INVENTORY MANAGEMENT SYSTEM FOR RETAILERS

PROJECT REPORT

Submitted by

TEAM ID: PNT2022TMID11947

ARAVINTH M

DINESH R

DHINESH U

ARUN A P

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

KSR COLLEGE OF ENGINEERING (Autonomous)

TIRUCHENGODE - 637 215

ANNA UNIVERSITY: CHENNAI 600 025

NOV-DEC 2022

INDEX

1. INTRODUCTION

1.1 Project Overview

This project is aimed at developing a desktop-based application named Inventory Management System for managing the inventory system of any organization. The Inventory Management System (IMS) refers to the system and processes to manage the stock of an organization with the involvement of a Technology system. This system can be used to store the details of the inventory, stock maintenance, update the inventory based on the sales details, and generate inventory reports weekly or monthly based. This project categorizes individual aspects of the inventory management system. An inventory Management System is important to ensure quality control in businesses that handle transactions revolving around consumer goods. Without proper inventory control, a large retail store may run out of stock on an important item. A good inventory management system will alert the retailer when it is time to record. An automated Inventory Management System helps to minimize errors while recording the stock.

1.2 Purpose

Retail inventory management is the process of ensuring you carry merchandise that shoppers want, with neither too little nor too much on hand. By managing inventory, retailers meet customer demand without running out of stock or carrying excess supply. In practice, effective retail inventory management results in lower costs and a better understanding of sales patterns. Retail inventory management tools and methods give retailers more information on which to run their businesses. Applications have been developed to help retailers track and manage stocks related to their own products. The System will ask retailers to create their accounts by providing essential details. Retailers can access their accounts by logging into the application. Once retailers successfully log in to the application they can update their inventory details, also users will be able to add new stock by submitting essential details related to the stock. They can view details of the current inventory. The System will automatically send an email alert to the retailers if there is no stock found in their accounts. So that they can order new stock.

2. LITERATURE SURVEY

2.1 Existing problem

1. Ordoro — e-commerce inventory management

Ordoro allows you to integrate your different sale channels to improve your fulfillment workflows with features such as shipping label creation, omnichannel inventory management,automated dropshipping, and more.

The most significant advantage of Ordoro is that getting started with the software is free. There are no set-up fees or monthly subscriptions, and users can access a free 30-day onboarding session.

Pros

-
-

Monitor multiple shipping options and compare prices of different delivery services in one place.

Easily create and save return labels information and email customers the label directly.

- Connect with e-commerce platforms such as BigCommerce to notify customers of shipping and tracking updates.

Cons

- Updates between Ordoro and e-commerce platforms are once every hour and not in real-time.
- Doesn’t integrate with many e-commerce platforms or marketplaces such as Amazon.

2. Upserve — restaurant inventory software

One of the best restaurant inventory management software — Upserve gives managers, and business owners access to Android or iOS-based POS systems.

This cloud-based solution for inventory management allows businesses to raise their productivity, track orders easily, and increase profits with a centralized platform to monitor their entire business. Also, since Upserve is a restaurant inventory

Pros

-

-

management system, you can set access levels so managers can approve changes to checks securely and set automated gratuity levels.

Easily track your inventory levels and see what needs to be ordered.

This software can help you streamline your ordering process and minimize errors, which can lead to significant savings over time.

- You can free up cash flow and increase your profits.

Cons

- Challenging to learn how to use all the features and functions of the software.
- If you do not have a strong internet connection, the software may not work as well or may be difficult to access.

3. Zoho inventory — inventory management software

Zoho inventory is a great solution for businesses that need help managing their inventory levels.

It offers real-time tracking and alerts to help businesses keep track of their stock levels and avoid stockouts. Zoho inventory

Pros

-

-

also integrates with other Zoho products,making it a comprehensive solution for businesses of all sizes.

Zoho inventory is a great option if you're looking for inventory management software to help streamline your business operations.

Zoho is easy and quick to learn, with great customer support keep track of your inventory levels and know when to order more products.

- Optimize your shipping and receiving processes.

Cons

- It does not offer a lot of features or customization options, which can make it challenging to use for some businesses.
- The software is not always accurate, leading to stock shortages or overages.

4. Square — POS system

Unlike some of the other best cloud-based inventory management software mentioned in this list, Square works offline and can accept payments, so your business can keep operating if there are issues with the internet. Square POS lets

Pros

-

-

managers and business owners easily process discounts and manage refunds. It does this by giving you the tools for inventory management, like saving product names, recording quantities, and pricing.

Pros

- Accepts both credit and debit cards.
- The hardware is lightweight and easy to move about.

- Easily manage sales in the Square POS database.

Cons

- Square card readers and POS hardware aren't available in some countries.
- Lacks the features for customization.

5. Monday.com — inventory control software

More specifically, monday.com Work OS is there to integrate with your other software systems to create harmony across your entire business. Monday.com is suitable for teams and businesses of all sizes as it supports the needs of any process, project, or workflow.

Best of all, you don't need to be a computer whizkid to get everything set up, as the Work OS provides a no code/low code open platform to create widgets, workflows, integrations, and apps.

Pros

- Ready-made project templates mean you can hit the ground running.
- Easy to navigate dashboards and charts.
- Flexibility by managing projects via various columns and view types, including Kanban, Gantt, tables, and more.

Cons

- The mobile version isn’t optimized and is even missing some functionality.
- Even with templates, using monday.com is complex and not primarily designed for inventory management, so is missing features dedicated manufacturing software has.

6. Spocket — dropshipping inventory management

Spocket enables sellers to search from thousands of US and EU suppliers to start their dropshipping business. They can use Spocket to gather data and order product samples before purchasing. If you’re looking to launch your Dropshipping business as effortlessly as possible, this is the tool for you.

Pros

- Real-time automatic stock level updates.
- Easily track and keep customers informed of status updates of inventory movements.
- One-click import of all your products onto Spocket.

Cons

- If you want to use suppliers outside of the US and EU, you’ll need to pay extra fees.
- Spocket doesn’t connect with marketplaces such as Amazon, eBay, Etsy, Wish, and Groupon.

2.2 References

1. Ashwini R. Patil, Smita V. Pataskar (2013), "Analyzing Material Management Techniques on Construction Project" International Journal of Engineering and Innovative Technology Vol.3, Issue 4, Pp. 96-100.

2. Khyomesh V. Patel,Prof. Chetna M. Vyas(2011), "Construction Materials Management On Project Sites" National Conference on Recent Trends in Engineering & Technology.

3. Narimah Kasim, Siti Radziah Liwan, Alina Shamsuddin, Rozlin Zainal, and Naadira Che Kamaruddin (2012),"Improving On-Site Materials Tracking For Inventory Management In Construction Projects" International Conference of Technology Management, Business and Entrepreneurship., Pp.447 .

4. Narimah Kasim , Aryani Ahmad Latiffi , Mohamad Syazli Fathi , (2013) "RFID Technology for Materials Management in Construction Projects – A Review" International JournaL of Construction Engineering and Management, 2(4A),pp. 7-12.

5.https://www.researchgate.net/publication/320239187_Wal-Mart%27s_Successfull y_Integrated_Supply_Chain_and_the_Necessity_of_Establishing_the_Triple-A_s upply_Chain_in_the_21st_century

6. Abbaterusso J. (2010): Supply chain management at Wal-Mart. Ivey Business School,The University of Western Ontario, London, Ontario.

7. Barry C.L. (2006): Breaking the chain. "Harper's Magazine" July, pp. 33-39.

8. Chandran P.M. (2003): Wal-Mart's supply chain management practices. ICFAI Center for Management Research (ICMR).

9.https://www.researchgate.net/publication/327793184_A_Study_of_Inventory _Management_System_Case_Study

10. L. Ling, Supply chain management: concepts, techniques and practices enhancing the value through collaboration. NJ: World Scientific, 2007. 372 M. Leseure, Key Concepts in Operations Management, 2010.

11. D.S. Plinere, A.N. Borisov, L. Ya. Aleksejeva, " Interaction of Software Agents in the Problem of Coordinating Orders," Automatic Control and Computer Sciences, 2015.

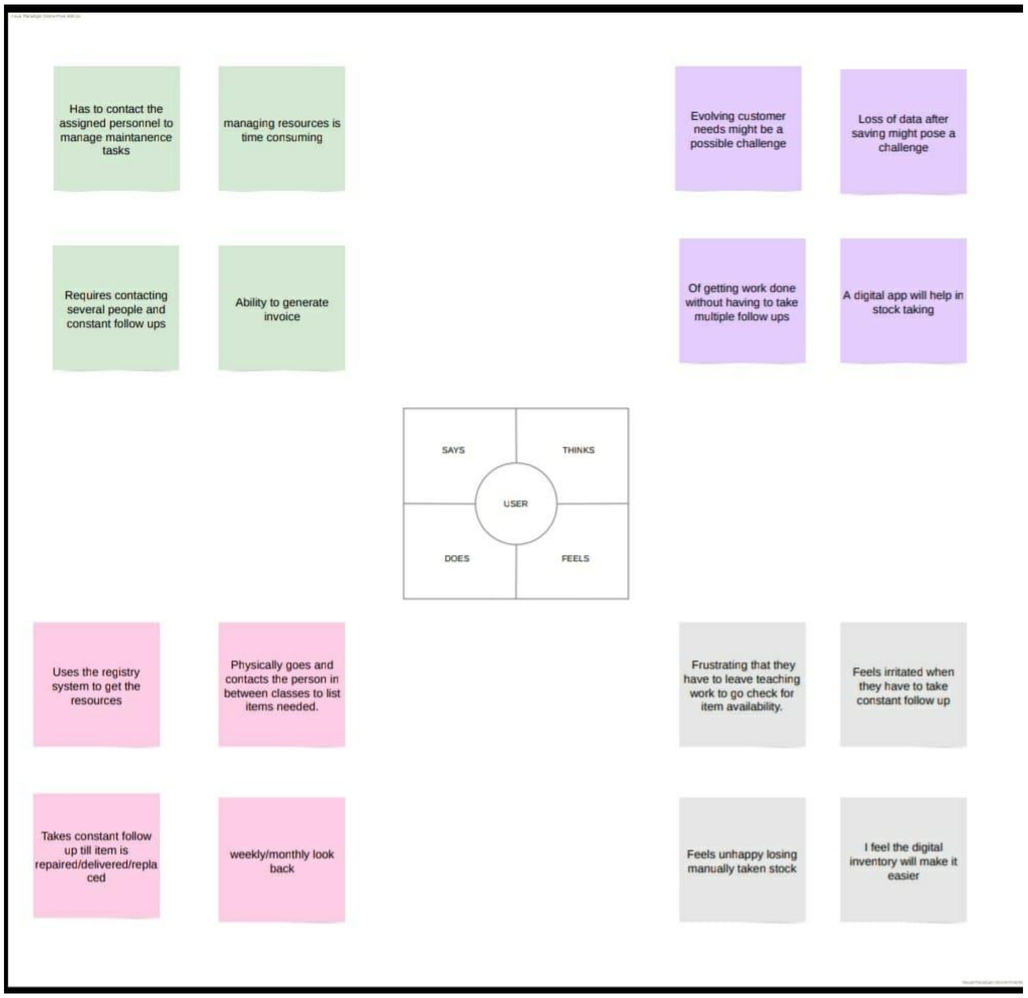
2.3 Problem Statement Definition

Inventories are necessary for sales, which generate profits, and poor management of inventories results in excess inventory, resulting in a lower return on capital invested, affecting the cash conversion cycle. The approximate cost to hold inventory is very high, so maintaining excessive levels of inventories can ruin the company, as they have to reduce prices and absorb losses, and if missing could reduce sales, now maintain inventory levels according to sales forecasts. The problem faced by the company is they do not have any systematic system to record and keep their inventory data. It is difficult for the admin to record the inventory data quickly and safely because they only keep it in the logbook and not properly organized. This project is aimed at developing a desktop-based application named Inventory Management System for managing the inventory system of any organization. The Inventory Management System (IMS) refers to the system and processes to manage the stock of an organization with the involvement of a Technology system. This system can be used to store the details of the inventory, stock maintenance, update the inventory based on the sales details, and generate inventory reports weekly or monthly based. This project categorizes individual aspects of the inventory management system. An inventory Management System is important to ensure quality control in businesses that handle transactions revolving around consumer goods. Without proper inventory control, a large retail store may run out of stock on an important item. A good inventory management system will alert the retailer when it is time to record. An automated Inventory Management System helps to minimize errors while recording the stock.

3. IDEATION & PROPOSED SOLUTION

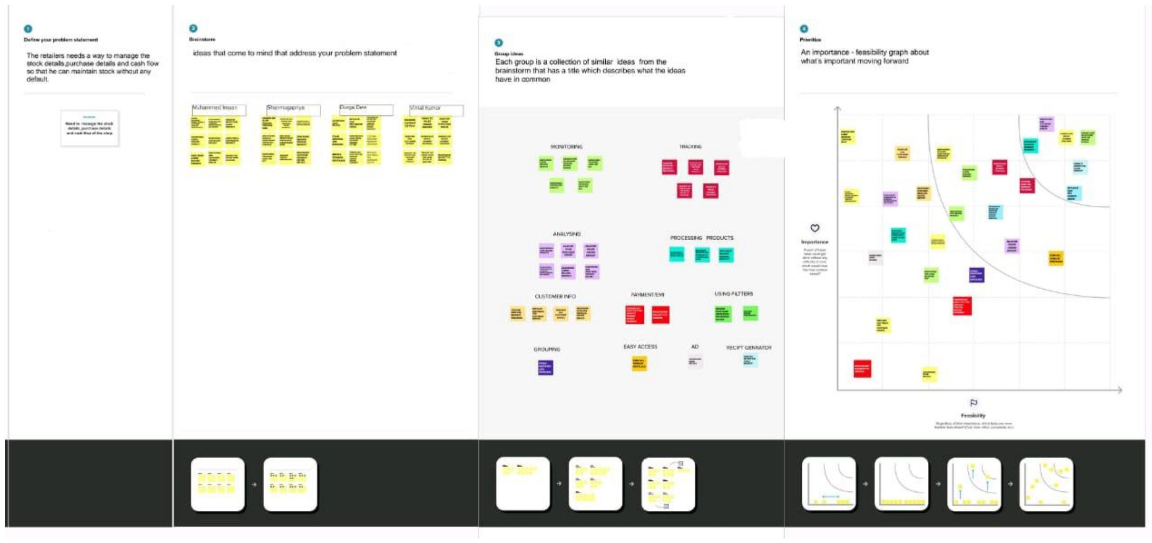
3.1 Empathy Map Canvas

An empathy map is used to gain deeper insights on the customer's interaction with the system. It gives an idea on what the user feels and experiences while using the system, what fears the user has respective to the system, etc. It also specifies how supportive the system environment is and what the users are likely to hear from the people around them regarding the usage of the system.


Has to contact the assigned personnel to manage maintanence tasks
managing resources is time consuming
Evolving customer needs might be a possible challenge
Loss of data after saving might pose a challenge
Requires contacting several people and constant follow ups
Ability to generate invoice
Of getting work done without having to take multiple follow ups
A digital app will help in stock taking
SAYS
THINKS
USER
DOES
FEELS
Uses the registry system to get the resources
Physically goes and contacts the person in between classes to list items needed.
Frustrating that they have to leave teaching work to go check for item availability.
Feels irritated when they have to take constant follow up
Takes constant follow up till item is repaired/delivered/replaced
weekly/monthly look back
Feels unhappy losing manually taken stock
I feel the digital inventory will make it easier

3.2 Ideation & Brainstorming

Ideation and Brainstorming are performed to generate ideas and solutions. Brainstorming is a group activity unlike ideation.


Define your problem statement
The retailers needs a way to manage the stock details,purchase details and cash flow so that he can maintain stock without any default.
Need to manage the stock details, purchase details and cash flow of the shop.
Brainstorm
ideas that come to mind that address your problem statement
Muhammed Imaan
Shanmugapriya
Durga Devi
Vimal Kumar
Group ideas
Each group is a collection of similar ideas from the brainstorm that has a title which describes what the ideas have in common
MONITORING
TRACKING
ANALYSING
PROCESSING PRODUCTS
CUSTOMER INFO
PAYMENT/EMI
USING FILTERS
GROUPING
EASY ACCESS
AD
RECIPT GENERATOR
Prioritize
An importance - feasibility graph about what's important moving forward
Importance
Feasibility

3.3 Proposed Solution

This project is aimed at developing a desktop-based application named Inventory Management System for managing the inventory system of any organization. This system can be used to store the details of the inventory, stock maintenance, update the inventory based on the sales details, and generate inventory reports weekly or monthly based.

Project Design Phase-I
Proposed Solution Template

Date	27 September 2022
Team ID	PNT2022TMID45981
Project Name	Project – INVENTORY MANAGEMENT SYSTEM FOR RETAILERS
Maximum Marks	2 Marks

Proposed Solution Template:

Project team shall fill the following information in proposed solution template.

S.No.	**Parameter**	**Description**
1.	Problem Statement (Problem to be solved)	Retail inventory management is the process of ensuring you carry merchandise that shoppers want, with neither too little nor too much on hand. By managing inventory, retailers meet customer demand without running out of stock or carrying excess supply.
2.	Idea / Solution description	This application helps the retailers to manage the stock details, purchase details and cash flow so that he can maintain stock details without any default. If the stock of the product goes low retailers will receive alert notification.

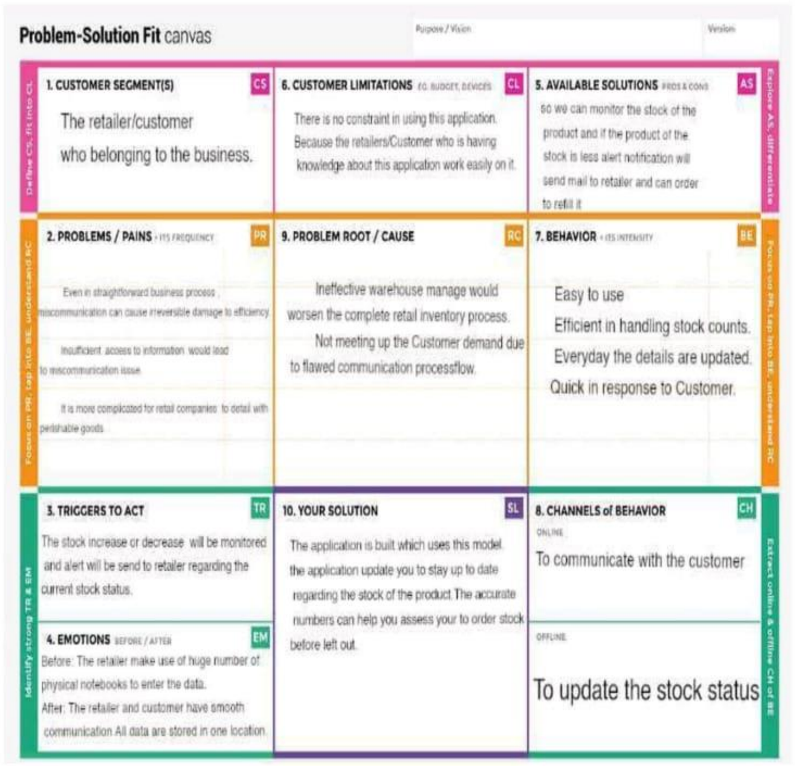
3.	Novelty / Uniqueness	The importance of inventory management cannot be stressed enough especially for e-commerce and online retail brands. Accurate inventory tracking allows brands to fulfil orders timely and accurately. Inventory management in businesses must grow as the company expands. With a strategic plan in place that optimizes the process of overseeing and managing inventory, including real-time data of inventory conditions and levels, companies can achieve inventory management benefits that include Accurate Order Fulfilment,Better Inventory Planning and Ordering, Increased Customer Satisfaction,Organised Warehouse,Minimise the Blockage of Financial Resources
4.	Social Impact / Customer Satisfaction	Inventory models can greatly impact the pricing strategies of products. Inventory management practice can lead to an enhanced competitive advantage and improvement organizational performance. It has a direct positive impact on organizational performance.
		CUSTOMER SATISFACTION : Inventory Management helps to maintain customer satisfaction when it comes to product returns. It helps multiple departments within a company to work together to improve their level of service.
5.	Business Model (Revenue Model)	Inventory management means a business strategy , which deals with managing order processing, manufacturing, storing, and selling raw materials and finished goods.
6.	Scalability of the Solution	To increase the scalability of the business, Inventory management is very helpful. This application will make business much more scalable so that one can continue building consistent growth and take advantage of increased scales.

3.4 Problem Solution fit

The Problem-Solution Fit means that the solution that is realized can actually solve the problem that the customer faces.

Problem Solution Fit

Date	20 September 2022
Team ID	PNT2022TMID45981
Project Name	INVENTORY MANAGEMENT SYSTEM FOR RETAILERS
Maximum Marks	4 Marks



Problem-Solution Fit canvas

Purpose / Vision

Version

Define CS, fit into CL

1. CUSTOMER SEGMENT(S) CS

The retailer/customer who belonging to the business.

6. CUSTOMER LIMITATIONS EG. BUDGET, DEVICES CL

There is no constraint in using this application. Because the retailers/Customer who is having knowledge about this application work easily on it.

5. AVAILABLE SOLUTIONS PROS & CONS AS

so we can monitor the stock of the product and if the product of the stock is less alert notification will send mail to retailer and can order to refill it

Explore AS, differentiate

Focus on PR, tap into BE, understand RC

2. PROBLEMS / PAINS + ITS FREQUENCY PR

Even in straightforward business process , miscommunication can cause irreversible damage to efficiency

Insufficient access to information would lead to miscommunication issue.

It is more complicated for retail companies to detail with perishable goods.

9. PROBLEM ROOT / CAUSE RC

Ineffective warehouse manage would worsen the complete retail inventory process.

Not meeting up the Customer demand due to flawed communication processflow.

7. BEHAVIOR + ITS INTENSITY BE

Easy to use

Efficient in handling stock counts.

Everyday the details are updated.

Quick in response to Customer.

Focus on PR, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TO ACT TR

The stock increase or decrease will be monitored and alert will be send to retailer regarding the current stock status.

4. EMOTIONS BEFORE / AFTER EM

Before: The retailer make use of huge number of physical notebooks to enter the data.

After: The retailer and customer have smooth communication All data are stored in one location.

10. YOUR SOLUTION SL

The application is built which uses this model. the application update you to stay up to date regarding the stock of the product The accurate numbers can help you assess your to order stock before left out.

8. CHANNELS of BEHAVIOR CH

ONLINE

To communicate with the customer

OFFLINE

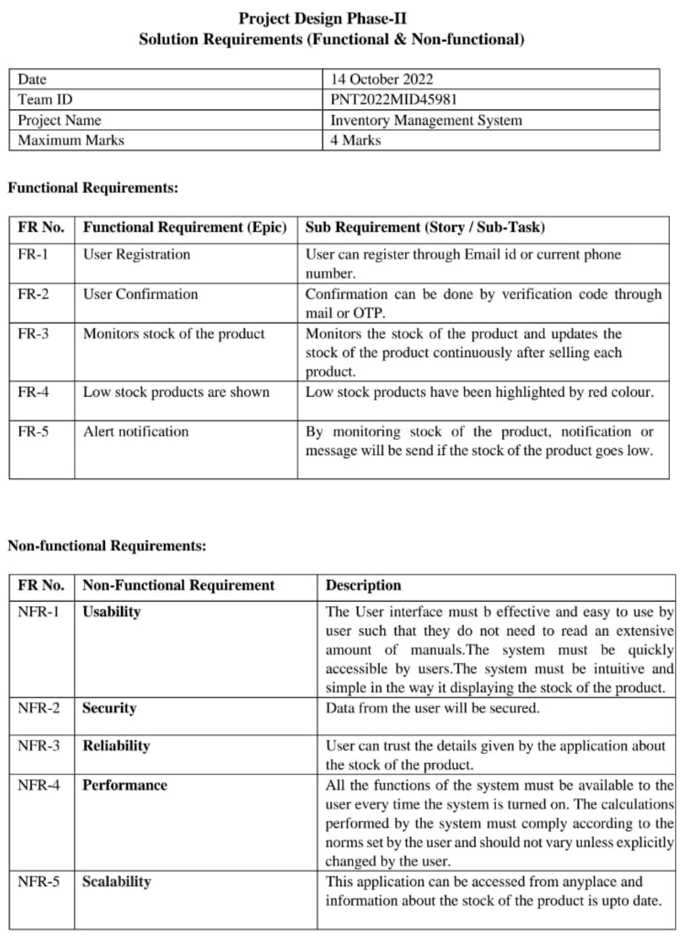
To update the stock status

Extract online & offline CH of BE

4. REQUIREMENT ANALYSIS

4.1 Functional requirements

Functional Requirements specify the features and functions of the proposed system.



Project Design Phase-II
Solution Requirements (Functional & Non-functional)

Date	14 October 2022
Team ID	PNT2022MID45981
Project Name	Inventory Management System
Maximum Marks	4 Marks

Functional Requirements:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	User can register through Email id or current phone number.
FR-2	User Confirmation	Confirmation can be done by verification code through mail or OTP.
FR-3	Monitors stock of the product	Monitors the stock of the product and updates the stock of the product continuously after selling each product.
FR-4	Low stock products are shown	Low stock products have been highlighted by red colour.
FR-5	Alert notification	By monitoring stock of the product, notification or message will be send if the stock of the product goes low.

Non-functional Requirements:

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The User interface must b effective and easy to use by user such that they do not need to read an extensive amount of manuals.The system must be quickly accessible by users.The system must be intuitive and simple in the way it displaying the stock of the product.
NFR-2	**Security**	Data from the user will be secured.
NFR-3	**Reliability**	User can trust the details given by the application about the stock of the product.
NFR-4	**Performance**	All the functions of the system must be available to the user every time the system is turned on. The calculations performed by the system must comply according to the norms set by the user and should not vary unless explicitly changed by the user.
NFR-5	**Scalability**	This application can be accessed from anyplace and information about the stock of the product is upto date.

4.2 Non-Functional requirements

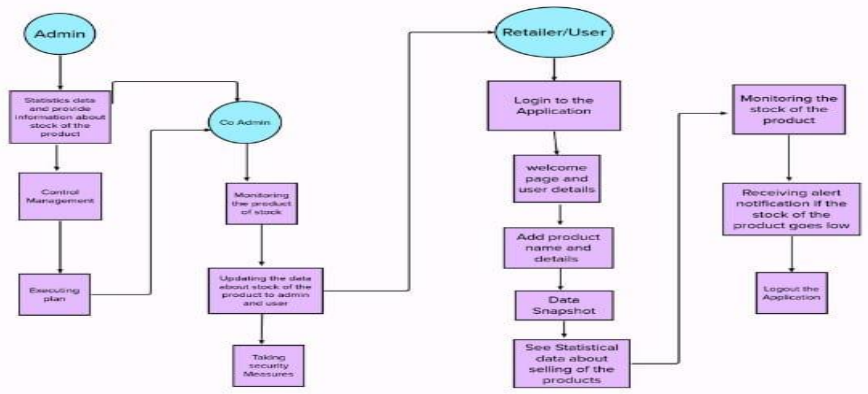
Non functional requirements specify the general properties of the proposed system.

1. **User Friendly.** The system should use familiar user interfaces such as that used to surf the Internet.
2. **Modularity.** The Inventory Control system should be able to operate on its own. It is independent of all other software systems except the underlying operating system.
3. **Robustness.** The system shall be built with a robust error recovery routines to handle system failures and to enable 24 x 7 operation, 24 hours per day, 7 days a week.
4. **Reliability of access.** The system should be reliably accessed over the company intranet.

5. PROJECT DESIGN

5.1 Data Flow Diagrams

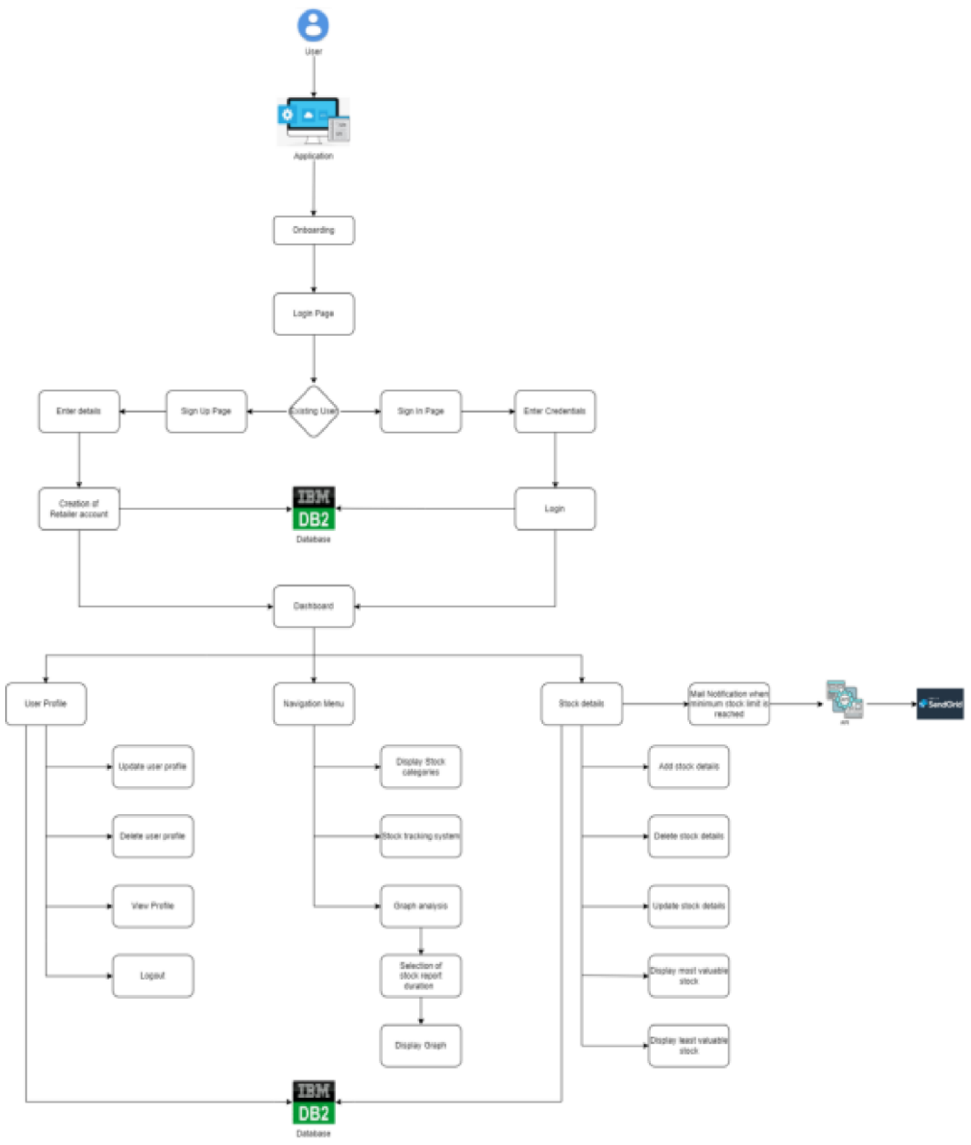
A data flow diagram or DFD(s) maps out the flow of information for any process or system. DFDs help you better understand process or system operation to discover potential problems, improve efficiency, and develop better processes.


Admin
Statistics data and provide information about stock of the product
Control Management
Executing plan
Co Admin
Monitoring the product of stock
Updating the data about stock of the product to admin and user
Taking security Measures
Retailer/User
Login to the Application
welcome page and user details
Add product name and details
Data Snapshot
See Statistical data about selling of the products
Monitoring the stock of the product
Receiving alert notification if the stock of the product goes low
Logout the Application

5.2 Solution Architecture

Solution architecture is the process of developing solutions based on predefined processes, guidelines and best practices with the objective that the developed solution fits within the enterprise architecture in terms of information architecture, system portfolios, integration requirements, etc.

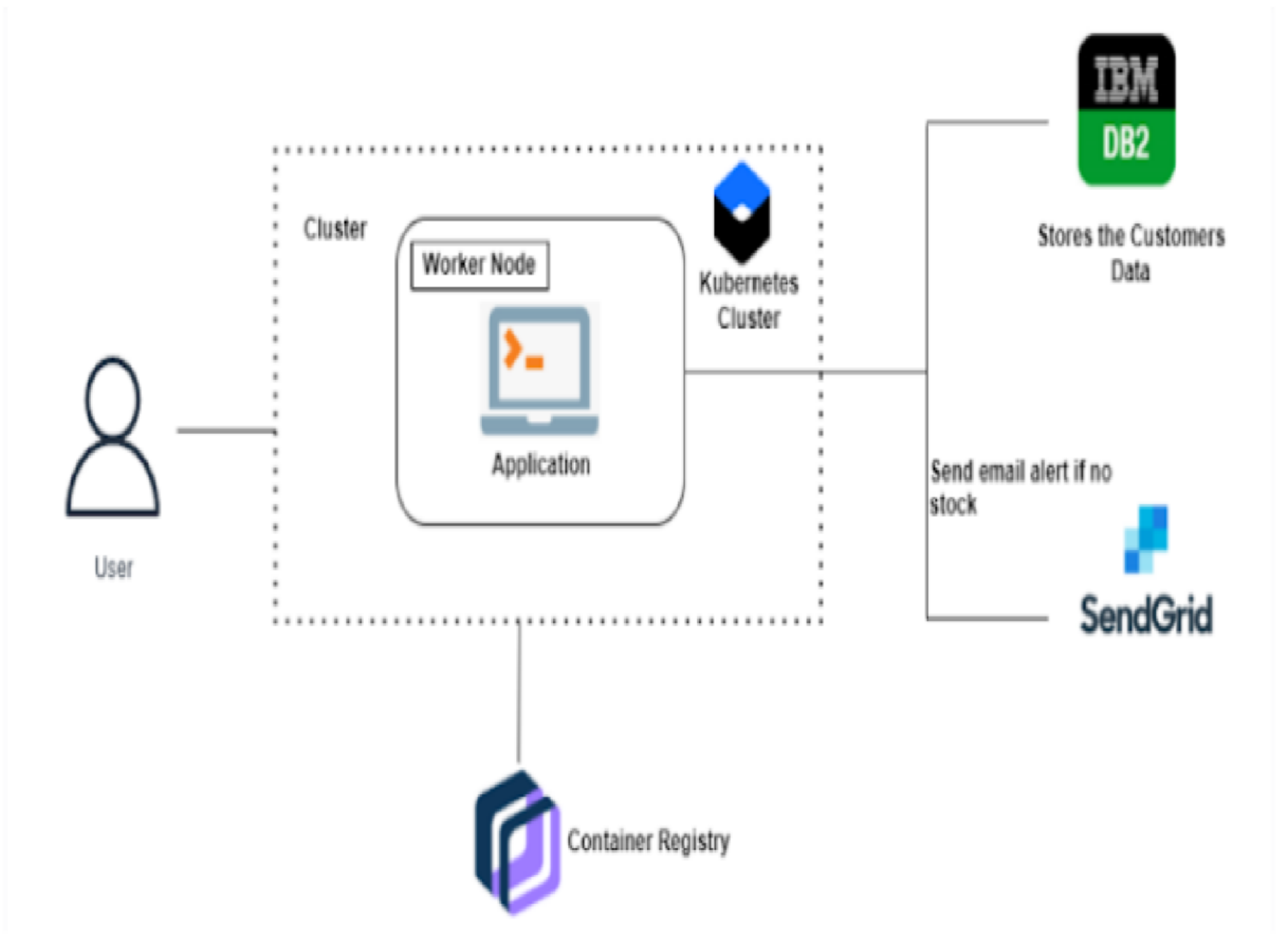
Solution Architecture:



5.3 Technical Architecture:

Technical architecture involves the development of a technical blueprint regarding the arrangement, interaction, and interdependence of all elements so that system-relevant requirements are met.

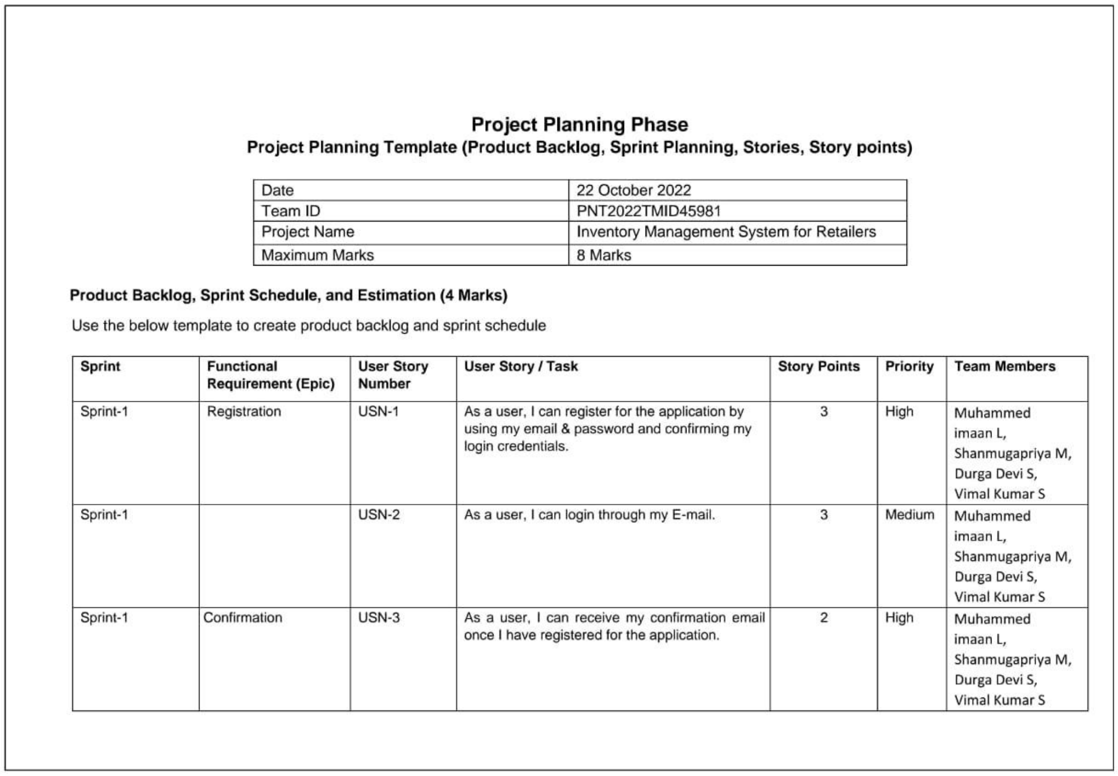
Technical Architecture:



6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

The purpose of sprint planning is to define what can be delivered in the sprint and how that work will be achieved. Sprint planning is done in collaboration with the whole team.



Project Planning Phase

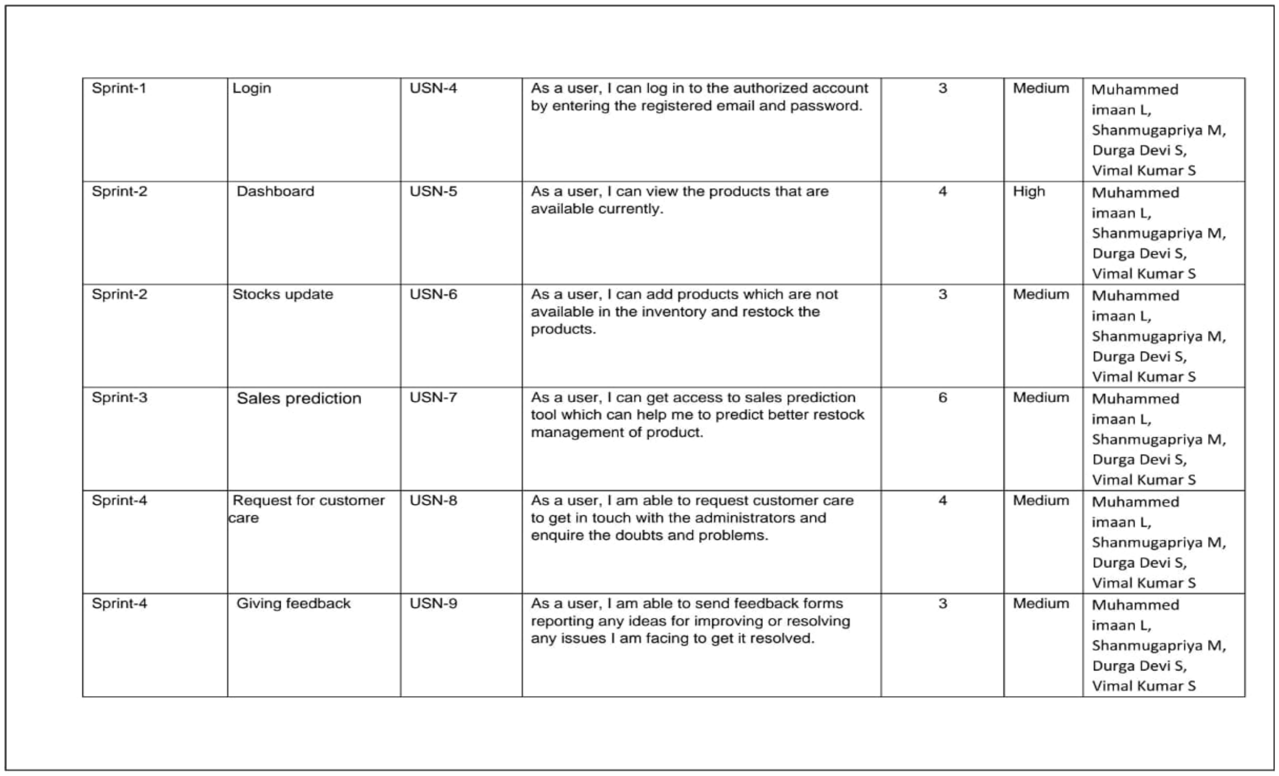
Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Date	22 October 2022
Team ID	PNT2022TMID45981
Project Name	Inventory Management System for Retailers
Maximum Marks	8 Marks

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Use the below template to create product backlog and sprint schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by using my email & password and confirming my login credentials.	3	High	Muhammed imaan L, Shanmugapriya M, Durga Devi S, Vimal Kumar S
Sprint-1		USN-2	As a user, I can login through my E-mail.	3	Medium	Muhammed imaan L, Shanmugapriya M, Durga Devi S, Vimal Kumar S
Sprint-1	Confirmation	USN-3	As a user, I can receive my confirmation email once I have registered for the application.	2	High	Muhammed imaan L, Shanmugapriya M, Durga Devi S, Vimal Kumar S

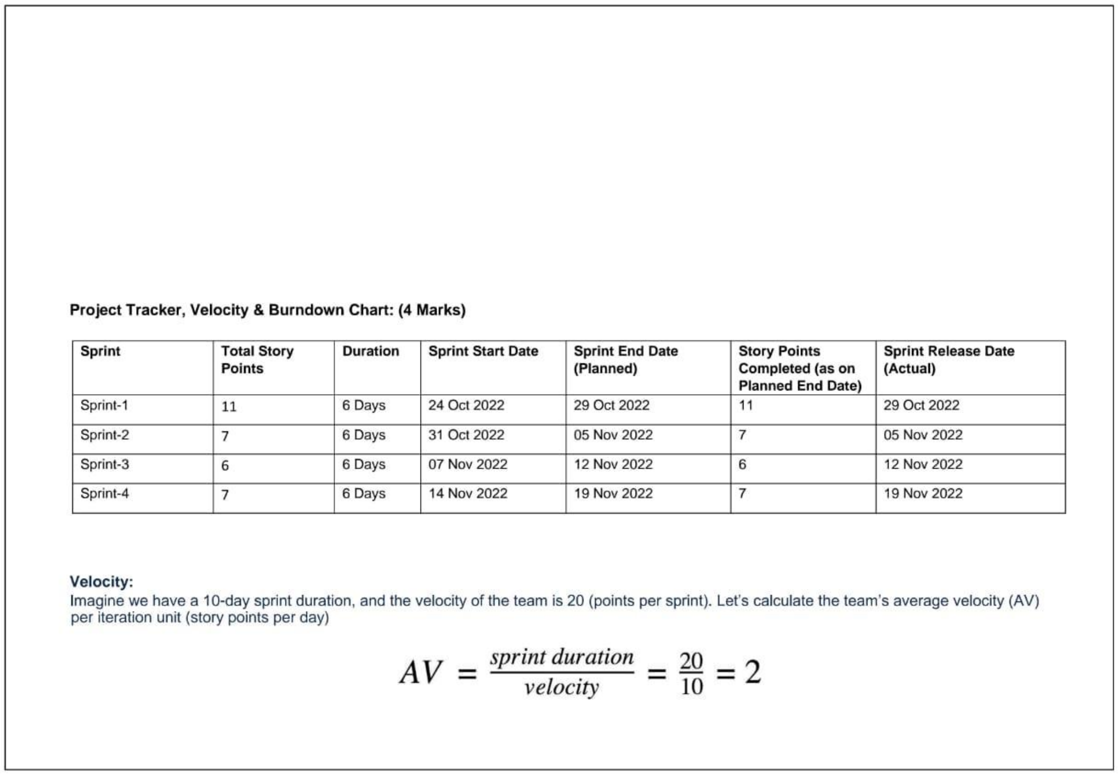


Sprint-1	Login	USN-4	As a user, I can log in to the authorized account by entering the registered email and password.	3	Medium	Muhammed imaan L, Shanmugapriya M, Durga Devi S, Vimal Kumar S
Sprint-2	Dashboard	USN-5	As a user, I can view the products that are available currently.	4	High	Muhammed imaan L, Shanmugapriya M, Durga Devi S, Vimal Kumar S
Sprint-2	Stocks update	USN-6	As a user, I can add products which are not available in the inventory and restock the products.	3	Medium	Muhammed imaan L, Shanmugapriya M, Durga Devi S, Vimal Kumar S
Sprint-3	Sales prediction	USN-7	As a user, I can get access to sales prediction tool which can help me to predict better restock management of product.	6	Medium	Muhammed imaan L, Shanmugapriya M, Durga Devi S, Vimal Kumar S
Sprint-4	Request for customer care	USN-8	As a user, I am able to request customer care to get in touch with the administrators and enquire the doubts and problems.	4	Medium	Muhammed imaan L, Shanmugapriya M, Durga Devi S, Vimal Kumar S
Sprint-4	Giving feedback	USN-9	As a user, I am able to send feedback forms reporting any ideas for improving or resolving any issues I am facing to get it resolved.	3	Medium	Muhammed imaan L, Shanmugapriya M, Durga Devi S, Vimal Kumar S

6.2 Sprint Delivery Schedule

- Agile sprints typically last from one week to one month. The goal of sprints is to put pressure on teams to innovate and deliver more quickly, hence the shorter the sprint, the better.
- Sprint planning is **an event in scrum that kicks off the sprint**.
- The purpose of sprint planning is to define what can be delivered in the sprint and how that work will be achieved.

- Sprint planning is done in collaboration with the whole scrum team.

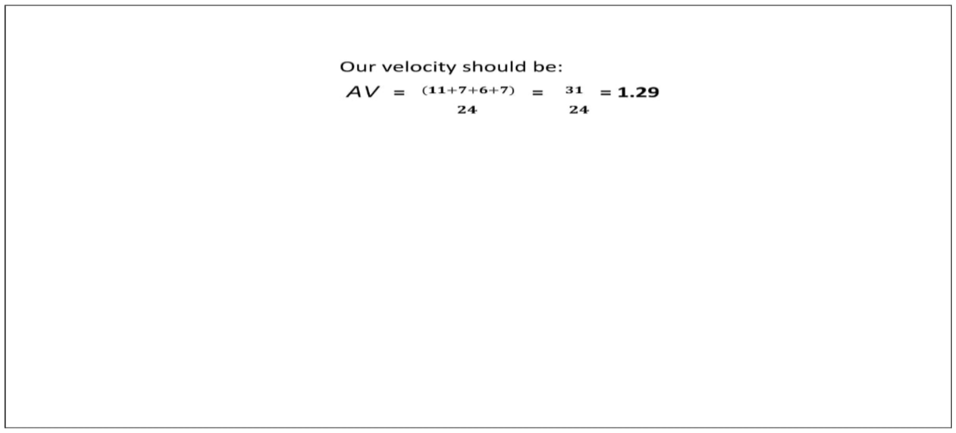


Project Tracker, Velocity & Burndown Chart: (4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	11	6 Days	24 Oct 2022	29 Oct 2022	11	29 Oct 2022
Sprint-2	7	6 Days	31 Oct 2022	05 Nov 2022	7	05 Nov 2022
Sprint-3	6	6 Days	07 Nov 2022	12 Nov 2022	6	12 Nov 2022
Sprint-4	7	6 Days	14 Nov 2022	19 Nov 2022	7	19 Nov 2022

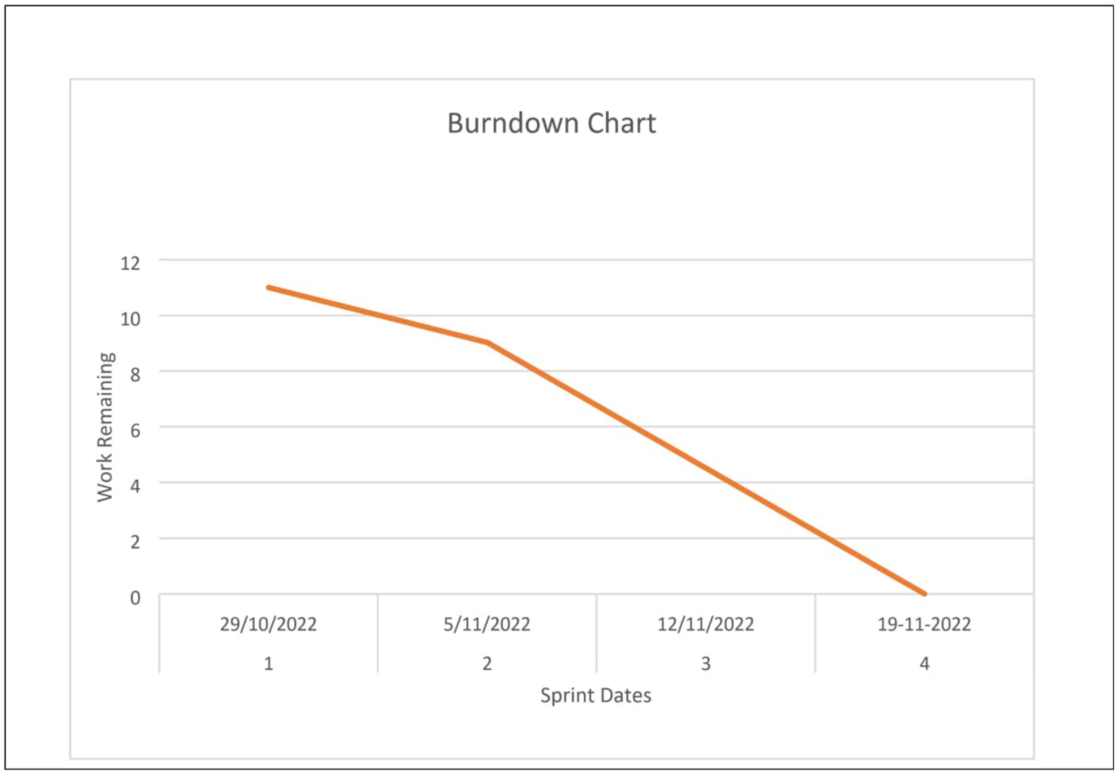
Velocity:
Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$



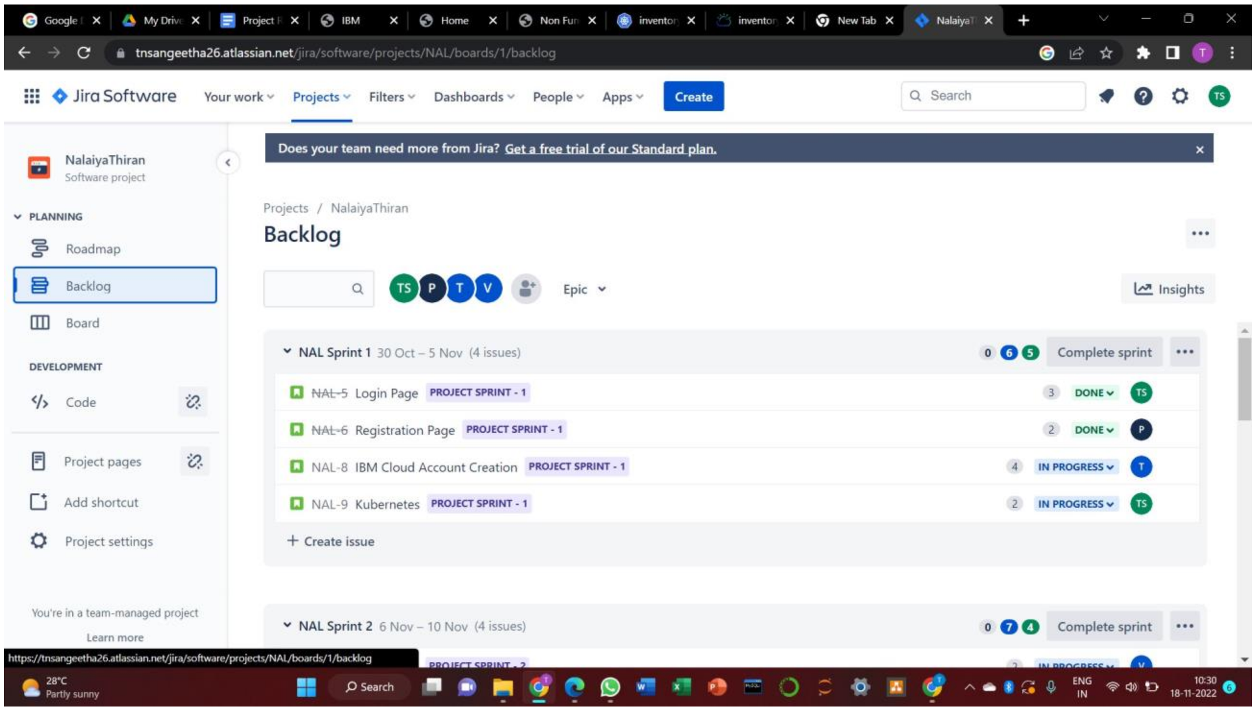
Our velocity should be:

$$AV = \frac{(11+7+6+7)}{24} = \frac{31}{24} = \mathbf{1.29}$$



6.3 Reports from JIRA Backlog:

A backlog is a list of issues that’s related to the project and the functions of the system. It makes it simple to make, store, manage a variety of problems including the ones the team is working on.


tnsangeetha26.atlassian.net/jira/software/projects/NAL/boards/1/backlog

Does your team need more from Jira? Get a free trial of our Standard plan.

NalaiyaThiran
Software project

PLANNING
Roadmap
Backlog
Board

DEVELOPMENT
Code

Project pages
Add shortcut
Project settings

You're in a team-managed project
Learn more

Projects / NalaiyaThiran

Backlog

Epic
Insights

NAL Sprint 1 30 Oct – 5 Nov (4 issues) 0 6 5 Complete sprint

~~NAL-5~~ Login Page	PROJECT SPRINT - 1	3	DONE	TS
~~NAL-6~~ Registration Page	PROJECT SPRINT - 1	2	DONE	P
NAL-8 IBM Cloud Account Creation	PROJECT SPRINT - 1	4	IN PROGRESS	T
NAL-9 Kubernetes	PROJECT SPRINT - 1	2	IN PROGRESS	TS

+ Create issue

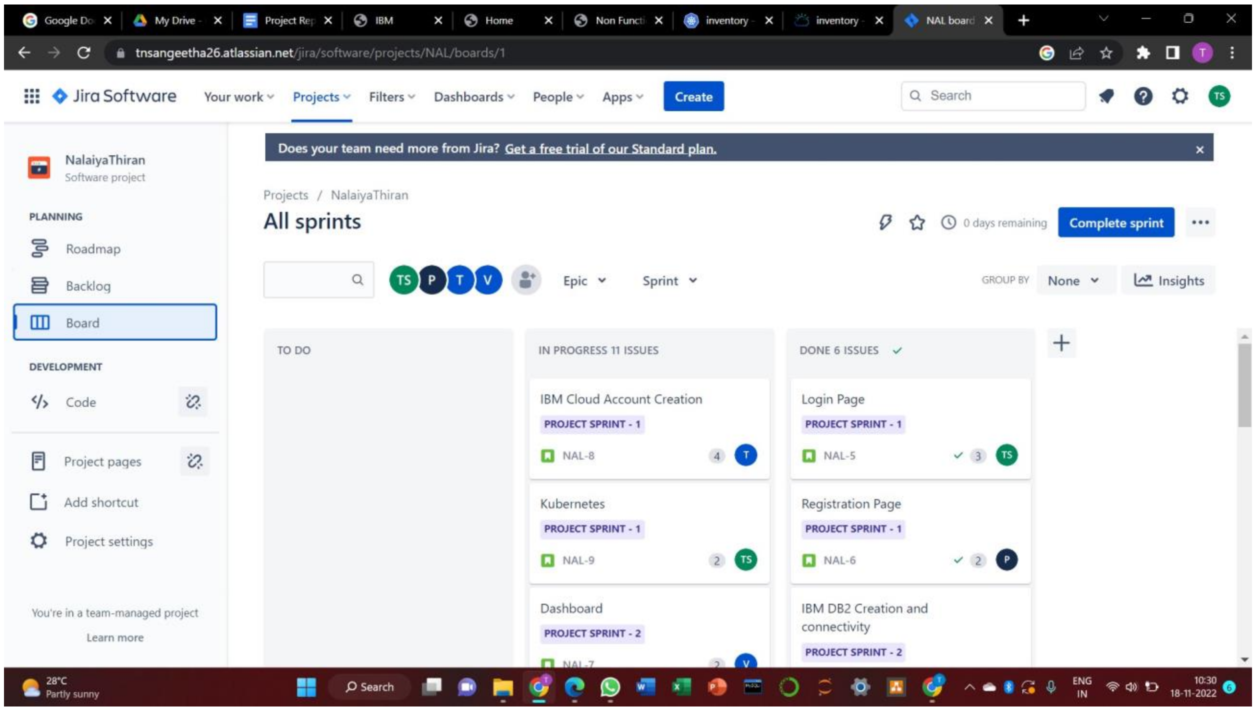
NAL Sprint 2 6 Nov – 10 Nov (4 issues) 0 7 4 Complete sprint

https://tnsangeetha26.atlassian.net/jira/software/projects/NAL/boards/1/backlog

28°C Partly sunny
10:30 18-11-2022

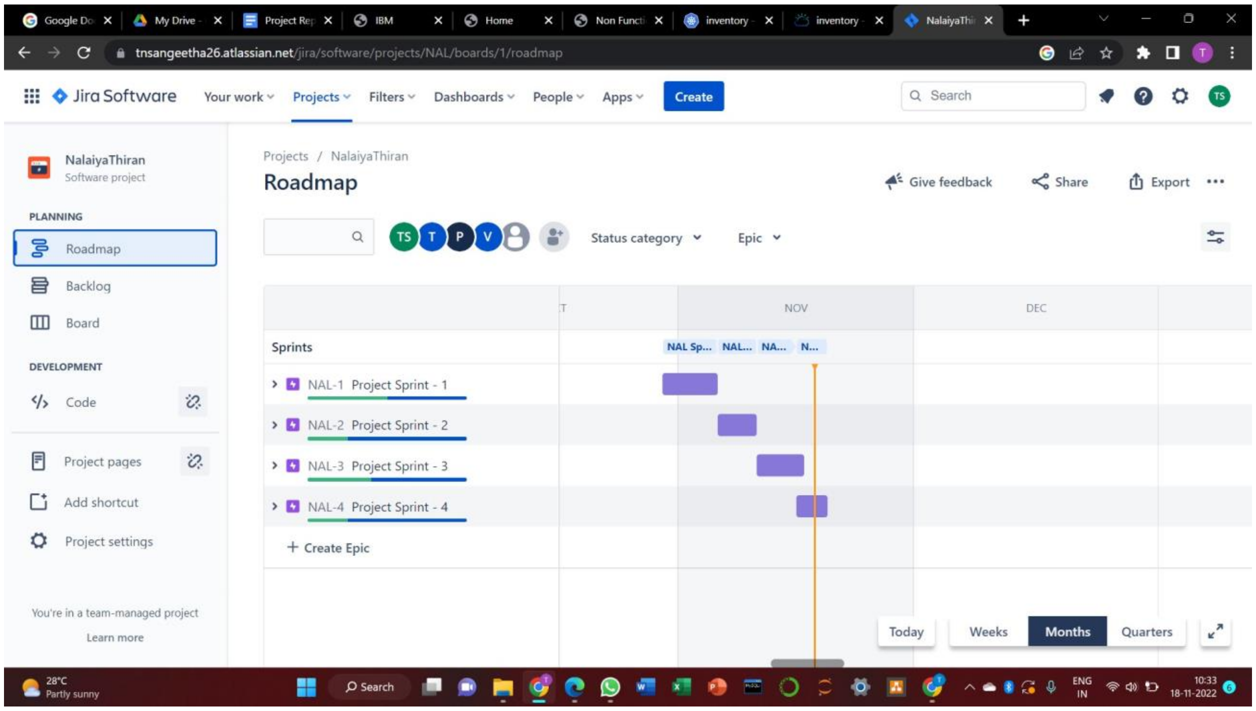
Board:

A board reflects your team's process, tracking the status of work. The columns on the board represent the status of your team's issues. The visual representation of the work helps in discussing and tracking the progress of the project from start to finish.



Roadmap:

A roadmap offers quick and easy planning that helps teams better manage their dependencies and track progress on the big picture in real-time.



7. CODING & SOLUTIONING

Python – app.py:

```
from pickle import TRUE
from inventorymanagement import app
from flask import Flask, request, Response


app = Flask(__name__)
if __name__ == '__main__':
    app.run(debug=TRUE)
```

Feature 1:

home.html:

```
 {% extends "layout.html" %}

{% block content %}
    <div class="col-md-12">
        <div class="row">
```

```
                <div class="col-lg-3 col-md-6">
                <a href="{{ url_for('view_product') }}"
style="text-decoration:none; color: white;">
                <div class="border rounded navbar-dark bg-dark">
                    <div class="panel-heading">
                        <br>
                        <div class="row">
                            <div class="col-md-8 offset-md-3 text-right">
                                <div class="huge">{{ products }}</div>
                                <div>Total Products</div>
                                <div>View Products</div>
                            </div>
                        </div>
                        <br>
                    </div>
                </div>
                </a>
            </div>
            <div class="col-lg-3 col-md-6">
                <a href="{{ url_for('view_location') }}"
style="text-decoration:none; color: white;">
                <div class="border rounded navbar-dark bg-dark">
                    <div class="panel-heading">
                        <br>
                        <div class="row">
                            <div class="col-md-8 offset-md-3 text-right">
                                <div class="huge">{{ locations }}</div>
                                <div>Total Locations</div>
                                <div>View Locations</div>
                            </div>
                        </div>
                        <br>
                    </div>
                </div>
                </a>
            </div>
            <div class="col-lg-3 col-md-6">
                <a href="#" style="text-decoration:none; color: white;">
                <div class="border rounded navbar-dark bg-dark">
```

```
                    <div class="panel-heading">
                        <br>
                        <div class="row">
                            <div class="col-md-8 offset-md-3 text-right">
                                <div class="huge">{{ sales }}</div>
                                <div>Total Sales</div>
                                <div>View Sales</div>
                            </div>
                        </div>
                        <br>
                    </div>
                </div>
                </a>
            </div>
            <div class="col-lg-3 col-md-6">
                <a href="{{ url_for('view_productmovement') }}"
style="text-decoration:none; color: white;">
                <div class="border rounded navbar-dark bg-dark">
                    <div class="panel-heading">
                        <br>
                        <div class="row">
                            <div class="col-md-8 offset-md-3 text-right">
                                <div class="huge">{{ movements }}</div>
                                <div>Total Movement</div>
                                <div>Product Movement</div>
                            </div>
                        </div>
                        <br>
                    </div>
                </div>
                </a>
            </div>
        </div>
    </div>
{% endblock content %}
```

Feature 2 :

login.html:

```
{% extends "layout.html" %}
{% block content %}
    <div class="col-md-6 offset-md-3 border pt-2">
        <br>
        <div class="row">
            <div class="col-md-12">
                <form method="POST" action="">
                {{ form.hidden_tag() }} <!--CSRF TOKEN -->
                    <fieldset class="form-group">
                        <div class="form-group">
                             {{ form.email.label(class="form-control-label")
}}

                            {% if form.email.errors %}
                                {{ form.email(class="form-control
form-control-lg is-invalid") }}
                                <div class="invalid-feedback">
                                     {% for error in form.username.errors %}
                                        <span>{{ error }}</span>
                                    {% endfor %}
                                 </div>
                            {% else %}
                                {{ form.email(class="form-control
form-control-lg") }}
                            {% endif %}
                        </div>
                        <div class="form-group">
                            {{
form.password.label(class="form-control-label") }}

                            {% if form.password.errors %}
                                {{ form.password(class="form-control
form-control-lg is-invalid") }}
                                <div class="invalid-feedback">
```

```
                                     {% for error in form.password.errors %}
                                        <span>{{ error }}</span>
                                    {% endfor %}
                                </div>
                            {% else %}
                                {{ form.password(class="form-control
form-control-lg") }}
                            {% endif %}
                        </div>
                        <div class="form-check">
                            {{ form.remember(class="form-check-input") }}
                            {{ form.remember.label(class="form-check-label")
}}
                        </div>
                    </fieldset>
                    <div class="form-group">
                        {{ form.submit(class="btn btn-success navbar-dark
bg-dark") }}
                    </div>
                </form>
            </div>
            <div class="col-md-12 border-top pt-2">
                <small class="text-muted">
                    D'ont Have an Account ? <a class="ml-2" href="{{
url_for('register') }}">Register Now</a>
                </small>
            </div>
        </div>
    </div>
{% endblock content %}
```

final.css:

```
body {
  background-color: #f8f8f8;
}
#wrapper {
  width: 100%;
```

```
}
#page-wrapper {
  padding: 0 15px;
  min-height: 568px;
  background-color: white;
}
@media (min-width: 768px) {
  #page-wrapper {
    position: inherit;
    margin: 0 0 0 250px;
    padding: 0 30px;
    border-left: 1px solid #e7e7e7;
  }
}
.navbar-top-links {
  margin-right: 0;
}
.navbar-top-links li {
  display: inline-block;
}
.navbar-top-links li:last-child {
  margin-right: 15px;
}
.navbar-top-links li a {
  padding: 15px;
  min-height: 50px;
}
.navbar-top-links .dropdown-menu li {
  display: block;
}
.navbar-top-links .dropdown-menu li:last-child {
  margin-right: 0;
}
.navbar-top-links .dropdown-menu li a {
  padding: 3px 20px;
  min-height: 0;
}
.navbar-top-links .dropdown-menu li a div {
  white-space: normal;
```

```
}
.navbar-top-links .dropdown-messages,
.navbar-top-links .dropdown-tasks,
.navbar-top-links .dropdown-alerts {
  width: 310px;
  min-width: 0;
}
.navbar-top-links .dropdown-messages {
  margin-left: 5px;
}
.navbar-top-links .dropdown-tasks {
  margin-left: -59px;
}
.navbar-top-links .dropdown-alerts {
  margin-left: -123px;
}
.navbar-top-links .dropdown-user {
  right: 0;
  left: auto;
}
.sidebar .sidebar-nav.navbar-collapse {
  padding-left: 0;
  padding-right: 0;
}
.sidebar .sidebar-search {
  padding: 15px;
}
.sidebar ul li {
  border-bottom: 1px solid #e7e7e7;
}
.sidebar ul li a.active {
  background-color: #eeeeee;
```

```
}
.sidebar .arrow {
  float: right;
}
.sidebar .fa.arrow:before {
  content: "\f104";
}
.sidebar .active > a > .fa.arrow:before {
  content: "\f107";
}
.sidebar .nav-second-level li,
.sidebar .nav-third-level li {
  border-bottom: none !important;
}
.sidebar .nav-second-level li a {
  padding-left: 37px;
}
.sidebar .nav-third-level li a {
  padding-left: 52px;
}
@media (min-width: 768px) {
  .sidebar {
    z-index: 1;
    position: absolute;
    width: 250px;
    margin-top: 51px;
  }
```

main.js:

```
!function(t,e){"object"==typeof exports&&"undefined"!=typeof
module?e(exports,require("jquery"),require("popper.js")):"function"==typeof
define&&define.amd?define(["exports","jquery","popper.js"],e):e(t.bootstrap=
{},t.jQuery,t.Popper)}(this,function(t,e,h){"use strict";function
i(t,e){for(var n=0;n<e.length;n++){var
i=e[n];i.enumerable=i.enumerable||!1,i.configurable=!0,"value"in
i&&(i.writable=!0),Object.defineProperty(t,i.key,i)}}function
```

```
s(t,e,n){return     e&&i(t.prototype,e),n&&i(t,n),t}function     l(r){for(var
t=1;t<arguments.length;t++){var
o=null!=arguments[t]?arguments[t]:{},e=Object.keys(o);"function"==typeof
Object.getOwnPropertySymbols&&(e=e.concat(Object.getOwnPropertySymbols(o).fi
lter(function(t){return
Object.getOwnPropertyDescriptor(o,t).enumerable}))),e.forEach(function(t){va
r                            e,n,i;e=r,i=o[n=t],n                          in
```

```
e?Object.defineProperty(e,n,{value:i,enumerable:!0,configurable:!0,writable:
!0}):e[n]=i})}return
r}e=e&&e.hasOwnProperty("default")?e.default:e,h=h&&h.hasOwnProperty("defaul
t")?h.default:h;var
r,n,o,a,c,u,f,d,g,_,m,p,v,y,E,C,T,b,S,I,A,D,w,N,O,k,P,j,H,L,R,x,W,U,q,F,K,M,
Q,B,V,Y,z,J,Z,G,$,X,tt,et,nt,it,rt,ot,st,at,lt,ct,ht,ut,ft,dt,gt,_t,mt,pt,vt
,yt,Et,Ct,Tt,bt,St,It,At,Dt,wt,Nt,Ot,kt,Pt,jt,Ht,Lt,Rt,xt,Wt,Ut,qt,Ft,Kt,Mt,
Qt,Bt,Vt,Yt,zt,Jt,Zt,Gt,$t,Xt,te,ee,ne,ie,re,oe,se,ae,le,ce,he,ue,fe,de,ge,_
e,me,pe,ve,ye,Ee,Ce,Te,be,Se,Ie,Ae,De,we,Ne,Oe,ke,Pe,je,He,Le,Re,xe,We,Ue,qe
,Fe,Ke,Me,Qe,Be,Ve,Ye,ze,Je,Ze,Ge,$e,Xe,tn,en,nn,rn,on,sn,an,ln,cn,hn,un,fn,
dn,gn,_n,mn,pn,vn,yn,En,Cn,Tn,bn,Sn,In,An,Dn,wn,Nn,On,kn,Pn,jn,Hn,Ln,Rn,xn,W
n,Un,qn,Fn=function(i){var        e="transitionend";function        t(t){var
e=this,n=!1;return
i(this).one(l.TRANSITION_END,function(){n=!0}),setTimeout(function(){n||l.tr
iggerTransitionEnd(e)},t),this}var
l={TRANSITION_END:"bsTransitionEnd",getUID:function(t){for(;t+=~~(1e6*Math.r
andom()),document.getElementById(t););return
t},getSelectorFromElement:function(t){var
e=t.getAttribute("data-target");e&&"#"!==e||(e=t.getAttribute("href")||"");t
ry{return                   document.querySelector(e)?e:null}catch(t){return
null}},getTransitionDurationFromElement:function(t){if(!t)return       0;var
e=i(t).css("transition-duration");return
parseFloat(e)?(e=e.split(",")[0],1e3*parseFloat(e)):0},reflow:function(t){re
turn
t.offsetHeight},triggerTransitionEnd:function(t){i(t).trigger(e)},supportsTr
ansitionEnd:function(){return
Boolean(e)},isElement:function(t){return(t[0]||t).nodeType},typeCheckConfig:
function(t,e,n){for(var                         i                          in
n)if(Object.prototype.hasOwnProperty.call(n,i)){var
```

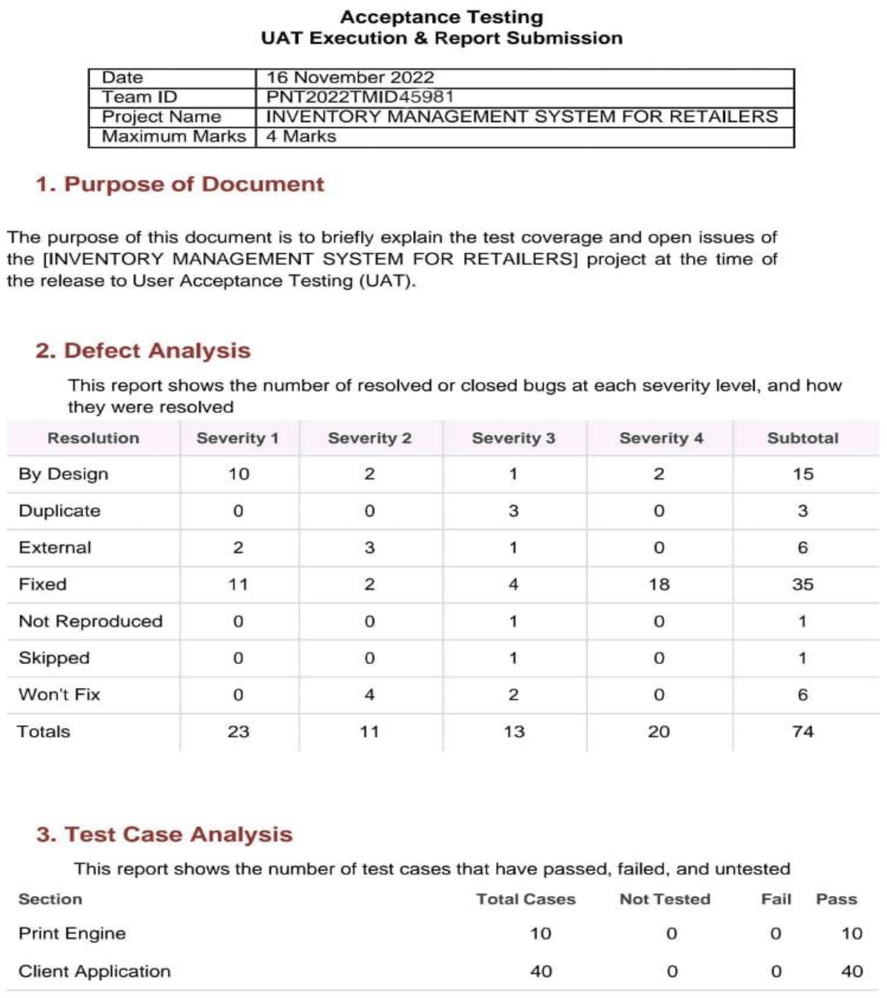
```
r=n[i],o=e[i],s=o&&l.isElement(o)?"element":(a=o,{}.toString.call(a).match(/
\s([a-z]+)/i)[1].toLowerCase());if(!new      RegExp(r).test(s))throw      new
Error(t.toUpperCase()+': Option "'+i+'" provided type "'+s+'" but expected
type                            "'+r+'".')}var                        a}};return
i.fn.emulateTransitionEnd=t,i.event.special[l.TRANSITION_END]={bindType:e,de
legateType:e,handle:function(t){if(i(t.target).is(this))return
t.handleObj.handler.apply(this,arguments)}},l}(e),Kn=(n="alert",a="."+(o="bs
.alert"),c=(r=e).fn[n],u={CLOSE:"close"+a,CLOSED:"closed"+a,CLICK_DATA_API:"
click"+a+".data-api"},f="alert",d="fade",g="show",_=function(){function
i(t){this._element=t}var     t=i.prototype;return     t.close=function(t){var
e=this._element;t&&(e=this._getRootElement(t)),this._triggerCloseEvent(e).is
DefaultPrevented()||this._removeElement(e)},t.dispose=function(){r.removeDat
a(this._element,o),this._element=null},t._getRootElement=function(t){var
e=Fn.getSelectorFromElement(t),n=!1;return
e&&(n=document.querySelector(e)),n||(n=r(t).closest("."+f)[0]),n},t._trigger
CloseEvent=function(t){var                         e=r.Event(u.CLOSE);return
r(t).trigger(e),e},t._removeElement=function(e){var
n=this;if(r(e).removeClass(g),r(e).hasClass(d)){var
t=Fn.getTransitionDurationFromElement(e);r(e).one(Fn.TRANSITION_END,function
(t){return             n._destroyElement(e,t)}).emulateTransitionEnd(t)}else
this._destroyElement(e)},t._destroyElement=function(t){r(t).detach().trigger
(u.CLOSED).remove()},i._jQueryInterface=function(n){return
this.each(function(){var                     t=r(this),e=t.data(o);e||(e=new
i(this),t.data(o,e)),"close"===n&&e[n](this)})},i._handleDismiss=function(e)
{return
function(t){t&&t.preventDefault(),e.close(this)}},s(i,null,[{key:"VERSION",g
et:function(){return"4.1.3"}}]),i}(),r(document).on(u.CLICK_DATA_API,'[data-
dismiss="alert"]',_._handleDismiss(new
_)),r.fn[n]=_._jQueryInterface,r.fn[n].Constructor=_,r.fn[n].noConflict=func
tion(){return
r.fn[n]=c,_._jQueryInterface},_),Mn=(p="button",y="."+(v="bs.button"),E=".da
ta-api",C=(m=e).fn[p]
```

8. TESTING

8.1 Test Cases

8.2 User Acceptance testing

Before deploying the software application to a production environment the end user or client performs a type of testing known as user acceptance testing, or UAT to ensure whether the software functionalities serve the purpose of development.



Acceptance Testing
UAT Execution & Report Submission

Date	16 November 2022
Team ID	PNT2022TMID45981
Project Name	INVENTORY MANAGEMENT SYSTEM FOR RETAILERS
Maximum Marks	4 Marks

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [INVENTORY MANAGEMENT SYSTEM FOR RETAILERS] project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

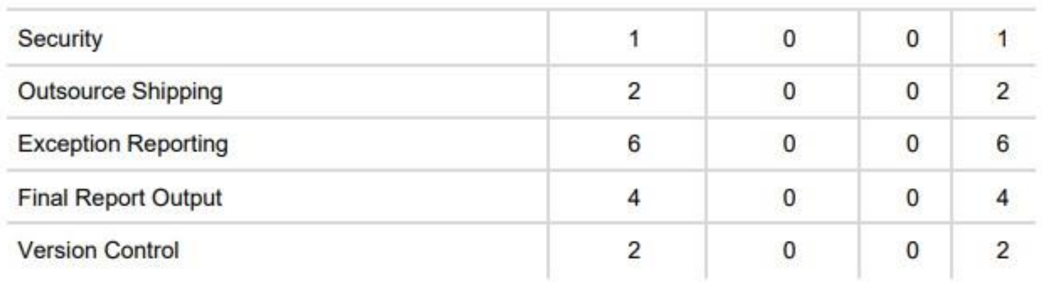
This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	10	2	1	2	15
Duplicate	0	0	3	0	3
External	2	3	1	0	6
Fixed	11	2	4	18	35
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	0	1
Won't Fix	0	4	2	0	6
Totals	23	11	13	20	74

3. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	10	0	0	10
Client Application	40	0	0	40

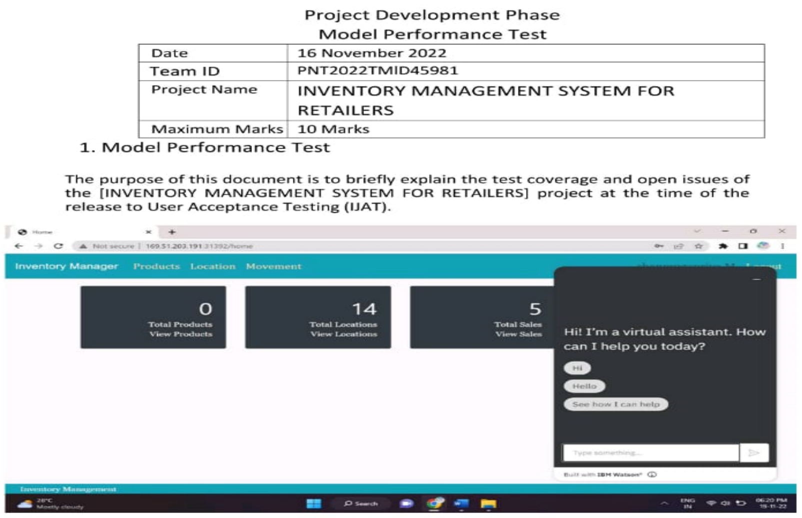


Security	1	0	0	1
Outsource Shipping	2	0	0	2
Exception Reporting	6	0	0	6
Final Report Output	4	0	0	4
Version Control	2	0	0	2

9. RESULTS

9.1 Performance Metrics

Metrics are a baseline for performance tests. Monitoring the correct parameters will help you detect areas that require increased attention and find ways to improve them.



Project Development Phase
Model Performance Test

Date	16 November 2022
Team ID	PNT2022TMID45981
Project Name	INVENTORY MANAGEMENT SYSTEM FOR RETAILERS
Maximum Marks	10 Marks

1. Model Performance Test

The purpose of this document is to briefly explain the test coverage and open issues of the [INVENTORY MANAGEMENT SYSTEM FOR RETAILERS] project at the time of the release to User Acceptance Testing (IJAT).

Not secure | 169.51.203.191:31392/home

Inventory Manager Products Location Movement

0
Total Products
View Products

14
Total Locations
View Locations

5
Total Sales
View Sales

Hi! I'm a virtual assistant. How can I help you today?

9. ADVANTAGES & DISADVANTAGES

Advantages:

- It helps to maintain the right amount of stocks
- It leads to a more organized warehouse
- It saves time and money
- Improves efficiency and productivity
- Flexibility

Disadvantages:

- Increased space is need to hold the inventory
- High implementation costs

- Some methods and strategies of inventory management can be relatively complex and difficult to understand
- Holding inventory can result to a greater risk of loss to devaluation (changes in price)

10. CONCLUSION

To conclude, Inventory Management System is a simple desktop based application basically suitable for small organization. It has basic items which are used for the small organization. Our team is successful in making the application where we can update, insert and delete the item as per the requirement. This application matches for small organizations where there are small limited if warehouses.Through it has some limitations, our team strongly believes that the implementation of this system will surely benefit the organization.

11. FUTURE SCOPE

Since this project was started with very little knowledge about the InventoryManagement System, we came to know about the enhancement capability during the process of building it. Some of the scope we can increase forthe betterment and effectiveness oar listed below:

- Interactive user interface design.
- Manage Stock Godown wise.
- Online payment system can be added.
- Making the system flexible in any type.
- Sales and purchase return system will be added in order to make return of products.
- Lost and breakage

12. APPENDIX

Source Code

app.py

```
from pickle import TRUE from
inventorymanagement import app from
flask import Flask, request, Response

app = Flask(__name__) if
__name__ == '__main__':
app.run(debug=TRUE)
```

__init__.py

```
from flask import Flask
from flask_sqlalchemy import SQLAlchemy
from flask_bcrypt import Bcrypt from
flask_login import LoginManager from
flaskext.mysql import MySQL
```

```
app = Flask(__name__)
app.config['SECRET_KEY'] = '82e65b56c16931a98ff8341e28059a89'

################User Login SQLAlchemy################

app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///site.db'
db = SQLAlchemy(app)

################Products MySQL################

app.config['MYSQL_DATABASE_USER'] = 'root'
app.config['MYSQL_DATABASE_PASSWORD'] = 'password'
app.config['MYSQL_DATABASE_DB'] =
'inventory_management'
app.config['MYSQL_DATABASE_HOST'] = 'localhost' mysql =
MySQL(app) print("Connection done") bcrypt = Bcrypt(app)
login_manager = LoginManager(app)

from inventorymanagement import routes #to avoid circular imports issue
```

form.py

```
from flask_wtf import FlaskForm from
inventorymanagement import app, mysql, db
from wtforms import StringField, PasswordField, SubmitField, BooleanField, IntegerField,
SelectField, DateField
from wtforms.validators import DataRequired, Length, Email, EqualTo, ValidationError
from inventorymanagement.models import User

######################Users####################################

class RegistrationForm(FlaskForm):
    username = StringField('Username', validators=[DataRequired(), Length(min=2, max=20)])
    email = StringField('Email', validators=[DataRequired(), Email()]) password =
    PasswordField('Password',         validators=[DataRequired()])         confirm_password         =
    PasswordField('Confirm Password', validators=[DataRequired(),
EqualTo('password')]) submit =
   SubmitField('Sign Up')

    def validate_username(self, username):
        user = User.query.filter_by(username = username.data).first()
        if user: raise ValidationError('Username Taken')

    def validate_email(self, email):
        user = User.query.filter_by(email = email.data).first()
        if user: raise ValidationError('Email Taken')

class LoginForm(FlaskForm):
    email = StringField('Email', validators=[DataRequired(), Email()])
    password = PasswordField('Password',
    validators=[DataRequired()]) remember = BooleanField('Remember
    Me') submit = SubmitField('Login')

#######################Products####################################

class AddProduct(FlaskForm):
    name = StringField('Product Name', validators=[DataRequired()])
    submit = SubmitField('Add Product')

#######################Locations####################################
class AddLocation(FlaskForm):
    name = StringField('Location Name', validators=[DataRequired()])
    submit = SubmitField('Add Location')

#######################ProductMovements###################################
#

class ProductMovement(FlaskForm):
    name = StringField('Product Name', validators=[DataRequired()])
    #timestamp = DateField('Date', validators=[DataRequired()])
    #fromLocation = SelectField('From Location', validators=[DataRequired()])
    #toLocation = SelectField('To Location', validators=[DataRequired()])
    #quantity = SelectField('Quantity', validators=[DataRequired()])
    #email = StringField('Email', validators=[DataRequired(), Email()])
    submit = SubmitField('Move Product')
models.py
from datetime import datetime
from inventorymanagement import db, login_manager
from flask_login import UserMixin

@login_manager.user_loader def
load_user(user_id): return
User.query.get(int(user_id))

class User(db.Model, UserMixin):
    user_id = db.Column(db.Integer, primary_key=True) username =
    db.Column(db.String(20), unique=True, nullable=False) email =
    db.Column(db.String(120), unique=True, nullable=False) password
    = db.Column(db.String(60), nullable=False)

    def __repr__(self): return
        f"User('{self.username}', '{self.email}')"

    def is_authenticated(self):
        return True

    def is_active(self):
        return True

    def is_anonymous(self):
        return True

    def get_id(self):
        return str(self.user_id)
```

routes.py

```
import datetime
from flask import render_template, url_for, flash, redirect, request
from inventorymanagement import app, bcrypt, mysql, db
from          inventorymanagement.forms        import   RegistrationForm,
    LoginForm,       AddProduct, AddLocation, ProductMovement from
inventorymanagement.models import User
from flask_login import login_user, current_user, logout_user, login_required from wtforms
import StringField, PasswordField, SubmitField, BooleanField, IntegerField, DateTimeField,
SelectField, Label
from wtforms.validators import DataRequired, Length, Email, EqualTo, ValidationError

@app.route("/")
@app.route("/anon") def anon():
return redirect(url_for('login'))

@app.route("/home")
def home():
  conn = mysql.connect()
  cursor = conn.cursor()
          cursor.execute("SELECT COUNT(product_id) FROM product WHERE user_id="+
str(current_user.user_id) +"")
  products = cursor.fetchone()
  cursor.execute("SELECT COUNT(location_id) FROM location")
  locations = cursor.fetchone()
         cursor.execute("SELECT COUNT(*) FROM productmovement WHERE user_id="+
str(current_user.user_id) +"")
  movements = cursor.fetchone()
  sales = 5
            return    render_template('home.html',    title='Home',    products=products[0],
locations=locations[0], sales=sales, movements=movements[0])

@app.route("/about") def about(): return
render_template('about.html', title="About")
```

```
########################Users######################################

@app.route("/register", methods=['GET', 'POST'])
def register():
    if current_user.is_authenticated:
        return redirect(url_for('home'))
    form = RegistrationForm() if
    form.validate_on_submit():
        hashed_password = bcrypt.generate_password_hash(form.password.data).decode('utf-8')
                            user    =    User(username=form.username.data,    email=form.email.data,
password=hashed_password)
        db.session.add(user)
        db.session.commit()
        flash('Your Account has been created', 'success')
        return redirect(url_for('login'))
    return render_template('register.html', title='Register', form=form)

@app.route("/login", methods=['GET', 'POST'])
def login():
    if current_user.is_authenticated:
        return redirect(url_for('home'))
    form = LoginForm() if
    form.validate_on_submit():
        user = User.query.filter_by(email=form.email.data).first() if user and
        bcrypt.check_password_hash(user.password, form.password.data):
            login_user(user, remember=form.remember.data)
            next_page = request.args.get('next')
            return redirect(next_page) if next_page else redirect(url_for('home'))
        else:
            flash('Login Unsuccessful', 'danger')
    return render_template('login.html', title='Login', form=form)

@app.route("/logout") def
logout(): logout_user() return
redirect(url_for('anon'))

########################Products######################################
```

```
@app.route("/add_product", methods=['GET', 'POST'])
@login_required
def add_product():
    conn = mysql.connect()
    cursor = conn.cursor()
    cursor.execute("SELECT location_name FROM
    location") locations = cursor.fetchall() places = [] for
    location in locations:
        places.append(location[0])

    form = AddProduct()

    if form.validate_on_submit():
        name = form.name.data
        input_values = request.form.getlist('places[]')
        totalquantity = 0

        locationinventory = "INSERT INTO `locationinventory`("

        for count,place in enumerate(places):
            locationinventory = locationinventory + "`"+ place +"`"
            if count != len(places)-1:
                locationinventory = locationinventory + ","
        locationinventory = locationinventory + ",`user_id`) VALUES ("

        for count,input_value in enumerate(input_values): totalquantity
            = totalquantity + int(input_value) locationinventory =
            locationinventory + "'"+ input_value +"'" if count !=
            len(input_values)-1: locationinventory = locationinventory +
            ","

        locationinventory = locationinventory + ",'"+ str(current_user.user_id) +"')" location =
                "INSERT INTO `product`(`product_name`,`product_quantity`,`user_id`)
VALUES ('"+ name +"','"+ str(totalquantity) +"','"+ str(current_user.user_id)
        +"')" print(locationinventory) cursor.execute(locationinventory)
        conn.commit() cursor.execute(location) conn.commit() conn.close()
        flash('Done', 'success')
        return redirect(url_for('view_product'))
    return render_template('add_product.html', title='Product', form=form, locations=locations)

@app.route("/edit_product?<int:product_id>", methods=['GET', 'POST'])
def edit_product(product_id): form = AddProduct() conn =
mysql.connect() cursor = conn.cursor()
    values = "Select * from product WHERE product_id="+ str(product_id)
    +"" values = cursor.execute(values) values = cursor.fetchone()
    locations = "SELECT location_name FROM
    location" locations = cursor.execute(locations)
    locations = cursor.fetchall() places = [] for location in
    locations: places.append(location[0])
        inventory = "Select * from locationinventory WHERE locationinventory_id="+
str(product_id) +""
    inventory =
    cursor.execute(inventory) inventory
    = cursor.fetchone() ranges =
    len(locations) quantities = [] for
    inventory in inventory:
        quantities.append(inventory)
    print(quantities) for
    count in range(2):
        quantities.pop(0)
    print(quantities) if
    form.validate_on_submit():
    name = form.name.data
        input_values =
        request.form.getlist('places[]')
        print(input_values) totalquantity = 0
        locationinventory = "UPDATE `locationinventory` SET "

        for index in range(ranges): locationinventory = locationinventory + "`" +
                    locations[index][0] + "`='" +
str(input_values[index]) +"'" totalquantity = totalquantity +
          int(input_values[index]) if index != len(input_values)-
          1: locationinventory = locationinventory + ","
```

```
                    locationinventory = locationinventory + " WHERE `locationinventory_id`=" +
str(product_id) location = "UPDATE `product` SET `product_name`='"+ name
     +"',`product_quantity`='"+
str(totalquantity) +"' WHERE product_id="+ str(product_id)
    print(locationinventory)
    cursor.execute(locationinventory) conn.commit()
    print(location) cursor.execute(location) conn.commit()
    conn.close()
    flash('Done', 'success')
    return redirect(url_for('view_product'))
         return render_template('edit_product.html', title='Product', form=form, values=values,
locations=locations, quantities=quantities, ranges=ranges)

@app.route("/product_info?<int:product_id>", methods=['GET', 'POST'])
def product_info(product_id): form = AddProduct() conn =
mysql.connect() cursor = conn.cursor()
  values = "Select * from product WHERE product_id="+ str(product_id)
  +"" values = cursor.execute(values) values = cursor.fetchone()
  locations = "SELECT location_name FROM
  location" locations = cursor.execute(locations)
  locations = cursor.fetchall() places = [] for location in
  locations:
    places.append(location[0])
      inventory = "Select * from locationinventory WHERE locationinventory_id="+
str(product_id) +""
  inventory =
  cursor.execute(inventory) inventory
  = cursor.fetchone() ranges =
  len(locations) quantities = [] for
  inventory in inventory:
    quantities.append(inventory)
  for count in range(2):
    quantities.pop(0)
         return render_template('product_info.html', title='Product', form=form, values=values,
locations=locations, quantities=quantities, ranges=ranges)
```

```
@app.route("/view_product")
@login_required def
view_product():
    conn = mysql.connect()
    cursor = conn.cursor()
            products   =   cursor.execute("SELECT   *   FROM   product   WHERE   user_id='"+
str(current_user.user_id)+"'")
    products = cursor.fetchall()
    inventory_places = cursor.execute("SELECT * FROM locationinventory") inventory_places
    = cursor.fetchall() return      render_template('view_product.html',          title='Product',
       products=products,
inventory_places=inventory_places)

#######################Locations####################################

@app.route("/add_location", methods=['GET', 'POST'])
@login_required
def add_location():
    form = AddLocation() if
    form.validate_on_submit():
    conn = mysql.connect()
    cursor = conn.cursor()
                              count = cursor.execute("SELECT location_name FROM location WHERE
location_name='"+ (form.name.data).replace(" ", "_") +"'")
        if count == 0:
                              cursor.execute("INSERT INTO `location`(`location_name`) VALUES ('"+
(form.name.data).replace("   ",   "_")   +"')")   conn.commit()   cursor.execute("ALTER   TABLE
           locationinventory ADD COLUMN "+
(form.name.data).replace(" ", "_") +" INTEGER DEFAULT
           0") conn.commit() conn.close()
           flash('Location Added', 'success')
           return
           redirect(url_for('view_location'))
        else:
           conn.close()
```

```
            flash('Location Exixts', 'danger')
            return
            redirect(url_for('add_location'))

    return render_template('add_location.html', title='Location', form=form)
    @app.route("/edit_location?<int:location_id>", methods=['GET', 'POST'])
@login_required def
edit_location(location_id):
form = AddLocation() conn =
mysql.connect() cursor =
conn.cursor()
    cursor.execute("SELECT * FROM location WHERE location_id='"+ str(location_id)
    +"'") location = cursor.fetchone() if form.validate_on_submit():
            cursor.execute("UPDATE location SET location_name='"+ form.name.data +"' WHERE
location_id='"+ str(location_id) +"'")
        conn.commit() flash('Updated!',
        'success') return
        redirect(url_for('view_location'))
    elif request.method == 'GET':
        form.name.data = location_id
    return render_template('edit_location.html', title='Location', form=form, location=location)

@app.route("/view_location")
@login_required def
view_location(): conn =
mysql.connect() cursor =
conn.cursor()
    cursor.execute("SELECT * FROM location")
    locations = cursor.fetchall()
    return render_template('view_location.html', title='Location', locations=locations)

#######################ProductMovements####################################
#

@app.route("/add_productmovement?<int:product_id>", methods=['GET', 'POST'])
@login_required def
add_productmovement(product_id):
    form = ProductMovement()
    conn = mysql.connect()
    cursor = conn.cursor()
    cursor.execute("SELECT
      product_name      FROM
      product  WHERE
      product_id="+
str(product_id) +"")
    product_name = cursor.fetchone()
    cursor.execute("SELECT location_name FROM
    location") locations = cursor.fetchall() time =
    datetime.date.today() ranges = len(locations)
        cursor.execute("SELECT * FROM locationinventory WHERE locationinventory_id="+
str(product_id) +"")
    inventory =
    cursor.fetchone() quantities
    = [] for inventory in
    inventory:
       quantities.append(inventory)
    for count in range(2):
       quantities.pop(0)
    print(quantities[5]) if
    form.validate_on_submit():
    product_name = form.name.data
       from_location =
       request.values.get('fromLocation') to_location =
       request.values.get('toLocation') quantity =
       request.values.get('quantity') date =
       request.values.get('timestamp') email =
       request.values.get('email')
       query = "SELECT "+ str(from_location) +","+ str(to_location) +" FROM locationinventory
WHERE locationinventory_id="+ str(product_id) +""
       cursor.execute(query) value = cursor.fetchone()
       from_location_qty = value[0] - int(quantity)
       to_location_qty = value[1] + int(quantity)
                   query   =   "UPDATE   locationinventory   SET   "+   str(from_location)   +"='"+
str(from_location_qty) +"', "+ str(to_location) +"='"+ str(to_location_qty)
    +"' WHERE locationinventory_id="+ str(product_id) +"" print(query)
cursor.execute(query) conn.commit() query = "INSERT INTO
`productmovement`(`product_id`, `product_name`, `from_location_name`, `to_location_name`,
    `product_quantity`, `timestamp`, `user_id`) VALUES ('"+
    str(product_id) +"','"+ form.name.data +"','"+ str(from_location) +"','"+
str(to_location) +"','"+ quantity +"','"+ date +"','"+ str(current_user.user_id) +"')" print(query)
cursor.execute(query) conn.commit() conn.close() flash('Updated!', 'success') return
redirect(url_for('view_location')) return render_template('add_productmovement.html',
title='Movement', form=form, time=time, email=current_user.email,
product_name=product_name[0], locations=locations, quantities=quantities, ranges=ranges)

@app.route("/edit_productmovement?<int:productmovement_id>")
@login_required def
edit_productmovement(productmovement_id):
  conn = mysql.connect()
  cursor = conn.cursor()
    cursor.execute("SELECT * FROM productmovement WHERE productmovement_id="+
str(productmovement_id) +"")
  query = cursor.fetchone() product_id = query[1] from_location = query[3] to_location =
  query[4] quantity = query[5] cursor.execute("SELECT "+ str(from_location) +","+
  str(to_location) +" FROM
locationinventory WHERE locationinventory_id="+ str(product_id) +"")
  inventory = cursor.fetchone()
  from_location_qty = inventory[0] + quantity
  to_location_qty = inventory[1] - quantity
        cursor.execute("UPDATE locationinventory SET "+ str(from_location)
    +"='"+ str(from_location_qty) +"', "+ str(to_location) +"='"+
    str(to_location_qty) +"' WHERE locationinventory_id="+ str(product_id) +"")
  conn.commit()
     cursor.execute("DELETE FROM productmovement WHERE productmovement_id="+
str(productmovement_id) +"")
  conn.commit()
  return redirect(url_for('view_productmovement'))
  return render_template('', title='Movement', form=form)
```

```
@app.route("/view_productmovement")
@login_required def
view_productmovement():
conn = mysql.connect()
cursor = conn.cursor()
        cursor.execute("SELECT * FROM productmovement WHERE user_id="+
str(current_user.user_id) +"")
  movements = cursor.fetchall()
  counts = len(movements)
            return render_template('view_productmovement.html', title='Movement',
movements=movements, counts=counts)
```

about.html

```
{% extends "layout.html" %}
{% block content %}
    <div class="col-md-12">
        <h2>About us</h2>
    </div>
{% endblock content %}
```

```
add_location.html
{% extends "layout.html" %}
{% block content %}
  <div class="col-md-6 offset-md-3">
    <div class="huge">Add Location</div>
  </div>
  <div class="col-md-6 offset-md-3 border pt-2">
    <br>
    <div class="row">
      <div class="col-md-12">
        <form method="POST" action="">
        {{ form.hidden_tag() }} <!--CSRF TOKEN -->
          <fieldset class="form-group">
            <div class="form-group">
```

```
                    {{ form.name.label(class="form-control-label") }}
                    {{ form.name(class="form-control form-control-lg") }}
                </div>
            </fieldset>
            <div class="form-group">
                {{ form.submit(class="btn btn-success navbar-dark bg-dark") }}
            </div>
        </form>
      </div>
    </div>
  </div> {% endblock
content %}
```

add_product.html

```
{% extends "layout.html" %}
{% block content %}
  <div class="col-md-6 offset-md-3">
    <div class="huge">Add Product</div>
  </div>
  <div class="col-md-6 offset-md-3 border pt-2">
    <div class="row">
      <div class="col-md-12">
        <form method="POST" action="">
        {{ form.hidden_tag() }} <!--CSRF TOKEN -->
            <fieldset class="form-group">
                <div class="form-group">
                    {{ form.name.label(class="form-control-label") }}
                    {{ form.name(class="form-control form-control-lg") }}
                </div>
            </fieldset>
      </div>
    </div>
    <div class="row">
      <div class="col-md-12">
        <label class="form-control-label">Locations</label>
      </div>
      <div class="col-md-12" style="height:400px; overflow: auto;">
```

```
                    <fieldset>
                        {% for location in locations %}
                            <div class="form-group">
                                <label class="form-control-label">{{ location[0] }}</label>
                                 <input class="form-control form-control-lg" id="places[]" name="places[]"
value="0">
                            </div>
                        {% endfor %}
                    </fieldset>
            </div>
        </div>
        <br>
        <div class="row">
            <div class="col-md-12">
                    <fieldset>
                        <div class="form-group">
                            {{ form.submit(class="btn btn-success navbar-dark bg-dark") }}
                        </div>
                    </fieldset>
                </form>
            </div>
        </div>
    </div>
{% endblock content %}
```

add_productmovement.html

```
{% extends "layout.html" %}
{% block content %}
    <div class="col-md-6 offset-md-3">
        <div class="huge">Move Product</div>
    </div>
    <div class="col-md-6 offset-md-3 border pt-2">
        <br>
        <div class="row">
            <div class="col-md-12">
                <form method="POST" action="">
                {{ form.hidden_tag() }} <!--CSRF TOKEN -->
                <fieldset class="form-group">
                    <div class="form-group">
                        <label class="form-control-label">Product Name</label>
                                <input name="name" id="name" class="form-control form-control-lg"
value="{{ product_name }}" readonly>
                    </div>
                    <div class="form-group">
                                    <label class="form-control-label">Product Quantities at Respective
Locations</label>
                    </div>
                    <div class="form-group" style="height:100px;overflow: auto;">
                        {% for index in range(ranges) %}
                            {% if quantities[index] != 0 %}
                                <div class="form-group">
                                    <input class="form-control form-control-md" value="Quantity at {{
locations[index][0] }} location is {{ quantities[index] }}" disabled>
                                </div>
                            {% endif %}
                        {% endfor %}
                    </div>
                    <div class="form-group">
                        <label class="form-control-label">From Location</label>
                                <select name="fromLocation" id="fromLocation" class="form-control
form-control-lg">
                        {% for index in range(ranges) %}
                            {% if quantities[index] != 0%}
                                <option value="{{ locations[index][0] }}" data-qty="{{ quantities[index]
}}">{{ locations[index][0] }}</option>
                            {% endif %}
                        {% endfor %}
                        </select>
                    </div>
                    <div class="form-group">
                        <label class="form-control-label">To Location</label>
                                    <select name="toLocation" id="toLocation" class="form-control
form-control-lg">
                        {% for index in range(ranges) %}
```

```
                                        <option value="{{ locations[index][0] }}">{{ locations[index][0]
}}</option>
                    {% endfor %}
                    </select>
                  </div>
                  <div class="form-group">
                    <label class="form-control-label">Quantity</label>
                    <select name="quantity" id="quantity" class="form-control form-control-lg">

                    </select>
                  </div>
                  <div class="form-group">
                    <label class="form-control-label">Date</label>
                                          <input name="timestamp" id="timestamp" class="form-control
form-control-lg" value="{{ time }}" readonly>
                  </div>
                  <div class="form-group">
                    <label class="form-control-label">Email</label>
                              <input name="email" id="email" class="form-control form-control-lg"
value="{{ email }}" readonly>
                  </div>
                </fieldset>
                <div class="form-group">
                  {{ form.submit(class="btn btn-success navbar-dark bg-dark") }}
                </div>
              </form>
            </div>
          </div>
        </div>
        <script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js"></script>
        <script language="JavaScript" type="text/javascript">
          $("#fromLocation").change(function () { value =
            $(this).find(':selected').data('qty'); select =
            document.getElementById('quantity');
            $('#quantity').html(' ');
            append = '';
            for (i = 1; i < value+1; i++) {
                append = append + '<option id="remove" value="'+ i +'">'+ i +'</option>';
            }
            select.innerHTML += append;
        });
    </script>
    <script type="text/javascript">
        $(document).ready(function(){ value =
            $(this).find(':selected').data('qty'); select =
            document.getElementById('quantity'); append
            = '';
            for (i = 1; i < value+1; i++) {
                append = append + '<option id="remove" value="'+ i +'">'+ i +'</option>';
            }
            select.innerHTML += append;
            });
    </script>
{% endblock content %}
```

edit location.html

```
{% extends "layout.html" %}
{% block content %}
    <div class="col-md-6 offset-md-3">
        <div class="huge">Edit Location</div>
    </div>
    <div class="col-md-6 offset-md-3 border pt-2">
        <br>
        <div class="row">
            <div class="col-md-12">
                <form method="POST" action="">
                {{ form.hidden_tag() }} <!--CSRF TOKEN -->
                    <fieldset class="form-group">
                        <div class="form-group">
                            <label class="form-control-label" for="name">Edit Location Name</label>
                                    <input class="form-control form-control-lg" id="name" name="name"
```

```
required type="text" value="{{ location[1] }}">
                        </div>
                    </fieldset>
                    <div class="form-group">
                           <input type="submit" class="btn btn-success navbar-dark bg-dark" value="Edit
Location">
                    </div>
                </form>
            </div>
        </div>
    </div>
{% endblock content %}
```

edit_product.html

```
{% extends "layout.html" %}
{% block content %}
    <div class="col-md-6 offset-md-3">
        <div class="huge">Edit Product</div>
    </div>
    <div class="col-md-6 offset-md-3 border pt-2">
        <div class="row">
            <div class="col-md-12">
                <form method="POST" action="">
                {{ form.hidden_tag() }} <!--CSRF TOKEN -->
                    <fieldset class="form-group">
                        <div class="form-group">
                            <label class="form-control-label">Product Name</label>
                                        <input class="form-control form=control-lg" name="name" id="name"
value="{{ values[1] }}">
                        </div>
                    </fieldset>
            </div>
        </div>
        <div class="row">
            <div class="col-md-12">
                <label class="form-control-label">Locations</label>
            </div>
            <div class="col-md-12" style="height:400px; overflow: auto;">
                    <fieldset>
                        {% for index in range(ranges) %}
                            <div class="form-group">
                                <label class="form-control-label">{{ locations[index][0] }}</label>
                                 <input class="form-control form-control-lg" id="places[]" name="places[]"
value="{{ quantities[index] }}">
                            </div>
                        {% endfor %}
                    </fieldset>
            </div>
        </div>
        <br>
        <div class="row">
            <div class="col-md-12">
                    <fieldset>
                        <div class="form-group">
                                                <input type="submit" class="btn btn-success navbar-dark bg-dark"
value="Edit Product">
                        </div>
                    </fieldset>
                </form>
            </div>
        </div>
    </div>
{% endblock content %}
```

home.html

```
{% extends "layout.html" %}
{% block content %}
    <div class="col-md-12">
        <div class="row">
            <div class="col-lg-3 col-md-6">
                <a href="{{ url_for('view_product') }}" style="text-decoration:none; color: white;">
    <div class="border rounded navbar-dark bg-dark">
        <div class="panel-heading">
            <br>
            <div class="row">
                <div class="col-md-8 offset-md-3 text-right">
                    <div class="huge">{{ products }}</div>
                    <div>Total Products</div>
                    <div>View Products</div>
                </div>
            </div>
            <br>
        </div>
    </div>
    </a>
</div>
<div class="col-lg-3 col-md-6">
    <a href="{{ url_for('view_location') }}" style="text-decoration:none; color: white;">
    <div class="border rounded navbar-dark bg-dark">
        <div class="panel-heading">
            <br>
            <div class="row">
                <div class="col-md-8 offset-md-3 text-right">
                    <div class="huge">{{ locations }}</div>
                    <div>Total Locations</div>
                    <div>View Locations</div>
                </div>
            </div>
            <br>
        </div>
    </div>
    </a>
</div>
<div class="col-lg-3 col-md-6">
    <a href="#" style="text-decoration:none; color: white;">
    <div class="border rounded navbar-dark bg-dark">
        <div class="panel-heading">
            <br>
```

```
                <div class="row">
                    <div class="col-md-8 offset-md-3 text-right">
                        <div class="huge">{{ sales }}</div>
                        <div>Total Sales</div>
                        <div>View Sales</div>
                    </div>
                </div>
                <br>
            </div>
        </div>
        </a>
    </div>
    <div class="col-lg-3 col-md-6">
         <a href="{{ url_for('view_productmovement') }}" style="text-decoration:none; color:
white;">
        <div class="border rounded navbar-dark bg-dark">
            <div class="panel-heading">
                <br>
                <div class="row">
                    <div class="col-md-8 offset-md-3 text-right">
                        <div class="huge">{{ movements }}</div>
                        <div>Total Movement</div>
                        <div>Product Movement</div>
                    </div>
                </div>
                <br>
            </div>
        </div>
        </a>
    </div>
  </div>
</div>
{% endblock content %}
```

layout.html

```
<!doctype html>
<html lang="en">
```

```
<head>
  <!-- Required meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <!-- Bootstrap CSS -->
  <link rel="stylesheet" href="static/css/bootstrap.min.css">

  <!-- Sb Admin CSS -->
  <link rel="stylesheet" href="static/css/sb-admin-2.css">

    {% if title%}
      <title>{{title}}</title>
    {% else %}
      <title>Inventory Manager</title>
    {% endif %}
  </head>
  <body>
    <nav class="navbar navbar-expand-lg navbar-dark bg-dark fixed-top">
      {% if current_user.is_authenticated %}
        <a class="navbar-brand" href="{{ url_for('home') }}">Inventory Manager</a>
      {% else %}
        <a class="navbar-brand" href="{{ url_for('anon') }}">Inventory Manager</a>
      {% endif %}
                    <button   class="navbar-toggler"    type="button" data-toggle="collapse"
data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
      </button>

      <div class="collapse navbar-collapse" id="navbarSupportedContent">
        <ul class="navbar-nav mr-auto">
          {% if current_user.is_authenticated %}
            <li class="nav-item">
              <a class="nav-link" href="{{ url_for('view_product') }}">Products</a>
            </li>
            <li class="nav-item">
```

```
                <a class="nav-link" href="{{ url_for('view_location') }}">Location</a>
              </li>
              <li class="nav-item">
                              <a class="nav-link" href="{{ url_for('view_productmovement')
}}">Movement</a>
              </li>
            {% else %}
              <li class="nav-item">
                <a class="nav-link" href="{{ url_for('about') }}">About</a>
              </li>
            {% endif %}
          </ul>
          <ul class="navbar-nav my-2 my-lg-0">
            {% if current_user.is_authenticated %}
              <li class="nav-item">
                <a class="nav-link" href="#">Hi, {{ current_user.email }}</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="{{ url_for('logout') }}">Logout</a>
              </li>
            {% else %}
              <li class="nav-item">
                <a class="nav-link" href="{{ url_for('register') }}">Register</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="{{ url_for('login') }}">Login</a>
              </li>
            {% endif %}
          </ul>
        </div>
      </nav>
      <div class="container-fluid" style="margin-top:80px;">
        <div class="row">
          <div class="col-lg-10 col-md-10 offset-lg-1 offset-md-1">
            <div class="row">
              {% with messages = get_flashed_messages(with_categories=true) %}
                {% if messages %}
                  {% for category, message in messages%}
                    <div class="alert alert-{{category}}">
                      {{ message }}
                    </div>
                  {% endfor %}
                {% endif %}
              {% endwith %}
              {% block content %}
              {% endblock %}
            </div>
          </div>
        </div>
      </div>
          <div class="footer navbar-dark bg-dark" style="position: fixed;left: 0;bottom: 0;width:
100%;color: white;">
        <div class="row">
          <div class="col-md-2">
            <small class="text-muted">Inventory Management</small>
          </div>
        </div>
      </div>


         <!-- jQuery first, then Popper.js, then Bootstrap JS -->
                                      <script      src="static/js/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi
6jizo" crossorigin="anonymous"></script>
                                           <script      src="static/js/popper.min.js"
integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvC
WPIPm49" crossorigin="anonymous"></script>
                                          <script      src="static/js/bootstrap.min.js"
integrity="sha384-ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ
5stwEULTy" crossorigin="anonymous"></script>
  </body>
</html>
```

login.html

```
{% extends "layout.html" %}
{% block content %}
    <div class="col-md-6 offset-md-3 border pt-2">
        <br>
        <div class="row">
            <div class="col-md-12">
                <form method="POST" action="">
                {{ form.hidden_tag() }} <!--CSRF TOKEN -->
                    <fieldset class="form-group">
                        <div class="form-group">
                            {{ form.email.label(class="form-control-label") }}

                            {% if form.email.errors %}
                                {{ form.email(class="form-control form-control-lg is-invalid") }}
                                <div class="invalid-feedback">
                                    {% for error in form.username.errors %}
                                        <span>{{ error }}</span>
                                    {% endfor %}
                                </div>
                            {% else %}
                                {{ form.email(class="form-control form-control-lg") }}
                            {% endif %}
                        </div>
                        <div class="form-group">
                            {{ form.password.label(class="form-control-label") }}

                            {% if form.password.errors %}
                                {{ form.password(class="form-control form-control-lg is-invalid") }}
                                <div class="invalid-feedback">
                                    {% for error in form.password.errors %}
                                        <span>{{ error }}</span>
                                    {% endfor %}
                                </div>
                            {% else %}
                {{ form.password(class="form-control form-control-lg") }}
              {% endif %}
            </div>
            <div class="form-check">
              {{ form.remember(class="form-check-input") }}
              {{ form.remember.label(class="form-check-label") }}
            </div>
          </fieldset>
          <div class="form-group">
            {{ form.submit(class="btn btn-success navbar-dark bg-dark") }}
          </div>
        </form>
      </div>
      <div class="col-md-12 border-top pt-2">
        <small class="text-muted">
           D'ont Have an Account ? <a class="ml-2" href="{{ url_for('register') }}">Register
Now</a>
        </small>
      </div>
    </div>
  </div>
{% endblock content %}
```

product_info.html

```
{% extends "layout.html" %}
{% block content %}
  <div class="col-md-6 offset-md-3">
    <div class="huge">View Product</div>
  </div>
  <div class="col-md-6 offset-md-3 border pt-2">
    <div class="row">
      <div class="col-md-12">
        {{ form.hidden_tag() }} <!--CSRF TOKEN -->
          <fieldset class="form-group">
            <div class="form-group">
              <label class="form-control-label">Product Name</label>
                    <input class="form-control form=control-lg" name="name" id="name"
```

```
value="{{ values[1] }}" disabled>
                    </div>
                  </fieldset>
              </div>
          </div>
          <div class="row">
              <div class="col-md-12">
                  <label class="form-control-label">Locations</label>
              </div>
              <div class="col-md-12" style="height:400px; overflow: auto;">
                    <fieldset>
                      {% for index in range(ranges) %}
                          <div class="form-group">
                              <label class="form-control-label">{{ locations[index][0] }}</label>
                               <input id="places[]" name="places[]" class="form-control form-control-lg"
value="{{ quantities[index] }}" disabled>
                          </div>
                      {% endfor %}
                    </fieldset>
              </div>
          </div>
          <br>
          <div class="row">
              <div class="col-md-12">
                  <div class="form-group">
                       <a class="btn btn-success navbar-dark bg-dark" href="{{ url_for('view_product')
}}">Back to Products</a>
                  </div>
              </div>
          </div>
      </div>
{% endblock content %}
```

register.html

```
{% extends "layout.html" %}
{% block content %}
    <div class="col-md-6 offset-md-3 border pt-2">
        <br>
        <div class="row">
            <div class="col-md-12">
                <form method="POST" action="">
                {{ form.hidden_tag() }} <!--CSRF TOKEN -->
                    <fieldset class="form-group">
                        <div class="form-group">
                            {{ form.username.label(class="form-control-label") }}

                            {% if form.username.errors %}
                                {{ form.username(class="form-control form-control-lg is-invalid") }}
                                <div class="invalid-feedback">
                                    {% for error in form.username.errors %}
                                        <span>{{ error }}</span>
                                    {% endfor %}
                                </div>
                            {% else %}
                                {{ form.username(class="form-control form-control-lg") }}
                            {% endif %}
                        </div>
                        <div class="form-group">
                            {{ form.email.label(class="form-control-label") }}

                            {% if form.email.errors %}
                                {{ form.email(class="form-control form-control-lg is-invalid") }}
                                <div class="invalid-feedback">
                                    {% for error in form.username.errors %}
                                        <span>{{ error }}</span>
                                    {% endfor %}
                                </div>
                            {% else %}
                                {{ form.email(class="form-control form-control-lg") }}
                            {% endif %}
                        </div>
                        <div class="form-group">
                            {{ form.password.label(class="form-control-label") }}

                            {% if form.password.errors %}
                                {{ form.password(class="form-control form-control-lg is-invalid") }}
                                <div class="invalid-feedback">
                                    {% for error in form.password.errors %}
                                        <span>{{ error }}</span>
                                    {% endfor %}
                                </div>
                            {% else %}
                                {{ form.password(class="form-control form-control-lg") }}
                            {% endif %}
                        </div>
                        <div class="form-group">
                            {{ form.confirm_password.label(class="form-control-label") }}

                            {% if form.confirm_password.errors %}
                                 {{ form.confirm_password(class="form-control form-control-lg is-invalid")
}}
                                <div class="invalid-feedback">
                                    {% for error in form.confirm_password.errors %}
                                        <span>{{ error }}</span>
                                    {% endfor %}
                                </div>
                            {% else %}
                                {{ form.confirm_password(class="form-control form-control-lg") }}
                            {% endif %}
                        </div>
                    </fieldset>
                    <div class="form-group">
                        {{ form.submit(class="btn btn-success navbar-dark bg-dark") }}
                    </div>
                </form>
            </div>
            <div class="col-md-12 border-top pt-2">
                <small class="text-muted">
                            Already Have an Account ? <a class="ml-2" href="{{ url_for('login') }}">Sign
```

```
In</a>
                </small>
            </div>
        </div>
    </div>
{% endblock content %}

view_location.html
{% extends "layout.html" %}
{% block content %}
    <div class="col-md-12">
        <div class="row">
            <div class="col-md-12">
                <small class="text-muted">
                    <a href="{{ url_for('home') }}">Inventory Manager </a>
                    / Location
                </small>
            </div>
        </div>
    </div>
    <div class="col-md-12">
        <div class="row">
            <div class="col-md-12 border rounded pt-2">
                <div class="row">
                    <div class="col-md-10">
                        <h2>Inventory Location</h2>
                    </div>
                    <div class="col-md-2 float-right">
                        {% set href = url_for('add_location') %}
                                                   <button class="btn btn-success navbar-dark bg-dark" onclick="
window.location.href='{{ href }}';">Add Location</button>
                    </div>
                </div>
            </div>
        </div>
    </div>
    <div class="col-md-12">
    <br>
    <div class="row">
      <div class="col-md-12">
        <div class="row">
          <div class="col-md-12">
            <div class="row">
              <div class="col-md-12 border rounded pt-2" style=" height:50px;">
                <div class="row">
                  <div class="col border-left pt-2">
                    Location Id
                  </div>
                  <div class="col border-left pt-2">
                    Location Name
                  </div>
                  <div class="col border-left pt-2">
                    Edit
                  </div>
                </div>
              </div>
            </div>
            <br>
          </div>
          <div class="col-md-12" style="height:600px; overflow: auto;">
          {% for location in locations %}
            <div class="row">
              <div class="col-md-12 border rounded pt-2" style=" height:50px;">
                <div class="row">
                  <div class="col border-left pt-2">
                    {{ location[0] }}
                  </div>
                  <div class="col border-left pt-2">
                    {{ location[1] }}
                  </div>
                  <div class="col border-left pt-2">
                              <a href="{{ url_for('edit_location', location_id=location[0])
}}">Edit</a>
                  </div>
```

```
                                        </div>
                                    </div>
                                </div>
                            {% endfor %}
                            </div>
                        </div>
                    </div>
                </div>
                <br>
            </div>
{% endblock content %}
```

view_product.html

```
{% extends "layout.html" %}
{% block content %}
    <div class="col-md-12">
        <div class="row">
            <div class="col-md-12">
                <small class="text-muted">
                    <a href="{{ url_for('home') }}">Inventory Manager </a>
                    / Product
                </small>
            </div>
        </div>
    </div>
    <div class="col-md-12">
        <div class="row">
            <div class="col-md-12 border rounded pt-2">
                <div class="row">
                    <div class="col-md-10">
                        <h2>Inventory Product</h2>
                    </div>
                    <div class="col-md-2 float-right">
                        {% set href = url_for('add_product') %}
                                                            <button class="btn btn-success navbar-dark bg-dark" onclick="
window.location.href='{{ href }}';">Add Product</button>
                    </div>
            </div>
         </div>
      </div>
   </div>
   <div class="col-md-12">
      <br>
      <div class="row">
         <div class="col-md-12">
            <div class="row">
               <div class="col-md-12">
                  <div class="row">
                     <div class="col-md-12 border rounded pt-2" style=" height:50px;">
                        <div class="row">
                           <div class="col pt-2">
                              Product Id
                           </div>
                           <div class="col border-left pt-2">
                              Name
                           </div>
                           <div class="col border-left pt-2">
                              Quantity
                           </div>
                           <div class="col border-left pt-2">
                              View Quantities
                              </div>
                           <div class="col border-left pt-2">
                              Edit
                           </div>
                           <div class="col border-left pt-2">
                              Move
                           </div>
                        </div>
                     </div>
                  </div>
                  <br>
               </div>
               <div class="col-md-12" style="height:600px; overflow: auto;">
```

```
                        {% for product in products %}
                            <div class="row">
                                <div class="col-md-12 border rounded pt-2" style=" height:50px;">
                                    <div class="row">
                                        <div class="col pt-2">
                                            {{ product[0] }}
                                        </div>
                                        <div class="col border-left pt-2">
                                            {{ product[1] }}
                                        </div>
                                        <div class="col border-left pt-2">
                                            {{ product[2] }}
                                        </div>
                                        <div class="col border-left pt-2">
                                                    <a href="{{ url_for('product_info', product_id=product[0])
}}">View Quantities</a>
                                        </div>
                                        <div class="col border-left pt-2">
                                                    <a href="{{ url_for('edit_product', product_id=product[0])
}}">Edit</a>
                                        </div>
                                        <div class="col border-left pt-2">
                                            <a href="{{ url_for('add_productmovement', product_id=product[0])
}}">Move</a>
                                        </div>
                                    </div>
                                </div>
                            </div>
                        {% endfor %}
                    </div>
                </div>
            </div>
        </div>
        <br>
    </div>
{% endblock content %}
```

product_movement.html

```
{% extends "layout.html" %}
{% block content %}
    <div class="col-md-12">
        <div class="row">
            <div class="col-md-12">
                <small class="text-muted">
                    <a href="{{ url_for('home') }}">Inventory Manager </a>
                    / Product
                </small>
            </div>
        </div>
    </div>
    <div class="col-md-12">
        <div class="row">
            <div class="col-md-12 border rounded pt-2">
                <div class="row">
                    <div class="col-md-10">
                        <h2>Inventory Product Movement</h2>
                    </div>
                </div>
            </div>
        </div>
    </div>
    <div class="col-md-12">
        <br>
        <div class="row">
            <div class="col-md-12">
                <div class="row">
                    <div class="col-md-12">
                        <div class="row">
                            <div class="col-md-12 border rounded pt-2" style=" height:50px;">
                                <div class="row">
                                    <div class="col pt-2">
                                        Product Id
                                    </div>
                                    <div class="col border-left pt-2">
                                        Name
                                    </div>
                                    <div class="col border-left pt-2">
                                        From Location
                                    </div>
                                    <div class="col border-left pt-2">
                                        To Location
                                    </div>
                                    <div class="col border-left pt-2">
                                        Quantity
                                    </div>
                                    <div class="col border-left pt-2">
                                        Date
                                    </div>
                                    <div class="col border-left pt-2">
                                        Revert
                                    </div>
                                </div>
                            </div>
                        </div>
                        <br>
                    </div>
                    <div class="col-md-12" style="height:600px; overflow: auto;">
                        {% for count in range(counts) %}
                            <div class="row">
                                <div class="col-md-12 border rounded pt-2" style=" height:50px;">
                                    <div class="row">
                                        <div class="col pt-2">
                                            {{ movements[count][1] }}
                                        </div>
                                        <div class="col border-left pt-2">
                                            {{ movements[count][2] }}
                                        </div>
                                        <div class="col border-left pt-2">
                                            {{ movements[count][3] }}
                                        </div>
                                        <div class="col border-left pt-2">
                                            {{ movements[count][4] }}
                                        </div>
                                        <div class="col border-left pt-2">
                                            {{ movements[count][5] }}
                                        </div>
                                        <div class="col border-left pt-2">
                                            {{ movements[count][6] }}
                                        </div>
                                        <div class="col border-left pt-2">
                                                                <a href="{{ url_for('edit_productmovement',
                    productmovement_id=movements[count][0]) }}">Revert</a>
                                        </div>
                                    </div>
                                </div>
                            </div>
                        {% endfor %}
                    </div>
                  </div>
                </div>
              </div>
              <br>
            </div>
                    {% endblock content %}
```

DockerFile

```
FROM python:3.7

COPY    requirements.txt /opt/python/requirements.txt
RUN pip install -r /opt/python/requirements.txt \
&& rm -rf /opt/python && pip install gunicorn==20.1.0
COPY app /opt/app
EXPOSE 8000
WORKDIR /opt/app
```

```
CMD gunicorn --bind 0.0.0.0:8000 --access-logfile - --error-logfile - run:app
```

docker-compose.yaml

```
version: "3.1"

services:
  db: image:
    mariadb:10.5 ports:
      -         "3306:3306"volu
    mes:
      -         "./inventory_man
    agement.sql:/docker-
    entrypoint-
    initdb.d/inventory_manage
    ment.sql"environment:
      MARIADB_RANDOM_ROOT_PASSWORD: yes
      MARIADB_DATABASE: inventory_management
      MARIADB_USER: admin
      MARIADB_PASSWORD: password

  inventory-management:
    build: .
    ports:
      -         "8000:8000"envi
    ronment:
      MYSQL_DATABASE_USER: admin
      MYSQL_DATABASE_PASSWORD: password
      MYSQL_DATABASE_DB: inventory_management
      MYSQL_DATABASE_HOST: db
                                        SQLALCHEMY_DATABASE_URI:
mysql+pymysql://admin:password@db/inventory_management
```

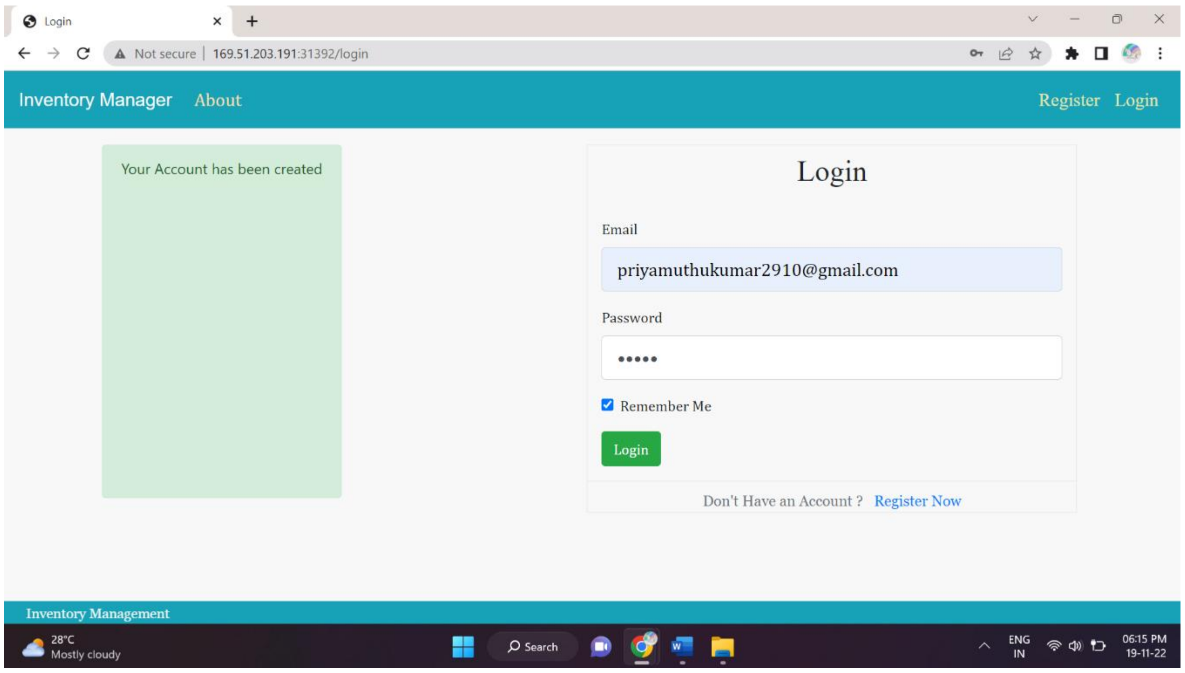
app.yml

```
---

apiVersion: apps/v1
kind: Deployment
metadata: name:
inventory-mgmt
spec:
  selector:
    matchLabels:
      app: inventory-mgmt
  strategy: type:
    Recreate
  template:
    metadata:
      labels: app:
        inventory-mgmt
    spec:
      containers:
           - image: jp.icr.io/inventory1/inventory-
             management:0.0.2 imagePullPolicy:
             IfNotPresent name: inventory-mgmt env:
           - name:
             MYSQL_DATABASE_HOSTvalue:
             "mysql"
           - name: MYSQL_DATABASE_DBvalue:
             "inventory_management"
           - name:
             MYSQL_DATABASE_USERvalue:
             "admin"
           - name:
             MYSQL_DATABASE_PASSWORDvalu
             e: "password"
           - name:
             SQLALCHEMY_DATABASE_URI
             value: mysql+pymysql://admin:password@mysql/inventory_management
         ports:
           - containerPort: 8000 name: http
```

```
          resources:
            requests:
              memory: "256Mi"
                      cpu: "500m"
--apiVersion:
v1 kind:
Service
metadata:
name: app
spec:
  ports:
    -  port: 8000
   selector: app:
   inventory-mgmt

--apiVersion:
v1 kind:
Service
metadata:
name: app-np
spec:
  ports:
    -  port: 8000
   selector: app:
   inventory-mgmt
   type: NodePort

mysql.yml
--apiVersion:
apps/v1 kind:
Deployment
metadata: name:
mysql
spec: selector:
  matchLabels:
  app: mysql
  strategy: type:
    Recreate
  template:
    metadata: labels:

        app: mysql
    spec:
      containers:
            - image: mariadb:10.5 name: mysql
              env:
            - name:
              MARIADB_RANDOM_ROOT_PAS
              SWORDvalue: "true"
            - name:
              MARIADB_DATABASEvalue:
              "inventory_management"
            - name: MARIADB_USERvalue:
              "admin"
            - name:
              MARIADB_PASSWORDvalue:
              "password"
          ports:
            - containerPort: 3306 name: mysql
          resources:
            requests:
              memory: "256Mi"

cpu: "250m"
            limits:
              memory: "1024Mi"
              cpu: "1000m"
          volumeMounts:
            # - name: mysql-persistent-storage
            #   mountPath: /var/lib/mysql
            - name: inventory-management-
              datamountPath: /docker-entrypoint-
              initdb.d/inventory_management.sql
              subPath: inventory_management.sql
```

```
   volumes:
    # - name: mysql-persistent-storage
    #   persistentVolumeClaim:
    #     claimName: mysql-pv-claim
        - name: inventory-management-data
      configMap:
        name: inventory-management-data
--apiVersion:
v1 kind:
Service
metadata:
name: mysql
spec:
 ports:
        - port: 3306 selector:
  app: mysql
--apiVersion: v1 kind: ConfigMap
metadata: name: inventory-
management-data
data:
 inventory_management.sql: |
```

Output



Login

Not secure | 169.51.203.191:31392/login

Inventory Manager About Register Login

Your Account has been created

Login

Email

priyamuthukumar2910@gmail.com

Password

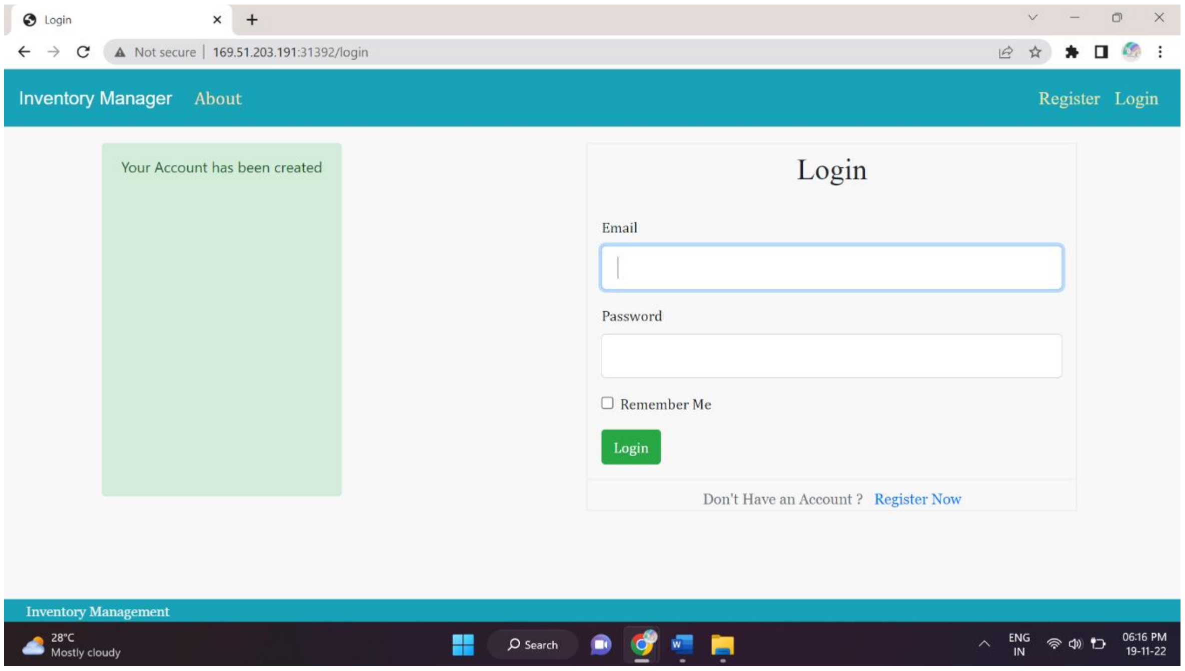
•••••

Remember Me

Login

Don't Have an Account ? Register Now

Inventory Management



Login

Not secure | 169.51.203.191:31392/login

Inventory Manager About Register Login

Your Account has been created

Login

Email

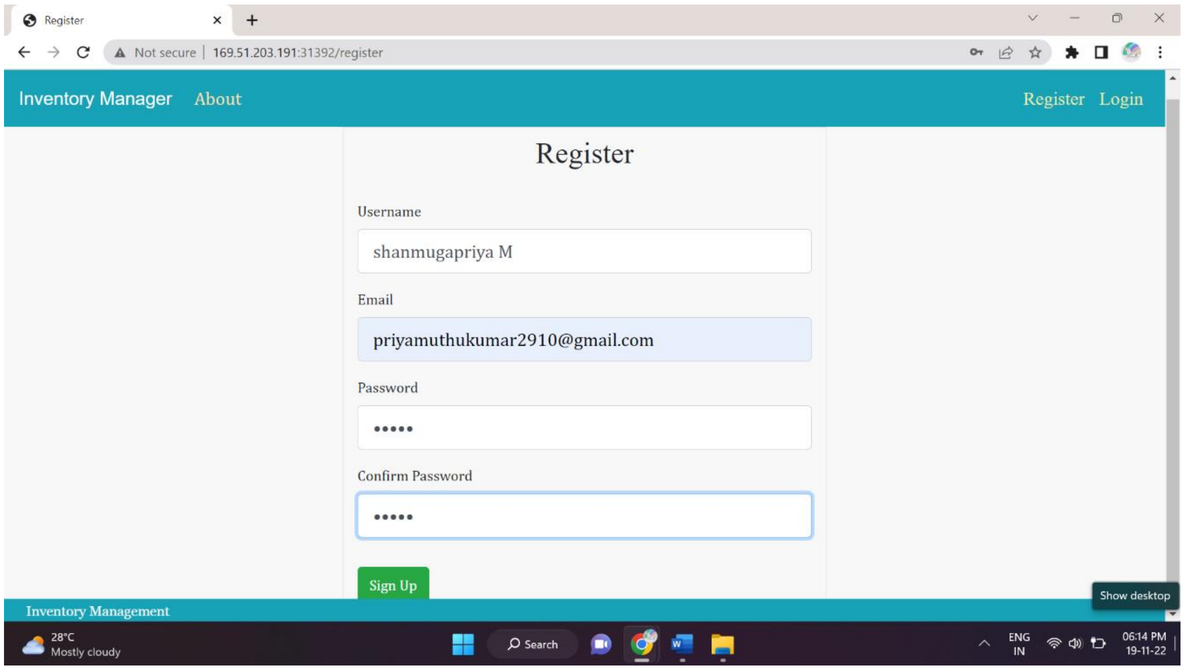
Password

Remember Me

Login

Don't Have an Account ? Register Now

Inventory Management



Register

Not secure | 169.51.203.191:31392/register

Inventory Manager About Register Login

Register

Username: shanmugapriya M

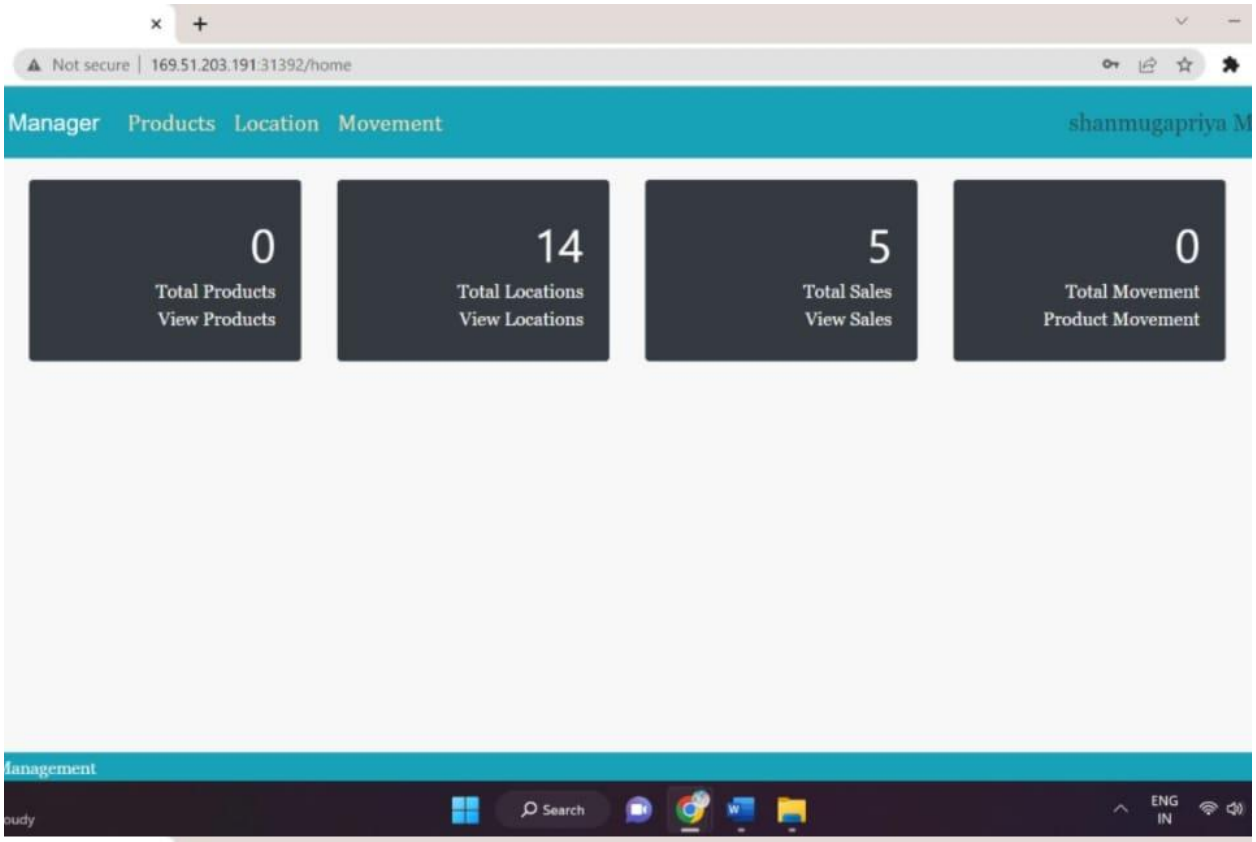
Email: priyamuthukumar2910@gmail.com

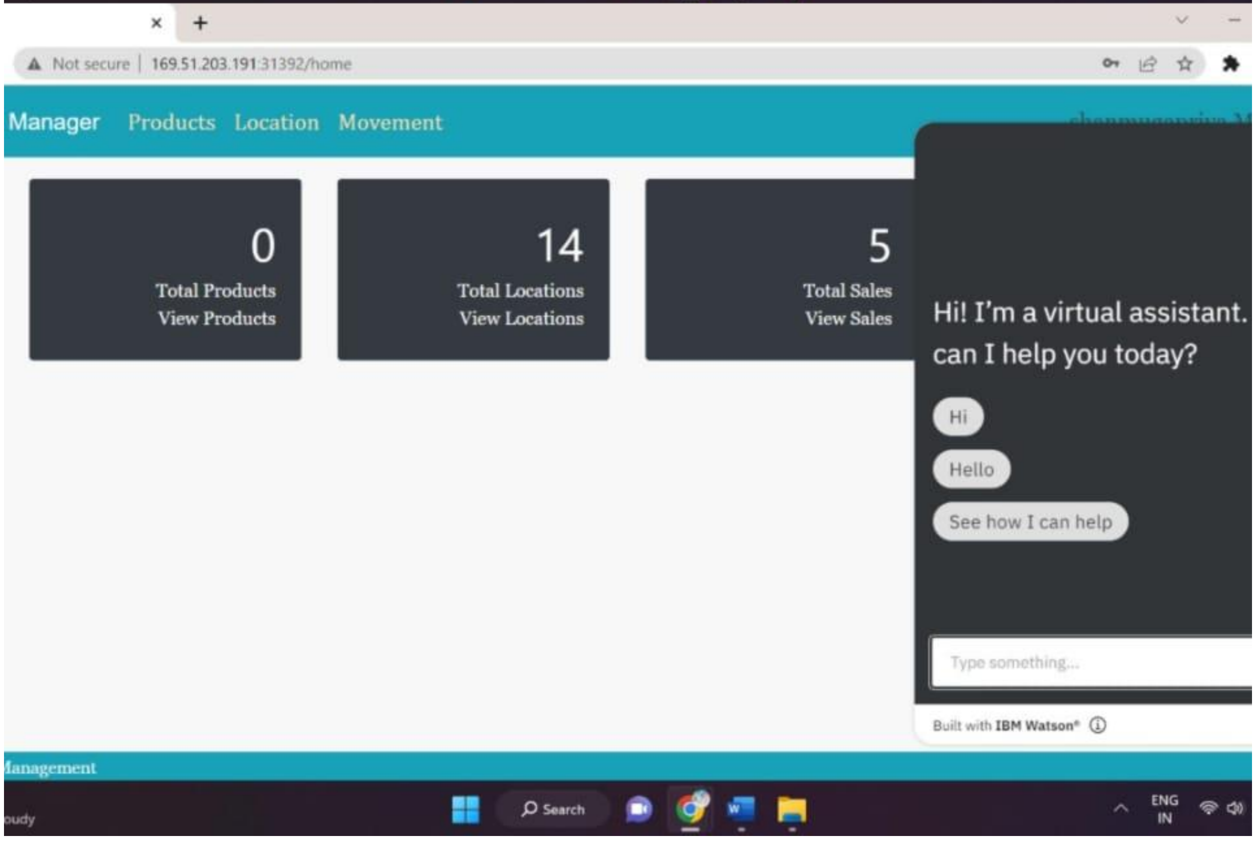
Password: •••••

Confirm Password: •••••

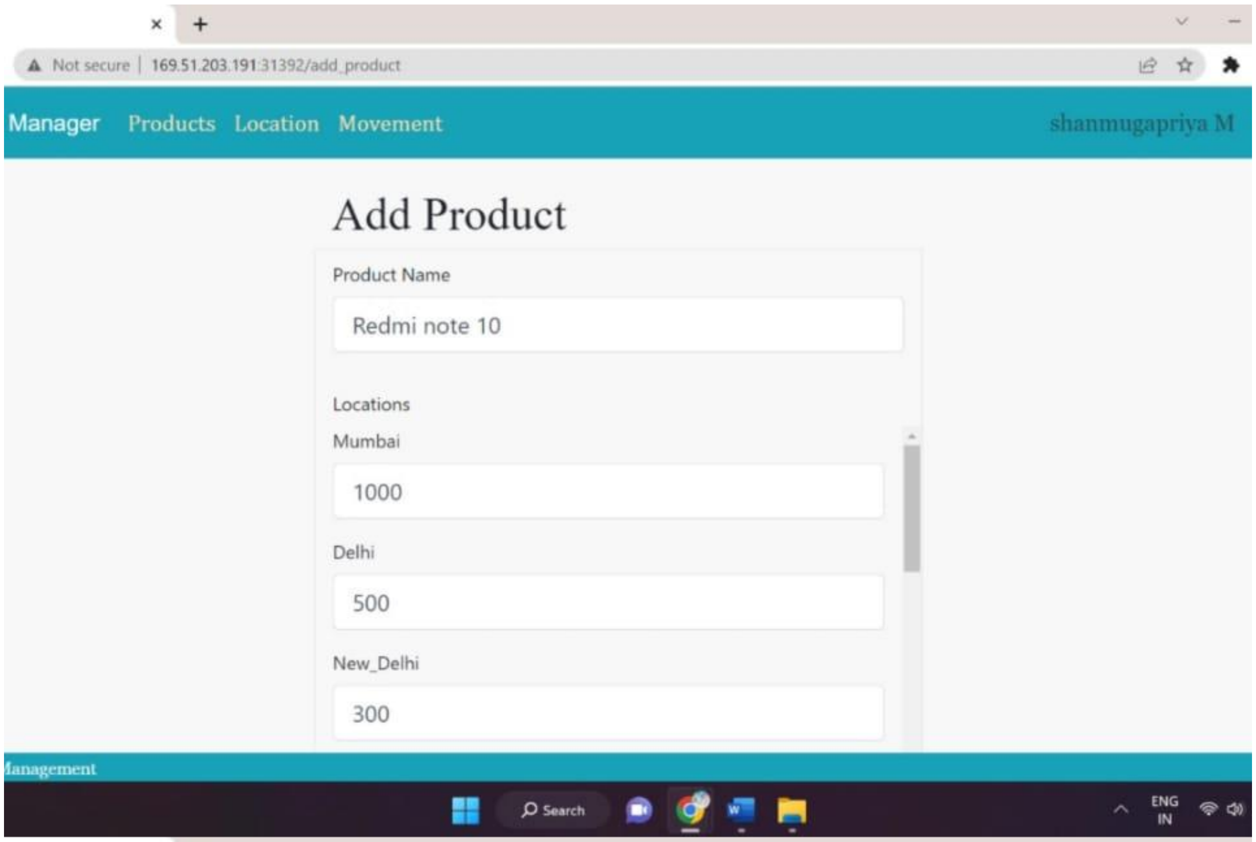
Sign Up

Inventory Management





Hi! I'm a virtual assistant. can I help you today?


Not secure | 169.51.203.191:31392/add_product

Manager Products Location Movement shanmugapriya M

Add Product

Product Name: Redmi note 10

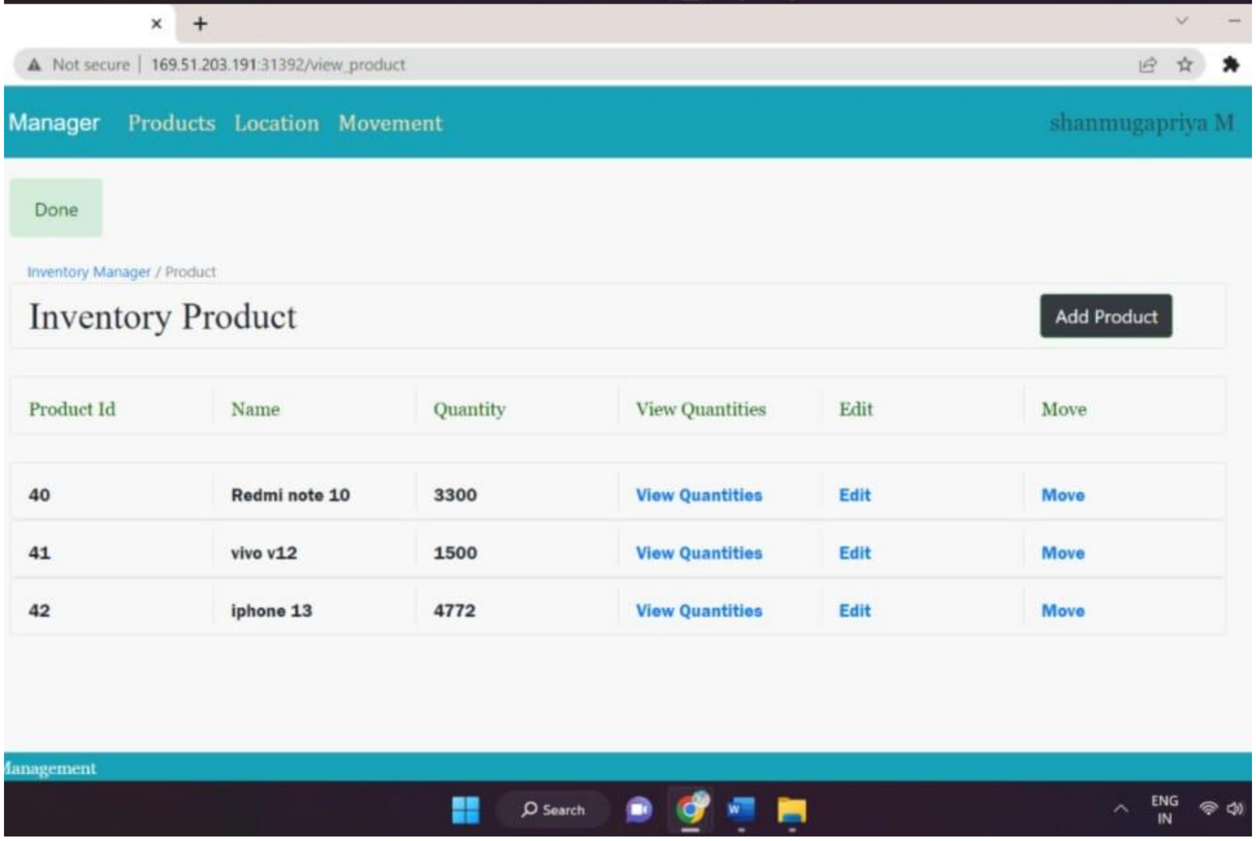
Locations

Mumbai: 1000

Delhi: 500

New_Delhi: 300

lanagement


Not secure | 169.51.203.191:31392/view_product

Manager Products Location Movement shanmugapriya M

Done

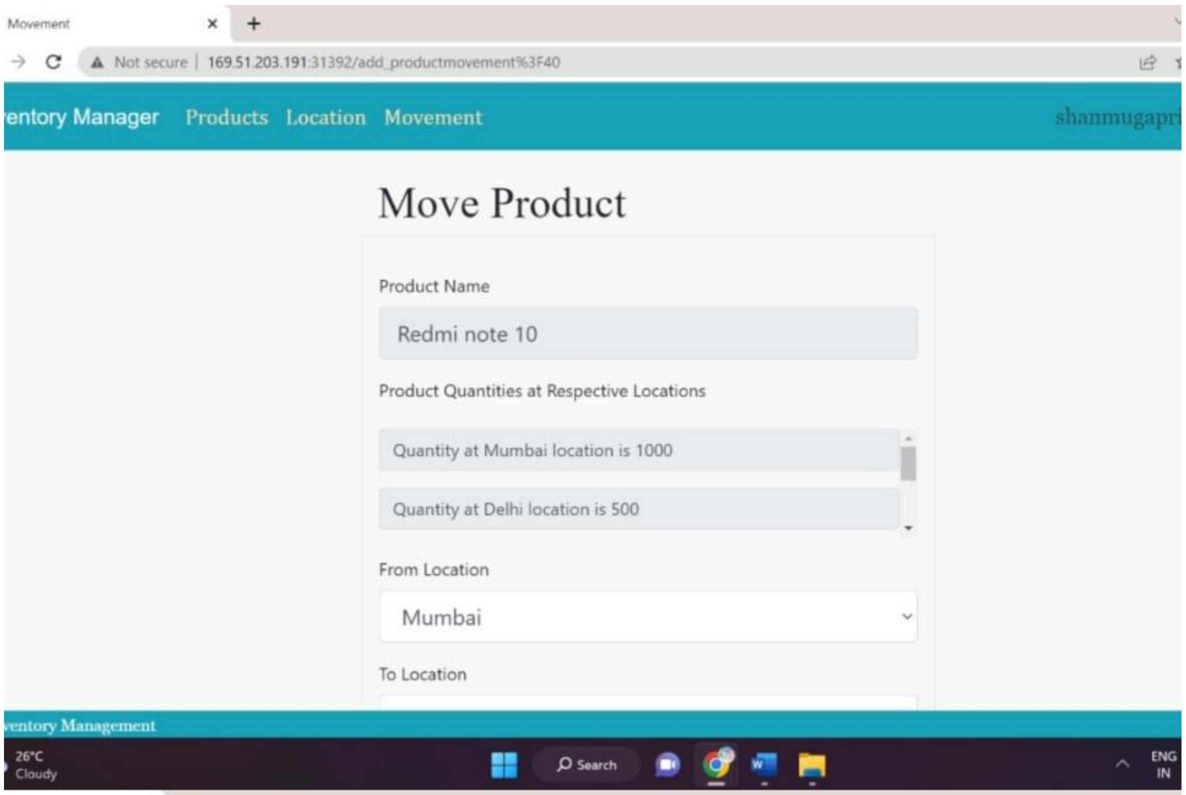
Inventory Manager / Product

Inventory Product

Add Product

Product Id	Name	Quantity	View Quantities	Edit	Move
40	Redmi note 10	3300	View Quantities	Edit	Move
41	vivo v12	1500	View Quantities	Edit	Move
42	iphone 13	4772	View Quantities	Edit	Move

lanagement


Movement

Not secure | 169.51.203.191:31392/add_productmovement%3F40

entory Manager Products Location Movement shanmugapri

Move Product

Product Name

Redmi note 10

Product Quantities at Respective Locations

Quantity at Mumbai location is 1000

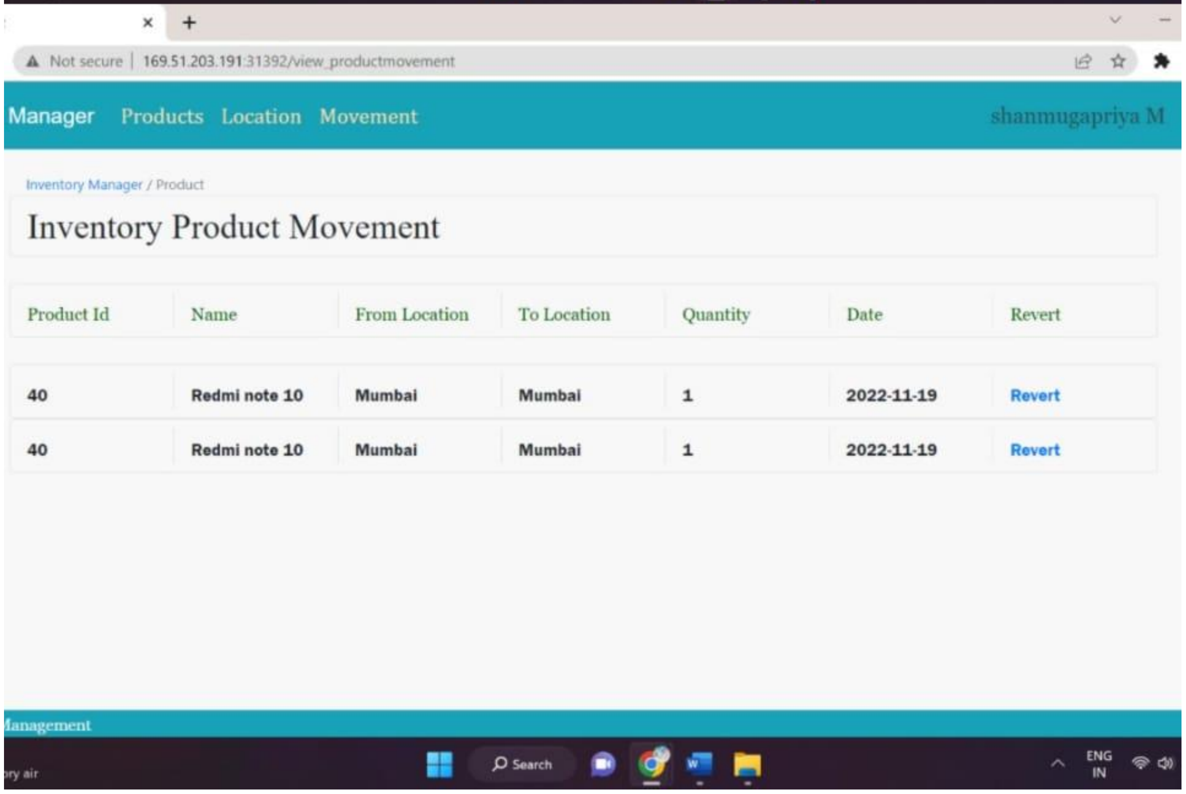
Quantity at Delhi location is 500

From Location

Mumbai

To Location

entory Management


Not secure | 169.51.203.191:31392/view_productmovement

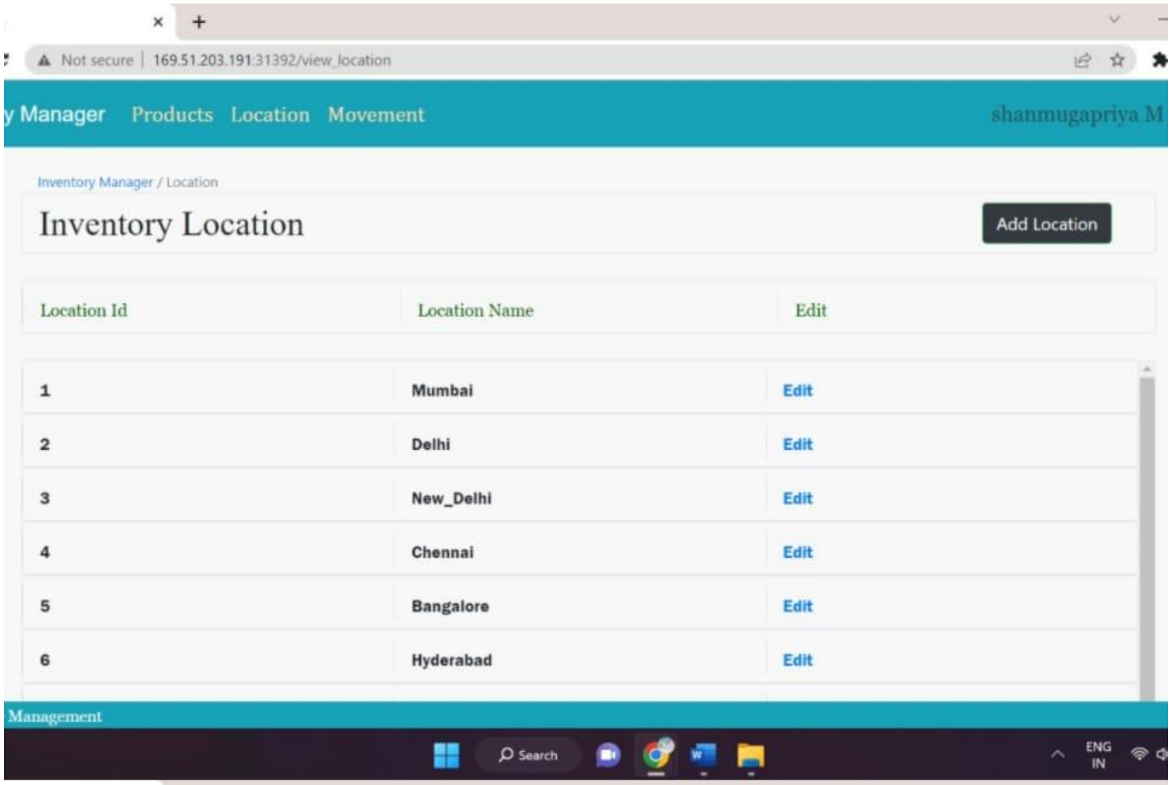
Manager Products Location Movement shanmugapriya M

Inventory Manager / Product

Inventory Product Movement

Product Id	Name	From Location	To Location	Quantity	Date	Revert
40	Redmi note 10	Mumbai	Mumbai	1	2022-11-19	Revert
40	Redmi note 10	Mumbai	Mumbai	1	2022-11-19	Revert

lanagement



Not secure | 169.51.203.191:31392/view_location

y Manager Products Location Movement shanmugapriya M

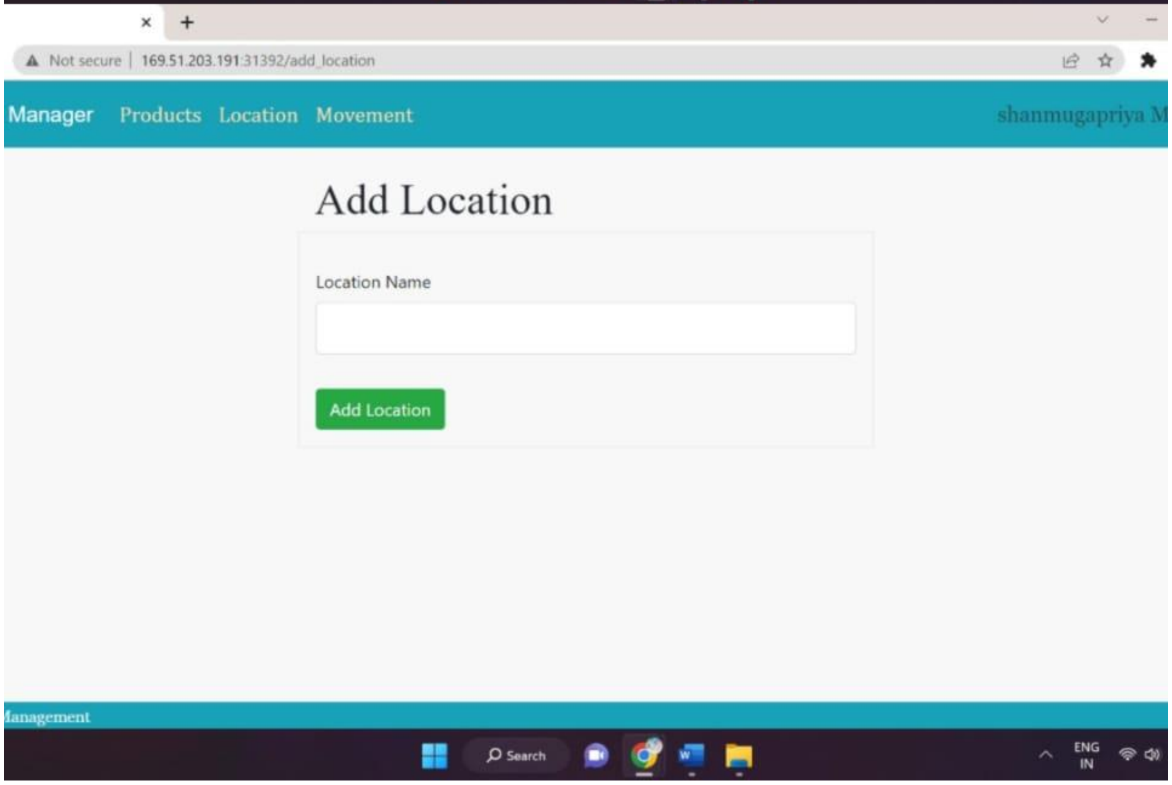
Inventory Manager / Location

Inventory Location

Add Location

Location Id	Location Name	Edit
1	Mumbai	Edit
2	Delhi	Edit
3	New_Delhi	Edit
4	Chennai	Edit
5	Bangalore	Edit
6	Hyderabad	Edit

Management



Not secure | 169.51.203.191:31392/add_location

Manager Products Location Movement shanmugapriya M

Add Location

Location Name

Add Location

lanagement

GitHub Link:

https://github.com/IBM-EPBL/IBM-Project-47081-

1660796442

Project Demo Link:

https://drive.google.com/file/d/1TEvlQxAvfyZYiGdR3Kbj
9i9hnEZZCOTr/view?usp=drivesdk