-ocr**: for reading labels or receipts.
- **PostgreSQL**: for storing relational data.
- **flutter**: framework for mobile app development.

19.3 Products That Can Be Copied

There are no known products that can legally be copied to form a solution for this problem.

20 New Problems

20.1 Effects on the Current Environment

Insert your content here.

20.2 Effects on the Installed Systems

Insert your content here.

20.3 Potential User Problems

Insert your content here.

20.4 Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

Insert your content here.

20.5 Follow-Up Problems

Insert your content here.

21 Tasks

21.1 Project Planning

Insert your content here.

21.2 Planning of the Development Phases

Insert your content here.

22 Migration to the New Product

22.1 Requirements for Migration to the New Product

N/A

22.2 Data That Has to be Modified or Translated for the New System

N/A

23 Costs

Insert your content here.

24 User Documentation and Training

24.1 User Documentation Requirements

Insert your content here.

24.2 Training Requirements

Insert your content here.

25 Waiting Room

Insert your content here.

26 Ideas for Solution

Insert your content here.

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. **What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.**

   For this application, many skills will be required in order to ensure this project's success. This includes a combination of project specific knowledge as well as a few less technical skills. Together, these skills will ensure the smooth development of our application but will also likely prove useful in the future as we move forward with our careers.

   (a) Working knowledge of *Flutter* development and front-end design
   (b) Knowledge working with *Node.js* backends and REST APIs
   (c) Skills related to the use of the *Tesseract-OCR* engine
   (d) Working knowledge of *PostgreSQL*
   (e) Experience working with Natural Language Processing or Machine Learning
   (f) Writing proper software documentation
   (g) Familiarity and experience with software testing/coverage
   (h) Presentation and communication skills for elicitation and demos

2. **For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?**

For the above skills, there are many approaches that could be taken to learning and developing them. In terms of skills *(a)* to *(e)*, these would be categorized as development oriented skills. These would involve knowledge that could directly apply to the technology being employed for this capstone project. Since these are each popular technologies in their own right, many resources could be accessed to learn about them. Watching *YouTube* videos and tutorials would be one resource that could be used to strengthen and learn these skills. *YouTube* tutorials often provide more practical applications for the skill in question and the site has content that covers a wide variety of topics. Additionally, official documentation could be read in order to see how the original developers intend the technology to be used. This would provide a more in-depth understanding of each aspect of the technology. There are also plenty of third-party online articles that could be referenced when learning these types of skills. One last approach that could be taken is referencing open-source projects using these technologies to see how they are used in real development.

Skills *(f)* and *(g)* are technical but less directly involved with the coding portion of development. Despite this, there are still plenty of resources that could be used to learn and familiarize ourselves with them. One place to look could be previous coursework, in particular, *SFWRENG 3RA3* and *SFWRENG 3S03* notes could be used to reference documentation and software testing respectively. Furthermore, resources for these topics also exist on *YouTube* which could be leveraged in addition to third-party articles.

Lastly, skill *(h)* is a general skill that is important for all engineers to be comfortable with. One approach to mastering this skill could be practical applications, including but not limited to holding mock presentations or focus groups with the team. Practicing presentations and communication is one of the easiest ways to help develop this skill. Moreover, there are many seminars and courses online that help teach strong communication practices such as ones on *Coursera* or *LinkedIn Learning*.

With these in consideration, each team member has listed their chosen skills, their chosen approaches from above, and why they chose to pursue these skills below.

Ryan Yeh
Chosen Skills: (a), (f), (g), (h)
At the beginning of this project, I was assigned as the lead for documentation and testing. As such, I would like to take the opportunity to strengthen my understanding of these skills in order to meet my team's expectations. Outside of this project though, I believe these skills will be very helpful regardless of what type of developer I end up becoming. Additionally, I wanted to strengthen my understanding and skills in front-end development. UI design is a software field that I have an interest in but lack sufficient experience currently. Therefore, I would love to take this opportunity to learn more about it and strengthen my understanding of it. Lastly, I chose communication skills as something to focus on due to my belief that it is something I could improve upon. Communication and presentation skills will always be universally needed and putting an emphasis on it now would greatly help me in the future.

My approach for developing my documentation and software testing skills will be to refer to previous year course notes. This will help me to reinforce the material I have already learned and refresh myself on topics I may have forgotten. I also plan to read third-party articles as a supplement to learn more about certain topics that may not have been covered in as much detail in class. For *Flutter* and front-end design, I plan to watch *YouTube* tutorials in order to get more a hands-on experience with the technology. I also plan to reference documentation and articles in order to further my learning beyond the videos watched. Finally, I will leverage my group in order to strengthen my communication skills. I will try and place myself into positions that force me to practice communication such as leading meetings, as well as practicing presentations with the rest of my team.

Sawyer Tang
Chosen Skills:
Insert your content here.

Jason Nam
Chosen Skills:
Insert your content here.

Allan Fang

Chosen Skills:
Insert your content here.

Added Section - Appendix - Stakeholder Requirements Elicitation Notes

- Consider your favourite or most used mobile apps, what elements of the UI stand out? What in your opinion makes an appealing user interface?
  - Minimize options to create a cleaner UI.
  - Utilization of easily accessible menu bars (like Instagram) to improve navigation UX.
  - Consistent and cohesive colour theme throughout.
  - Use of icons that clearly convey what the option is meant for.
  - Put options in consistent locations to minimize hand movements (easy to tap interface).
- When you think of intuitive applications, what about the application makes the UI or UX intuitive?
  - Many apps borrow UI layouts from each other, helps with familiarity in terms of navigation.
  - Always accessible navigation options.
  - Proper organizing of features (unrelated features should be separated on the UI).
  - Swipe navigation and minimal long presses.
  - Saving state when closed to minimize user input on future uses.
- How important do you consider application feedback and what feedback would you expect from an application? (i.e. loading indicators)
  - High importance to communicate errors and loading to users.
  - Should always convey when issues happen or something is being done.

  - When fetching information or other longer duration tasks, loading bar should be utilized.
  - Utilization of messages or pop ups if resources are unavailable or something goes wrong.

- When learning a new app, how long would you consider a reasonable amount of time to consider an app easy-to-use?
  - Average acceptable time to consider an app intuitive would be to be able to figure out basic functionality within 15 minutes.

- What features would you want to help with the learning experience of a new app?
  - The use of a tutorial on first launch to take users through the UI and different features.
  - The use of tooltips for certain options and features.
  - A button that users can press to access a Help menu or FAQ.

- How do you usually go about learning a new application?
  - All participants said they tend to brute force applications in order to learn (i.e. clicking around and seeing how the application responds).
  - As an addition to this brute force approach, participants explained how useful an undo feature would be as well as always accessible navigation.

- If you were using an application, what personalizations/accessibility settings would you consider a must?
  - Dark/Light mode.
  - Ability to change text sizes.
  - Multiple language support.
  - Allowing users to enable or disable notifications if used.
  - Alternate color palettes to help with those with visual impairment like color blindness.

- Consider an app you use often, generally, how long would you say you wait for operations to complete? How long would you be willing to wait for an application before closing the app?
  - Ideally, regular operations should take a second or less.
  - At most, participants would wait an average of 12 seconds before closing the app or thinking something went wrong.
- What kind of phone do you use? (Apple/Android/Other)
  - All participants used Apple devices.
- On a scale of 1-10, for an app of this nature, how important would you consider accuracy of the system and data? How important would you consider responsiveness on the same scale?
  - All participants considered accuracy of the features and data to be the most important (10) due to the financial aspect of the app.
  - Responsiveness was considered important (8) in order to ensure a positive user experience. If the app does not feel good to use, most participants said they would find another app instead.
- How much do you consider the size of an application you install? How big of an app would you consider reasonable?
  - Most participants said they did not pay much attention to the size of applications when installing them.
  - Participants said they would consider anything over 1GB unreasonable for an app of this nature.