A WEB-BASED WAREHOUSE INVENTORY MANAGEMENT SYSTEM FOR STOCK ITEMS TRACKING

ELAINE LOW JING YI

UNIVERSITI TUNKU ABDUL RAHMAN

A WEB-BASED WAREHOUSE INVENTORY MANAGEMENT SYSTEM FOR STOCK ITEMS TRACKING

ELAINE LOW JING YI

A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Science (Honours) Software Engineering

Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

September 2022

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature : 

Name : Elaine Low Jing Yi

ID No. : 18UEB03140

Date : 08/09/2022

APPROVAL FOR SUBMISSION

I certify that this project report entitled **"A WEB-BASED WAREHOUSE INVENTORY MANAGEMENT SYSTEM FOR STOCK ITEMS TRACKING"** was prepared by **ELAINE LOW JING YI** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Science (Hons) Software Engineering at Universiti Tunku Abdul Rahman.

Approved by,

Signature : *Lai*

Supervisor : Dr Lai Yen Lung

Date :

Signature :

Co-Supervisor :

Date :

ACKNOWLEDGEMENTS

I would like to thank everyone who contributed to the successful completion of this project and express my gratitude to my supervisor, Dr Lai Yen Lung for his guidance and enormous patience throughout the completion of this project as well as Mr Sor Kean Vee for his advice and suggestions to improve this project.

In addition, I would also like to express my gratitude to my loving family members and friends who had given me support and encouragement during my degree studies.

ABSTRACT

Warehouses act as an important role in the supply chain to store inventories. To keep up with the customer's demand on time, inventories need to be organized and tracked with proper inventory system control so that business operations are done efficiently. However, in the current competitive market environment, warehouses have to deal with discrepancies between the recorded inventory count and the actual item count. Besides, to maintain the accurate inventory count, inventory system control such as cycle counting need to be in place, in which extra manpower is needed to classify the stock keeping unit (SKUs) and schedule the cycle counting process. Moreover, the existing SKU classification methods have their limitations in classifying the SKU. Therefore, a web-based warehouse inventory management system was implemented to address the mentioned problems. The system was developed using the phased-development methodology and was divided into three phases to develop features including the automated cycle counting scheduler, real-time check-in/ out stock module, report management, and the management of warehouses, users, inventories and categories. The front end of the system was developed using Vue.js whereas the back end (APIs) was developed with Pusher Channels for the real-time inventory stock tracking module. ng module. Web service testing was carried out using Postman and the entire system was deployed to Heroku after all test cases of web service testing passed for user acceptance testing and usability testing. The average SUS score was 85.5, hence, it could be concluded that the user interface of implemented system is simple and easy to use. Besides, 100 iterations of check-in/out stock were done and the time differences between creating stock and receiving API message was observed and recorded. The average delay time for the real-time functionality was around 1 second. In a nutshell, the implemented system successfully addressed and achieved the main project objectives to keep track of inventory stocks in real-time, automate cycle counting scheduler and improve and implement the existing SKU classification with higher completeness and efficiency.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS / ABBREVIATIONS

API	Application Programming Interface
AUP	Agile Unified Process
CRUD	Create, Read, Update, Delete
CSS	Cascading Style Sheets
DFD	Data Flow Diagram
ERD	Entity Relationship Diagram
ERP	Enterprise Resource Planning
FDD	Feature-Driven Development
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
IMS	Inventory Management System
IRA	Inventory Record Accuracy
JAD	Joint Application Development
JSON	JavaScript Object Notation
JWT	JSON Web Token
KEGA	Key Economic Growth Activities
MVC	Model-View-Controller
PHP	Hypertext Preprocessor
RAD	Rapid Application Development
SDLC	Software Development Life Cycle

SKU	Stock Keeping Unit
TDD	Test-Driven Development
TOC	Theory of Constraints
UAT	User Acceptance Testing
UI	User Interface
WBS	Work Breakdown Structure
WMS	Warehouse Management System

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Introduction

Manufacturing companies process raw materials into finished goods, which will then be sold into the market. To meet production needs, warehouses often act as an important role to store all the raw materials before being processed into final products. This is important in the supply chain to reduce transportation costs, achieve economies of scale in terms of producing or purchasing, provide value-added processes, and lastly shorten the response time (Harjono, 2019).

To keep up with the customer's demand on time, keeping track and managing inventories are mostly important for easier and smoother operational activity and operation cycle. By keeping the inventories organised and tracked, inefficient work such as halting business operations to recount all stocks in the warehouses yearly to make sure the inventory count in the system is the same as the real inventory count can be avoided. Besides, it is also necessary to have a proper inventory system control as it may affect the production process if inventories were not restocked in time and have an impact on the financial statements of the company (Harjono, 2019).

Shared Prosperity Vision 2030, which was announced by the Malaysian government on 5th October 2019, aims to provide a decent standard of living to Malaysians by 2030. In this Shared Prosperity Vision, it proposed seven strategic thrusts, one of which is the Key Economic Growth Activities (KEGA), which consists of activities which demonstrate the potential to achieve Malaysia's aspiration to reach high-level economic development. Among the total of 15 KEGAs, KEGA 14 which is the advanced and modern services stated that the services sector, as the largest sector in Malaysia, must be modernised and diversified to be in line with the current demand of the market and advancement of technology (Ministry of Economic Affairs, 2019). This project is able to fulfil this activity by digitising the inventory recording process and making it to be in real-time so that inventory data is consistent across multiple locations to support the business operations to be in line with current market demand. This warehouse inventory management system can also reduce the workload of scheduling for cycle counting by automating the analysis and scheduling process.

This project is initiated to analyse the issues faced in inventory management of warehouses and propose solutions regarding issues found. This chapter will be discussing the background of the problem, the problem statements, project objectives, proposed solution, proposed approach as well as the project scope.

1.2 Background of the Problem

As the world population rises from day to day, demand for items and foods increases. Hence, the supply chain and manufacturing companies have to keep pace with this situation. Warehouses, the "middleman" of supply chain and manufacturing operations, also became greatly important in today's competitive market environment.

Although there are many types of warehouses such as raw materials warehouses, distribution warehouses, etc, most of them have a similar pattern of material flow. The warehouses need to deal with daily operations and transactions such as receiving goods, putting away, order picking, accumulating, sorting, packing, cross-docking, shipping, and internal replenishment (Ramaa, Subramanya, Rangaswamy, 2012). The performance of a warehouse and the accuracy of every transaction data dramatically affect the efficiency and accuracy of the whole supply chain.

To maintain the inventory accuracy in the warehouses, physical inventory counts were performed, usually once or twice a year, to count all inventories in the warehouses before they compile their annual financial reports. This solution works for companies with only a minimal number of inventories. Large companies with thousands of items normally need to halt their operations for a week or more for a full physical inventory count. This will cause them to provide poor customer service and some discrepancies due to internal or temporary staff not recording some inventory correctly or classifying items incorrectly, which cannot be avoided (Jenkins, 2020).

On the other hand, cycle counting solves the issues raised on physical inventory counts as it counts small, preselected sections of inventory periodically. It can save a lot of time while improving inventory accuracy. Besides, by doing cycle counting, companies no longer need to shut down their operations for a few days to perform a full inventory count. Hence, cycle counting has become a popular inventory management strategy for companies across all industries due to its benefits that included higher accuracy on inventory assessments, fewer errors, zero costs for employee overtime, and early detection of thefts in a timelier manner (Schwarz, 2021). Nevertheless, the accuracy of the cycle counting is affected by the inventory counts

since cycle counting needs to match the actual count with the inventory counts to calculate the inventory record accuracy (IRA). Besides, the schedule of cycle counting should also be managed wisely.

1.3 Problem Statement

This section discusses the three problem statements noticed from the problem formulation phase, which include the inaccuracy of inventory records, difficulties in scheduling cycle counting progress in the most efficient way and limitations in the SKU classification method.

1.3.1 Inaccuracy of inventory record

The inventory records in the warehouse system should be real-time and accurate, which can keep track of the items coming in and going out (Curley, 2015).

In the current competitive market environment, manufacturing businesses have to move quickly, which means that the inventory count should be updated in real-time, and the availability of inventory counts should be at all times. If failed to do so, there will be discrepancies between the inventory count in the ERP system and the actual item count (Wisys, n.d.). When the warehouses have to deal with multiple locations, outstanding transactions, and paperwork lags, the inventory count might not be updated in real-time, if manual recordings were used. As a result, false variances will occur, and the inventory count will be inaccurate (Schwarz, 2021). This will cause a lot of problems such as being unable to replenish stocks at the right time or carrying too many stocks which increase the warehouse' expenses. Inventory shortage can then lead to a loss of revenue due to the inability to fulfil orders (NC-Vision, 2021).

Hence, a warehouse system that is able to keep track of the inventory stocks in real time is needed. For instance, the system from different locations should be able to record the name of items, amounts, and required details when the items are received and shipped out on one end.

1.3.2 Difficulties in scheduling cycle counting progress in the most efficient way

Ideally, the cycle counting process should be able to cover all SKUs needed to be counted within the time interval specified.

However, extra manpower would be needed to classify the items into respective counting groups by following the specified documented process and arranging the cycle counting based on their category, counting frequency, and time interval if done manually.

An auto-scheduler should be used to schedule the cycle counting process based on settings entered by the warehouse manager.

1.3.3 Limitation in SKU classification methods.

The SKU classification method implemented should be able to define the importance of each inventory in a wider paradigm, so that more classifiers are considered to classify the SKU.

There are several types of inventory management techniques such as ABC analysis and VED analysis, being the most common inventory control methods among all. Each technique has its limitation in classifying the SKU for cycle counting. For instance, ABC analysis only classifies the SKU based on their usage value or stock value (Drakeley, 2021). It classifies inventories into three categories:

1. Category A is the fastest-moving and most expensive product. The threshold can be determined by the organisation. The most common threshold is the top 20% of the inventory list by velocity.
2. Category C is usually the bottom 15% to 20% of the item list by velocity.
3. Other inventories that did not fall into categories A and C will be in category B (Curley, 2017).

Stock keeping units in A class will be counted more often, followed by B class and finally C class. For example, all SKUs in A class are counted daily, followed by B class to be counted weekly and C class to be counted monthly. The cycle counting frequency of each class can be defined by the organization.

However, some of the inventories may not hold to this paradigm. For example, critical medicine such as an inhaler or snake venom vaccine should be given higher priority when comes to cycle counting. This is to ensure that vital drugs that are lifesaving are always present in the pharmacy or hospital. VED analysis, on the other hand, fits this purpose in classifying the inventories into vital (V), essential (E) or desirable (D) based on their criticality, unfortunately, with the limitation of not

considering the stock value of the inventories. In other words, pure ABC analysis has the limitation of VED analysis's strength and vice versa.

In fact, there is a matrix analysis of ABC-VED, which can satisfy both purposes. Instead of categorising them into A, B, C or V, E, D, ABC-VED analysis cross-tabulates both together and come up with new classes namely I, II and III (Ceylan and Bulkan, 2022). Cross tabulation is meant to overcome the limitation of both ABC and VED analysis by considering all possible combinations of A, B, C and V, E, D, ie., AV, AE, AD, BV, BE, BD, CV, CE, and CD before further classify them into the new classes I, II and III. Nevertheless, ABC-VED analysis is not in which the classification of groups I, II and III only considered the nine cross-tabulated classes. This is mainly due to the classification of the inventories' criticality has only considered three classes: V, E, and D, which limited the resultant number of cross-tabulated classes for a wider range of criticality assessments. In practice, different inventories can be assigned with different criticality scores (more than three) where a change in an inventories' operational context might cause its criticality value to increase or decrease (English and Yunusa-Kaltungo, 2022).

According to ISO 55000, "a critical asset, can be safety-critical, environment-critical, or performance-critical, is one that has the potential to significantly impact the achievement of the organization's objectives". As a result, it is appealing that a more quantitative way of classifying inventories that considers a broader range of inventories by their criticality (not restricted to only V, E, and D) be established.

1.4 Project Objectives

The objectives to be achieved in this project are

1. To develop an application to record all in-and-out of the warehouse inventories from different locations in real-time
2. To develop an automated scheduler which can categorise SKUs, create schedules, and allocate them to cycle counting personnel.
3. To propose and implement an improvement over existing SKU classification to achieve a higher degree of completeness and efficiency.

1.5 Project Solution

To solve the problems addressed above, a web-based warehouse inventory management system was proposed. The targeted users will be the admin, warehouse manager, and warehouse staff. Besides, this system will be able to update inventory in real-time to avoid inconsistencies or false variances, with an auto-scheduler that schedules cycle counting based on the enhanced ABC-VED cycle counting method.

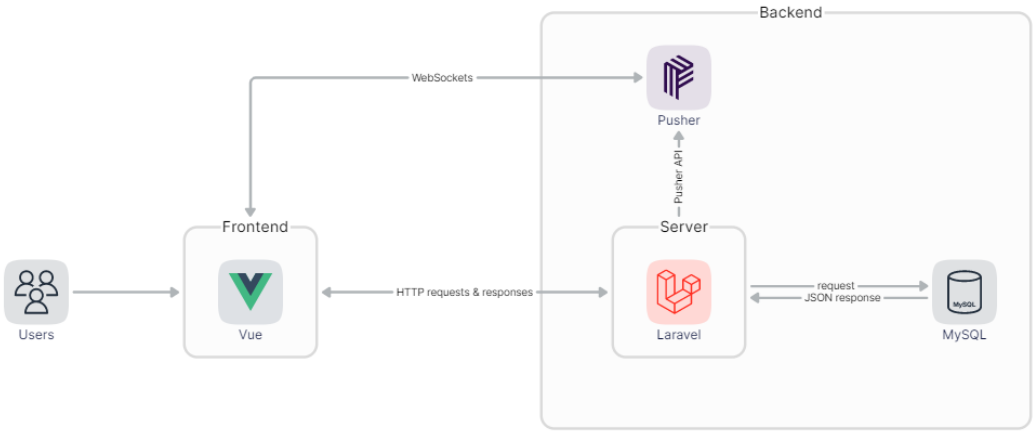

Figure 1.1：System overview

Figure 1.1 illustrates the high-level system overview for this project. The frontend development will be using Vue.js, whereas the backend development will be using Laravel with MySQL database. Once an HTTP request has been made, the server will send the event through Pusher API to the Pusher server and Pusher will push to clients via implementing WebSocket. This provides the ability of this warehouse inventory management system to update data in real time, which means that clients from different sites or locations will be able to view the updates of the inventory stocks without having to refresh the page.

1.6 Project Approach

The proposed system development methodology for this project is the Phased-Development Based Rapid Application Development (RAD).

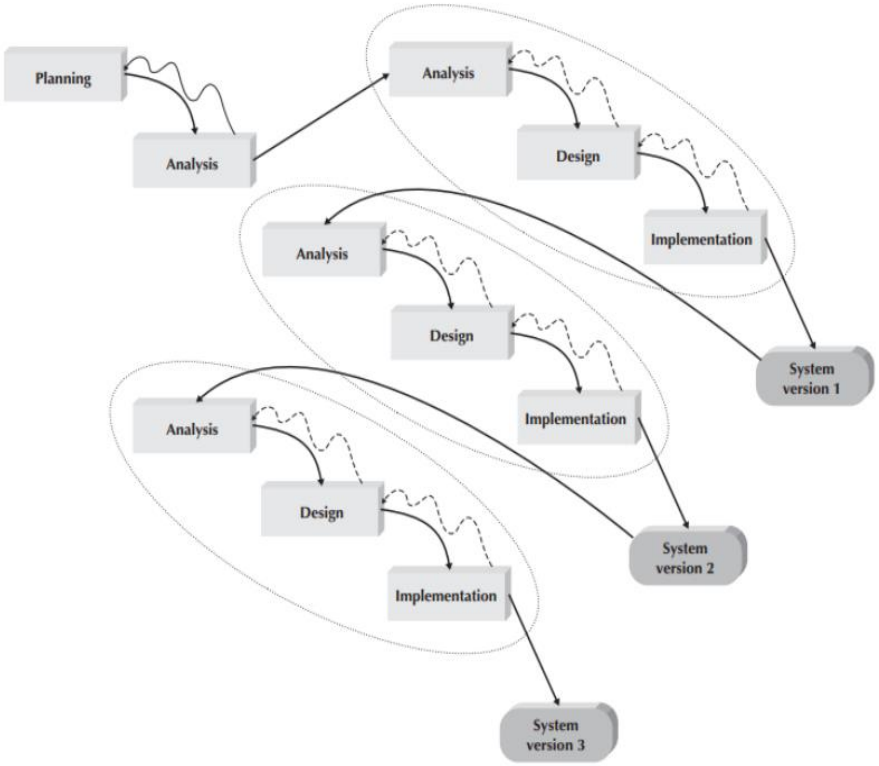

Figure 1.2: Phased Development Based Methodology
(Dennis, Wixom and Tegarden, 2015)

This methodology divides the whole system into several versions of the system which will be developed sequentially. The overall system concept will be determined in the analysis phase, and the requirements can be categorised into several versions. The most important requirements can be prioritised in the first version and proceed to the design and implementation phase which only includes the requirements for the first version. After version 1 is implemented, another round of analysis, design and implementation can be performed for version 2. This process iterates until the fully functional system has been developed. As a result, a system with useful functions can be developed quickly (Dennis, Wixom, and Tegarden, 2015).

For this project, the system can be divided into three versions in which the first version would be the modules that are the most important. This involves the setting up of databases, preparing sample data and authentication as they are important for later development. Besides, the first module will also be included in this version, which is the real-time checking in and out stock module. The second version will include the auto-scheduler module as they are the main objectives of the project. The third version will be developing the report management which will include the daily

report, cycle counting approval report and cycle counting summary report. All CRUD of users, inventories, warehouses and categories will also be done in this version.

1.7 Project Scope

This project develops a simple web-based warehouse management system, which focuses on inventory management. This web application can be accessed by the admin, warehouse manager, and staff. The responsibilities and actions for each role are listed in Table 1.1. This warehouse management system involves three main modules, which are inventory stock tracking, auto-scheduler, and report management.

Table 1:1: Roles and Responsibilities for Warehouse Inventory Management System

Roles	**Actions/ Responsibilities**
Admin	- Add, delete, update and view users, and set roles for them - Add, delete, update and view inventories of all warehouses - Add, delete, update and view a warehouse. - Add, delete, update and view a category. - View daily reports of in-and-out of inventories from all warehouses - View the summary report for cycle counting of all warehouses
Warehouse manager	- View all inventories in the system - Start a cycle counting process - View all upcoming cycle counting SKUs and reassign staff to replace the original staff responsible, if necessary - View current cycle counting settings - Approve or reject cycle count pending approval report - View daily reports of in-and-out of inventories from own warehouse - View the summary report for cycle counting for own warehouse - View all staff in the warehouse
Staff	- Check-in and out stock - View the cycle counting schedule assigned to them - Perform cycle counting and submit cycle count report - View all inventories in the system

1.7.1 Inventory Stock Tracking

The first module is the inventory stock tracking functionality in which users can record the in-and-out operations. For example, when a distributor arrives and takes inventories from the warehouse, the staff in charge counts the inventories taken and updates the system by searching the inventory IDs or names and entering the amount taken. These changes will be updated in real-time to the database. Other staff in the warehouse will also have the newest updates in their system without the need to refresh the page.

1.7.2 Cycle Counting Auto Scheduler

Secondly, this system provides an auto-scheduler which will be able to automatically generate schedules and assign them to staff based on settings entered by the manager. The classification of SKUs will be using the proposed classification method that improves the ABC-VED classification method with higher completeness and efficiency. Using the count frequency of each class and workday calendar of the warehouse entered by the warehouse manager, the auto-scheduler can determine which and how many inventories need to be counted daily and the auto-scheduler will auto-assign SKUs to staff that was assigned to cycle counting evenly. Then, the staff can select an SKU to start performing the cycle count and enter the actual count into the system. After the staff has performed cycle counting, the system will generate a cycle count pending approval report for the manager to approve and update the count to the system.

1.7.3 Report Management

Thirdly, this system auto-compiles the entries entered by cycle counting staff and generates the cycle count pending approval report, which will calculate the variances with the recorded inventory count with the date counted. The warehouse manager can approve the adjustment of the actual inventory count or reject the count in which the staff will need to perform the cycle counting to that SKU again. If the manager approves the approval report, the cycle counting summary will be generated with the inventory record accuracy (IRA). Warehouse managers and admin can view the summary reports and sort the list of summary reports by ascending order of inventory record accuracy (IRA) to find out the lowest IRA and define the root cause of low

accuracy and try to solve the issue. The system also generates daily reports for warehouse managers and admin to view the transactions carried out each day.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Warehouse inventory management systems can be complicated system as it involves a lot of processes and holds an important role in business operations. Hence, a literature review was conducted to better understand areas that are related to the proposed idea of this project. This chapter aims to:

1. Review similar projects on warehouse inventory management system
2. Understands cycle counting classification methods
3. Learns to perform cycle counting scheduling
4. Identify a suitable development methodology for this project

2.2 Similar Projects Review

2.2.1 Challenges of existing system

James (2016) stated that the old system used by Smart shoppers' in Masaska faced an inefficiency in record tracking because they mainly rely on paperwork to record the products and stock information. As a result, office spaces were required to compile those manual files, which also increases the cost of the business. A lot of time was also wasted retrieving and manipulating the data. Tahir (2020) also added that a lot of time was wasted using hard copy documentation in Walid Halal to search for particular records, and data duplication is common with an update and delete error. Besides, he also reminded that the manual ledger system adopted by Walid Halal also faced the challenges of wear and tear of ledger books which leads to loss of data.

2.2.2 Features of inventory management system

James (2016), Maharjan and Humagain (2016), and Tahir (2020) have developed an inventory management system to solve various problems that occur due to paperwork and manual files. Their projects were reviewed and for easier illustration, the features developed are tabulated in Table 2.1.

Table 2:1: Summary of features developed by James (2016), Maharjan and Humagain (2016), and Tahir (2020)

Sources / Features	James (2016)	Maharjan and Humagain (2016)	Tahir (2020)
Register customer	✓		
Create warehouse		✓	
Stock CRUD	✓		✓
Category CRUD	✓		
Inventory CRUD	✓		✓
Record sales to customer	✓	✓	
Record purchases from supplier		✓	
Record sales and purchases return		✓	
View reports		✓	
Backup and restore data		✓	

2.2.3 Significance and Limitation

The system developed benefits in a way that it increases the efficiency of business operations. James (2016) and Tahir (2020) save Smart shoppers' expenses on purchasing paper and ink to document products and stock data. Besides, they also reduced the fraud that may be caused by paper-based recording, by developing an inventory management system to keep track of inventories neatly. Customers of Smart

shoppers and Walid Halal Spices also benefit in a way that they are better served due to faster information access using Inventory Management System (IMS). Maharjan and Humaigan (2016), on the other hand, enables organisations with more than one warehouse to use the IMS for easier inventory management. His system also provides reports for daily sales and purchases which ease management decisions.

However, there were some limitations in the projects mentioned above, in which Maharjan and Humaigan (2016) stated that the application developed is not applicable for warehouses that have multiple levels of roles such as admin, warehouse manager and normal staff, meaning that no authorization was done to grant access of actions to specific roles in the IMS. Besides, the system developed is not applicable for businesses that have large quantities of inventories. This is due to the lack of searching features as mentioned by Tahir (2020).

2.3 Cycle Counting

The cycle counting method is a kind of inventory control method in a way that it fixes the mismatch of inventory counts recorded in IMS and the actual count of the stock keeping units (SKUs) in the warehouse. It can happen anytime depending on the business operations. The main advantage of cycle counting compared to the traditional physical inventory is that cycle counting will not need to shut down their business to count all items in warehouses (Shen, 2017). Fathoni, Ridwan and Santosa (2019) also listed some of the benefits of using cycle counting in which fewer experienced and knowledgeable people were needed for this task. This helps in a way that experienced and expert people can be allocated to other crucial parts of the business operations. Besides, they also confirmed that cycle counting improves the process by reducing inaccuracy of inventory records by tracking and fixing discrepancies between records and actual counts.

In cycle counting, a good and stable sample for cycle counting is critical as a sample that is too small cannot represent the entire population and a large sample may defeat the purpose of sampling. However, no matter how accurate the cycle counting was performed and how perfect the samples were, errors in the cycle counting process will always occur and these errors are important to define the root cause of the existing inventory problems instead of just simply correcting the inventory errors whenever errors occur as it will not help in a long run (Shen, 2017). Hence, the calculation of inventory record accuracy (IRA) is important to ease management decisions to correct

the existing problems that cause inventory accuracy. Efforts should always be done to achieve a high IRA.

2.3.1 Traditional ABC analysis

Shen (2017) researched the four most commonly used cycle counting techniques, which include control group cycle counting, random sample cycle counting, ABC cycle counting and process control cycle counting. Among all of the four techniques, the ABC cycle counting and process control cycle counting focus more on warehouses and distribution centres whereas the rest of the two techniques are more on store-level cycle counting.

Shen (2017) and Mahtamtama, Ridwan and Santosa (2018) stated that ABC analysis's major approach is to assign SKUs by classifying the SKUs based on the 80/20 rule, which means that 80% of the cumulative revenue or stock value in the warehouse is from 20% of the SKUs. SKUs will be classified into three classes, in which SKUs that fall into 80% of the revenue or stock value will be in class A, 85%-95% in class B and the rest of it will be in class C. The cycle counting frequency of SKUs in class A will be the highest, followed by class B. Class C will be counted least frequently. Figure 2.1 shows the general concepts of classification of ABC analysis.

Mahtamtama, Ridwan and Santosa (2018) researched the calculation steps for stock value to classify the SKU. Firstly, the stock value should be counted by multiplying the unit cost of each product or inventory with the quantity on hand. Secondly, in order to categorise the SKUs, the stock value needs to be converted to a range of 0 to 1. In order to do this, all stock values for all SKUs should be calculated first and the maximum and minimum stock value will be used to calculate the transformed stock value. The calculation formula is as follows:

$$\text{Stock Value} = \text{Unit cost of SKU} * \text{Quantity on Hand}, \tag{2.1}$$

$$\text{Transformed Stock Value} = \frac{\text{SKU stock value} - \text{Min stock value}}{\text{Max stock value} - \text{Min stock value}}$$

After the transformed stock value has been calculated, the SKUs can be classified using the ratio mentioned above. The cycle counting frequency adopted by PT XYZ is in Table 2.2. However, the cycle counting frequency varies between organisations and it is decided by the management.

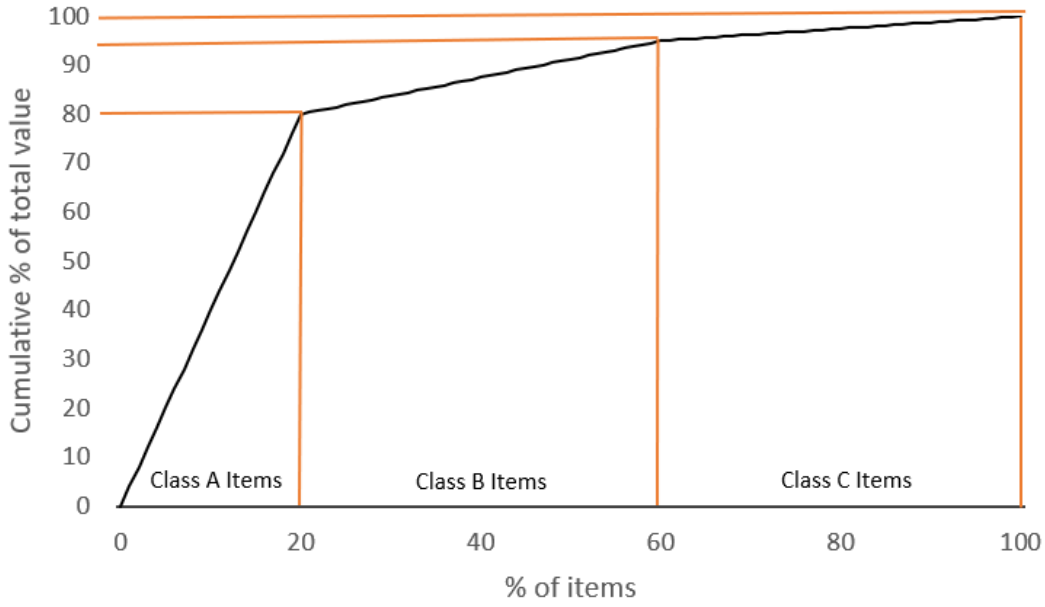

Figure 2.1: General concepts of ABC cycle counting method

Table 2:2: Counting Frequency Adopted by PT XYZ

(Mahtamtama, Ridwan and Santosa, 2018)

ABC class	Cycle counting frequency
A	Weekly
B	Monthly
C	Quarterly

2.3.2 ABC-VED

In some cases, the ABC analysis alone is not sufficient especially when the business inventories have criticality. For example, drugs which are critical for life but not having high stock value as the disease may be not common but deadly if without medicine control.

Fathoni, Ridwan and Santosa (2019) developed a new stock-taking policy, which is the ABC- VED cycle counting, with the purpose of improving the record accuracy of a pharmaceutical company. VED analysis classifies the SKUs into three groups, which are vital (V), essential (E) and Desirable (D).

2.3.2.1 Classification of SKUs

The classification is based on the criticality of the inventories. In ABC-VED analysis, the drugs are classified into 9 subclasses, each class was labelled by two alphabets: the first one indicates ABC analysis while the second one indicates VED analysis. The 9 subclasses will then be categorised into 3 groups: I, II, and III. The classes in ABC-VED analysis are in Table 2.3 (Fathoni, Ridwan and Santosa, 2019).

Table 2:3: ABC-VED classification

(Fathoni, Ridwan and Santosa, 2019)

Groups	Subclasses
I	AV, AE, AD, BV, CV
II	BE, BD, CE
III	CD

To prove the effectiveness and efficiency of the new stocking policy, Fathoni, Ridwan and Santosa (2019) analysed the classification of using ABC analysis alone, VED analysis alone, and ABC-VED analysis, which are tabulated in Table 2.4.

Table 2:4: Analysis of cycle counting policies

(Fathoni, Ridwan and Santosa, 2019)

Cycle counting policy	Classification Result	Limitation	Summary
ABC analysis	A: 733 items (79.97% of total sales) B: 146 items (2.72% of total sales) C: 185 items (0.65% of total sales)	Failed to pay high attention to the vital items that include: B: 146 items (2.72% of total sales) C: 185 items (0.65% of total sales)	Vital items which have the highest criticality are not given the highest priority as they are in class B and C.
VED analysis	V + E: 74.37% of total sales	Ignores 170 items in A class (17.97% of total sales)	Items that made a high volume of sales are not

		which is in the desirable (D) category	considered because they are in class D
ABC-VED analysis	I: 1064 items (33.1% of total sales) II: 1256 items (14.2% of total sales) III: 893 items (2.4% of total sales)	None or not mentioned	None or not mentioned

2.3.2.2 Strength and Limitation

In a nutshell, the ABC analysis has the strength of taking monetary aspects of the inventories into classification, whereas the VED analysis includes the criticality of the products to classify the SKUs. Hence, pure ABC analysis has the limitation of not taking the criticality of inventories into account and the same goes for pure VED analysis in which it is not taking the monetary value of the inventories such as sales value, stock value and so on into consideration. The ABC-VED analysis, on the other hand, solves both limitations of ABC and VED analysis by combining them and reclassifying them into three categories (I, II and III).

2.3.3 Cycle Counting Scheduling

Some research was done on cycle counting scheduling for the purpose of learning how to schedule the cycle counting. Scheduling using the ABC-VED classification approach and buffer-time management approach was covered in this section.

2.3.3.1 ABC-VED Scheduling

To ease the cycle counting process, Fathoni, Ridwan and Santosa (2019) included the generation of schedules of cycle counting in the inventory control system with the ABC-VED method. They calculated and tabulated the daily count for each group I, II and III in Table 2.5.

Table 2:5: Daily counts for Group I, II, and III

(Fathoni, Ridwan and Santosa, 2019)

Group	Items	Frequency	Frequency (in a year)	Daily Count
I	1064	Monthly	12	35
II	1256	Every 2 months	6	21
III	893	Every 10 months	1.2	3
Total	3213		19.2	63

The calculation of the daily count is done by dividing the total items in each category by the time frame given to count all items for that category. For example, by referring to Table 2.5, the group I has a total of 1064 items and all items need to be counted monthly (in 30 days). Hence, the calculation steps are as follows:

$$\text{Daily count} = \frac{\text{Total items}}{\text{frequency}}, \tag{2.2}$$

$$\text{Daily count} = \frac{1064}{30 \text{ days}},$$

$$\text{Daily count} = 35.4$$

$$(\text{means } 35 \text{ SKUs need to be counted daily for Group I})$$

After the daily count has been calculated, the cycle counting schedule can be generated based on the business operation. There are a few aspects that need to be considered in scheduling which are: how many shifts in a day, the working days of the warehouse and how many workers are assigned to perform cycle counting in each shift. For example, in a scenario of 3 shifts and 2 workers in each shift, the total daily count of 63 can be divided equally into 20 items per shift, and assign 10 items per worker in the shift. Figure 2.2 shows the UI of the cycle counting schedule developed.

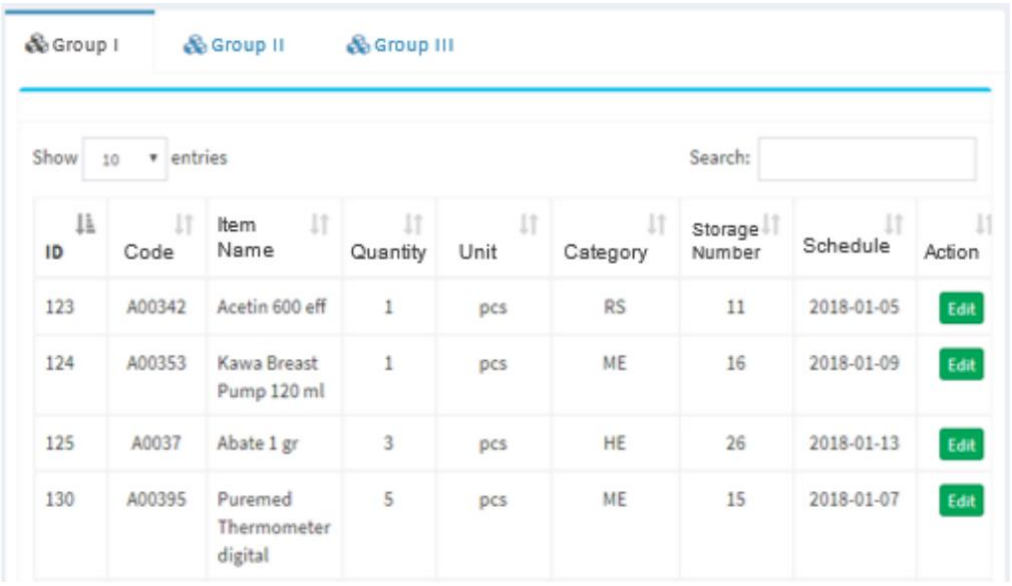

Figure 2.2: Cycle counting Schedule UI

(Fathoni, Ridwan and Santosa, 2019)

2.3.3.2 Scheduling Using Buffer Time Management

There is a method to protect the schedule of cycle counting, which is by using the buffer time management. According to Reyes, Alvarez. and Vasquez (2016), the Theory of Constraints (TOC) is an approach to identify constraints or target levels to improve business performance. Mahtamtama, Ridwan and Santosa (2018) summarised that TOC is basically a philosophy for scheduling and inventory controlling which proposes interdependent links between departments, functions, and resources and weak links between them may cause the potential for better performances to be hidden. Hence, TOC aims to improve those weak links by finding bottlenecks in the system. Often, TOC buffers will be placed before the controlled constraints so that the constraints will never be in shortfall. It is stated by Reyes, Alvarez and Vasquez (2016) that TOC buffers are divided into 5 zones in which 3 of them are the main zones and the rest of the 2 are additional zones for further analysis. Figure 2.3 shows the buffer zones and penetration rates.

< 0%	BLUE	TOO MUCH STOCK
0% - 33%	GREEN	
33% - 66%	YELLOW	
66% - 100%	RED	
> 100%	BLACK	MISS SALE

Figure 2.3: TOC Buffer zones and penetration rates of each zone
(Reyes, Alvarez. and Vasquez, 2016)

Mahtamtama, Ridwan and Santosa (2018) adopted the buffer time management which defined buffer time as the time between the last date cycle counting was performed and the deadline for the next cycle count. Figure 2.4 is the illustration of the buffer time management. The formula for calculating the percentage of remaining buffer time is as follows:

$$\% \text{ of remaining buffer time} = \frac{\text{Buffer time left in days}}{\text{Total buffer time}} * 100\% \quad (2.3)$$

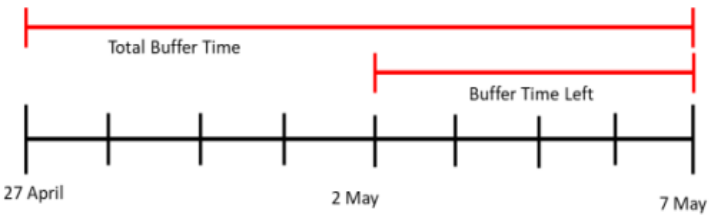

Figure 2.4: Illustration of buffer time management
(Mahtamtama, Ridwan and Santosa, 2018)

Mahtamtama, Ridwan and Santosa (2018) developed the conceptual model of their research as in Figure 2.5, and the data inputs needed to conduct the research are tabulated in Table 2.6. Figure 2.5 shows the problem-solving schematics developed by Mahtamtama, Ridwan and Santosa (2018) which explains the process flow of cycle counting scheduling.

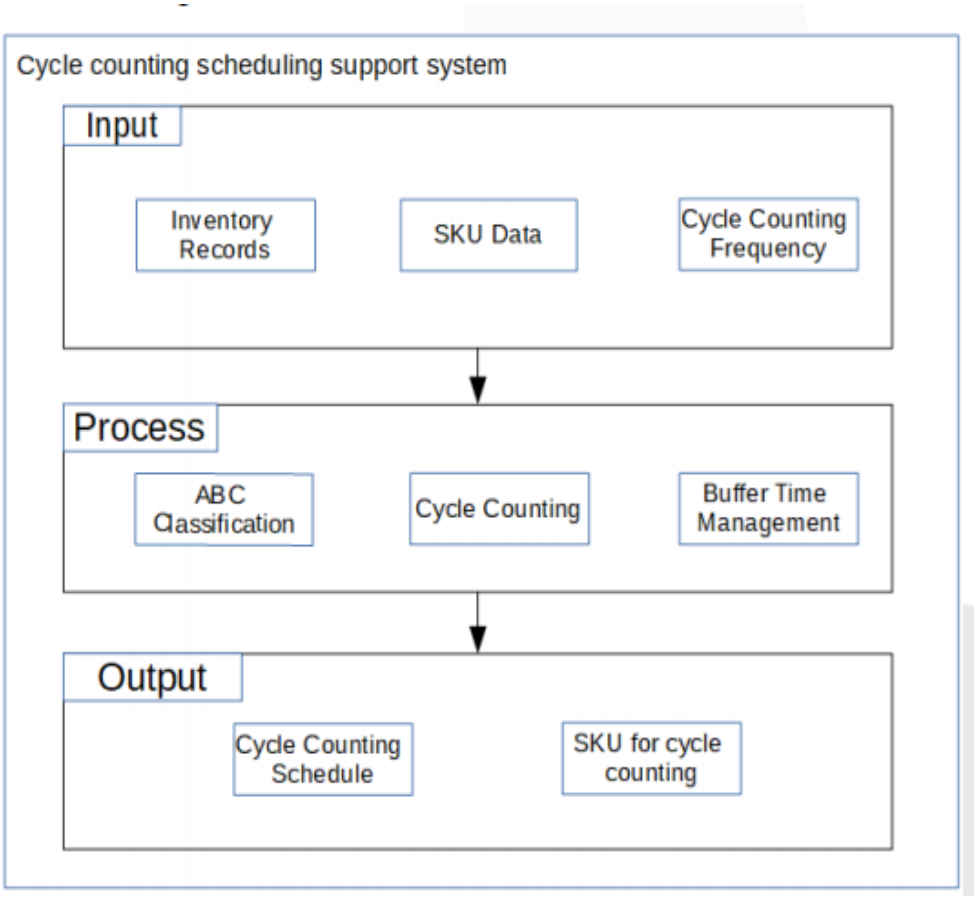

Figure 2.5: Conceptual model of cycle counting scheduling support system (Mahtamtama, Ridwan and Santosa, 2018)

Table 2:6: Necessary data for each input

(Mahtamtama, Ridwan and Santosa, 2018)

Inputs	**Necessary Data**
SKU data	Item category, item code and unit cost
Inventory records	Last stock count data and number of items
Cycle counting frequency	Determined by management

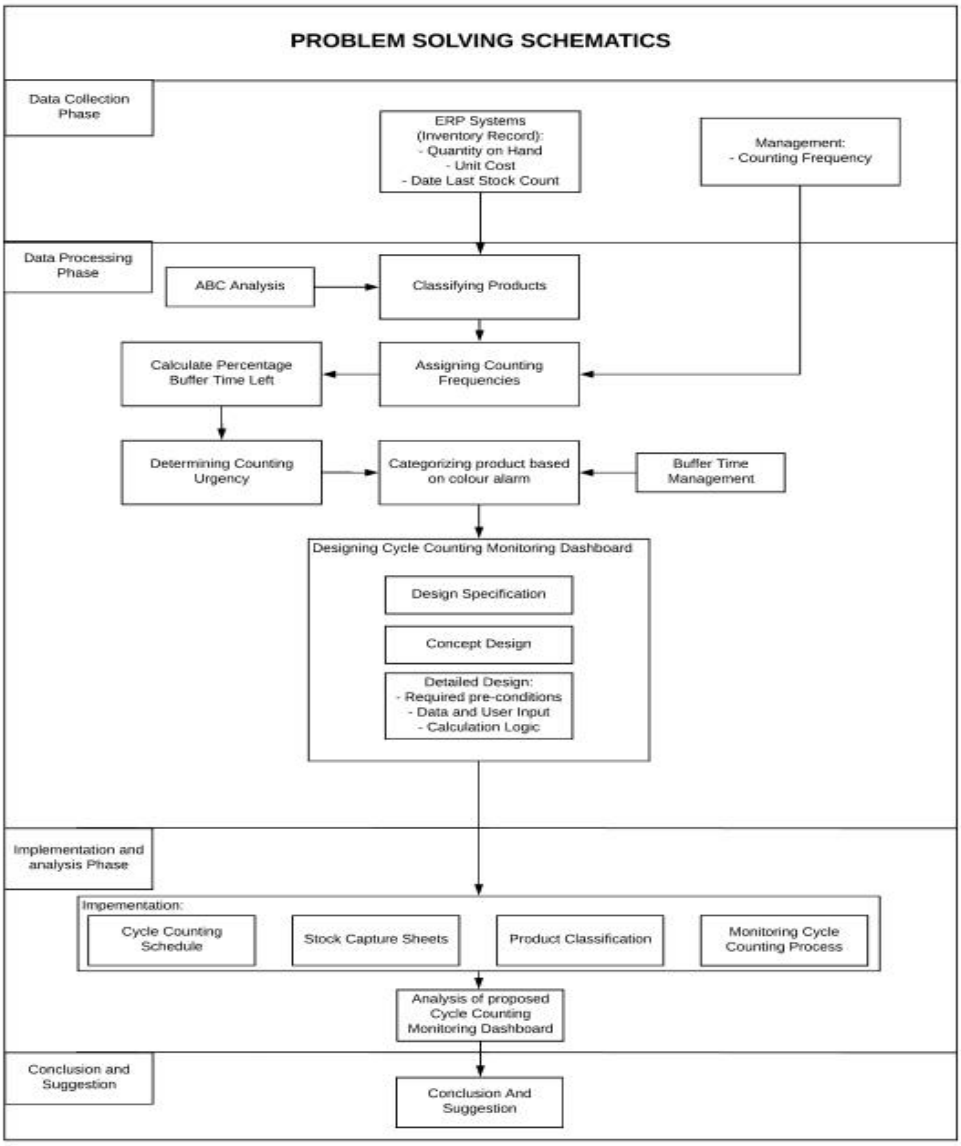

Figure 2.6: Problem solving schematics

(Mahtamtama, Ridwan and Santosa, 2018)

According to Figure 2.6, after the SKUs have been classified using ABC classification, counting frequencies for each class can be assigned like Figure 2.7 to

calculate the next stock count date. Then, the calculation of the percentage of buffer time left can be done and classify the percentage of buffer time left by counting urgency. Figure 2.8 shows the colour coding of each counting frequency class. Lastly, the final categorization result is shown in Figure 2.9. The information in Figure 2.9 provides information on which time to be counted first.

Category	Stock Code	ABC Class	Count Frequency
Cocoa Butter	DF 200_25	C	Quarterly
Cocoa Butter	DF 100	B	Monthly
Cocoa Powder	DF 700-11M_600	A	Weekly
Cocoa Mass	DF 300_20	A	Weekly
Cocoa Mass	DF 3200	A	Weekly

Figure 2.7: Counting frequency for each class

(Mahtamtama, Ridwan and Santosa, 2018)

Counting Urgency	Colour Code
Low	Green
Medium	Yellow
High	Red
Critical	Black

Figure 2.8: Colour coding of counting urgency class

(Mahtamtama, Ridwan and Santosa, 2018)

Category	Stock Code	Count Frequency	Days Between Count	Date Last Stock Count	Stock Count Due Date	Days Left	%Buffer Time Left	Count Urgency
Cocoa Powder	DF 705-11	Weekly	7	7-Jan-18	14-Jan-18	7	100.00%	Low
Cocoa Butter	DF 102	Weekly	7	1-Jan-18	8-Jan-18	1	14.29%	High
Cocoa Mass	DF 304_20	Weekly	7	1-Jan-18	8-Jan-18	1	14.29%	High
Cocoa Powder	DF 700-11BR	Weekly	7	4-Jan-18	11-Jan-18	4	57.14%	Medium
Cocoa Powder	DF 700-11M_600	Weekly	7	7-Jan-18	14-Jan-18	7	100.00%	Low
Cocoa Mass	DF 300_20	Weekly	7	1-Jan-18	8-Jan-18	1	14.29%	High
Cocoa Mass	DF 3200	Weekly	7	4-Jan-18	11-Jan-18	4	57.14%	Medium
Cocoa Powder	DF 720-11	Weekly	7	1-Jan-18	8-Jan-18	1	14.29%	High

Figure 2.9: Final categorization results

(Mahtamtama, Ridwan and Santosa, 2018)

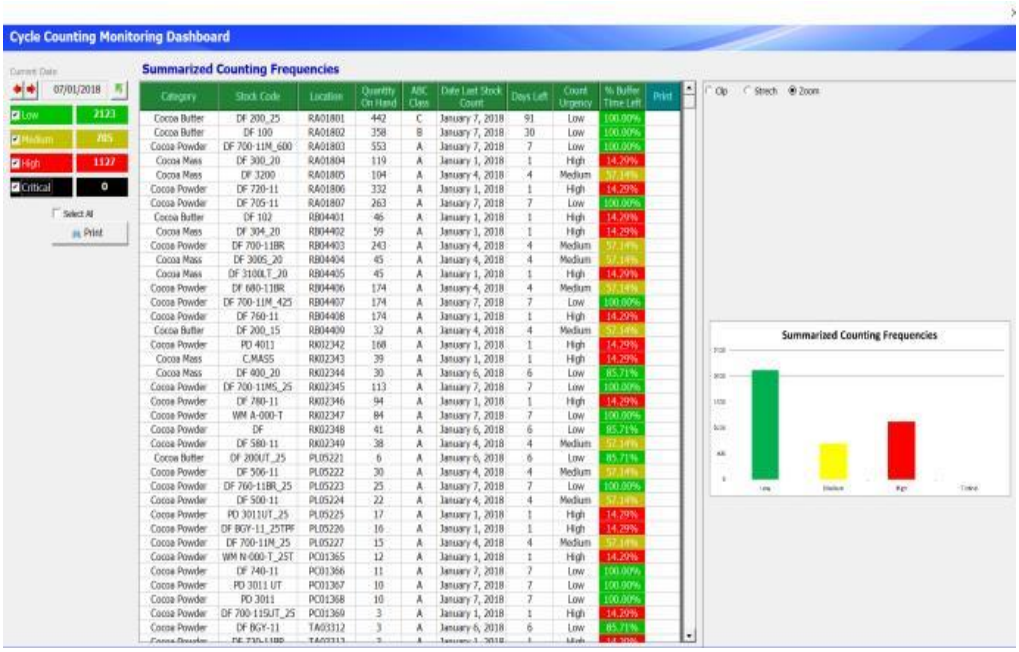

Figure 2.10: Layout of monitoring dashboard

(Mahtamtama, Ridwan and Santosa, 2018)

2.4 Software Development Methodologies

Software development methodologies are different and unique based on the order and focus of each SDLC phase. There are many categories of system development methodologies such as structure design, rapid application development (RAD) and agile development.

2.4.1 Waterfall Method

In the waterfall approach, the key deliverables are very long for each phase and approval from the project sponsor is needed to end the phase and move to the next phase. Although it is possible to go backwards in the SDLC, it is extremely difficult in the waterfall method. The advantage of the waterfall method is that it defines the system requirements long before the coding starts and changes in the requirements during the project progress are lowered. The disadvantage of this methodology is that the design of the system must be completely specified before coding starts and a lot of time is required to deliver the system from the completion of the analysis phase (Dennis, Wixom, and Tegarden, 2015).

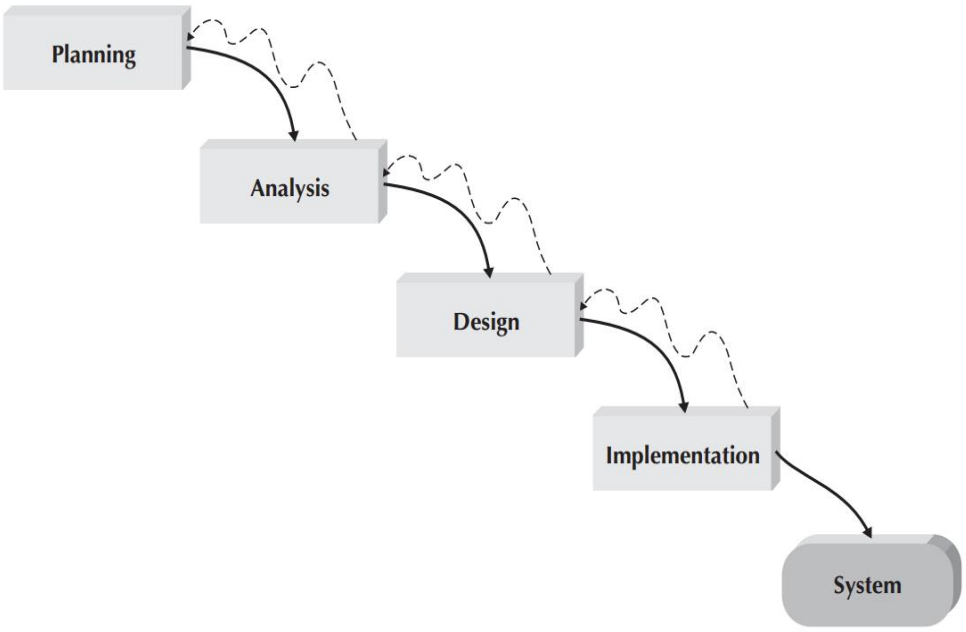

Figure 2.11: Waterfall development approach
(Dennis, Wixom, and Tegarden, 2015)

2.4.2 Rapid Application Development (RAD)

Rapid application development (RAD) divides software designs into four phases, which are requirement planning, user design, construction phase and cutover phase. This methodology solves the disadvantages of the waterfall method mentioned above by adjusting the SDLC phases to deliver some part of the system faster to the project sponsor or client (Dennis, Wixom, and Tegarden, 2015). Sasmito, Wibowo and Dairoh (2020) proved this statement by developing a web-based geographic information system (GIS) using RAD and compared to the older existing system that was developed by the waterfall method. The results showed that the GIS was developed within 84 days using RAD, which is 5 weeks faster than the system developed using the waterfall approach, and with better quality. This is done by the nature of RAD which focuses on the development process and shortens the planning process. Shaydulin and Sybrandt (2017) added that RAD also ensures the incorporation of user feedback into the system earlier. However, they also noticed that RAD might cause poor design because developers seek short-term functionality without paying attention to the technical debt.

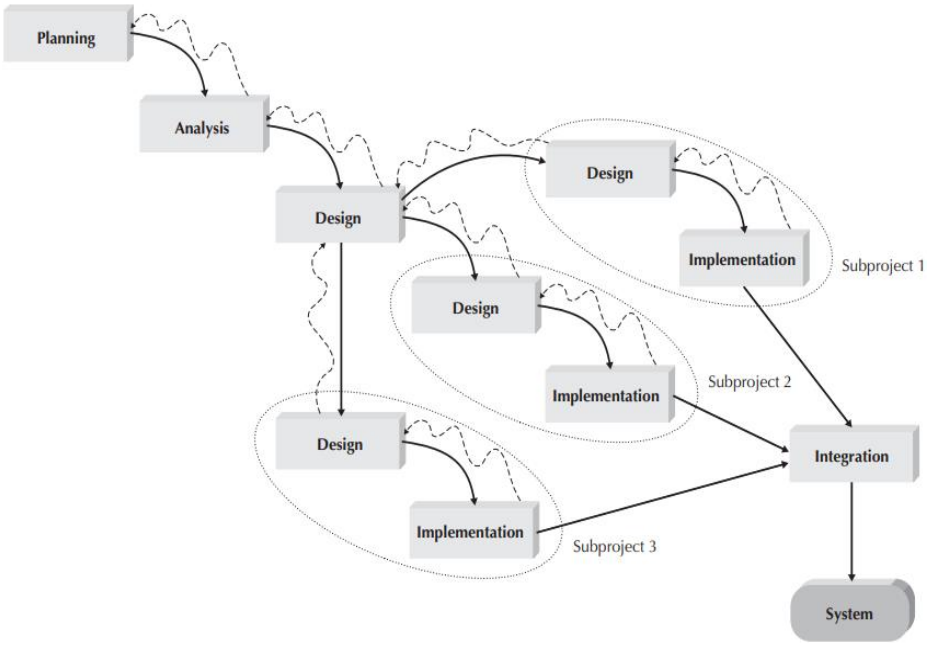

Figure 2.12: RAD approach

(Dennis, Wixom, and Tegarden, 2015)

2.4.2.1 Phased Development Approach

The phased development-based methodology is a kind of RAD, in which it breaks the whole system into several versions and develops it sequentially. The analysis phase will first identify the whole system, project team, users and system sponsors, then, the requirements will be divided into several versions. The most important and basic requirements will be included in the first version. This is supported by the case study of the phased implementation of the Industry 4.0 Learning Factory between 2015 to 2020 done by F Leal, Zancul and Schützer (2021), in which the development of Learning Factories was divided into three phases in which the first phase (2015-2017) is the concept design and pilot which covers the development of the limited demonstration space with students, graduate and the partner companies' support. In the second phase (2018-2019), the structure, equipment and physical space were expanded and the didactic application and research results were achieved. In the third phase (2020), the sustainable operational model was built with the purpose of strengthening relationships with the industry.

The advantage of this methodology is similar to RAD's in a way that it gets a useful system to the hands of users quicker (Dennis, Wixom, and Tegarden, 2015). F Leal, Zancul and Schützer (2021) found out that the phased implementation is

beneficial to Learning Factories because its design and implementation are resource-intensive and complex. However, this methodology also has disadvantages in which the users will begin to use the system that is incomplete, hence, the selection of important requirements for each version especially the first version is crucial (Dennis, Wixom, and Tegarden, 2015).

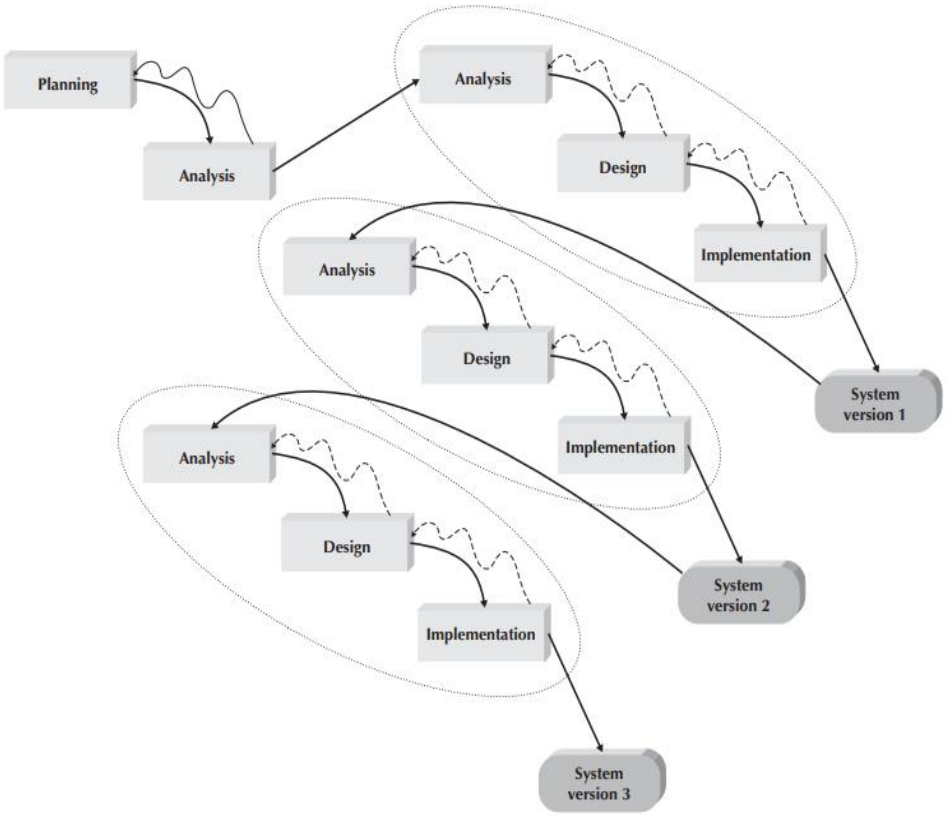

Figure 2.13: Phased development-based approach
(Dennis, Wixom, and Tegarden, 2015)

2.4.3 Agile Development Methodology

Agile development streamlines the software development process by eliminating the modelling and documentation overhead as well as time spent on these tasks. Instead, it emphasises simple and interactive application development. There are a few criticisms of agile development in which it is unrealistic if the development team requires co-location. Secondly, the development process can devolve into a prototyping approach if the agile development is not carefully managed. Thirdly, lack of actual documentation also will cause auditability issues in the system in which the system development and the development process cannot be assured. The last criticism

of agile is based on whether agile development can deliver large systems that are mission-critical (Dennis, Wixom, and Tegarden, 2015).

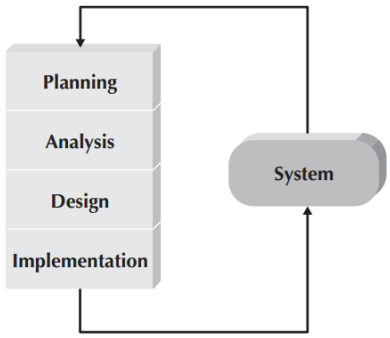

Figure 2.14: Agile development approach
(Dennis, Wixom, and Tegarden, 2015)

2.4.4 Choosing a methodology

The survey conducted by Vijayasarathy, et al. (2016 cited in Shaydulin and Sybrandt, 2017) on 153 developers to describe their software development process listed the seven most commonly used methodology which includes waterfall, agile unified process (AUP), scrum, test-driven development (TDD), rapid application development (RAD), joint application development (JAD) and lastly feature-driven development (FDD). In order to evaluate these seven methodologies, Shaydulin and Sybrandt (2017) defined a set of criteria which evaluates on quality and agility of the methodology as shown in Figure 2.14. On the other hand, Dennis, Wixom, and Tegarden (2015) also tabulated the comparison of software development methodologies which includes structured design methodologies (waterfall and parallel), RAD (phased, prototyping, throwaway prototyping) and agile methodologies (XP and SCRUM) in Figure 2.15.

		Methodology						
		Waterfall	AUP	Scrum	TDD	RAD	JAD	FDD
Quality Criteria	Requirements flexibility	No	Yes	Yes	Yes	Yes	Yes	No
	Requirements fulfillment guarantee	Yes	Yes	Yes	No [1]	Yes	No	Yes
	Cost estimation	Yes	Yes	Yes	No	Yes	No	Yes
	Cost estimates refinement	No	Yes	Yes	No	Yes	No	Yes
	Validation	Yes	Yes [2]	Yes [3]	Yes	Yes	Yes	Yes
	Quick validation	No	Yes [2]	Yes [3]	Yes	Yes	Yes	Yes
	Focus on customer	No	Yes [4]	Yes	No	Yes	Yes	No
	Understandability guarantee	Yes [5]	No	No	No	No	Yes [6]	No
	Technical debt control	Yes	No	No	Yes	No	No	No
Agility Criteria	Prioritizes added value	No	Yes	Yes	Yes	Yes	Yes	Yes
	Allows partial requirements	No	Yes	Yes	Yes	Yes	Yes	Yes
	Focuses on small teams	No	Yes [7]	Yes	Yes	Yes	Yes	Yes
	Develops minimal viable architecture	No	Yes	Yes	Yes	Yes	Yes	Yes
	Produces minimal documentation	No	Yes	Yes	Yes	Yes	No	Yes
	Relies heavily on customer feedback	No	Yes	Yes	No	Yes	Yes	No
	Susceptible to unforeseen risks	No	Yes	Yes	Yes	No	Yes	Yes

Figure 2.15: Methodologies evaluation results

(Shaydulin and Sybrandt, 2017)

Ability to Develop Systems	Structured Methodologies		RAD Methodologies			Agile Methodologies	
	Waterfall	Parallel	Phased	Prototyping	Throwaway Prototyping	XP	SCRUM
With Unclear User Requirements	Poor	Poor	Good	Excellent	Excellent	Excellent	Excellent
With Unfamiliar Technology	Poor	Poor	Good	Poor	Excellent	Good	Good
That Are Complex	Good	Good	Good	Poor	Excellent	Good	Good
That Are Reliable	Good	Good	Good	Poor	Excellent	Excellent	Excellent
With a Short Time Schedule	Poor	Good	Excellent	Excellent	Good	Excellent	Excellent
With Schedule Visibility	Poor	Poor	Excellent	Excellent	Good	Excellent	Excellent

Figure 2.16: Criteria in selecting a development methodology

(Dennis, Wixom, and Tegarden, 2015)

2.5 Conclusion

In conclusion, this literature review covers the similar project review which includes three similar systems on warehouse inventory management systems and findings on the reviewed projects' challenges, features, significance and limitations were done. It was found that paperwork causes inefficiency in record tracking and data duplication and adaptation of an automated inventory management system can reduce the fraud and expenses used in buying paper and ink. Nevertheless, there were some limitations in the project reviews in which the system is not applicable for warehouses with multiple roles and it is also not suitable for large quantities of inventories due to lacking of searching features.

A review on ABC analysis was done to understand the calculation steps for classifying SKUs and ABC-VED analysis was also included to deal with warehouses that have inventories with criticality. Besides, ABC-VED analysis was proven to be

more effective and efficient compared to ABC and VED analysis alone and it covers more SKUs. Scheduling using ABC-VED is different compared to scheduling using buffer time management in a way that ABC-VED calculates the number of SKUs to be counted daily in each category and assigns them to workers who perform cycle counting evenly, whereas buffer time management colour-code the SKUs based on counting urgency and prioritise the counting for SKUs that have highest counting urgency. As the outcome of the literature review in this section, the ABC-VED analysis will be adopted in this project as it is proven to have higher efficiency compared to pure ABC and VED analysis. Additionally, this method will be further improved during implementation by using numerical values to replace the V, E and D classes so that the criticality is no longer restricted to three classes and the classification of SKU will not be limited to nine subclasses as illustrated in Table 2.3. The system can directly assign cycle counting schedules to workers which enables cycle counting to be performed in a more "scheduled manner" instead of following the counting urgency in buffer time management which can cause confusion if more than one worker was assigned to perform cycle counting. Besides, counting all SKUs within the counting frequency (eg. in one month) may not be achieved as the number of SKUs to be counted daily in each category is not fixed in buffer time management.

Last but not least, software development methodologies which include the waterfall model, RAD and agile methods were studied. Further study was done on RAD and phased development which led to a conclusion that the phased development-based approach is the most suitable development methodology for the project.

CHAPTER 3

METHODOLOGY AND WORK PLAN

3.1 Introduction

This chapter will be discussing the methodology and work plan for this project. The adopted development methodology is a phased development-based methodology which will include three phases in which the details of each phase are outlined in this chapter. Besides, the work breakdown structure (WBS) and the Gantt chart are developed for project scheduling. Last but not least, the development tools are selected and described.

3.2 Phased Development-Based Methodology

The phased development-based methodology is chosen as the development methodology of this project. This methodology breaks the development of software into several sets of phases, each handling different aspects of software development. There are five phases in this methodology which are planning, analysis, design, implementation and testing. In this project, the whole system will be divided into three phases as shown in Figure 3.1 in which the first phase includes features with the highest priority. This is to make sure that the main functions of the system are developed first and leave out minor features first. The second phase of development and testing will only be able to start if the development and testing in the first phase have been completed. After the third phase has been done, the whole system has been completed and the closing phase will be carried out for documentation of the project.

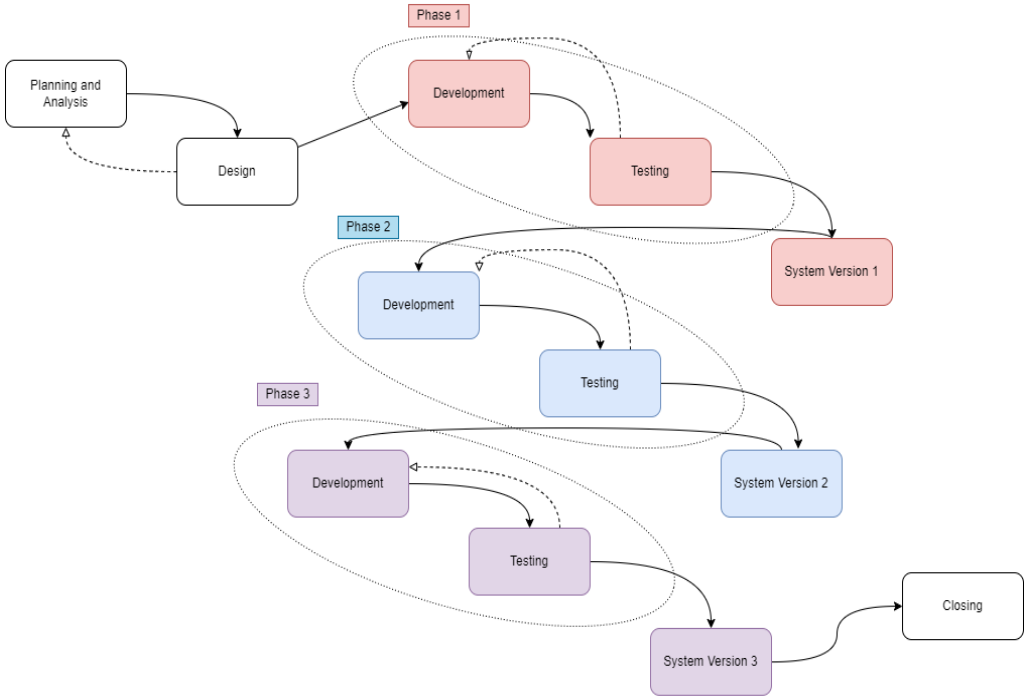

Figure 3.1: Phased development-based approach

3.2.1 Planning and Analysis

This project starts with the planning phase and is followed by the analysis phase. The planning phase will include the identification of problems, objectives and scope, whereas analysis will involve requirement gathering, similar project review and selection of appropriate development tools and development methodology. WBS, which is the key deliverable for project scheduling, will be developed in the planning phase after a sufficient understanding of problems, objectives and project scope.

3.2.1.1 Identify Problems, Project Objectives and Scope

The first task of this project proposal is to identify the problems regarding warehouse inventory management without a warehouse inventory management system. By reviewing the business operations of warehouses, some problems were identified:

1. Inaccuracy of inventory record
2. Difficulties in scheduling the cycle counting progress in the most efficient way
3. Limitation in SKU classification methods

The project objectives were then identified with the purpose of solving each of the problem statements identified above. The objectives are listed below:

1. To develop an application to record all in-and-out of the warehouse inventories from different locations in real-time.
2. To develop an automated scheduler which can categorise SKUs, create schedules, and allocate them to cycle counting personnel.
3. To propose and implement an improvement over existing SKU classification to achieve a higher degree of completeness and efficiency.

The objectives declared above will be the main goals to achieve for this project. With objectives in place, the project scope was also determined in which this system will have three main roles, which are admin, warehouse manager, and warehouse staff. Each role will have different actions to perform using the system. The project scope also covers basic info on what the system will be able to do in three aspects that are in line with the three objectives which include inventory tracking, cycle counting auto-scheduler, and report management.

3.2.1.2 Requirement Gathering and Elicitation

To gather requirements on the warehouse inventory management system, a questionnaire was designed and distributed to targeted users to collect opinions from them. The collected responses will be converted into the functional requirements of this system. Besides, similar projects were also reviewed to analyse their features and limitations so that they can be improved for this project.

3.2.1.2.1 Questionnaire

The questionnaire was designed using a Google Form and distributed to 20 targeted users who work in a warehouse and understand the basic operations in a warehouse. The responses cover age from 21 to 60 years old and 70% of them is male respondents.

There are three sections after the demographic information collection. The first section collects some general information on the respondents' warehouse. The results show that the majority of the respondents already have a warehouse management system in place, and most of them have multiple warehouses at different locations.

The second section is to collect respondents' opinions on the inventory accuracy. Surprisingly, only half of the respondents use a warehouse management

system, while the rest of them still implement Excel or paper recording. Besides, most of the respondents agreed that there are discrepancies between the inventory record and the actual count. Most of them also supported that the inventory count is inconsistent across multiple sites and the inventory data accuracy will affect their business operation in a way that extra time is consumed to investigate the discrepancies and correct the data.

The third section collects respondents' thoughts on the physical count and cycle counting. From the responses collected, there are several significant findings in which more than 90% of the respondents supported that SKU, cycle counting, and calculation of the inventory record accuracy are hard to be done manually. Besides, they also agreed that dealing with physical reports is troublesome.

3.2.1.2.2 Similar Project Review

3 projects on the warehouse inventory management system were reviewed and features developed in the projects reviewed were tabulated in Appendix A. There are no common features that all three systems acquire, however, there were common limitations on the three systems which the system developed did not contain cycle counting features. Besides, there were no searching features in place, which makes the system inconvenient to use if the quantity of inventories becomes large. Moreover, there was no authorization in place for different roles, meaning that no control of access to specific actions was in place. Hence, cycle counting management, the ability to search through inventories and authorization should be included as part of the functional requirements in this project.

Besides limitations, the projects reviewed also proved that implementation of a warehouse inventory system has increased the efficiency of business operations by reducing the frauds that are caused by the paper recording and by gaining faster information access. Besides, the expenses of business were also reduced by eliminating the use of paper and ink.

3.2.1.2.3 Project Scheduling

Project scheduling is also one of the key deliverables of the planning phase. For this phase, a work breakdown structure (WBS) will be developed using a top-down

approach, meaning that high-level tasks can be identified first, and broken down into smaller subtasks. A WBS will contain information such as the duration of the task and the dependency of the task. Task dependencies exist when one task cannot be started or another task has yet to complete. A WBS basically represents all of the small milestones that a project will need to achieve.

Gantt charts will also be developed in this project to present the tasks of WBS in a graphical way. Since this project will be developed using a phased development approach, the Gantt chart will also divide the tasks of development and testing into three phases. Gantt chart helps to illustrate the timeline of the project tasks so that the status of a task can be easily determined eg. ahead of schedule or behind schedule.

3.2.2 Design

After the analysis of the scope has been done, the design phase starts by determining the software architecture as shown in Figure 3.2. A use case diagram will be created to model the system behaviour and identify the interactions between the system and its actors as this project consists of three different roles that can perform different actions. Next, an entity relationship diagram will also be included in the design phase to show relationships between entities of the system to support the set-up of the database in the implementation phase.

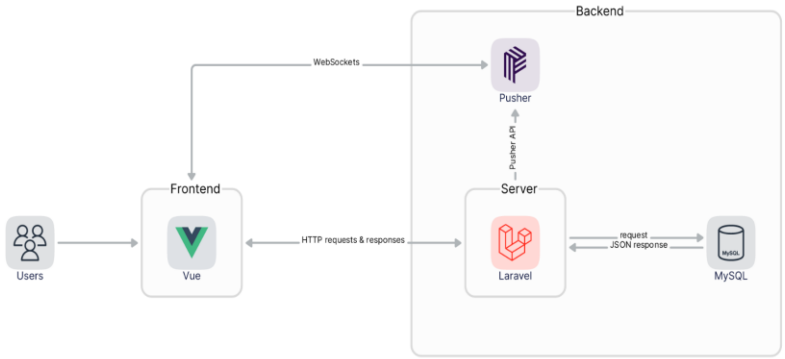

Figure 3.2: System architecture

A prototype is also developed to illustrate the approximate user interface design of the warehouse inventory management system. Throughout the development

of wireframes of the prototype, necessary data columns needed are better visualised. Besides, the developed wireframe also assists in the development and enhancement of the use case description as well. The wireframes will be used as guidance throughout the actual development of the system. All wireframes created for the modules will be included in Chapter 4.

3.2.3 Development and Testing

The development phase starts after the design phase has been completed. In this phase, the development team will start to build the warehouse inventory management system with the support of diagrams provided in the design phase. After the development phase is done, unit testing will be carried out to test out the features developed. In this project, there will be three phases that consist of a development and testing process and each of the phases will contribute to different parts of the system. Upon completion of the three phases, several testing will be carried out to test out the final build.

3.2.3.1 Phase 1

In phase 1, the repository will be set up with Laravel and Vue.js framework. The database will also be set up in which all tables will be created and some dummy data will be inserted into each table for easier testing purposes for later development. Besides, the authentication of users will also be done in this phase so that the authorization of actions can be implemented in later development. This phase will consist of the first project objective for this warehouse inventory management system which is to develop a module that allows checking in and out stock in real-time. Unit testing will be carried out, followed by integration testing to make sure that each module is working fine with the other.

3.2.3.2 Phase 2

After the basic features mentioned in phase 1 have been completed and tested, this phase will include the auto-scheduler for cycle counting, which is the largest module of this system that covers the second and third project objectives. Unit testing and system integration testing will be carried out to make sure that features developed in this phase are working fine and cope well with those developed in phase 1.

3.2.3.3 Phase 3

When all of the features are completed, phase 3 will be completing the reports for checking in and out stock and cycle counting since both of the module has been done in the previous phase. All CRUDs will also be done in this phase. The unit testing and system integration tests must be passed before moving to the closing phase.

3.2.4 Closing

The closing phase will be the phase in which documentation will be done. The documentation will provide a detailed description of the developed system, with snapshots attached. Conclusion with the objectives achievement and limitations of this project will also be included in the documentation.

3.3 Project Plan

This section depicts the whole project plan with a detailed work breakdown structure (WBS) and a Gantt chart.

3.3.1 WBS

1.0 Planning and analysis

- 1.1 Identify problems
  - 1.1.1 Identify background of problems
  - 1.1.2 Develop problem statements
- 1.2 Identify project objectives
  - 1.2.1 Draft out project objectives
  - 1.2.2 Refine project objectives with SMART metrics
- 1.3 Propose solution
  - 1.3.1 Develop system architecture
- 1.4 Decide project approach
  - 1.4.1 Decide research approach
  - 1.4.2 Decide development approach
- 1.5 Develop project scope
  - 1.5.1 Identify target user
    - 1.5.1.1 Decide on user roles
    - 1.5.1.2 Define each roles' responsibilities
  - 1.5.2 Define system scope
    - 1.5.2.1 Define inventory tracking scope
    - 1.5.2.2 Define cycle counting auto scheduler scope
    - 1.5.2.3 Define report management scope
- 1.6 Literature review
  - 1.6.1 Similar project review
    - 1.6.1.1 Review related projects
    - 1.6.1.2 Tabulate features development in each project
    - 1.6.1.3 Identify significance and limitation
  - 1.6.2 Review on cycle counting
    - 1.6.2.1 Cycle counting classification review
      - 1.6.2.1.1 Traditional ABC analysis
      - 1.6.2.1.2 ABC-VED

1.6.2.1.3 Identify strengths and limitation

1.6.2.2 Cycle counting scheduling

1.6.2.2.1 ABC-VED scheduling

1.6.2.2.2 Buffer- time management scheduling

1.6.3 Software development methodologies review

1.6.3.1 Waterfall method

1.6.3.2 Rapid application development (RAD)

1.6.3.3 Agile development methodology

1.6.3.4 Comparison of methodologies

1.7 Requirement gathering and elicitation

1.7.1 Questionnaire

1.7.1.1 Structure questionnaire questions

1.7.1.2 Gather responses

1.7.1.3 Develop facts finding

1.7.1.3.1 Analyse and interpret data collected

1.7.2 Review related projects

1.7.2.1 Identify common features

1.7.2.2 Identify significance of project

1.7.2.3 Identify limitations

1.7.3 Develop requirements specification

1.7.3.1 Develop functional requirements

1.7.3.2 Develop non-functional requirements

1.8 Project planning

1.8.1 Create WBS

1.8.2 Create Gantt chart

1.9 Decide development tools

2.0 Design

2.1 Develop system architecture

2.2 Develop use case diagram

2.3 Develop use case diagram description

2.4 Develop ERD diagram

2.5 Design prototype

3.0 Development and Testing Phase 1

3.1 Setup project

3.1.1 Create project repository

3.1.2 Set up MySQL database

3.1.3 Create tables with dummy data

3.1.4 Connect application to database

3.2 Authentication and authorization of users

3.2.1 Define roles

3.2.2 Define guards, gates, and policies for each role

3.2.3 Develop login module

3.3 Develop inventory tracking module

3.3.1 Develop user interface

3.3.2 Develop algorithm to check in and out inventories

3.3.3 Implement Pusher API

3.4 Testing

3.4.1 Unit testing

3.4.2 Integration testing

4.0 Development and Testing Phase 2

4.1 Develop cycle counting auto scheduler

4.1.1 Develop user interface

4.1.2 Develop scheduling algorithm

4.1.2.1 Retrieve necessary information from database

4.1.2.2 Classify SKUs

4.1.2.3 Schedule SKUs

4.1.2.4 Assign SKUs to responsible staff

4.2 Testing

4.2.1 Unit testing

4.2.2 Integration testing

5.0 Development and Testing Phase 3

5.1 Develop report generation

5.1.1 Develop cycle count report

5.1.1.1 Develop pending approval report

5.1.1.2 Develop summary report

5.1.2 Develop daily report

5.2 Develop CRUD of users

5.3 Develop CRUD of inventories

5.4 Develop CRUD of warehouses

5.5 Develop CRUD of categories

5.6 Testing

5.6.1 Unit testing

5.6.2 Integration testing

6.0 Closing

6.1 Conduct web service testing

6.2 Conduct real time check in/out module testing

6.3 Conduct user acceptance testing (UAT) & usability testing

6.3.1 Prepare test cases

6.3.2 Prepare user satisfaction survey

6.3.2 Decide UAT participants

6.3.3 Conduct UAT

6.3.4 Collect UAT results

6.3.5 Collect and calculate average SUS score

6.4 Design poster

6.5 Finalise project documentation

6.6 Prepare presentation slide

3.3.2 Gantt chart

3.3.2.1 Planning and analysis

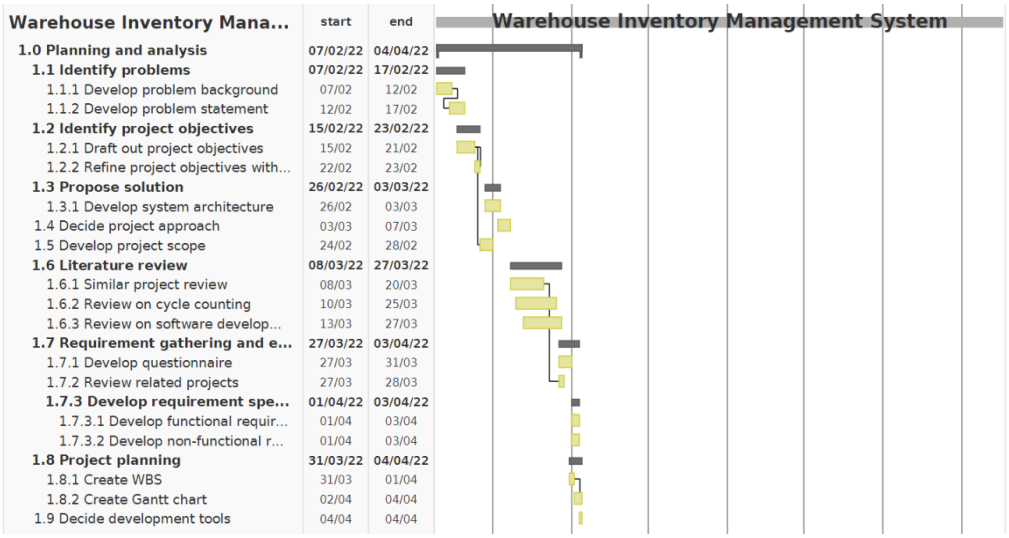

Figure 3.3: Planning and analysis phase timeline

3.3.2.2 Design

Figure 3.4: Design phase timeline

3.3.2.3 Development and testing phase 1

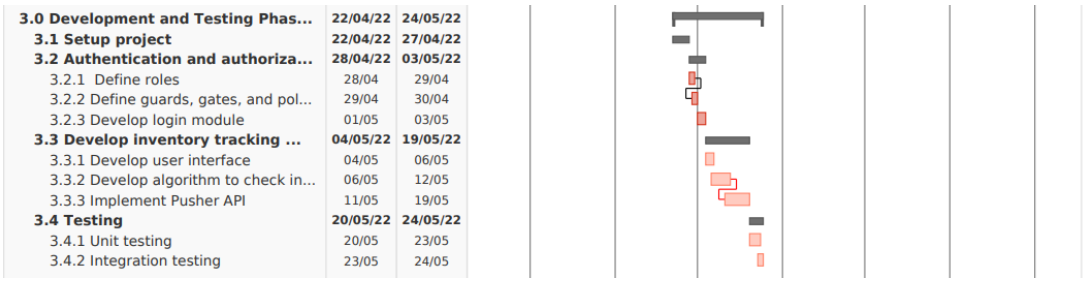

Figure 3.5: Development and testing phase 1 timeline

3.3.2.4 Development and testing phase 2

Figure 3.6: Development and testing phase 2 timeline

3.3.2.5 Development and testing phase 3

Figure 3.7: Development and testing phase 3 timeline

3.3.2.6 Closing

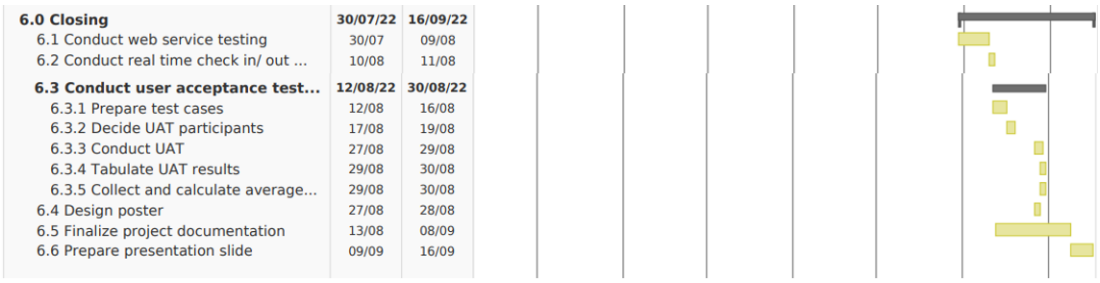

Figure 3.8: Closing phase timeline

3.4 Development Tools

This section outlines all development tools required in developing the implemented system, which includes Visual Studio Code as the IDE tool, Axure RP 10 as the prototyping tool, Vue.js and Laravel for the frontend and backend framework, MySQL database to store data and lastly, GitHub and the SourceTree for the version control of the project.

3.4.1 Visual Studio Code

Visual studio code is an IDE tool that will support the development of this project. This is because Visual Studio Code supports a great number of packages and extensions which will ease the coding process. For example, PHP Intelephense is an extension which is full of essential features that support productive PHP development. Besides, extensions for Vue.js and Laravel which support syntax highlighting features are also crucial to provide faster detection of syntax errors. Besides useful extensions, visual studio code also provides version control features in which conflicts can be solved efficiently and changes made in previous commits can be reverted easily. Last but not least, the file searching feature of visual studio code using Ctrl + P also saves a lot of time and provides convenience to locate a file instead of browsing through the repository for one file.

3.4.2 Axure RP 10

In order to develop the prototype of the warehouse inventory management system, Axure RP 10 is selected to create the high-fidelity prototype as interaction events, conditional logic, working forms and multistate containers are available to be applied to the prototype. Besides, Axure RP is easy to use as the wireframes can be created by using drag-and-drop only.

3.4.3 Vue.js

Vue.js is a progressive JavaScript framework that provides component-based programming which enables user interfaces to be developed efficiently. This framework is lightweight and easy to pick up because no comprehensive understanding of JSX or TypeScript is needed, instead, only basic knowledge of HTML, CSS and JavaScript will be needed to be able to pick up Vue.js. Besides, this framework works perfectly with Laravel.

3.4.4 Laravel

For backend development, Laravel is selected as the PHP framework which uses an MVC design pattern. One of the reasons for choosing a PHP framework rather than the JavaScript framework like Node JS for backend is that the warehouse inventory management system is mainly on data retrieval and manipulation, hence, the PHP framework would be more suitable rather than the JavaScript framework or Python-based backend.

3.4.5 MySQL

MySQL is an open-source, free-to-use, relational database which facilitates the management of databases effectively. It has multiple advantages such as it is stable, and reliable in terms of data security. Besides, it is open source and compatible with a wide range of systems, database models and programming languages.

3.4.6 GitHub

GitHub will be used as the code repository hosting platform for version control of this project. Pushing changes or commits to GitHub periodically is important to account for accidents such as laptop breaking down, project file loss, or malware-attack. Besides, using GitHub, changes from old versions can be reverted if the latest version is buggy.

3.4.7 SourceTree

SourceTree is a free-to-use desktop client that simplifies the interactions with Git repositories. For example, pulling changes or pushing changes to the server can be done in only a few clicks. Besides, all codes in each commit on any branch can be visualised clearly to keep track of the changes. By using SourceTree, we can decide on which line, hunk or file to be committed to the server by staging or discarding it. Most importantly, by eliminating the use of git commands, we can focus more on the coding part.

3.5 Conclusion

In conclusion, the phased development methodology is chosen as the software development approach for this project and tasks for each phase are specified. In this

section, a work breakdown structure and Gantt chart are created to visualise the timeline of the whole project. The project is starting from 7th February 2022 to 16th September 2022, with a total duration of 221 days. Last but not least, Visual Studio Code, Axure RP9, Vue.js, Laravel, MySQL, GitHub and SourceTree are chosen as the development tools for this project.

CHAPTER 4

PROJECT SPECIFICATION

4.1 Introduction

After the requirement gathering and elicitation, the initial specification is determined in this chapter. First of all, facts findings on the questionnaire responses are developed. Then, a use case diagram and use case description are prepared to illustrate how actors interact with the warehouse inventory management system. An entity-relationship diagram (ERD) is also created to show the relationships between entities of the system. Lastly, a prototype is developed to draft the user interfaces of the system.

4.2 Facts Finding

A questionnaire was distributed to targeted users who work or understand the operations of warehouses to understand more about the software requirements and a total of 20 responses were collected. There are a total of 4 sections which include demographic information, general information, inventory accuracy and the physical count or cycle counting.

4.2.1 Demographic Information

The first section is to collect demographic information from the respondents. The responses cover ages ranging from 21 to 60 years old. Figure 4.1 shows that most of the respondents are in the range of 41 to 50 years old, which contributes to 40% of the total respondents, followed by 31 to 40 years old, which covers 35% of total respondents. The gender distribution of the respondents of the questionnaire is 70% of male respondents and 30% of female respondents, as shown in Figure 4.2.

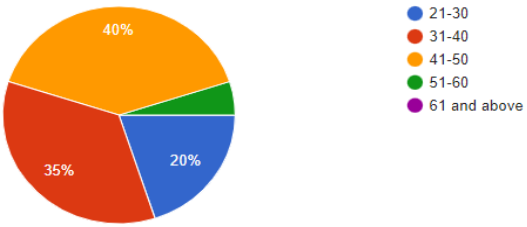

Figure 4.1: Age of 20 respondents

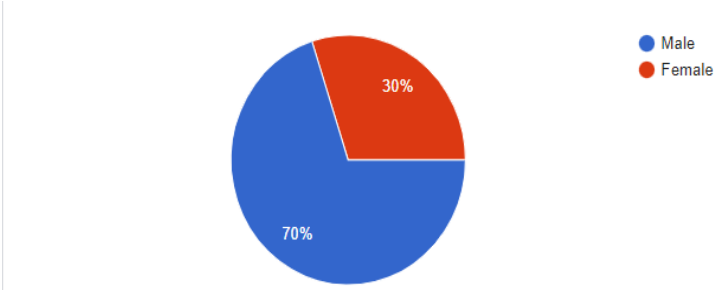

Figure 4.2: Gender of the respondents

4.2.2 General Information

The second section of the questionnaire aims to collect some general information regarding the warehouse management of the respondents. The first question is to ask whether the respondents' warehouse(s) has a warehouse management system already in place. Figure 4.3 shows that most of the respondents (70%) are already using a warehouse management system to manage their warehouse. However, there are still a significant number (30%) of respondents who have yet to implement a warehouse management system in their warehouse.

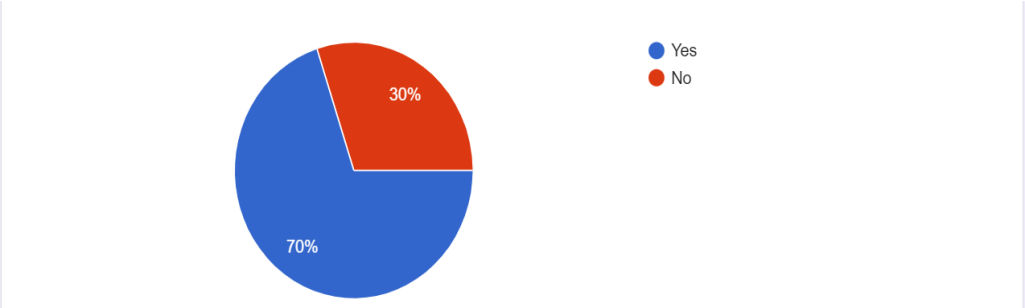

Figure 4.3: Statistic of respondents to have warehouse management system in place in their warehouse or not

The second question is asking whether the respondent has more than one warehouse from multiple locations and the result in Figure 4.4 shows that 65% of the respondents have multiple warehouses at different locations.

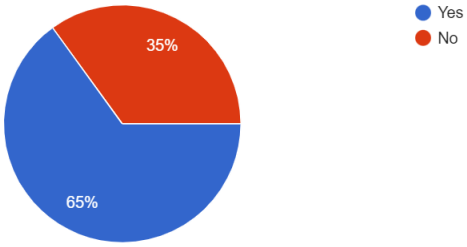

Figure 4.4: Distribution of respondents who has more than one warehouse from multiple locations or not

4.2.3 Inventory Accuracy

This section aims to collect information on the accuracy of inventories in the respondent's warehouse(s). The first question asks respondents about their method to record inventory and the results show that only half of the respondents use a warehouse management system or similar software to record their inventory, whereas another 40% of the respondents use Excel or online spreadsheets for inventory recordings and 10% of the respondents still have paper records, as shown in Figure 4.5.

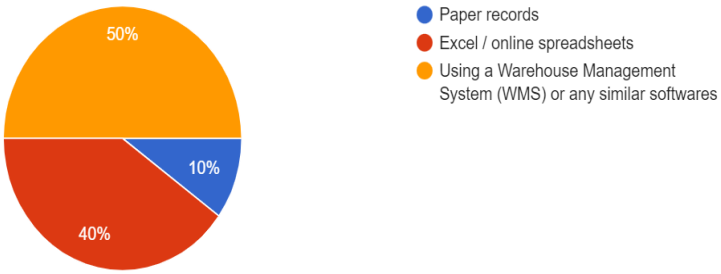

Figure 4.5: Methods of inventory recording of respondents

The second question asked whether there are any discrepancies between the inventory record and the actual count and the results are shown in Figure 4.6. Surprisingly, a high percentage of 75% of respondents responded that the inventory record does not match the actual count of the inventory.

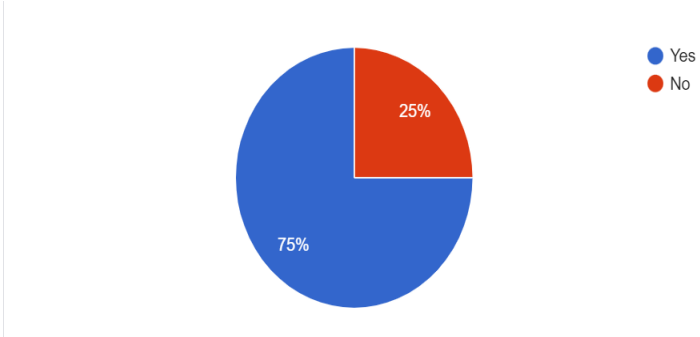

Figure 4.6: Statistic of whether any discrepancies occur between inventory record and actual count

The third question also asked if the inventory count is inconsistent across multiple locations. The result in Figure 4.7 shows that 80% of the respondents responded yes to this question, which means that most of the respondents are facing the issue of having the same inventory to have different counts recorded in the system from multiple sites.

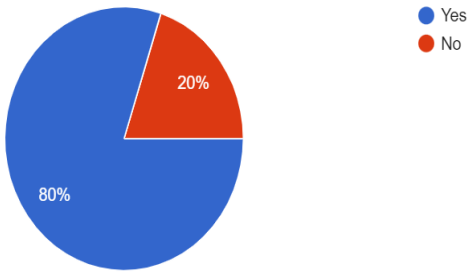

Figure 4.7: Statistic of whether any inconsistency of inventory count occurs across multiple location

The fourth question asked respondents their opinion on whether inventory data accuracy affects business operation. Figure 4.8 shows that 90% of the respondents agreed that data accuracy would affect their normal business operations. The fifth question further confirmed with the respondents why they think that it would affect them. 89.5% of the respondents agreed that inventory accuracy would affect business operation in a way that additional time would be needed to research the discrepancies and correct the data. The results are shown in Figure 4.9.

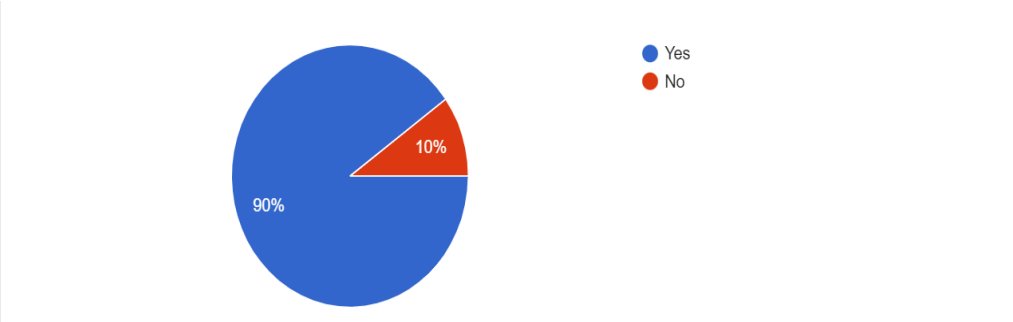

Figure 4.8: Statistic of whether inventory and data accuracy would affect the business operation

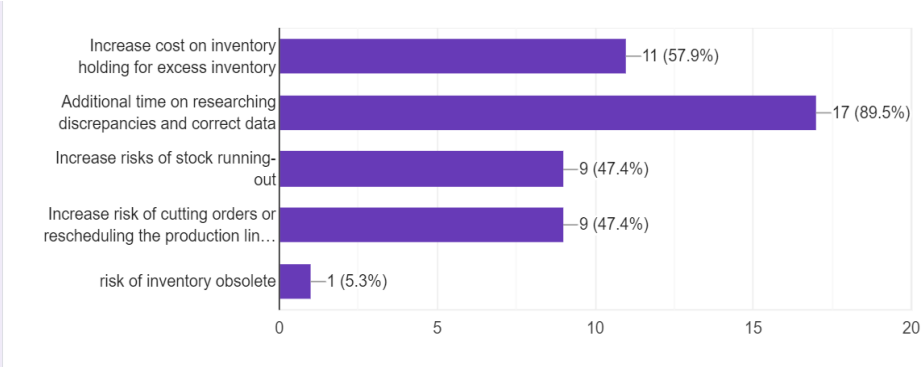

Figure 4.9: Possible reasons on why data accuracy would affect business operation

4.2.4 Physical Count / Cycle Counting

This is the last section of the questionnaire and it aims to collect the opinions of respondents on cycle counting and inventory controls. As the first question, the inventory control method implemented by respondents is collected. From Figure 4.10, it is shown that half of the respondents utilise both physical inventory count and cycle counting as their inventory control methods.

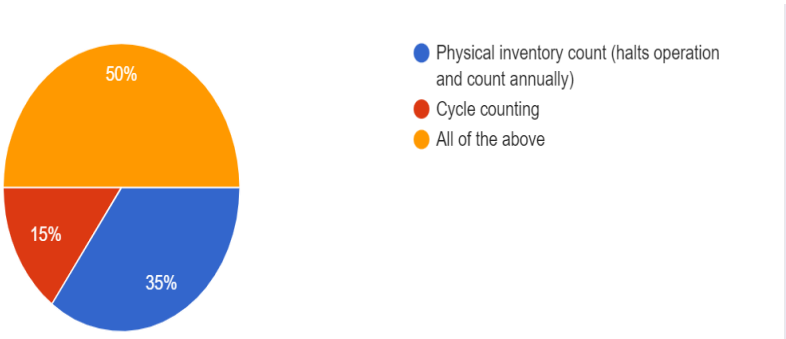

Figure 4.10: Distribution of inventory control methods implemented in respondents' warehouse(s)

From question 2 to question 9, respondents were given a statement in each question and were asked whether they agree or disagree on a scale of 1 to 5, in which 1 means strongly disagree and 5 is strongly agree. For easier illustration purposes, the responses for questions 2 to 9 are tabulated in Table 4.1.

Table 4:1: Summary of responses on physical counting or cycle counting

No	Questions	1 (Strongly disagree)	2 (Strongly agree)	3 (Neutral)	4 (Agree)	5 (Strongly Agree)
1	Manual inventory recording/ counting without a WMS system is troublesome.	-	-	5%	40%	55%
2	It is difficult to classify SKUs manually.	-	-	10%	50%	40%
3	Extra manpower and resources will be needed to analyse and schedule for physical counting and cycle counting.	-	-	15%	30%	55%
4	It is difficult to keep track of the cycle counting progress manually.	-	-	5%	40%	55%
5	Continuous operations and incoming transactions while performing physical count/cycle count may cause false variances in inventory counts.	-	5%	-	35%	60%

Table 4.1 (Continued)

6	There will be a lot of hassle if were to deal with all of the physical reports regarding physical counts/ cycle counting.	-	5%	5%	35%	55%
7	Adjusting inventory records should be in real-time to avoid inconsistencies.	-	-	5%	35%	60%
8	Extra effort will be needed to calculate inventory record accuracy (IRA) to keep track of the performance.	-	-	5%	45%	50%

From Table 4.1, the first question to the fourth question proved that most respondents agreed that doing inventory recording, classifying SKU, scheduling cycle counting, and keeping track of cycle counting progress manually is inefficient and troublesome. For the fifth question, 95% of the respondents agreed that continuous and incoming transaction while performing physical count or cycle counting will cause variances in inventory count. Question 6 showed that 90% of the respondents supported those physical reports management is inefficient. Question 7 evidenced that 95% of respondents want inventory records to be updated in real time. In the last question, 95% of respondents agreed that extra efforts are needed to calculate IRA.

4.2.5 Summary of survey

From the survey, several important points on the requirements of the warehouse inventory management system were obtained from the respondents. Firstly, it is proven that most of the respondents are facing the issue of discrepancies between inventory records and actual count, as well as the discrepancies between inventory records across multiple locations and these issues affect their business operation in various ways as stated above. Hence, warehouse inventory management systems should be able to update the stock data in real time, and cycle counting features should be included. Secondly, respondents would like to have inventory recording, SKU classification, cycle counting scheduling, and cycle counting progress monitoring to be done automatically by the system. Thirdly, the warehouse inventory management system should be able to manage reports and calculate IRA automatically.

4.3 Requirement Specification

This section depicts the system requirement specification, which can be classified into two main categories which are functional requirements and non-functional requirements.

4.3.1 Functional Requirement

A system must adhere to the formalized functional requirements in order to suit the needs of the end user. All of these features must be included in the system as stipulated by the contract. These are depicted or stated as the system's required input, the action taken, and the desired result. System functional requirements are essentially the user-stated requirements that are evident in the finished product, as opposed to non-functional needs (Zhou, 2004). In this project, the functional requirements are collected from the similar projects review in the literature review sections and from the questionnaire responses for requirements elicitation in Section 4.2. The functional requirements are divided into three different roles: admin, warehouse manager and staff as shown in Table 4.2.

Table 4:2: Functional requirements by roles

Roles	Functional Requirements
Admin	1. The system should allow admin to view, add, update and delete the inventory of a warehouse. 2. The system should allow admin to view, add, update, and delete a warehouse from the system. 3. The system should allow admin to view, add, update and delete a user in the system. 4. The system should allow admin to view, add, update and delete a category from the system. 5. The system should allow admin to view daily reports of in-and-out inventories of all warehouses. 6. The system should allow admin to view the summary report for cycle counting of all warehouses.
Warehouse manager	7. The system should allow managers to start a new cycle counting process. 8. The system should allow managers to view all upcoming cycle counting SKUs. 9. The system should allow managers to view current cycle counting settings.

	10. The system should allow a manager to view daily reports of in-and-out inventories of his or her warehouse only. 11. The system should allow managers to view summary reports for cycle counting of his or her warehouse only. 12. The system should allow managers to view, approve or reject cycle count pending approval reports. 13. The system should allow managers to view all staffs in the warehouse.
Staff	14. The system should allow staff to check in-and-out of the inventory stock. 15. The system should allow staff to view cycle count schedules that are assigned to them. 16. The system should allow staff to perform cycle counting on the SKU assigned.
Manager and staff	17. The system should allow the manager and staff to view all inventories in the system.
Admin, manager and staff	18. The system should allow admin, manager and staff to login into their account.
System	19. The system should calculate inventory record accuracy (IRA) of each SKU and generate summary reports for each cycle count input from staff. 20. The system should classify SKUs, calculate the number of SKUs to be counted daily for each class, generate schedules and assign to staff available evenly.

4.3.2 Non-Functional Requirement

The non-functional requirement describes a system performance feature. It encompasses all requirements that are not covered by functional requirements. They define criteria that determine the functionality of a system rather than specific behaviour (Chung, Nixon, Yu and Mylopoulos, 2012). The IEEE-Std 830 – 1993 (IEEE Computer Society, n.d.) lists 13 non-functional requirements to be included in a Software Requirements Document:

1. Performance requirements
2. Interface requirements
3. Operational requirements
4. Resource requirements
5. Verification requirements
6. Acceptance requirements
7. Documentation requirements
8. Security requirements
9. Portability requirements
10. Quality requirements
11. Reliability requirements
12. Maintainability requirements
13. Safety requirements

In this project, we focus on three main non-functional requirements, which are the performance (usability) requirement, security requirements and verification requirements.

4.3.2.1 Performance-(usability) Requirement

1. The system should ask for confirmation for every destructive actions.

4.3.2.2 Security Requirement

1. The system should protect users' credentials in which passwords should be encrypted.
2. The system should not be accessed by an unauthorised third party.

4.3.2.3 Verification Requirement

1. The users of the warehouse inventory management system should authenticate themselves using username and password.

4.4 System Use Case

Use cases are a list of actions that define the interaction between the user (admin, warehouse manager and staff) and the implemented system. System use cases detailed specific processes inside the implemented system to reach users' goals. It is used in the analysis phase to identify, describe and clarify the functional requirements from the end users' perspective, including the dependencies among use cases as well. In this section, a use case diagram will be shown along with the use case description for each use case.

4.4.1 Use Case Diagram

Figure 4.11 shows the use case diagram of the warehouse inventory management system.

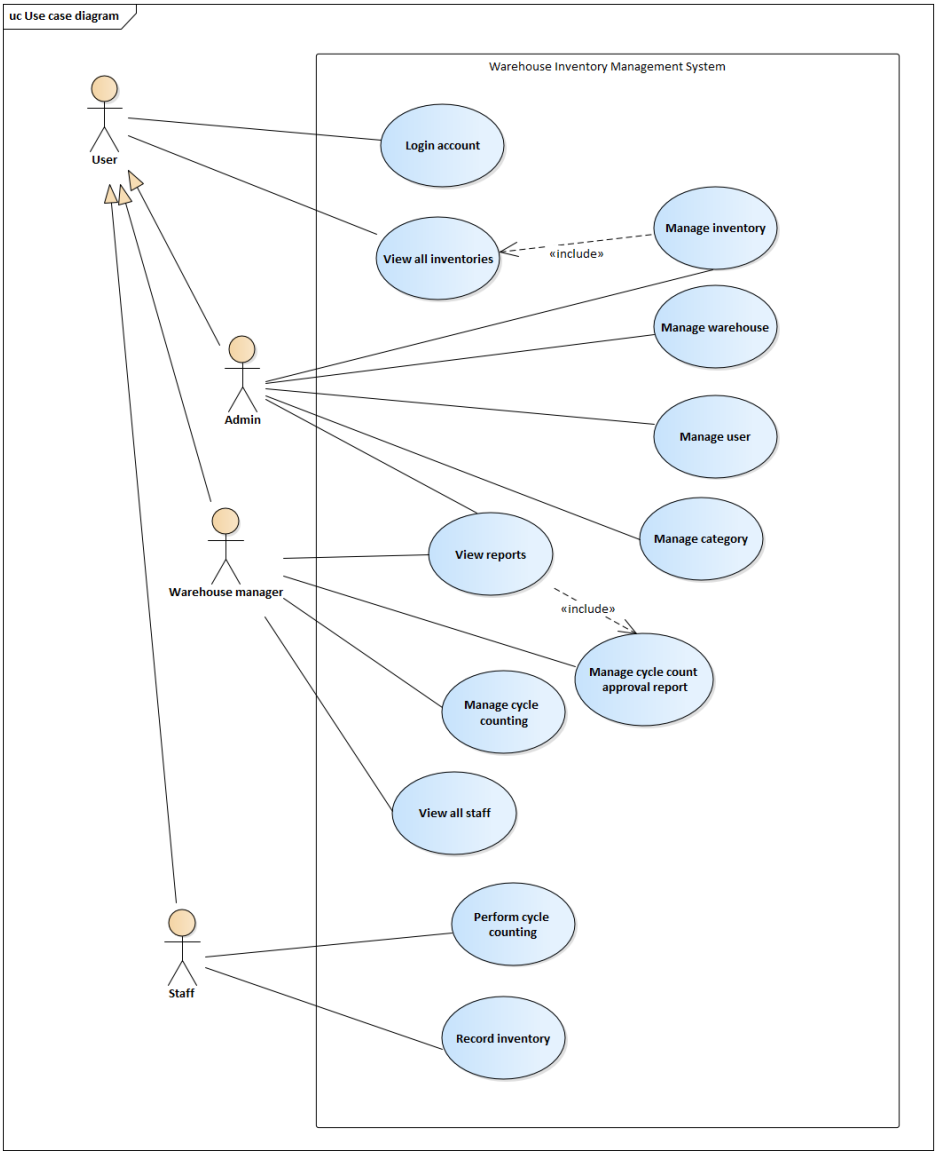

Figure 4.11: Use case diagram

4.4.2 Use Case Description

Table 4:3: Use case description of login account

<table>
<tr><td colspan="2">Use Case Name: Login account</td><td>ID: 1</td><td>Importance Level: High</td></tr>
<tr><td colspan="2">Primary Actor: User (admin, warehouse manager, staff)</td><td colspan="2">Use Case Type: Detail, essential</td></tr>
<tr><td colspan="4">Stakeholders and Interests:
User - wants to login their account and access the system.</td></tr>
<tr><td colspan="4">Brief Description: This use case describes how users login their account before accessing the system.</td></tr>
<tr><td colspan="4">Trigger: When user wants to access the system.</td></tr>
<tr><td colspan="4">Relationships:
Association : User
Include : -
Extend : -
Generalization : -</td></tr>
<tr><td colspan="4">Normal Flow of Events:
1. The user opens the application.
2. The user enters their username and password to login the account.
3. The application verifies the username and password.
4. The system displays the homepage depending on the user's role.</td></tr>
<tr><td colspan="4">Sub-flows:</td></tr>
<tr><td colspan="4">Alternate/Exceptional Flows:
3a. If the username and password is incorrect or mismatched, the system will prompt the user to re-enter again.
4a. If users log in their account for the first time, a message prompt for setting their password will be displayed.
1. The system replaces the temporary password with the new password entered by the user.</td></tr>
</table>

Table 4:4: Use case description of manage inventory

Use Case Name: Manage inventory	ID: 2	Importance Level: High
Primary Actor: Admin	Use Case Type: Detail, essential	
Stakeholders and Interests: Admin - wants to view, add, delete and update the inventory of a warehouse.		
Brief Description: This use case describes how admin view, add, delete and update inventory of a warehouse.		
Trigger: New type of inventory arrives one of the warehouse or inventory details such as price changes or inventory is removed from the warehouse.		
Relationships: Association : Admin Include : View inventories Extend : - Generalisation : -		
Normal Flow of Events: 1. Admin login into the application. 2. Admin chooses to manage inventory from the dashboard. 3. The system executes view all inventories use case as the homepage of manage inventory module. 4. Admin wants to add, update or delete an inventory. 4.1. If admin wants to add new inventory to one or more warehouse The S-2: add inventory subflow is performed. 4.2. If admin wants to update an inventory The S-3: update inventory subflow is performed. 4.3. If admin wants to delete an inventory The S-4: delete inventory subflow is performed.		
Sub-flows: S-2: Add new inventory 1. Admin wants to add new inventory. 2. System displays a modal pop up to prompt input. 3. Admin enters inventory name, starting stock, cost per unit, category priority and warehouse name. 4. System auto-generates id and adds the new inventory to the database. 5. System shows inventory successfully added message. S-3: Update inventory 1. Admin selects an inventory to update from the inventory list. 2. System displays update inventory modal pop-up to prompt input. 3. Admin edit the inventory name, cost per unit, category and priority.		

4. System updates the inventory with new inputs by admin.

S-4: Delete inventory

1. Admin selects an inventory to delete from the inventory list.
2. System displays delete inventory modal pop-up for confirmation.
3. Admin confirms to delete the inventory.
4. System soft-deletes the inventory and back to manage-inventory page.

Alternate/Exceptional Flows:

S-3, 4a. If the user changed the category of the inventory, the storage bin for the inventory will also be changed.

1. The system will check if the new category's storage bins are full or not.
2. If all bins for the new category is full, the system will ask the user to empty a bin for the new category first, else, the system will assign the inventory to the empty bin.
3. System will show an alert showing the new storage bin number.

Table 4:5: Use case description of view all inventories

Use Case Name: View all inventories	ID: 3	Importance Level: High
Primary Actor: User (Admin, warehouse manager, staff)	Use Case Type: Detail, essential	
Stakeholders and Interests: User - wants to view all inventories of the warehouse.		
Brief Description: This use case describes how admin, warehouse manager, and staff view all inventories in the warehouse.		
Trigger: Admin, warehouse manager or staff wants to view all inventories.		
Relationships: Association : User Include : - Extend : - Generalization : -		
Normal Flow of Events: 1. The user logins into the application. 2. The user chooses to view all inventories. 3. System displays inventory list 3.1. If the user is admin, the system displays all inventories of all warehouses. 3.2. If the user is a warehouse manager or staff, the system displays all inventories of their warehouse. 4. The user can select one of the inventories to view details. 5. The system returns inventory details which include the transaction history of that inventory.		
Sub-flows:		
Alternate/Exceptional Flows: 3.1a: Admin selects a particular warehouse to view its inventories. 1. Admin enters the desired warehouse to filter. 2. The system filters the inventory list and returns inventories belonging to the selected warehouse only.		

Table 4:6: Use case description of manage warehouse

Use Case Name: Manage warehouse	ID: 4	Importance Level: High
Primary Actor: Admin	Use Case Type: Detail, essential	
Stakeholders and Interests: Admin - wants to add, view, update and delete warehouse		
Brief Description: This use case describes how admin view, add, update and delete warehouse.		
Trigger: Admin wants to manage the warehouse.		
Relationships: Association : Admin Include : - Extend : - Generalization : -		
Normal Flow of Events: 1. The admin logins the application. 2. The admin chooses to manage the warehouse. 3. The system displays all warehouses. 4. The admin can view, add, update or delete a warehouse. 4.1. The admin wants to view a warehouse's details. S-1: View warehouse detail subflow is performed. 4.2. The admin wants to add a new warehouse. S-2: Add new warehouse subflow is performed. 4.3. The admin wants to update the warehouse. S-3: Update a warehouse subflow is performed. 4.4. The admin wants to delete the warehouse. S-4: Delete a warehouse subflow is performed.		
Sub-flows: S-1: View warehouse detail 1. Admin selects desired warehouse to view from the list. 2. System displays the warehouse details and all staff in the warehouse. S-2: Add new warehouse 1. Admin wants to add a new warehouse. 2. System displays a modal pop up to prompt input. 3. Admin enters name, location, number of bins and zones. 4. System creates the storage bins and add the new warehouse into the database. S-3: Update a warehouse 1. Admin wants to update a warehouse. 2. System displays a modal pop up to prompt input.		

3. Admin edit the location or warehouse manager. 4. System updates the warehouse details. S-4: Delete a warehouse 1. Admin wants to delete a warehouse. 2. System displays a confirmation modal to prompt for admin's confirmation. 3. Admin confirms to delete the warehouse. 4. System deletes the warehouse.
Alternate/Exceptional Flows:

Table 4:7: Use case description of manage user

Use Case Name: Manage user	ID: 5	Importance Level: High
Primary Actor: Admin	Use Case Type: Detail, essential	
Stakeholders and Interests: Admin - wants to view, add, update or delete a user.		
Brief Description: This use case shows how admins view, add, update or delete a user.		
Trigger: Admin wants to view, add, update or delete a user.		
Relationships: Association : Admin Include : - Extend : - Generalization : -		
Normal Flow of Events: 1. The admin logins the application. 2. The admin chooses to manage the user. 3. The system displays all users. 4. The admin can view, add, update or delete a user. 4.1. The admin wants to view a user's details. S-1: View user detail subflow is performed. 4.2. The admin wants to add a new user. S-2: Add new user subflow is performed. 4.3. The admin wants to update the user. S-3: Update user subflow is performed. 4.4. The admin wants to delete the user. S-4: Delete user subflow is performed.		
Sub-flows: S-1: View user detail subflow 1. Admin selects the desired user to view from the list. 2. System displays the user details. S-2: Add new user subflow 1. Admin wants to add a new user. 2. System displays a modal pop up to prompt input. 3. Admin enters name, email, contact number, IC number, role, warehouse, employed in, address and a username. 4. System will use the phone number as a temporary password and create the user into the database. S-3: Update user subflow 1. Admin wants to update a user.		

2. System displays a modal pop up to prompt input. 3. Admin enters a new email, contact number, role, warehouse or address. 4. System updates the user details. S-4: Delete user subflow 1. Admin wants to delete a user. 2. System displays a confirmation modal to prompt for admin's confirmation. 3. Admin confirms to delete the user. 4. System change the user's status to inactive.
Alternate/Exceptional Flows:

Table 4:8: Use case description of view reports

Use Case Name: View reports	ID: 6	Importance Level: High
Primary Actor: Admin, warehouse manager	Use Case Type: Detail, essential	
Stakeholders and Interests: Admin - wants to view daily reports and cycle counting summary reports of all warehouses. Warehouse manager - wants to view daily reports and cycle counting summary reports of his warehouse.		
Brief Description: This use case describes how admin and warehouse manager view daily reports and cycle counting summary reports.		
Trigger: Admin or warehouse manager wants to view daily reports or cycle counting summary reports.		
Relationships: Association : Admin, warehouse manager Include : Manage cycle counting approval report Extend : - Generalization : -		
Normal Flow of Events: 1. Admin or warehouse manager logins into the application. 2. Admin or warehouse manager wants to view reports. 2.1. Admin or warehouse manager wants to view daily reports S-1: View daily report subflow is performed. 2.2. Admin or warehouse manager wants to view summary reports S-2: View summary report subflow is performed. 2.3. Warehouse manager wants to manage the cycle counting approval report. S-3: Manage cycle count approval report use case is performed.		
Sub-flows: S-1: View daily report subflow 1. Admin or warehouse manager chooses to view daily reports. a. If the admin wants to view daily reports i. System displays all daily reports of all warehouses ii. The admin wants to select specific warehouse to view iii. System filters and displays daily reports for that warehouse only. b. If the warehouse manager wants to view daily reports i. System displays the daily reports of warehouse manager's warehouse only 2. Admin or warehouse manager can select one of the reports displayed in the listing for stock in or stock out details. 3. System displays all stock in or stock out details for the selected report. S-2: View summary report subflow		

4. Admin or warehouse manager chooses to view summary reports.
    a. If the admin wants to view summary reports
        i. System displays all summary reports of all warehouses
    b. If the warehouse manager wants to view summary reports
        i. System displays all cycle counting summary reports of warehouse manager's warehouse only
5. Admin or warehouse manager can select one of the SKU ID displayed in the listing for summary report details.
6. System displays a modal popup to display the summary report details.

Alternate/Exceptional Flows:

Table 4:9: Use case description of manage cycle counting approval report

Use Case Name: Manage cycle counting approval report	ID: 7	Importance Level: High
Primary Actor: Warehouse manager	Use Case Type: Detail, essential	
Stakeholders and Interests: Warehouse manager - wants to manage cycle counting approval reports of his warehouse.		
Brief Description: This use case describes how warehouse managers view and approve cycle counting approval reports.		
Trigger: Admin or warehouse manager wants to manage cycle counting approval reports.		
Relationships: Association : Warehouse manager Include : - Extend : - Generalization : -		
Normal Flow of Events: 1. Warehouse manager logins into the application. 2. Warehouse manager wants to manage the cycle counting approval reports. 3. System displays a list of cycle counting approval reports. 4. Warehouse manager chooses one report from the list to approve. 5. System prompt confirmation from warehouse manager. 6. Warehouse manager confirms the action. 7. System updates the system count with the variance. 8. System calculates the IRA and generates the cycle counting summary report.		
Sub-flows:		
Alternate/Exceptional Flows: 4a: If the manager rejects the report 1. The system will display a popup modal to ask if manager wants to reassign the staff to recount the SKU. 2. If the manager confirms to recount the SKU, the system will generate a schedule and reassign to the staff, else, the system will change the status of the report as “rejected” without updating the inventory count.		

Table 4:10: Use case description of manage cycle counting

<table>
<tr><td colspan="2">Use Case Name: Manage cycle counting</td><td>ID: 8</td><td>Importance Level: High</td></tr>
<tr><td colspan="2">Primary Actor: Warehouse manager</td><td colspan="2">Use Case Type: Detail, essential</td></tr>
<tr><td colspan="4">Stakeholders and Interests:
Warehouse manager - wants to manage the cycle counting</td></tr>
<tr><td colspan="4">Brief Description: This use case describes how warehouse manager manages the cycle counting</td></tr>
<tr><td colspan="4">Trigger: Warehouse manager wants to manage the cycle counting</td></tr>
<tr><td colspan="4">Relationships:
Association : Warehouse manager
Include : -
Extend : -
Generalization : -</td></tr>
<tr><td colspan="4">Normal Flow of Events:
1. Warehouse manager logins into the application.
2. Warehouse manager chooses to manage the cycle counting.
2.1. If warehouse manager wants to start the cycle counting
S-1: Start cycle counting subflow is performed.
2.2. If warehouse manager wants to view upcoming cycle counting SKUs
S-2: View upcoming cycle counting SKUs subflow is performed.</td></tr>
<tr><td colspan="4">Sub-flows:
S-1: Start cycle counting subflow
1. Warehouse manager chooses to start a new cycle counting process.
2. System displays a start cycle counting form to prompt warehouse manager’s input for workdays of warehouse, counting frequency for each cycle counting class, staff assignment, inventories to be included and start and end date of the cycle counting.
3. System classifies the SKU and calculates the daily count for each group.
4. System displays a confirmation modal with the details on classification and daily counts and prompts for warehouse manager’s confirmation.
5. Warehouse manager confirms and the cycle counting process is started or restarted.
6. System generates all schedules and assign to cycle count personnel evenly.</td></tr>
</table>

S-2: View upcoming cycle counting SKUs subflow is performed.

1. Warehouse manager wants to view upcoming cycle counting SKUs.
2. System displays the lists of upcoming SKUs by cycle counting classes.
3. Warehouse manager wants to reassign staff or view cycle counting settings
    a. If manager wants to reassign staff
        i. System prompts manager to select the staff to replace and staff to assign
        ii. System moves all schedules assigned for staff to replace and reassign to new staff selected.
    b. If manager wants to view cycle counting settings
        i. System shows a modal pop-up with all current cycle counting settings

Alternate/Exceptional Flows:

Table 4:11: Use case description of record inventory

Use Case Name: Record inventory	ID: 9	Importance Level: High
Primary Actor: Staff	Use Case Type: Detail, essential	
Stakeholders and Interests: Staff - wants to record inventory stock		
Brief Description: This use case describes how staff records inventory stock.		
Trigger: Staff wants to check in/ out inventory stock		
Relationships: Association : Staff Include : - Extend : - Generalization : -		
Normal Flow of Events: 1. Staff logins into the application. 2. Staff chooses to record inventory. 3. System displays the inventory list of the staff's warehouse. 4. Staff searches the inventory by ID or name. 5. Staff checks in or checks out the desired inventory. 6. System prompt inputs from staff. 7. Staff enters quantity to check in or check out and the remarks. 8. System records the data into the database.		
Sub-flows:		
Alternate/Exceptional Flows:		

Table 4:12: Use case description of perform cycle counting

Use Case Name: Perform cycle counting	ID: 10	Importance Level: High
Primary Actor: Staff	Use Case Type: Detail, essential	
Stakeholders and Interests: Staff - wants to perform cycle counting		
Brief Description: This use case describes how staff performs cycle counting		
Trigger: Staff wants to perform cycle counting		
Relationships: Association : Staff Include : - Extend : - Generalization : -		
Normal Flow of Events: 1. Staff logins into the application. 2. System displays a list of upcoming, pending approval and completed cycle counting SKUs. 3. Staff chooses one SKU from the list of upcoming SKUs to perform count. 4. System displays the bin location of the SKU and the system count in a modal popup to prompt the actual count of the SKU from staff. 5. Staff enters the actual count into the modal popup. 6. System records the actual count, calculates variances and generates a cycle count approval report. 7. System moves the counted SKU to the pending approval list.		
Sub-flows:		
Alternate/Exceptional Flows: 7a: If the cycle count approval report is rejected by the warehouse manager, the SKU will be placed into the upcoming list again.		

Table 4:13: Use case description of view all staff

Use Case Name: View all staff	ID: 11	Importance Level: High
Primary Actor: Warehouse manager	Use Case Type: Detail, essential	
Stakeholders and Interests: Warehouse manager - wants to view all staff in his warehouse.		
Brief Description: This use case describes how a warehouse manager views all staff in his warehouse.		
Trigger: Warehouse manager wants to view staff information.		
Relationships: Association : Warehouse manager Include : - Extend : - Generalization : -		
Normal Flow of Events: 1. Warehouse manager logins to the system. 2. Warehouse manager wants to view the staff in the warehouse. 3. System displays the details of the warehouse and a staff list. 4. Warehouse manager chooses a staff member to view his or her details. 5. System displays the staff's details.		
Sub-flows:		
Alternate/Exceptional Flows:		

Table 4:14: Use case description of manage category

<table>
<tr><td colspan="2">Use Case Name: Manage category</td><td>ID: 12</td><td>Importance Level: High</td></tr>
<tr><td colspan="2">Primary Actor: Admin</td><td colspan="2">Use Case Type: Detail, essential</td></tr>
<tr><td colspan="4">Stakeholders and Interests:
Admin - wants to add, view, update and delete category</td></tr>
<tr><td colspan="4">Brief Description: This use case describes how admin view, add, update and delete category.</td></tr>
<tr><td colspan="4">Trigger: Admin wants to manage the category.</td></tr>
<tr><td colspan="4">Relationships:
Association : Admin
Include : -
Extend : -
Generalization : -</td></tr>
<tr><td colspan="4">Normal Flow of Events:
5. The admin logins the application.
6. The admin chooses to manage the category.
7. The system displays all categories.
8. The admin can add, update or delete a category.
8.1. The admin wants to add a new category.
S-1: Add new category subflow is performed.
8.2. The admin wants to update the category.
S-2: Update a category subflow is performed.
8.3. The admin wants to delete the category.
S-3: Delete a category subflow is performed.</td></tr>
<tr><td colspan="4">Sub-flows:
S-1: Add new category
1. Admin wants to add a new category.
2. System displays a modal pop up to prompt input.
3. Admin enters the category name.
4. System creates the category and save into database.
S-2: Update a category
1. Admin wants to update a category.
2. System displays a modal pop up to prompt input.
3. Admin edit the category name
4. System updates the category.
S-3: Delete a category
1. Admin wants to delete a category.
2. System displays a confirmation modal to prompt for admin’s confirmation.</td></tr>
</table>

3. Admin confirms to delete the category. 4. System deletes the category.
Alternate/Exceptional Flows:

4.5 Entity Relationship Diagram (ERD)

Entity Relationship Diagram (ERD) illustrates all entities needed for the implemented system and how each entity is related to each other. Figure 4.12 shows the ERD diagram of the implemented warehouse inventory management system.

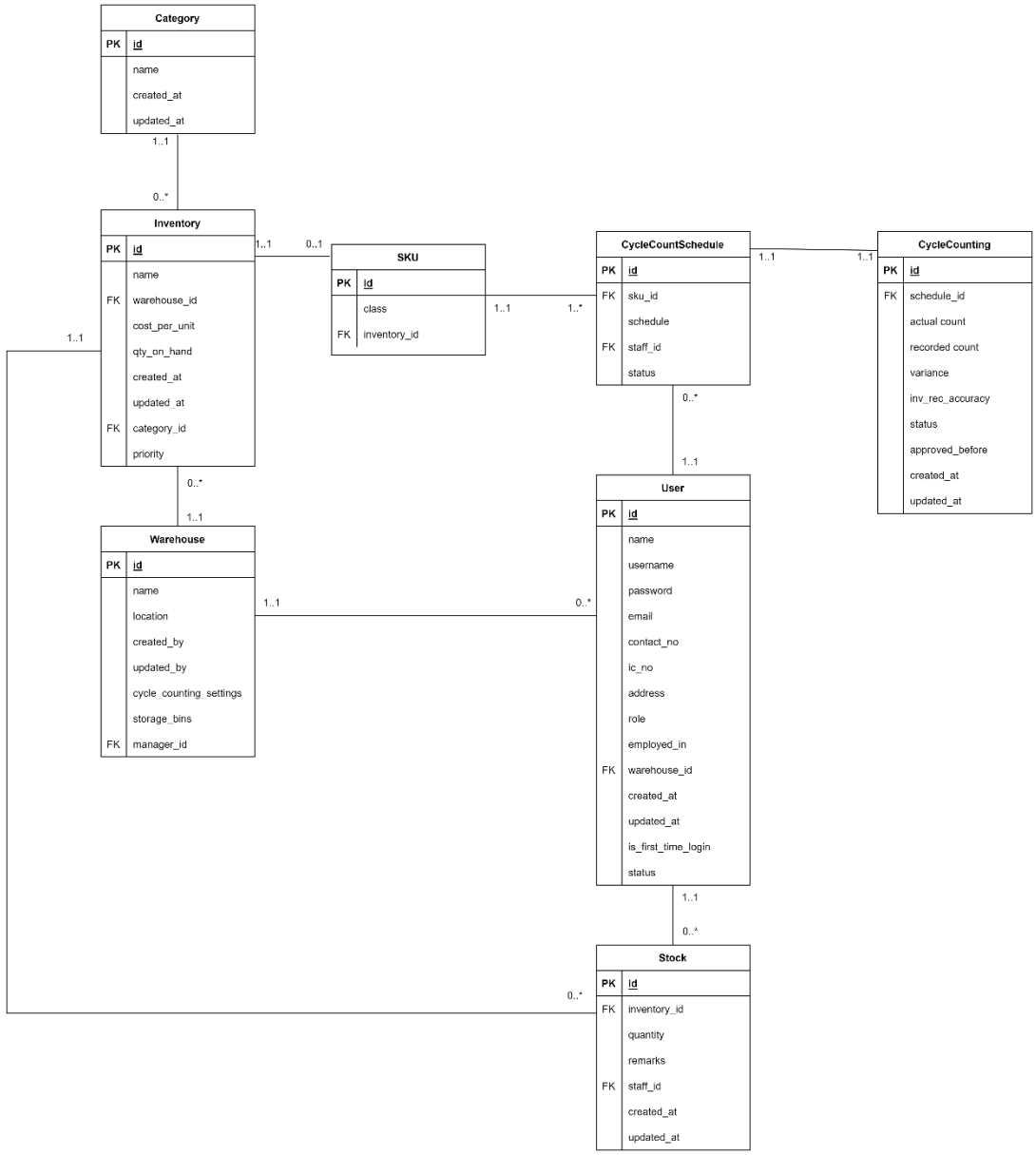

Figure 4.12: ERD diagram

4.6 Prototype

The prototype development was developed based on the three main modules specified in the project scope, which will include the module of inventory stock recording, cycle counting auto-scheduler and report management.

4.6.1 Inventory stock recording

The first module included in the prototype development is the inventory stock recording. Figure 4.13 shows the wireframe of staff viewing all inventories on the record inventory page. After the staff selects one of the inventories to check in / check out, an inventory modal pop-up will be displayed to prompt for the quantity being taken out or received, then, a confirmation modal is shown for staff's confirmation as shown in Figure 4.14 and Figure 4.15.

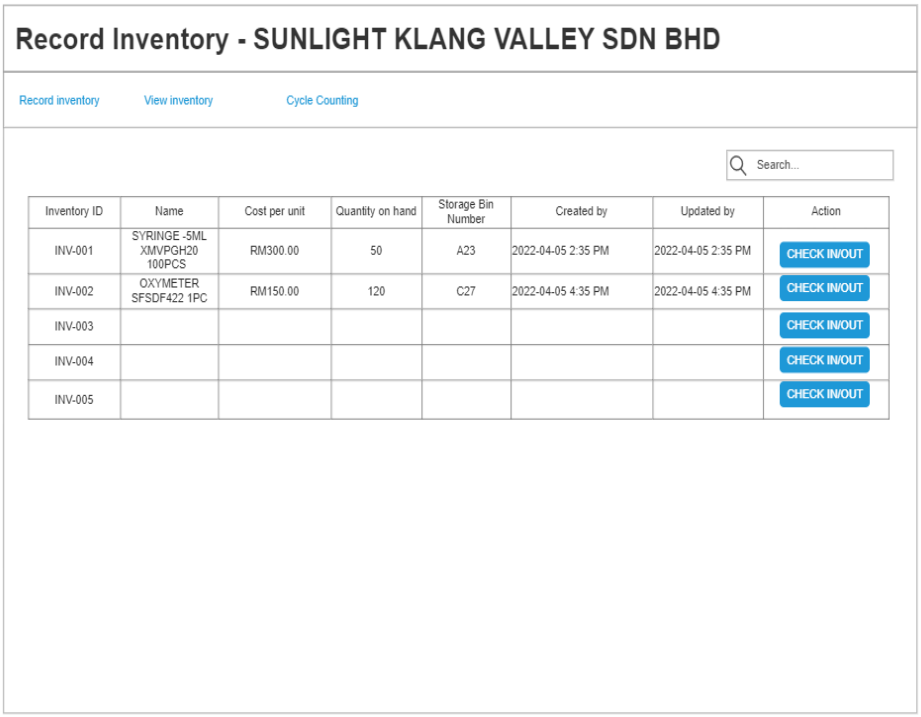

Figure 4.13: Staff- record inventory page

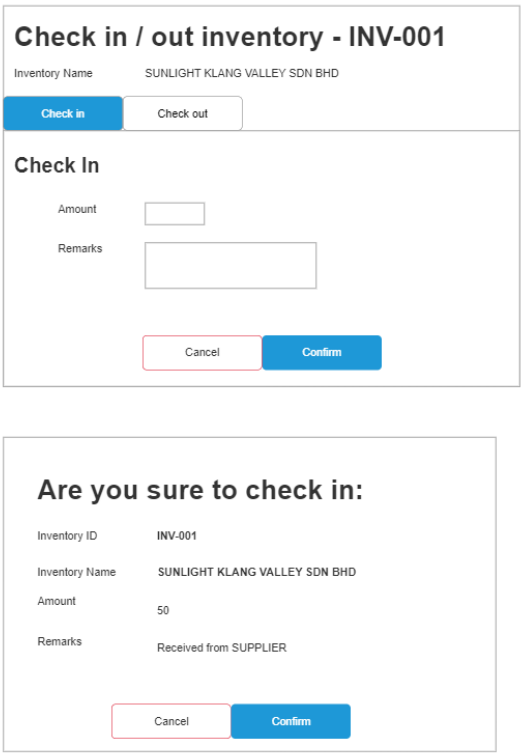

Figure 4.14: Staff- check in inventory popup

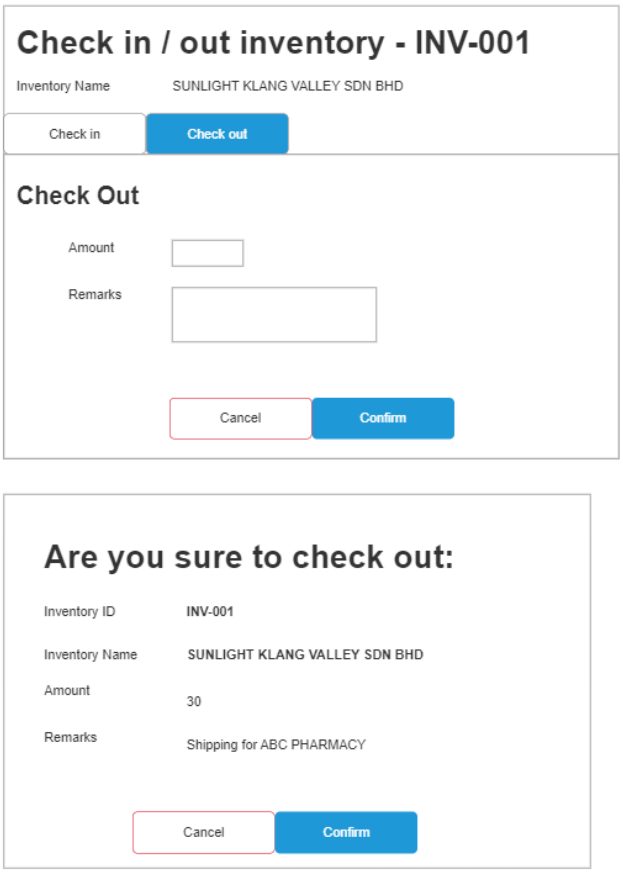

Figure 4.15: Staff- check out inventory popup

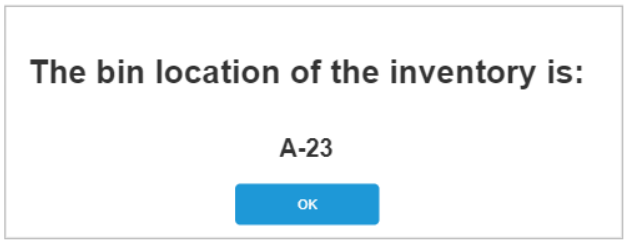

Figure 4.16: Staff- show bin location after check in inventory

4.6.2 Cycle counting auto scheduler

The second included module is the cycle counting auto-scheduler. The first wireframe for this module is the manage-cycle-counting feature which only warehouse managers can access. For the homepage of the manage-cycle-counting view, there are two actions available, which are starting a new cycle counting or viewing upcoming cycle counting SKUs as shown in Figure 4.17.

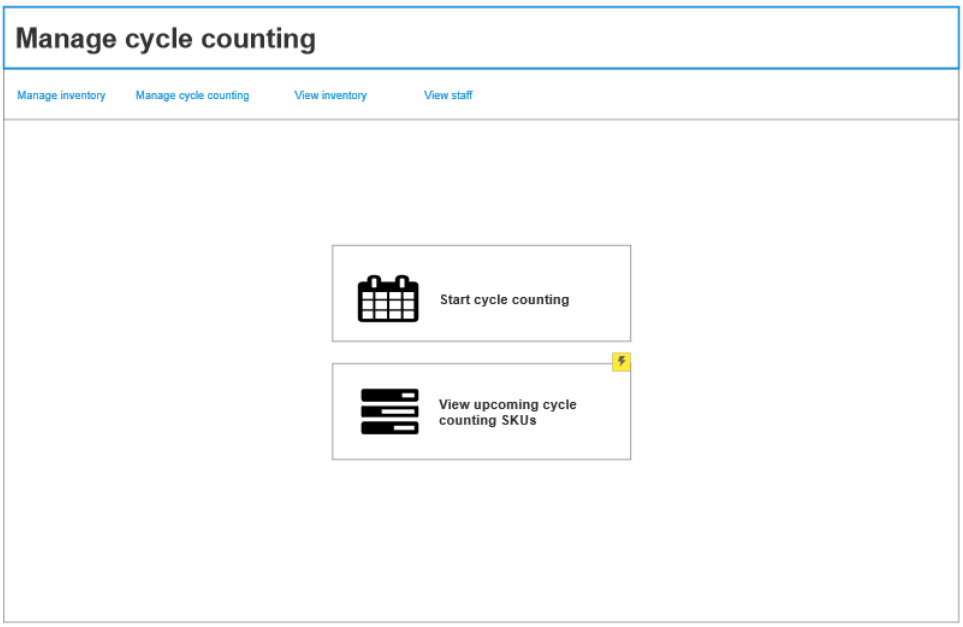

Figure 4.17: Warehouse manager- manage cycle counting page

For starting a new cycle counting, a few inputs will be required which include the workday of the warehouse, the counting frequency of each cycle counting class, staff to be assigned to cycle counting, SKUs to be included, and the start and end date of the cycle counting as shown in Figure 4.18.

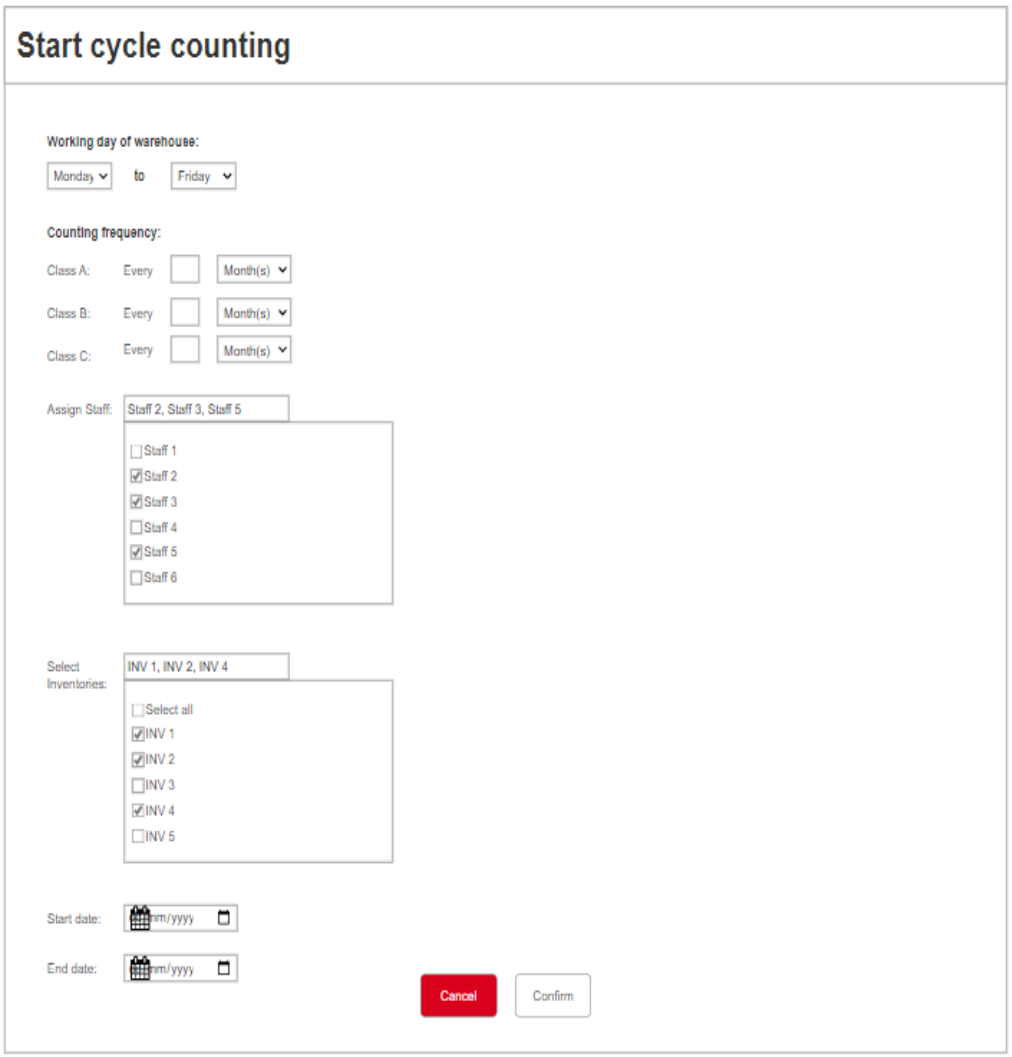

Figure 4.18: Warehouse manager- start cycle counting

After the necessary details have been filled, the daily counts for each cycle counting class will be calculated and a confirmation dialog will pop up to display the daily counts and prompt for confirmation as shown in Figure 4.19. Upon confirmation, the system will auto-generate all schedules for cycle counting according to the start and end date entered and assign to staff evenly.

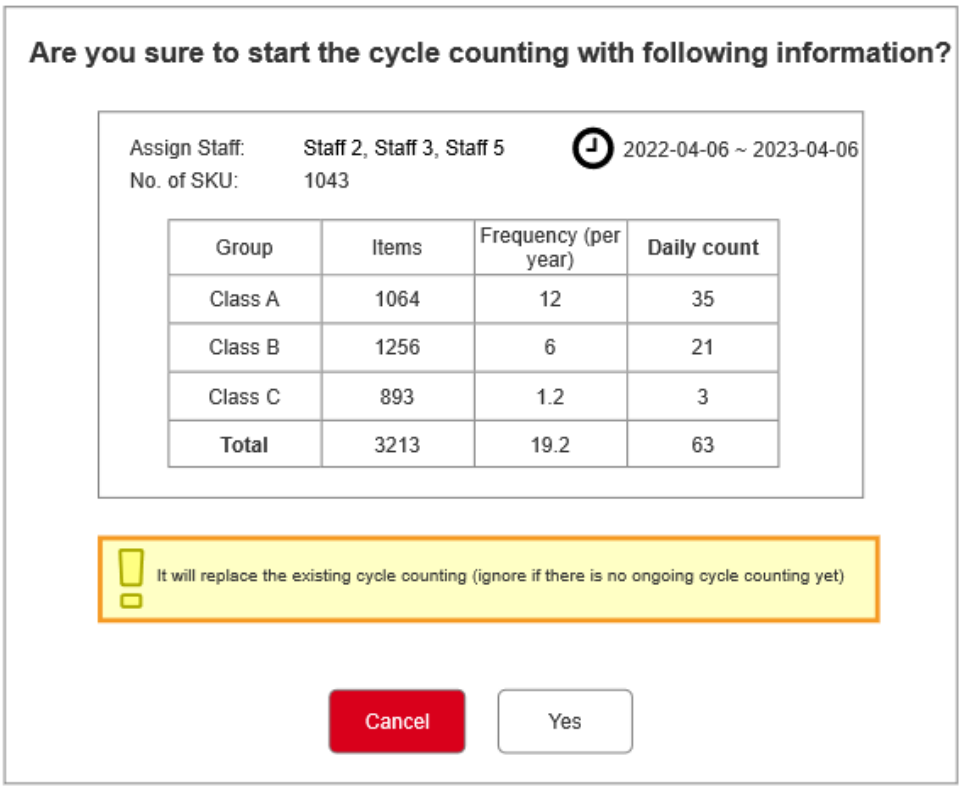

Figure 4.19: Warehouse manager- start cycle counting confirmation popup

Next, the warehouse manager is able to view all upcoming cycle counting schedules in this module as shown in Figure 4.20.

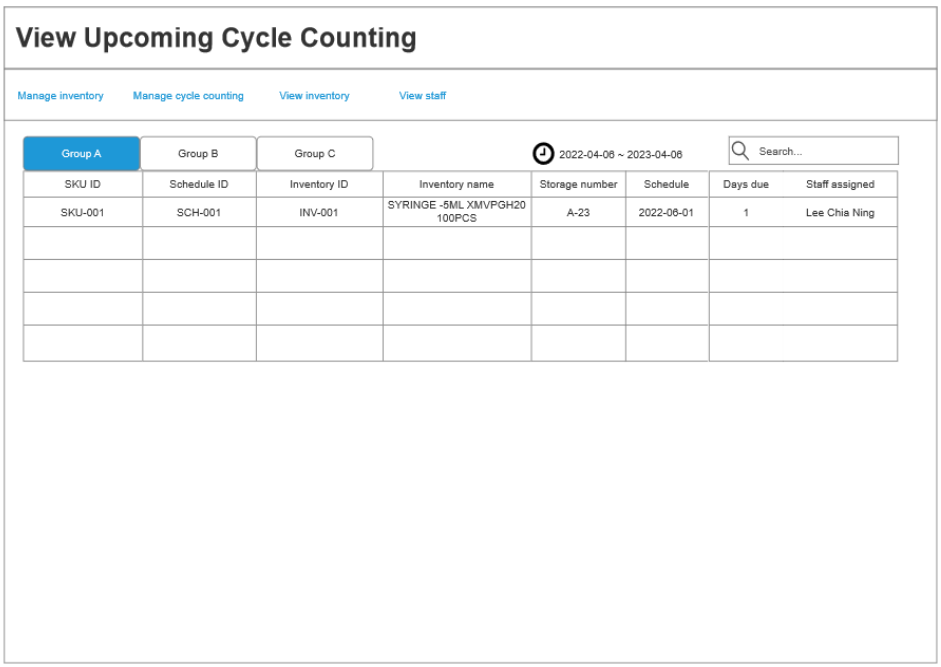

Figure 4.20: Warehouse manager- view upcoming cycle counting

Staff, on the other hand, will be able to view all SKUs assigned to them in Figure 4.21 and perform the counting before the deadline. When the staff chooses one SKU from the list to initiate the count, a popup will be displayed as shown in Figure 4.22 to show the bin location of that SKU and prompt for the actual count from the staff.

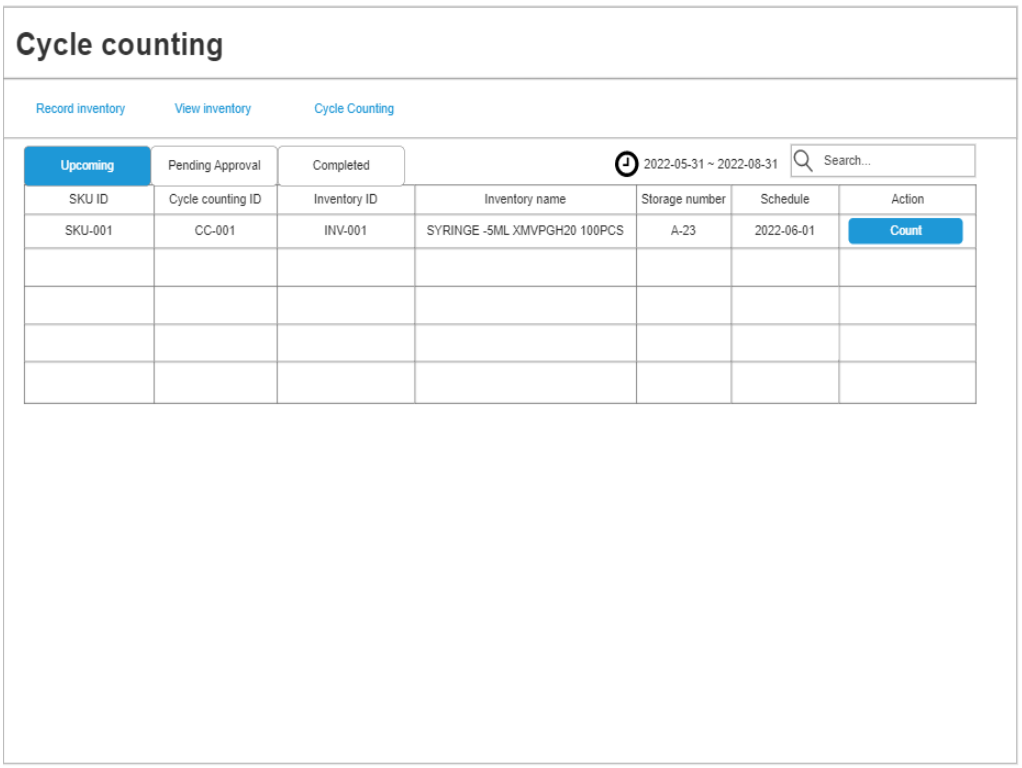

Figure 4.21: Staff- perform cycle counting

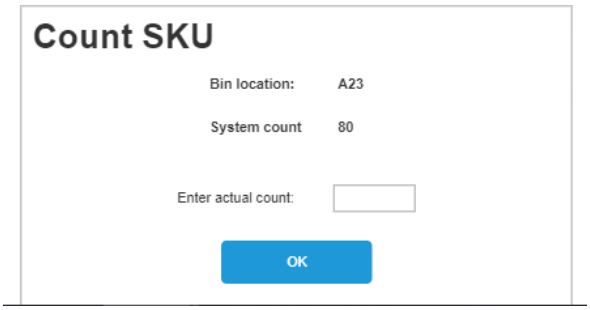

Figure 4.22: Staff- count SKU

4.6.3 Report Management

The third module is the report management. The report management can be accessed by the warehouse manager and the admin. For the warehouse manager, all information retrieved will be on his own warehouse only, whereas the admin will be able to view all reports from all warehouses.

The first report in this module is the daily reports, which contains information on all transactions of a warehouse on a specific date. The list of all reports is shown in Figure 4.23 for the warehouse manager and Figure 4.24 for the admin. The viewing of daily report details is in Figure 4.25.

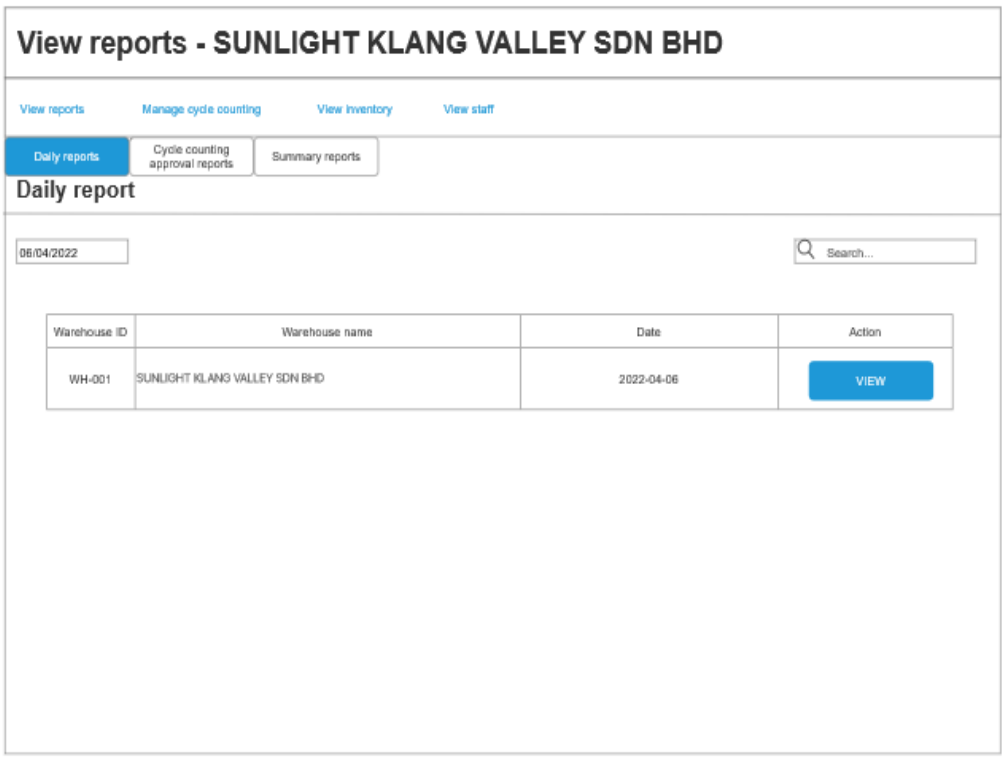

Figure 4.23: Warehouse manager - view daily reports

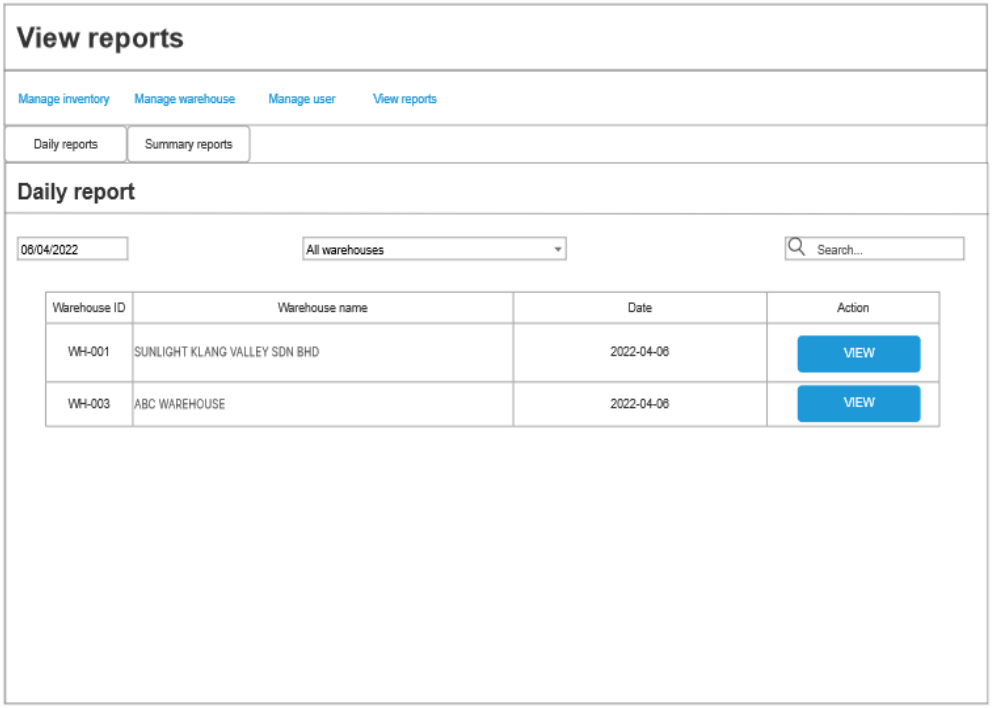

Figure 4.24: Admin - view daily reports

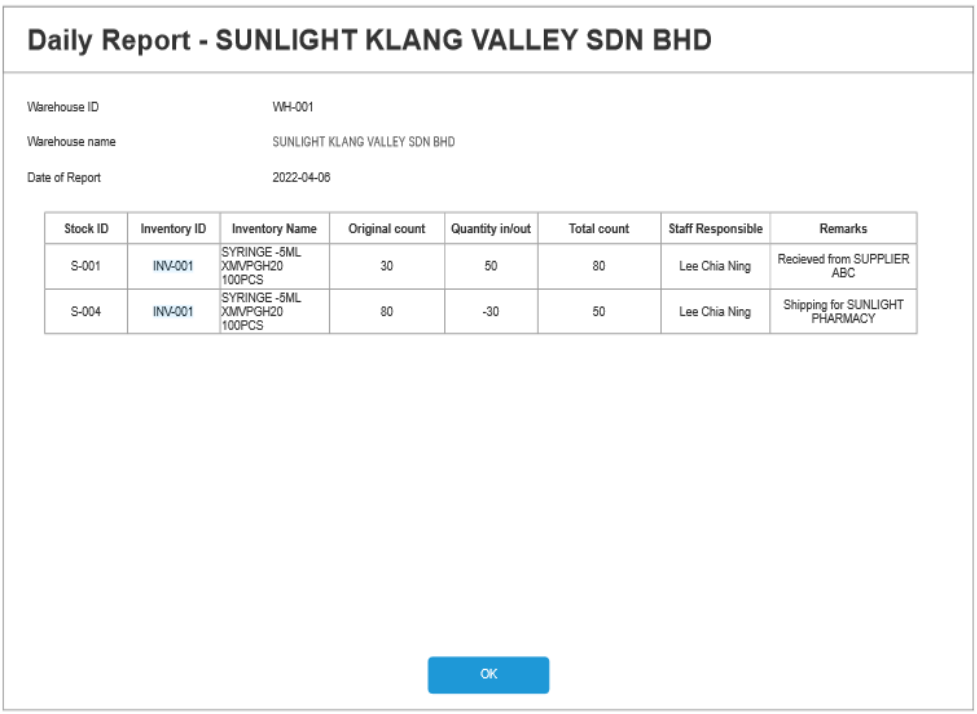

Figure 4.25: Warehouse manager & admin- view daily report details

The second report is the cycle counting summary report. Figure 4.26 and 4.27 shows the view of the warehouse manager and admin whereas Figure 4.28 shows the details of summary report.

Figure 4.26: Warehouse manager- view cycle counting summary reports

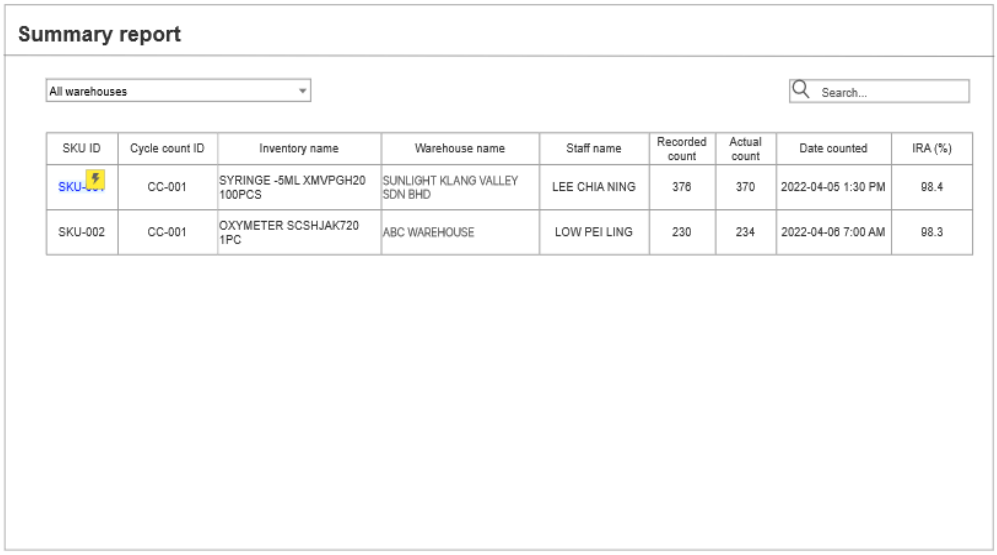

Figure 4.27: Admin- view cycle counting summary reports

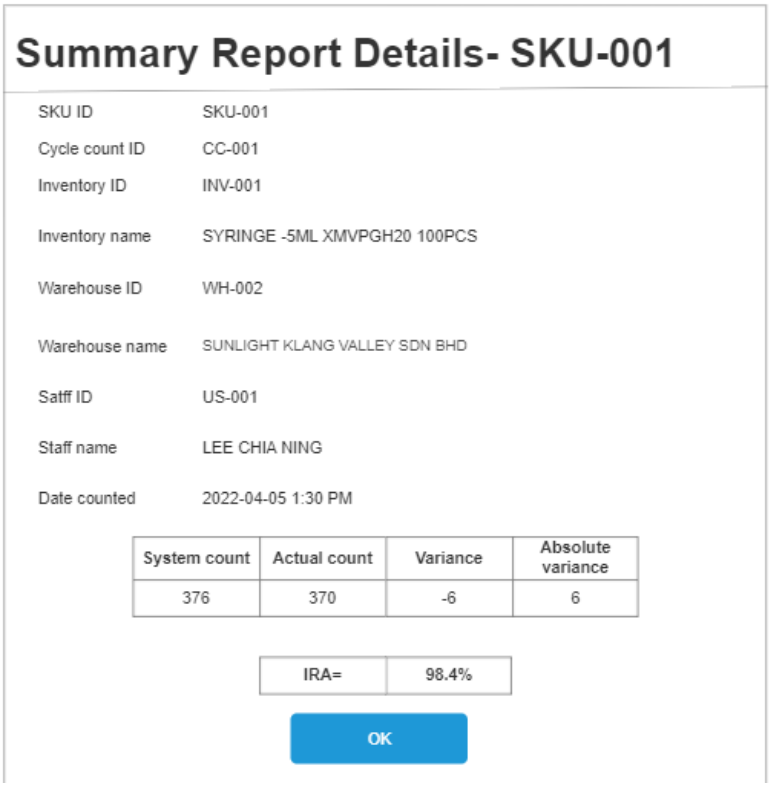

Figure 4.28: Warehouse manager & admin- view cycle counting summary report details

The third report is the cycle counting approval report, which can be accessed by the warehouse manager only as shown in Figure 4.29. The warehouse manager can select one of the reports and approve or reject it. By approving the report, the variance will be added on top of the inventory of that report, whereas by rejecting it, the cycle counting SKU will be returned to the warehouse staff for recounting. The confirmation modal of the approve and reject action is shown in Figure 4.30 and Figure 4.31.

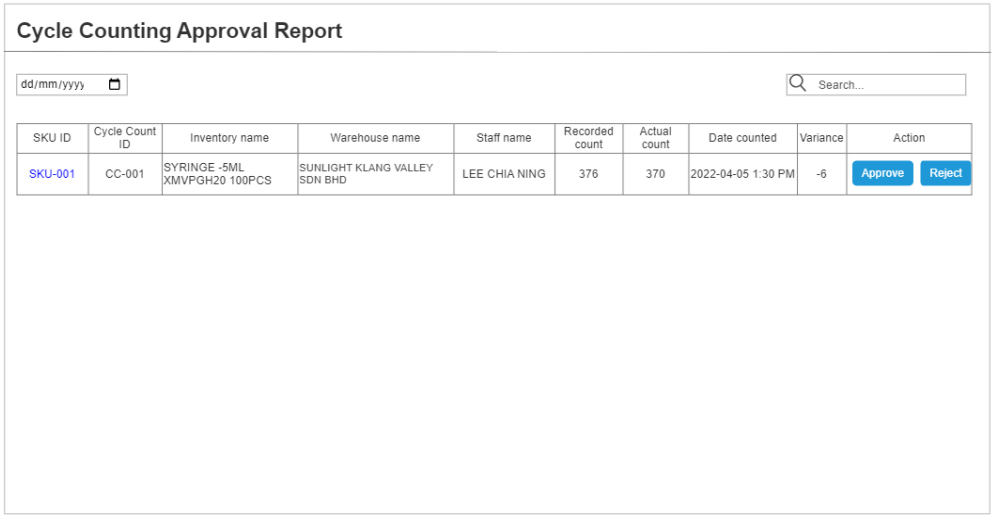

Figure 4.29: Warehouse manager - view cycle counting approval reports

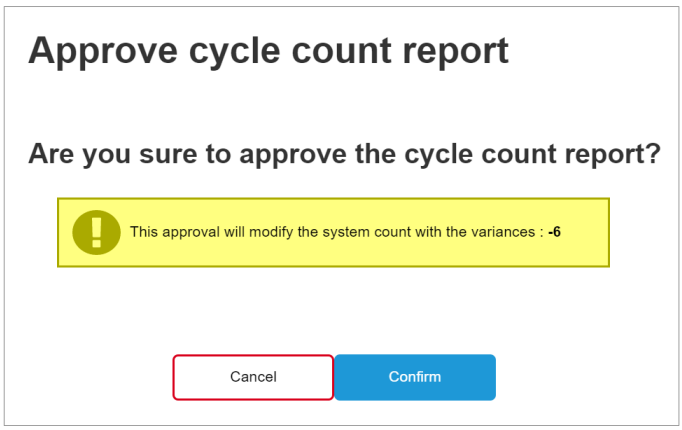

Figure 4.30: Warehouse manager - approve cycle counting approval reports

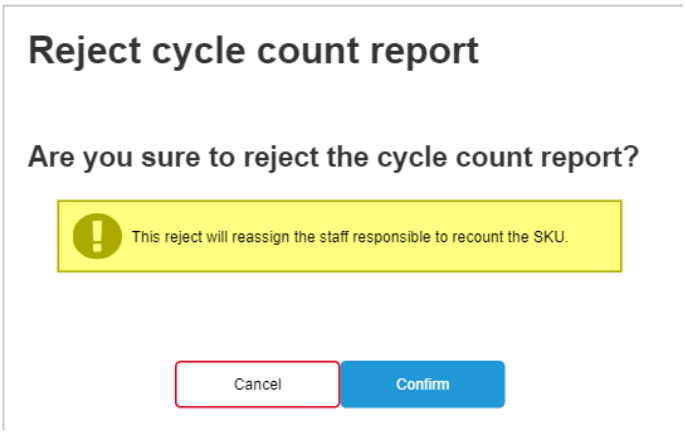

Figure 4.31: Warehouse manager - reject cycle counting approval reports

4.7 Summary

This chapter discussed the facts-findings and functional and non-functional requirements. Besides, use case diagram and use case diagram description to visualise the interaction between admin, warehouse manager, staff and system are also included. An ERD diagram is designed to show the relationship of entities. Lastly, screenshots of the developed low-fidelity prototype are also included.

CHAPTER 5

SYSTEM DESIGN

5.1 Introduction

In this chapter, the system architecture design and modelling diagrams which include data flow diagrams and user interface flow diagrams are described. For the data flow diagram, the processes are according to the use cases mentioned in earlier chapters (Figure 4.11). Context diagram, level-0 DFD, level-1 DFD, level-2 DFD and level-3 DFD are presented. The user interface flow diagrams are drawn based on three types of users: staff, admin and manager. Lastly, screenshots of the implemented web-based system's user interface design are provided. All screenshots are arranged based on use cases as well for easier reading and understanding purposes.

5.2 System Architecture Design

The implemented system's architecture design is shown in Figure 5.1. The entire system is hosted on Heroku in which users or testers can access via *https://fyp-wims.herokuapp.com/*. The application is a web-based application developed in which its frontend is being developed using Vue.js whereas the RESTful APIs are built using the Laravel framework, which includes API authentication as well. The implemented system will send HTTP requests using the API endpoints in which Laravel will fetch data from the MySQL database and returns data to the frontend in JSON response format. For the check-in and check-out stock module, Laravel will create an event using Pusher API in which the Pusher will broadcast an API message to all clients that subscribed to the Pusher channel.

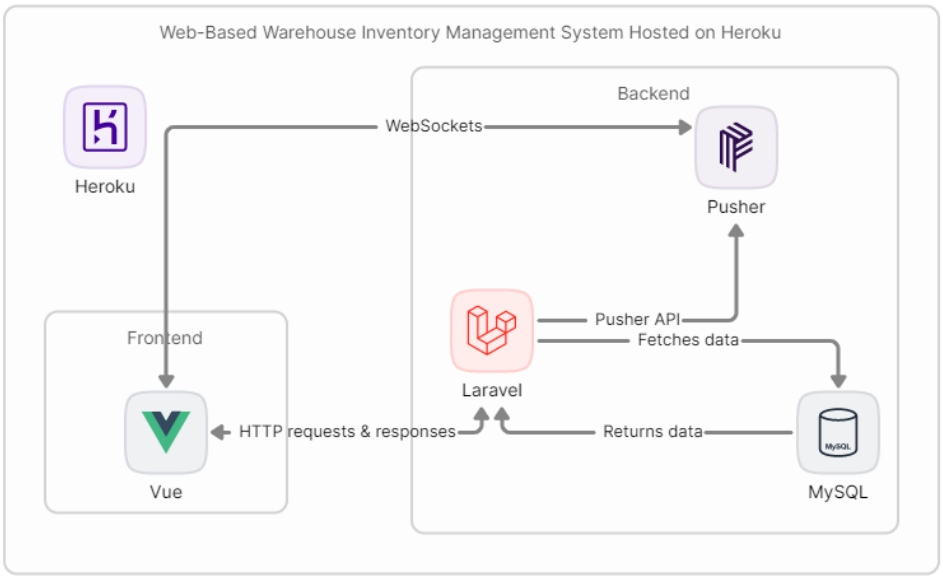

Figure 5.1: System architecture design

5.3 Modelling Diagram

This section shows modelling diagrams to visually describes the implemented system. The modelling diagrams will include data flow diagrams and interface flow diagrams.

5.3.1 Data Flow Diagram

A data flow diagram (DFD) maps the flow of data across the implemented system. Multi-level data flow diagrams show how data is handled from a lower perspective. In this section, a context diagram, DFD level-1, DFD level-2, DFD level-3 and DFD level-4 are drawn to show the data flow in the warehouse inventory management system.

5.3.1.1 Context Diagram

The context diagram is the basic overview of the data flow across the entire system as shown in Figure 5.2.

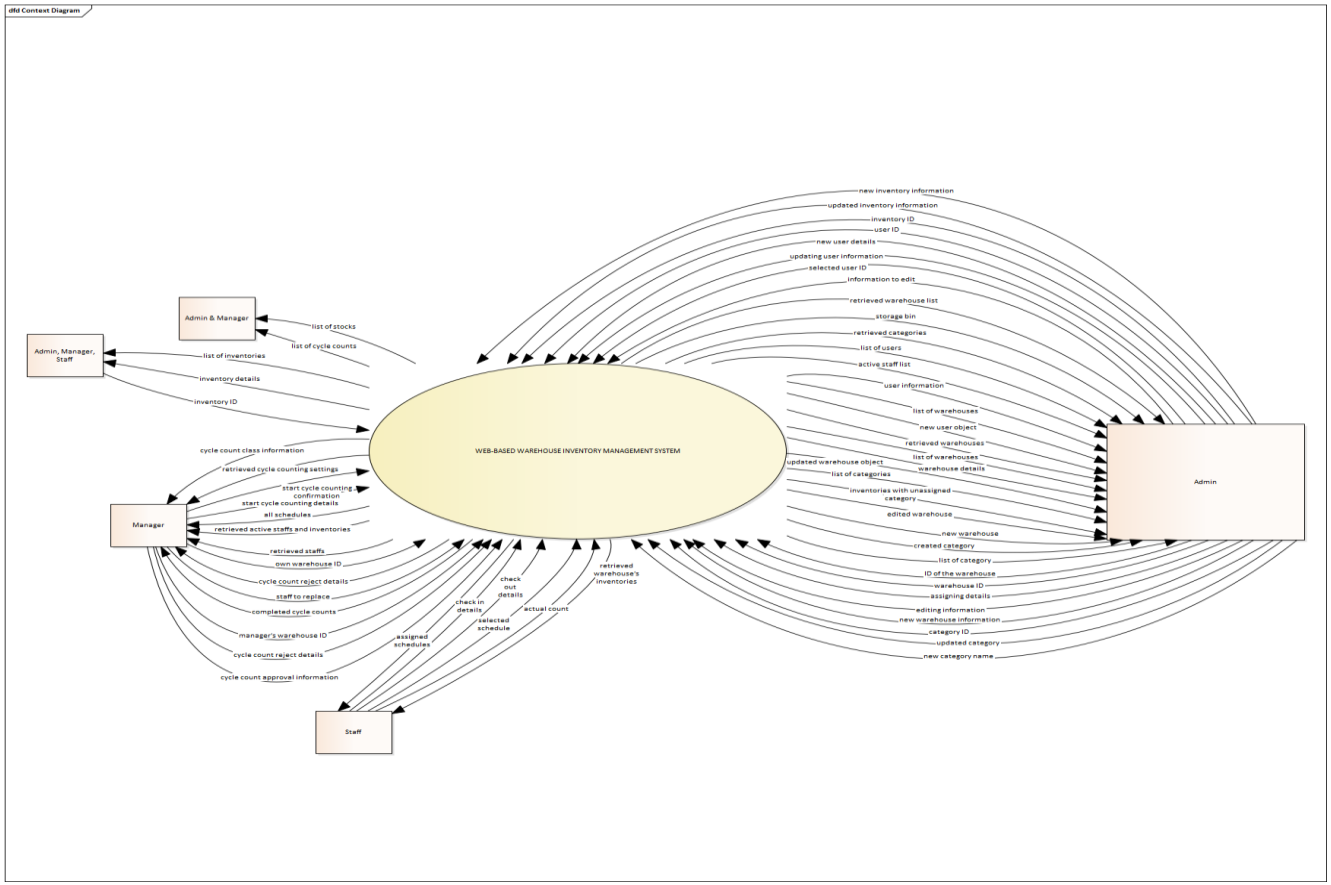

Figure 5.2: Context diagram

5.3.1.2 Data Flow Diagram Level-1

The DFD level-1 shows the detailed breakouts of the context diagram. At this level, all main processes of the system are highlighted and the data stores are also shown. Figure 5.3 shows the DFD level-1 of the implemented system in which the data flow between actors (users), processes and data stores are shown.

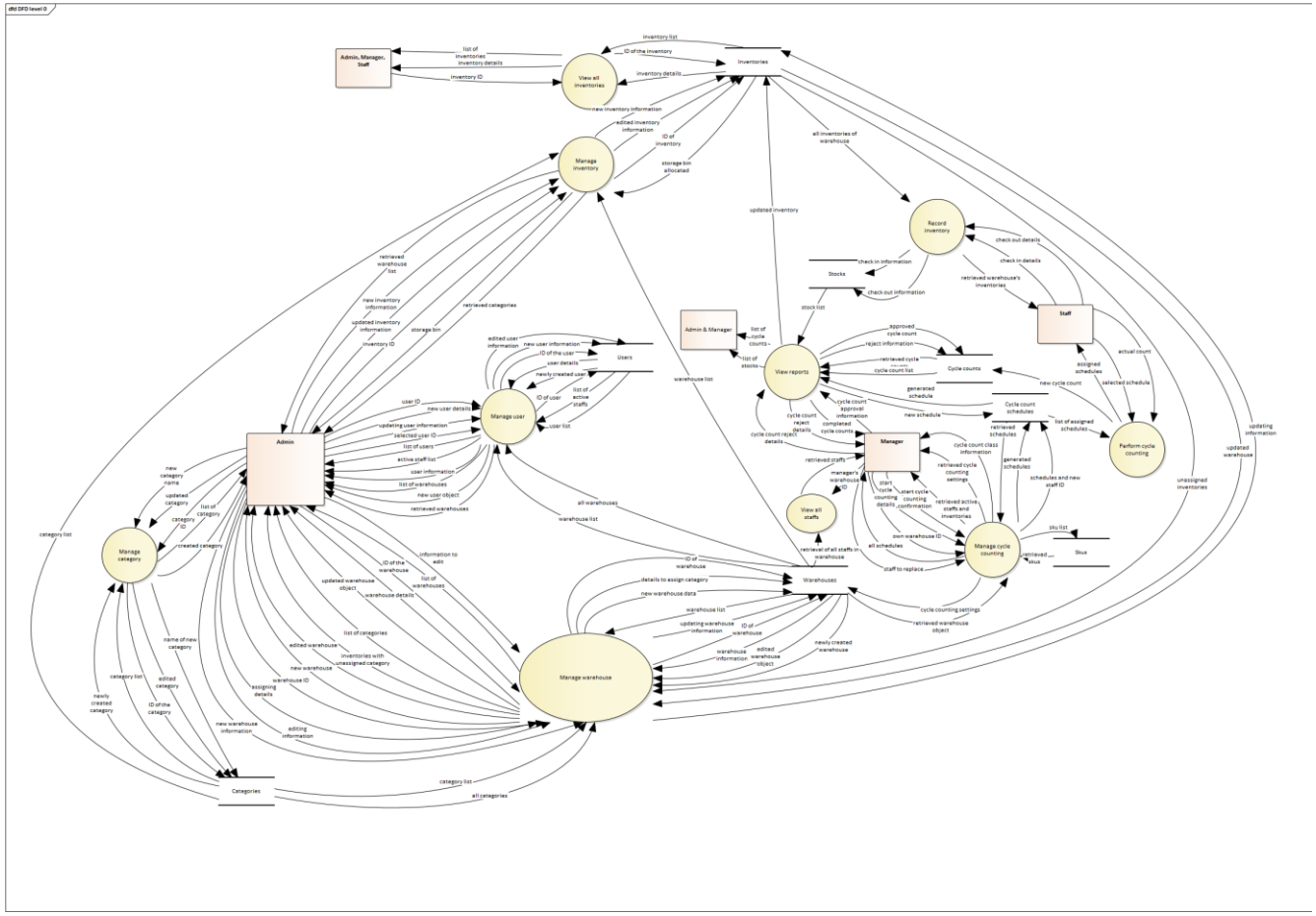

Figure 5.3: Data flow diagram level-1

5.3.1.3 Data Flow Diagram Level-2

DFD level-2 goes deeper into the processes of DFD level-1 which will show the subprocesses inside a process in DFD level-1. The processes in DFD level-1 that need to be detailed out are manage inventory process, manage user process, manage warehouse process, manage category process, manage cycle counting process, perform cycle counting process, record inventory process and view reports process.

5.3.1.3.1 Manage Inventory

Figure 5.4 shows the DFD level-2 of the manage inventory process from DFD level-1.

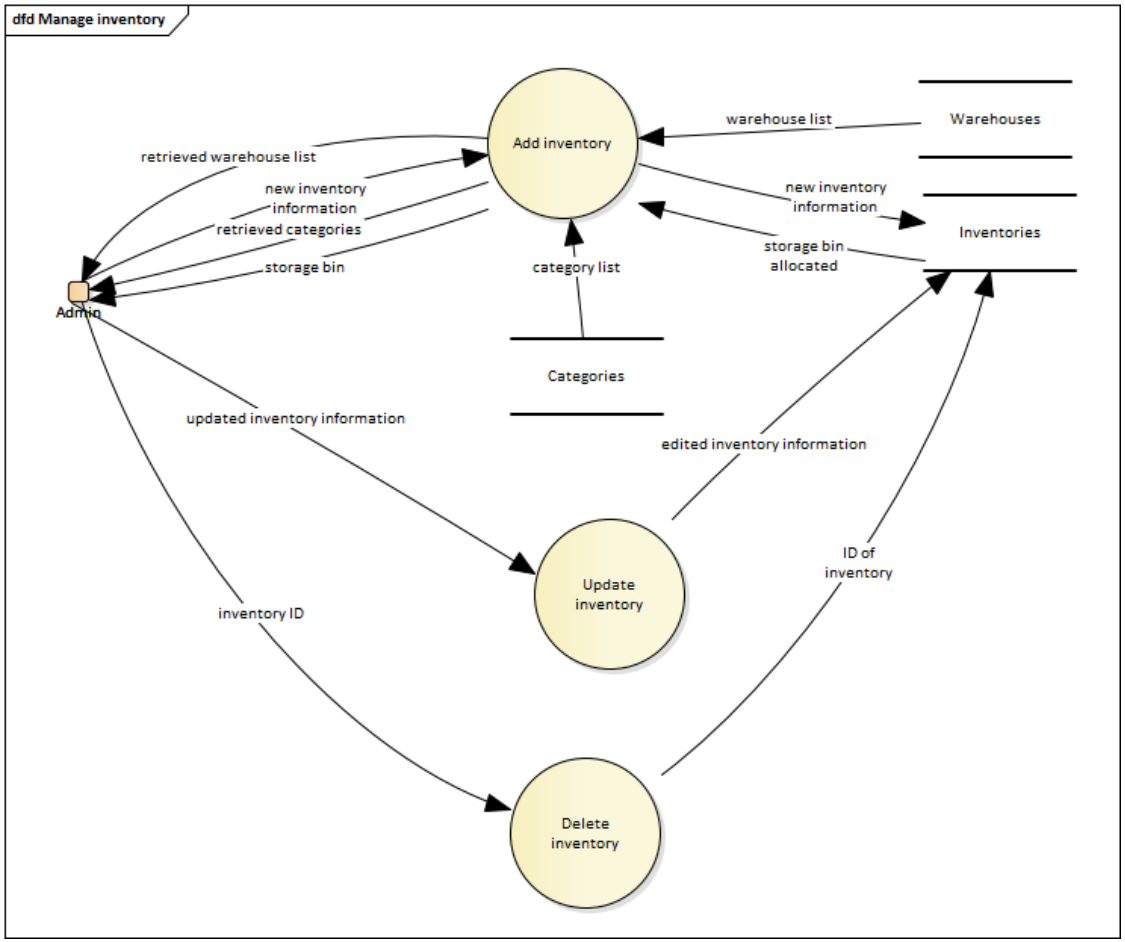

Figure 5.4: Data flow diagram level-2 (Manage inventory)

5.3.1.3.2 Manage User

Figure 5.5 shows the subprocesses of manage user process from DFD level-1.

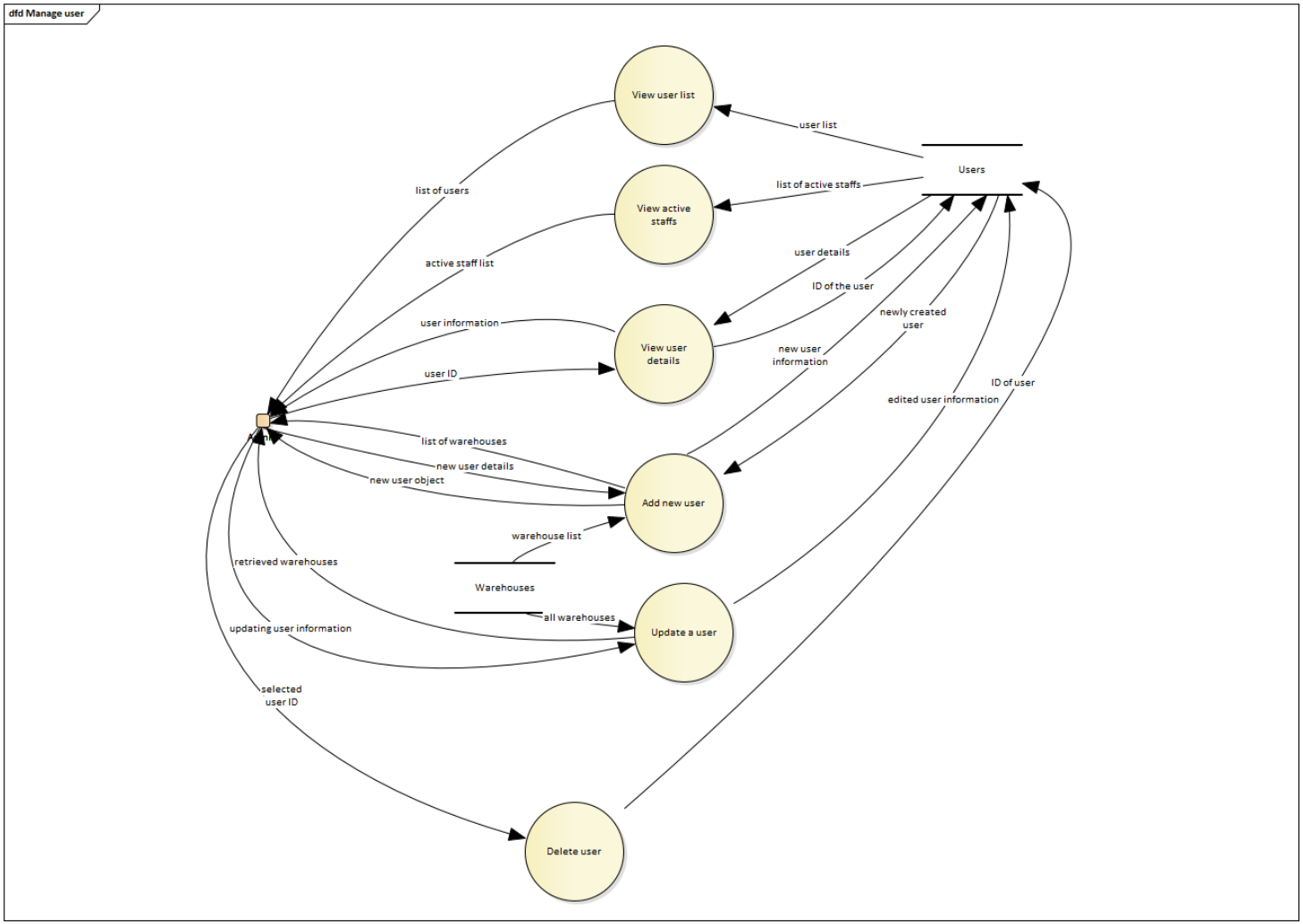

Figure 5.5: Data flow diagram level-2 (Manage user)

5.3.1.3.3 Manage Warehouse

Figure 5.6 shows the subprocesses of manage warehouse process from DFD level-1.

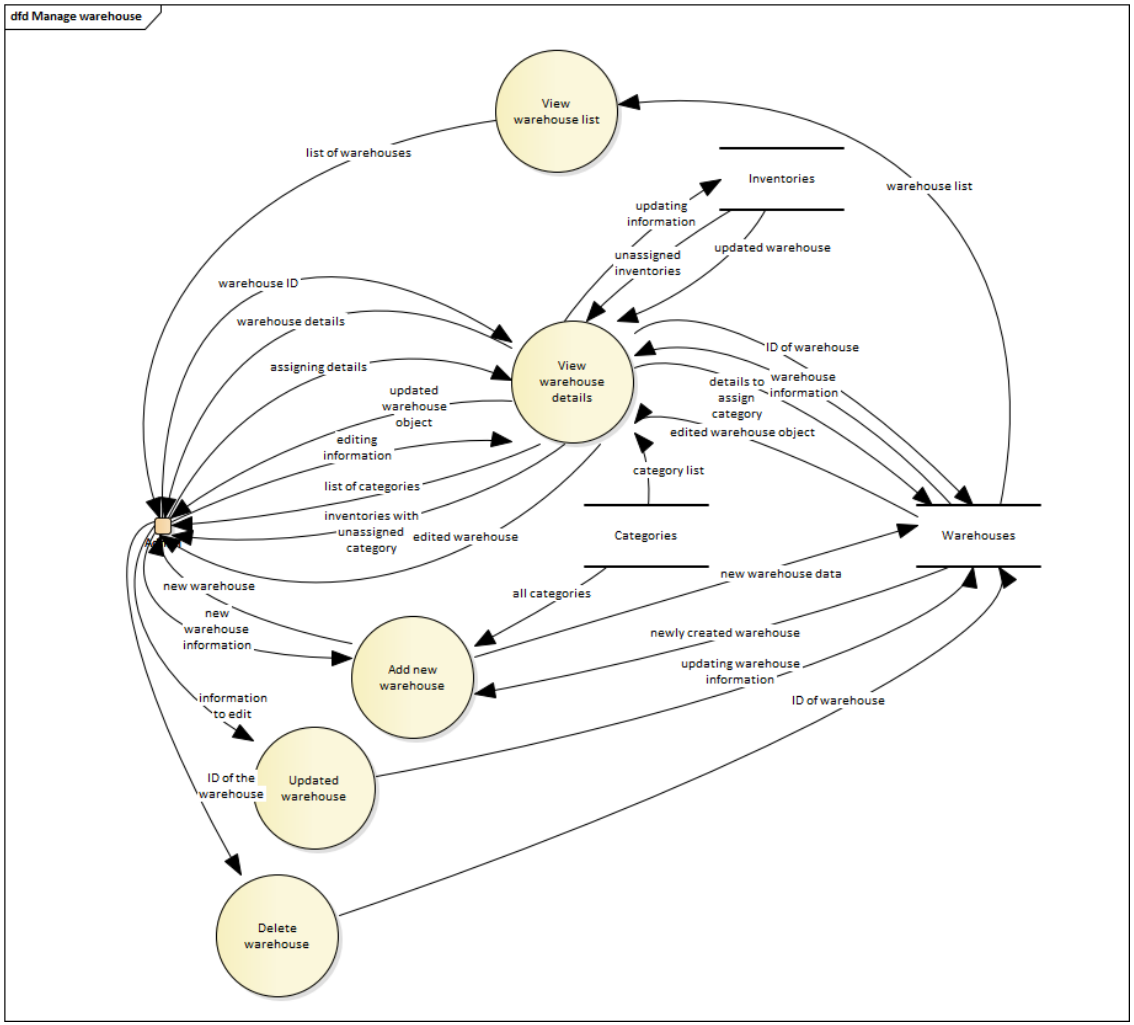

Figure 5.6: Data flow diagram level-2 (Manage warehouse)

5.3.1.3.4 Manage Category

Figure 5.7 shows the subprocesses of manage category process from DFD level-1.

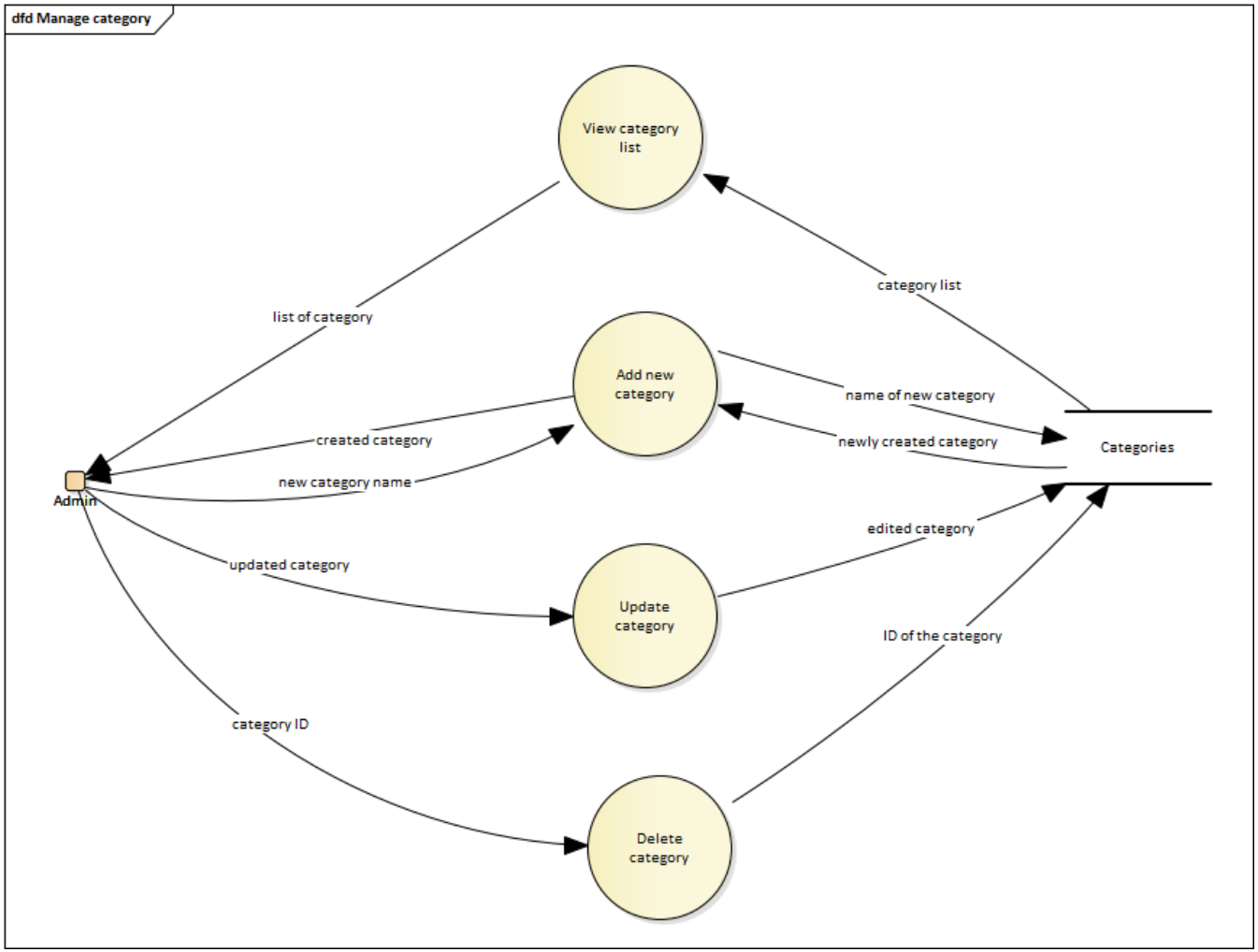

Figure 5.7: Data flow diagram level-2 (Manage category)

5.3.1.3.5 Manage Cycle Counting

Figure 5.8 shows the subprocesses of manage cycle counting process from DFD level-1.

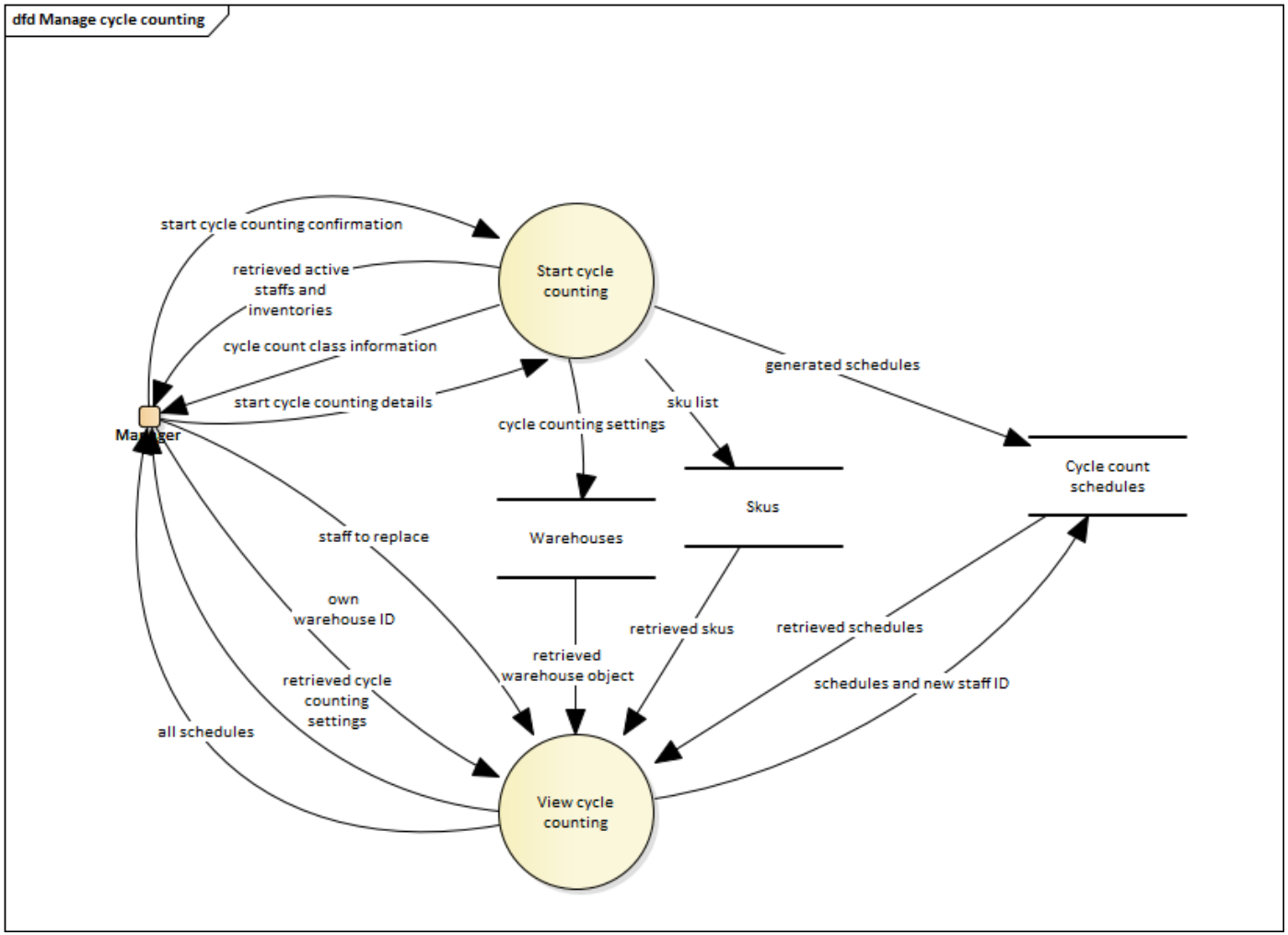

Figure 5.8: Data flow diagram level-2 (Manage cycle counting)

5.3.1.3.6 Perform Cycle Counting

Figure 5.9 shows the subprocesses of perform cycle counting process from DFD level-1.

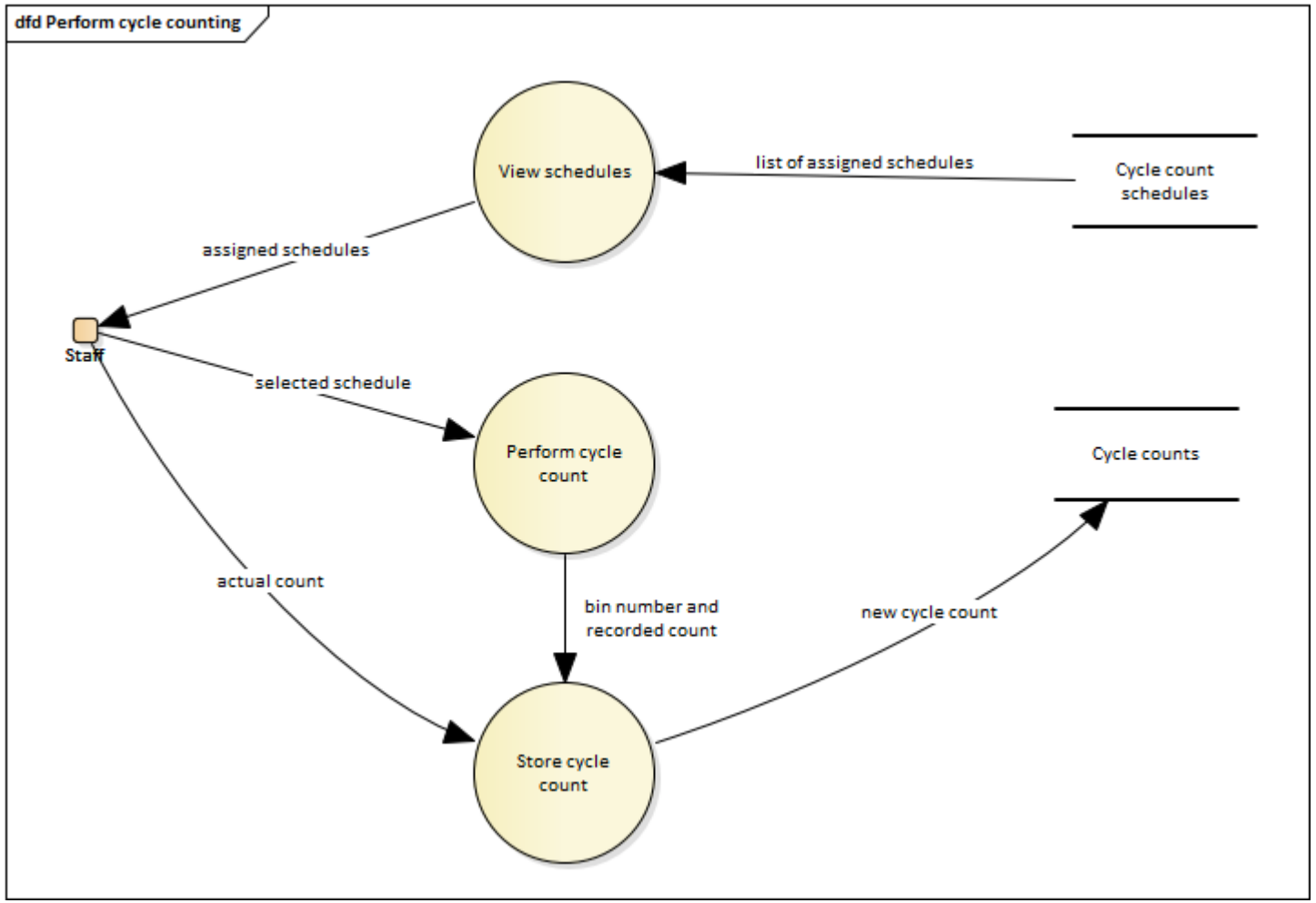

Figure 5.9: Data flow diagram level-2 (Perform cycle counting)

5.3.1.3.7 Record Inventory

Figure 5.10 shows the subprocesses of record inventory process from DFD level-1.

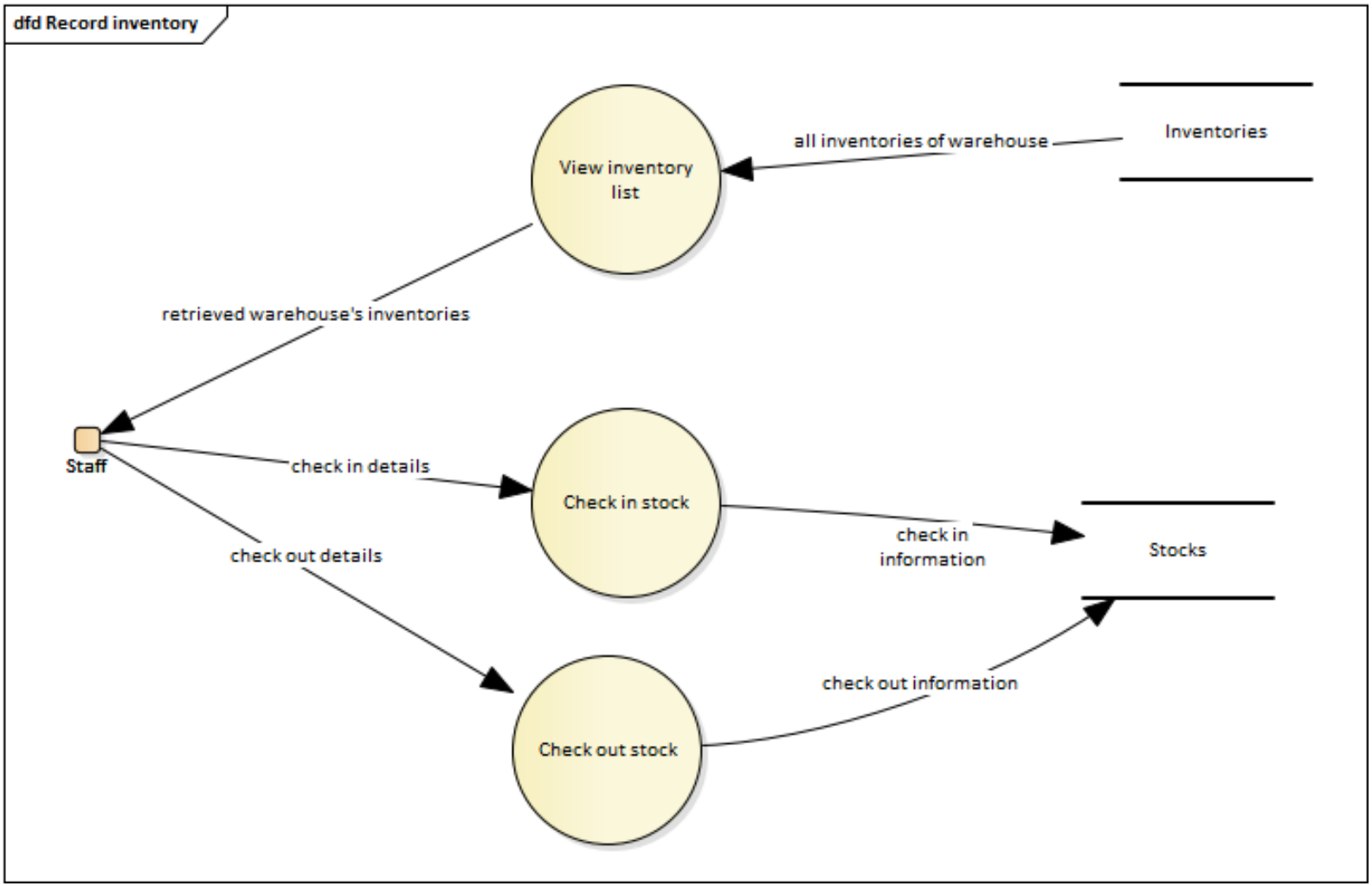

Figure 5.10: Data flow diagram level-2 (Record inventory)

5.3.1.3.8 View reports

Figure 5.11 shows the subprocesses of view reports process from DFD level-1.

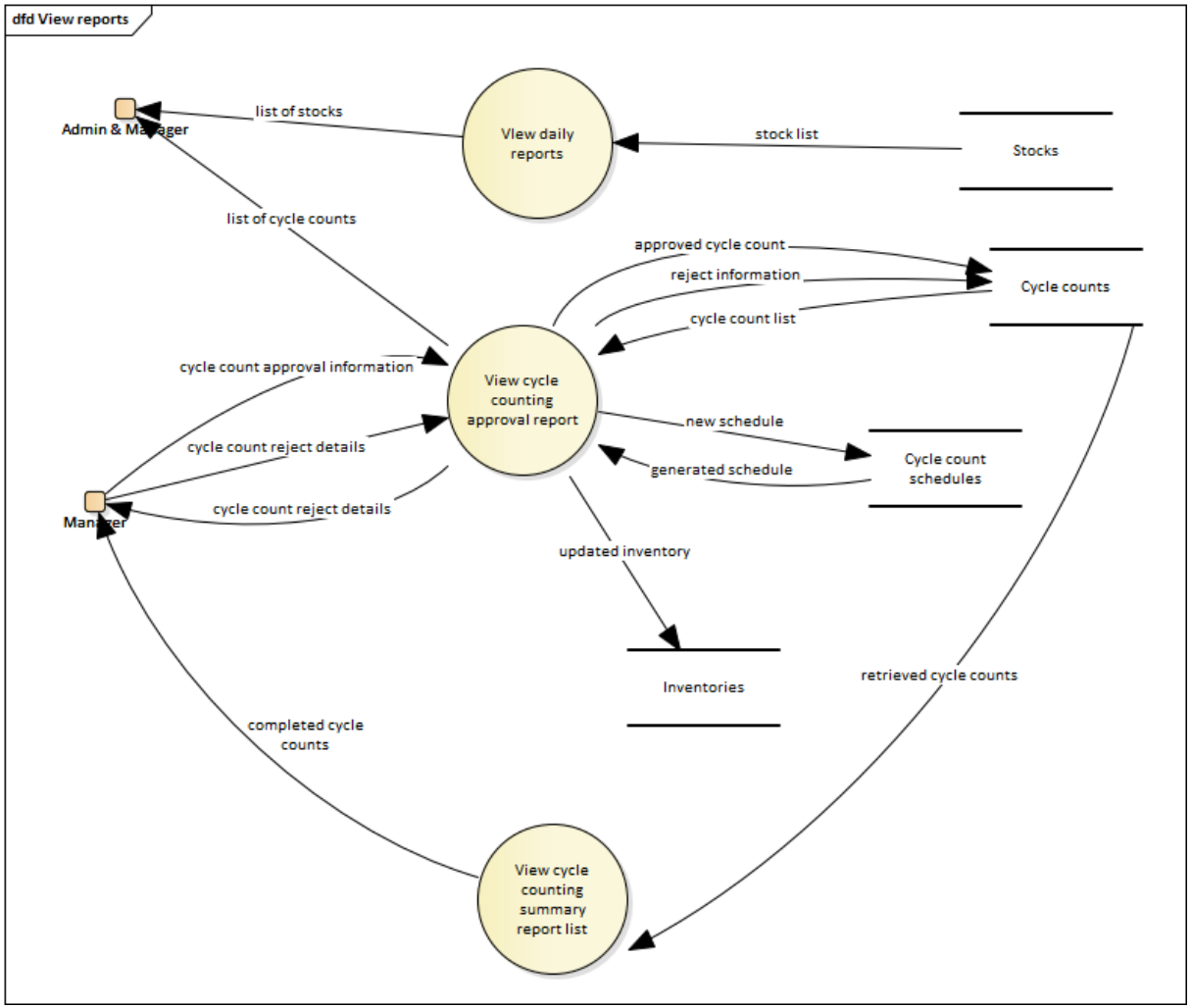

Figure 5.11: Data flow diagram level-2 (View reports)

5.3.1.4 Data Flow Diagram Level-3

Similar to DFD level-2, the DFD level-3 goes deeper into processes in DFD level-2. This section includes the view warehouse details process, view cycle counting approval reports process, start cycle counting process and view cycle counting proves from DFD level-2.

5.3.1.4.1 View Warehouse Details

Figure 5.12 shows the subprocesses of view warehouse details process from DFD level-2 (Figure 5.6).

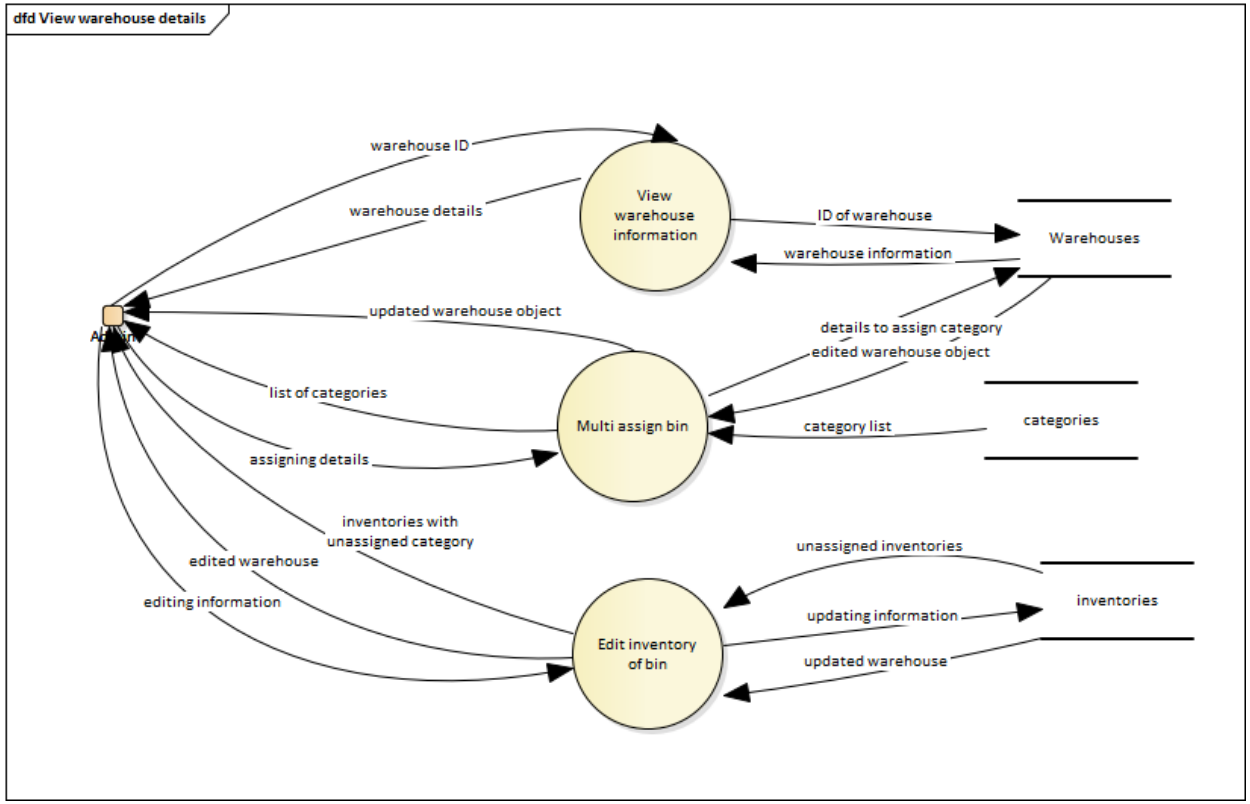

Figure 5.12: Data flow diagram level-3 (View warehouse details)

5.3.1.4.2 View Cycle Counting Approval Reports

Figure 5.13 shows the subprocesses of view cycle counting approval reports process from DFD level-2 (Figure 5.11).

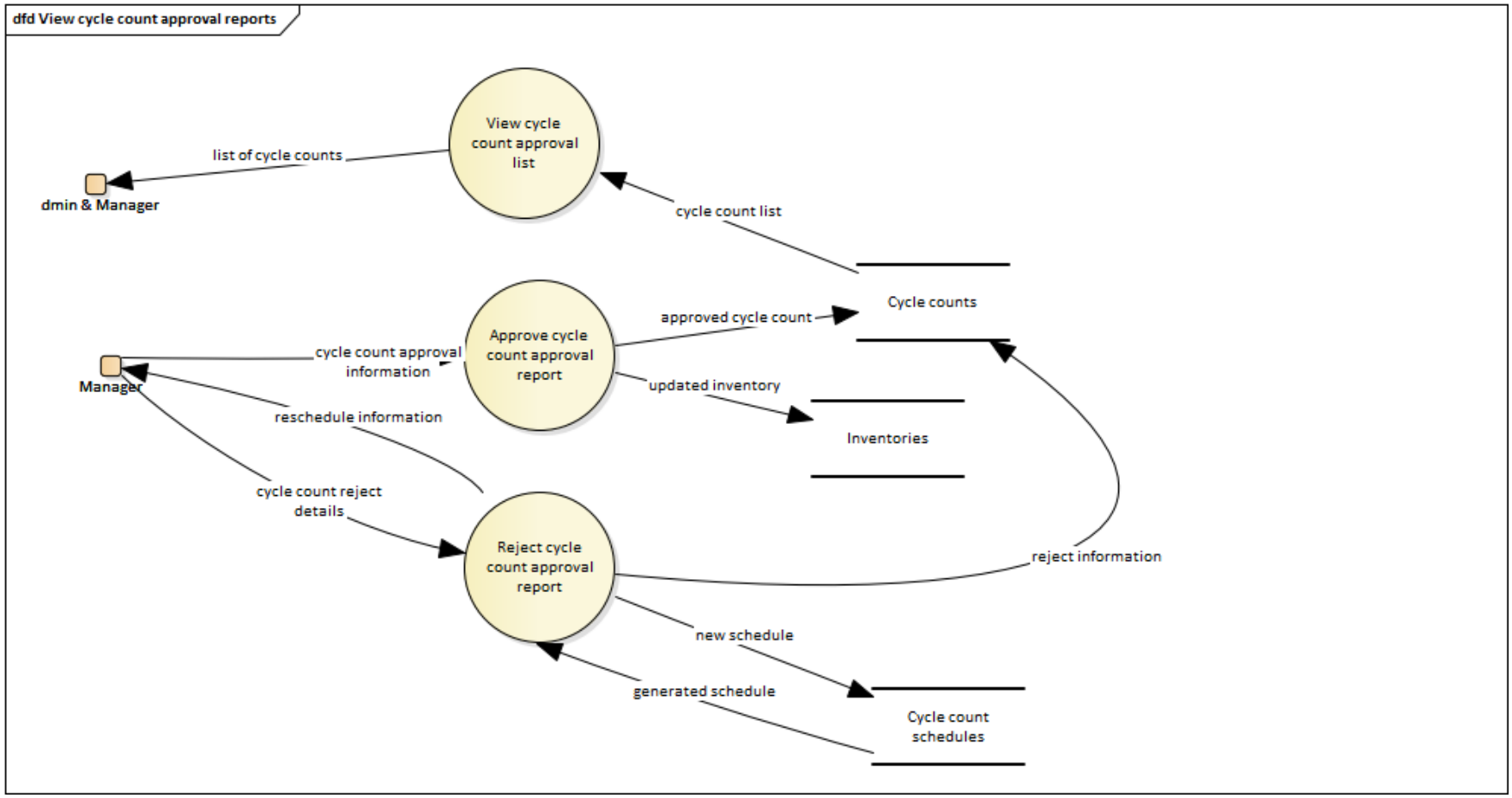

Figure 5.13: Data flow diagram level-3 (View cycle counting approval reports)

5.3.1.4.3 Start Cycle Counting

Figure 5.14 shows the subprocesses of start cycle counting process from DFD level-2 (Figure 5.8).

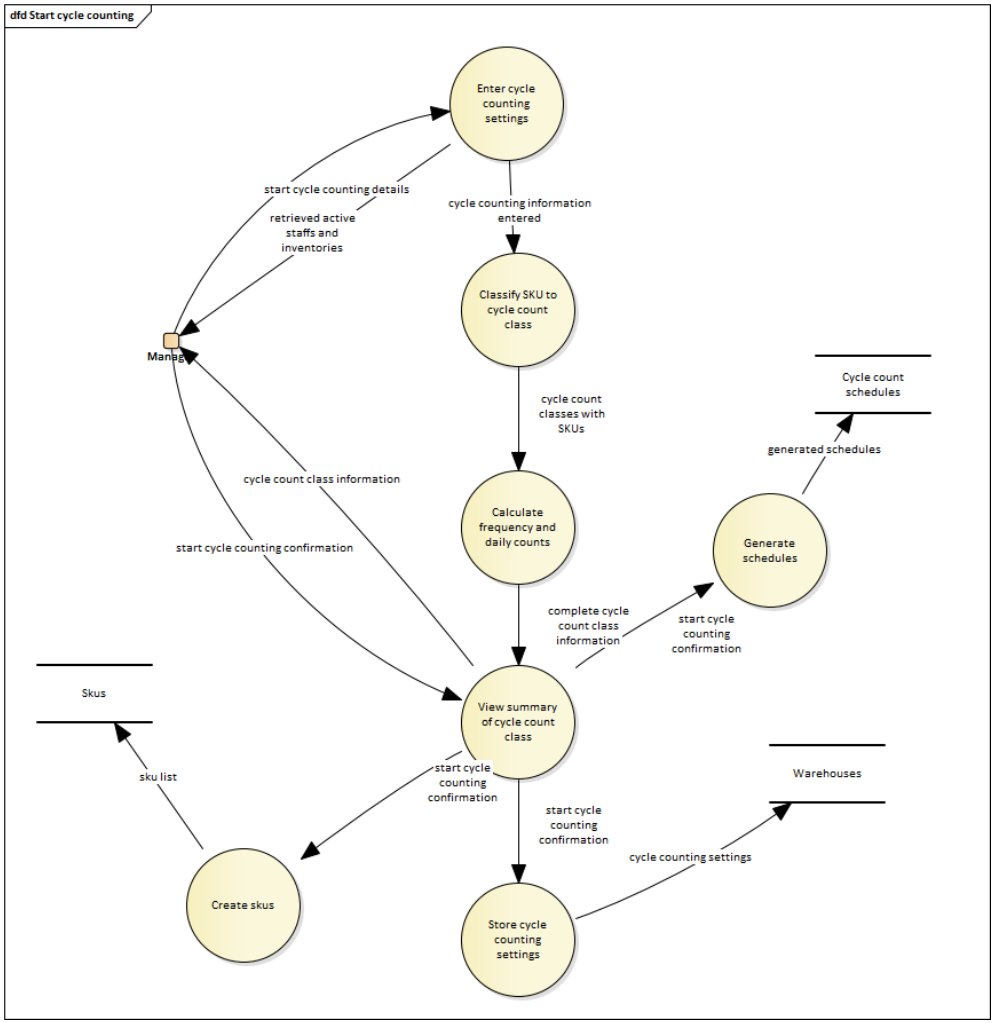

Figure 5.14: Data flow diagram level-3 (Start cycle counting)

5.3.1.4.4 View Cycle Counting

Figure 5.13 shows the subprocesses of view cycle counting process from DFD level-2 (Figure 5.8).

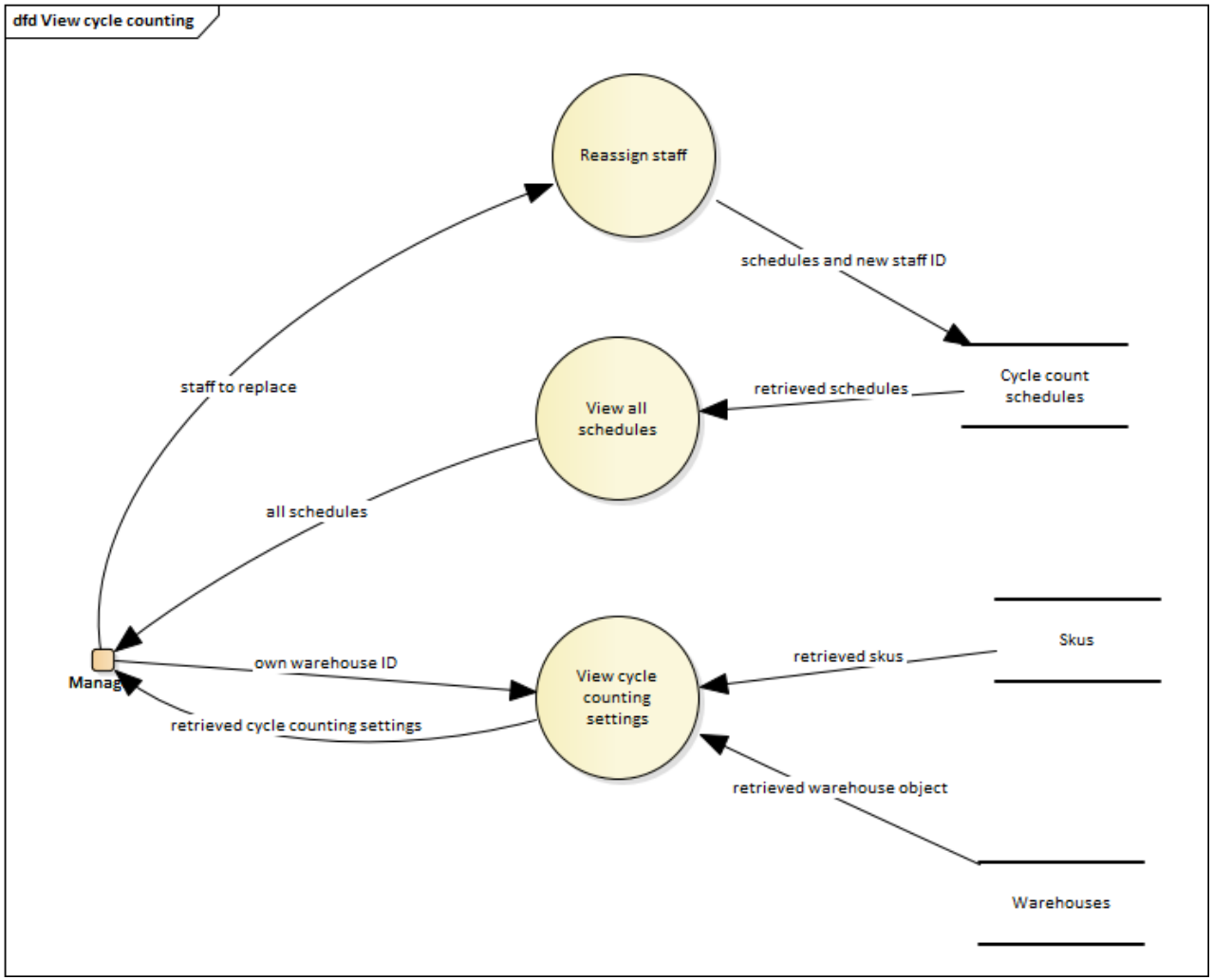

Figure 5.15: Data flow diagram level-3 (View cycle counting)

5.3.1.5 Data Flow Diagram Level-4

DFD level-4 shows subprocesses of processes in DFD level-3. This section includes the approve cycle count approval report process and the reject cycle count approval report process from DFD level-3.

5.3.1.5.1 Approve Cycle Count Approval Report

Figure 5.16 shows the subprocesses of approve cycle count approval report process from DFD level-3 (Figure 5.13).

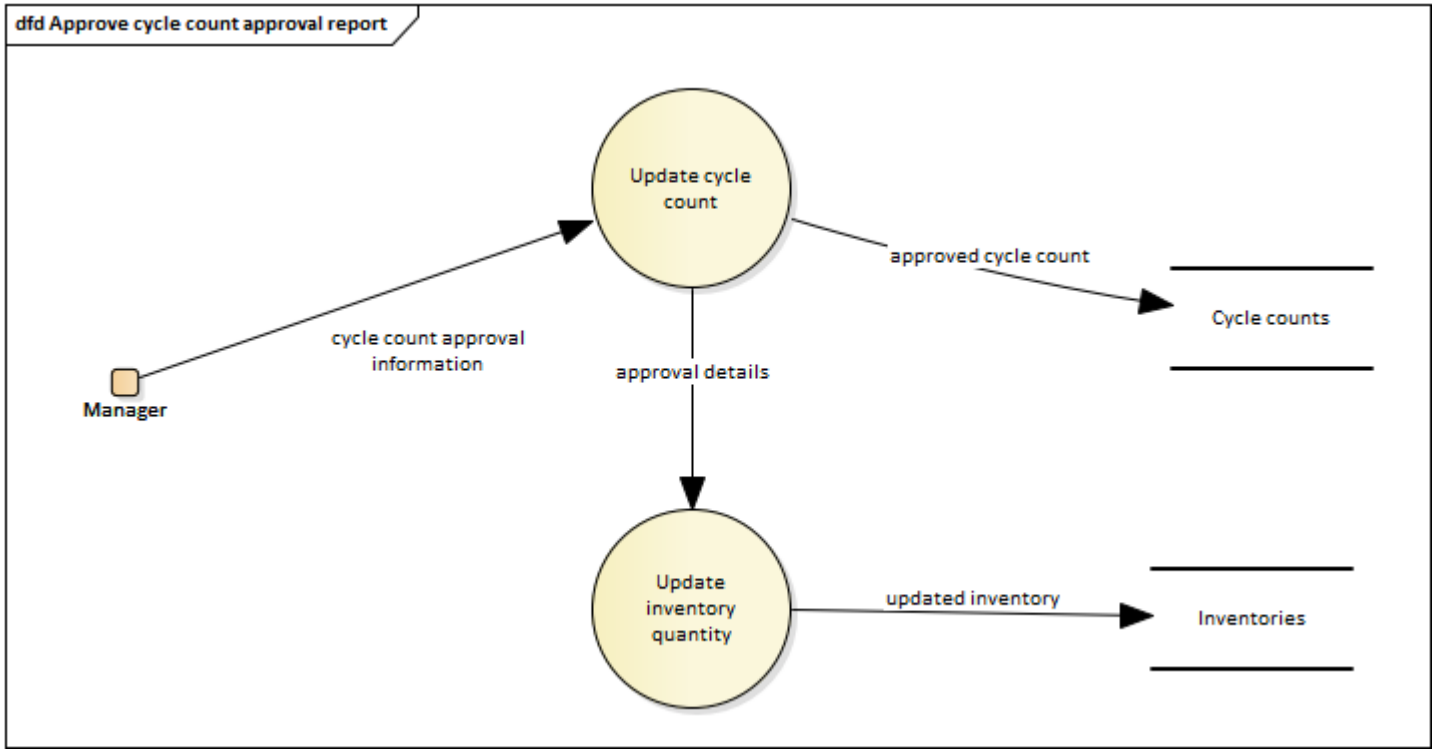

Figure 5.16: Data flow diagram level-4 (Approve cycle count approval report)

5.3.1.5.2Reject Cycle Count Approval Report

Figure 5.17 shows the subprocesses of reject cycle count approval report process from DFD level-3 (Figure 5.13).

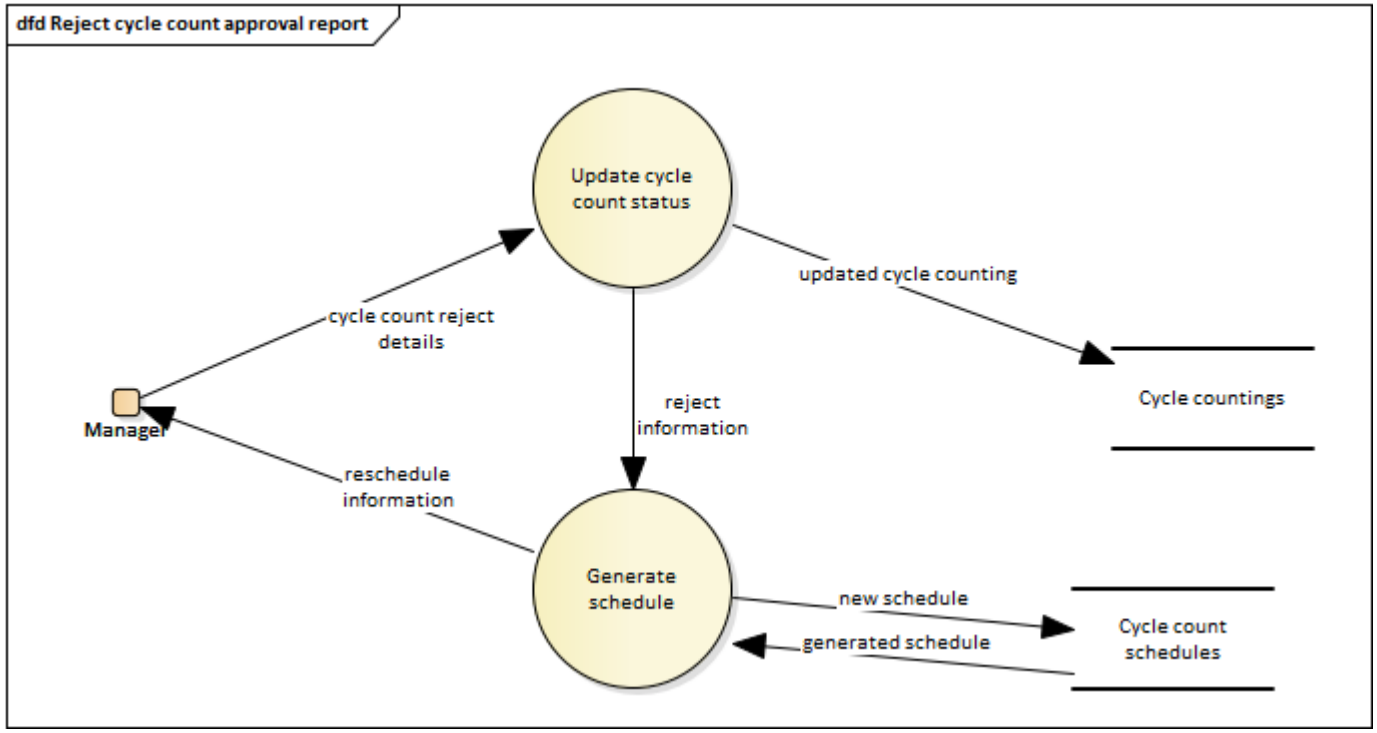

Figure 5.17: Data flow diagram level-4 (Reject cycle count approval report)

5.3.2 Interface Flow Diagram

In this section, interface flow diagrams were drawn to illustrate the flow of the warehouse inventory management system. A total of three diagrams were drawn to show the user interfaces flow for three user roles covered in this project: staff (Figure 5.18), admin (Figure 5.19) and manager (Figure 5.20).

5.3.2.1 Staff

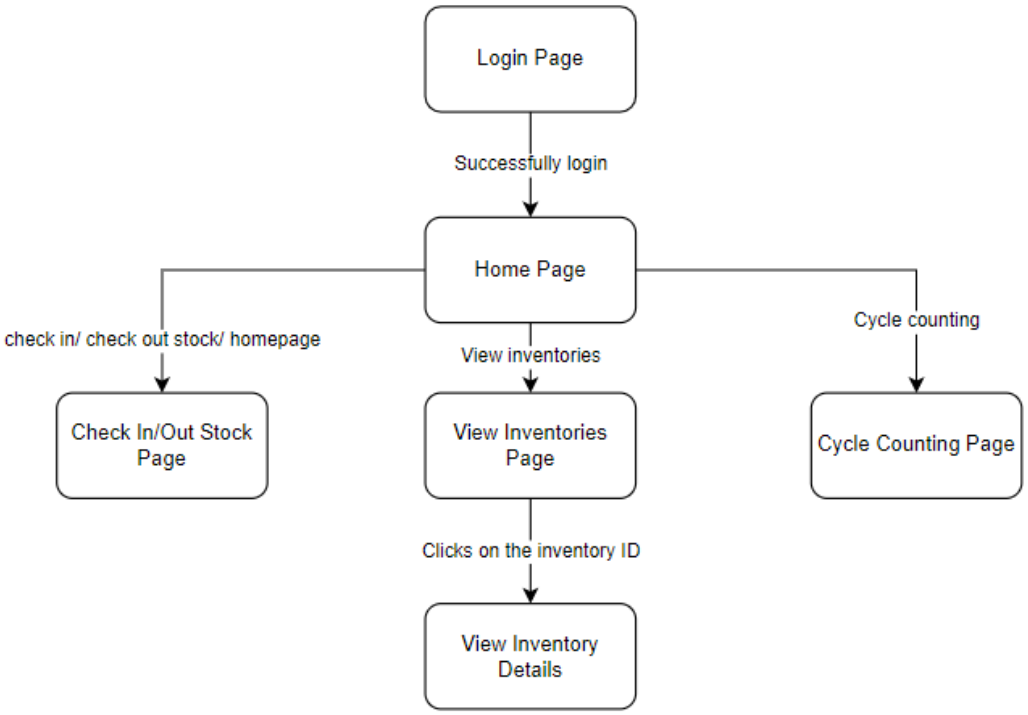

Figure 5.18: Interface flow diagram for staff role

5.3.2.2 Admin

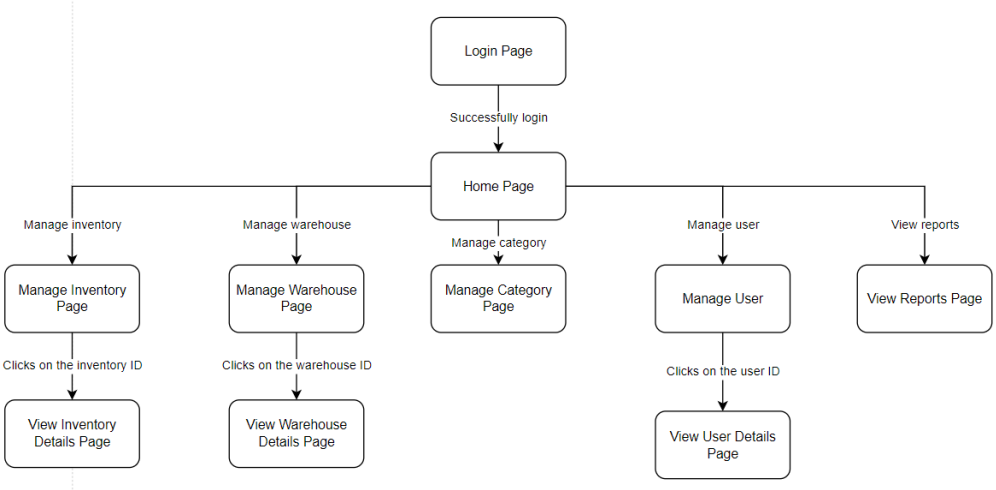

Figure 5.19: Interface flow diagram for admin role

5.3.2.3 Manager

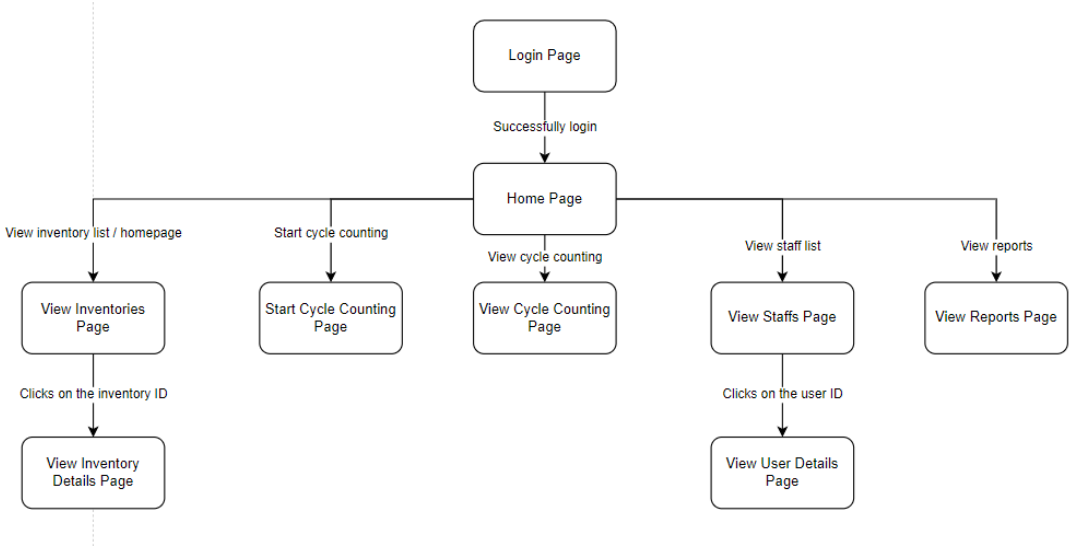

Figure 5.20: Interface flow diagram for manager role

5.4 User Interface Design

This section shows all user interface designs implemented in the system and is divided into subsections based on user roles: admin, manager and staff.

5.4.1 User (Admin, Manager and Staff)

This section shows the module that can be used by all users, which is the login module so that the user can log in to the application. After login, they will be presented with different modules based on their roles and will be covered in the following sections.

5.4.1.1 Login Account

The Figure 5.21 shows the user interface design for the login account page.

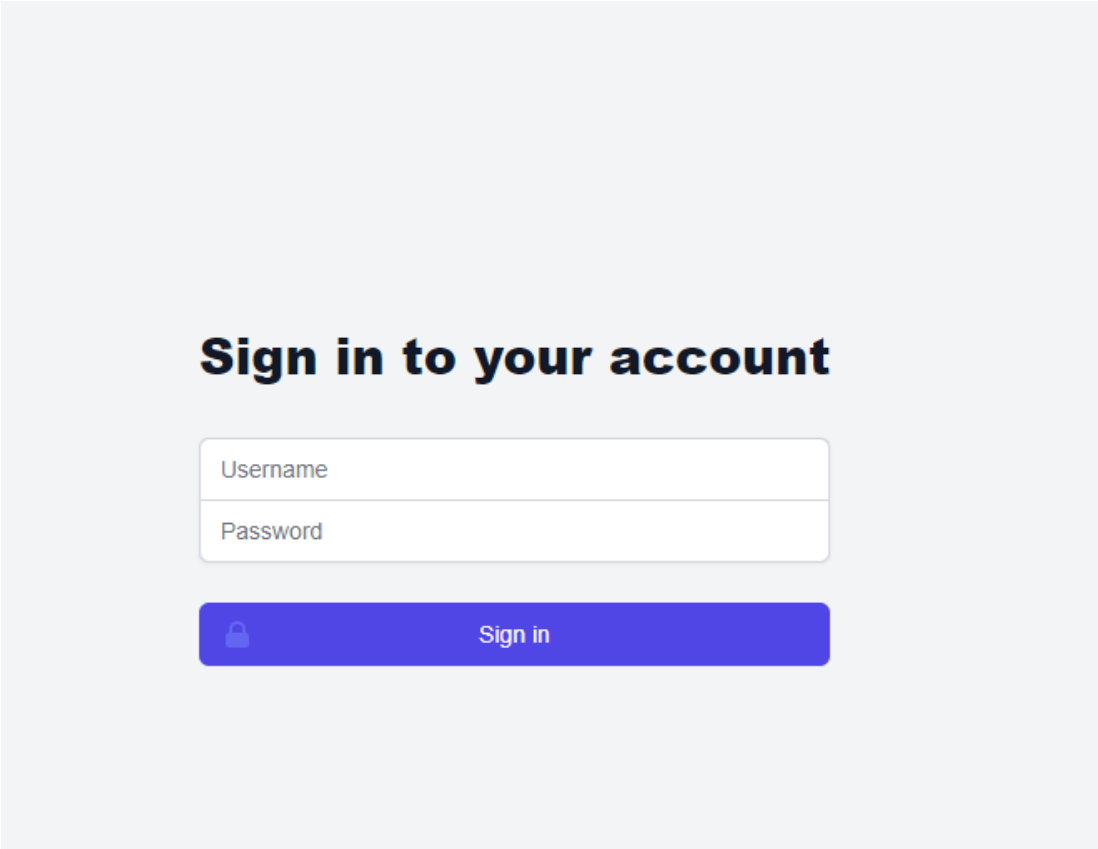

Figure 5.21: User interface design for login account

5.4.2 Admin

This section will list all modules for user role admin, which covers the manage inventory, manage warehouse, manage category, and manage user modules.

5.4.2.1 Manage Inventory

Figure 5.22 shows the user interface design for the manage inventory page and the Figure 5.23 shows the add new inventory modal that shows up after the user clicked on the “Add” button on the manage inventory page. Figure 5.24 is the update inventory modal displayed if the user clicked on the “Edit” button for one of the inventories

displayed on the manage inventory page, and the Figure 5.25 shows the confirmation modal of the inventory deletion.

5.4.2.1.1 Manage inventory page

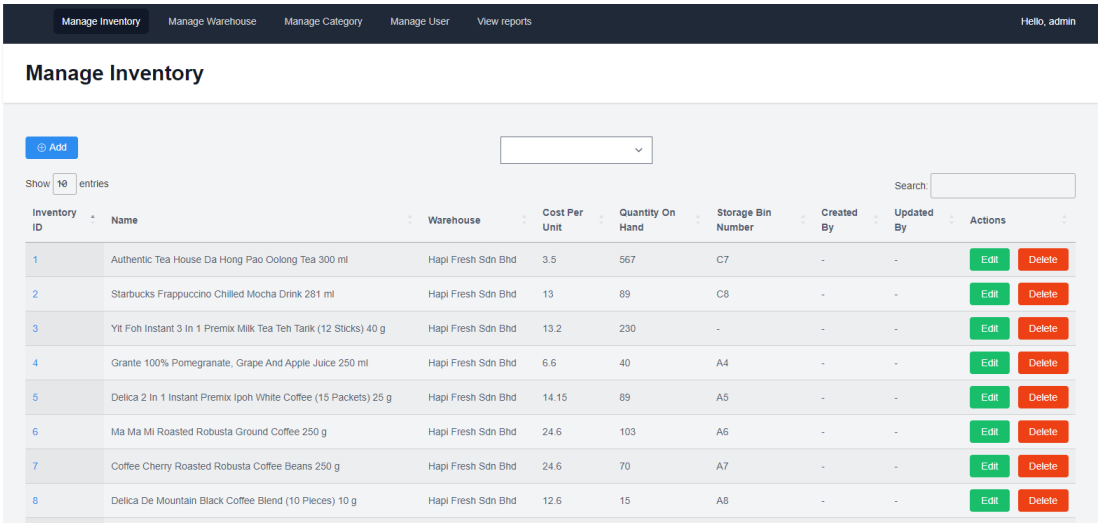

Figure 5.22: User interface design for manage inventory page

5.4.2.1.2 Add inventory

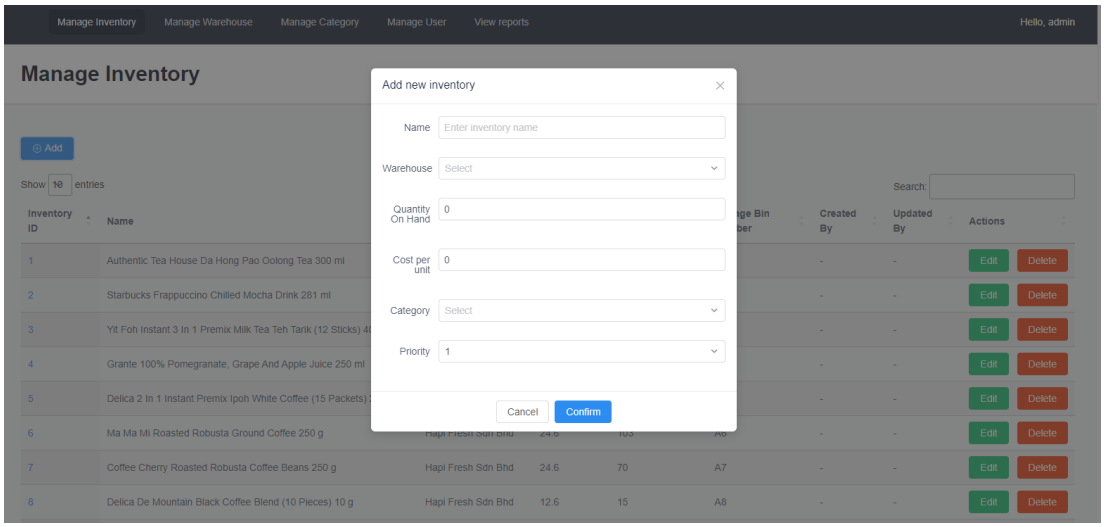

Figure 5.23: User interface design for adding inventory

5.4.2.1.3 Update inventory

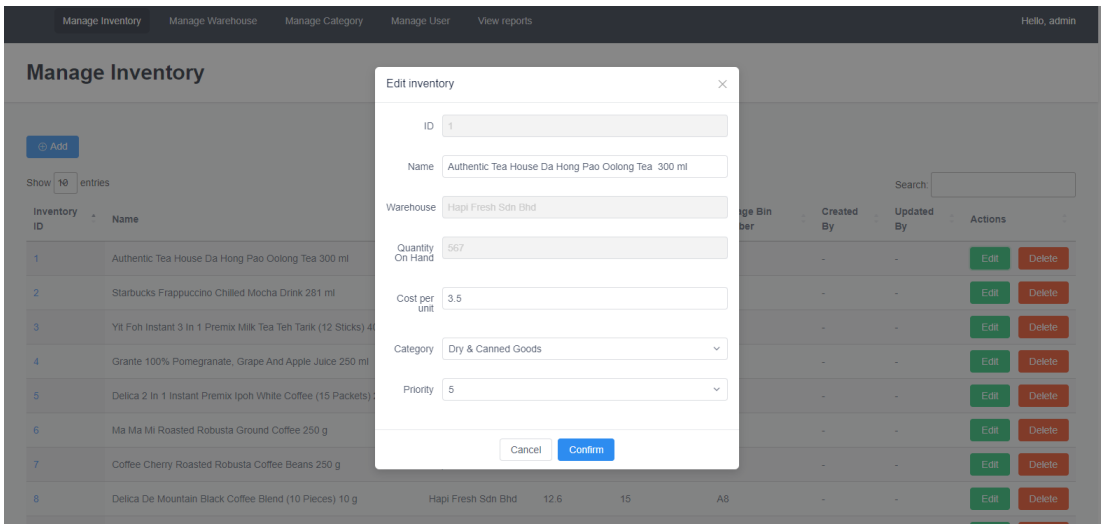

Figure 5.24: User interface design for updating inventory

5.4.2.1.4 Delete inventory

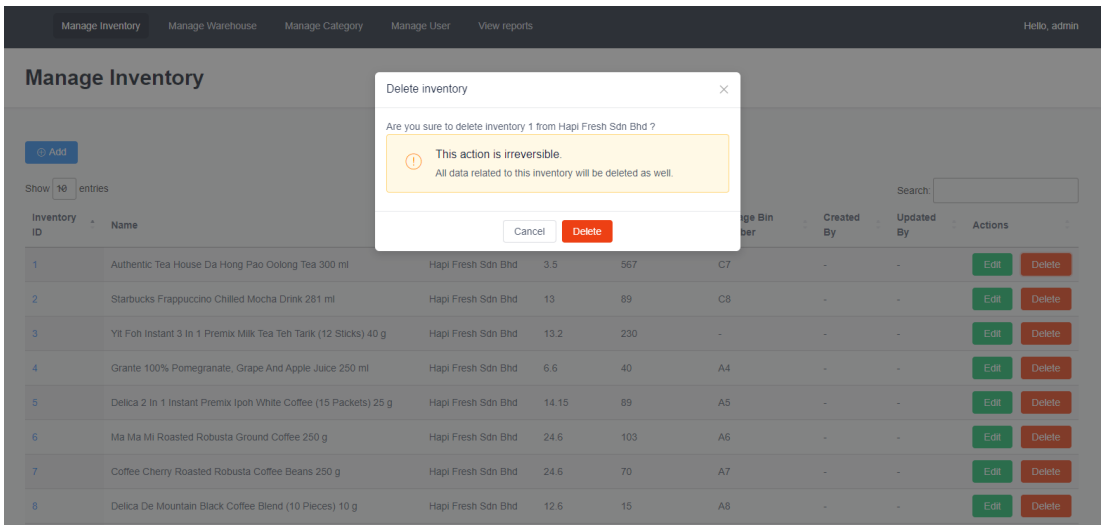

Figure 5.25: User interface design for deleting inventory

5.4.2.2 Manage Warehouse

Figure 5.26 shows the user interface design for the manage warehouse page. Figure 5.27 and Figure 5.28 are the user interface designs that display the warehouse details once the user clicked on the warehouse ID on the manage warehouse page. On the warehouse details page, the staff list is shown in Figure 5.27 whereas the list of storage bins of the warehouse is shown in Figure 5.28. The multi-assigning category to storage bins and the editing inventory of storage bin modules are shown in Figure 5.29 and 5.30 respectively, whereas the addition, update and deletion of a warehouse are in Figure 5.31, 5.32 and 5.33 respectively.

5.4.2.2.1 Manage warehouse page

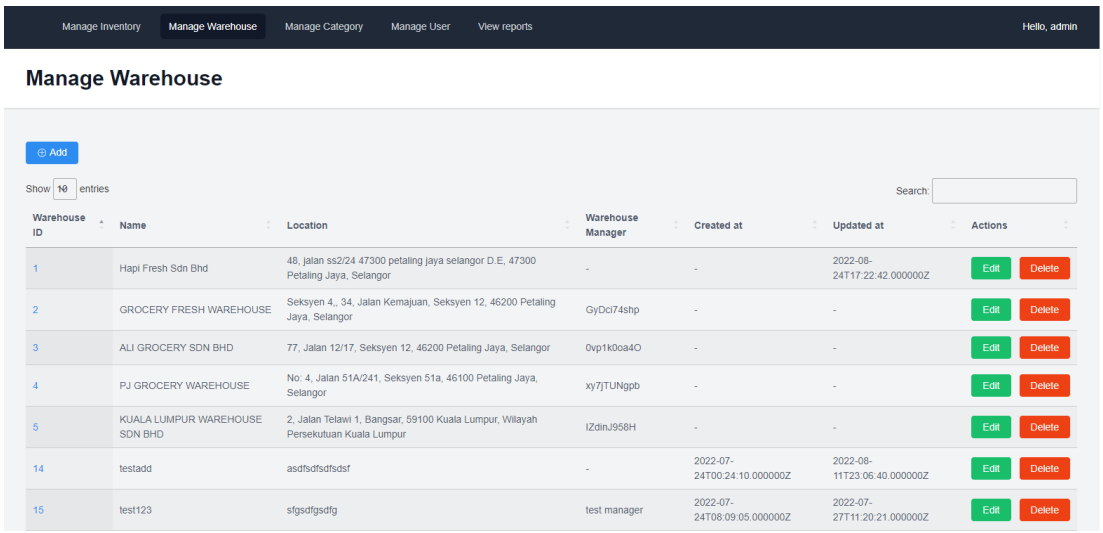

Figure 5.26: User interface design for manage warehouse page

5.4.2.2.2 View warehouse details & view staff list

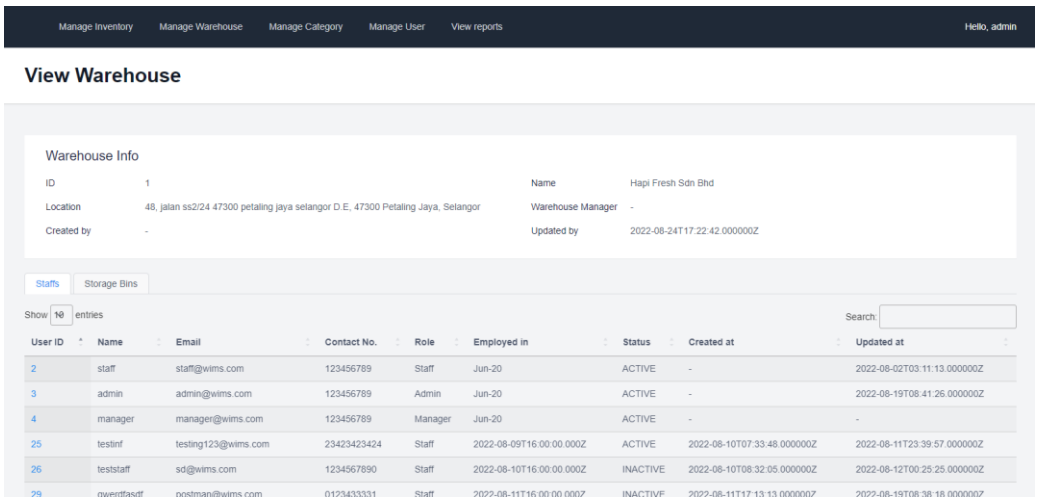

Figure 5.27: User interface design for view warehouse details & view staff list

5.4.2.2.3 View warehouse details & view storage bins

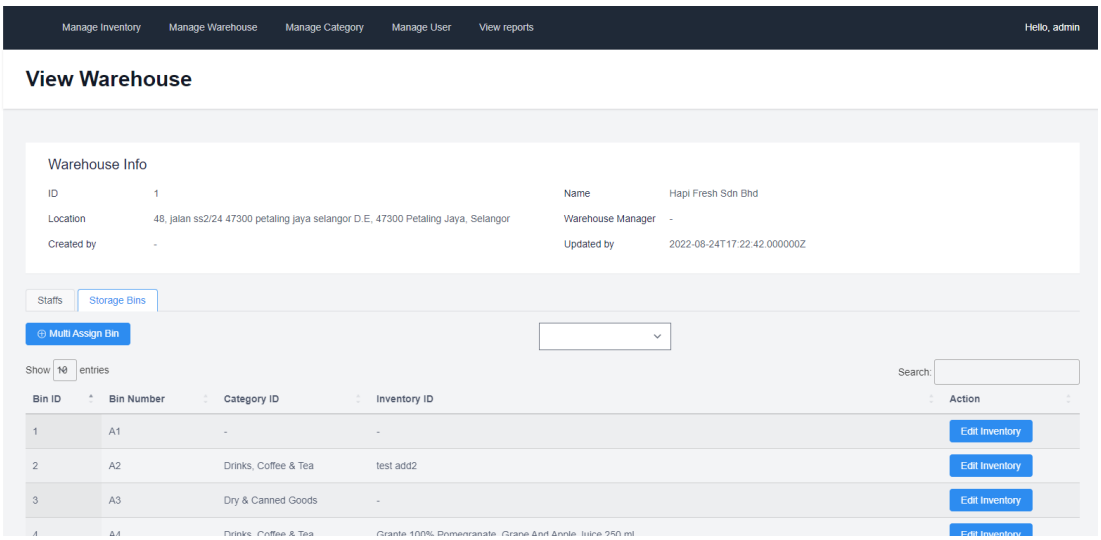

Figure 5.28: User interface design for view warehouse details & view storage bins

5.4.2.2.4 Multi-assign storage bins

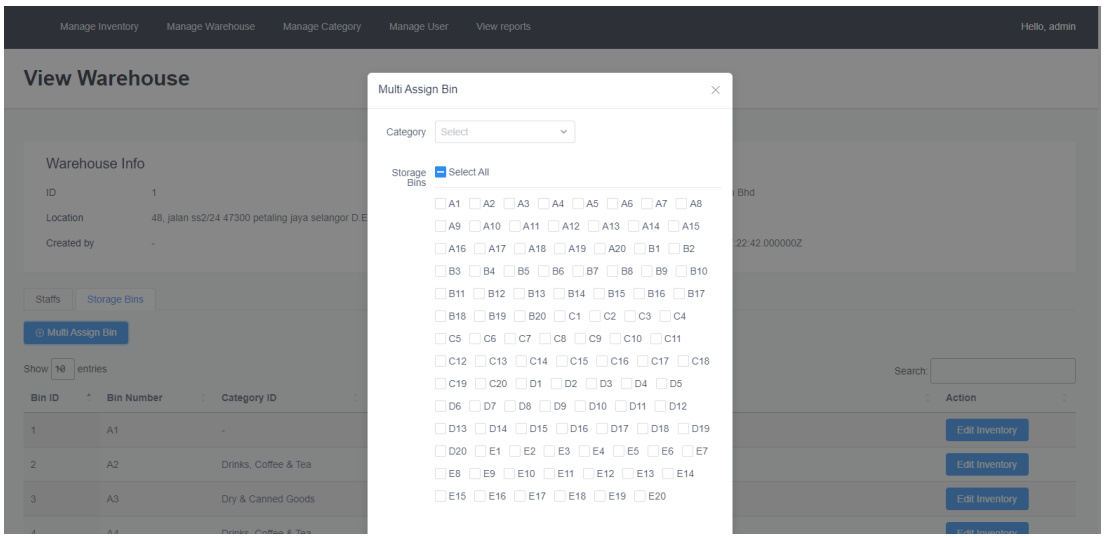

Figure 5.29: User interface design for multi-assigning storage bins

5.4.2.2.5 Edit storage bin's inventory

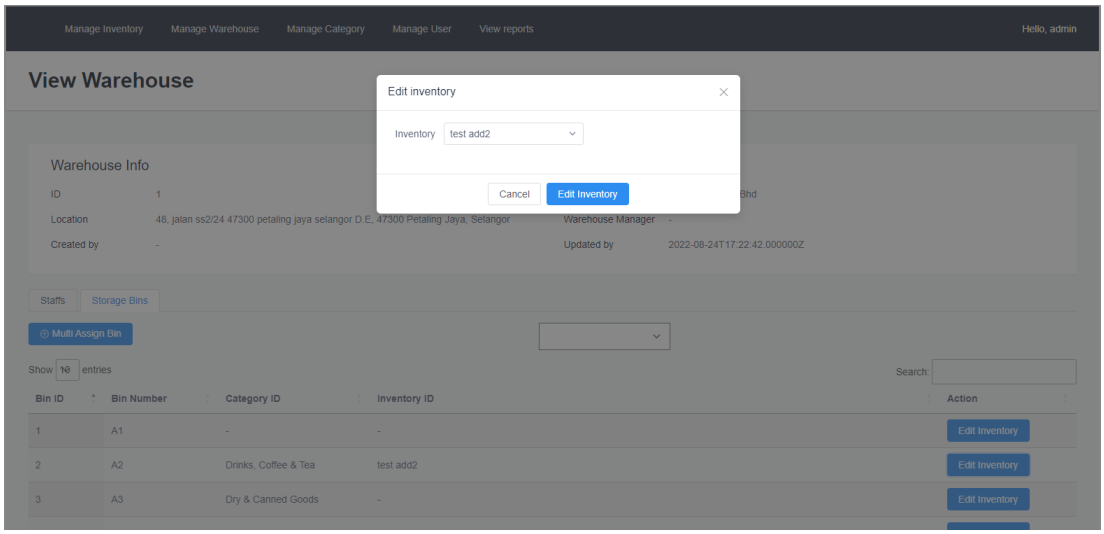

Figure 5.30: User interface design for editing storage bin's inventory

5.4.2.2.6 Add new warehouse

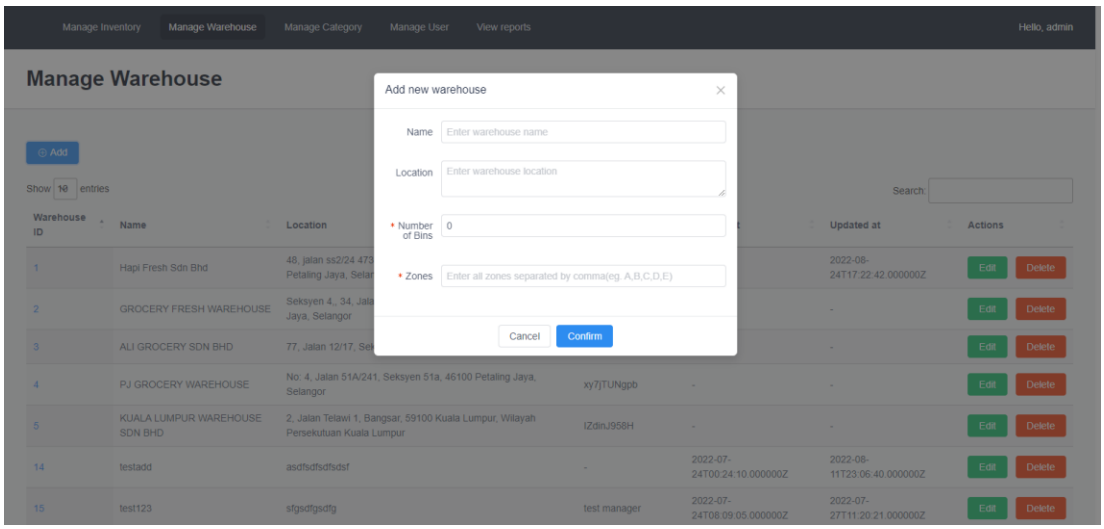

Figure 5.31: User interface design for adding new warehouse

5.4.2.2.7 Edit warehouse details

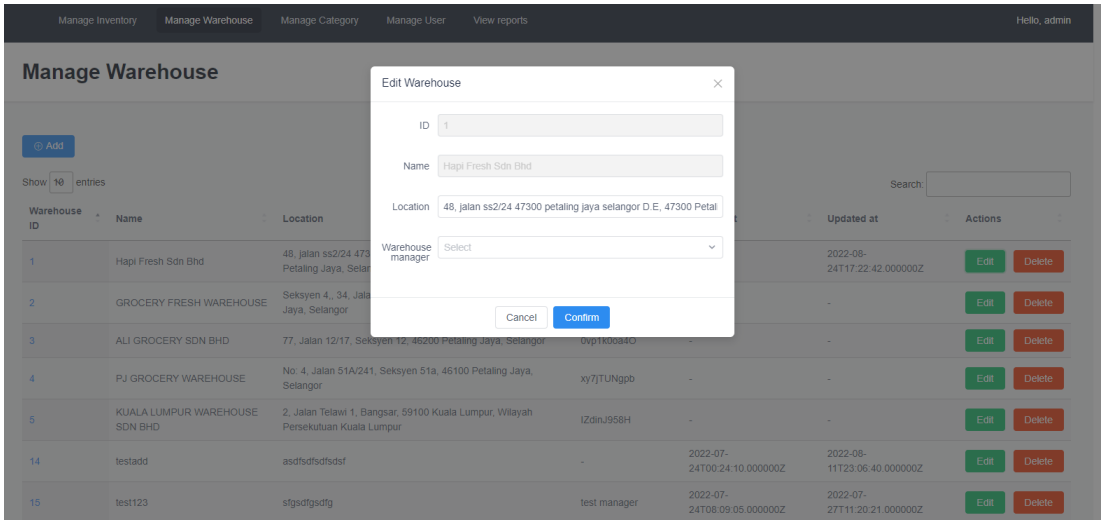

Figure 5.32: User interface design for editing warehouse details

5.4.2.2.8 Delete warehouse

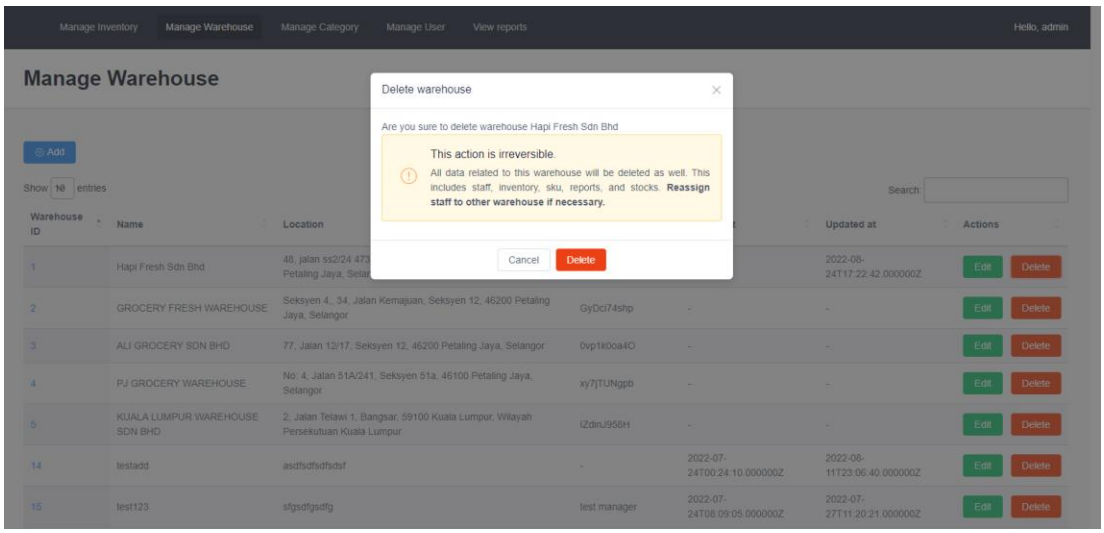

Figure 5.33: User interface design for deleting warehouse

5.4.2.3 Manage Category

Figure 5.34 shows the user interface design for the manage category page and the add, update and delete of category is shown in Figure 5.35, 5.36 and 5.37 respectively.

5.4.2.3.1 Manage category page

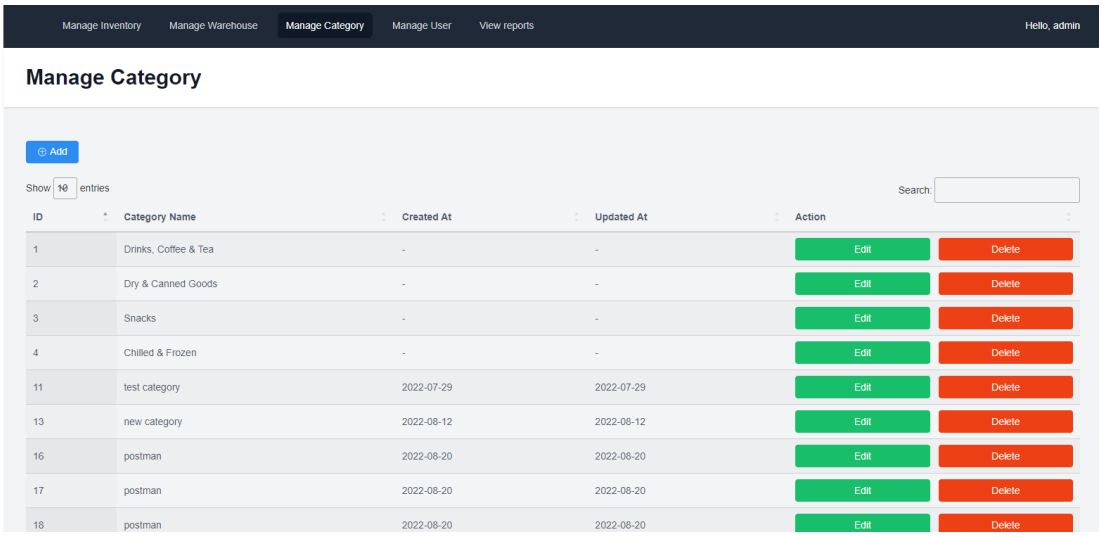

Figure 5.34: User interface design for managing category page

5.4.2.3.2 Add new category

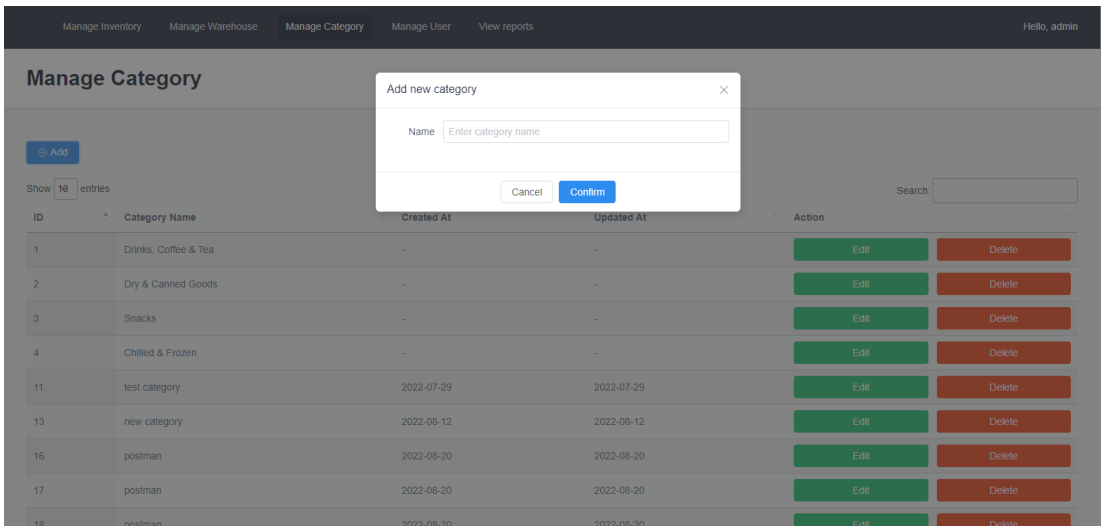

Figure 5.35: User interface design for adding new category

5.4.2.3.3 Edit category

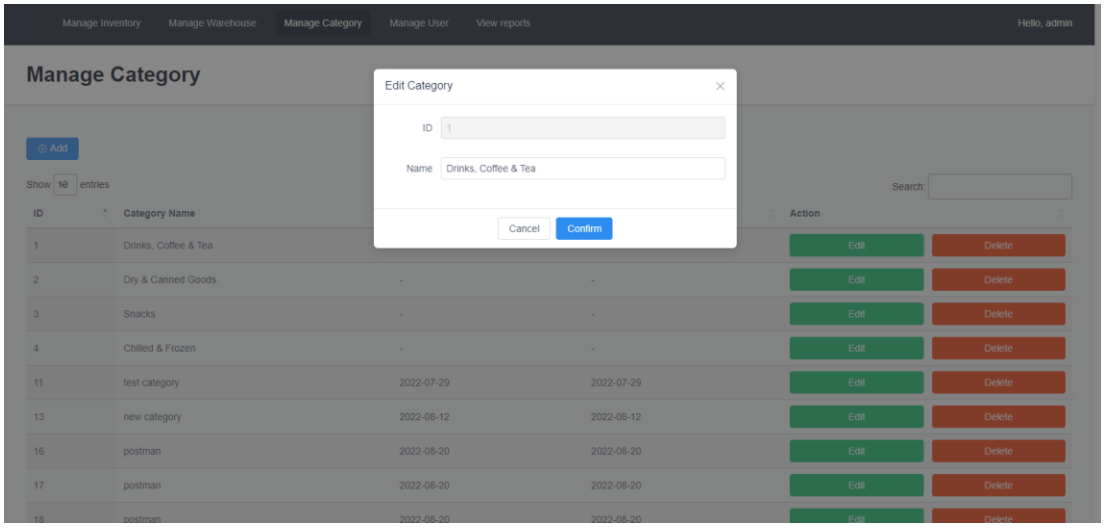

Figure 5.36: User interface design for editing category

5.4.2.3.4 Delete category

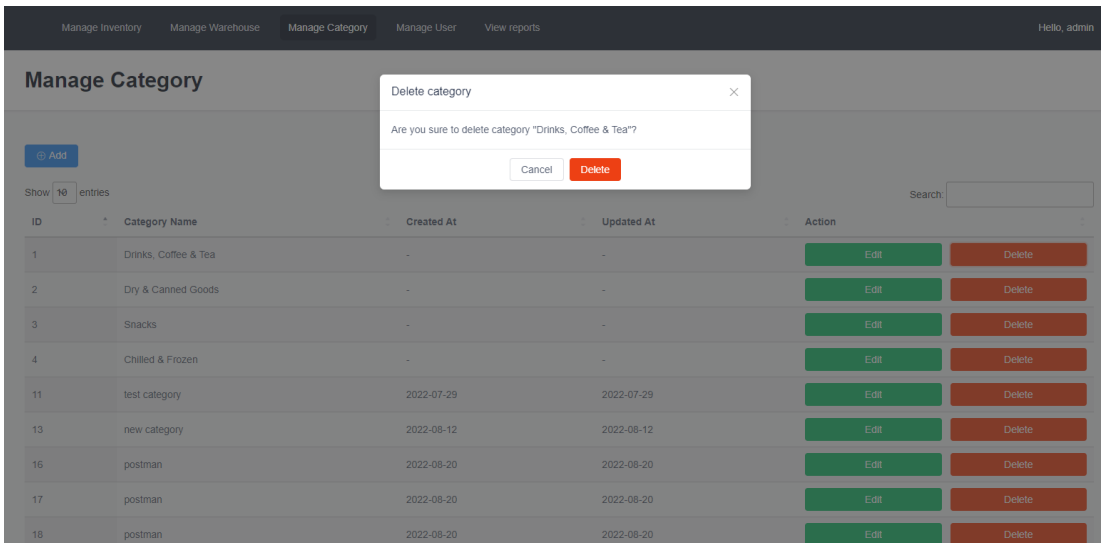

Figure 5.37: User interface design for deleting category

5.4.2.4 Manage User

The manage user page, user details page, add new user, update existing user and delete user interface design are shown in Figure 5.38, Figure 5.39, Figure 5.40, Figure 5.41 and Figure 5.42 respectively.

5.4.2.4.1 Manage user page

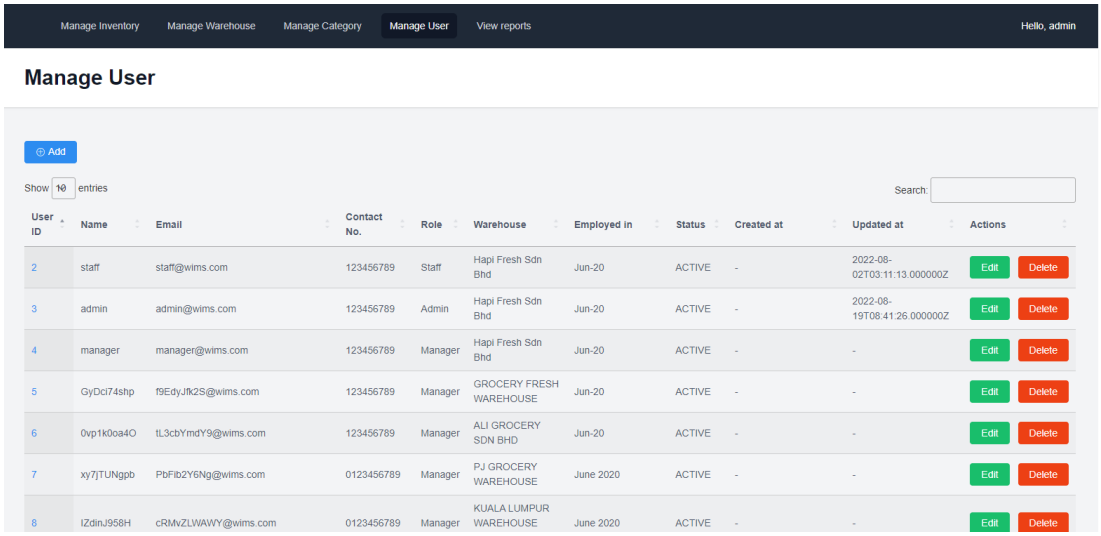

Figure 5.38: User interface design for manage user page

5.4.2.4.2 View user details

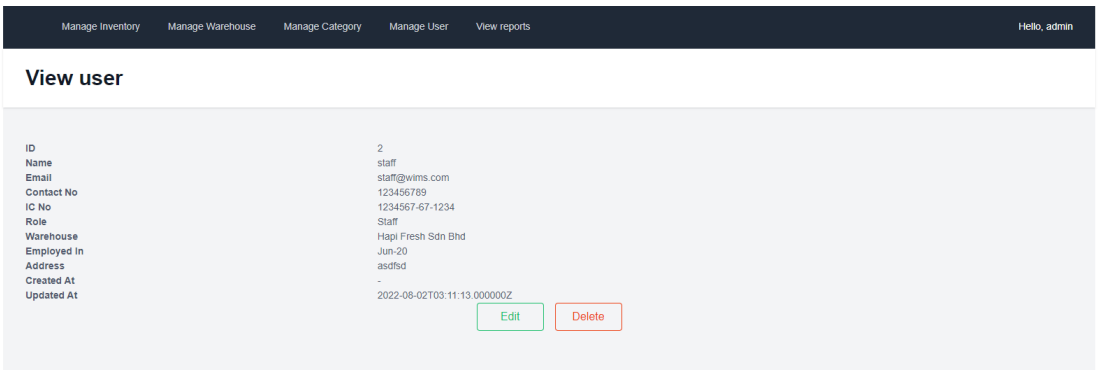

Figure 5.39: User interface design for viewing user details

5.4.2.4.3 Add new user

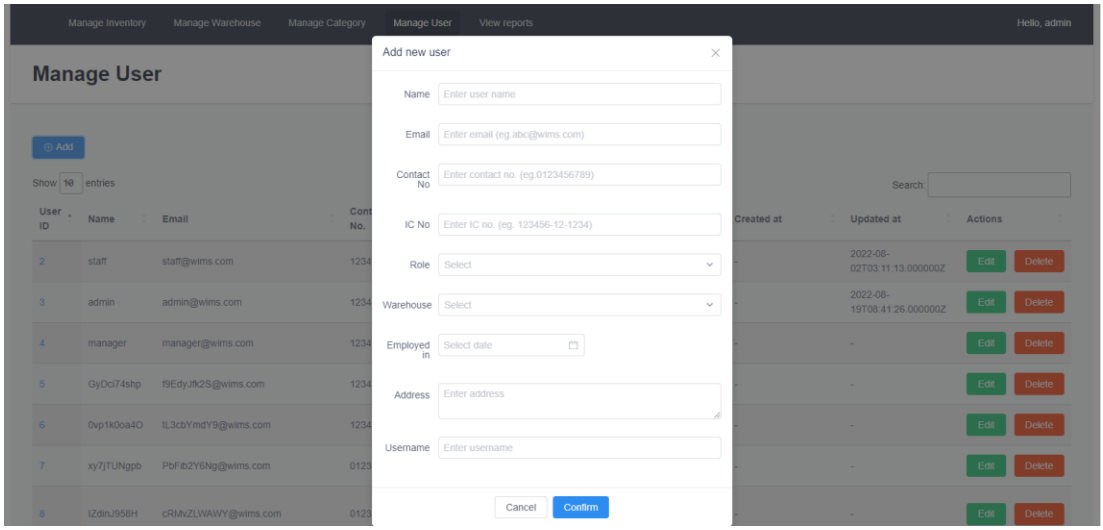

Figure 5.40: User interface design for adding new user

5.4.2.4.4 Edit user

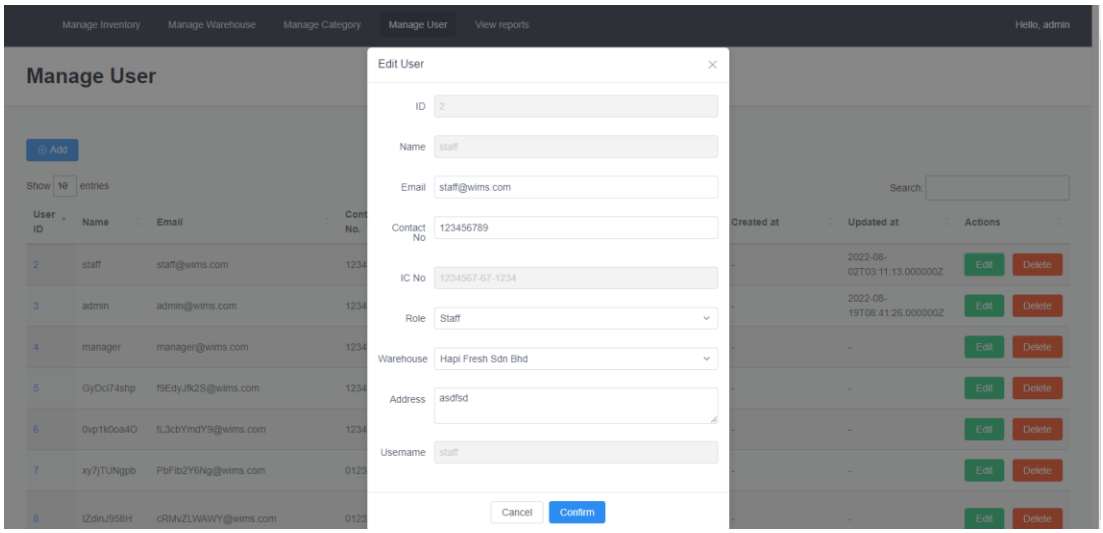

Figure 5.41: User interface design for editing user

5.4.2.4.5 Delete user

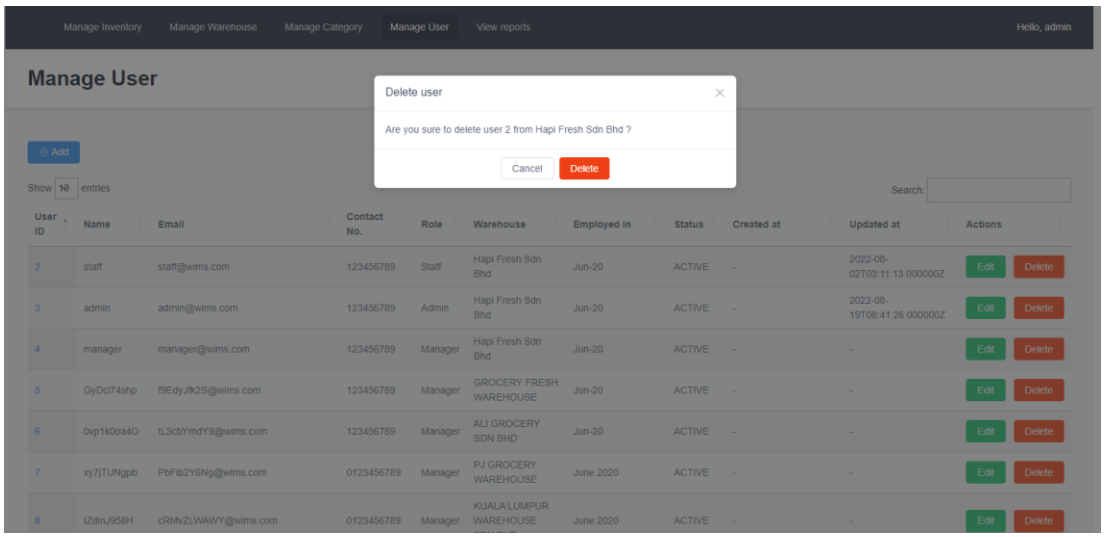

Figure 5.42: User interface design for deleting user

5.4.3 Manager

This section will show all modules that are available for user role manager, which includes the manage cycle count approval report module, manage cycle counting and view all staff modules.

5.4.3.1 Manage Cycle Count Approval Report

The Figure 5.43 displays the user interface design for viewing the list of cycle counting approval reports. Users can approve or reject the approval report in which the confirmation modal designs are shown in Figure 5.44 and Figure 5.45 respectively.

5.4.3.1.1 View reports- cycle count approval reports

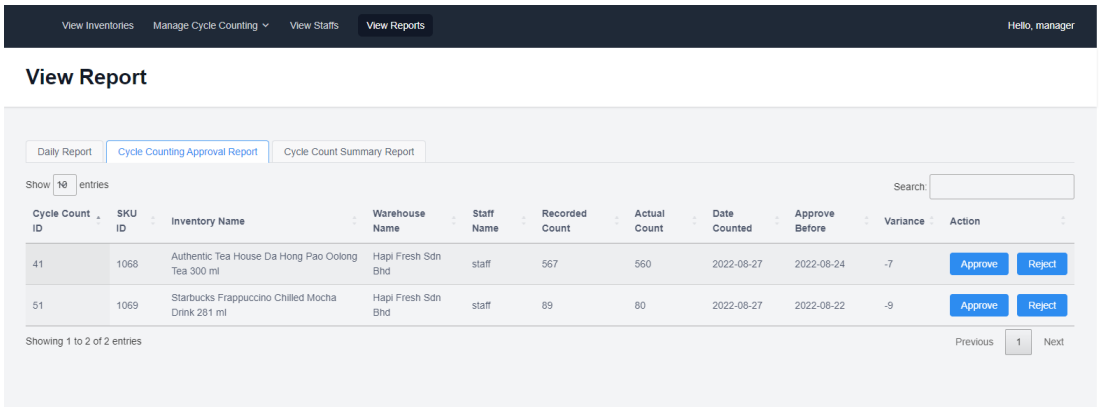

Figure 5.43: User interface design for viewing cycle count approval reports

5.4.3.1.2 Approve cycle count approval report

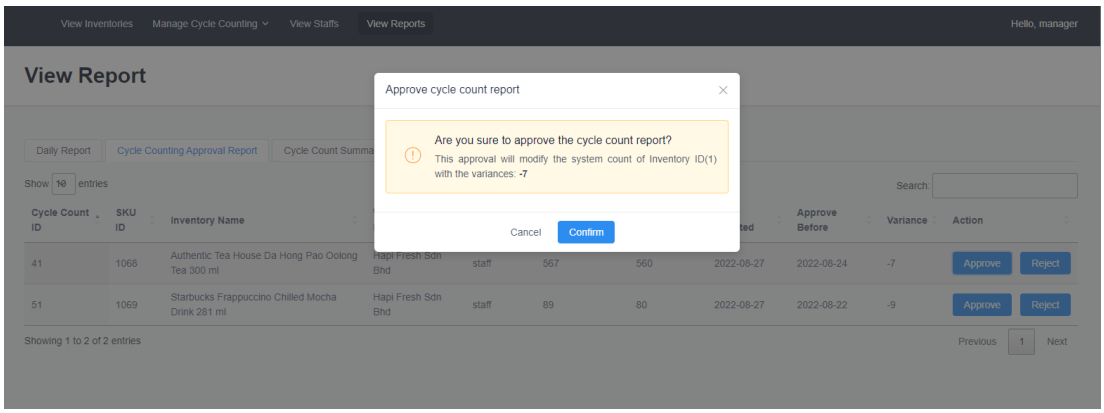

Figure 5.44: User interface design for approving cycle count approval report

5.4.3.1.3 Reject cycle count approval report

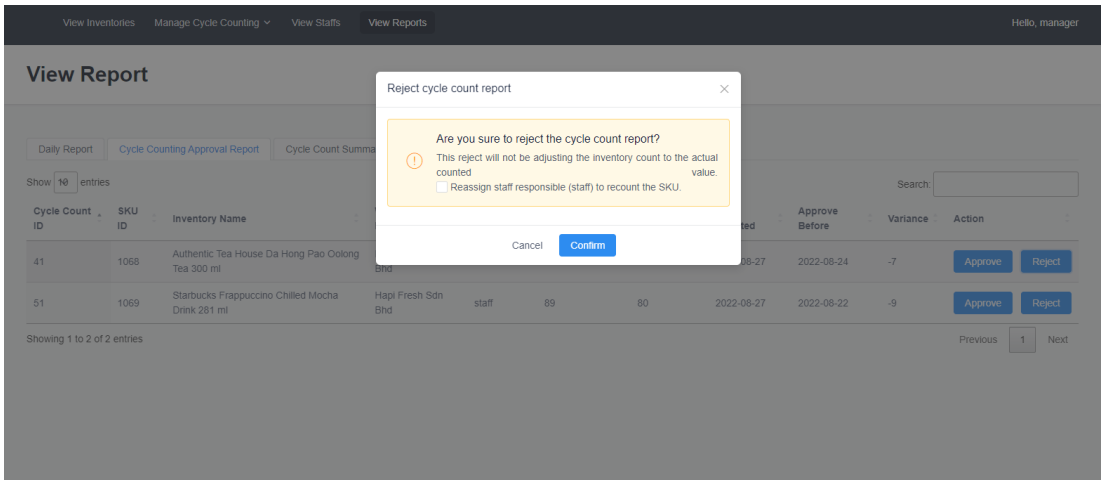

Figure 5.45: User interface design for rejecting cycle count approval report

5.4.3.2 Manage Cycle Counting

Managers can also manage the cycle counting by either starting a new cycle counting or viewing the current cycle counting schedules.

5.4.3.2.1 Start cycle counting

The start cycle counting page interface design is shown in Figure 5.46. The manager can submit the form after filling in all of the information required and the system will automatically generate cycle counting schedules based on the inputs.

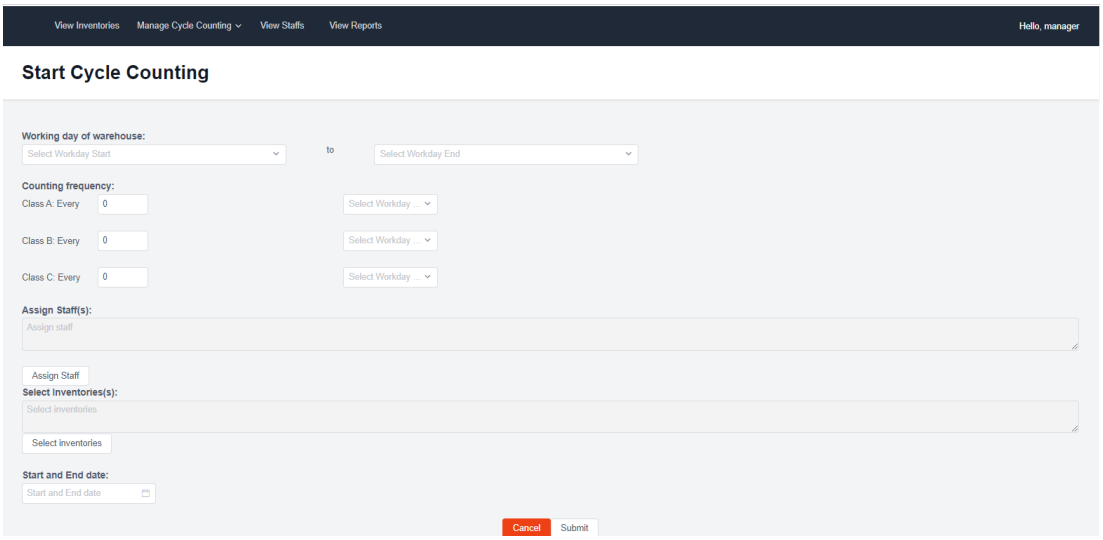

Figure 5.46: User interface design for starting cycle counting

5.4.3.2.2 View cycle counting

Figure 5.47 shows the view cycle counting page that displays all cycle counting schedules generated. The warehouse manager can reassign all schedules assigned to one staff to another staff if the original staff is leaving his or her position as shown in Figure 5.48. Additionally, the manager can also view the current cycle counting settings and the assigned staff as well as selected SKUs as shown in Figure 5.49, Figure 5.50 and Figure 5.51 respectively.

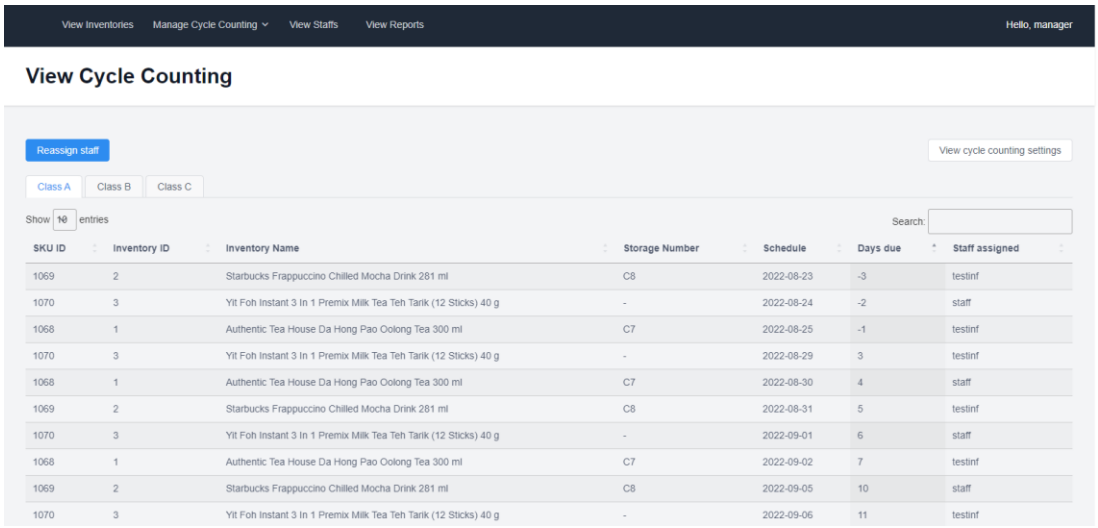

Figure 5.47: User interface design for view cycle counting page

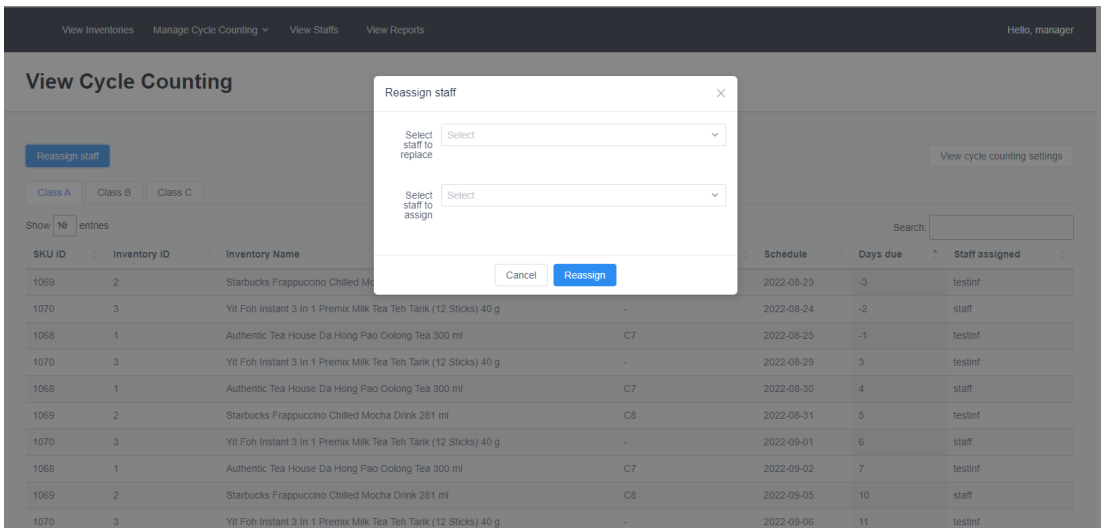

Figure 5.48: User interface design for reassigning staff

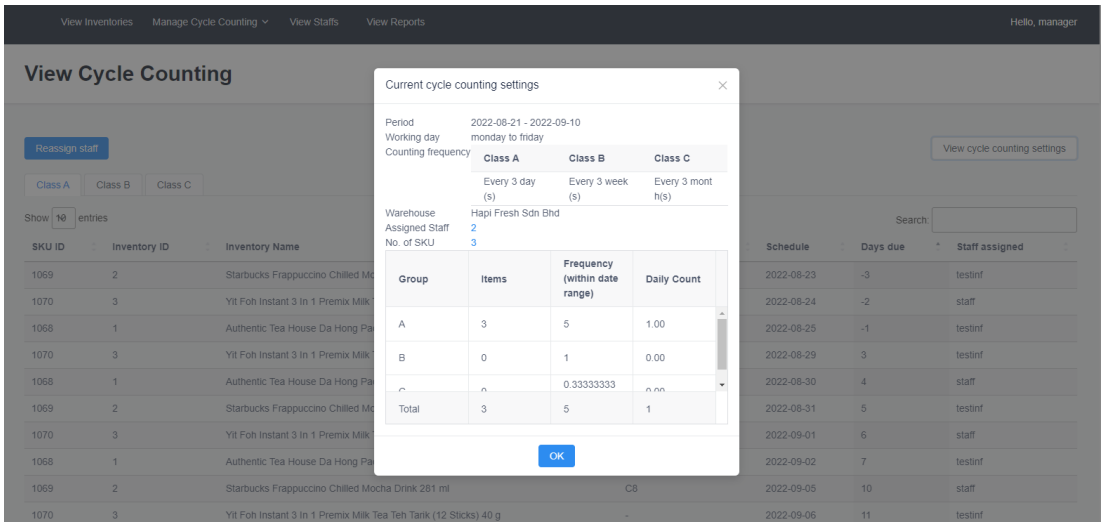

Figure 5.49: User interface design for viewing cycle counting settings

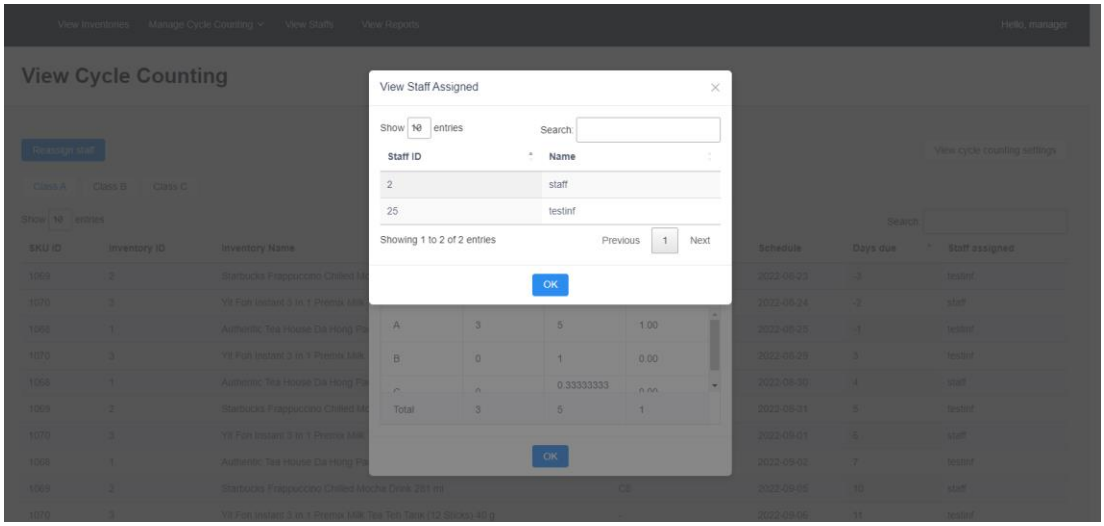

Figure 5.50: User interface design for viewing assigned staff for cycle counting

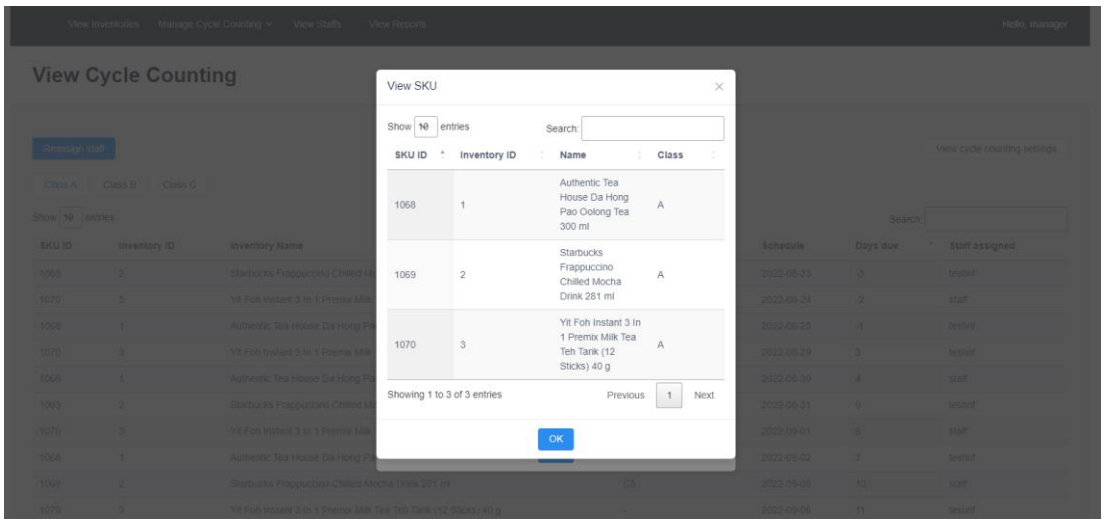

Figure 5.51: User interface design for viewing SKUs for cycle counting

5.4.3.3 View All Staff

The Figure 5.52 shows the interface design of the view staff module, in which the manager's warehouse details and all staff working in the manager's warehouse are displayed.

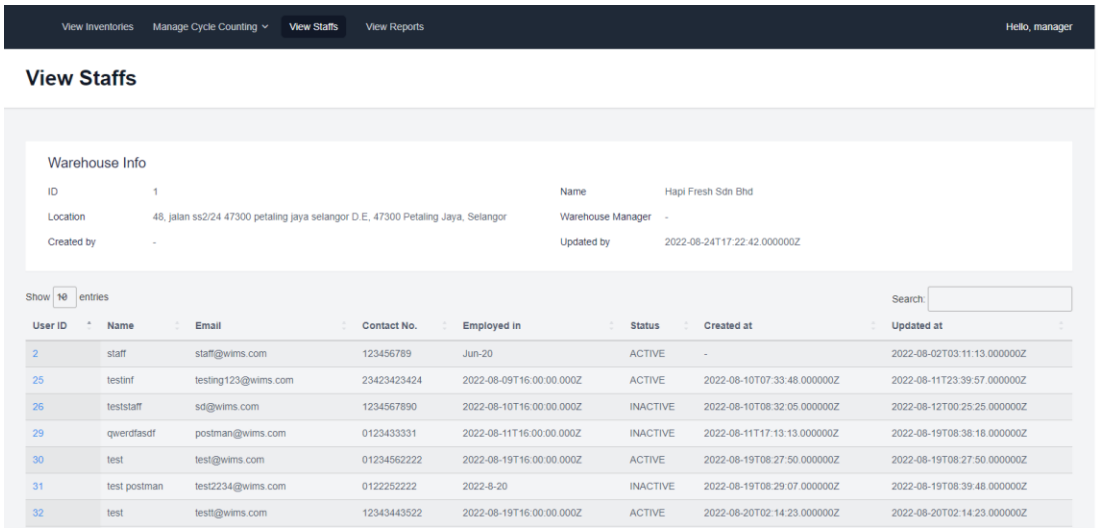

Figure 5.52: User interface design for viewing all staff

5.4.4 Staff

This section will list out all modules for the user role staff, which are the perform cycle counting and record inventory modules.

5.4.4.1 Perform Cycle Counting

The Figure 5.53 shows the list of cycle counting schedules assigned to the staff. The staff can click the "Count" button on one of the cycle-counting schedules to perform cycle counting as shown in Figure 5.54 and Figure 5.55.

5.4.4.1.1 Cycle counting page

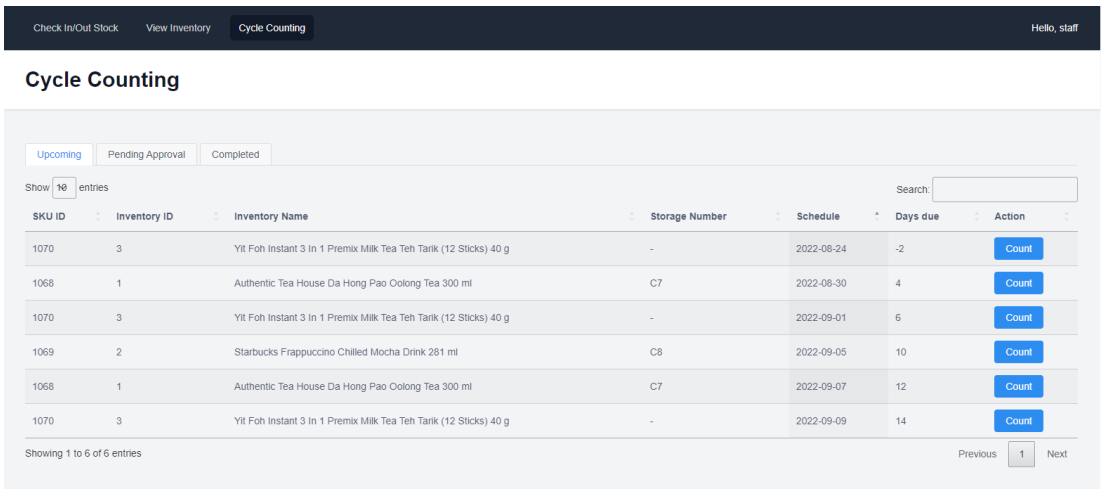

Figure 5.53: User interface design for cycle counting page

5.4.4.1.2 Count SKU (no bin assigned)

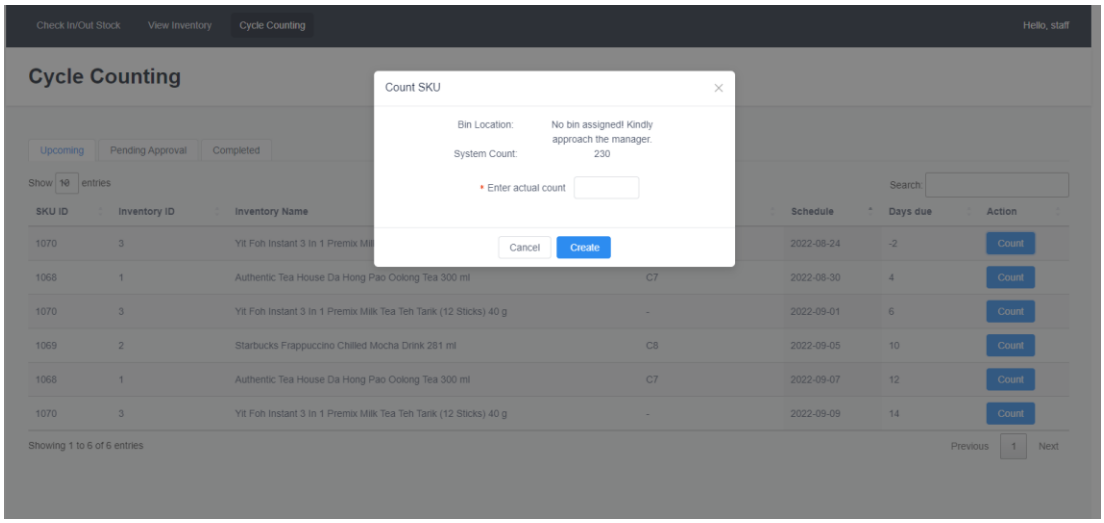

Figure 5.54: User interface design for counting SKU with no bin assigned

5.4.4.1.3 Count SKU (with bin assigned)

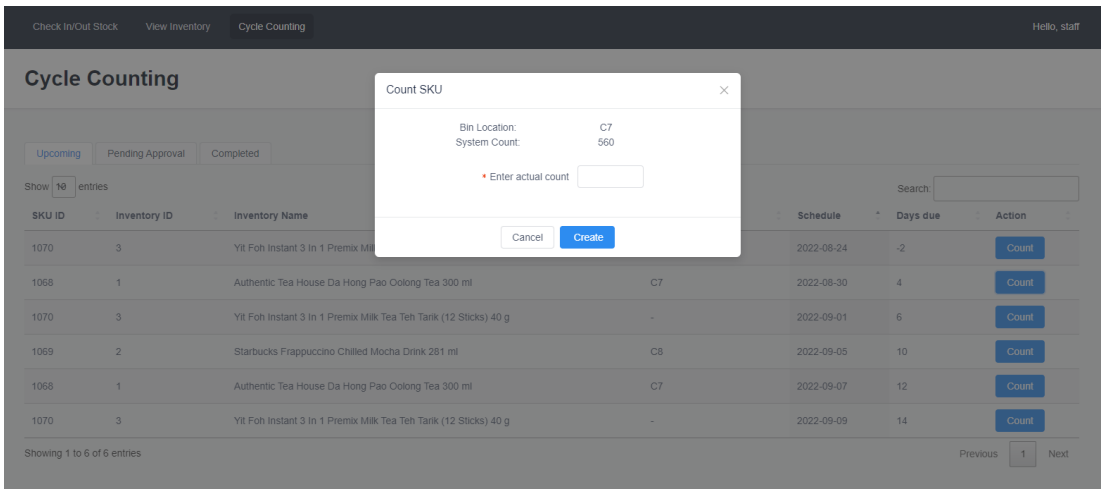

Figure 5.55: User interface design for counting SKU with bin assigned

5.4.4.2 Record Inventory

On the check-in/ out stock page, the list of inventories in the staff's warehouse is displayed in Figure 5.56. The staff can choose one of the inventories to check in or check out some stocks. The checking in and checking out stock modal is as shown in Figure 5.57 and Figure 5.58 respectively.

5.4.4.2.1 Check in/out stock page

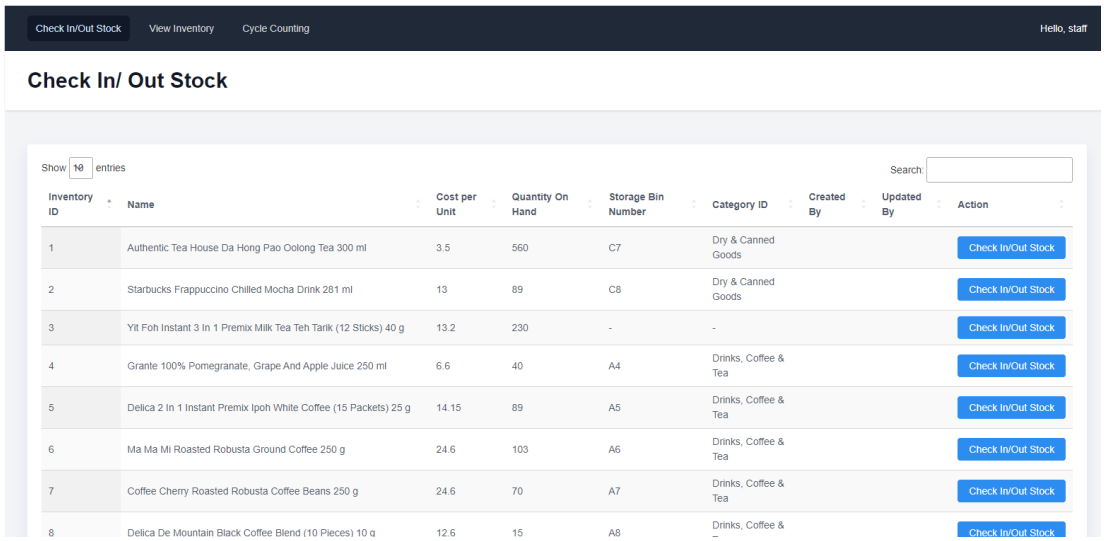

Figure 5.56: User interface design for check in/out stock page

5.4.4.2.2 Check in stock

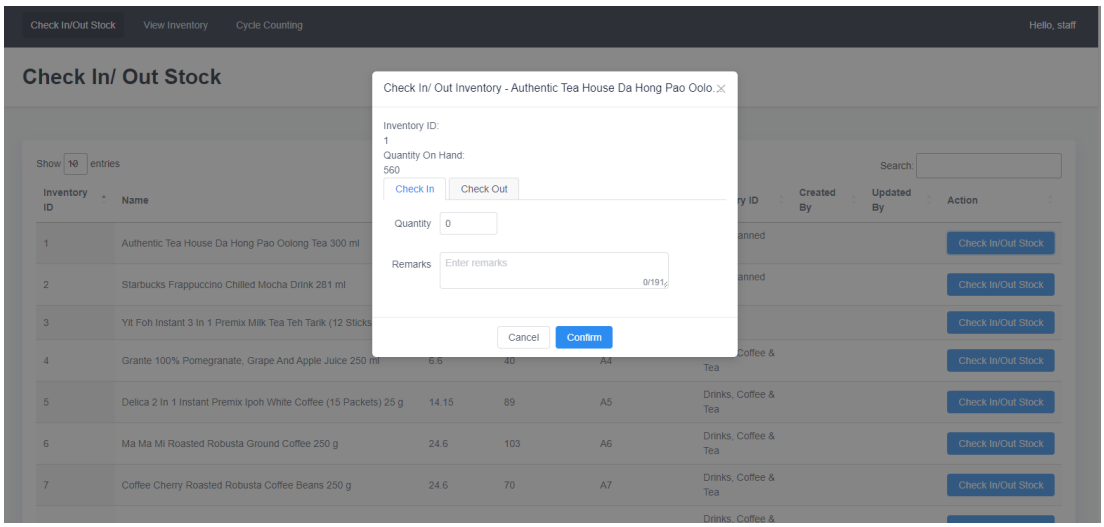

Figure 5.57: User interface design for checking in stock

5.4.4.2.3 Check out stock

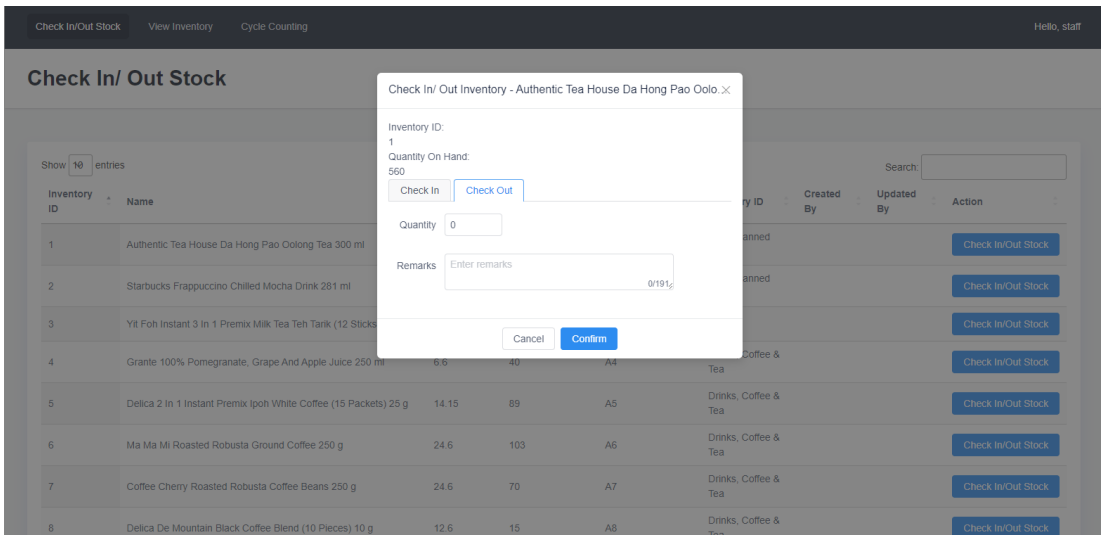

Figure 5.58: User interface design for checking out stock

5.4.5 Staff & Manager

This section outlines the view all inventories module that is available for staff and manager.

5.4.5.1 View All Inventories

The Figure 5.59 shows the user interface design for the view inventory page, whereas the Figure 5.60 shows the inventory details page when the user clicked on the inventory ID on the view inventory page.

5.4.5.1.1 View inventory page

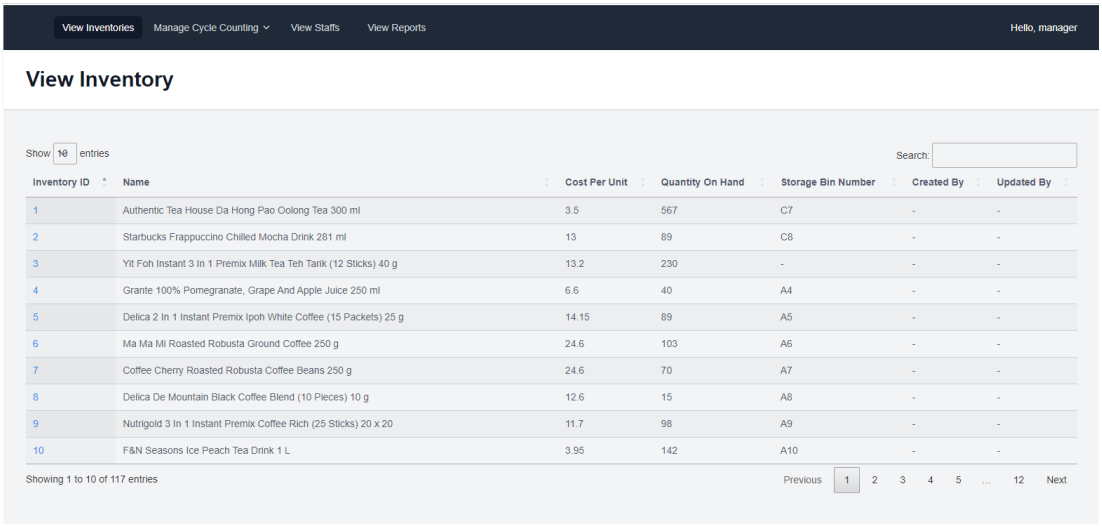

Figure 5.59: User interface design for view inventory page

5.4.5.1.2 View inventory details

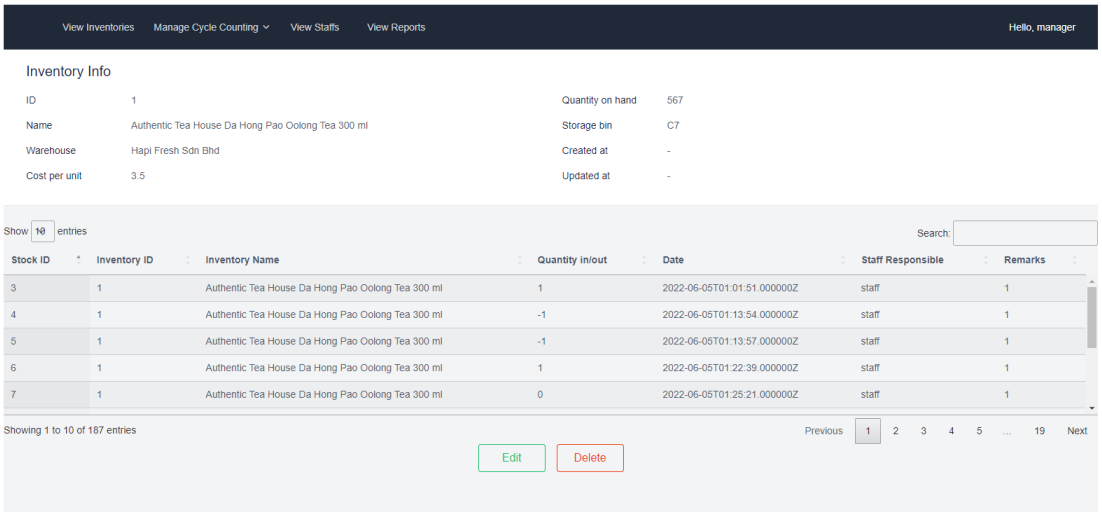

Figure 5.60: User interface design for view inventory details

5.4.6 Admin & Manager

This section lists the modules made available for both user roles: admin and manager, which is the view reports module that allow them to view the daily reports and the cycle counting summary reports.

5.4.6.1 View Reports

Figure 5.61 shows the list of daily reports on the view report page. When the user clicks the “View” button, the selected daily report details will be displayed as shown in Figure 5.62. Figure 5.63 and Figure 5.64, on the other hand, show the list of cycle counting summary reports and the selected summary report details respectively.

5.4.6.1.1 View reports- daily reports

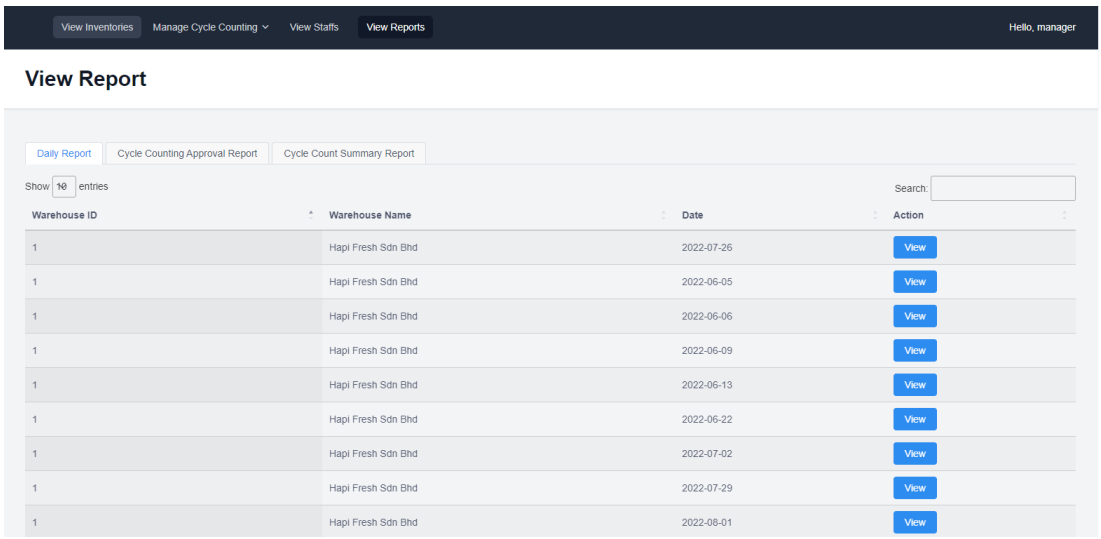

Figure 5.61: User interface design for viewing daily reports

5.4.6.1.2 View daily report details

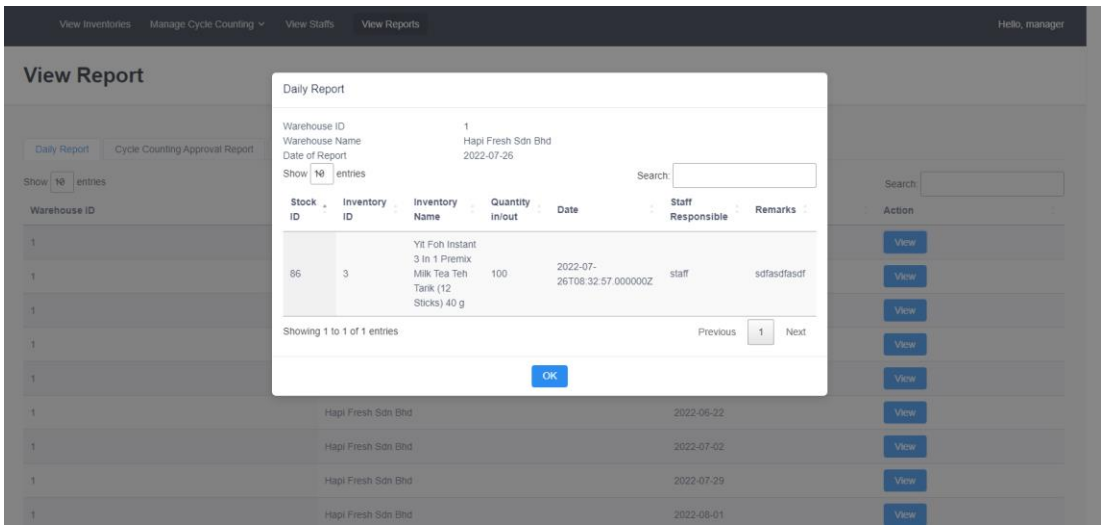

Figure 5.62: User interface design for viewing daily report details

5.4.6.1.3 View reports- cycle count summary reports

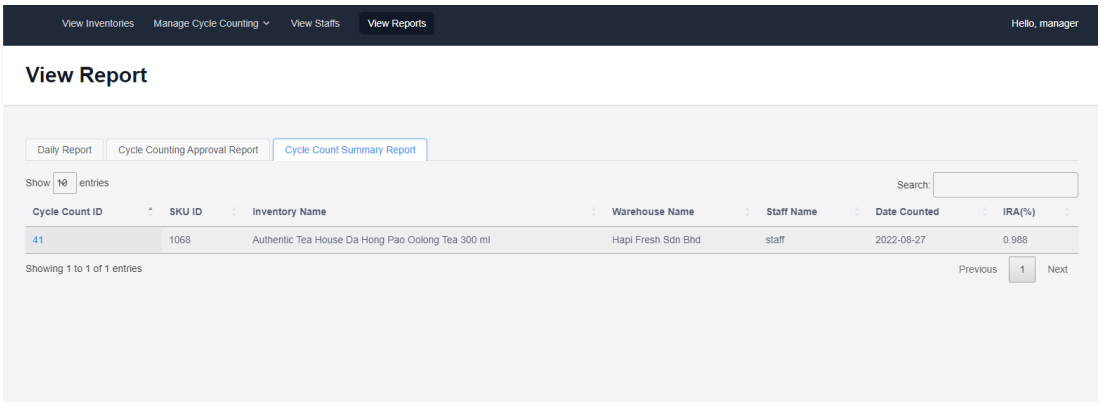

Figure 5.63: User interface design for viewing cycle count summary reports

5.4.6.1.4 View cycle count summary report details

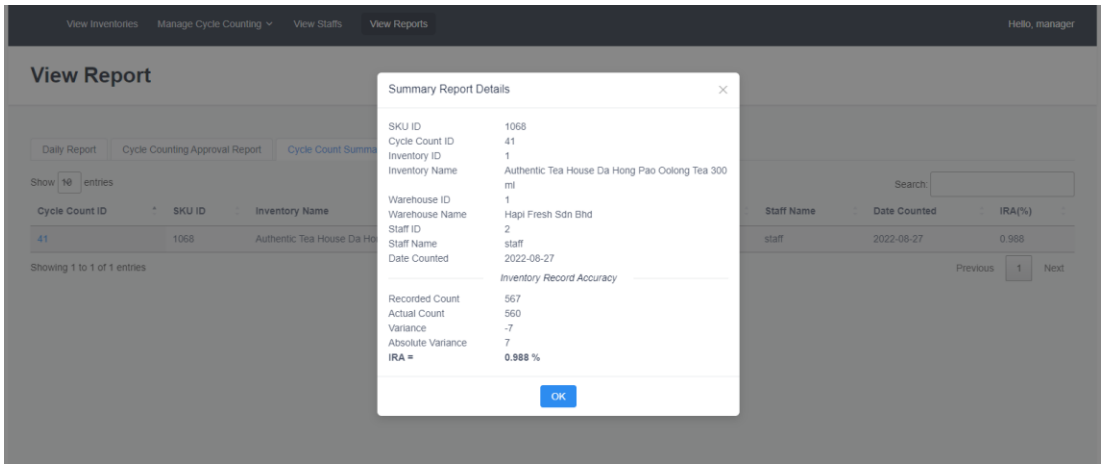

Figure 5.64: User interface design for viewing cycle count summary report details

5.5 Summary

In short, this chapter discussed the system architecture diagram, which detailed out the communication between the frontend and the backend as well as the role of Pusher in the system architecture on the real-time feature by broadcasting API messages to all clients once an event is created from the backend. Modelling diagrams that include data flow diagrams and interface flow diagrams, as well as the screenshots of implemented system's user interface design are also outlined and arranged according to the user roles for easier understanding purposes.

CHAPTER 6

SYSTEM IMPLEMENTATION

6.1 Introduction

This chapter documents the implementation details of the proposed system. The chapter divides into three parts. The first part is the backend part which includes the list of REST APIs used in the implemented system, the JWT authentication which secures the REST APIs and the Pusher Channel implemented for the real-time functionality. The second part covers all third-party user interface (UI) and JavaScript libraries used in frontend and the third part is the deployment of the implemented system to Heroku so that it can be accessed by user acceptance and usability testers later on.

6.2 Backend

This section will cover the backend setup of the implemented system, which will include the API List, JWT authentication to secure the REST APIs and the Pusher Channels setup for the real-time functionality.

6.2.1 API List

For the communication between the implemented system and the database, REST APIs are used extensively for every request. The implemented system will call an API to view, create, update or delete data from the database. All API endpoints used in the implemented system are listed in Table 6.1.

Table 6:1: List of API endpoints

Endpoints	Method	Description	Parameters required
/api/login?username={username}&password={password}	POST	To login into the system	1. Username 2. Password
/api/reset-password?new_password={new_password}&confirm_password={confirm_password}"	POST	To reset the password	1. New password 2. Confirm password
/api/logout	POST	To logout from the system	
/api/inventories	GET	To retrieve the list of inventories	
/api/inventory/{id}	GET	To retrieve an inventory's details	Inventory ID
/api/getStocksByInventory/{id}	GET	To retrieve all stocks of an inventory	Inventory ID
/api/inventories-unassigned-category	GET	To retrieve all inventories with no category	
/api/inventory?name={name}&warehouse_id={warehouse_id} &qty_on_hand={qty_on_hand} &cost_per_unit={cost_per_unit} &priority={priority}	POST	To add a new inventory	1. Name 2. Warehouse ID 3. Quantity on hand 4. Cost per unit 5. Priority
/api/inventory/{inventory_id}?name={name} &category_id={category_id} &cost_per_unit={cost_per_unit} &priority={priority}	PUT	To update an inventory	1. Inventory ID 2. Name 3. Category 4. Cost per Unit 5. Priority
/api/inventory/{id}	DELETE	To delete an inventory	1. Inventory ID
/api/warehouses	GET	To retrieve the list of warehouses	
/api/warehouse/{id}	GET	To retrieve a warehouse's details	1. Warehouse ID
/api/assign-category-to-bin/{id}	POST	To assign category to a warehouse's bin	1. Category ID 2. List of storage bin numbers to be assigned

/api/storage-bin-edit-inventory/{warehouse_id}?bin_id={bin_id} &inventory_id={inventory_id} &category_id={category_id}	POST	To edit the inventory in a storage bin	1. Warehouse ID 2. Storage bin ID 3. Category ID 4. Inventory ID
/api/warehouse	POST	To add a new warehouse	1. Warehouse name 2. Warehouse location 3. Storage bins
/api/warehouse/{id}?location={location}&manager_id={manager_id}	PUT	To update a warehouse	1. Warehouse ID 2. New warehouse location 3. New manager ID
/api/warehouse/{id}	DELETE	To delete a warehouse	1. Warehouse ID
/api/users	GET	To retrieve a list of users	
/api/active-staffs	GET	To retrieve only active staffs	
/api/user/{id}	GET	To retrieve a user detail	1. User ID
/api/user?name={name}&email={email}&contact_no={contact_no} &ic_no={ic_no} &role={role}&warehouse_id={warehouse_id} &employed_in={employed_in} &address={address} &username={username}	POST	To add a new user	1. User's name 2. User's email 3. User's Contact number 4. User's IC number 5. User's role 6. User's warehouse ID 7. User's employed date 8. User's address 9. User's username
/api/user/{id}?email={email} &contact_no={contact_no}&role={role}&warehouse_id={warehouse_id}&address={address}	PUT	To update a user	1. User ID 2. User's email 3. User's Contact number 4. User's role 5. User's warehouse ID

			6. User's address
/api/user/{id}	DELETE	To delete an existing user	
/api/categories	GET	To retrieve a list of categories	
/api/category?name={name}	POST	To add a new category	1. New category's name
/api/category/{category_id}?name={name}	PUT	To update a new category	1. Desired category's ID 2. New category name
/api/category/{category_id}	DELETE	To delete a category	1. Desired category's ID
/api/stocks	GET	To retrieve all inventory stocks	
/api/cycle-counts	GET	To retrieve all cycle count records	
/api/approve-cycle-count?cycle_counting_id={cycle_counting_id} &ira={ira}&inventory_id={inventory_id} &variance={variance}	POST	To approve a cycle count	1. Cycle counting ID 2. IRA 3. Inventory ID 4. Variance
/api/reject-cycle-count?cycle_counting_id={cycle_counting_id} &recount={recount} &schedule_date={schedule_date}	POST	To reject a cycle count	1. Cycle counting ID report 2. Recount 3. Schedule date
/api/storeCycleCountingSettings/{warehouseId}?working_day_start={working_day_start} &working_day_end={working_day_end} &cycle_count_class={cycle_count_class} &warehouse_id={warehouse_id} &staff_ids={staff_ids} &inventory_ids={inventory_ids} &start_end_date={start_end_date}	PUT	To update the cycle counting settings of a warehouse	1. Warehouse ID 2. Starting of working day 3. Ending of working day 4. Cycle count classes information 5. warehouse ID 6. staff ids 7. inventory ids 8. start and end date
/api/sku	POST	To store all SKUs into the database	Array of SKUs

/api/schedule	POST	To store all cycle count schedules into the database	Array of schedules
/api/skus	GET	To retrieve a list of SKUs	
/api/schedules	GET	To retrieve a list of cycle count schedules	
/api/reassignStaff	POST	To reassign all schedules allocated from one staff to another staff	1. Array of schedule ids 2. New staff ID
/api/cycle-count?schedule_id={schedule_id} &actual_count={actual_count} &recorded_count={recorded_count} &variance={variance}&inv_rec_accuracy={inv_rec_accuracy} &status={status}	POST	To create a new cycle count record	1. Cycle Count Schedule ID 2. Actual count 3. Recorded count 4. Variance 5. Inventory record accuracy calculated 6. Status
/api/stock?inventory_id={inventory_id} &mode={mode} &quantity={quantity}&remarks={remarks}	POST	To create a stock record (Check in/ check out)	1. Inventory ID 2. Check In/Out Mode 3. Quantity 4. Remarks

6.2.2 Secure the REST API

JWT Web tokens are an open way to represent users' identities during a two-party communication. For example, when two systems are exchanging data, the JWT token can be used to identify the user without having to send private data for every communication. For REST API context, the user can login to the system and once the user credentials are verified, the JWT Token will be created and returned to the web application (Doglio, 2021). Then, the web application can use the JWT Token on every request so that only authenticated users can access the APIs. In this way, the APIs can be secured while not having to send private data on every request to API. The overall mechanism of JWT is shown in Figure 6.1.

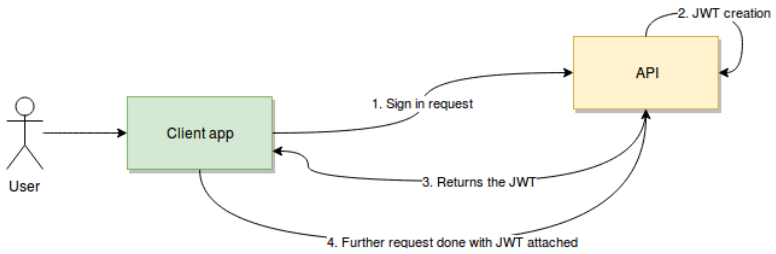

Figure 6.1: Overall mechanism of JWT

(Doglio, 2021)

6.2.3 Pusher Channel

Pusher Channels facilitates the communications between the server, app and devices. When an event happens on the app, the app will notify all other clients using the app about the changes. In this project, when a staff checks in or checks out an inventory stock, the Pusher Channel will broadcast the event to other staff that are using the system. This means that other staff can immediately see the changes without refreshing the page. Figure 6.2 shows the overview of the Pusher Channel dashboard.

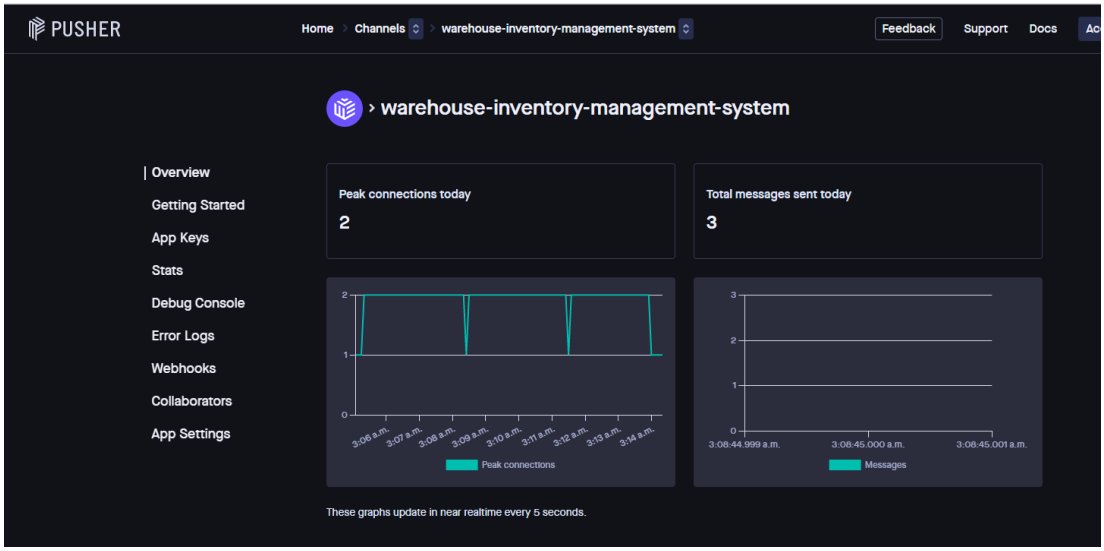

Figure 6.2: Overview of Pusher Channels dashboard

When two devices are using the system, both of them will be subscribed to the channel “check-in-out-stock.<warehouse_id>. The debug console in the Pusher Channel dashboard will show the events of the channel subscription in Figure 6.3. If one of the users is checking in or out stock, an event will be created which will be broadcasted by the pusher as an API message with the updated inventory object as shown in Figure 6.4 and an event will be created. In this case, this API message will be sent to both users that are subscribed to the channel.

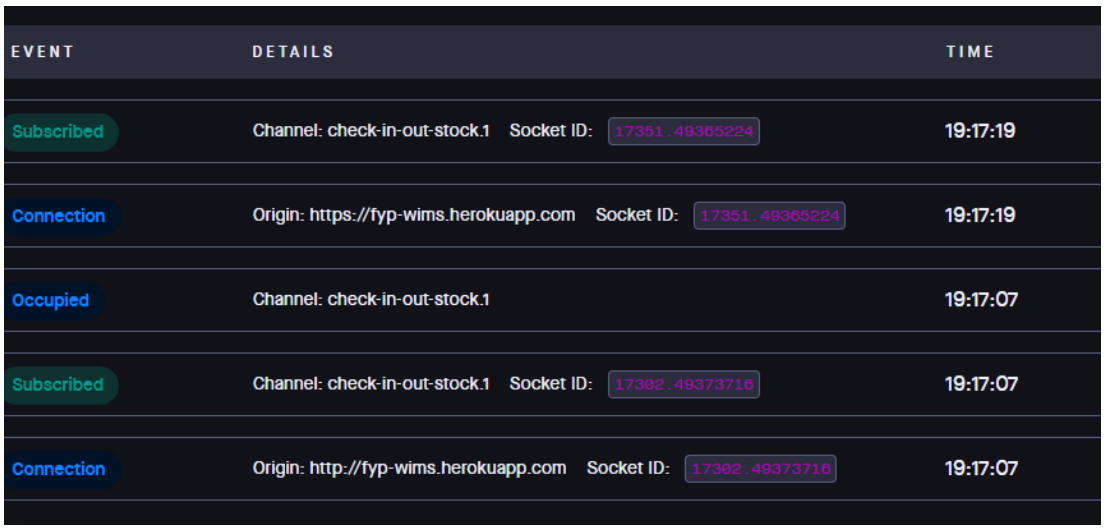

Figure 6.3: Events created for channel subscription

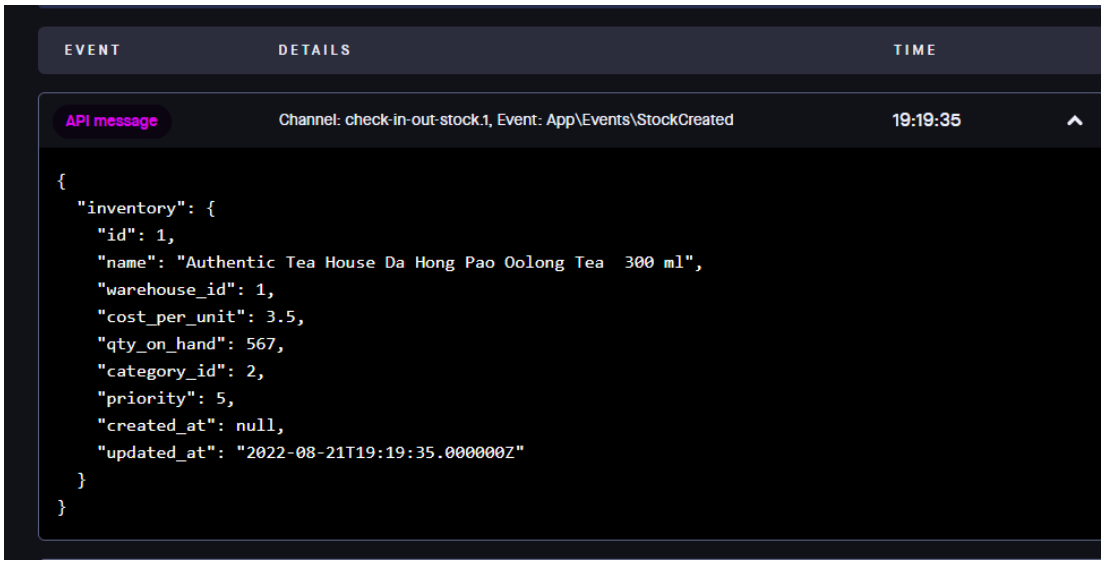

Figure 6.4: API message broadcasted to all subscribers of the channel.

6.3 Frontend

This section will include the user interface libraries and JavaScript libraries used in the frontend development of the implemented system.

6.3.1 UI Libraries

IView UI library is used in this project for most of the UI components. This library provides more than 80 base components that are mainly used for enterprise-level middle and background systems. In this project, all buttons, forms, inputs, modals, alerts, notices and tabs are integrated from this library. An example of a notice component utilized in the web application is shown in Figure 6.5. Besides, this library also allows the developer to customize the components to suit their needs. For example, some modals' footer in the web-based warehouse inventory system is customized so that only one button is displayed as shown in Figure 6.6.

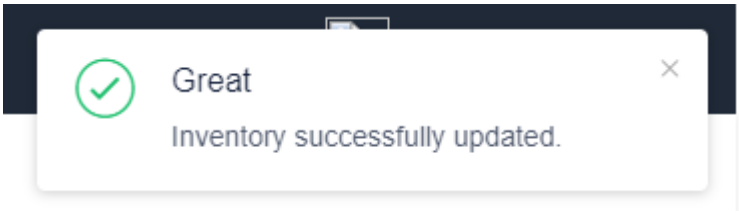

Figure 6.5: Success notice component

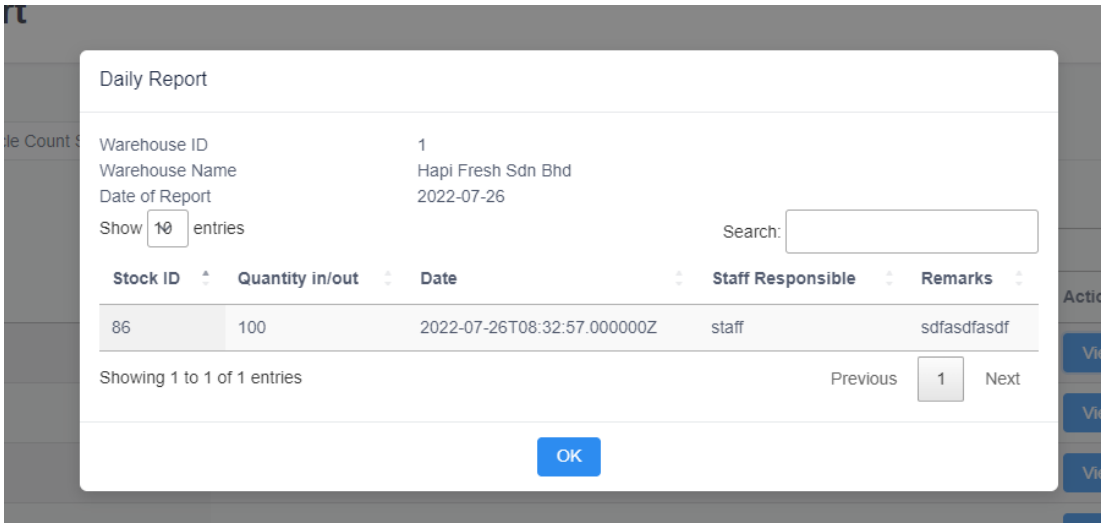

Figure 6.6: Modal with customized footer.

Another UI library used in this project is the TailwindUI. TailwindUI is also a UI library that provides beautifully designed components and templates. The login page and the top navigation bar components from the TailwindUI are used in this project as shown in Figure 6.7 and Figure 6.8. Besides, it also provides utilities that include layouts, flexbox and grid, spacing, sizing, typography and many more which are utilized in the project.

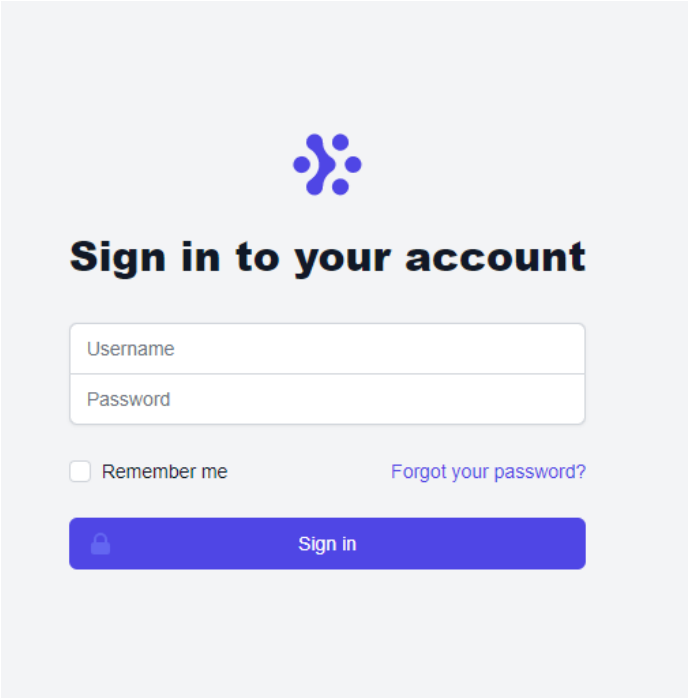

Figure 6.7: Login component

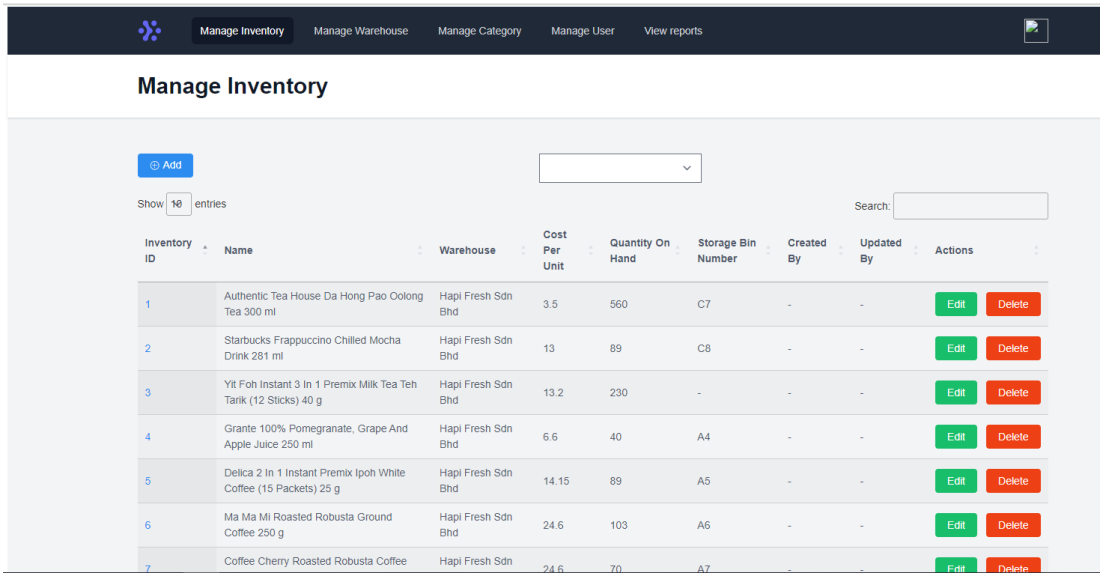

Figure 6.8: Navigation bar component

6.3.2 JS Libraries

In this web application, there are a lot of tables that are used to display data retrieved from the database. This is done by the DataTables, which is a plug-in for the jQuery JavaScript library. It is a highly flexible tool which adds advanced features to basic HTML tables. For example, DataTables allows pagination, instant search, multi-column ordering and many more just in several lines of code. There is also a more advanced feature provided by DataTables that is utilized in this project such as filtering the rows by column. This is used in filtering the list of inventories by warehouse and filtering the warehouse's storage bins by category as shown in Figure 6.9 and Figure 6.10.

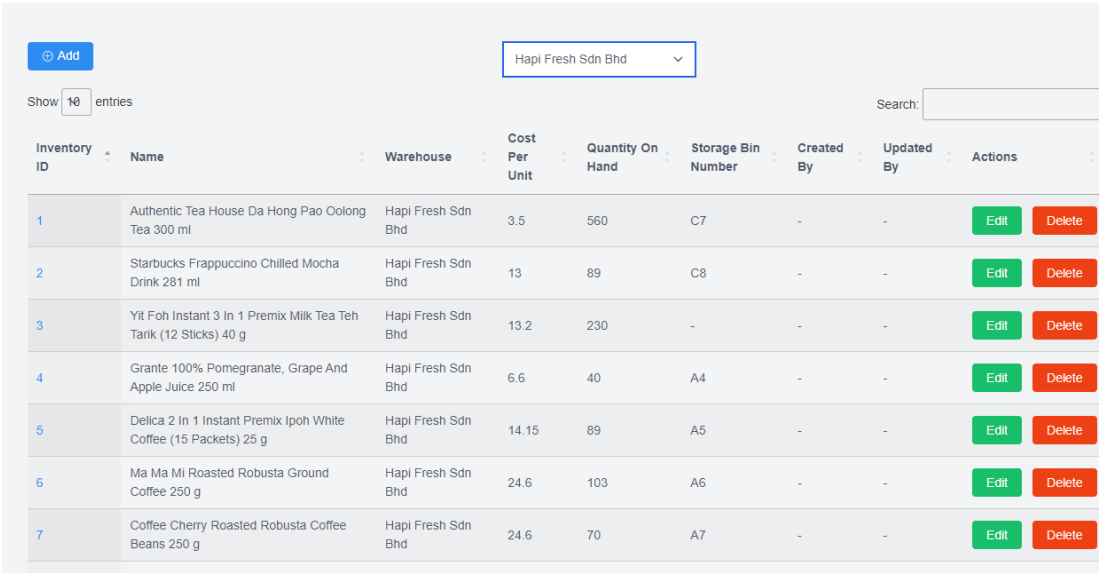

Figure 6.9: Filtering list of inventories by warehouse

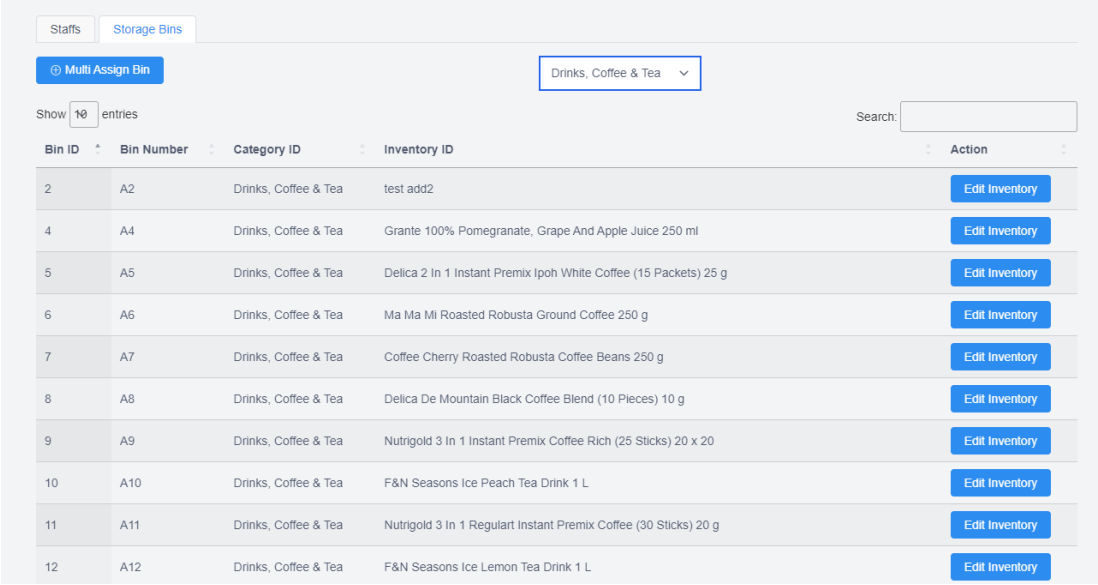

Figure 6.10: Filtering a warehouse's storage bins using category

Lodash, on the other hand, is used in this project for working with arrays, numbers, objects, strings, etc. It removes the hassle of dealing with them with plain JavaScript. For instance, there is a map function, which allows us to modify each element's value in an array with any value easily. Besides, the chain function can process any elements passed to it with several functions in one go. This can effectively reduce the lines of codes which can also improve the readability.

Moreover, momentJS is also used to parse, validate, manipulate and display the date in JavaScript. It is convenient to use momentJS to format the date from the database with UTC format to a fixed format or a more readable format before displaying out in the user interface. For instance, there is a global function created in the project, which uses momentJS to formate the date object before displaying it in the table cell.

6.4 Deployment to Heroku

To deploy the web application, Heroku is used to manage and automate the deployment. Heroku is a container-based cloud platform as a service (PaaS) owned by Salesforce that helps developers to deploy, manage and scale modern apps without being distracted by maintaining the servers, hardware or infrastructures. Heroku allows developers to create software up to a certain size and publish non-commercial apps such as proof of concepts, personal projects, etc for free.

Before deploying the developed web application to Heroku, the Git and Heroku CLI are installed beforehand and a Heroku account is created. Then, a new app can be created on Heroku and connected to the GitHub repository for the deployment method. By doing this, the changes made on GitHub will also automatically initiate Heroku to restart the server according to the necessary changes and republish it. The automatic deployment is also enabled so that every push to the master branch will be automatically deployed to the server. A file name Procfile is created to specify the public directory as the document root directory since this web application is built on the Laravel framework. Both PHP and node.js buildpack are added to compile both PHP and Vue.js.

Then, the ClearDB MySQL add-on provided by Heroku is used as the resource of the Heroku app to create the database instance. To connect with the MySQL database created, MySQL Workbench is installed. MySQL Workbench is a unified visual tool for data modelling, SQL development and an administration tool to configure servers, backup and many more. The connection can be added using MySQL Workbench to connect with the database instance created by using the connection parameters provided by the database instance. After the database is successfully connected, the complete database with tables and records done in PHPMyAdmin is exported out in SQL file format and imported into the database connected in MySQL Workbench. Lastly, the connection parameters and other necessary parameters such as app key, app URL and JWT secrets as shown in Figure 6.11 are added to the config vars of the Heroku app. Figure 6.12 shows the overview of the Heroku app with the ClearDB MySQL add-on installed. The deployed web application is shown in Figure 6.13.

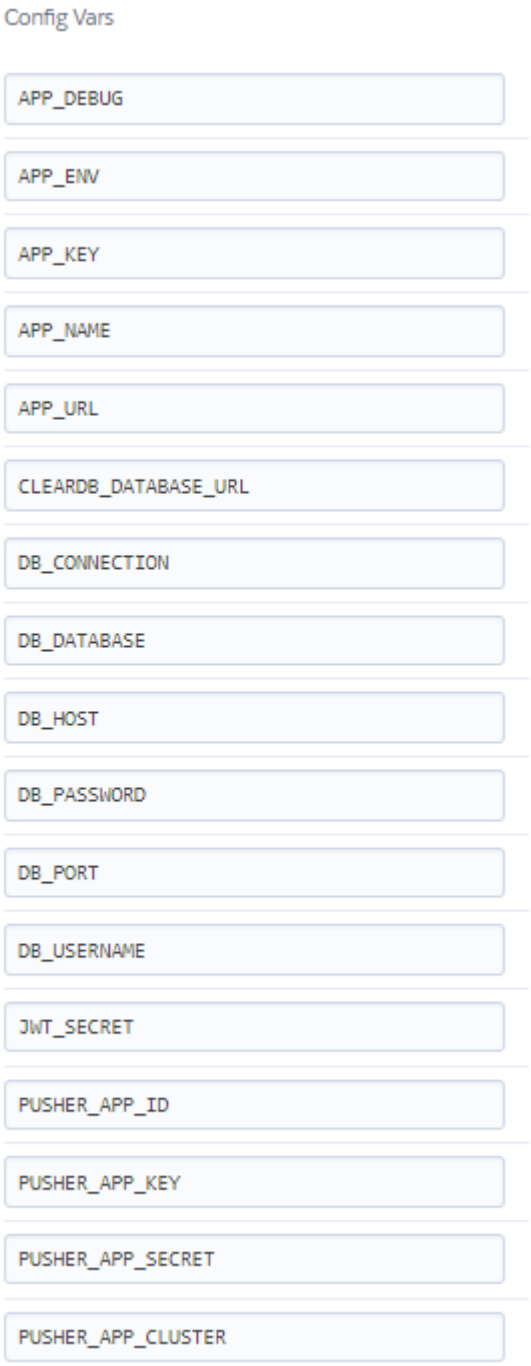

Figure 6.11: Config vars of the Heroku app

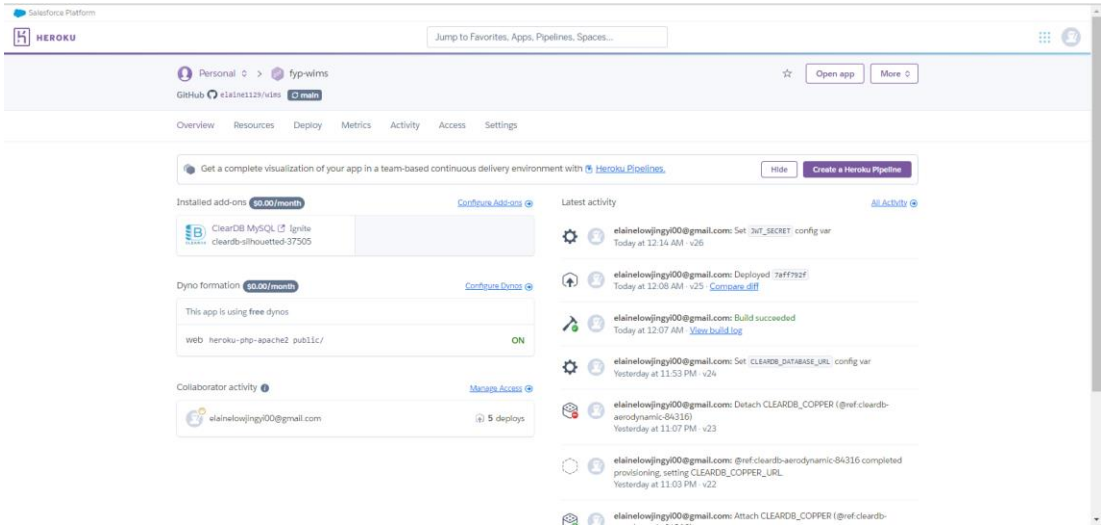

Figure 6.12: Overview of Heroku app

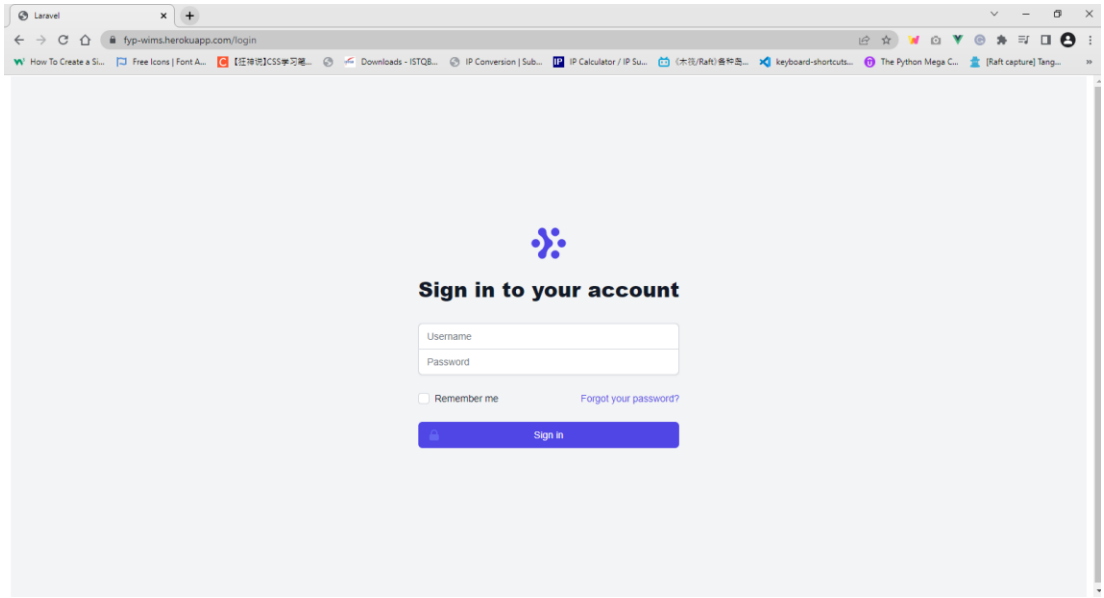

Figure 6.13: Deployed web-based inventory management system on Heroku

6.5 Conclusion

This chapter covers the implementation details for the intended system from the aspect of frontend and backend. For the frontend, all third-party libraries are covered whereas for the backend, the API list, JWT authentication and Pusher Channel are explained. Finally, after the development has been done, the implemented system is deployed to Heroku and the process of deployment is demonstrated in the last section.

CHAPTER 7

SYSTEM TESTING

7.1 Introduction

This chapter includes testing done after all development phases ended. The testing involved includes web service testing, real-time check-in/out module testing, user acceptance testing and usability testing. Besides, a traceability matrix is done between use cases, functional requirements, web service test cases and user acceptance test cases to validate that all use cases and functional requirements are tested during the system testing phase.

7.2 Web Service Testing

Web service testing is software testing that is used to validate the functionality, reliability security and performance of application program interface (APIs). It is comparable to unit testing as it tests specific codes instead of user interface objects. In this project, web service testing is performed with the help of Postman to validate the correctness of each API in returning the correct results when specified inputs are provided.

Table 7:1: Summary of web service testing's test cases with status

Test Case ID	Test Case Title	Status
WST001	Test Case of User Login	PASS
WST002	Test Case of Logging Out	PASS
WST003	Test Case of Viewing Inventory List	PASS
WST004	Test Case of Viewing Inventory Details	PASS
WST005	Test Case of Adding New Inventory with Category Assigned	PASS
WST006	Test Case of Adding New Inventory with No Category Assigned	PASS
WST007	Test Case of Updating an Inventory with Different Category	PASS
WST008	Test Case of Updating an Inventory with Same or No Category	PASS

WST009	Test Case of Deleting an Inventory	PASS
WST010	Test Case of Viewing Warehouse List	PASS
WST011	Test Case of Viewing Warehouse Details	PASS
WST012	Test Case of Multi-Assigning Storage Bins of a Warehouse	PASS
WST013	Test Case of Editing Inventory of a Warehouse Storage Bin	PASS
WST014	Test Case of Adding New Warehouse	PASS
WST015	Test Case of Updating a Warehouse	PASS
WST016	Test Case of Deleting a Warehouse	PASS
WST017	Test Case of Viewing User List	PASS
WST018	Test Case of Viewing Only Active Staffs	PASS
WST019	Test Case of Viewing User Details	PASS
WST020	Test Case of Adding New User	PASS
WST021	Test Case of Updating a Staff	PASS
WST022	Test Case of Deleting a User	PASS
WST023	Test Case of Resetting Password	PASS
WST024	Test Case of Viewing Category List	PASS
WST025	Test Case of Adding New Category	PASS
WST026	Test Case of Updating a Category	PASS
WST027	Test Case of Deleting a Category	PASS
WST028	Test Case of Viewing Daily Report List	PASS
WST029	Test Case of Viewing Cycle Counting Approval Report List	PASS
WST030	Test Case of Approving Cycle Counting Approval Report	PASS
WST031	Test Case of Rejecting Cycle Counting Approval Report with Recounting	PASS
WST032	Test Case of Rejecting Cycle Counting Approval Report without Recounting	PASS
WST033	Test Case of Viewing Cycle Counting Summary Report List	PASS
WST034	Test Case of Starting Cycle Counting	PASS

WST035	Test Case of Viewing Cycle Counting	PASS
WST036	Test Case of Reassigning Staff for Cycle Count Schedules	PASS
WST037	Test Case of Viewing All Staffs	PASS
WST038	Test Case of Performing Cycle Counting	PASS
WST039	Test Case of Checking in Inventory Stock	PASS
WST040	Test Case of Checking out Inventory Stock	PASS

Table 7:2: Test case of user login

Test Case ID	WST001	Status	PASS
Test Case Title	Test Case of User Login		
Test Case Description	To verify the successful login status if the username and password provided by user exists in the database		
Endpoints Involved	“http://127.0.0.1:8000/api/login?username={username}&password={password}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user enters the username and password to the endpoint “http://127.0.0.1:8000/api/login?username={username}&password={password}” to login into the system.	1. Username 2. Password	JSON response with status code 200 which consists of the access token, token type, token expiry time and the logged in user object	The user logged in successfully.

Table 7:3: Test case of logging out

Test Case ID	WST002	Status	PASS
Test Case Title	Test Case of Logging Out		
Test Case Description	To verify if user can be logged out from the system		
Endpoints Involved	“http://127.0.0.1:8000/api/logout”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user log out from the system using endpoint “http://127.0.0.1:8000/api/logout”		JSON response with status code 200.	The user is logged out successfully.

Table 7:4: Test case of viewing inventory list

Test Case ID	WST003	Status	PASS
Test Case Title	Test Case of Viewing Inventory List		
Test Case Description	To verify if user can view the list of inventories		
Endpoints Involved	“http://127.0.0.1:8000/api/inventories”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves the list of inventories from endpoint “http://127.0.0.1:8000/api/inventories”		JSON response with an array of inventories and status code 200.	The list of inventories is retrieved successfully.

Table 7:5: Test case of viewing inventory details

Test Case ID	WST004	Status	PASS
Test Case Title	Test Case of Viewing Inventory Details		
Test Case Description	To verify if user can view the inventory details		
Endpoints Involved	“http://127.0.0.1:8000/api/inventory/{id}”, “http://127.0.0.1:8000/api/getStocksByInventory/{id}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves the inventory details from endpoint “http://127.0.0.1:8000/api/inventory/{id}”,”	1. Inventory ID	Inventory data object in JSON format with status code 200	The inventory data with all stocks related to it is retrieved successfully
2. The user retrieves all stock records of the inventory from the endpoint “http://127.0.0.1:8000/api/getStocksByInventory/{id}”	1. Inventory ID	JSON object with array of stocks of given inventory and status code 200	

Table 7:6: Test case of adding new inventory with category assigned

Test Case ID	WST005	Status	PASS
Test Case Title	Test Case of Adding New Inventory with Category Assigned		
Test Case Description	To verify if user can add a new inventory with a category assigned to it to the database		
Endpoints Involved	“http://127.0.0.1:8000/api/warehouses”, “http://127.0.0.1:8000/api/categories”, “http://127.0.0.1:8000/api/inventory?name={name}&warehouse_id={warehouse_id}&qty_on_hand={qty_on_hand}&cost_per_unit={cost_per_unit}&category_id={category_id}&priority={priority}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves all warehouses from the database using endpoint “http://127.0.0.1:8000/api/warehouses”		JSON object with array of warehouses and status code 200	The new inventory is created and added to the database.
2. The user retrieves all categories from the database using endpoint “http://127.0.0.1:8000/api/categories”		JSON object with array of categories and status code 200	
3. The user enters name, quantity on hand, cost per unit, priority and selects the warehouse and category from the retrieved list to endpoint http://127.0.0.1:8000/api/inventory?name={name}&warehouse_id={warehouse_id}&qty_on_hand={qty_on_hand}&cost_per_unit={cost_per_unit}&category_id={category_id}&priority={priority}	1. Name 2. Warehouse 3. Quantity on Hand 4. Cost per Unit 5. Category 6. Priority	JSON object with status code 200 and the storage bin object in which the new inventory is placed in	

Table 7:7: Test case of adding new inventory with no category assigned

Test Case ID	WST006	Status	PASS
Test Case Title	Test Case of Adding New Inventory with No Category Assigned		
Test Case Description	To verify if user can add a new inventory with no category assigned to it to the database		
Endpoints Involved	“http://127.0.0.1:8000/api/warehouses”, “http://127.0.0.1:8000/api/categories”, “http://127.0.0.1:8000/api/inventory?name={name}&warehouse_id={warehouse_id}&qty_on_hand={qty_on_hand}&cost_per_unit={cost_per_unit}&priority={priority}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves all warehouses from the database using endpoint “http://127.0.0.1:8000/api/warehouses”,		A JSON object with array of warehouses and status code 200	The new inventory is created and added to the database.
2. The user retrieves all categories from the database using endpoint “http://127.0.0.1:8000/api/categories”		A JSON object with array of categories and status code 200	
3. The user enters name, quantity on hand, cost per unit, priority and selects the warehouse from the retrieved list to endpoint “http://127.0.0.1:8000/api/inventory?name={name}&warehouse_id={warehouse_id}&qty_on_hand={qty_on_hand}&cost_per_unit={cost_per_unit}&priority={priority}”	1. Name 2. Warehouse 3. Quantity on Hand 4. Cost per Unit 5. Priority	A JSON object with status code 201 and the created inventory object	

Table 7:8: Test case of updating an inventory with different category

Test Case ID	WST007	Status	PASS
Test Case Title	Test Case of Updating an Inventory with Different Category		
Test Case Description	To verify if user can update an inventory with different category compared to the original data		
Endpoints Involved	“http://127.0.0.1:8000/api/inventory/{inventory_id}?name={name}&category_id={category_id}&cost_per_unit={cost_per_unit}&priority={priority}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user updates the name, category, cost per unit and priority of a selected inventory using endpoint “http://127.0.0.1:8000/api/inventory/{inventory_id}?name={name}&category_id={category_id}&cost_per_unit={cost_per_unit}&priority={priority}”	1. Inventory ID 2. Name 3. Category 4. Cost per Unit 5. Priority	A JSON object with storage bin data that is newly assigned to the inventory	The inventory has been updated with new information entered by user.

Table 7:9: Test case of updating an inventory with same or no category

Test Case ID	WST008	Status	PASS
Test Case Title	Test Case of Updating an Inventory with Same or No Category		
Test Case Description	To verify if user can update an inventory with the no category or same category as the original data.		
Endpoints Involved	“http://127.0.0.1:8000/api/inventory/{inventory_id}?name={name}&cost_per_unit={cost_per_unit}&priority={priority}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user updates the name, cost per unit and priority of a selected inventory using endpoint “http://127.0.0.1:8000/api/inventory/{inventory_id}?name={name}&cost_per_unit={cost_per_unit}&priority={priority}”	1. Inventory ID 2. Name 3. Cost per Unit 4. Priority	A JSON response with status code 200 and value 1	The inventory has been updated with new information entered by user.

Table 7:10: Test case of deleting an inventory

Test Case ID	WST009	Status	PASS
Test Case Title	Test Case of Deleting an Inventory		
Test Case Description	To verify if user can delete an inventory from the database		
Endpoints Involved	“http://127.0.0.1:8000/api/inventory/{id}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user deletes an inventory using endpoint “http://127.0.0.1:8000/api/inventory/{id}”.	1. Inventory ID	A JSON response with status code 200 and value 1	The inventory has been deleted from the database

Table 7:11: Test case of viewing warehouse list

Test Case ID	WST010	Status	PASS
Test Case Title	Test Case of Viewing Warehouse List		
Test Case Description	To verify if user can view the list of warehouses		
Endpoints Involved	“http://127.0.0.1:8000/api/warehouses”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves the warehouses from endpoint “http://127.0.0.1:8000/api/warehouses”.		A JSON response with array of warehouses and status code 200	The list of warehouses is retrieved successfully

Table 7:12: Test case of viewing warehouse details

Test Case ID	WST011	Status	PASS
Test Case Title	Test Case of Viewing Warehouse Details		
Test Case Description	To verify if user can view the warehouse details		
Endpoints Involved	“http://127.0.0.1:8000/api/warehouse/{id}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves the warehouse data from endpoint “http://127.0.0.1:8000/api/warehouse/{id}”	1. warehouse ID	A JSON object with status code 200 and the warehouse object	The warehouse data with list of storage bins and staffs is retrieved.

Table 7:13: Test case of multi-assigning storage bins of a warehouse

Test Case ID	WST012	Status	PASS
Test Case Title	Test Case of Multi-Assigning Storage Bins of a Warehouse		
Test Case Description	To verify if user can assign a category to multiple bins.		
Endpoints Involved	“http://127.0.0.1:8000/api/categories”, “http://127.0.0.1:8000/api/assign-category-to-bin/{id}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves the list of categories from endpoint “http://127.0.0.1:8000/api/categories”		A JSON object with array of categories and status code 200	The warehouse storage bins are updated and the category and storage bin data of those inventories that was assign to one the selected bins were set back to null
2. The user assigns multiple storage bins with one category using endpoint “http://127.0.0.1:8000/api/assign-category-to-bin/{id}”	1. Category ID 2. List of storage bin numbers to be assigned	A JSON object with the updated warehouse object and status code 200	

Table 7:14: Test case of editing inventory of a warehouse storage bin

Test Case ID	WST013	Status	PASS
Test Case Title	Test Case of Editing Inventory of a Warehouse Storage Bin		
Test Case Description	To verify if user can edit the inventory inside a warehouse's storage bin		
Endpoints Involved	"http://127.0.0.1:8000/api/inventories-unassigned-category", "http://127.0.0.1:8000/api/storage-bin-edit-inventory/{warehouse_id}?bin_id={bin_id}&inventory_id={inventory_id}&category_id={category_id}"		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves the list of inventories that are not assigned with a category from endpoint "http://127.0.0.1:8000/api/inventories-unassigned-category"		A JSON object with array of inventories that have no category and status code 200	The warehouse storage bin has been updated with the given inventory. The selected inventory is also updated with the assigned bin's category. If the inventory of the storage bin is cleared, then the category and storage bin information of the original inventory in the bin will also be removed.
2. The user edit the inventory in a selected storage bin of a warehouse using endpoint "http://127.0.0.1:8000/api/storage-bin-edit-inventory/{warehouse_id}?bin_id={bin_id}&inventory_id={inventory_id}&category_id={category_id}"	1. Warehouse ID 2. Storage bin ID 3. Category ID 4. Inventory ID	A JSON object with the updated warehouse object and status code 200	

Table 7:15: Test case of adding new warehouse

Test Case ID	WST014	Status	PASS
Test Case Title	Test Case of Adding New Warehouse		
Test Case Description	To verify if user can add a new warehouse		
Endpoints Involved	“http://127.0.0.1:8000/api/categories”, “http://127.0.0.1:8000/api/warehouse”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves a list of categories from endpoint “http://127.0.0.1:8000/api/categories”		A JSON object with array of categories and status code 200	The new warehouse is successfully created into the database.
2. The user sends information which includes name, location and storage bins for the new warehouse to endpoint “http://127.0.0.1:8000/api/warehouse”	1. New warehouse name 2. New warehouse location 3. Storage bins	A JSON object with the new warehouse object and status code 201	

Table 7:16: Test case of updating a warehouse

Test Case ID	WST015	Status	PASS
Test Case Title	Test Case of Updating a Warehouse		
Test Case Description	To verify if user can update an existing warehouse		
Endpoints Involved	“http://127.0.0.1:8000/api/warehouse/{id}?location={location}&manager_id={manager_id}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user updates location and warehouse manager of a warehouse using endpoint “http://127.0.0.1:8000/api/warehouse/{id}?location={location}&manager_id={manager_id}”	1. Warehouse ID 2. New warehouse location 3. New manager ID	A JSON object with status code 200 and value 1	The warehouse’s location is updated. Manager of the warehouse will only be updated if new manager is given.

Table 7:17: Test case of deleting a warehouse

Test Case ID	WST016	Status	PASS
Test Case Title	Test Case of Deleting a Warehouse		
Test Case Description	To verify if user can delete an existing warehouse		
Endpoints Involved	“http://127.0.0.1:8000/api/warehouse/{id}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user deletes a selected warehouse using endpoint “http://127.0.0.1:8000/api/warehouse/{id}”	1. Warehouse ID	A JSON object with status code 200 and value 1	The selected warehouse is deleted from the database. All staffs, inventories, cycle countings and stocks related to this warehouse are deleted.

Table 7:18: Test case of viewing user list

Test Case ID	WST017	Status	PASS
Test Case Title	Test Case of Viewing User List		
Test Case Description	To verify if user can view the list of users.		
Endpoints Involved	“http://127.0.0.1:8000/api/users”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves the list of users from endpoint “http://127.0.0.1:8000/api/users”		A JSON object with an array of users and status code 200	The list of users that includes both active and inactive users is retrieved

Table 7:19: Test case of viewing only active staffs

Test Case ID	WST018	Status	PASS
Test Case Title	Test Case of Viewing Only Active Staffs		
Test Case Description	To verify if user can view only active staffs. Inactive staffs, managers and admin will not be retrieved.		
Endpoints Involved	“http://127.0.0.1:8000/api/active-staffs”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves the list of active staffs from endpoint “http://127.0.0.1:8000/api/active-staffs”		A JSON object with an array of active staffs and status code 200	The list of active staffs is retrieved successfully

Table 7:20: Test case of viewing user details

Test Case ID	WST019	Status	PASS
Test Case Title	Test Case of Viewing User Details		
Test Case Description	To verify if user can view the selected user’s details.		
Endpoints Involved	“http://127.0.0.1:8000/api/user/{id}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves the user details of a selected user using endpoint “http://127.0.0.1:8000/api/user/{id}”	1. User ID	A JSON object with the retrieved user object and status code 200	The selected user object is successfully retrieved.

Table 7:21: Test case of adding new user

Test Case ID	WST020	Status	PASS
Test Case Title	Test Case of Adding New User		
Test Case Description	To verify if user can add a new user to a warehouse		
Endpoints Involved	"http://127.0.0.1:8000/api/warehouses", "http://127.0.0.1:8000/api/user?name={name}&email={email}&contact_no={contact_no}&ic_no={ic_no} &role={role}&warehouse_id={warehouse_id}&employed_in={employed_in}&address={address} &username={username}"		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves all warehouses from the endpoint "http://127.0.0.1:8000 /api/warehouses"		A JSON object with an array of warehouses and status code 200	A temporary password is created using the new user's contact number and the new user is successfully created into the database.
2. The user sends all information needed for a new warehouse to the endpoint "http://127.0.0.1:8000 /api/user?name={nam e}&email={email}&c ontact_no={contact_n o}&ic_no={ic_no}&r ole={role}&warehous e_id={warehouse_id} &employed_in={empl oyed_in}&address={a ddress} &username={usernam e}"	1. User's name 2. User's email 3. User's Contact number 4. User's IC number 5. User's role 6. User's warehouse ID 7. User's employed date 8. User's address 9. User's username	A JSON object with the newly created user object and status code 201	

Table 7:22: Test case of updating a staff

Test Case ID	WST021	Status	PASS
Test Case Title	Test Case of Updating a Staff		
Test Case Description	To verify if user can update the email, contact number, role, warehouse and address of an existing user		
Endpoints Involved	“http://127.0.0.1:8000/api/warehouses”, “http://127.0.0.1:8000/api/user/{id}?email={email} &contact_no={contact_no}&role={role}&warehouse_id={warehouse_id}&address={address}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves all warehouses from the endpoint “http://127.0.0.1:8000 /api/warehouses”		A JSON object with an array of warehouses and status code 200	The user is updated with the given information.
2. The user sends all information needed for a new warehouse to the endpoint “http://127.0.0.1:8000 /api/user/{id}?email= {email} &contact_no={contact _no}&role={role}&w arehouse_id={wareho use_id}&address={ad dress}”	1. User ID 2. User’s email 3. User’s Contact number 4. User’s role 5. User’s warehouse ID 6. User’s address	A JSON object with status code 200 and value 1	

Table 7:23: Test case of deleting a user

Test Case ID	WST022	Status	PASS
Test Case Title	Test Case of Deleting a User		
Test Case Description	To verify if user can delete an existing user		
Endpoints Involved	“http://127.0.0.1:8000/api/user/{id}”		
Test Steps	Test Data	Expected Result	Actual Result
1. User deletes a user using endpoint “http://127.0.0.1:8000/api/user/{id}”	1. User ID	An empty JSON object with status code 200	The selected user is set as inactive user

Table 7:24: Test case of resetting password

Test Case ID	WST023	Status	PASS
Test Case Title	Test Case of Resetting Password		
Test Case Description	To verify if user can reset the password on first time login		
Endpoints Involved	“http://127.0.0.1:8000/api/reset-password?new_password={new_password}&confirm_password={confirm_password}”		
Test Steps	Test Data	Expected Result	Actual Result
1. User sends the new password to the endpoint “http://127.0.0.1:8000/api/reset-password?new_password={new_password}&confirm_password={confirm_password}”	1. New password 2. Confirm password	A JSON object with the updated user object and status code 200	The logged in user’s password is updated.

Table 7:25: Test case of viewing category list

Test Case ID	WST024	Status	PASS
Test Case Title	Test Case of Viewing Category List		
Test Case Description	To verify if user can view the list of categories		
Endpoints Involved	“http://127.0.0.1:8000/api/categories”		
Test Steps	Test Data	Expected Result	Actual Result
1. User retrieves the category list from the endpoint “http://127.0.0.1:8000/api/categories”		An array of categories in the format of JSON and status code 200	The list of categories is retrieved successfully

Table 7:26: Test case of adding new category

Test Case ID	WST025	Status	PASS
Test Case Title	Test Case of Adding New Category		
Test Case Description	To verify if user can add a new category for all warehouses		
Endpoints Involved	“http://127.0.0.1:8000/api/category?name={name}”		
Test Steps	Test Data	Expected Result	Actual Result
1. User sends the new category name to the endpoint “http://127.0.0.1:8000/api/category?name={name}”	1. New category’s name	The newly created category object in the format of JSON and status code 201	The new category is created into the database successfully

Table 7:27: Test case of updating a category

Test Case ID	WST026	Status	PASS
Test Case Title	Test Case of Updating a Category		
Test Case Description	To verify if user can update an existing category's name		
Endpoints Involved	"http://127.0.0.1:8000/api/category/{category_id}?name={name}"		
Test Steps	Test Data	Expected Result	Actual Result
1. User sends the new category name to the endpoint "http://127.0.0.1:8000/api/category/{category_id}?name={name}"	1. Desired category's ID 2. New category name	A JSON object status code 200 and value 1	The selected category's name is updated.

Table 7:28: Test case of deleting a category

Test Case ID	WST027	Status	PASS
Test Case Title	Test Case of Deleting a Category		
Test Case Description	To verify if user can delete an existing category's name		
Endpoints Involved	"http://127.0.0.1:8000/api/category/{category_id}"		
Test Steps	Test Data	Expected Result	Actual Result
1. User deletes the selected category using the endpoint "http://127.0.0.1:8000/api/category/{category_id}"	1. Desired category's ID	A JSON object status code 200 and value 1	The selected category's is successfully deleted from the database

Table 7:29: Test case of viewing daily report list

Test Case ID	WST028	Status	PASS
Test Case Title	Test Case of Viewing Daily Report List		
Test Case Description	To verify if user can view the list of daily reports.		
Endpoints Involved	“http://127.0.0.1:8000/api/stocks”		
Test Steps	Test Data	Expected Result	Actual Result
1. User retrieved the daily reports from the endpoint “http://127.0.0.1:8000/api/stocks”		An array of inventory stock records in JSON format and status code 200	The list of daily reports is retrieved from the database successfully.

Table 7:30: Test case of viewing cycle counting approval report list

Test Case ID	WST029	Status	PASS
Test Case Title	Test Case of Viewing Cycle Counting Approval Report List		
Test Case Description	To verify if user can view the list of cycle counting approval reports		
Endpoints Involved	“http://127.0.0.1:8000/api/cycle-counts”		
Test Steps	Test Data	Expected Result	Actual Result
1. User retrieves the cycle count approval reports from the endpoint “http://127.0.0.1:8000/api/cycle-counts”		An array of cycle counting records in JSON format and status code 200	The list of cycle counting approval reports is retrieved from the database successfully.

Table 7:31: Test case of approving cycle counting approval report

Test Case ID	WST030	Status	PASS
Test Case Title	Test Case of Approving Cycle Counting Approval Report		
Test Case Description	To verify if user can approve one of the cycle-counting approval report from the list		
Endpoints Involved	“http://127.0.0.1:8000/api/approve-cycle-count?cycle_counting_id={cycle_counting_id}&ira={ira}&inventory_id={inventory_id}&variance={variance}”		
Test Steps	Test Data	Expected Result	Actual Result
1. User approves one of the cycle-counting approval reports from the list by sending the cycle counting id, IRA, inventory ID and variance to the endpoint “http://127.0.0.1:8000/api/approve-cycle-count?cycle_counting_id={cycle_counting_id}&ira={ira}&inventory_id={inventory_id}&variance={variance}”	1. Cycle counting ID of the cycle counting approval report 2. IRA calculated from variance 3. Inventory ID of the cycle counting approval report 4. Variance between the recorded count and the actual count	A JSON object with status code 200 and value 1	The selected cycle counting approval report is approved with the calculated IRA updated to the database. The status of the cycle counting changed from “PENDING” to “COMPLETED”. The recorded count of the involved inventory is modified with the variance.

Table 7:32: Test case of rejecting cycle counting approval report with recounting

Test Case ID	WST031	Status	PASS
Test Case Title	Test Case of Rejecting Cycle Counting Approval Report with Recounting		
Test Case Description	To verify if user can reject one of the cycle-counting approval report from the list. Schedule should be created for recounting.		
Endpoints Involved	“http://127.0.0.1:8000/api/reject-cycle-count?cycle_counting_id={cycle_counting_id}&recount={recount}&schedule_date={schedule_date}”		
Test Steps	Test Data	Expected Result	Actual Result
1. User rejects one of the cycle-counting approval reports from the list by sending the cycle counting id, recount and schedule date to the endpoint “http://127.0.0.1:8000/api/reject-cycle-count?cycle_counting_id={cycle_counting_id}&recount={recount}&schedule_date={schedule_date}”	1. Cycle counting ID of the cycle counting approval report 2. Recount (true or false) 3. Schedule date for recounting the cycle counting	A JSON object with the newly created schedule for recounting and status code 201.	The selected cycle counting approval report is rejected and a new schedule for recounting the sku is generated. The status of the cycle counting changed from “PENDING” to “REJECTED”.

Table 7:33: Test case of rejecting cycle counting approval report without recounting

Test Case ID	WST032	Status	PASS
Test Case Title	Test Case of Rejecting Cycle Counting Approval Report without Recounting		
Test Case Description	To verify if user can reject one of the cycle-counting approval report from the list with no recounting.		
Endpoints Involved	“http://127.0.0.1:8000/api/reject-cycle-count?cycle_counting_id={cycle_counting_id}&recount={recount}&schedule_date= {schedule_date}”		
Test Steps	Test Data	Expected Result	Actual Result
1. User rejects one of the cycle-counting approval reports from the list by sending the cycle counting id, recount and schedule date to the endpoint “http://127.0.0.1:8000/api/reject-cycle-count?cycle_counting_id={cycle_counting_id}&recount={recount}&schedule_date={schedule_date}”	1. Cycle counting ID of the cycle counting approval report 2. Recount (true or false) 3. Schedule date for recounting the cycle counting	A JSON object with status code 200 and value 1.	The selected cycle counting approval report is rejected. The status of the cycle counting changed from “PENDING” to “REJECTED”.

Table 7:34: Test case of viewing cycle counting summary report list

Test Case ID	WST033	Status	PASS
Test Case Title	Test Case of Viewing Cycle Counting Summary Report List		
Test Case Description	To verify if user can view the list of cycle counting summary reports		
Endpoints Involved	“http://127.0.0.1:8000/api/cycle-counts”		
Test Steps	Test Data	Expected Result	Actual Result
1. User retrieves the cycle count summary reports from the endpoint “http://127.0.0.1:8000/api/cycle-counts”		An array of cycle counting records in JSON format and status code 200	The list of cycle counting summary reports is retrieved from the database successfully.

Table 7:35: Test case of starting cycle counting

Test Case ID	WST034	Status	PASS
Test Case Title	Test Case of Starting Cycle Counting		
Test Case Description	To verify if user can start or restart the cycle counting by entering settings		
Endpoints Involved	“http://127.0.0.1:8000/api/active-staffs”, “http://127.0.0.1:8000/api/inventories”, “http://127.0.0.1:8000/api/storeCycleCountingSettings/{warehouseId}?working_day_start={working_day_start}&working_day_end={working_day_end}&cycle_count_class={cycle_count_class}&warehouse_id={warehouse_id}&staff_ids={staff_ids}&inventory_ids={inventory_ids}&start_end_date={start_end_date}”, “http://127.0.0.1:8000/api/sku”, “http://127.0.0.1:8000/api/schedule”		
Test Steps	Test Data	Expected Result	Actual Result
1. User retrieves the active staffs from the endpoint “http://127.0.0.1:8000/api/active-staffs”		A JSON object with an array of user objects that are active and status code 200	The list of active staffs of user’s warehouse is retrieved successfully.
2. User retrieves all inventories of his warehouse from the endpoint “http://127.0.0.1:8000/api/inventories”		A JSON object with an array of inventory objects in user’s warehouse and status code 200	The list of inventories of user’s warehouse is retrieved successfully.
3. User sends the workday start, workday end, cycle count classes, warehouse ID, staff ids, inventory ids, and start end date to the endpoint “http://127.0.0.1:8000	1. Warehouse’s ID 2. Starting of working day 3. Ending of working day 4. Cycle count classes information which includes class name (A, B or C), daily count for each class, frequency to count all skus for each class, number of skus per class, type	A warehouse object with the updated cycle counting settings and status code 200	User’s warehouse’s cycle counting settings is updated with the information given by user.

/api/storeCycleCountingSettings/{warehouseId}?working_day_start={working_day_start}&working_day_end={working_day_end}&cycle_count_class={cycle_count_class}&warehouse_id={warehouse_id}&staff_ids={staff_ids}&inventory_ids={inventory_ids}&start_end_date={start_end_date}",to save the settings.	of counting frequency (daily, weekly, monthly or yearly), and counting frequency per class (eg. 3 = every 3 days/weeks/months/year) 5. warehouse ID to save the settings to 6. staff ids to be assigned as cycle counting personnels 7. inventory ids to be assigned as sku for cycle counting 8. start and end date of the cycle counting period		
4. User sends the array of SKUs to the endpoint "http://127.0.0.1:8000/api/sku"	1. Array of SKUs which includes the inventory id and their corresponding cycle count class (A, B, or C)	A JSON object with status code 200 and value 1	The SKUs are inserted into the database successfully. All previous cycle counting's SKUs for user's warehouse are removed from the database.
5. User sends the array of cycle counting schedules to the endpoint "http://127.0.0.1:8000/api/schedule"	1. Array of schedules for cycle counting that includes the assigned staff ID, inventory ID and schedule date.	A JSON object with status code 200 and value 1	The schedules are inserted into the database successfully. All previous cycle counting's schedules for user's warehouse are removed from the database.

Table 7:36: Test case of viewing cycle counting

Test Case ID	WST035	Status	PASS
Test Case Title	Test Case of Viewing Cycle Counting		
Test Case Description	To verify if user can view current cycle counting of his or her warehouse. The user should be able to view the list of cycle counting schedules and the current cycle counting settings with the list of staffs involved and SKUs.		
Endpoints Involved	“http://127.0.0.1:8000/api/warehouse/{warehouseId}”, “http://127.0.0.1:8000/api/skus”, “http://127.0.0.1:8000/api/schedules”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves his warehouse’s details from endpoint “http://127.0.0.1:8000/api/warehouse/{warehouseId}”,	1. User’s warehouse ID	A JSON object with the retrieved warehouse object and status code 200	The warehouse details which include the cycle counting settings are retrieved from the database.
2. The user retrieves the SKUs for the cycle counting of his warehouse from endpoint “http://127.0.0.1:8000/api/skus”		A JSON object with an array of retrieved SKUs and status code 200	All SKUs included in user warehouse’s cycle counting are retrieved from the database.
3. The user retrieves all cycle countig schedules of his warehouse from endpoint “http://127.0.0.1:8000/api/schedules”		A JSON object with an array of retrieved cycle counting schedules and status code 200	All cycle counting schedules with status “OPEN” are retrieved successfully.

Table 7:37: Test case of reassigning staff for cycle count schedules

Test Case ID	WST036	Status	PASS
Test Case Title	Test Case of Reassigning Staff for Cycle Count Schedules		
Test Case Description	To verify if user can reassign all cycle counting schedules of a selected staff to a new staff.		
Endpoints Involved	“http://127.0.0.1:8000/api/active-staffs”, “http://127.0.0.1:8000/api/reassignStaff”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves all active staffs in his warehouse from endpoint “http://127.0.0.1:8000/api/active-staffs”		A JSON object with an array of retrieved active staffs and status code 200.	The list of active staffs of user’s warehouse is retrieved successfully.
2. The user sends the schedules and new staff to assigned to the endpoint “http://127.0.0.1:8000/api/reassignStaff”	1. Array of schedule ids to be assigned to new staff. 2. New staff ID	A JSON object with status code 200 and value 1	All schedules of the original assigned staff have been successfully assigned to the new staff.

Table 7:38: Test case of viewing all staffs

Test Case ID	WST037	Status	PASS
Test Case Title	Test Case of Viewing All Staffs		
Test Case Description	To verify if user can view all staffs in a warehouse.		
Endpoints Involved	“http://127.0.0.1:8000/api/warehouse/{warehouseId}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves the warehouse details from the endpoint “http://127.0.0.1:8000/api/warehouse/{warehouseId}”	1. Warehouse ID	A JSON object with the retrieved warehouse object and status code 200	The warehouse details that includes all staffs related to the warehouse are retrieved successfully.

Table 7:39: Test case of performing cycle counting

Test Case ID	WST038	Status	PASS
Test Case Title	Test Case of Performing Cycle Counting		
Test Case Description	To verify if user can perform cycle counting on one of the upcoming cycle-counting schedules		
Endpoints Involved	“http://127.0.0.1:8000/api/schedules”, “http://127.0.0.1:8000/api/cycle-count?schedule_id={schedule_id}&actual_count={actual_count}&recorded_count={recorded_count}&variance={variance}&inv_rec_accuracy={inv_rec_accuracy}&status={status}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves all upcoming cycle counting schedules from endpoint “http://127.0.0.1:8000/api/schedules”		A JSON object with an array of retrieved cycle counting schedules and status code 200	All cycle counting schedules assigned to the user are retrieved.
2. The user sends the cycle count information to the endpoint “http://127.0.0.1:8000/api/cycle-count?schedule_id={schedule_id}&actual_count={actual_count}&recorded_count={recorded_count}&variance={variance}&inv_rec_accuracy={inv_rec_accuracy}&status={status}”	1. Cycle Count Schedule ID 2. Actual count 3. Recorded count 4. Variance between actual and recorded count 5. Inventory record accuracy calculated 6. Status	The newly created cycle counting object and status 201.	A cycle counting data is created with the IRA calculated into the database. The counted schedule’s status changed from “OPEN” to “CLOSED”

Table 7:40: Test case of checking in inventory stock

Test Case ID	WST039	Status	PASS
Test Case Title	Test Case of Checking in Inventory Stock		
Test Case Description	To verify if user can check in the inventory stock		
Endpoints Involved	“http://127.0.0.1:8000/api/inventories”, “http://127.0.0.1:8000/api/stock?inventory_id={inventory_id}&mode={mode} &quantity={quantity}&remarks={remarks}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves all inventories of his warehouse from endpoint “http://127.0.0.1:8000 /api/inventories”		A JSON object with an array of retrieved inventories and status code 200	
2. The user sends the inventory id, check in/out mode, quantity and remarks to the endpoint “http://127.0.0.1:8000 /api/stock?inventory_id={inventory_id}&mode={mode} &quantity={quantity} &remarks={remarks} ”	1. Inventory ID 2. Check In/Out Mode – “check-in” 3. Quantity 4. Remarks	The newly created stock object and status 201.	The inventory stock is created into the database. The inventory quantity is updated with the check in quantity

Table 7:41: Test case of checking out inventory stock

Test Case ID	WST040	Status	PASS
Test Case Title	Test Case of Checking out Inventory Stock		
Test Case Description	To verify if user can check out the inventory stock		
Endpoints Involved	“http://127.0.0.1:8000/api/inventories”, “http://127.0.0.1:8000/api/stock?inventory_id={inventory_id}&mode={mode} &quantity={quantity}&remarks={remarks}”		
Test Steps	Test Data	Expected Result	Actual Result
1. The user retrieves all inventories of his warehouse from endpoint “http://127.0.0.1:8000/api/inventories”		A JSON object with an array of retrieved inventories and status code 200	
2. The user sends the inventory id, check in/out mode, quantity and remarks to the endpoint “http://127.0.0.1:8000/api/stock?inventory_id={inventory_id}&mode={mode} &quantity={quantity} &remarks={remarks} ”	1. Inventory ID 2. Check In/Out Mode – “checkout” 3. Quantity 4. Remarks	The newly created stock object and status 201.	The inventory stock is created into the database. The inventory quantity is updated with the checkout quantity

7.3 Realtime Check In/Out Module Testing

This testing is done manually by utilizing the Console time() function from JavaScript to keep track of the time differences between creating new stock and receiving the API message broadcasted by Pusher. To ensure accuracy, 100 stocks will be created manually and the time differences are recorded in Table 7.42 and converted into graphical representation for easier analysis as shown in Figure 7.1.

Table 7:42: Testing iterations of delay time

Iteration No.	Time Difference (Delay Time) (ms)
1	839.364013671875
2	804.43798828125
3	1300.865966796875
4	1040.69482421875
5	826.22607421875
6	813.696044921875
7	833.98486328125
8	806.329833984375
9	829.08984375
10	835.852783203125
11	814.08203125
12	811.736083984375
13	956.326904296875
14	1076.707763671875
15	803.6591796875
16	819.56201171875
17	803.190185546875
18	797.010986328125
19	1417.396728515625
20	976.5830078125
21	1073.643798828125
22	825.501953125
23	828.118896484375
24	801.23486328125

25	891.909912109375
26	873.7431640625
27	824.130859375
28	810.4287109375
29	847.43310546875
30	856.81005859375
31	880.105224609375
32	1290.8330078125
33	852.7109375
34	834.806884765625
35	907.81591796875
36	1421.77783203125
37	872.830810546875
38	849.484130859375
39	842.723876953125
40	1359.113037109375
41	860.568115234375
42	885.2978515625
43	1392.31103515625
44	834.68505859375
45	1260.85595703125
46	826.39697265625
47	1379.18798828125
48	1038.342041015625
49	1028.7900390625
50	1225.70068359375
51	804.845947265625
52	1143.73095703125
53	1284.787109375
54	819.579833984375
55	878.468017578125
56	854.807861328125

57	808.14794921875
58	818.85498046875
59	983.770263671875
60	1056.886962890625
61	1383.447998046875
62	1034.9091796875
63	808.204833984375
64	859.890869140625
65	1287.89501953125
66	873.10986328125
67	805.27587890625
68	1076.5810546875
69	913.40478515625
70	817.7109375
71	907.080810546875
72	829.441650390625
73	1143.14111328125
74	846.197265625
75	802.955078125
76	1255.902099609375
77	826.59912109375
78	823.554931640625
79	913.60986328125
80	831.35498046875
81	842.8876953125
82	833.81494140625
83	805.175048828125
84	810.923095703125
85	816.511962890625
86	810.6708984375
87	1487.77197265625
88	897.81396484375

89	1099.4169921875
90	1356.26318359375
91	804.204833984375
92	816.962890625
93	1052.086181640625
94	1066.708251953125
95	795.73193359375
96	822.212158203125
97	1201.85791015625
98	812.48193359375
99	1317.908935546875
100	1062.9638671875

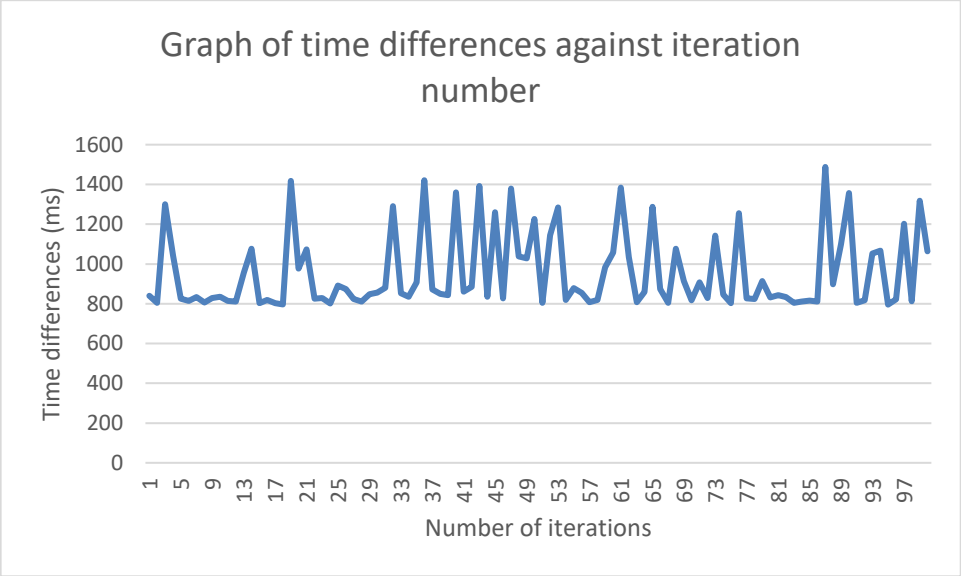

Figure 7.1: Graph of time differences against iteration number

From the graph in Figure 7.1, the maximum time difference is 1487.771973ms whereas the minimum time difference is 795.7319336ms. The average time difference calculated is 956.9260571ms, which is approximately equals to 1 second.

7.4 User Acceptance Testing and Usability Testing

User acceptance testing is carried out to determine whether the warehouse inventory management system can be accepted or not, whereas the usability testing is done to

identify any usability issues that can cause confusion or difficulties to the user while completing their task. There will be five targeted users in this section. Two of them are warehouse workers who are familiar with the business requirements in the warehouse (expert users) and another three of them are non-warehouse workers (novice users) to make sure that the system developed can also be adopted by users that are not familiar with the warehouse operations, which can prove the usability of the system. This testing is performed after web service testing is completed and all issues have been fixed. Both user acceptance testing and usability testing will be combined using the same set of test cases and will cover all important modules in this system, which adhere to the three project objectives mentioned above. All tests were done anonymously to protect the testers' privacy.

The testing will be done online, with a test form given to the user which includes all modules needed to be tested with detailed steps to perform the actions. They can access the deployed application with the URL *https://fyp-wims.herokuapp.com/* and the needed credentials were given to them for testing purposes. The developer kept the communication during the online session to a minimum level to give testers time to figure out their way to complete all of the test cases and observed the results obtained by the user for each test case. The developer was required to fill in the status and comments after the tester performed each test case. Status "PASS" indicates that the user has successfully performed the action with the observed result similar to the expected result whereas "FAIL" will be given if the user got a result other than the expected result or failed to complete the action due to any reason such as error occurred, confusions caused by the user interface, etc. The developer will be always on standby while the testers are performing the user acceptance testing to answer any question or provide a temporary solution to solve errors faced to ensure that the testing is done smoothly and all test cases are covered successfully.

After user acceptance testing has been done, the usability form will be given to the user to evaluate the usability of the system and a SUS score will be calculated afterwards for analysis.

7.4.1 User Acceptance Testing Result

The results of the user acceptance testing are attached in Appendix C. There is only one failed test case and several comments given by the testers that require attention as shown in Table 7.43.

Table 7:43: Summary of User Acceptance Testing Result

Test Case ID	Status	Error occurred/ Comments	Action Taken
TC013	Fail	Edited the phone number, the new phone number cannot be used as the password, have to use the phone number that is created upon adding the new user	After investigation and testing, it was found out that the tester edited the phone number of the created user. However, the default password still remained as the previous phone number. Therefore, the action taken was to add an if-condition when updating a user to make sure that the default password is updated accordingly as well.
TC001	Pass	'Opps' error message	After testing, it was found out that the error notification only show 'Opps' as the title without any proper message which makes tester to be confused. This is caused by the login module not implementing the handleApiError global methods written. Hence, the action taken to fix this was to add condition for unauthorized error in the global method and implement it in the module.
TC003	Pass	Cancel button and close button not working	After testing, the cancel button of the edit category modal does not close the modal. Hence, the action taken is to add the closing modal action to the cancel button.
TC020	Pass	No notification but got update	After testing, the approve cycle count approval action did not produce any success message which caused confusion to the tester. Hence, action to add a success notification is taken.

TC021	Pass	No notification but got update	After testing, the reject cycle count approval action did not produce any success message which caused confusion to the tester. Hence, action to add a success notification is taken.

7.4.2 Usability Testing Result

The System Usability Scale (SUS) was invented by John Brooke in 1986 as a 'quick and dirty' way to measure a system's usability. It considers and measures the combination of a system's efficiency, intuitiveness, ease of use and user satisfaction (Chinn, 2022). One notable study of finding usability problems in a system is conducted by Neilson and Launder (1993). With the assumption of the finding of usability problems are independent of whether they have been found before and independent of each other, Neilson and Launder found out that the detection of usability problems (as a function of the number of users tested or heuristic evaluator used) can be modelled as a *Poisson* process. Particularly, the number of usability problems found, denoted as $F(i, N, \lambda)$, in a usability test with i users can be described as:

$$F(i, N, \lambda) = N(1 - (1 - \lambda)^i), \qquad (7.1)$$

where N is the total number of usability problems in the system design and λ is the probability of discovering the average number of usability problems while testing with a single user. Substitute the typical value for λ =0.31 (Neilson and Launder, 1993), it is easy to verify that $F(i = 5, N, \lambda = 0.31) \approx 0.85N$, which means that more than 80% of the total number of usability problems can be discovered by only involving five users for testing. This is due to the fact that the more users we include in the test, the less information we can get (a usability issue), since we started to notice the same things over and over again.

A Google Form on the user satisfaction survey was given to the tester after they have performed the user acceptance testing as shown in Appendix D. The results were summarized in Table 7.44 with the SUS score calculated. The average SUS score is 85.5, which is considered excellent according to the scoring shown in Figure 7.2.

Table 7:44: SUS score collected

Tester	**Score for each item**										**Total**
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	
Tester #1	4	2	4	4	5	1	4	2	5	2	77.5
Tester #2	4	4	4	1	3	1	4	2	5	4	70
Tester #3	5	2	5	3	5	1	5	1	4	3	85
Tester #4	5	1	5	1	4	1	5	1	5	2	95
Tester #5	5	1	5	1	5	1	5	1	5	1	100
Average SUS Score											85.5

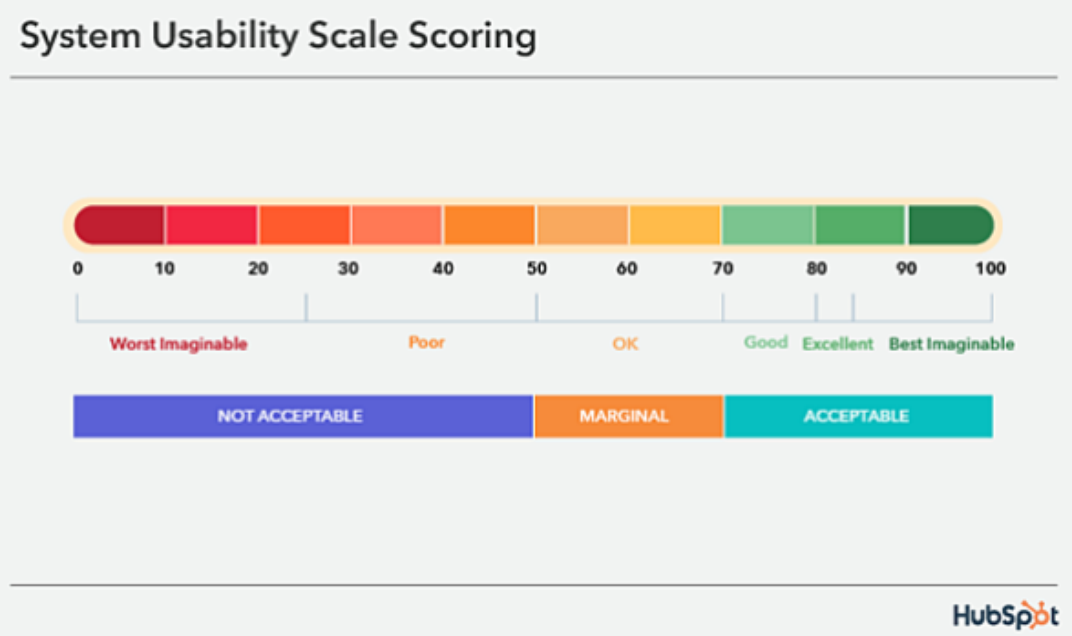

Figure 7.2: System Usability Scale Scoring

(Chinn, 2022)

Aside from the 10 questions asked to calculate the SUS score, there are also four open-ended questions were given to the testers to collect their opinions on the system. The responses were tabulated in Table 7.45. It could be summarized that the user likes the simple user interfaces of the system and what they like least about the system is that the table does not update after every action and requires them to refresh the page to show updates. The second part that they do not like about the system is that the data loading is very slow. Hence, these two parts that the testers mentioned will be taken into consideration for recommendations on future works.

Table 7:45: Responses on open-ended questions

Question	**Responses**
What do you like best about the system	simple and clean interface
	Simple but useful design
	easy to use, simple UI
What do you like least about the system	Table does not update, sometimes login tokens expired and does not redirect to login
	need wait for the data to load
	Some pages require refresh
	takes time to load data
	need keep refresh
If you were to describe this site to a colleague in a sentence or two, what would you say?	Easy to use and implement
	a web app that can replace most of the paper work in warehouse
Do you have any other final comments or questions?	very nice

7.5 Traceability between Use Cases, Functional Requirements and Test Cases

This section will show the traceability between the use cases, functional requirements, web service test cases and user acceptance test cases. The use cases and the functional requirements are tabulated with specific ID in Section 7.5.1 and Section 7.5.2 respectively. Lastly, the traceability matrix is shown in Section 7.5.3.

7.5.1 Use Cases Table

Table 7.46 lists the use cases with use case ID and name.

Table 7:46: Use Cases Table

Use Case ID	Use Case Name
UC001	Login account
UC002	Manage inventory
UC003	View all inventories
UC004	Manage warehouse
UC005	Manage user
UC006	Manage category
UC007	View reports
UC008	Manage cycle count approval report
UC009	Manage cycle counting
UC010	View all staff
UC011	Perform cycle counting
UC012	Record inventory

7.5.2 Functional Requirements Table

Table 7.47 shows the functional requirements table with functional requirements ID and functional requirements statements.

Table 7:47: Functional requirements table

Functional Requirement ID	Functional Requirement Statement
FR001	The system should allow admin to add, update and delete the inventory of a warehouse.
FR002	The system should allow admin to view, add, update, and delete a warehouse from the system.
FR003	The system should allow admin to view, add, update and delete a user in the system.
FR004	The system should allow admin to view, add, update and delete a category from the system
FR005	The system should allow admin to view daily reports of in-and-out inventories of all warehouses.
FR006	The system should allow admin to view the summary report for cycle counting of all warehouses.
FR007	The system should allow managers to start a new cycle counting process.
FR008	The system should allow managers to view all upcoming cycle counting SKUs.
FR009	The system should allow managers to view current cycle counting settings.
FR010	The system should allow a manager to view daily reports of in-and-out inventories of his or her warehouse only.
FR011	The system should allow managers to view summary reports for cycle counting of his or her warehouse only.

FR012	The system should allow managers to view, approve or reject cycle count pending approval reports.
FR013	The system should allow managers to view all staffs in the warehouse.
FR014	The system should allow staff to check in-and-out of the inventory stock.
FR015	The system should allow staff to view cycle count schedules that are assigned to them.
FR016	The system should allow staff to perform cycle counting on the SKU assigned.
FR017	The system should allow admin, manager and staff to login into their account.
FR018	The system should allow the admin, manager and staff to view all inventories in the system.
FR019	The system should calculate inventory record accuracy (IRA) of each SKU and generate summary reports for each cycle count input from staff.
FR020	The system should classify SKUs, calculate the number of SKUs to be counted daily for each class, generate schedules and assign to staff available evenly.

7.5.3 Traceability Matrix

Traceability follows the requirements' life if it is done well. It started from the time requirements originated and continued on to fulfil the requirement. Hence, traceability ensures that the requirements fulfil the intended goals. Besides, traceability simplifies decision-making by allowing project teams to understand how the software design is impacted by the requirements. For instance, if there is a change in requirements, the impact of the changes across development can be easily analysed.

The traceability matrix maps the use cases, functional requirements, web service test cases and UAT test cases using their ID as shown in Table 7.48 to make sure that the web service testing and the UAT testing covered all use cases and functional requirements. The use case ID was formalized in Table 7.46 under Section 7.5.1, whereas the functional requirement ID was formalized under Section 7.5.2, Table 7.47. Web service testing IDs and their title were described under Section 7.2, Table 7.1. The test cases of the user acceptance test were covered in Appendix B.

Table 7:48: Traceability matrix between use cases, functional requirements, web service testing and user acceptance testing

Use Case ID	Functional Requirement ID	Web Service Testing ID	User Acceptance Test Case ID
UC001	FR017	WST001, WST002, WST023	TC001, TC013
UC002	FR001	WST005, WST006, WST007, WST008, WST009	TC008, TC009, TC024
UC003	FR018	WST003, WST004	TC010
UC004	FR002	WST010, WST011, WST012, WST013, WST014, WST015, WST016	TC004, TC007, TC027, TC011, TC012
UC005	FR003	WST017, WST018, WST019, WST020, WST021, WST022	TC005, TC006, TC025

UC006	FR004	WST024, WST025, WST026, WST027	TC002, TC003, TC026
UC007	FR005, FR006, FR010, FR011	WST028, WST033	TC023, TC022
UC008	FR012, FR019	WST029, WST030, WST031, WST032,	TC020, TC021
UC009	FR007, FR008, FR009, FR020	WST034, WST035, WST036	TC014, TC015
UC010	FR013	WST037	TC016
UC011	FR015, FR016	WST038	TC017
UC012	FR014	WST039, WST040	TC018, TC019

CHAPTER 8

CONCLUSIONS AND RECOMMENDATIONS

8.1 Introduction

From the planning phase to the closing phase, this project took approximately 7 months to complete. This project aims to develop a web-based warehouse inventory management system for stock item tracking. In the planning and analysis phase, problems were researched and formulated to come out with the project objectives, proposed solution, project approach and scope. Literature reviews were also done which researched similar projects, different cycle counting method and their application and lastly the software development methodology. A questionnaire was also distributed to warehouse users to gather requirements and develop functional and non-functional requirements. To plan out the schedule and keep track of the project progress, a web breakdown structure (WBS) and Gantt chart were used. In the design phase, several diagrams were drawn to illustrate the design of the system. Involve diagrams were use case diagram, use case description, ERD diagram, data flow diagram and interface flow diagram.

Moving to the development phase, the whole system was broken down into three phases, following the phased-development methodology. The first phase focuses on the setting up of frameworks, preparing data, authentication of users, and the development of the first objective, which is the real-time check-in/out module. The second phase focused on the auto-scheduler for the cycle counting and the third phase covered the rest of the modules, which includes the report management, and all CRUDs. During each development phase, testing will be done to ensure that there were no errors before proceeding to the next phase.

For the closing phase, web service testing, user acceptance testing, usability testing, and testing for the real-time check-in/out module were done. An average SUS score of 85.5 was calculated, which is considered excellent in terms of the usability of the system. For the user acceptance testing, most test cases passed except one of the test cases (TC013), which was fixed after the testing. There were some comments received on passed test cases that were also fixed according to their feedback.

Responses from both user acceptance testing and usability testing will be considered for future works.

In later sections of this chapter, the objectives achievement, project limitations and recommendations for future works will be covered.

8.2 Objectives Achievement

In general, all objectives were achieved and delivered successfully and the details will be explained in this section.

8.2.1 Objective 1

Objective 1 aimed to develop an application to record all in-and-out of the warehouse inventories from different locations in real-time. This is achieved as the check-in/out module is developed in the implemented web-based warehouse inventory management system. The real-time functionality was also implemented using the Pusher Channels, as described in Chapter 6.2.3. This was tested in the Chapter 7.3 in which the check-in/out action was performed 100 times and the time difference between the time creating a new inventory stock and receiving the API message from Pusher Channels was recorded and tabulated in Table 7.42. The average time difference is around 1 second, which achieves the objective 1 with a delay time of less than 2 seconds.

8.2.2 Objective 2

Objective 2 is to develop an automated scheduler that can categorise SKUs, create schedules, and allocate them to cycle counting. The automated scheduler was successfully implemented in the system which can automatically classify the inventories selected for cycle counting to their respective cycle count class by calculating their stock value, create schedules according to workdays given by the manager and assign them to the allocated personnel evenly once the manager starts a new cycle counting process.

8.2.3 Objective 3

Objective 3 aims to propose and implement an improvement over existing SKU classification to achieve a higher degree of completeness and efficiency. Objective 3 was achieved by implementing the improved classification method by replacing the V,

E, and D classes from the ABC-VED method with priorities 1, 2, 3, 4, 5 and so on. As a result, the classification now is more complete as it is not only limited to the nine subclasses, instead, it can be more numbers of subclasses based on the maximum priority allowed in the system.

Aside from improvement for a more complete SKU classification from Objective 3, the efficiency was also ensured. The conventional ABC-VED classification method is to classify the selected inventories using the ABC classification method and VED classification method separately, which means that each inventory will hold two alphabets, one is A or B or C and another is V or E or D. Combining the two classes of each inventory will form the 9 subclasses AV, AE, AD, BV, BE, BD, CV, CE, and CE. Finally, they will be further classified into three classes I, II, and III as shown in Chapter 2 Table 2.3. To implement the conventional ABC-VED, this can be done by simply hardcoding the if-else statement for the classification of subclasses to the group I, II and III as shown in Figure 8.1. However, the efficiency of this algorithm is very dependent on the number of subclasses. If it was being extended in which there will be more classes instead of only V, E and D classes for the criticality of the inventories, then, there will be more subclasses to be classified into the final three classes (I, II and III) for cycle counting. For instance, if there are 1000 subclasses, then it is not possible to hardcode the if-else statement.

```
if (subclass=="AV"||subclass=="AE"||subclass=="AD"||subclass=="BV"||subclass=="CV"){
sku.class = "I"
}else if (subclass=="BE"||subclass=="BD"||subclass=="CE"){
sku.class = "II"
}else{
sku.class = "III"
}
```

Figure 8.1: Coding implementation of ABC-VED

8.2.3.1 SKU Classification Complexity

We will examine the complexity of these SKU classifications by their algorithm in terms of pseudo-code in order to have a more in-depth discussion.

8.2.3.1.1 ABC and ABC-VED Classification

Table 8:1: Pseudocode of ABC and ABC-VED algorithm

ABC approach: Input: N number of SKUs, thresholds $t_A = 0.2, t_C = 0.5$	**ABC-VED approach:** Input: N number of SKUs with quantity in hand, criticality classes {V, E, D}, $t_A = 0.2, t_C = 0.5$
1. **For each** SKU **in** SKUs: Compute SKU stock value; $SV =$ unit cost of SKU $\times$ Quantity in hand; Output all computed SKU stock value as $[SV_1, SV_2, \dots, SV_N]$; 2. Let Max stock value = max($[SV_1, SV_2, \dots, SV_N]$); 3. Let Min stock value = min($[SV_1, SV_2, \dots, SV_N]$); 4. **For each** SV **in** $[SV_1, SV_2, \dots, SV_N]$: Compute the transform value $TV = \frac{SV - \text{Min stock value}}{\text{Max stock value} - \text{Min stock value}}$, where $TV \in [0,1]$; **If** $TV \leq t_A$: categorize TV in class A **Elseif** $TV \geq t_C$: categorize TV in class C **Else**: categorize TV in class B Record the class where TV is categorized in	1. **For each** SKU **in** SKUs: Compute SKU stock value; $SV =$ unit cost of SKU $\times$ Quantity in hand; Set all computed SKU stock values as $[SV_1, SV_2, \dots, SV_N]$; 2. Let Max stock value = max($[SV_1, SV_2, \dots, SV_N]$); 3. Let Min stock value = min($[SV_1, SV_2, \dots, SV_N]$); 4. **For each** SV **in** $[SV_1, SV_2, \dots, SV_N]$: Compute the transform value $TV = \frac{SV - \text{Min stock value}}{\text{Max stock value} - \text{Min stock value}}$, where $TV \in [0,1]$; **If** $TV \leq t_A$: categorize TV in class A **Elseif** $TV \geq t_C$: categorize TV in class C **Else**: categorize TV in class B Record the class where TV is categorized in

5. Let Z be the all the classes for each TV is categorized in.	5. Let X∈{A, B, C} be the class where *TV* is categorized in STEP 4, and Y∈{V,E,D} ; 6. Categorize *TV* in XY sub-class based on the item criticality; 7. Compute all possible pair-wise combination (sub-class) of X and Y, i.e., output all possible sub-classes as S = [AV, AE, AD, BV, BE, BD, CV, CE, CD] 8. Using S to construct the reference classes C1= $\{AV, AE, AD, BV, CV\}$, C2=$\{BE, BD, CE\}$, C3={CD} 9. **For each** sub-class **in** S: **If** sub-class =XY∈ C1, categorize TV in group I **Elseif** sub-class =XY∈ C2, categorize TV in group II **Else** categorize TV in group III Record the group where TV is categorized in 10. Let Z be the all the groups for each TV is categorized in.
Output: Z	Output: Z

By looking at the comparison of pseudocode, the ABC-VED algorithms involved additional steps to classify all SKUs into their respective cycle count class. For the ABC algorithm, there are two for-loops (step-1 and step-4) involved, hence, the complexity of the ABC algorithm is $O(N) + O(N)$, which results to $O(N)$. On the other hand, the ABC-VED algorithm has additional steps to compute the cycle count class of the SKUs as compared to the ABC algorithm. In step-7 of the ABC-VED pseudocode, it will loop for 9 times following the number of subclasses (combination of ABC classes with VED classes). Since the number of subclasses is fixed in the conventional ABC-VED algorithm, the complexity of this step is $O(1)$. Therefore, the complexity of this algorithm is $O(N)(\text{step1}) + O(N)(\text{step4}) + O(1)(\text{step7}) + O(N)(\text{step9})$, which results to $O(N)$ as the final complexity. In this case, the complexity of both the ABC and ABC-VED algorithm is the same.

This conventional classification limits the criticality of the inventories, in which the inventories can only be either vital(V), essential(E), or desirable(D). Therefore, the completeness of this algorithm is improved in Objective 3 by allowing the criticality classes to have more than three classes. However, if the number of criticality classes is not fixed, the complexity in step-7 of the ABC-VED pseudocode will be $O(M)$, with M as the number of criticality class. This is because the complexity now will be $O(3M)$ as the for-loop will now loop for $3 \times M$ times to get all combinations of subclasses and the complexity of the algorithm is increased to be $O(N + M)$. This means that the complexity depends on N (number of SKUs) and M (number of criticality classes).

8.2.3.1.2 Proposed Classification Method

Table 8:2: Pseudocode of proposed classification algorithm

Proposed approach

Input: N number of SKUs with quantity in hand, criticality classes $\{1,2, \dots M\}$, $t_A = 0.2, t_C = 0.5$

1. **For each** SKU **in** SKUs:

   Based on the SKU criticality, assign a value of $P \in \{1,2, \dots \mathrm{M}\}$ to it, i.e, higher value of P means higher criticality;

   Compute new SKU stock value $SV' = \text{unit cost of SKU} \times \text{Quantity in hand} \times P$;

   Set all computed SKU stock values as $[SV'_1, SV'_2, \dots, SV'_N]$;

2. Let Max stock value = max($[SV'_1, SV'_2, \dots, SV'_N]$);

3. Let Min stock value = min($[SV'_1, SV'_2, \dots, SV'_N]$);

4. **For each** SV' **in** $[SV'_1, SV'_2, \dots, SV'_N]$:

   Compute the new transform value $TV' = \frac{SV' - \text{Min stock value}}{\text{Max stock value} - \text{Min stock value}}$, where $TV' \in [0,1]$;

   If $TV' \leq t_A$: categorize TV' in class A

   Elseif $TV' \geq t_C$: categorize TV' in class C

   Else: categorize TV' in class B

   Record the class where TV is categorized in

5. Let Z be the all the classes where each TV is categorized in.

Output: Z

In the proposed classification method, the criticality class is no longer limited to only 3 classes (V, E and D) but now is numerically represented (priority: from 1 to M). By doing this, the priority can be combined with the calculation of stock value in the ABC classification method. This direct computation reduced the complexity by eliminating the necessity of computing all possible pair-wise combinations of ABC classes and VED classes into the final cycle count class (step-6 to step-9 in ABC-VED

pseudocode). With appropriated threshold t_A and t_C, the categorization of the transform stock value simply follows the ABC categorization method, hence the proposed improvement method enjoys both ABC and VED SKU classification benefits with an extension of a higher degree of criticality classes, up to arbitrary M criticality classes, and preserved the system efficiency with complexity order of $O(N)$.

8.3 Project Limitations

There are some limitations noticed while using the system. The first limitation is that the loading time for displaying the rows of the table is usually 2 to 3 seconds. This is because the API endpoints needed some time for fetching the data from the database, especially when the data is large.

The second limitation is the storage bins creation. This is because the categories are shared among warehouses, hence, the storage bins will be created evenly for each of the categories in the database. However, this is not realistic as some warehouses may not store inventory for certain categories. Besides, warehouses might not have the same number of bins for each zone as well.

The third limitation is that the application will log the user out after 1 hour. This will cause the user to keep logging into the system after the login token expires, which is very inconvenient in day-to-day operations.

The fourth limitation is that only the check-in/out module is updated in real-time. The implemented system only creates events for check-in/out action that will broadcast the API message to other ends. This is because the check-in/out module is very crucial to keep track of the inventory stock accurately to prevent a situation like finding out that the actual quantity on hand in the warehouse is actually lesser than the quantity on hand stated in the system due to the reason that some stocks have been checked out from the other end and the user did not refresh the page. However, other modules should also be updated in real-time to eliminate the hassle of keep on refreshing the page to get the updated information.

8.4 Recommendations for future work

This system should be continuously enhanced and improved so that it can be more stable and provide a better user experience in the future. Hence, this section proposes some recommendations for future work (but not limited to) in Table 8.3.

Table 8:3: Recommendations for future work

Recommendations	**Description**
Enhance the flexibility of storage bins creation while creating new warehouse	The implemented system should be able to allow users to choose number of bins for each zone and assign category to each bins according to their preference. This gives convenience to warehouses that have inconsistent number of bins for each zone.
Add create storage bin feature	The implemented system should allow users to add extra storage bin after the warehouse has been created and has the flexibility to add it in a specified location in the list of storage bins. This feature is important as sometime warehouses may be expanded or renovated to have extra bins to be added to the system.
Add remember me token for authentication	The implemented system should allow user to keep on accessing the system if the user chose to remember his or her login while logging in. This provides better user experience as the user will not need to keep logging in to the system after the login token expired.
Implement real-time feature to all modules of the system	The implemented system should be able to update all data in real time. This feature will allow users to always keep track of the latest information and updates made from other ends. This is crucial especially for actions such as deleting action because the user might happen to perform some updates on an item which was

	deleted by other user on other end. Then, faulty data will be created which will cause an error to occur.

REFERENCES

Ceylan, Z. and Bulkan, S., 2022. Drug Inventory Management of a Pharmacy using ABC and VED Analysis. *Eurasian Journal of Health Technology Assessment*, 2(1), p.15.

Chinn, A., 2022. What's the System Usability Scale (SUS) & How Can You Use It?. [online] HubSpot. Available at: <https://blog.hubspot.com/service/system-usability-scale-sus> [Accessed 31 August 2022].

Chung, L., Nixon, B.A., Yu, E. and Mylopoulos, J., 2012. *Non-functional requirements in software engineering* (Vol. 5). Springer Science & Business Media.

Curley, R., 2015. *4 Best Practices For Cycle Count Procedures* [online]. Available at <https://info.archon-interactive.com/blog/4-best-practices-for-cycle-count-procedures > [Accessed on 26 February, 2022].

Curley, R., 2016. Paper-based processes vs. warehouse management systems [online]. Available at <https://info.archon-interactive.com/blog/paper-based-processes-vs.-warehouse-management-systems > [Accessed 8 April 2022].

Dennis, A., Wixom, B. and Tegarden, D., 2015. *Systems analysis and design: An object-oriented approach with UML*. John wiley & sons.

Doglio, F., 2022. How to secure a REST API using JWT. [online] LogRocket. Available at: <https://blog.logrocket.com/how-to-secure-a-rest-api-using-jwt-7efd83e71432/> [Accessed 23 August 2022].

Drakeley, P., 2021. *How to Calculate ABC Classification - A Working Example* [online]. Available at: <https://www.eazystock.com/uk/blog-uk/abc-classification-calculation-inventory-management/ > [Accessed on 9 March, 2022].

English, L. and Yunusa-Kaltungo, A., 2022. A practical application of methodologies to determine asset criticality and work order prioritization. Engineering Reports, p.e12555.

Fathoni, F.A., Ridwan, A.Y. and Santosa, B., 2019, March. Development of inventory control application for pharmaceutical product using abc-ved cycle counting method to increase inventory record accuracy. In *2018 International Conference on Industrial Enterprise and System Engineering (ICoIESE 2018)* (pp. 266-271). Atlantis Press.

F Leal, L., Zancul, E. and Schützer, K., 2021. Industry 4.0 Learning Factory Phased Development. *Eduardo and Schützer, Klaus, Industry, 4*.

Harjono, H., 2019. *Warehouse management system and its impact on data accuracy*. [online] President University. Available at <https://budi.rahardjo.id/files/courses/2020/DL-023201905028-Report-Heru-Rev1.pdf > [Accessed 3 March 2022].

Hodgson, R., 2019. Paper-based warehousing costs you more than money [online]. Available at <https://www.theaccessgroup.com/en-gb/blog/paper-based-warehousing-costs-you-more-than-money/ > [Accessed on 8 April 2022]

IEEE Computer Society. Software Engineering Standards Committee and IEEE-SA Standards Board, 1998. *IEEE recommended practice for software requirements specifications* (Vol. 830, No. 1998). IEEE.

International Standards Organisation. ISO 55000:2014 Asset Management - Overview, Principles and Terminology. ISO; 2014.

James, K., 2016. Design and implementation of an automated inventory management system case study: Smart Shoppers Masaka. Bachelor. Kampala International University. Available at: < https://ir.kiu.ac.ug/bitstream/20.500.12306/2297/1/DESIGN%20AND%20IMPLEMENTATION%20OF%20AN%20AUTOMATED%20INVENTORY%20MANAGEMENT%20SYSTEM.pdf > [Accessed 20 March 2022].

Jenkins, A., 2022. *20 Inventory Management Challenges And Solutions For 2022 And Beyond* [online]. Available at <https://www.netsuite.com/portal/resource/articles/inventory-management/inventory-management-challenges.shtml > [Accessed on 26 February 2022].

Maharjan, A. S. and Humagain, M., 2016. *Inventory Management System*. Bachelor. Pokhara University. Available at: <https://www.academia.edu/26003928/Final_Year_Project_On_Inventory_Management_System_Submitted_By > [Accessed 20 March 2022].

Mahtamtama, E., Ridwan, A.Y. and Santosa, B., 2018, October. Development Of Cycle Counting Monitoring Dashboard With Buffer Time Management For Cocoa Company. In *2018 12th International Conference on Telecommunication Systems, Services, and Applications (TSSA)* (pp. 1-5). IEEE.

Ministry of Economic Affairs, 2019. *Shared Prosperity Vision 2030*. [pdf] Putrajaya: Ministry of Economic Affairs. Available at: <https://www.epu.gov.my/sites/default/files/2020-02/Shared%20Prosperity%20Vision%202030.pdf >[Accessed 11 March 2022].

NC-Vision, 2021, *Top 3 Warehouse Management Challenges to Watch Out for In 2021* [online]. Available at: <https://nc-vision.com/top-3-warehouse-management-challenges-to-watch-out-for-in-2021and-how-to-overcome-them/ > [Accessed on 27 February, 2022].

Nielsen, J., & Landauer, T.K. (1993). A mathematical model of the finding of usability problems. Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems.

Ramaa, A., Subramanya, K.N. and Rangaswamy, T.M., 2012. Impact of warehouse management system in a supply chain. *International Journal of Computer Applications*, *54*(1).

Reyes, J., Alvarez, K. and Vasquez, R., 2016. Dynamic buffer management for raw material supply in the footwear industry. *Journal of Industrial and Intelligent Information Vol*, *4*(1).

Sasmito, G.W., Wibowo, D.S. and Dairoh, D., 2020. Implementation of Rapid Application Development Method in the Development of Geographic Information Systems of Industrial Centers. *Journal of information and communication convergence engineering*, *18*(3), pp.194-200.

Schwarz, L., 2021.. *Inventory Cycle Counting 101: Best Practices & Benefits* [online]. Available at <https://www.netsuite.com/portal/resource/articles/inventory-management/using-inventory-control-software-for-cycle-counting.shtml#:~:text=What%20Is%20Cycle%20Counting%3F,have%20counted%20all%20their%20goods > [Accessed on 26 February 2022].

Shaydulin, R. and Sybrandt, J., 2017. To agile, or not to agile: A comparison of software development methodologies. *arXiv preprint arXiv:1704.07469*.

Shen, Y., 2017. *Applied Cycle Counting Method for Grocery Retail Industry* (Doctoral dissertation, Pennsylvania State University).

Tahir, D., 2020. *Design and implementation of an inventory management system for Walid Halal Spices*. Bachelor. Baze University. Available at: < https://portal.bazeuniversity.edu.ng/student/assets/thesis/20210215004507841505869.pdf > [Accessed 20 March 2022].

WiSys, n.d, *3 Common Inventory Management Challenges* [online]. Available at: <https://www.wisys.com/3-common-inventory-management-challenges/ > [Accessed on 26 February, 2022].

Zhou, J.Y., 2004. *Functional requirements and non-functional requirements: a survey* (Doctoral dissertation, Concordia University).

APPENDICES

APPENDIX A: Questionnaire Form

A WEB-BASED WAREHOUSE INVENTORY MANAGEMENT SYSTEM FOR STOCK ITEMS TRACKING

Dear respondents,

I am Elaine Low Jing Yi, a year 3 undergraduate student of Bachelor of Science (HONS) Software Engineering from Lee Kong Chian Faculty of Engineering & Science (LKCFES), Universiti Tunku Abdul Rahman (UTAR), Sungai Long Campus.

The aim of this questionnaire is to gather opinions on impact of data accuracy of warehouse inventories and issue regarding physical inventory counts and cycle counting. This will contributes to my final year project (FYP), which is to develop a web-based warehouse inventory management system particularly for stock item tracking and cycle counting.

Respondents must fulfill the criteria listed below:
- own or working in a warehouse
- Understand basic operations in the warehouse
- ability to understand English

This questionnaire will consists of four sections:
- Section A: Demographic Information
- Section B: General Information
- Section C: Inventory Accuracy
- Section D: Physical counting/ Cycle counting

The approximate time to complete the questionnaire is around **5 to 10 minutes**. All of the responses will stay anonymous and the data collected will only be used in requirements gathering for my final year project.

If you are willing to participate in this questionnaire, please answer all of the questions that you feel relatable and leave out those that do not apply to your scenario.

If you have any questions on this questionnaire, kindly contact me via elainelowjingyi00@1utar.my. Thank you for your time in answering this questionnaire. Your responses is highly appreciated as it greatly assists me in my final year project.

*Required

1. I have read the information above. By clicking on the "Agree", I am consenting to participate in this survey/questionnaire project. *

   Mark only one oval.

   ◯ Agree

Section A: Demographic Information

2. Name

   _____

3. Gender *

   Mark only one oval.

   ◯ Male
   ◯ Female

4. Age *

   Mark only one oval.

   ◯ 21-30
   ◯ 31-40
   ◯ 41-50
   ◯ 51-60
   ◯ 61 and above

5. Race *

Mark only one oval.

- ◯ Malay
- ◯ Chinese
- ◯ Indian
- ◯ Other: _____

6. Email

_____

Section B: General Information

7. Do you operate your own warehouse? *

Mark only one oval.

- ◯ Yes
- ◯ No

8. Do you operate a warehouse for others? *

Mark only one oval.

- ◯ Yes
- ◯ No

9. Is there any warehouse management system in use? *

Mark only one oval.

- ◯ Yes
- ◯ No

10. More than one warehouse from different location/sites? *

Mark only one oval.

- ◯ Yes
- ◯ No

11. Warehouse Name/Location (insert multiple if more than one)

_____

_____

_____

_____

_____

12. What kind of commodities are stored in the warehouse(s)? *

_____

_____

_____

_____

_____

13. What is your role/position in warehouse management? *

_____

Section C: Inventory accuracy

14. Methods of inventory recording *

Mark only one oval.

- ◯ Paper records
- ◯ Excel / online spreadsheets
- ◯ Using a Warehouse Management System (WMS) or any similar softwares
- ◯ Other: _____

15. Is there any discrepancies between inventory count recorded and the actual count? *

Mark only one oval.

- ◯ Yes
- ◯ No

16. Is there any inconsistency of inventory count across multiple location/sites? *

Mark only one oval.

- ◯ Yes
- ◯ No

17. Would inventory and data accuracy affect business operation? *

Mark only one oval.

- ◯ Yes
- ◯ No

18. If yes, why do you think inventory accuracy would affect business operation?

Tick all that apply.

- [ ] Increase cost on inventory holding for excess inventory
- [ ] Additional time on researching discrepancies and correct data
- [ ] Increase risks of stock running-out
- [ ] Increase risk of cutting orders or rescheduling the production line, which impacts the service to customers
- [ ] Other: _____

19. In your opinion, what is the main cause of inaccuracy of inventory record? *

_____

_____

_____

_____

_____

Section D: Physical counting/ Cycle counting

20. For inventory controls, which of the following method is implemented in your warehouse? *

Mark only one oval.

- ◯ Physical inventory count (halts operation and count annually)
- ◯ Cycle counting
- ◯ All of the above
- ◯ Other: _____

Instruction: Please answer all questions by choosing the most appropriate answer.
1 = Strongly Disagree 2 = Disagree 3 = Neutral 4 = Agree 5 = Strongly Agree

21. Manual inventory recording/ counting without WMS system is troublesome. *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

22. It is difficult to classify SKUs manually. *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

23. Extra manpower and resources will be needed to analyze and schedule for physical counting and cycle counting. *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

24. It is difficult to keep track of the cycle counting progress manually. *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

25. Continuous operations and incoming transactions while performing physical count/cycle count may cause false variances in inventory counts. *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

26. There will be a lot of hassle if were to deal with all of the physical reports regarding physical counts/ cycle counting. *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

27. Adjusting inventory record should be in real-time to avoid inconsistencies. *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

28. Extra effort will be needed to calculate inventory record accuracy (IRA) to keep track of the performance. *

Mark only one oval.

	1	2	3	4	5	
Strongly disagree	◯	◯	◯	◯	◯	Strongly agree

Thank you.

Your response is much appreciated.

APPENDIX B: User Acceptance Testing Form

User Acceptance Testing Form						
Tester's Name				Testing Start Date/ Time		
				Testing End Date/ Time		
Test Case ID	Module	Test Case Title	Test Steps	Expected Results	Status (Pass/Fail)	Comments
TC001	Login account	Login an account	1.User enters username and password of **ADMIN** account in the login page. 2.User clicks on the login button.	User will successfully logged-in into the system.		
TC002	Manage category	Add a new category	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage category" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage category screen. 4. User enters the name for the new category. 5. User clicks on "confirm" button.	The new category will be added successfully with a success notification.		
TC003	Manage category	Update a category name	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage category" tab from the top navigation bar. 2. User selects **the newly created category** from the displayed list and clicks on the "Edit" button.	The name of the category will be updated successfully with a success notification.		

			3. User edits the name of the category from the modal popup 4. After editing, user clicks on the "Confirm" button.			
TC004	Manage warehouse	Add a new warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage warehouse" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage warehouse screen. 4. User enters the name, location, number of bins (enter 100) and zones (enter "A,B,C") for the new warehouse. 5. User clicks on "confirm" button after all information has been entered.	The warehouse is added successfully with a success notification.		
TC005	Manage user	Add a new user	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage user" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage user screen. 4. User enters the name, email, contact number, IC number, role (select "STAFF"), employed in, address and username for the new user.	Two new users will be added successfully with a success notification.		

			5. For the warehouse input, user selects the newly created warehouse. 6. User clicks on “confirm” button after all information has been entered. 7. Repeat step 3 to 6 to create another user with STAFF role.			
TC006	Manage user	Update a user’s detail	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User selects **a newly created user** from the displayed list and clicks on the “Edit” button. 4. User edits some information of the user (except the warehouse and role) on the modal pop-up 5. After editing, user clicks on the “Confirm” button.	The details of the selected user will be updated successfully with a success notification.		
TC007	Manage warehouse	Update a warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User selects the **newly created warehouse** from the displayed list and clicks on the “Edit” button. 4. User edits the information by **adding a newly created user in**	The warehouse details are updated successfully with a success notification.		

			the warehouse manager field on the modal pop-up 5. After editing, user clicks on the "Confirm" button.			
TC008	Manage inventory	Add new inventory	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage inventory screen. 4. For warehouse and category fields, selects **the warehouse and category created earlier**. 4. User enters the name, quantity on hand, cost per unit and priority for the inventory. 5. User clicks on "confirm" button after all information has been entered. 6. Repeat step 3 to 5 to create one more inventory.	Inventories are added successfully and a notification of the bin number assigned to newly added inventory is shown.		
TC009	Manage inventory	Update an inventory	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User selects **a newly created inventory** from the displayed list and clicks on the "Edit" button.	The selected inventory details are updated according to user's input.		

			4. User edits the information of the inventory on the modal pop-up 5. After editing, user clicks on the "Confirm" button.			
TC010	View all inventories	View inventory and its details	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User view the list of inventories displayed. 4. User clicks on one of the inventory ID to view the details.	User will successfully view the inventory list and will be redirected to the inventory details page once an inventory id is clicked.		
TC011	Manage warehouse	Edit inventory to storage bin	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage warehouse" tab from the top navigation bar. 3. User clicks on **newly created warehouse's ID** to view the details. 4. In the warehouse details page, user selects the "storage bin" tab to view the list of storage bins in the warehouse 5. User selects the bin with the newly created inventory (can use the search to search for the inventory) and clicks the "Edit inventory" button.	The selected inventory will be added to the selected storage bin of the warehouse.		

			6. User clears the current inventory assigned on the modal popup and click on the “Edit inventory” button.			
TC012	Manage warehouse	Assign category to a storage bin	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User clicks on **newly created warehouse’s ID** to view the details. 4. In the warehouse details page, user selects the “storage bin” tab to view the list of storage bins in the warehouse 5. User clicks on the “Multi assign bin” to assign category to bins. 6. User selects a category and checks desired bin to assign. 7. User clicks on the “Assign” button.	The category of the selected bins is changed to the selected category.		
TC013	Login account	Reset password	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage User” tab and finds the newly created user with “MANAGER” role. 3. User takes note on the username and contact number.	A success notification will be shown.		

			4. User logouts and logins into the system using the username and contact number as the password. 5. User resets the password			
TC014	Manage cycle counting	Start a new cycle counting	1. User logins into the system using **the created manager-role user credentials (skip this step if you completed TC013)** 2. Note that there are **TWO** inventories listed in the View Inventory page, if no, kindly go back to **TC008** to create an inventory. 3. User clicks on the "Start Cycle counting" from the "Manage Cycle counting" tab dropdown on the top navigation bar. 3. User fills in all of the information in the form displayed. 4. User selects all inventories in the "Select Inventories" field. 4. User clicks the "Submit" button after completing the form. 5. User clicks the "Create" button in the summary modal popup.	A success notification will be shown stating that the cycle counting is started. Schedules will be created for all SKUs and assigned to the selected staffs evenly.		
TC015	Manage cycle counting	View all upcoming cycle counting SKUs	1. User clicks on the "View Cycle counting" from the "Manage Cycle counting" tab dropdown on the top navigation bar.	The list of schedules for the cycle counting and the settings will be displayed accordingly.		

			2. User clicks on the “View cycle count settings” button on the top right of the displayed table.			
TC016	View all staff	View all staff in a warehouse	1. User clicks on the “View Staff” tab on the top navigation bar. 2. User clicks on one of the staff ID from the list to view the staff details.	The list of staffs and their details will be displayed accordingly.		
TC017	Perform cycle counting	Perform cycle counting for an inventory	1. Repeat TC013 to login and reset password for the created **STAFF-role** user. 2. User clicks on the “Cycle Counting” tab from the top navigation bar. 3. User selects the “Upcoming” tab and views the list of upcoming cycle counting schedules 3. User chooses the first one in the list and clicks the “Count” button. 4. User enters the actual count on the modal popup 5. User clicks the “Create” button 6. User refresh the page. 7. Repeat step 3 to 6 to count another one.	The cycle counting record is created with success notification.		
TC018	Record inventory	Check in stock for an inventory	1. User clicks on the “Check In/Out Stock” tab from the top navigation bar. 2. User views the list of inventories displayed.	The selected inventory’s quantity on hand will be updated in real time without refreshing the page.		

			3. User chooses one of the inventories and clicks on the “Check In/Out Stock” button. 4. User selects the “Check In” tab in the modal popup. 5. User enters the quantity and remarks and clicks the “Confirm” button.			
TC019	Record inventory	Check out stock for an inventory	1. User clicks on the “Check In/Out Stock” tab from the top navigation bar. 2. User views the list of inventories displayed. 3. User chooses one of the inventories and clicks on the “Check In/Out Stock” button. 4. User selects the “Check Out” tab in the modal popup. 5. User enters the quantity and remarks and clicks the “Confirm” button.	The selected inventory’s quantity on hand will be updated in real time without refreshing the page.		
TC020	Manage cycle count approval report	Approve cycle count approval report	1. User logins into the system using **the created manager-role user credentials.** 2. User clicks on the “View reports” tab from the top navigation bar. 3. User selects the “Cycle Count Approval Report” tab and views the list of approval reports listed.	The summary report for the selected cycle counting will be generated with IRA calculated.		

			4. User clicks the “Approve” button for one of the approval reports. 5. User clicks the “Confirm” button in the confirmation modal. 6. User refresh the page.			
TC021	Manage cycle count approval report	Reject cycle count approval report	1. User clicks on the “View reports” tab from the top navigation bar. 2. User selects the “Cycle Count Approval Report” tab and views the list of approval reports listed. 3. User clicks the “Reject” button for one of the approval reports. 4. User checks the checkbox for reassigning staff to recount the SKU. 5. User clicks the “Confirm” button in the confirmation modal.	A success message will be shown to indicate the success rescheduling.		
TC022	View reports	View cycle counting summary report	1. User clicks on the “View reports” tab from the top navigation bar. 2. User selects the “Cycle Count Summary Report” tab and views the list of summary reports listed. 3. User clicks on the cycle count ID for one of the summary reports	The list of summary reports and their details will be displayed accordingly.		
TC023	View reports	View daily reports	1. User clicks on the “View reports” tab from the top navigation bar.	The list of daily reports and their details and stocks will be displayed accordingly.		

			2. User selects the “Daily report” tab and views the list of daily reports listed. 3. User clicks on the “View” button for one of the daily reports			
TC024	Manage inventory	Delete an inventory	1. User logins into the system using **ADMIN** account provided. 2. User clicks on the “Manage inventory” tab from the top navigation bar. 3. User **finds one of the newly created inventories** from the displayed list (can utilize the filter or search function) and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The selected inventory is successfully deleted from the warehouse.		
TC025	Manage user	Delete a user	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User selects **one of the created users** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The status of the selected user will be set to “INACTIVE”.		
TC026	Manage category	Delete a category	1. User logins into the system using **ADMIN** account.	The selected category will be deleted from the list.		

			2. User clicks on the “Manage category” tab from the top navigation bar. 3. User **selects the newly created category** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.			
TC027	Manage warehouse	Delete a warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User **selects the newly created warehouse** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The selected warehouse is successfully deleted with a success message.		

APPENDIX C: User Acceptance Testing Results

Expert testers

User Acceptance Testing Form						
Tester's Name	Teh Khai Ming			Testing Start Date/ Time	27/8/2022 10:07pm	
				Testing End Date/ Time	27/8/2022 12.06am	
Test Case ID	Module	Test Case Title	Test Steps	Expected Results	Status (Pass/Fail)	Comments
TC001	Login account	Login an account	1.User enters username and password of **ADMIN** account in the login page. 2.User clicks on the login button.	User will successfully logged-in into the system.	Pass	
TC002	Manage category	Add a new category	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage category" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage category screen. 4. User enters the name for the new category. 5. User clicks on "confirm" button.	The new category will be added successfully with a success notification.	Pass	Added 'khai ming'
TC003	Manage category	Update a category name	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage category" tab from the top navigation bar.	The name of the category will be updated successfully with a success notification.	Pass	Edit to 'not khai ming'

			2. User selects **the newly created category** from the displayed list and clicks on the “Edit” button. 3. User edits the name of the category from the modal popup 4. After editing, user clicks on the “Confirm” button.			
TC004	Manage warehouse	Add a new warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User clicks on the “Add” button on the top right of the Manage warehouse screen. 4. User enters the name, location, number of bins (enter 100) and zones (enter “A,B,C”) for the new warehouse. 5. User clicks on “confirm” button after all information has been entered.	The warehouse is added successfully with a success notification.	Pass	data (khai ming, kota Kinabalu, 100, (A,B,C))
TC005	Manage user	Add a new user	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User clicks on the “Add” button on the top right of the Manage user screen. 4. User enters the name, email, contact number, IC number, role	Two new users will be added successfully with a success notification.	pass	

			(select “STAFF”), employed in, address and username for the new user. 5. For the warehouse input, user selects the newly created warehouse. 6. User clicks on “confirm” button after all information has been entered. 7. Repeat step 3 to 6 to create another user with STAFF role.			
TC006	Manage user	Update a user’s detail	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User selects **a newly created user** from the displayed list and clicks on the “Edit” button. 4. User edits some information of the user (except the warehouse and role) on the modal pop-up 5. After editing, user clicks on the “Confirm” button.	The details of the selected user will be updated successfully with a success notification.	Pass	
TC007	Manage warehouse	Update a warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User selects the **newly created warehouse** from the displayed	The warehouse details are updated successfully with a success notification.	Pass	

			list and clicks on the “Edit” button. 4. User edits the information by **adding a newly created user in the warehouse manager field** on the modal pop-up 5. After editing, user clicks on the “Confirm” button.			
TC008	Manage inventory	Add new inventory	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage inventory” tab from the top navigation bar. 3. User clicks on the “Add” button on the top right of the Manage inventory screen. 4. For warehouse and category fields, selects **the warehouse and category created earlier**. 4. User enters the name, quantity on hand, cost per unit and priority for the inventory. 5. User clicks on “confirm” button after all information has been entered. 6. Repeat step 3 to 5 to create one more inventory.	Inventories are added successfully and a notification of the bin number assigned to newly added inventory is shown.	Pass	
TC009	Manage inventory	Update an inventory	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage inventory” tab from the top navigation bar.	The selected inventory details are updated according to user’s input.	Pass	

			3. User selects **a newly created inventory** from the displayed list and clicks on the "Edit" button. 4. User edits the information of the inventory on the modal pop-up 5. After editing, user clicks on the "Confirm" button.			
TC010	View all inventories	View inventory and its details	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User view the list of inventories displayed. 4. User clicks on one of the inventory ID to view the details.	User will successfully view the inventory list and will be redirected to the inventory details page once an inventory id is clicked.	Pass	
TC011	Manage warehouse	Edit inventory to storage bin	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage warehouse" tab from the top navigation bar. 3. User clicks on **newly created warehouse's ID** to view the details. 4. In the warehouse details page, user selects the "storage bin" tab to view the list of storage bins in the warehouse 5. User selects the bin with the newly created inventory (can use the search to search for the	The selected inventory will be added to the selected storage bin of the warehouse.	Pass	

			inventory) and clicks the “Edit inventory” button. 6. User clears the current inventory assigned on the modal popup and click on the “Edit inventory” button.			
TC012	Manage warehouse	Assign category to a storage bin	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User clicks on **newly created warehouse’s ID** to view the details. 4. In the warehouse details page, user selects the “storage bin” tab to view the list of storage bins in the warehouse 5. User clicks on the “Multi assign bin” to assign category to bins. 6. User selects a category and checks desired bin to assign. 7. User clicks on the “Assign” button.	The category of the selected bins is changed to the selected category.	Pass	
TC013	Login account	Reset password	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage User” tab and finds the newly created user with “MANAGER” role.	A success notification will be shown.	Fail	Edited the phone number, the new phone number cannot be used as the password, have to use the phone number

			3. User takes note on the username and contact number. 4. User logouts and logins into the system using the username and contact number as the password. 5. User resets the password			that is created upon adding the new user
TC014	Manage cycle counting	Start a new cycle counting	1. User logins into the system using **the created manager-role user credentials (skip this step if you completed TC013)** 2. Note that there are **TWO** inventories listed in the View Inventory page, if no, kindly go back to **TC008** to create an inventory. 3. User clicks on the "Start Cycle counting" from the "Manage Cycle counting" tab dropdown on the top navigation bar. 3. User fills in all of the information in the form displayed. 4. User selects all inventories in the "Select Inventories" field. 4. User clicks the "Submit" button after completing the form. 5. User clicks the "Create" button in the summary modal popup.	A success notification will be shown stating that the cycle counting is started. Schedules will be created for all SKUs and assigned to the selected staffs evenly.	Pass	
TC015	Manage cycle counting	View all upcoming cycle counting SKUs	1. User clicks on the "View Cycle counting" from the "Manage	The list of schedules for the cycle counting and	Pass	

			Cycle counting" tab dropdown on the top navigation bar. 2. User clicks on the "View cycle count settings" button on the top right of the displayed table.	the settings will be displayed accordingly.		
TC016	View all staff	View all staff in a warehouse	1. User clicks on the "View Staff" tab on the top navigation bar. 2. User clicks on one of the staff ID from the list to view the staff details.	The list of staffs and their details will be displayed accordingly.	Pass	
TC017	Perform cycle counting	Perform cycle counting for an inventory	1. Repeat TC013 to login and reset password for the created **STAFF-role** user. 2. User clicks on the "Cycle Counting" tab from the top navigation bar. 3. User selects the "Upcoming" tab and views the list of upcoming cycle counting schedules 3. User chooses the first one in the list and clicks the "Count" button. 4. User enters the actual count on the modal popup 5. User clicks the "Create" button 6. User refresh the page. 7. Repeat step 3 to 6 to count another one.	The cycle counting record is created with success notification.	Pass	
TC018	Record inventory	Check in stock for an inventory	1. User clicks on the "Check In/Out Stock" tab from the top navigation bar.	The selected inventory's quantity on hand will be updated in	Pass	

			2. User views the list of inventories displayed. 3. User chooses one of the inventories and clicks on the "Check In/Out Stock" button. 4. User selects the "Check In" tab in the modal popup. 5. User enters the quantity and remarks and clicks the "Confirm" button.	real time without refreshing the page.		
TC019	Record inventory	Check out stock for an inventory	1. User clicks on the "Check In/Out Stock" tab from the top navigation bar. 2. User views the list of inventories displayed. 3. User chooses one of the inventories and clicks on the "Check In/Out Stock" button. 4. User selects the "Check Out" tab in the modal popup. 5. User enters the quantity and remarks and clicks the "Confirm" button.	The selected inventory's quantity on hand will be updated in real time without refreshing the page.	Pass	
TC020	Manage cycle count approval report	Approve cycle count approval report	1. User logins into the system using **the created manager-role user credentials.** 2. User clicks on the "View reports" tab from the top navigation bar. 3. User selects the "Cycle Count Approval Report" tab and views the list of approval reports listed.	The summary report for the selected cycle counting will be generated with IRA calculated.	Pass	

			4. User clicks the “Approve” button for one of the approval reports. 5. User clicks the “Confirm” button in the confirmation modal. 6. User refresh the page.			
TC021	Manage cycle count approval report	Reject cycle count approval report	1. User clicks on the “View reports” tab from the top navigation bar. 2. User selects the “Cycle Count Approval Report” tab and views the list of approval reports listed. 3. User clicks the “Reject” button for one of the approval reports. 4. User checks the checkbox for reassigning staff to recount the SKU. 5. User clicks the “Confirm” button in the confirmation modal.	A success message will be shown to indicate the success rescheduling.	Pass	
TC022	View reports	View cycle counting summary report	1. User clicks on the “View reports” tab from the top navigation bar. 2. User selects the “Cycle Count Summary Report” tab and views the list of summary reports listed. 3. User clicks on the cycle count ID for one of the summary reports	The list of summary reports and their details will be displayed accordingly.	Pass	
TC023	View reports	View daily reports	1. User clicks on the “View reports” tab from the top navigation bar.	The list of daily reports and their details and stocks will be displayed accordingly.	Pass	

			2. User selects the “Daily report” tab and views the list of daily reports listed. 3. User clicks on the “View” button for one of the daily reports			
TC024	Manage inventory	Delete an inventory	1. User logins into the system using **ADMIN** account provided. 2. User clicks on the “Manage inventory” tab from the top navigation bar. 3. User **finds one of the newly created inventories** from the displayed list (can utilize the filter or search function) and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The selected inventory is successfully deleted from the warehouse.	Pass	
TC025	Manage user	Delete a user	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User selects **one of the created users** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The status of the selected user will be set to “INACTIVE”.	Pass	
TC026	Manage category	Delete a category	1. User logins into the system using **ADMIN** account.	The selected category will be deleted from the list.	Pass	

			2. User clicks on the “Manage category” tab from the top navigation bar. 3. User **selects the newly created category** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.			
TC027	Manage warehouse	Delete a warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User **selects the newly created warehouse** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The selected warehouse is successfully deleted with a success message.	Pass	

User Acceptance Testing Form						
Tester's Name				Testing Start Date/ Time		
				Testing End Date/ Time		
Test Case ID	Module	Test Case Title	Test Steps	Expected Results	Status (Pass/Fail)	Comments
TC001	Login account	Login an account	1.User enters username and password of **ADMIN** account in the login page. 2.User clicks on the login button.	User will successfully logged-in into the system.	Pass	'Opps' error message
TC002	Manage category	Add a new category	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage category" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage category screen. 4. User enters the name for the new category. 5. User clicks on "confirm" button.	The new category will be added successfully with a success notification.	Pass	
TC003	Manage category	Update a category name	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage category" tab from the top navigation bar. 2. User selects **the newly created category** from the displayed list and clicks on the "Edit" button. 3. User edits the name of the category from the modal popup	The name of the category will be updated successfully with a success notification.	Pass	Cancel button and close button not working

			4. After editing, user clicks on the "Confirm" button.			
TC004	Manage warehouse	Add a new warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage warehouse" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage warehouse screen. 4. User enters the name, location, number of bins (enter 100) and zones (enter "A,B,C") for the new warehouse. 5. User clicks on "confirm" button after all information has been entered.	The warehouse is added successfully with a success notification.	Pass	
TC005	Manage user	Add a new user	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage user" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage user screen. 4. User enters the name, email, contact number, IC number, role (select "STAFF"), employed in, address and username for the new user.	Two new users will be added successfully with a success notification.	Pass	

			5. For the warehouse input, user selects the newly created warehouse. 6. User clicks on “confirm” button after all information has been entered. 7. Repeat step 3 to 6 to create another user with STAFF role.			
TC006	Manage user	Update a user’s detail	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User selects **a newly created user** from the displayed list and clicks on the “Edit” button. 4. User edits some information of the user (except the warehouse and role) on the modal pop-up 5. After editing, user clicks on the “Confirm” button.	The details of the selected user will be updated successfully with a success notification.	Pass	
TC007	Manage warehouse	Update a warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User selects the **newly created warehouse** from the displayed list and clicks on the “Edit” button. 4. User edits the information by **adding a newly created user in**	The warehouse details are updated successfully with a success notification.	Pass	

			the warehouse manager field on the modal pop-up 5. After editing, user clicks on the "Confirm" button.			
TC008	Manage inventory	Add new inventory	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage inventory screen. 4. For warehouse and category fields, selects **the warehouse and category created earlier**. 4. User enters the name, quantity on hand, cost per unit and priority for the inventory. 5. User clicks on "confirm" button after all information has been entered. 6. Repeat step 3 to 5 to create one more inventory.	Inventories are added successfully and a notification of the bin number assigned to newly added inventory is shown.	Pass	
TC009	Manage inventory	Update an inventory	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User selects **a newly created inventory** from the displayed list and clicks on the "Edit" button.	The selected inventory details are updated according to user's input.	Pass	

			4. User edits the information of the inventory on the modal pop-up 5. After editing, user clicks on the "Confirm" button.			
TC010	View all inventories	View inventory and its details	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User view the list of inventories displayed. 4. User clicks on one of the inventory ID to view the details.	User will successfully view the inventory list and will be redirected to the inventory details page once an inventory id is clicked.	Pass	
TC011	Manage warehouse	Edit inventory to storage bin	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage warehouse" tab from the top navigation bar. 3. User clicks on **newly created warehouse's ID** to view the details. 4. In the warehouse details page, user selects the "storage bin" tab to view the list of storage bins in the warehouse 5. User selects the bin with the newly created inventory (can use the search to search for the inventory) and clicks the "Edit inventory" button.	The selected inventory will be added to the selected storage bin of the warehouse.	Pass	

			6. User clears the current inventory assigned on the modal popup and click on the “Edit inventory” button.			
TC012	Manage warehouse	Assign category to a storage bin	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User clicks on **newly created warehouse’s ID** to view the details. 4. In the warehouse details page, user selects the “storage bin” tab to view the list of storage bins in the warehouse 5. User clicks on the “Multi assign bin” to assign category to bins. 6. User selects a category and checks desired bin to assign. 7. User clicks on the “Assign” button.	The category of the selected bins is changed to the selected category.	Pass	
TC013	Login account	Reset password	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage User” tab and finds the newly created user with “MANAGER” role. 3. User takes note on the username and contact number.	A success notification will be shown.	Pass	

			4. User logouts and logins into the system using the username and contact number as the password. 5. User resets the password			
TC014	Manage cycle counting	Start a new cycle counting	1. User logins into the system using **the created manager-role user credentials (skip this step if you completed TC013)** 2. Note that there are **TWO** inventories listed in the View Inventory page, if no, kindly go back to **TC008** to create an inventory. 3. User clicks on the "Start Cycle counting" from the "Manage Cycle counting" tab dropdown on the top navigation bar. 3. User fills in all of the information in the form displayed. 4. User selects all inventories in the "Select Inventories" field. 4. User clicks the "Submit" button after completing the form. 5. User clicks the "Create" button in the summary modal popup.	A success notification will be shown stating that the cycle counting is started. Schedules will be created for all SKUs and assigned to the selected staffs evenly.	Pass	
TC015	Manage cycle counting	View all upcoming cycle counting SKUs	1. User clicks on the "View Cycle counting" from the "Manage Cycle counting" tab dropdown on the top navigation bar.	The list of schedules for the cycle counting and the settings will be displayed accordingly.	Pass	

			2. User clicks on the “View cycle count settings” button on the top right of the displayed table.			
TC016	View all staff	View all staff in a warehouse	1. User clicks on the “View Staff” tab on the top navigation bar. 2. User clicks on one of the staff ID from the list to view the staff details.	The list of staffs and their details will be displayed accordingly.	Pass	
TC017	Perform cycle counting	Perform cycle counting for an inventory	1. Repeat TC013 to login and reset password for the created **STAFF-role** user. 2. User clicks on the “Cycle Counting” tab from the top navigation bar. 3. User selects the “Upcoming” tab and views the list of upcoming cycle counting schedules 3. User chooses the first one in the list and clicks the “Count” button. 4. User enters the actual count on the modal popup 5. User clicks the “Create” button 6. User refresh the page. 7. Repeat step 3 to 6 to count another one.	The cycle counting record is created with success notification.	Pass	
TC018	Record inventory	Check in stock for an inventory	1. User clicks on the “Check In/Out Stock” tab from the top navigation bar. 2. User views the list of inventories displayed.	The selected inventory’s quantity on hand will be updated in real time without refreshing the page.	Pass	

			3. User chooses one of the inventories and clicks on the “Check In/Out Stock” button. 4. User selects the “Check In” tab in the modal popup. 5. User enters the quantity and remarks and clicks the “Confirm” button.			
TC019	Record inventory	Check out stock for an inventory	1. User clicks on the “Check In/Out Stock” tab from the top navigation bar. 2. User views the list of inventories displayed. 3. User chooses one of the inventories and clicks on the “Check In/Out Stock” button. 4. User selects the “Check Out” tab in the modal popup. 5. User enters the quantity and remarks and clicks the “Confirm” button.	The selected inventory’s quantity on hand will be updated in real time without refreshing the page.	Pass	
TC020	Manage cycle count approval report	Approve cycle count approval report	1. User logins into the system using **the created manager-role user credentials.** 2. User clicks on the “View reports” tab from the top navigation bar. 3. User selects the “Cycle Count Approval Report” tab and views the list of approval reports listed.	The summary report for the selected cycle counting will be generated with IRA calculated.	Pass	No notification but got update

			4. User clicks the “Approve” button for one of the approval reports. 5. User clicks the “Confirm” button in the confirmation modal. 6. User refresh the page.			
TC021	Manage cycle count approval report	Reject cycle count approval report	1. User clicks on the “View reports” tab from the top navigation bar. 2. User selects the “Cycle Count Approval Report” tab and views the list of approval reports listed. 3. User clicks the “Reject” button for one of the approval reports. 4. User checks the checkbox for reassigning staff to recount the SKU. 5. User clicks the “Confirm” button in the confirmation modal.	A success message will be shown to indicate the success rescheduling.	Pass	No notification but got update
TC022	View reports	View cycle counting summary report	1. User clicks on the “View reports” tab from the top navigation bar. 2. User selects the “Cycle Count Summary Report” tab and views the list of summary reports listed. 3. User clicks on the cycle count ID for one of the summary reports	The list of summary reports and their details will be displayed accordingly.	Pass	
TC023	View reports	View daily reports	1. User clicks on the “View reports” tab from the top navigation bar.	The list of daily reports and their details and stocks will be displayed accordingly.	Pass	

			2. User selects the “Daily report” tab and views the list of daily reports listed. 3. User clicks on the “View” button for one of the daily reports			
TC024	Manage inventory	Delete an inventory	1. User logins into the system using **ADMIN** account provided. 2. User clicks on the “Manage inventory” tab from the top navigation bar. 3. User **finds one of the newly created inventories** from the displayed list (can utilize the filter or search function) and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The selected inventory is successfully deleted from the warehouse.	Pass	
TC025	Manage user	Delete a user	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User selects **one of the created users** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The status of the selected user will be set to “INACTIVE”.	Pass	
TC026	Manage category	Delete a category	1. User logins into the system using **ADMIN** account.	The selected category will be deleted from the list.	Pass	

			2. User clicks on the “Manage category” tab from the top navigation bar. 3. User **selects the newly created category** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.			
TC027	Manage warehouse	Delete a warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User **selects the newly created warehouse** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The selected warehouse is successfully deleted with a success message.	Pass	

Novice testers

User Acceptance Testing Form						
Tester's Name	Low Ee Lyne			Testing Start Date/ Time	27/8/2022 11:17am	
				Testing End Date/ Time	27/8/2022 12.06pm	
Test Case ID	Module	Test Case Title	Test Steps	Expected Results	Status (Pass/Fail)	Comments
TC001	Login account	Login an account	1.User enters username and password of **ADMIN** account in the login page. 2.User clicks on the login button.	User will successfully logged-in into the system.	Pass	
TC002	Manage category	Add a new category	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage category" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage category screen. 4. User enters the name for the new category. 5. User clicks on "confirm" button.	The new category will be added successfully with a success notification.	Pass	
TC003	Manage category	Update a category name	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage category" tab from the top navigation bar.	The name of the category will be updated successfully with a success notification.	Pass	

			2. User selects **the newly created category** from the displayed list and clicks on the “Edit” button. 3. User edits the name of the category from the modal popup 4. After editing, user clicks on the “Confirm” button.			
TC004	Manage warehouse	Add a new warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User clicks on the “Add” button on the top right of the Manage warehouse screen. 4. User enters the name, location, number of bins (enter 100) and zones (enter “A,B,C”) for the new warehouse. 5. User clicks on “confirm” button after all information has been entered.	The warehouse is added successfully with a success notification.	Pass	
TC005	Manage user	Add a new user	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User clicks on the “Add” button on the top right of the Manage user screen. 4. User enters the name, email, contact number, IC number, role	Two new users will be added successfully with a success notification.	pass	

			(select “STAFF”), employed in, address and username for the new user. 5. For the warehouse input, user selects the newly created warehouse. 6. User clicks on “confirm” button after all information has been entered. 7. Repeat step 3 to 6 to create another user with STAFF role.			
TC006	Manage user	Update a user’s detail	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User selects **a newly created user** from the displayed list and clicks on the “Edit” button. 4. User edits some information of the user (except the warehouse and role) on the modal pop-up 5. After editing, user clicks on the “Confirm” button.	The details of the selected user will be updated successfully with a success notification.	Pass	
TC007	Manage warehouse	Update a warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User selects the **newly created warehouse** from the displayed	The warehouse details are updated successfully with a success notification.	Pass	

			list and clicks on the “Edit” button. 4. User edits the information by **adding a newly created user in the warehouse manager field** on the modal pop-up 5. After editing, user clicks on the “Confirm” button.			
TC008	Manage inventory	Add new inventory	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage inventory” tab from the top navigation bar. 3. User clicks on the “Add” button on the top right of the Manage inventory screen. 4. For warehouse and category fields, selects **the warehouse and category created earlier**. 4. User enters the name, quantity on hand, cost per unit and priority for the inventory. 5. User clicks on “confirm” button after all information has been entered. 6. Repeat step 3 to 5 to create one more inventory.	Inventories are added successfully and a notification of the bin number assigned to newly added inventory is shown.	Pass	
TC009	Manage inventory	Update an inventory	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage inventory” tab from the top navigation bar.	The selected inventory details are updated according to user’s input.	Pass	

			3. User selects **a newly created inventory** from the displayed list and clicks on the "Edit" button. 4. User edits the information of the inventory on the modal pop-up 5. After editing, user clicks on the "Confirm" button.			
TC010	View all inventories	View inventory and its details	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User view the list of inventories displayed. 4. User clicks on one of the inventory ID to view the details.	User will successfully view the inventory list and will be redirected to the inventory details page once an inventory id is clicked.	Pass	
TC011	Manage warehouse	Edit inventory to storage bin	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage warehouse" tab from the top navigation bar. 3. User clicks on **newly created warehouse's ID** to view the details. 4. In the warehouse details page, user selects the "storage bin" tab to view the list of storage bins in the warehouse 5. User selects the bin with the newly created inventory (can use the search to search for the	The selected inventory will be added to the selected storage bin of the warehouse.	Pass	

			inventory) and clicks the “Edit inventory” button. 6. User clears the current inventory assigned on the modal popup and click on the “Edit inventory” button.			
TC012	Manage warehouse	Assign category to a storage bin	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User clicks on **newly created warehouse’s ID** to view the details. 4. In the warehouse details page, user selects the “storage bin” tab to view the list of storage bins in the warehouse 5. User clicks on the “Multi assign bin” to assign category to bins. 6. User selects a category and checks desired bin to assign. 7. User clicks on the “Assign” button.	The category of the selected bins is changed to the selected category.	Pass	
TC013	Login account	Reset password	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage User” tab and finds the newly created user with “MANAGER” role.	A success notification will be shown.	Pass	

			3. User takes note on the username and contact number. 4. User logouts and logins into the system using the username and contact number as the password. 5. User resets the password			
TC014	Manage cycle counting	Start a new cycle counting	1. User logins into the system using **the created manager-role user credentials (skip this step if you completed TC013)** 2. Note that there are **TWO** inventories listed in the View Inventory page, if no, kindly go back to **TC008** to create an inventory. 3. User clicks on the "Start Cycle counting" from the "Manage Cycle counting" tab dropdown on the top navigation bar. 3. User fills in all of the information in the form displayed. 4. User selects all inventories in the "Select Inventories" field. 4. User clicks the "Submit" button after completing the form. 5. User clicks the "Create" button in the summary modal popup.	A success notification will be shown stating that the cycle counting is started. Schedules will be created for all SKUs and assigned to the selected staffs evenly.	Pass	
TC015	Manage cycle counting	View all upcoming cycle counting SKUs	1. User clicks on the "View Cycle counting" from the "Manage	The list of schedules for the cycle counting and	Pass	

			Cycle counting" tab dropdown on the top navigation bar. 2. User clicks on the "View cycle count settings" button on the top right of the displayed table.	the settings will be displayed accordingly.		
TC016	View all staff	View all staff in a warehouse	1. User clicks on the "View Staff" tab on the top navigation bar. 2. User clicks on one of the staff ID from the list to view the staff details.	The list of staffs and their details will be displayed accordingly.	Pass	
TC017	Perform cycle counting	Perform cycle counting for an inventory	1. Repeat TC013 to login and reset password for the created **STAFF-role** user. 2. User clicks on the "Cycle Counting" tab from the top navigation bar. 3. User selects the "Upcoming" tab and views the list of upcoming cycle counting schedules 3. User chooses the first one in the list and clicks the "Count" button. 4. User enters the actual count on the modal popup 5. User clicks the "Create" button 6. User refresh the page. 7. Repeat step 3 to 6 to count another one.	The cycle counting record is created with success notification.	Pass	
TC018	Record inventory	Check in stock for an inventory	1. User clicks on the "Check In/Out Stock" tab from the top navigation bar.	The selected inventory's quantity on hand will be updated in	Pass	

			2. User views the list of inventories displayed. 3. User chooses one of the inventories and clicks on the “Check In/Out Stock” button. 4. User selects the “Check In” tab in the modal popup. 5. User enters the quantity and remarks and clicks the “Confirm” button.	real time without refreshing the page.		
TC019	Record inventory	Check out stock for an inventory	1. User clicks on the “Check In/Out Stock” tab from the top navigation bar. 2. User views the list of inventories displayed. 3. User chooses one of the inventories and clicks on the “Check In/Out Stock” button. 4. User selects the “Check Out” tab in the modal popup. 5. User enters the quantity and remarks and clicks the “Confirm” button.	The selected inventory’s quantity on hand will be updated in real time without refreshing the page.	Pass	
TC020	Manage cycle count approval report	Approve cycle count approval report	1. User logins into the system using **the created manager-role user credentials.** 2. User clicks on the “View reports” tab from the top navigation bar. 3. User selects the “Cycle Count Approval Report” tab and views the list of approval reports listed.	The summary report for the selected cycle counting will be generated with IRA calculated.	Pass	

			4. User clicks the “Approve” button for one of the approval reports. 5. User clicks the “Confirm” button in the confirmation modal. 6. User refresh the page.			
TC021	Manage cycle count approval report	Reject cycle count approval report	1. User clicks on the “View reports” tab from the top navigation bar. 2. User selects the “Cycle Count Approval Report” tab and views the list of approval reports listed. 3. User clicks the “Reject” button for one of the approval reports. 4. User checks the checkbox for reassigning staff to recount the SKU. 5. User clicks the “Confirm” button in the confirmation modal.	A success message will be shown to indicate the success rescheduling.	Pass	
TC022	View reports	View cycle counting summary report	1. User clicks on the “View reports” tab from the top navigation bar. 2. User selects the “Cycle Count Summary Report” tab and views the list of summary reports listed. 3. User clicks on the cycle count ID for one of the summary reports	The list of summary reports and their details will be displayed accordingly.	Pass	
TC023	View reports	View daily reports	1. User clicks on the “View reports” tab from the top navigation bar.	The list of daily reports and their details and stocks will be displayed accordingly.	Pass	

			2. User selects the “Daily report” tab and views the list of daily reports listed. 3. User clicks on the “View” button for one of the daily reports			
TC024	Manage inventory	Delete an inventory	1. User logins into the system using **ADMIN** account provided. 2. User clicks on the “Manage inventory” tab from the top navigation bar. 3. User **finds one of the newly created inventories** from the displayed list (can utilize the filter or search function) and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The selected inventory is successfully deleted from the warehouse.	Pass	
TC025	Manage user	Delete a user	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User selects **one of the created users** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The status of the selected user will be set to “INACTIVE”.	Pass	
TC026	Manage category	Delete a category	1. User logins into the system using **ADMIN** account.	The selected category will be deleted from the list.	Pass	

			2. User clicks on the “Manage category” tab from the top navigation bar. 3. User **selects the newly created category** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.			
TC027	Manage warehouse	Delete a warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User **selects the newly created warehouse** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The selected warehouse is successfully deleted with a success message.	Pass	

User Acceptance Testing Form						
Tester's Name	Low Jia Wei			Testing Start Date/ Time	28/8/2022 4:23pm	
				Testing End Date/ Time	28/8/2022 5.12 pm	
Test Case ID	Module	Test Case Title	Test Steps	Expected Results	Status (Pass/Fail)	Comments
TC001	Login account	Login an account	1.User enters username and password of **ADMIN** account in the login page. 2.User clicks on the login button.	User will successfully logged-in into the system.	Pass	
TC002	Manage category	Add a new category	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage category" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage category screen. 4. User enters the name for the new category. 5. User clicks on "confirm" button.	The new category will be added successfully with a success notification.	Pass	
TC003	Manage category	Update a category name	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage category" tab from the top navigation bar. 2. User selects **the newly created category** from the displayed list and clicks on the "Edit" button. 3. User edits the name of the category from the modal popup	The name of the category will be updated successfully with a success notification.	Pass	

			4. After editing, user clicks on the "Confirm" button.			
TC004	Manage warehouse	Add a new warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage warehouse" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage warehouse screen. 4. User enters the name, location, number of bins (enter 100) and zones (enter "A,B,C") for the new warehouse. 5. User clicks on "confirm" button after all information has been entered.	The warehouse is added successfully with a success notification.	Pass	
TC005	Manage user	Add a new user	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage user" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage user screen. 4. User enters the name, email, contact number, IC number, role (select "STAFF"), employed in, address and username for the new user.	Two new users will be added successfully with a success notification.	pass	

			5. For the warehouse input, user selects the newly created warehouse. 6. User clicks on “confirm” button after all information has been entered. 7. Repeat step 3 to 6 to create another user with STAFF role.			
TC006	Manage user	Update a user’s detail	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User selects **a newly created user** from the displayed list and clicks on the “Edit” button. 4. User edits some information of the user (except the warehouse and role) on the modal pop-up 5. After editing, user clicks on the “Confirm” button.	The details of the selected user will be updated successfully with a success notification.	Pass	
TC007	Manage warehouse	Update a warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User selects the **newly created warehouse** from the displayed list and clicks on the “Edit” button. 4. User edits the information by **adding a newly created user in**	The warehouse details are updated successfully with a success notification.	Pass	

			the warehouse manager field on the modal pop-up 5. After editing, user clicks on the "Confirm" button.			
TC008	Manage inventory	Add new inventory	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage inventory screen. 4. For warehouse and category fields, selects **the warehouse and category created earlier**. 4. User enters the name, quantity on hand, cost per unit and priority for the inventory. 5. User clicks on "confirm" button after all information has been entered. 6. Repeat step 3 to 5 to create one more inventory.	Inventories are added successfully and a notification of the bin number assigned to newly added inventory is shown.	Pass	
TC009	Manage inventory	Update an inventory	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User selects **a newly created inventory** from the displayed list and clicks on the "Edit" button.	The selected inventory details are updated according to user's input.	Pass	

			4. User edits the information of the inventory on the modal pop-up 5. After editing, user clicks on the "Confirm" button.			
TC010	View all inventories	View inventory and its details	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User view the list of inventories displayed. 4. User clicks on one of the inventory ID to view the details.	User will successfully view the inventory list and will be redirected to the inventory details page once an inventory id is clicked.	Pass	
TC011	Manage warehouse	Edit inventory to storage bin	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage warehouse" tab from the top navigation bar. 3. User clicks on **newly created warehouse's ID** to view the details. 4. In the warehouse details page, user selects the "storage bin" tab to view the list of storage bins in the warehouse 5. User selects the bin with the newly created inventory (can use the search to search for the inventory) and clicks the "Edit inventory" button.	The selected inventory will be added to the selected storage bin of the warehouse.	Pass	

			6. User clears the current inventory assigned on the modal popup and click on the “Edit inventory” button.			
TC012	Manage warehouse	Assign category to a storage bin	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User clicks on **newly created warehouse’s ID** to view the details. 4. In the warehouse details page, user selects the “storage bin” tab to view the list of storage bins in the warehouse 5. User clicks on the “Multi assign bin” to assign category to bins. 6. User selects a category and checks desired bin to assign. 7. User clicks on the “Assign” button.	The category of the selected bins is changed to the selected category.	Pass	
TC013	Login account	Reset password	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage User” tab and finds the newly created user with “MANAGER” role. 3. User takes note on the username and contact number.	A success notification will be shown.	Pass	

			4. User logouts and logins into the system using the username and contact number as the password. 5. User resets the password			
TC014	Manage cycle counting	Start a new cycle counting	1. User logins into the system using **the created manager-role user credentials (skip this step if you completed TC013)** 2. Note that there are **TWO** inventories listed in the View Inventory page, if no, kindly go back to **TC008** to create an inventory. 3. User clicks on the "Start Cycle counting" from the "Manage Cycle counting" tab dropdown on the top navigation bar. 3. User fills in all of the information in the form displayed. 4. User selects all inventories in the "Select Inventories" field. 4. User clicks the "Submit" button after completing the form. 5. User clicks the "Create" button in the summary modal popup.	A success notification will be shown stating that the cycle counting is started. Schedules will be created for all SKUs and assigned to the selected staffs evenly.	Pass	
TC015	Manage cycle counting	View all upcoming cycle counting SKUs	1. User clicks on the "View Cycle counting" from the "Manage Cycle counting" tab dropdown on the top navigation bar.	The list of schedules for the cycle counting and the settings will be displayed accordingly.	Pass	

			2. User clicks on the “View cycle count settings” button on the top right of the displayed table.			
TC016	View all staff	View all staff in a warehouse	1. User clicks on the “View Staff” tab on the top navigation bar. 2. User clicks on one of the staff ID from the list to view the staff details.	The list of staffs and their details will be displayed accordingly.	Pass	
TC017	Perform cycle counting	Perform cycle counting for an inventory	1. Repeat TC013 to login and reset password for the created **STAFF-role** user. 2. User clicks on the “Cycle Counting” tab from the top navigation bar. 3. User selects the “Upcoming” tab and views the list of upcoming cycle counting schedules 3. User chooses the first one in the list and clicks the “Count” button. 4. User enters the actual count on the modal popup 5. User clicks the “Create” button 6. User refresh the page. 7. Repeat step 3 to 6 to count another one.	The cycle counting record is created with success notification.	Pass	
TC018	Record inventory	Check in stock for an inventory	1. User clicks on the “Check In/Out Stock” tab from the top navigation bar. 2. User views the list of inventories displayed.	The selected inventory’s quantity on hand will be updated in real time without refreshing the page.	Pass	

			3. User chooses one of the inventories and clicks on the “Check In/Out Stock” button. 4. User selects the “Check In” tab in the modal popup. 5. User enters the quantity and remarks and clicks the “Confirm” button.			
TC019	Record inventory	Check out stock for an inventory	1. User clicks on the “Check In/Out Stock” tab from the top navigation bar. 2. User views the list of inventories displayed. 3. User chooses one of the inventories and clicks on the “Check In/Out Stock” button. 4. User selects the “Check Out” tab in the modal popup. 5. User enters the quantity and remarks and clicks the “Confirm” button.	The selected inventory’s quantity on hand will be updated in real time without refreshing the page.	Pass	
TC020	Manage cycle count approval report	Approve cycle count approval report	1. User logins into the system using **the created manager-role user credentials.** 2. User clicks on the “View reports” tab from the top navigation bar. 3. User selects the “Cycle Count Approval Report” tab and views the list of approval reports listed.	The summary report for the selected cycle counting will be generated with IRA calculated.	Pass	

			4. User clicks the “Approve” button for one of the approval reports. 5. User clicks the “Confirm” button in the confirmation modal. 6. User refresh the page.			
TC021	Manage cycle count approval report	Reject cycle count approval report	1. User clicks on the “View reports” tab from the top navigation bar. 2. User selects the “Cycle Count Approval Report” tab and views the list of approval reports listed. 3. User clicks the “Reject” button for one of the approval reports. 4. User checks the checkbox for reassigning staff to recount the SKU. 5. User clicks the “Confirm” button in the confirmation modal.	A success message will be shown to indicate the success rescheduling.	Pass	
TC022	View reports	View cycle counting summary report	1. User clicks on the “View reports” tab from the top navigation bar. 2. User selects the “Cycle Count Summary Report” tab and views the list of summary reports listed. 3. User clicks on the cycle count ID for one of the summary reports	The list of summary reports and their details will be displayed accordingly.	Pass	
TC023	View reports	View daily reports	1. User clicks on the “View reports” tab from the top navigation bar.	The list of daily reports and their details and stocks will be displayed accordingly.	Pass	

			2. User selects the “Daily report” tab and views the list of daily reports listed. 3. User clicks on the “View” button for one of the daily reports			
TC024	Manage inventory	Delete an inventory	1. User logins into the system using **ADMIN** account provided. 2. User clicks on the “Manage inventory” tab from the top navigation bar. 3. User **finds one of the newly created inventories** from the displayed list (can utilize the filter or search function) and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The selected inventory is successfully deleted from the warehouse.	Pass	
TC025	Manage user	Delete a user	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User selects **one of the created users** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The status of the selected user will be set to “INACTIVE”.	Pass	
TC026	Manage category	Delete a category	1. User logins into the system using **ADMIN** account.	The selected category will be deleted from the list.	Pass	

			2. User clicks on the “Manage category” tab from the top navigation bar. 3. User **selects the newly created category** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.			
TC027	Manage warehouse	Delete a warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User **selects the newly created warehouse** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The selected warehouse is successfully deleted with a success message.	Pass	

User Acceptance Testing Form						
Tester's Name				Testing Start Date/ Time	29/8/2022 3.23pm	
				Testing End Date/ Time	29/8/2022 4.32 pm	
Test Case ID	Module	Test Case Title	Test Steps	Expected Results	Status (Pass/Fail)	Comments
TC001	Login account	Login an account	1.User enters username and password of **ADMIN** account in the login page. 2.User clicks on the login button.	User will successfully logged-in into the system.	Pass	
TC002	Manage category	Add a new category	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage category" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage category screen. 4. User enters the name for the new category. 5. User clicks on "confirm" button.	The new category will be added successfully with a success notification.	Pass	
TC003	Manage category	Update a category name	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage category" tab from the top navigation bar. 2. User selects **the newly created category** from the displayed list and clicks on the "Edit" button. 3. User edits the name of the category from the modal popup	The name of the category will be updated successfully with a success notification.	Pass	

			4. After editing, user clicks on the "Confirm" button.			
TC004	Manage warehouse	Add a new warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage warehouse" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage warehouse screen. 4. User enters the name, location, number of bins (enter 100) and zones (enter "A,B,C") for the new warehouse. 5. User clicks on "confirm" button after all information has been entered.	The warehouse is added successfully with a success notification.	Pass	
TC005	Manage user	Add a new user	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage user" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage user screen. 4. User enters the name, email, contact number, IC number, role (select "STAFF"), employed in, address and username for the new user.	Two new users will be added successfully with a success notification.	pass	

			5. For the warehouse input, user selects the newly created warehouse. 6. User clicks on “confirm” button after all information has been entered. 7. Repeat step 3 to 6 to create another user with STAFF role.			
TC006	Manage user	Update a user’s detail	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User selects **a newly created user** from the displayed list and clicks on the “Edit” button. 4. User edits some information of the user (except the warehouse and role) on the modal pop-up 5. After editing, user clicks on the “Confirm” button.	The details of the selected user will be updated successfully with a success notification.	Pass	
TC007	Manage warehouse	Update a warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User selects the **newly created warehouse** from the displayed list and clicks on the “Edit” button. 4. User edits the information by **adding a newly created user in**	The warehouse details are updated successfully with a success notification.	Pass	

			the warehouse manager field on the modal pop-up 5. After editing, user clicks on the "Confirm" button.			
TC008	Manage inventory	Add new inventory	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User clicks on the "Add" button on the top right of the Manage inventory screen. 4. For warehouse and category fields, selects **the warehouse and category created earlier**. 4. User enters the name, quantity on hand, cost per unit and priority for the inventory. 5. User clicks on "confirm" button after all information has been entered. 6. Repeat step 3 to 5 to create one more inventory.	Inventories are added successfully and a notification of the bin number assigned to newly added inventory is shown.	Pass	
TC009	Manage inventory	Update an inventory	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User selects **a newly created inventory** from the displayed list and clicks on the "Edit" button.	The selected inventory details are updated according to user's input.	Pass	

			4. User edits the information of the inventory on the modal pop-up 5. After editing, user clicks on the "Confirm" button.			
TC010	View all inventories	View inventory and its details	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage inventory" tab from the top navigation bar. 3. User view the list of inventories displayed. 4. User clicks on one of the inventory ID to view the details.	User will successfully view the inventory list and will be redirected to the inventory details page once an inventory id is clicked.	Pass	
TC011	Manage warehouse	Edit inventory to storage bin	1. User logins into the system using **ADMIN** account. 2. User clicks on the "Manage warehouse" tab from the top navigation bar. 3. User clicks on **newly created warehouse's ID** to view the details. 4. In the warehouse details page, user selects the "storage bin" tab to view the list of storage bins in the warehouse 5. User selects the bin with the newly created inventory (can use the search to search for the inventory) and clicks the "Edit inventory" button.	The selected inventory will be added to the selected storage bin of the warehouse.	Pass	

			6. User clears the current inventory assigned on the modal popup and click on the “Edit inventory” button.			
TC012	Manage warehouse	Assign category to a storage bin	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User clicks on **newly created warehouse’s ID** to view the details. 4. In the warehouse details page, user selects the “storage bin” tab to view the list of storage bins in the warehouse 5. User clicks on the “Multi assign bin” to assign category to bins. 6. User selects a category and checks desired bin to assign. 7. User clicks on the “Assign” button.	The category of the selected bins is changed to the selected category.	Pass	
TC013	Login account	Reset password	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage User” tab and finds the newly created user with “MANAGER” role. 3. User takes note on the username and contact number.	A success notification will be shown.	Pass	

			4. User logouts and logins into the system using the username and contact number as the password. 5. User resets the password			
TC014	Manage cycle counting	Start a new cycle counting	1. User logins into the system using **the created manager-role user credentials (skip this step if you completed TC013)** 2. Note that there are **TWO** inventories listed in the View Inventory page, if no, kindly go back to **TC008** to create an inventory. 3. User clicks on the "Start Cycle counting" from the "Manage Cycle counting" tab dropdown on the top navigation bar. 3. User fills in all of the information in the form displayed. 4. User selects all inventories in the "Select Inventories" field. 4. User clicks the "Submit" button after completing the form. 5. User clicks the "Create" button in the summary modal popup.	A success notification will be shown stating that the cycle counting is started. Schedules will be created for all SKUs and assigned to the selected staffs evenly.	Pass	
TC015	Manage cycle counting	View all upcoming cycle counting SKUs	1. User clicks on the "View Cycle counting" from the "Manage Cycle counting" tab dropdown on the top navigation bar.	The list of schedules for the cycle counting and the settings will be displayed accordingly.	Pass	

			2. User clicks on the “View cycle count settings” button on the top right of the displayed table.			
TC016	View all staff	View all staff in a warehouse	1. User clicks on the “View Staff” tab on the top navigation bar. 2. User clicks on one of the staff ID from the list to view the staff details.	The list of staffs and their details will be displayed accordingly.	Pass	
TC017	Perform cycle counting	Perform cycle counting for an inventory	1. Repeat TC013 to login and reset password for the created **STAFF-role** user. 2. User clicks on the “Cycle Counting” tab from the top navigation bar. 3. User selects the “Upcoming” tab and views the list of upcoming cycle counting schedules 3. User chooses the first one in the list and clicks the “Count” button. 4. User enters the actual count on the modal popup 5. User clicks the “Create” button 6. User refresh the page. 7. Repeat step 3 to 6 to count another one.	The cycle counting record is created with success notification.	Pass	
TC018	Record inventory	Check in stock for an inventory	1. User clicks on the “Check In/Out Stock” tab from the top navigation bar. 2. User views the list of inventories displayed.	The selected inventory’s quantity on hand will be updated in real time without refreshing the page.	Pass	

			3. User chooses one of the inventories and clicks on the “Check In/Out Stock” button. 4. User selects the “Check In” tab in the modal popup. 5. User enters the quantity and remarks and clicks the “Confirm” button.			
TC019	Record inventory	Check out stock for an inventory	1. User clicks on the “Check In/Out Stock” tab from the top navigation bar. 2. User views the list of inventories displayed. 3. User chooses one of the inventories and clicks on the “Check In/Out Stock” button. 4. User selects the “Check Out” tab in the modal popup. 5. User enters the quantity and remarks and clicks the “Confirm” button.	The selected inventory’s quantity on hand will be updated in real time without refreshing the page.	Pass	
TC020	Manage cycle count approval report	Approve cycle count approval report	1. User logins into the system using **the created manager-role user credentials.** 2. User clicks on the “View reports” tab from the top navigation bar. 3. User selects the “Cycle Count Approval Report” tab and views the list of approval reports listed.	The summary report for the selected cycle counting will be generated with IRA calculated.	Pass	

			4. User clicks the “Approve” button for one of the approval reports. 5. User clicks the “Confirm” button in the confirmation modal. 6. User refresh the page.			
TC021	Manage cycle count approval report	Reject cycle count approval report	1. User clicks on the “View reports” tab from the top navigation bar. 2. User selects the “Cycle Count Approval Report” tab and views the list of approval reports listed. 3. User clicks the “Reject” button for one of the approval reports. 4. User checks the checkbox for reassigning staff to recount the SKU. 5. User clicks the “Confirm” button in the confirmation modal.	A success message will be shown to indicate the success rescheduling.	Pass	
TC022	View reports	View cycle counting summary report	1. User clicks on the “View reports” tab from the top navigation bar. 2. User selects the “Cycle Count Summary Report” tab and views the list of summary reports listed. 3. User clicks on the cycle count ID for one of the summary reports	The list of summary reports and their details will be displayed accordingly.	Pass	
TC023	View reports	View daily reports	1. User clicks on the “View reports” tab from the top navigation bar.	The list of daily reports and their details and stocks will be displayed accordingly.	Pass	

			2. User selects the “Daily report” tab and views the list of daily reports listed. 3. User clicks on the “View” button for one of the daily reports			
TC024	Manage inventory	Delete an inventory	1. User logins into the system using **ADMIN** account provided. 2. User clicks on the “Manage inventory” tab from the top navigation bar. 3. User **finds one of the newly created inventories** from the displayed list (can utilize the filter or search function) and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The selected inventory is successfully deleted from the warehouse.	Pass	
TC025	Manage user	Delete a user	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage user” tab from the top navigation bar. 3. User selects **one of the created users** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The status of the selected user will be set to “INACTIVE”.	Pass	
TC026	Manage category	Delete a category	1. User logins into the system using **ADMIN** account.	The selected category will be deleted from the list.	Pass	

			2. User clicks on the “Manage category” tab from the top navigation bar. 3. User **selects the newly created category** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.			
TC027	Manage warehouse	Delete a warehouse	1. User logins into the system using **ADMIN** account. 2. User clicks on the “Manage warehouse” tab from the top navigation bar. 3. User **selects the newly created warehouse** from the displayed list and clicks on the “Delete” button. 4. User clicks on the “Delete” button on the confirmation modal.	The selected warehouse is successfully deleted with a success message.	Pass	

APPENDIX D: User Satisfaction Survey Form

User Satisfaction Survey

1. I think that I would like to use this system/ website.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

2. I found the system/website unnecessarily complex.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

3. I thought the system/website was easy to use

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

4. I think that I would need the support of a technical person to be able to use this system/website.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

5. I found this system/website was easily moved through without a lot of backtracking or data re-entry.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

6. I thought there was too much inconsistency in this system/website.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

7. I would imagine that most people would learn to use this website very quickly.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

8. I found the system/website very awkward to use.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

9. I felt very confident using the system/website.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

10. I needed to learn a lot of things before I could get going with this system/website.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

11. What do you like best about the system

12. What do you like least about the system

13. If you were to describe this site to a colleague in a sentence or two, what would you say?

14. Do you have any other final comments or questions?