**StockFlow: Inventory Management System for Small Retail Businesses**

Name: Sarah Jane Liew Yun En

Coventry Student ID: 14008913

Supervisor Name: Ms Thirunageswari

BCTCUN

School of Engineering and Technology, INTI

International College Penang

8th November 2024

**6001CEM Declaration of Originality**

I declare that this project is all my own work and has not been copied in part or in whole from any other source except where duly acknowledged. As such, all use of previously published work (from books, journals, magazines, internet etc.) has been acknowledged by citation within the main report to an item in the References or Bibliography lists. I also agree that an electronic copy of this project may be stored and used for the purposes of plagiarism prevention and detection.

**Statement of copyright**

I acknowledge that the copyright of this project report, and any product developed as part of the project, belong to Coventry University. Support, including funding, is available to commercialise products and services developed by staff and students. Any revenue that is generated is split with the inventor/s of the product or service. For further information please see www.coventry.ac.uk/ipr or contact ipr@coventry.ac.uk.

**Statement of ethical engagement**

I declare that a proposal for this project has been submitted to the Coventry University ethics monitoring website (https://ethics.coventry.ac.uk/) and that the application number is listed below (Note: Projects without an ethical application number will be rejected for marking)

A close-up of a signature

Description automatically generated  
Signed: Date:

5 November 2024

Please complete all fields.

|  |  |
| --- | --- |
| First Name: | Sarah Jane |
| Last Name: | Liew Yun En |
| Student ID number | 14008913 |
| Supervisor Name | Ms Thirunageswari |

This form must be completed, scanned and included with your project submission to Turnitin. Failure to append these declarations may result in your project being rejected for marking.

**Table of Contents**

[ABSTRACT 1](#_Toc181996688)

[CHAPTER 1: INTRODUCTION 1](#_Toc181996689)

[1.1 Problem Statement 1](#_Toc181996690)

[1.2 Detailed Research Questions 1](#_Toc181996691)

[1.3 Objectives 1](#_Toc181996692)

[1.4 Clients, Audience, and Motivation 2](#_Toc181996693)

[1.5 Project Scope 2](#_Toc181996694)

[CHAPTER 2: LITERATURE REVIEW 4](#_Toc181996695)

[2.1 Inventory Management Systems 4](#_Toc181996696)

[2.1.1 Criteria for Inventory Management Systems 4](#_Toc181996697)

[2.1.2 Inventory Management Strategies 5](#_Toc181996698)

[2.1.2.1 Vendor Managed Inventory (VMI) 5](#_Toc181996699)

[2.1.2.2 Customer Managed Inventory (CMI) 5](#_Toc181996700)

[2.1.2.3 Joint Management Inventory (JMI) 6](#_Toc181996701)

[2.1.3 Comparison of Inventory Management Systems 6](#_Toc181996702)

[2.1.3.1 Side-by-Side Comparison of Features 1](#_Toc181996703)

[2.1.4 Inventory Replenishment Methods 1](#_Toc181996704)

[2.1.5 Forecasting Methods 1](#_Toc181996705)

[2.1.6.1 Estimating When to Reorder Goods 1](#_Toc181996706)

[2.1.6.2 Machine Learning Training Models for Demand Forecasting 2](#_Toc181996707)

[CHAPTER 3: RESEARCH METHODOLOGY 3](#_Toc181996708)

[3.1 Methodology Used 3](#_Toc181996709)

[3.1.1 Plan 4](#_Toc181996710)

[3.1.2 Design 4](#_Toc181996711)

[3.1.2.1 Use Case Diagram 7](#_Toc181996712)

[3.1.2.2 Entity Relation Diagram 8](#_Toc181996713)

[3.1.3 Develop 9](#_Toc181996714)

[3.1.4 Test 9](#_Toc181996715)

[3.1.5 Deploy 10](#_Toc181996716)

[3.1.6 Review 10](#_Toc181996717)

[CHAPTER 4: PROJECT IMPLEMENTATION 11](#_Toc181996718)

[4.1 Stock Management 11](#_Toc181996719)

[CHAPTER 5: TESTING & EVALUATION 12](#_Toc181996720)

[5.1 Functional Testing 12](#_Toc181996721)

[5.1.1 Integration Testing 12](#_Toc181996722)

[5.1.1.1 Sign Up 1](#_Toc181996723)

[5.1.1.2 Login feature 2](#_Toc181996724)

[5.1.1.3 Product details 4](#_Toc181996725)

[5.1.1.4 Purchase Orders 10](#_Toc181996726)

[5.1.2 System Testing 13](#_Toc181996727)

[5.2 Non-Functional Testing 13](#_Toc181996728)

[5.2.1 Security Testing 13](#_Toc181996729)

[5.2.1.1 Role Based Access Control 13](#_Toc181996730)

[5.2.1.2 Authentication 15](#_Toc181996731)

[CHAPTER 6: CONCLUSION 1](#_Toc181996732)

[APPENDIX A: GITHUB LINK 2](#_Toc181996733)

[APPENDIX B: CODE SCREENSHOTS 2](#_Toc181996734)

**List of Tables**

[Table 2.1.2.5..1 Comparison of IMS Features 1](#_Toc169820102)

**List of Figures**

[Figure 3.1.2.1.1 StockFlow Use Case Diagram 8](#_Toc169820093)

**List of Abbreviations**

**ML** Machine Learning

**IMS** Inventory Management System

**SKU** Stock Keeping Unit

**ERP** Enterprise Resource Planning

**VMI** Vendor Managed Inventory

**CMI** Customer Managed Inventory

**JMI**  Joint Management Inventory

**POS** Point of sale

**ERP**  Enterprise Resource Planning

**LT** Lead Time

ABSTRACT

Small retail businesses are typically managed by one store owner, therefore the assistance from an efficient inventory management system (IMS) is crucial. However, current IMS problems include inaccurate forecasting due to inaccurate data, overstocking and understocking of goods, and ordering errors. StockFlow aims to provide small retailers an IMS solution that efficiently manages stocks, reduces ordering errors, and forecasts stock replenishments to reduce overstocking and understocking. StockFlow will enable inventory data entry via barcode scanning and allow for data adjustments. Inventory will be monitored by their stock keeping units (SKU) for robust tracking, and low stock alerts will notify retailers for stock replenishment actions. Stocks can be ordered through the system with direct vendor interaction. Owners can view reports on inventory and demand forecasts for better decision making. Development for this project will follow the Agile methodology.

CHAPTER 1: INTRODUCTION

# 1.1 Problem Statement

Small retail businesses usually have its business management activities handled by one owner (Li, 2024, p. 221). Therefore, the need for an inventory management system (IMS) that efficiently tracks inventory and analyses business performance from sales analytics is crucial for strategizing for business improvement and making decisions (Oyewole *et al.,* 2024, p. 568). However, several inventory management issues still persist today (Maplesden, 2024, para. 1).

Inaccurate data caused by bad quality of data that is outdated, inaccurately recorded, or incomplete leads to poor demand forecasting (Oyewole *et al.,* 2024, p. 570). These errors result in insufficient stock levels, overstocking of goods (Li, 2024, p. 224), and ordering errors (Maplesden, 2024, para. 15).

StockFlow aims to provide small retail businesses a solution for keeping track of inventory items in an organised manner and reduce errors during the ordering of stock. This project also aims to enable effective forecasting of stock demands to aid the decision-making process when placing orders for stocking.

# 1.2 Detailed Research Questions

1) How will the system efficiently manage stocks and reduce ordering errors?

2) How will the system effectively forecast stock replenishment according to demands and reduce overstocking?

# 1.3 Objectives

1. To develop an inventory management system that can efficiently manage stocks and reduce ordering errors.
2. To develop an inventory management system that can effectively forecast stock demands for ordering and reduce understocking and overstocking.

# 1.4 Clients, Audience, and Motivation

The clients of this system are small physical retail businesses who seek to smoothly keep track of stocks and purchases accurately and to make inventory managerial decisions based on sales, demand, and trends in the market. With demand forecasting and analytics, store owners will be able to determine the stock variation and number of stocks to order.

The audience consists of two main entities: the retail stakeholders consisting of the owner and inventory management personnel, and the vendors suppling the products. This system is developed to assist the store owner in managing the business processes relating to the management of stocks and identifying the actions to be taken based on sales trends and demands. An intuitive interface enables inventory management personnel to perform in-store stock management processes in an organised and accurate manner, in which the status of stocks will accurately reflect on the system. Vendors will be able to view quotation requests and purchase orders stating quantities and product types requested by the store owner.

The motivation for this project is to address the challenges of frequent overstocking and understocking as well as the reoccurrence of ordering errors in inventory management for small retail businesses. Types of ordering errors areas include but are not limited to the ordering of wrong quantities and product types, ordering too early or late, and not ordering the correct items in season. These factors will bring sales costs, profit loss, a tarnished image of the business, and will limit cash flow – which are costs the owner will have to bear. This system will ease the burden of stock management operations and allow more bandwidth for store owners or business managers to strategize for management flow optimisation.

# 1.5 Project Scope

StockFlow will be developed as a web-based system aimed at small physical retail store businesses to make the inventory management process more efficient by adopting a clean, user-friendly intuitive design where functionalities are easily accessible. Users can enter inventory data by product and batch via manual entry of stock keeping units (SKU) or barcode scanning with ability to adjust data, group products, and track product returns.

There are 3 user-types in the system, the administrative (admin) role used by store owners, the regular staff (inventory management personnel), and the vendor role via the vendor portal. Admin and staff roles will have access to efficient tools for inventory operations such as entering, viewing, updating and deleting inventory details such as product names, product types, quantities, etc. Whereas the admin can view additional analytics, reports, and create purchase orders for new stock.

The system will make use of traditional forecasting techniques to forecast when to reorder inventory based on the buffer stock count. Seasonal trend forecasting will also provide insight to potential time-based market demands, and it will be done using a Machine Learning (ML) model. From there, store owners or business managers can know the type of products and quantities to order and make better decisions for the business’ inventory management processes. Specific reports like inventory balance and forecasted sales trends can be exported for auditing, stakeholder reporting, and other collaborative purposes.

When stocks fall below the buffer stock levels set by the admin, the system will give an alert and suggest an automatic creation of a quotation request or a purchase order to be sent to the vendor. Orders can be grouped, compiled, and sent to the respective vendors, where vendor contacts can be pre-configured along with customised notes of the type of products bought from them.

CHAPTER 2: LITERATURE REVIEW

# 2.1 Inventory Management Systems

**Inventory Management Systems (IMS) are designed to hold information about inventory and related management processes. These systems enable users to track and manage the flow of goods in the supply chain, from purchasing materials/goods to the point of sale (Munro, 2024). Without such a system, businesses will be operating on an ad hoc basis with stock and logistics records (e.g., goods purchasing, customer order fulfilment) scattered, making it difficult to know actual inventory levels and profitability, which can lead to profit loss (Munro, 2024). This also poses the risk of overstocking (having excess stocks) and understocking (having insufficient stocks).**

## ****2.1.1 Criteria for Inventory Management Systems****

**To ensure seamless operation,** there are several characteristics an efficient IMS should have**. According to Barry (2023), a robust database is essential for storing stock keeping units (SKU) that specify product type, variant (colour, model, etc.), and manufacturer. Other additional information tied with SKUs includes vendor and shipping details to trace products back to their source in case of recalls or defects. Modern IMS should also come equipped with demand forecasting capabilities to predict when items will go out of stock based on sales plans, rate of sale, and vendor lead times (the time taken for goods to be delivered by vendors from the time of purchase).**

**The inclusion of analytics and goods-purchasing functionalities are crucial for streamlining business planning and logistics operations for obtaining goods. To facilitate intuitive analysis and decision-making, users should be able to view and manage sales-channel-specific inventory levels and sales data (e.g., store owners can plan and prepare stock levels for upcoming promotions and assess sales performance and demand associated with the promotion). It is important that analytics provide actionable insights, not just data. For example, the system should identify best-selling products using inventory metrics like turnover by product/SKU and suggest which products to stock up on and which products to apply discounts to clear overstock.**

**Finally, the IMS should offer customizable views and support integrations with hardware and other software/services. Users should be able to tailor data presentation to fit inventory management practices and business planning style.** Support for barcode scanners ensures accurate stock tracking and efficient product searches. **And integration of the IMS with other enterprise resource planning (ERP) software/services enables a holistic approach, enhancing analytics with customer feedback, financial planning, and more.**

## Inventory Management Strategies

According to Li (2024), there are 3 strategies for inventory management: Vendor Managed Inventory (VMI), Customer Managed Inventory (CMI), and Joint Management Inventory (JMI). This section will briefly review these strategies and identify the one that will be used for this project.

### 2.1.2.1 Vendor Managed Inventory (VMI)

Vendors will create inventory management strategies and stock replenishment plans for the client (the retailer) according to sales market fluctuations and the monitoring of inventory levels and sales information shared by clients. With this data, vendors can determine the batch size and the date to deliver by taking lead times and estimated shipping delays into account. Since vendors are more knowledgeable on the manufacturing process, it is easier to provide accurate estimates of goods arrival, reducing ordering errors. This method reduces the risk of overstocking and understocking by ±30% (Li, 2024, p. 222).

### 2.1.2.2 Customer Managed Inventory (CMI)

Unlike Vendor Managed Inventory (VMI), the responsibility for creating inventory management strategies and stock replenishment plans is done by the vendor’s clients (a.k.a. the retailer). This includes tasks like inventory tracking, goods-replenishment ordering, updating inventory data in the system and analysing demand forecasts (Cash Flow Inventory, 2023). Since retail businesses are closest to the consumer market, many in this sector believe that this method leads to a better understanding of market demands, enabling retailers to make more informed inventory decisions.

### 2.1.2.3 Joint Management Inventory (JMI)

Joint Management Inventory (JMI) is a collaborative approach combining manufacturing expertise with direct insights to the consumer market via point of sale (POS) for improved inventory management and faster response to market trends (Li, 2024, p. 222). Vendors and retailers exchange information such as real-time inventory levels, sales data, forecasts, and more.

StockFlow will use JMI as it offers an optimised inventory management strategy that reduces ordering errors, overstocking, and understocking issues.

## Comparison of Inventory Management Systems

This section compares the general features of 3 existing IMS: Zoho Inventory, Odoo, and ECOUNT, and will be put side by side for comparison against StockFlow.

### 2.1.3.1 Side-by-Side Comparison of Features

Table 2.1.2.5.. Comparison of IMS Features

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name of System** | **Zoho** | **Odoo** | **ECOUNT** | **StockFlow (current project)** |
| **Type of Input** | | |  | |
| **Barcode/QR scan** | ✔️ | ✔️ | ✔️ | ✔️ |
| **Manual entry** | ✔️ | ✔️ | ✔️ | ✔️ |
| **Product details adjustment** | ✔️ | ✔️ | ✔️ | ✔️  With mandatory logs stating reason for adjustment |
| **Inventory Tracking & Management** | | |  | |
| **Track product by SKU** | ✔️ | ✔️ | ✔️ | ✔️ |
| **Batch tracking** | ✔️ | ✔️ | ✔️ | ✔️ |
| **Product variant grouping** | ✔️ | ❌ | ❌ | ✔️ |
| **Product category** | ✔️ | ✔️ | ✔️ | ✔️ |
| **Product return management** | ✔️ | ✔️ | ✔️ | ✔️ |
| **Item lifespan/expiry date tracking** | ✔️ | ✔️  With additional reminders on expiration date & picking priority prompts | ✔️ | ✔️ |
| **Automatic reminders/alerts on expiry date** | ✔️ | ✔️ | ✔️ | ✔️ |
| **Setting buffer stock level** | ✔️ | ✔️ | ✔️ | ✔️ |
| **Low stock alerts and reporting** | ✔️ | ✔️ | ✔️     * Visual indicators in product list for inventory count below minimum stock level *(cell highlighted in red)*      * Warning message | ✔️ |
| **Low stock control** | Automatically prepare purchase order for products, manual review before confirming order | Send automated quotation request to vendor | * Option to block saving a sales order form when a product in the list is low on stocks | * In-system notification on low stock with additional functionality to notify via email |
| **Automatic reordering of stock** | ✔️  Done by Zoho Marketplace | ✔️ | ❌ | ✔️     * Option to automatically compose a quotation request to send to vendor |
| **Users** | | | |  |
| **Store owner** | ✔️ | ✔️ | ✔️  User access authorisation configuration | ✔️  User access authorisation to access all features |
| **Inventory Management Personnel** | ✔️ | ✔️ | ✔️  User access authorisation configuration | ✔️  User access authorisation  *(can access everything except purchase ordering and reporting features)* |
| **Vendor** | ✔️  Vendor can view purchase order, invoices and payment tracking | ❌ | ✔️  Can view purchase order and order fulfilment details through integration with ECOUNT’s Vendor Portal | ✔️  Can view purchase order, order fulfilment receipt upload, and status of order receival |
| **Inventory Ordering** | | | |  |
| **Communication with vendor via system** | ❌ | ✔️     * Integration with Odoo’s Purchasing app to send quotation requests via email      * Automated follow-ups with vendors like purchase orders acknowledgements and receipt schedule confirmation (verifying the arrival date of goods) | ✔️  Vendors can view purchase orders through ECOUNT’s Online Ordering System | ✔️ |
| **Generating purchase order** | ✔️ | ✔️  Done via integration with Zoho Books | ✔️ | ✔️ |
| **Goods arrival lead time factoring** | ✔️ | ✔️ | ❌ | ✔️ |
| **Check arrival date of stock** | ✔️ | ✔️ | ❌ | ✔️ |
| **Reporting** | | | |  |
| **Inventory balance report** | ✔️ | ✔️ | ✔️ | ✔️ |
| **Inventory Forecasting** | ❌ | ✔️  Forecast out-of-stock dates with customer orders, shipment dates, and manufacturing orders *(done with 3rd party integration)* | ❌ | ✔️ |
| **Analytics show best-selling products** | ✔️ | ❌  Sales analytics are done on other Odoo apps | ❌ | ✔️ |
| **Sales/demand forecasting** | ✔️ | ❌ | ✔️ | ✔️ |
| **Exportable reports** | ✔️ | ✔️ | ✔️ | ✔️ |
| **User experience** | | | | |
| **Suitable for small physical retail businesses** | ❌ | ❌ | ❌ | ✔️ |

Zoho, Odoo and ECOUNT are established service providers offering effective enterprise resource planning (ERP) solutions. While many of the features help reduce data inaccuracy, overstocking, understocking, and ordering errors, these capabilities are not offered together in the same system. Furthermore, these 3 systems are not well-suited for small physical retail businesses as they were built with large-scale warehousing management in mind, including other features that are redundant to smaller scale retailers. This makes the systems frustrating to use and may result in inefficient execution of inventory management.

StockFlow consolidates the scattered inventory management features into one system, simplifying the use for the targeted client while remaining as an efficient solution for the business.

## Inventory Replenishment Methods

**Safety Stock Reordering**

The concept of safety stock (buffer stock) is the rule of having a fixed quantity of stocks in the inventory as spares in the case of sudden surges in demand or unexpected vendor shipping delays. Safety stock reordering can be calculated as follows:

Safety Stock (SS) = Average Sales (AS) x Safety Lead Time

Reorder Point (RP) = Safety Stock (SS) + Average Sales (AS) x Total Lead Time

Total Lead Time = Order Lead Time + Delivery Lead Time

(Thieuleux, 2023)

**Min/Max Reordering**

The Min/Max reordering method consists of determining the minimum and maximum stock levels. It can be calculated as follows:

Minimum Level (MIN) = Average Sales (AS) x (Lead Time + Safety Lead Time)

Maximum Level (MAX) = Average Sales (AS) x Fixed Ratio (between 1.5 and 2.5)

(Thieuleux, 2023)

Safety stock reordering is more accurate than the MIN/MAX method, therefore StockFlow will use the safety stock reordering method.

## Forecasting Methods

### 2.1.6.1 Estimating When to Reorder Goods

**Lead Time forecasting**

Lead Time specifies the time taken for purchased goods to arrive its destination from the point of order confirmation (Reid, 2023). Lead Time consideration is crucial for ensure timely delivery of goods before inventory runs out. There are 3 types of Lead Time:

* + **Material Lead Time:** the time taken for vendors to deliver raw materials to the manufacturer
  + **Production Lead Time:** the duration between the submission of a manufacturing order and the moment the finished product is temporarily stored within the production plant or shipped out
  + **Customer Lead Time:** the time taken for the source company (the manufacturer or vendor) to produce the goods and/or deliver to the customer

In this case, Customer Lead Time is relevant to the scope. There are two ways to calculate Lead Time.

***Method 1***

Lead Time (LT) = Order Delivery Date – Order Request Date (Reid, 2023)

***Method 2***

Lead Time (LT) = Supply Delay + Reordering Delays (Reid, 2023)

Due to limited time for the project development, StockFlow will only incorporate Method 1 for calculating Lead Times.

### 2.1.6.2 Machine Learning Training Models for Demand Forecasting

**Prophet**

Prophet is an open-source forecasting tool developed by Meta (formerly Facebook) implemented in R and Python. It forecasts time series data which can account for non-linear trends paired with yearly, weekly, and daily seasonality along with holiday seasons. Missing data and trend spikes are handled by filling in the gaps with predicted results according to seasonal trends.

This model requires training directly on the machine before deploying to the web. The best language to train Prophet on is Python, as compared to R due to more community support.

**TensorFlow**

TensorFlow is an open-source ML system that hosts the entire Machine Learning cycle from the building and training of models to their deployment. TensorFlow offers extensive libraries and APIs tailored to the project needs such as TensorFlow Lite (mobile and Edge devices), TensorFlow.js (training and running of models directly in the web browser) and tf.data (preprocessing data and data input pipelines for ML models). TensorFlow offers time series forecasting which builds Deep Learning models such as Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN).

StockFlow will use Prophet as its forecasting tool due to its beginner friendliness.

CHAPTER 3: RESEARCH METHODOLOGY

# 3.1 Methodology Used

This project will follow the basic Agile methodology for the development process. Agile is a modern methodology that focuses on development rather than strict documentation, with its notable characteristic being an iterative development process. Unlike the rigidness of the traditional Waterfall model, Agile allows for revisiting initial plans and making changes for adaptability to changing requirements.

## 3.1.1 Plan

The first phase of the Agile methodology defines the project plan, outlining the problems the project will solve, the motivation behind the project initiation and the scope of the project. This is also where research on the project criteria and adopted processes within the scope is to be done for the project to work.

The conceptualisation of this project begins with sourcing for existing problems in inventory management systems through Google Search and Google Scholar. After gaining insight and elaborate consideration, the objectives of the project are formed and the requirements gathering phase will commence. This includes gathering criteria of an IMS and inventory management methods and strategies practiced in the industry

## 3.1.2 Design

The analysis and design phase expands on the planning phase with the gathering of technical requirements for the project such as information gathering the technologies to use for development and an understanding of how to use such technologies. The product backlog, project architecture, and prototyping will also be formed, and the clients of the project are defined.

The comparison of existing IMS features is being done during this phase to analyse the functionalities offered in the market as well as brainstorming solutions to bring valuable improvement to the potential market. With this, the product backlog is formed whilst doing deeper research of technologies required for the development.

Below is the general system backlog as of the proposal stage:

***Input:***

barcode scanning

manual entry

item adjustments + mandatory logs reasoning adjustment

***Tracking:***

Serial number tracking, batch tracking & SKU tracking

tracking date of item & batch entry

item variant grouping

item category (user needs to name the category)

item returns (scan back into system, increase inventory count)

lifespan/expiry date tracking & reminder

low stock tracking

buffer stock setting

visual cue of low stock

in-system notification + option to notify via email

Option to automatically compose quotation request to vendor

Products with the same targeted vendor will be automatically grouped together before sending off

* History of transactions/inventory movement
* Transaction logging (log every action e.g. made changes to TSHIRT\_RED)

***Inventory Ordering:***

* Implement estimated lead time for ordering (from inventory forecasted date)
* view estimated arrival of goods
* vendor contacts when creating quotation request/purchase order (can write note to state what kind of product the retailer usually buys from the specific vendor)

***Analytics:***

informative dashboard to view inventory activity and general analytics at a glance

inventory forecasting: calculating when to start reordering from buffer stock setting, min-max rules

view sales trends

***Reporting:***

inventory balance report

sales/demand forecasting report

***Generate docs:***

* purchase order (to send to vendor)
* reports

***Vendor interaction:***

quotation request/purchase order email to vendor

vendor portal: view purchase orders, upload order fulfilment receipt, and status of order receival

In the span of the proposal period, a Use Case diagram is created to visualise and strategize for the general function of the system.

### 3.1.2.1 Use Case Diagram

A screen shot of a cell phone

Description automatically generated

Figure 3.1.2.1.. StockFlow Use Case Diagram

### 3.1.2.2 Entity Relation Diagram

A black background with white text

Description automatically generated

Figure 3.1.2.2.. StockFlow Entity Relation Diagram

## 3.1.3 Develop

The development phase (also known as the implementation phase) is the commencement of the development work where the project will be developed according to project plan and system design. Project timelines will keep the development on track and ensure that the product is produced by a set date.

The Scrumban framework will take place here, implementing sprint reviews with the supervisor, personally reviewing the project progress and solving problems, as well as making use of the Kanban board to ensure development progress without being overloaded with tasks.

The following technologies will be used for this project:

**Frontend**

* HTML
* PHP
* JavaScript

**Backend**

* Python (for Prophet model building and training)
* Relational Database Management System (RDBMS)

The training datasets for the forecasting model will be retrieved from Kaggle.com.

## 3.1.4 Test

The testing phase focuses on testing the workability of the project deliverable throughout the development process up to the point nearing to deployment. This project will include 3 types of tests: unit testing to ensure each individual component of the system is functional, system testing to test its end-to-end functionality, and usability testing to test its ability to allow users to efficiency complete tasks and to fare its user satisfaction. Unit testing and system testing will be done throughout development, with emphasis on the final stages before usability testing and deployment. Usability testing will be done with the audiences of the system to obtain feedback on the system.

## 3.1.5 Deploy

This is when the project deliverable is ready to be released to the clients to use. The deployment for this project will be the final product that will be presented to the examiners for review, as well as a potential continuation of the project for proper release.

## 3.1.6 Review

This phase happens during development and after deployment where the project deliverable undergoes reviewing involving strategies for improvement (according to user feedback if after deployment). The final phase for this project will be the listing of future improvements upon evaluation of user feedback, examiner feedback, and personal review

CHAPTER 4: PROJECT IMPLEMENTATION

This section covers the features and functionalities implemented in the system. The main features available as of the documentation of this project include the implementation of stock management and purchase ordering features.

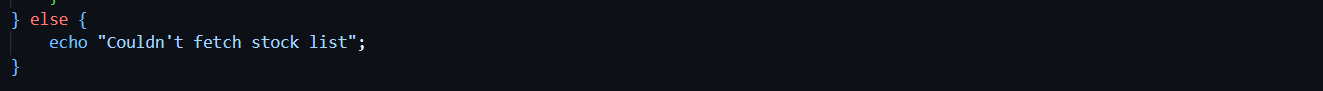
# 4.1 Stock Management

A screenshot of a computer

Description automatically generated

Figure 4.1.. Manage Stocks page

The staff and admin roles manage stocks through the *Manage Stocks* page where the interface allows the user to navigate to an *Add Item* interface to manually input product details, edit item add and edit item categories and/or delete products. The list of stocks added will be displayed in the main interface *(as shown in Figure 4.1.1)*, displaying the stock keeping unit (SKU), product name, serial number, product brand, product type, unit price, number of stock, and the vendor that distributed the stock. The user can filter through the list of items using the search bar where rows will be returned based on the product detail that is the closest matching to the search query.

A computer screen shot of a program code

Description automatically generatedA screen shot of a computer

Description automatically generatedA computer screen shot of a computer program

Description automatically generated

Figure 4.1.. Search query code snippet

## 4.1.1 Add Product

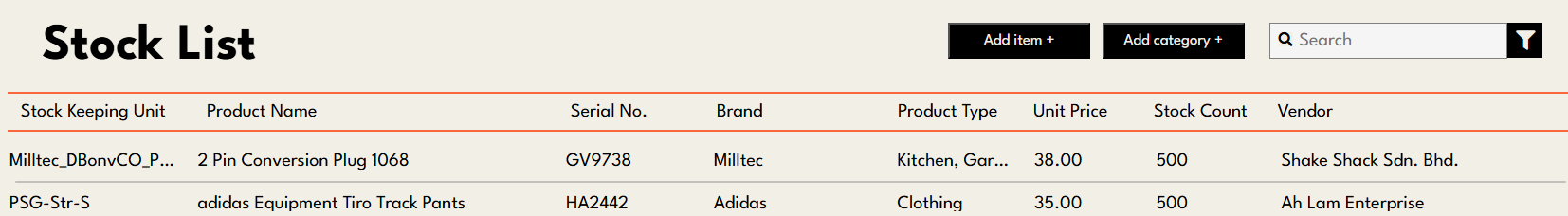


Figure 4.1.1.. Add Item button in Manage Stock interface

The user can navigate to the interface for adding products by clicking on the *Add Item* button.

A screenshot of a computer

Description automatically generated

Figure 4.1.1.. Add New Item interface

The *Add New Item* interface lets the user manually enter the product details, where all except the serial number, product type and product category fields are mandatory. The Product Types and Category fields are dropdowns which options can be configured by the user. The textarea is meant to be a field for the mandatory logging message that the user needs to enter before saving the item. Once the product is saved, the data will be saved to the database and will appear as part of the stocklist in the *Manage Stocks* interface.

A screenshot of a computer program

Description automatically generated

Figure 4.1.1.. Code snippet for inserting product details to the product table

A screen shot of a computer code

Description automatically generated

Figure 4.1.1.. Code snippet for inserting distributing vendor name to the vendor\_product linking table table

A screen shot of a computer code

Description automatically generated

Figure 4.1.1.. Code snippet for inserting product type into Product\_Type table

A computer screen with text on it

Description automatically generated

Figure 4.1.1.. Code snippet for inserting the log into Activity\_Logs table

Once the activity log is inserted, the user will be redirected to *Manage Stocks*.

## 4.1.2 Edit Product

A screenshot of a computer

Description automatically generated

Figure 4.1.2.. Button to enter Edit Item interface

The user can edit a product’s details by hovering over the product and clicking on the edit button to navigate to the *Edit Item* interface. A link carrying the target product ID is attached to the button to ensure that the system retrieves the correct product to edit.

A screen shot of a computer program

Description automatically generated

Figure 4.1.2.. Code snippet for navigating to the Edit Item interface with the productID atttached

A screenshot of a computer

Description automatically generated

Figure 4.1.2.. Edit Item interface with item details being edited

A screenshot of a computer program

Description automatically generated

Figure 4.1.2.. Code snippet retrieving target product details from database

The target product’s details are fetched from the database and auto populated into the respective fields where the user can directly make changes. In this example the SKU is edited from ‘Panasoni\_DBatt-\_AA-2’ to ‘PANASONIC BATTERY’, its stock count edited from 500 to 485, and product type changed from ‘Kitchen, Garden & Pets’ to ‘Cleaning & Household’. The user will need to log the changes before saving.

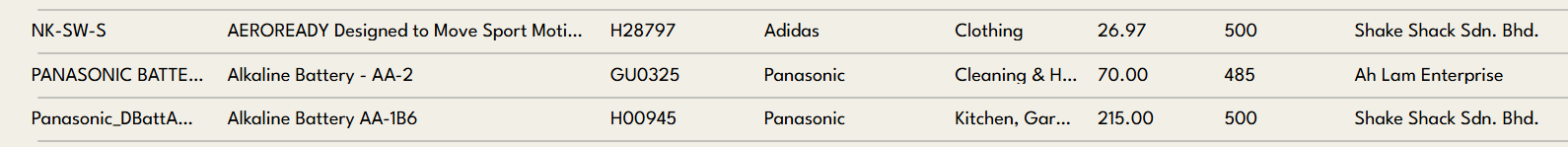


Figure 4.1.2.. Updated product details

After updating, the new product details will be displayed in the stocklist.

## 4.1.3 Deleting Product

A screenshot of a computer

Description automatically generated

Figure 4.1.3.. Delete Item button

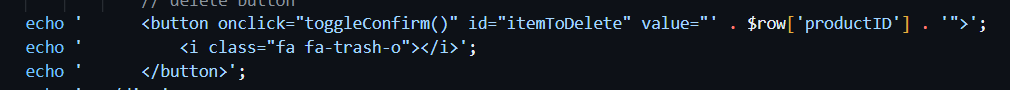


Figure 4.1.3.. Code snippet of button carrying produc tID

To delete a product, the user can hover over the product to find the delete button next to the edit button. The button carries the product ID in its button id and will trigger a pop-up panel asking the user to confirm deletion.

A screenshot of a computer

Description automatically generated

Figure 4.1.3.. Confirm delete panel

The product ID is passed from the delete button’s id to the ‘Yes’ button, that will send the product ID as the operation parameter and trigger the delete sequence when the user clicks on it.

A computer screen shot of a program code

Description automatically generatedA screen shot of a computer code

Description automatically generated

Figure 4.1.3.. Code snippet for passing of product ID to backend

A screen shot of a computer program

Description automatically generated

Figure 4.1.3.. Code snippet for the deletion sequence

## 4.1.4 Add and Edit Product Category

A screenshot of a computer

Description automatically generated

Figure 4.1.4.. Add/Edit product category button

The user can add and edit product categories by clicking on this button which will open a panel for the process.

A screenshot of a computer

Description automatically generated

Figure 4.1.4.. Add/Edit product category panel

The top panel allows the user to enter a new category name and save the new category into the database. When the product has been inserted into the database, a status message will appear, updating the user if the category was saved.

A screenshot of a computer

Description automatically generated

Figure 4.1.4.. Status message alert for adding new category process

A computer screen shot of a program code

Description automatically generatedA screen shot of a computer code

Description automatically generated

Figure 4.1.4.. Code snippet for handling the adding of category name

A screenshot of a computer program

Description automatically generated

Figure 4.1.4.. Code snippet for inserting category name into the database

A screenshot of a computer

Description automatically generated

Figure 4.1.4.. Select to edit category name

To edit the category name, the user can select the category from the dropdown and can choose between renaming and deleting the category.

A screenshot of a computer

Description automatically generated

Figure 4.1.4.. Editing category name

The user can enter the new name for the category in the text input, and hit the save button to save the changes.

A computer screen shot of a program code

Description automatically generatedA screen shot of a computer code

Description automatically generated

Figure 4.1.4.. Code snippet to pass new category name to backend

A screenshot of a computer program

Description automatically generated

Figure 4.1.4.. Backend file for handling insertion of new category name

A screenshot of a computer

Description automatically generated

Figure 4.1.4.. Updated category name

The updated category name will show on the dropdown in the frontend.

To delete the product category, the user can click on the delete button after selecting the dropdown value to reveal a panel asking the user to confirm delete.

A screenshot of a computer

Description automatically generated

When the user clicks ‘Yes’ the following code will execute the category deletion.

A computer screen shot of a program code

Description automatically generatedA screen shot of a computer code

Description automatically generated

Figure 4.1.4.. Frontend code snipet for handling category deletion event

A computer screen shot of a program code

Description automatically generated

Figure 4.1.4.. Backend code snippet for handling category name deletion

# 4.2 Purchase Orders

The purchase order function allows the user to generate purchase orders for vendors’ reference when ordering new stock.

A screenshot of a computer

Description automatically generated

Figure 4.1.4. Purchase Orders interface

The left panel holds the purchase orders that were made previously, while the main pane on the right is the space where the purchase order will be displayed when the user selects the purchase order from the left panel.

CHAPTER 5: TESTING & EVALUATION

In this section, the system will undergo two functional testing and one non-functional testing. Functional testing refers to a type of software testing that tests the developed system against its technical, functional requirements and specifications set for the system. Examples of functional testing include unit testing, integration testing, and system testing. On the other hand, non-functional testing tests the behaviour of the system according to users’ expectations for the system. Examples of non-functional testing include security testing, usability testing, and stress testing.

StockFlow’s functional testing will include integration testing and system testing while going for security testing for its non-functional testing.

# 5.1 Functional Testing

The following section will share results from the integration test and the system test.

## 5.1.1 Integration Testing

Integration testing assesses the interaction of the system’s front-end with the backend and how the final output measures up to its expected result. This section will test the integrations available in StockFlow and will display the results for each feature.

### 5.1.1.1 Sign Up

The sign-up page handles the registration of new users into the system, where the system interacts with the backend for new credential verification before insertion into the database. The following shows the test results for the available integration.

Table 5.1.1.1.. Integration Testing for Sign Up feature

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test scenario** | **Code snippet** | **Expected result** | **Result** | **Status** |
| Saving to database where all data is valid |  | User is redirected to login page |  | PASS |
| When invalid shop is entered |  | User is not permitted account creation and is required to re-enter correct information |  | PASS |
| When password does not meet minimum requirement |  | User is not permitted account creation and is required to re-enter correct information |  | PASS |
| When password confirm does not match first password set |  |  |  | PASS |

### 5.1.1.2 Login feature

The login page is responsible for ensuring that only registered and authorised users are allowed access to the system. The login feature takes the user input and compares it against the user details and credentials saved in the system, and redirects users to the landing page as soon as the input data matches the saved credentials.

Table 5.1.1.2.. Integration Testing for login feature

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test scenario** | **Code snippet** | **Expected result** | **Result** | **Status** |
| When the user’s credential input match the credentials held in the database |  | User is redirected to landing page |  | PASS |
| When user enters an email that does not exist |  | User is not permitted access |  | PASS |
| When user enters the wrong password |  | User is not permitted access |  | PASS |

### 5.1.1.3 Product details

This section tests on product details’ interaction with the backend, which includes the CRUD operations regarding product data.

Table 5.1.1.3.. Integration Testing on product details

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test scenario** | **Code snippet** | **Expected result** | **Result** | **Status** |
| Entering new product data |  | Product details will be inserted into the database |  | PASS |
| Viewing lists of products within the store |  | Stocklist will display all stocks from the store |  | PASS |
| Editing product details |  | Product data will be populated into the interface and edited product data will be updated in the database |  | PASS |
| Deleting product |  | Product will be removed from database |  | PASS |
| Creating new product category |  | A new product category will be created and saved to database |  | PASS |
| Viewing new product category |  | Product categories are retrieved from database and displayed In the respective dropdowns |  | PASS |

The insertion of product details successfully save all details set by the user (including optional fields where items may not be inserted). Products details can be retrieved, extracted, and displayed in the respective interfaces where the product details are needed. Products can also be deleted from the *Manage Stocks* interface. This means that the basic product CRUD functions have been fulfilled. Product categories can be created, viewed, updated, and deleted, making it fully functional. However, there is currently no medium for creating or editing product types, only reading. Therefore operation on product details are mostly functional apart from 1 feature that has not been set up.

### 5.1.1.4 Purchase Orders

This section tests the currently available backend interaction on items regarding purchase orders.

Table 5.1.1.4. Integration Testing on purchase orders

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test scenario** | **Code snippet** | **Expected result** | **Result** | **Status** |
| Creating new purchase order | Frontend:    Backend: | Purchase order details inserted into database |  | PASS |
| Viewing purchase order via system |  | Purchase order details can be extracted from database and displayed in interface |  | PASS |

The purchase order data can be inserted correctly and retrieved and displayed in the correct order. However, the total amount is not calculated and stored correctly, as a result, it captures the first selected extended price as the total amount in the database and displays accordingly. The list of orders displayed in the purchase order is also facing issues with displaying duplicates of certain order data. Purchase orders are only subjected for creating and reading as of the time of project completion. Therefore, the purchase order feature is not completely functional.

## 5.1.2 System Testing

System testing involves the testing of the system’s

# 5.2 Non-Functional Testing

The following section records the test results for security testing

## 5.2.1 Security Testing

This security test assesses the security features implemented in StockFlow and how it handles user authorisation and authentication.

### 5.2.1.1 Role Based Access Control

StockFlow implements the separation of access privileges and interfaces based on the user’s account type. The following test demonstrates the behaviour of the system when handling the use cases.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test scenario** | **Code snippet** | **Expected result** | **Result** | **Status** |
| Admin account logging into the system | Example page handling interface component to call: | The loaded interface will be the interface designed for the *admin role* | Admin account created:    Logged into the admin account:    Admin account has access to Reporting & Analytics | PASS |
| Staff account logging into the system | Example page handling interface component to call: | The loaded interface will be the interface designed for the *staff role* | Staff account created:  Logged into the staff account:    Staff accounts do not have Reporting & Analytics | PASS |

### 5.2.1.2 Authentication

This feature is in the login page where the user is verified before allowed access to the system.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test scenario** | **Code snippet** | **Expected result** | **Result** | **Status** |
| Entering an email that does not exist | A computer screen shot of white text  Description automatically generated | User is not permitted access to system |  | PASS |
| Entering the wrong password | A computer screen with white text  Description automatically generated | User is not permitted access to system |  | PASS |
| Clicking on sign up without password entry | Frontend:    Backend: | User is not permitted access to system |  | PASS |

Although implementation of security methods are simple and low-level, it ensures access to only authorised users. Users whose roles are not allowed privilege to certain features are guaranteed not to see the interface that is not meant for the user

CHAPTER 6: CONCLUSION

The features that were successfully implemented in StockFlow include the basic adding, viewing, editing, and deleting of stock and corresponding stock details, and the creating and viewing of purchase orders in its basic view. In this iteration of the project, stocks can only be added via manual insertion and purchase orders can only be viewed in a basic view format through the population of tables within the system. Currently, the limited features and interfaces are only created for the admin and staff roles.

As the purchase order feature is in its incomplete form and the sales functionality not yet configured, sales forecasting and prediction of stock order dates cannot be set up. This also spilled over to the main Dashboard where the stock status and notification were intended to be displayed based on stock predictions according to the proposed forecast.

In conclusion, StockFlow was unable to meet the proposed objectives from the initial planning stage due to the lack of functional features for efficient stock management and the lack of parameters available for sales forecasting. StockFlow is a major work in progress with the main features yet to be implemented and the forecasting feature to be included in future development of this project.

# APPENDIX A: GITHUB LINK

<https://github.com/SJ-earworm/StockFlow-Inventory-Management-System---SarahJaneLiewYunEn_FinalYearProject.git>

# APPENDIX B: CODE SCREENSHOTS

gai

**REFERENCES**

Maplesden, P. (2024) 20 common problems with Inventory Management: TechTarget, ERP. Available at: <https://www.techtarget.com/searcherp/tip/Common-problems-with-inventory-management> (Accessed: 14 May 2024).

Li, H. (2024) ‘Research on the problems and countermeasures of Minimart Inventory Management from The Perspective of Supply Chain’, Highlights in Business, Economics and Management, 24, pp. 221–226. doi:10.54097/56xexx36.

Adedoyin Tolulope Oyewole *et al.* (2024a) ‘Reviewing predictive analytics in supply chain management: Applications and Benefits’, *World Journal of Advanced Research and Reviews*, 21(3), pp. 568–574. doi:10.30574/wjarr.2024.21.3.0673.

Munro, O. (2024) *Inventory Management Systems*, *Unleashed Software*. Available at: <https://www.unleashedsoftware.com/inventory-management-guide/inventory-management-systems> (Accessed: 17 June 2024).

Barry, B. (2023) *6 features to look for in an inventory management system*, *F. Curtis Barry & Company*. Available at: <https://www.fcbco.com/blog/top-6-features-to-look-for-in-an-inventory-management-system> (Accessed: 17 June 2024).

Bloomenthal, A. (no date) *Sku: What it is and how it works*, *Investopedia*. Available at: <https://www.investopedia.com/terms/s/stock-keeping-unit-sku.asp> (Accessed: 17 June 2024).

Awati, R. and Essex, D. (2023) *What is vendor-managed inventory (VMI)?: Definition from TechTarget*, *ERP*. Available at: <https://www.techtarget.com/searcherp/definition/vendor-managed-inventory-VMI> (Accessed: 18 June 2024).

Cash Flow Inventory (no date) *Customer managed inventory: What it is, how it works*, *Cash Flow Inventory*. Available at: <https://cashflowinventory.com/blog/customer-managed-inventory/> (Accessed: 18 June 2024).

Laoyan, S. (2024) *What is agile methodology? (a beginner’s guide) [2024] • asana*, *Asana*. Available at: <https://asana.com/resources/agile-methodology> (Accessed: 15 June 2024).

Martins, J. (2024) *Understanding the iterative process, with examples*, *Asana*. Available at: <https://asana.com/resources/iterative-process> (Accessed: 21 June 2024).

Atlassian (no date) *Scrumban: Mastering Two agile methodologies*, *Atlassian*. Available at: <https://www.atlassian.com/agile/project-management/scrumban#:~:text=Agile%20project%20management.-,What%20is%20Scrumban%3F,work%2Din%2Dprogress%20limitations>. (Accessed: 10 June 2024).

*Online inventory management software: Zoho inventory* (no date) *Inventory management software | Online inventory management for US businesses - Zoho Inventory*. Available at: <https://www.zoho.com/inventory/> (Accessed: 13 June 2024).

*The #1 open source inventory management* (no date) *Odoo*. Available at: <https://www.odoo.com/app/inventory> (Accessed: 14 June 2024).

Ecount (no date) *Inventory management system with various reports for faster decision making*, *ECOUNT*. Available at: <https://www.ecount.com/my/ecount/product/inventory_inventory-management-system?p=Main> (Accessed: 14 June 2024).

Reid, H. (2023) *Lead time: Definition, examples, and formula*, *DCL Logistics*. Available at: <https://dclcorp.com/blog/inventory/lead-time/#:~:text=There%20isn’t%20one%20way,from%20the%20order%20delivery%20date.&text=When%20dealing%20with%20inventory%20management,delay%20and%20the%20reordering%20delay>. (Accessed: 19 June 2024).

Thieuleux, E. (2023) *Optimizing inventory : Min/max method or safety stock ?*, *AbcSupplyChain*. Available at: <https://abcsupplychain.com/inventory-optimization-min-max-method-or-safety-stock/#:~:text=The%20Min%2FMax%20method%20is,minimum%20and%20maximum%20stock%20levels.&text=Maximum%20level%20(MAX)%3A%20The,usually%20between%201.5%20and%202.5>. (Accessed: 21 June 2024).

*Forecasting at scale.* (no date) *Prophet*. Available at: <https://facebook.github.io/prophet/> (Accessed: 21 June 2024).

*Tensorflow* (no date) *TensorFlow*. Available at: <https://www.tensorflow.org/> (Accessed: 21 June 2024).

*Time Series forecasting  :  Tensorflow Core* (no date) *TensorFlow*. Available at: <https://www.tensorflow.org/tutorials/structured_data/time_series> (Accessed: 21 June 2024).

*Unit testing - javatpoint* (no date) *www.javatpoint.com*. Available at: <https://www.javatpoint.com/unit-testing> (Accessed: 21 June 2024).

*System testing - javatpoint* (no date) *www.javatpoint.com*. Available at: <https://www.javatpoint.com/system-testing> (Accessed: 21 June 2024).

*What is usability testing? - updated 2024* (2024) *The Interaction Design Foundation*. Available at: <https://www.interaction-design.org/literature/topics/usability-testing> (Accessed: 21 June 2024).

*Quote – purchase order – sales order – invoice – what is the difference?* (2023) *VAR Sales Management Blog: News, Tips & Many More By VARStreet*. Available at: <https://blog.varstreetinc.com/quote-purchase-order-sales-order-invoice-what-is-the-difference/#:~:text=A%20buyer%20sends%20a%20request,terms%20stated%20in%20the%20quote>. (Accessed: 07 September 2024).

*MySQL - Decimal* (no date) *Tutorialspoint*. Available at: <https://www.tutorialspoint.com/mysql/mysql-decimal.htm> (Accessed: 17 September 2024).

*MySQL 8.4 Reference Manual :: 13.2.5 automatic initialization and updating for timestamp and DATETIME* (no date) *MySQL*. Available at: <https://dev.mysql.com/doc/refman/8.4/en/timestamp-initialization.html> (Accessed: 17 September 2024).

GeeksforGeeks (2024) *How to insert form data into database using php ?*, *GeeksforGeeks*. Available at: <https://www.geeksforgeeks.org/how-to-insert-form-data-into-database-using-php/> (Accessed: 19 September 2024).

*PHP mysql insert query* (no date) *TutorialRepublic*. Available at: <https://www.tutorialrepublic.com/php-tutorial/php-mysql-insert-query.php> (Accessed: 19 September 2024).

Code Signing Store (2021) *What is the best hashing algorithm?.* Available at: <https://codesigningstore.com/what-is-the-best-hashing-algorithm> (Accessed: 30 September 2024).

*W3schools.com* (no date) *W3Schools Online Web Tutorials*. Available at: <https://www.w3schools.com/php/php_sessions.asp> (Accessed: 02 October 2024).

Sayed, I. (2021) *Connect python to mysql database with Pymysql and phpmyadmin.*, *Medium*. Available at: <https://imranhsayed.medium.com/connect-python-to-mysql-database-with-pymysql-and-phpmyadmin-aa1b6e89890> (Accessed: 16 October 2024).

GeeksforGeeks (2024) *Differences between functional and non-functional testing*, *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/differences-between-functional-and-non-functional-testing/ (Accessed: 08 November 2024).