IBM

Inventory Management System for Retailers

PRATHYUSHA ENGINEERING COLLEGE

Report

TEAM MEMBERS

DEEPAN.E

AKILA.P

KEERTHANA.S

SWEATHA.V

INDEX

1. INTRODUCTION

PROJECT OVERVIEW:

A difficult issue in supply chain management is inventory management. The company's issue is that they have no system in place to monitor inventory data. The store finds it challenging to keep track of the inventory information. The main challenge for every inventory stock management is keeping track of how much stock is bought and how much stock is used. In this case, a tool or system to help with inventory management would be useful. The term "inventory management" describes the control of the quantity, quality, location, and transportation of a wide range of goods used in manufacture by a range of commercial enterprises or in sales by a range of retailers.

PURPOSE:

Typically, inventory management systems are constrained and restricted to a predetermined range of items and are unable to be altered and expanded in response to the needs of the client. The Inventory Management System puts a lot of effort into making it flexible and simple for the end user to use, with ongoing customer support to change the use. Inventory Management System concentrates on making it easier by adding details of different other entities that are a part of the company, in contrast to other software that offers comparable features.

2. LITERATURE SURVEY

EXISTING PROBLEM:

1. Design of a Computerized Inventory Management System for Supermarkets

Aim:

The purpose of this article is to develop a computerised inventory management system that will help managers make decisions about how much inventory to order, when to order more inventory, and how to keep track of transactions.

Scope:

This work focuses on stock control, management, and tends to fix business abnormalities. It examines the capacity to view current ones as well as the opening of new stocks and stock updates.

Implementation:

New stock, stock orders, stock updates, product or item searches to determine availability, and stock reports are all handled electronically by this system.

- To ensure security, a login page is made.
- If it is successful, the splash screen will appear, followed by an automatic display of the main menu form.
- The user can select from the New Stock, Update Stock, Search, and View options on the onscreen menu.
- Stock information will be recorded in the user form.
- Which are later fetched from the database, displayed, and kept there.

2. The inventory management system for automobile spare parts in a central warehouse

Aim:

In order to manage the inventory of car parts in a central warehouse, this research seeks to design an expanded fuzzy neural network (EFNN) based decision support system.

Scope:

To achieve greater accuracy than using an artificial neural network. This project incorporates domain experts' expertise into enhanced fuzzy neural networks (EFNN), which create connection weights based on the fuzzy analytic hierarchy process (AHP) method without laboriously and slowly rotating them.

Implementation:

Three parts make up the proposed system.

The fuzzy AHP's hierarchical structure development

The subject matter specialists are questioned regarding the parts, demand, timing, sales, and other relevant aspects that affect the supply of spare parts.

Weights determination: Based on the suggested framework, a new questionnaire is created. Comparing pairs of items from each level to each element in the level above using questionnaire surveys. There is a 7-point scale in use.

Making decisions using EFNN.

To improve accuracy, the EFNN, a five-layered hybrid neural network with the ability to self-organize its activation function, is used.

3. Design of smart inventory management system for construction sector based on IoT and cloud computing.

Aim:

A novel approach to create a model and show how this can help construction sector in managing inventory of essential form work shuttering products.

Scope:

This research reveals that there could be an opportunity to approach barcodebased designs by amalgamating such with Cloud Computing, Arduino-based wireless station nodes, IoT and a secure form channel to access data through a dedicated web portal.

Implementation:

The proposed model is a novel Aluminium Shuttering Inventory Management System (ASIMS) consist of barcodes, Arduino-based IoT devices, wireless sensor networks and Cloud Computing to track Aluminium formwork shuttering components under actual field conditions.

- Upon receipt of Aluminium formwork shuttering components from vendor at site, a Goods Receipt Note (GRN) entry is passed in the system.
- Physical verification of the received items and GRN process have needed to be completed.
- The barcode labels for the items are generated and printed. The printed barcode labels are then affixed on the formwork shuttering components.
- Using our proposed application, the component is labelled and then mapped with corresponding geolocational coordinates to enable tracking.
- Aluminium formwork shuttering components are often cut and resized according to localized requirements. During such process of resizing, the created new items have to be checked, verified physically and logged using our proposed software. Again, new barcode labels have been generated for the new components derived from the parent item.

4. Design of smart inventory management system for construction sector based on IoT and cloud computing.

Aim:

A novel approach to Design of smart inventory management system for construction sector based on IoT and cloud computing.

Implementation:

Inventory management and working capital management are routine activities for a specific company and firms. Inventory is one of the most significant components of current assets and current assets, in general, are part of working capital, so it is crucial to understand the amount locked up in inventory and to manage the inventory in the best possible way. In this section of the research, we analyse and investigate the inventory management practises used by Steel Authority of India Limited.

A very important factor in the short-term liquidity position and a major factor in long-term profitability is the stock of raw materials, work in progress, finished goods, etc. In this research project, data for the previous five years is gathered from the company's annual reports, and various ratios are then applied to the data in order to quantify inventory efficiency. In addition, we used a variety of statistical tools to examine the behaviour of selected ratios.

5. Relationship Between Inventory Management and Profitability: An Empirical Analysis of Indian Cement Companies.

Aim:

An analysis on relationship Between Inventory Management and Profitability: An Empirical Analysis of Indian Cement Companies.

Implementation:

The goal of this essay is to investigate the connection between a firm's profitability and its inventory conversion period. For a sample of five top Indian cement companies over a ten-year period, from 2001–2010, the relationship between inventory management and profitability is examined. Gross operating profit is employed as the dependent variable as a measure of profitability. The impact of the inventory conversion period on gross operating profit is examined using regression analysis in this study, which also takes into account the firm's size, current ratio, and financial debt ratio as control variables. The findings show a significant negative linear association between profitability and the inventory conversion period.

The outcomes of this study corroborate earlier conclusions. According to the results, the inventory conversion period and firm profitability are inversely related; that is, as ICP days rise, a company's profitability declines, and vice versa. It was discovered that the ratio of financial debt and corporate profitability, as determined by GOP, are negatively correlated. Inferred from this was the notion that profitability rises when the ratio of financial debt falls. According to this study's findings, there was a positive association between firm size and GOP, indicating that profitability rises as business size increases. GOP support had a negative correlation to current ratio.

REFERENCES:

1. Abisoye, O. A., Boboye, F., & Abisoye, B. O. (2013). Design of a computerized inventory management system for supermarkets.
2. Li, S. G., & Kuo, X. (2008). The inventory management system for automobile spare parts in a central warehouse. Expert Systems with Applications, 34(2), 1144-1153.
3. Saleem, A. (2020). Automated inventory management systems and its impact on supply chain risk management in manufacturing firms of Pakistan. Int J Supply Chain Manag, 9, 220-231.
4. Bose, R., Mondal, H., Sarkar, I., & Roy, S. (2022). Design of smart inventory management system for construction sector based on IoT and cloud computing. e-Prime-Advances in Electrical Engineering, Electronics and Energy, 2, 100051.
5. Kasim, H., Zubieru, M., & Antwi, S. K. (2015). An assessment of the inventory management practices of small and medium enterprises (SMEs) in the Northern Region of Ghana. *European Journal of Business and Management*, *7*(20), 28-40.

6. Panigrahi, C. M. A. (2013). Relationship between inventory management and profitability: An empirical analysis of Indian cement companies. *Asia Pacific Journal of Marketing & Management Review, 2*(7).

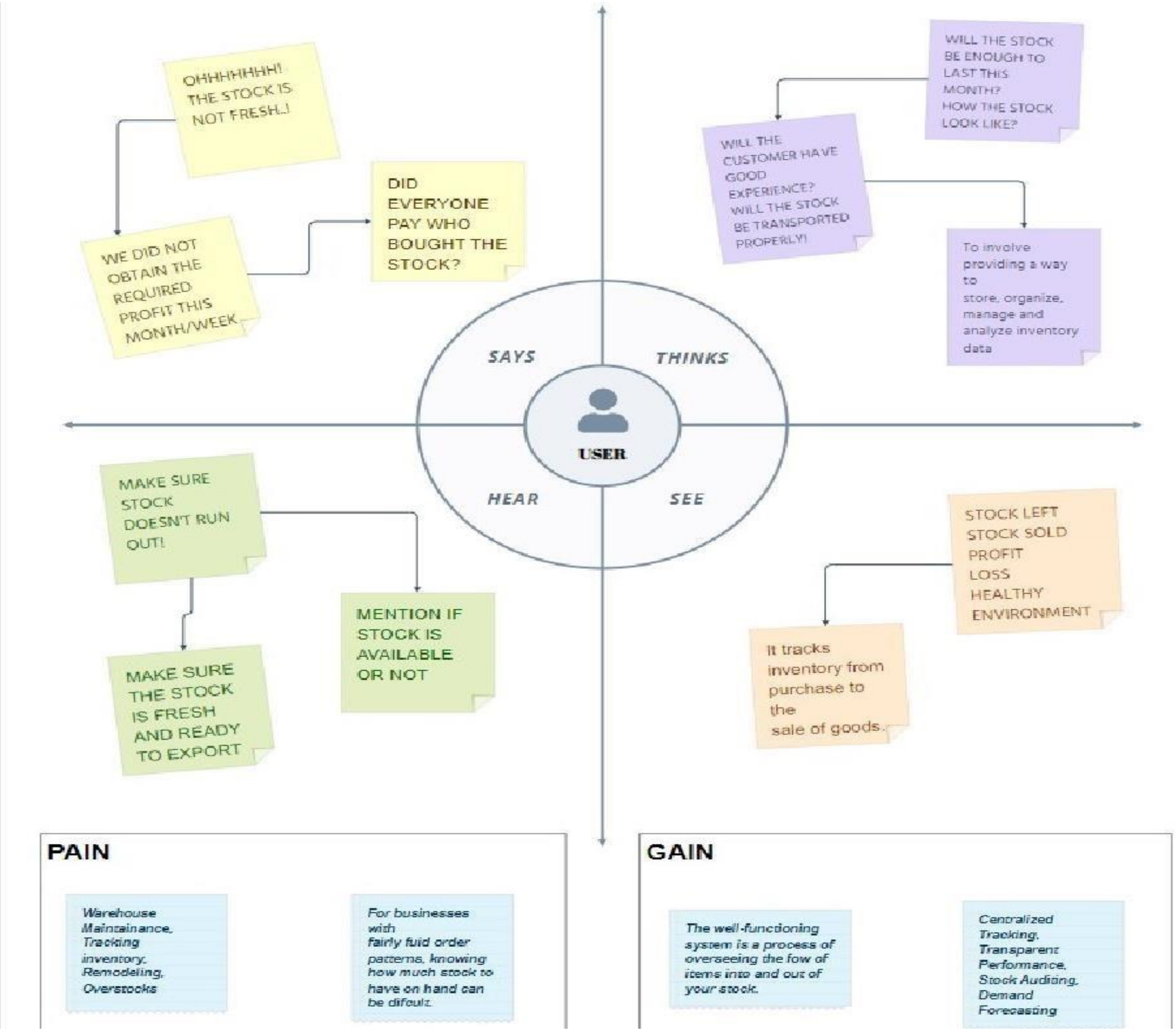
PROBLEM STATEMENT DEFINITION:

Inventory management is a challenging aspect of supply chain management. The problem for the company is that they don't have a system in place to track inventory data. It is difficult for the store to maintain track of the inventory data. Keeping track of how much stock is purchased and used is the key problem for inventory stock management. A system or tool to assist with inventory management would be helpful in this situation. The control of the quantity, quality, placement, and transportation of a large variety of items utilised in production by a variety of commercial enterprises or in sales by a variety of retailers is referred to as inventory management.

Most inventory management systems are limited to a fixed set of items and are not able to be changed or expanded in response to the client's needs. The Inventory Management System makes a lot of effort to be adaptable and userfriendly, and it offers ongoing customer assistance to adjust the use. Unlike other software that provides equivalent functions, Inventory Management System focuses on simplifying it by adding details of many other entities that are a part of the firm.

3. IDEATION AND PROPOSED SOLUTION EMPATHY

MAP CANVAS:



SAYS

OHHHHHHH! THE STOCK IS NOT FRESH..!

WE DID NOT OBTAIN THE REQUIRED PROFIT THIS MONTH/WEEK

DID EVERYONE PAY WHO BOUGHT THE STOCK?

THINKS

WILL THE STOCK BE ENOUGH TO LAST THIS MONTH? HOW THE STOCK LOOK LIKE?

WILL THE CUSTOMER HAVE GOOD EXPERIENCE? WILL THE STOCK BE TRANSPORTED PROPERLY!

To involve providing a way to store, organize, manage and analyze inventory data

USER

HEAR

MAKE SURE STOCK DOESN'T RUN OUT!

MENTION IF STOCK IS AVAILABLE OR NOT

MAKE SURE THE STOCK IS FRESH AND READY TO EXPORT

SEE

STOCK LEFT
STOCK SOLD
PROFIT
LOSS
HEALTHY ENVIRONMENT

It tracks inventory from purchase to the sale of goods.

PAIN

Warehouse Maintainance, Tracking inventory, Remodeling, Overstocks

For businesses with fairly fuid order patterns, knowing how much stock to have on hand can be difcult.

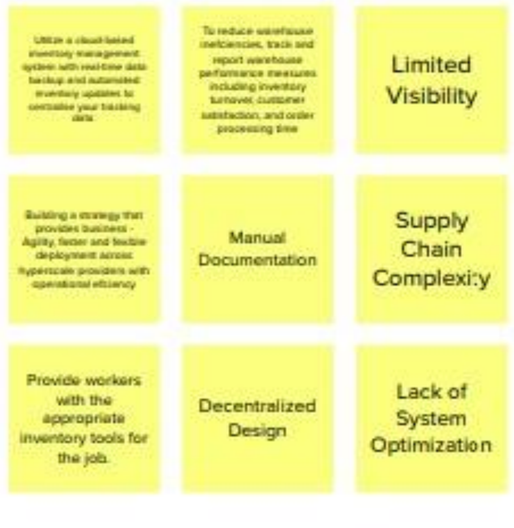
GAIN

The well-functioning system is a process of overseeing the fow of items into and out of your stock.

Centralized Tracking, Transparent Performance, Stock Auditing, Demand Forecasting

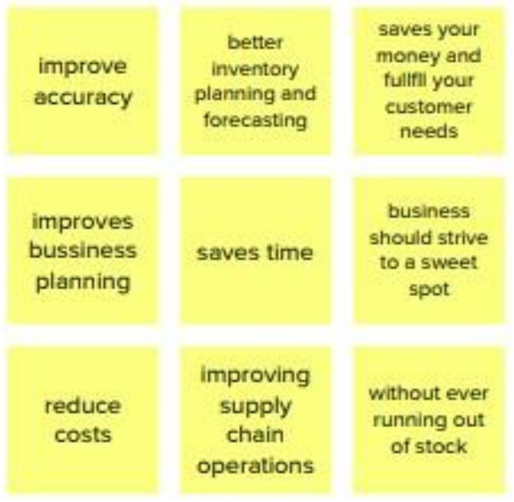
IDEATION AND BRAINSTORMING:

DEEPAN E



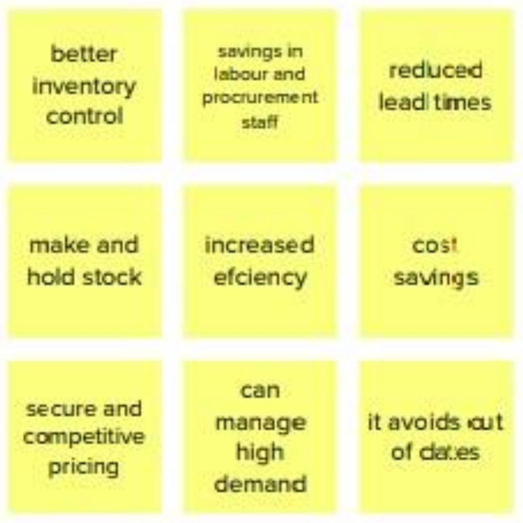
Utilize a cloud-based inventory management system with real-time data backup and automated inventory updates to centralize your tracking data	To reduce warehouse inefficiencies, track and report warehouse performance measures including inventory turnover, customer satisfaction, and order processing time	Limited Visibility
Building a strategy that provides business - Agility, faster and flexible deployment across hyperscale providers with operational efficiency	Manual Documentation	Supply Chain Complexity
Provide workers with the appropriate inventory tools for the job.	Decentralized Design	Lack of System Optimization

AKILA P



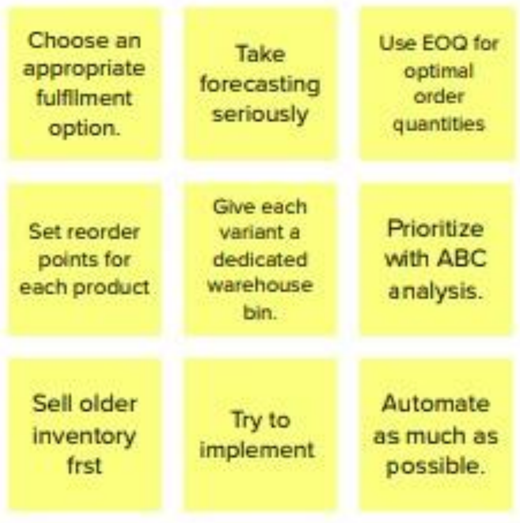
improve accuracy	better inventory planning and forecasting	saves your money and fullfil your customer needs
improves bussiness planning	saves time	business should strive to a sweet spot
reduce costs	improving supply chain operations	without ever running out of stock

KEERTHANA S



better inventory control	savings in labour and procrurement staff	reduced lead times
make and hold stock	increased efficiency	cost savings
secure and competitive pricing	can manage high demand	it avoids out of dates

SWEATHA V



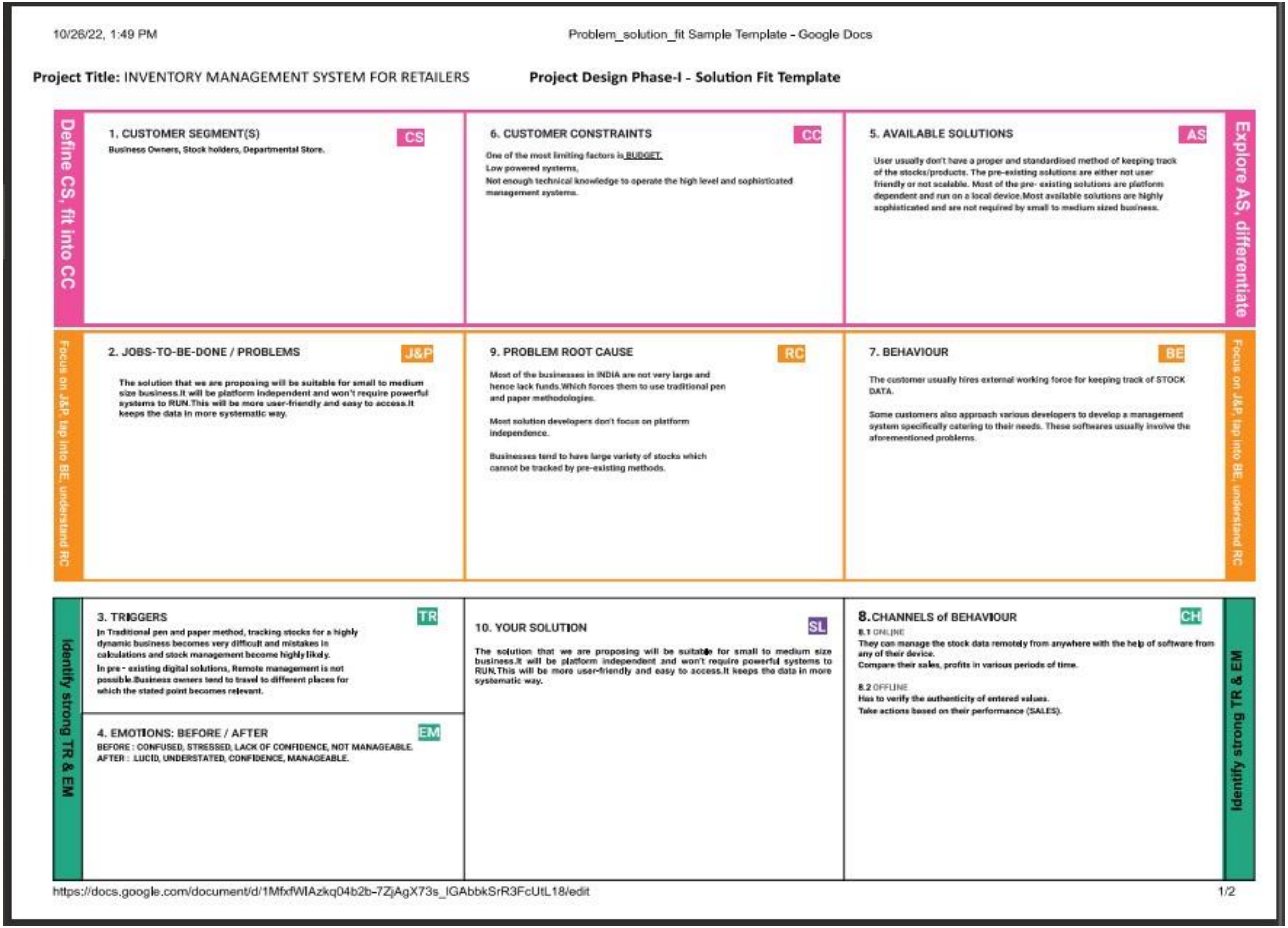
Choose an appropriate fulfilment option.	Take forecasting seriously	Use EOQ for optimal order quantities
Set reorder points for each product	Give each variant a dedicated warehouse bin.	Prioritize with ABC analysis.
Sell older inventory first	Try to implement	Automate as much as possible.

PROPOSED SOLUTION:

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	• Customers are dissatisfied with the retailer's store because it lacks sufficient supplements and deliveries were not produced on time. • The retailers typically experience problems recording the stocks and its threshold limit available.
2.	Idea / Solution description	• The daily update system in this proposed system will be activated once a product is sold or renewed more. • The availability of the products is monitored daily, and an alarm system is kept active to warn of any products that fall below the predetermined threshold. • All consumers can open an account by registering, and they will then receive login information they can use whenever they want to buy stocks. • The application gives clients access to information about all of the current stock options as well as when new stock will be put on sale in the store.
3.	Novelty / Uniqueness	• Specific machine learning techniques are utilised to forecast the high-demand seasonal products that can be made available at that time. • Based on their acceptance, cost, and levels of consumer pleasure and trust, specific product predictions of the bestselling brand will be put into practise. • If a product that customers have been searching for is not available, notifications will be issued to the shops so that the product can be stocked up quickly.

4.	Social Impact / Customer Satisfaction	• The clients will be extremely satisfied because less time will be wasted looking for a product that isn't available. • Why If the system is automated every day and at every purchase, the workload of the retailers will be kept to a minimum. • As a result of receiving prompt and suitable responses from shops, customer happiness will increase.
5.	Business Model (Revenue Model)	Hereby we can provide a robust and most reliable inventory management system by using: ML algorithms for all prediction needs employing all historical data since datasets are unquestionably abundant. Has the best business advertising models available. To develop a plan for preventing losses. To guarantee the system of perpetual, global product availability.
6.	Scalability of the Solution	• The use of a system that everyone and anywhere may use can enable even the average person purchase the products. • The daily and regular updating of stock purchases to stop inventory shrinkage.

PROBLEM SOLUTION FIT:



Project Title: INVENTORY MANAGEMENT SYSTEM FOR RETAILERS

Project Design Phase-I - Solution Fit Template

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

Business Owners, Stock holders, Departmental Store.

6. CUSTOMER CONSTRAINTS CC

One of the most limiting factors is BUDGET.
Low powered systems,
Not enough technical knowledge to operate the high level and sophisticated management systems.

5. AVAILABLE SOLUTIONS AS

User usually don't have a proper and standardised method of keeping track of the stocks/products. The pre-existing solutions are either not user friendly or not scalable. Most of the pre- existing solutions are platform dependent and run on a local device.Most available solutions are highly sophisticated and are not required by small to medium sized business.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

The solution that we are proposing will be suitable for small to medium size business.It will be platform independent and won't require powerful systems to RUN.This will be more user-friendly and easy to access.It keeps the data in more systematic way.

9. PROBLEM ROOT CAUSE RC

Most of the businesses in INDIA are not very large and hence lack funds.Which forces them to use traditional pen and paper methodologies.

Most solution developers don't focus on platform independence.

Businesses tend to have large variety of stocks which cannot be tracked by pre-existing methods.

7. BEHAVIOUR BE

The customer usually hires external working force for keeping track of STOCK DATA.

Some customers also approach various developers to develop a management system specifically catering to their needs. These softwares usually involve the aforementioned problems.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

In Traditional pen and paper method, tracking stocks for a highly dynamic business becomes very difficult and mistakes in calculations and stock management become highly likely.
In pre - existing digital solutions, Remote management is not possible.Business owners tend to travel to different places for which the stated point becomes relevant.

4. EMOTIONS: BEFORE / AFTER EM

BEFORE : CONFUSED, STRESSED, LACK OF CONFIDENCE, NOT MANAGEABLE.
AFTER : LUCID, UNDERSTATED, CONFIDENCE, MANAGEABLE.

10. YOUR SOLUTION SL

The solution that we are proposing will be suitable for small to medium size business.It will be platform independent and won't require powerful systems to RUN.This will be more user-friendly and easy to access.It keeps the data in more systematic way.

8.CHANNELS of BEHAVIOUR CH

8.1 ONLINE
They can manage the stock data remotely from anywhere with the help of software from any of their device.
Compare their sales, profits in various periods of time.

8.2 OFFLINE
Has to verify the authenticity of entered values.
Take actions based on their performance (SALES).

Identify strong TR & EM

4. REQUIREMENT ANALYSIS PHASE FUNCTIONAL REQUIREMENTS:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registering through a form Registering through mail
FR-2	User Confirmation	Email confirmation OTP confirmation
FR-3	Login	Log in to the application by entering required credentials (email ID and password)
FR-4	Dashboard	View the products details (Name, quantity)
FR-5	Add items to the Inventory list	Users can add items that they wish to buy to the inventory
FR-6	Stock Updation	Increasing the availability of a particular product

NON-FUNCTIONAL REQUIREMENTS:

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	If the system has a steep learning curve, then it would mostly not be purchased by the company needing an inventory management system. • The UI is simple and easy to navigate □ Consistent design and colours are used. • The webpages are responsive • Email delivery is to be fast
NFR-2	**Security**	Security refers to the safety and management of the inventory of a company such that only authorised personnel are allowed to access them. • Login system is used to provide authentication. • Users need to create account and verify it with their email OTP. • Cookie based security is user for authentication on client side.

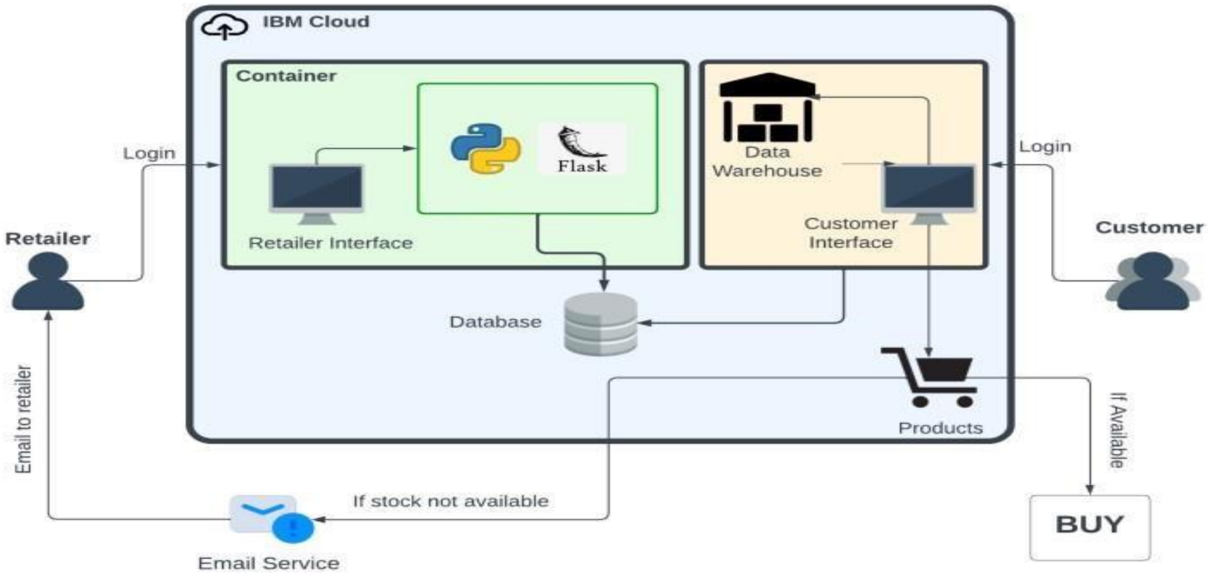
NFR-3	**Reliability**	• Exception handling will be done at the code level to ensure that the app performs well even when errors happen in the runtime • Multiple instances of the App would be onlin ensure continued operation

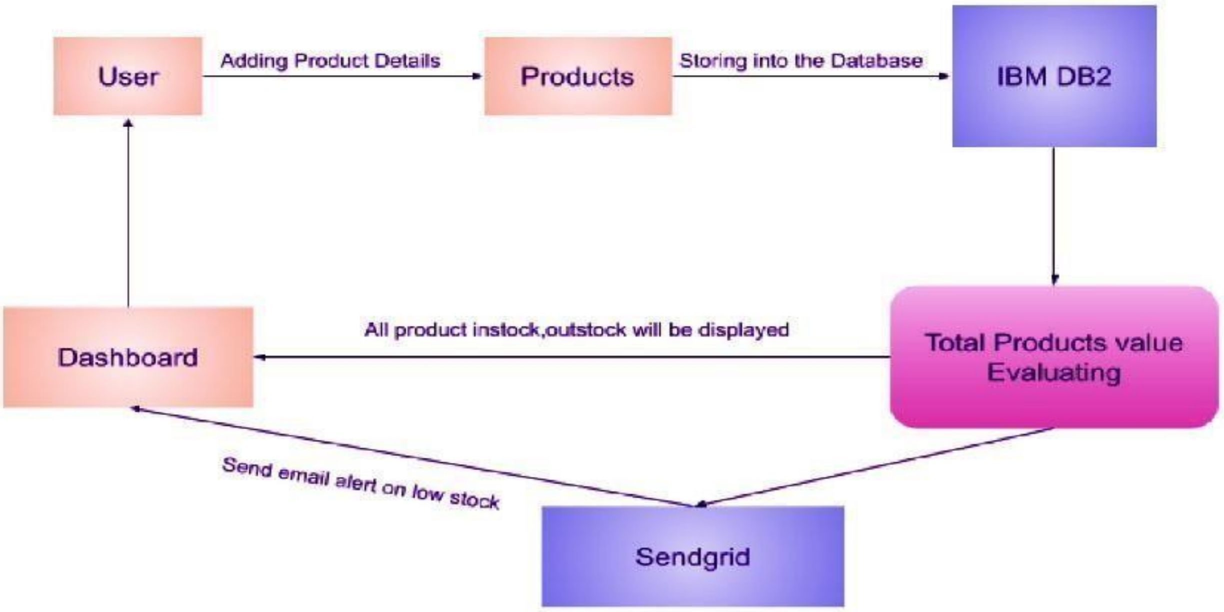
NFR-4	**Performance**	Performance of an inventory management system depends on the efficiency with which various tasks in it can be executed. • Reduces manpower, cost and saves time. Emails will be sent automatically when stocks are not available. • Makes the business process more efficient. • Improves organizations performance. • It will be performed fast and secure even at the lower bandwidth
NFR-5	**Availability**	The use of IBM DB2 ensures high availability
NFR-6	**Scalability**	The scalability of an inventory management system refers to the extensibility of its operations. • DB2 is highly Scalable • The code is developed efficiently to easily add new features without many changes by reusing the code. • Docker in IBM Container registry is used which is highly scalable

5. PROJECT DESIGN

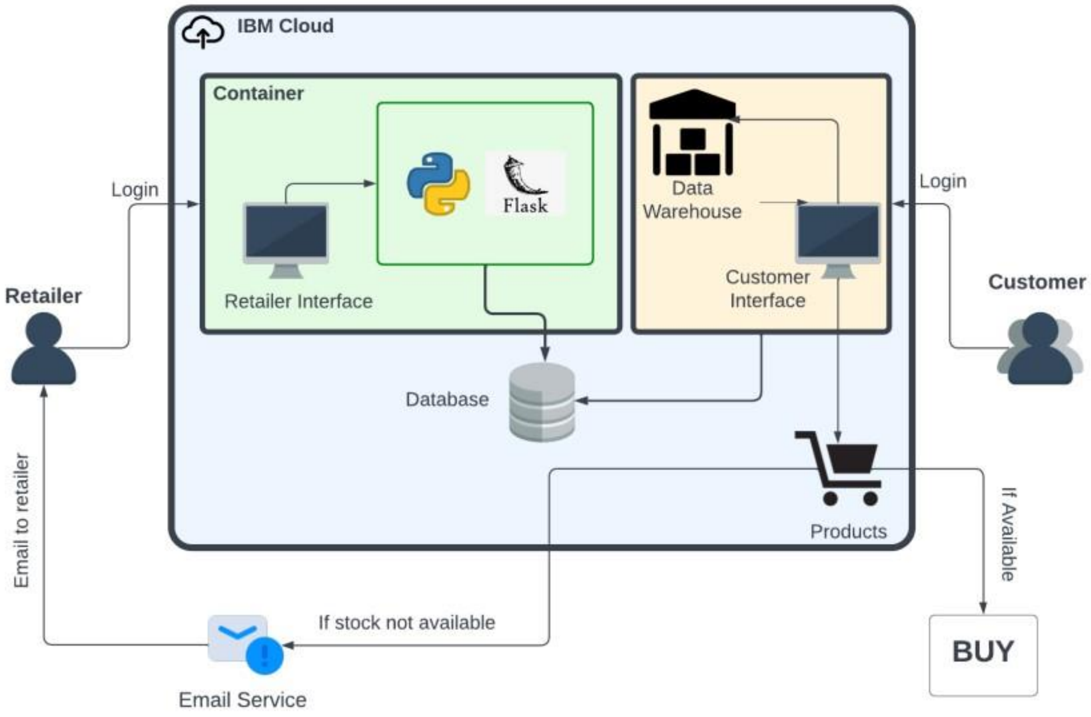
DATA FLOW DIAGRAMS:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.




User
Adding Product Details
Products
Storing into the Database
IBM DB2
Total Products value Evaluating
All product instock,outstock will be displayed
Dashboard
Send email alert on low stock
Sendgrid

SOLUTION ARCHITECTURE:



TECHNOLOGY STACK:

Table-1: Components & Technologies:

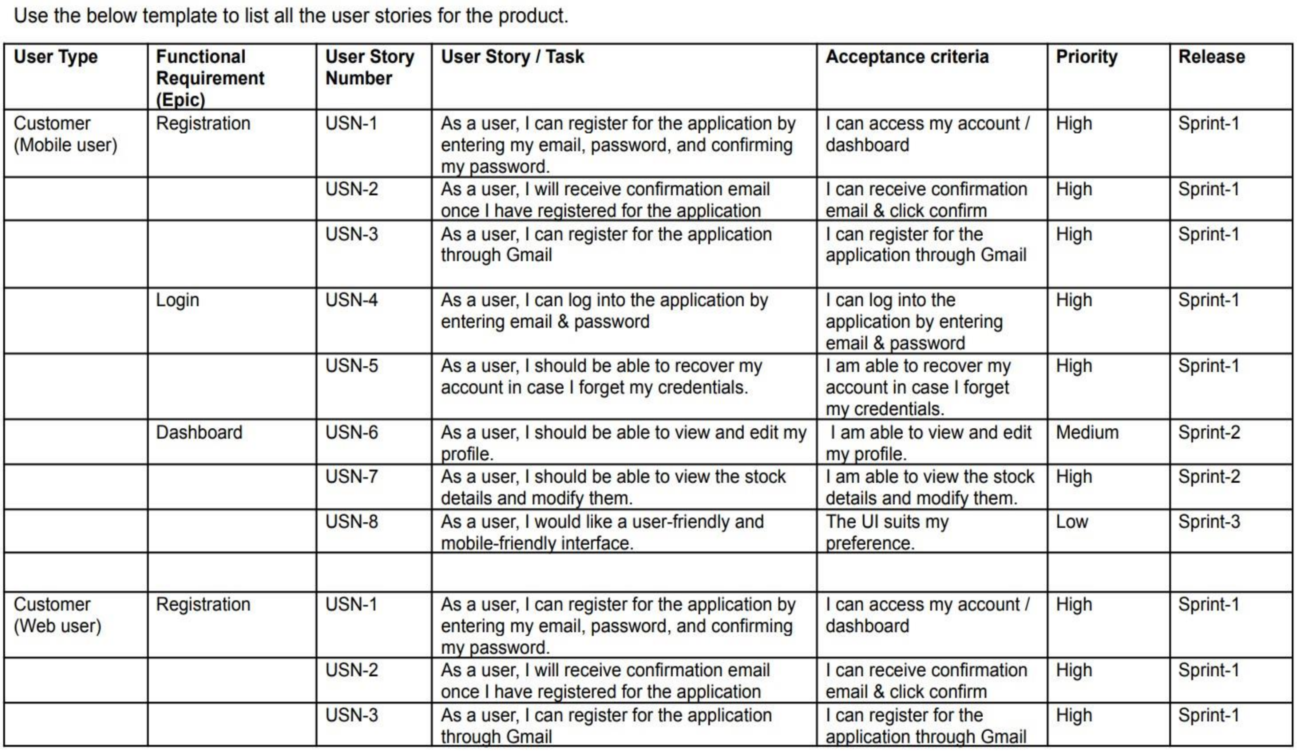
S.No	Component	Description	Technology
1.	User Interface	How user interacts with application e.g., Web UI, Mobile App, Chatbot etc.	HTML, CSS, JavaScript

2.	Application Logic-1	General purpose language	Python
3.	Application Logic-2	Containerization of platform	Docker
4.	Application Logic-3	Micro web framework	Flask
5.	Database	Data Type, Configurations etc.	MySQL
6.	Cloud Database	Database Service on Cloud	IBM DB2
7.	File Storage	File storage requirements	IBM Block Storage or Other Storage Service or Local Filesystem
8.	Infrastructure (Server / Cloud)	Application Deployment on Local System / Cloud Local Server Configuration: Cloud Server Configuration	Local, Cloud Foundry, Kubernetes, etc.

Table-2: Application Characteristics:

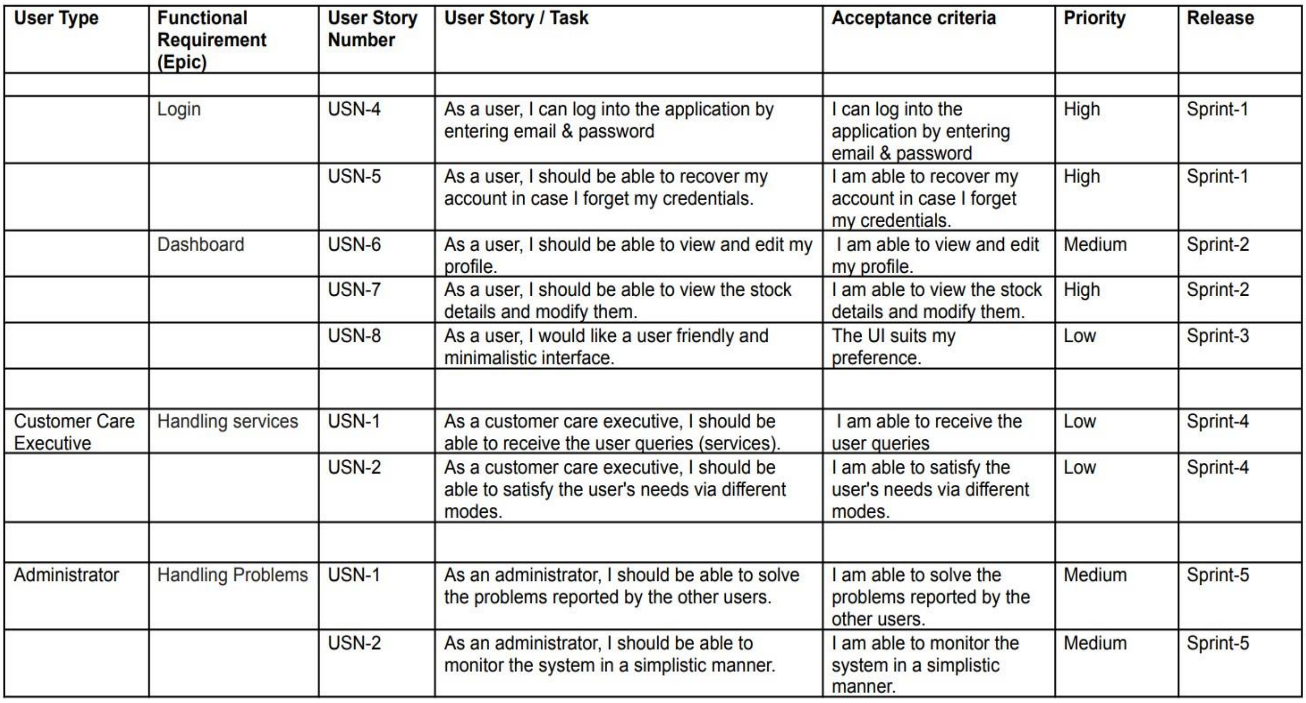
2.	Security Implementations	List all the security / access controls implemented, use of firewalls etc.	Authentication and authorization using password and usernames.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Microservices)	3-tier architecture-User, Retailer, Cloud Service

USER STORIES:



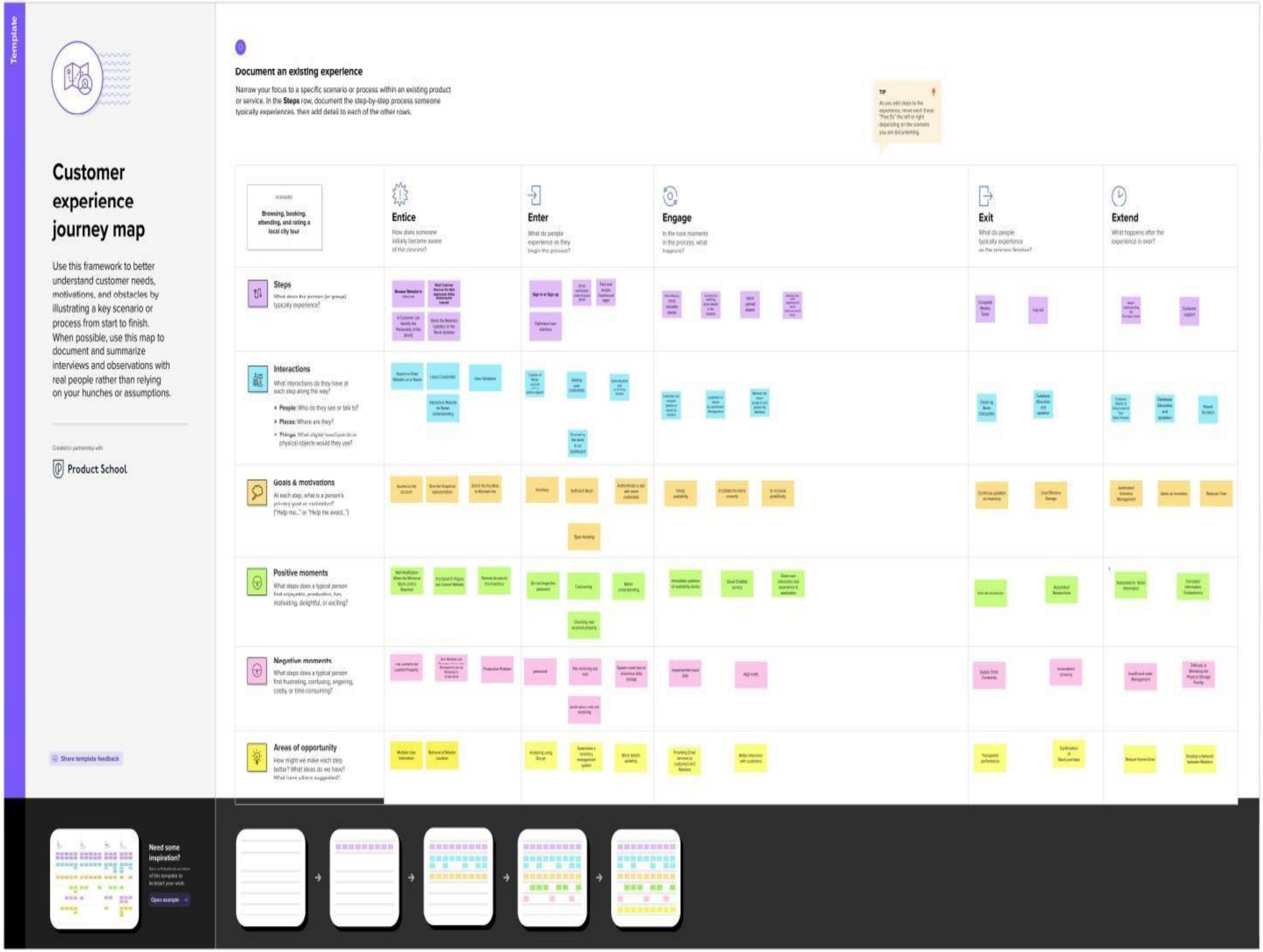
Use the below template to list all the user stories for the product.

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Gmail	I can register for the application through Gmail	High	Sprint-1
	Login	USN-4	As a user, I can log into the application by entering email & password	I can log into the application by entering email & password	High	Sprint-1
		USN-5	As a user, I should be able to recover my account in case I forget my credentials.	I am able to recover my account in case I forget my credentials.	High	Sprint-1
	Dashboard	USN-6	As a user, I should be able to view and edit my profile.	I am able to view and edit my profile.	Medium	Sprint-2
		USN-7	As a user, I should be able to view the stock details and modify them.	I am able to view the stock details and modify them.	High	Sprint-2
		USN-8	As a user, I would like a user-friendly and mobile-friendly interface.	The UI suits my preference.	Low	Sprint-3
Customer (Web user)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I can register for the application through Gmail	I can register for the application through Gmail	High	Sprint-1



User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
	Login	USN-4	As a user, I can log into the application by entering email & password	I can log into the application by entering email & password	High	Sprint-1
		USN-5	As a user, I should be able to recover my account in case I forget my credentials.	I am able to recover my account in case I forget my credentials.	High	Sprint-1
	Dashboard	USN-6	As a user, I should be able to view and edit my profile.	I am able to view and edit my profile.	Medium	Sprint-2
		USN-7	As a user, I should be able to view the stock details and modify them.	I am able to view the stock details and modify them.	High	Sprint-2
		USN-8	As a user, I would like a user friendly and minimalistic interface.	The UI suits my preference.	Low	Sprint-3
Customer Care Executive	Handling services	USN-1	As a customer care executive, I should be able to receive the user queries (services).	I am able to receive the user queries	Low	Sprint-4
		USN-2	As a customer care executive, I should be able to satisfy the user's needs via different modes.	I am able to satisfy the user's needs via different modes.	Low	Sprint-4
Administrator	Handling Problems	USN-1	As an administrator, I should be able to solve the problems reported by the other users.	I am able to solve the problems reported by the other users.	Medium	Sprint-5
		USN-2	As an administrator, I should be able to monitor the system in a simplistic manner.	I am able to monitor the system in a simplistic manner.	Medium	Sprint-5

CUSTOMER JOURNEY MAP:



Customer experience journey map

Use this framework to better understand customer needs, motivations, and obstacles by illustrating a key scenario or process from start to finish. When possible, use this map to document and summarize interviews and observations with real people rather than relying on your hunches or assumptions.

Created in partnership with

Product School

Share template feedback

Document an existing experience

Narrow your focus to a specific scenario or process within an existing product or service. In the **Steps** row, document the step-by-step process someone typically experiences, then add detail to each of the other rows.

TIP
As you add steps to the experience, move each these "Five Es" the left or right depending on the scenario you are documenting.

SCENARIO Browsing, booking, attending, and rating a local city tour	**Entice** How does someone initially become aware of this process?	**Enter** What do people experience as they begin the process?	**Engage** In the core moments in the process, what happens?	**Exit** What do people typically experience as the process finishes?	**Extend** What happens after the experience is over?
Steps What does the person (or group) typically experience?	Browse Website in internet [illegible] A Customer can identify the Personality of the Brand [illegible]	Sign in or Sign up [illegible] [illegible] Optimized user interface	[illegible] [illegible] [illegible] [illegible]	Complete Weekly Tasks Log out	[illegible] Customer support
Interactions What interactions do they have at each step along the way? - **People:** Who do they see or talk to? - **Places:** Where are they? - **Things:** What digital touchpoints or physical objects would they use?	Search or Enter Website url or Name User Credentials User Validation Interactive Website for Better Understanding	[illegible] Getting user Credentials [illegible] [illegible]	[illegible] [illegible] [illegible]	[illegible] Database Allocation and updation	[illegible] Database Allocation and Updation [illegible]
Goals & motivations At each step, what is a person's primary goal or motivation? ("Help me..." or "Help me avoid...")	Access to the account Give the Graphical representation [illegible]	Inventory Sufficient Stock Authenticate a user with same credentials Spot Avoiding	Timely availability To Update the stock correctly to increase productivity	Continous updation on inventory Cost Effective Storage	Automated Inventory Management Alerts on Inventory Reduces Time
Positive moments What steps does a typical person find enjoyable, productive, fun, motivating, delightful, or exciting?	Mail Notification When the Minimum Stock Limit is Reached [illegible] Remote Access to the inventory	[illegible] Cost saving Better understanding [illegible]	Immediate updation of availability stocks Good Chatbot service Good user interaction and experience in application	[illegible] Automated Researches	Automated In Stock Information Increased Information Transparency
Negative moments What steps does a typical person find frustrating, confusing, angering, costly, or time-consuming?	[illegible] [illegible] Production Problem	password Not receiving any mail System crash due to enormous data storage [illegible]	Inappropriate input data High traffic	Supply Chain Constantly Inconsistent Inventory	Insufficient order Management Difficulty in Managing the Physical Storage Facility
Areas of opportunity How might we make each step better? What ideas do we have? What have others suggested?	Multiple User Interaction [illegible]	Analyzing using Graph Awareness a inventory management system Stock details updating	Providing Email services to customers and Retailers Better interaction with customers	Transparent performance Confirmation of Stock purchase	Reduce Human Error Develop a Network between Retailers

Need some inspiration?
See a finished version of this template to kickstart your work.
Open example →

6. PROJECT PLANNING & SCHEDULING

SPRINT PLANNING AND EXECUTION:

Activity Number	Activity Name	Detailed Activity Description	Assigned To	Duration (Start to End Date)	Status
1	Create Flask Project	An application Frame work written in Python.	Sweatha		Completed
2	Create IBM Cloud	Create and log into IBM Cloud.	Keerthana		
3	Install IBM Cloud CLI	General-Purpose developer tool that provides access to your IBM Cloud Account.	Sweatha		
4	Docker CLI	Use Docker CLI configuration to customize settings.	Sweatha		
5	Create Account in SendGrid	Create account in SendGrid to send mails.	Sweatha		
Implementing Web Application					
6	Create UI to interact with Application	Pages such as Registration, Login page, Displaying items etc.	Sweatha		Completed
7	Create IBM Db2 and connect with Python	Create IBM Db2 service in IBM Cloud and connect with python code using DB.	keerthana		
8	SendGrid Integration with Python Code	To send emails from the applications we need to integrate the SendGrid Service.	keerthana		Completed
Deployment of App in IBM Cloud					
9	Containerize the App	Need to create Docker Image of the application and push into the IBM Container Registry.			Completed
10	Upload Image to IBM Container Registry	Upload the Image to IBM Container Registry.	Keerhana		
11	Deploy in Kubernetes Cluster	Once the image is uploaded the IBM Container registry deploy the image to IBM Kubernetes Cluster.	Keerthana		
Ideation Phase					
12	Literature Survey on the Selected Project and Information Gathering	Gather information about various other inventory management systems	Deepan		Completed
13	Prepare Empathy Map	Analyse the gain and pain of the proposed project	Akila		
14	Ideation and Brainstorming	Team members pool in their ideas on the project	Deepan		
Project Design Phase I					
15	Proposed Solution	Layout the solution to be implemented	Akila		

16	Problem Solution Fit	Explore, identify and propose the use cases	Deepan	Completed
17	Solution Architecture	Diagrammatically depict the solution	Akila	

	Project design Phase II			
18	Customer Journey	Tells about how customers feel about the modules	Deepan	Completed
19	Functional Requirement	Specifies both functional and non-functional requirements	Akila	
20	Dataflow Diagram	Flowgraph of the project	Deepan	
21	Technology Architecture	Tech stacks used in the model is listed	Akila	
	Project Planning Phase			
22	Prepare Milestone and Activity List	Roles and responsibilities of each team member	Deepan	Completed
23	Sprint Delivery Plan	Various stages of the project is divided into modules	Akila	
	Project Development Phase			
24	Sprint 1	Login and Registration details	Akila	Completed
25	Sprint 2	Dashboard and add items into the cart	Deepan	
26	Sprint 3	Stock update	Akila	
27	Sprint 4	Request to customer care and contact administrator	Deepan	

SPRINT DELIVERY 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 entering my email, password, and confirming my password.	2	High	4
Sprint-1		USN-2	As a user, I can register for the application through Email.	1	Medium	4
Sprint-1	Confirmation	USN-3	As a user, I will receive confirmation email once I have registered for the application.	2	Medium	4
Sprint-1	Login	USN-4	As a user, I can log into the application by entering email & password	2	High	4
Sprint-2	Dashboard	USN-5	As a user, I can view the products which are available	4	High	4
Sprint-2	Add items to cart	USN-6	As a user, I can add the products I wish to buy to the carts.	5	Medium	4
Sprint-3	Stock Update	USN-7	As a user, I can add products which are not available in the dashboard to the stock list.	5	Medium	4
Sprint-4	Request to Customer Care	USN-8	As a user, I can contact the Customer Care Executive and request any services I want from the customer care.	5	Low	4
Sprint-4	Contact Administrator	USN-9	I can be able to report any difficulties I experience as a report	5	Medium	4

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Dat (Actual)
Sprint-1	7	6 Days	24 Oct 2022	29 Oct 2022	7	29 Oct 2022
Sprint-2	9	6 Days	31 Oct 2022	05 Nov 2022	9	05 Nov 2022
Sprint-3	5	6 Days	07 Nov 2022	12 Nov 2022	5	12 Nov 2022
Sprint-4	**10**	**6 Days**	**14 Nov 2022**	**19 Nov 2022**	**10**	**19 Nov 2022**

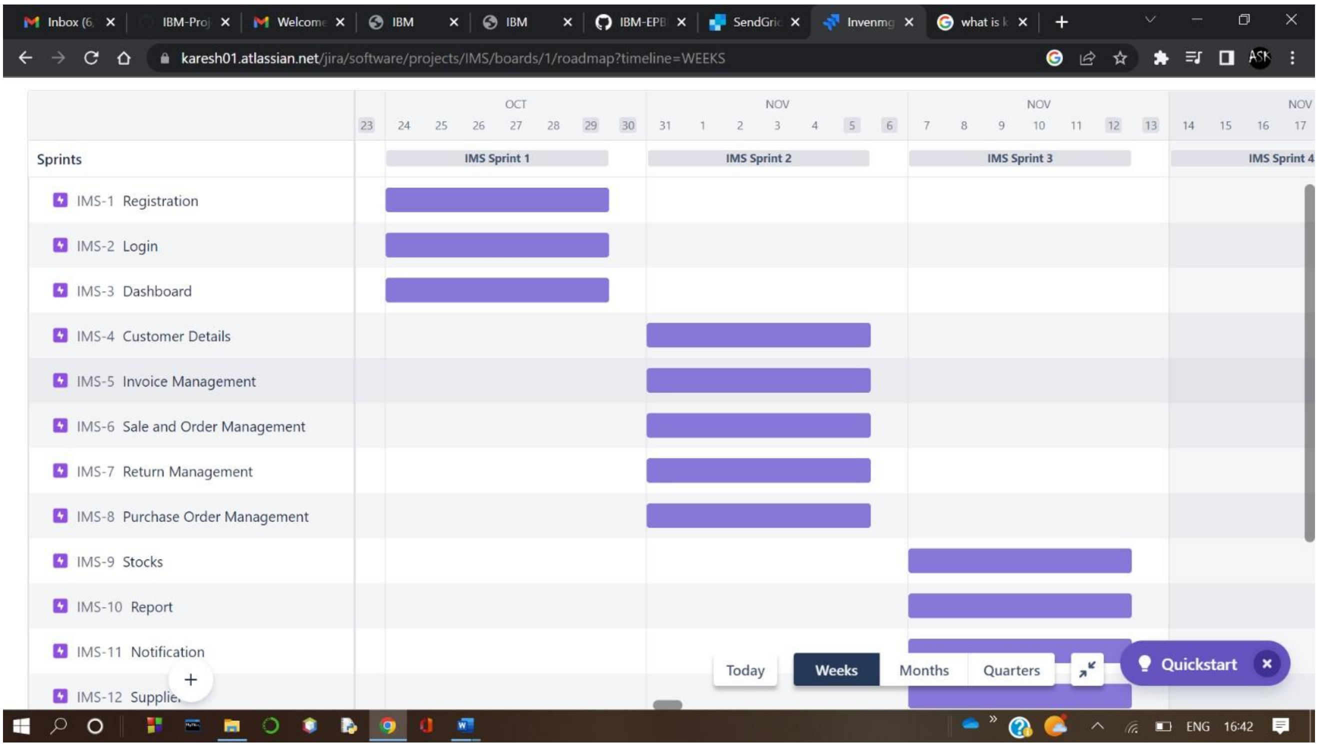
Velocity:

Sprints	Sprint Duration	Velocity	Actual Velocity
Sprint-1	6	7	0.85

Sprint-2	6	9	0.66
Sprint-3	6	5	1.2
Sprint-4	6	10	0.6

REPORTS FROM JIRA:

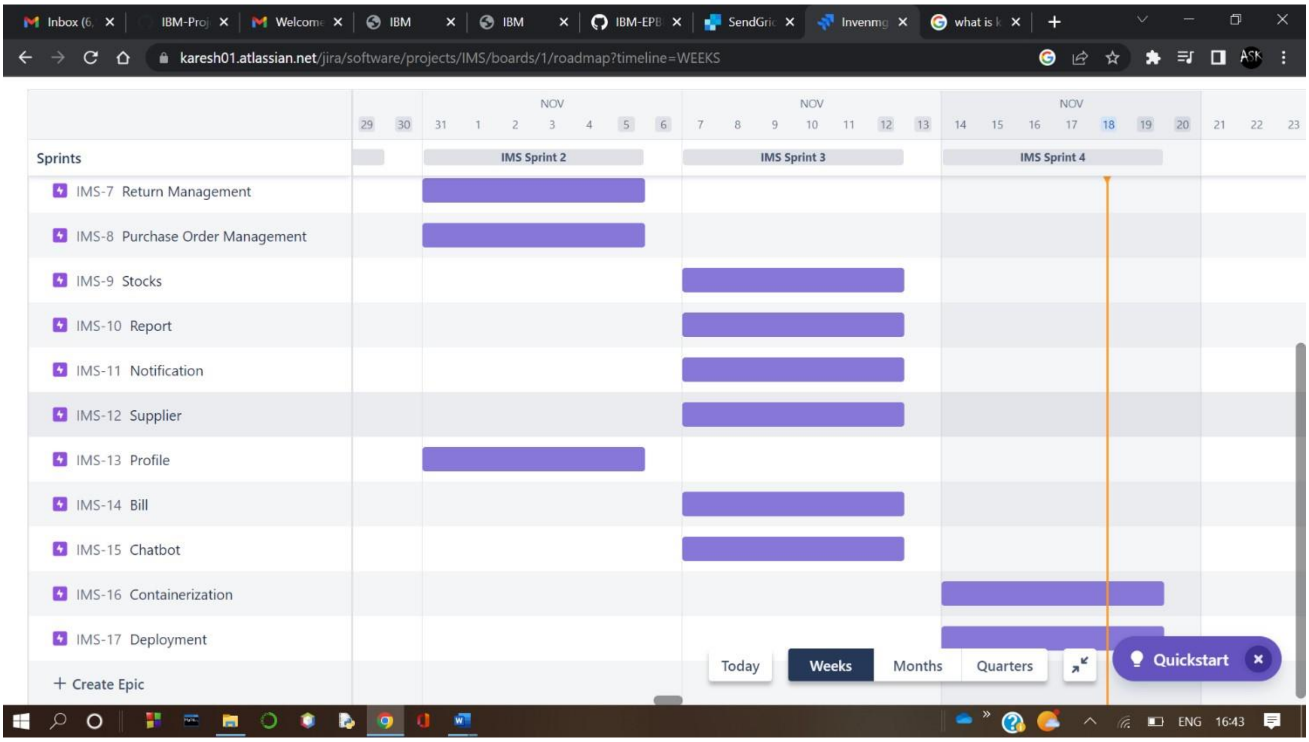
ROADMAP:



karesh01.atlassian.net/jira/software/projects/IMS/boards/1/roadmap?timeline=WEEKS

Sprints

- IMS-1 Registration
- IMS-2 Login
- IMS-3 Dashboard
- IMS-4 Customer Details
- IMS-5 Invoice Management
- IMS-6 Sale and Order Management
- IMS-7 Return Management
- IMS-8 Purchase Order Management
- IMS-9 Stocks
- IMS-10 Report
- IMS-11 Notification
- IMS-12 Supplie



Sprints

IMS-7 Return Management

IMS-8 Purchase Order Management

IMS-9 Stocks

IMS-10 Report

IMS-11 Notification

IMS-12 Supplier

IMS-13 Profile

IMS-14 Bill

IMS-15 Chatbot

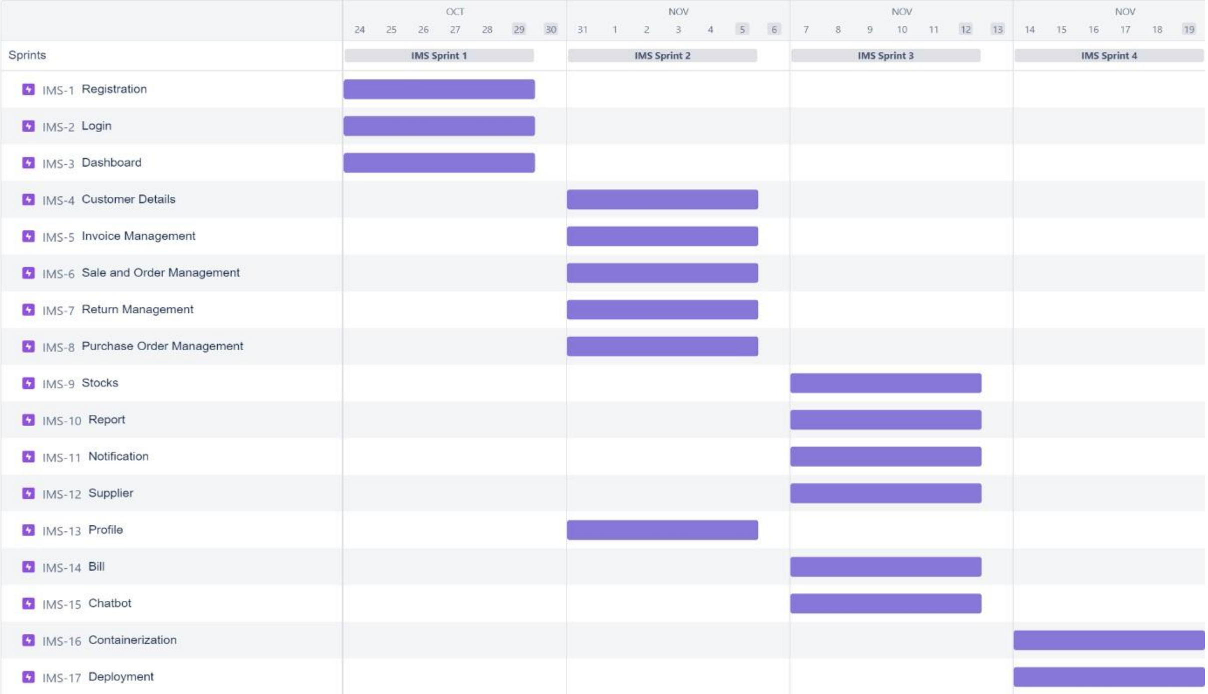
IMS-16 Containerization

IMS-17 Deployment

+ Create Epic

USAGE OF THE JIRA SOFTWARE:

Roadmap for the project:



Sprints

IMS-1 Registration

IMS-2 Login

IMS-3 Dashboard

IMS-4 Customer Details

IMS-5 Invoice Management

IMS-6 Sale and Order Management

IMS-7 Return Management

IMS-8 Purchase Order Management

IMS-9 Stocks

IMS-10 Report

IMS-11 Notification

IMS-12 Supplier

IMS-13 Profile

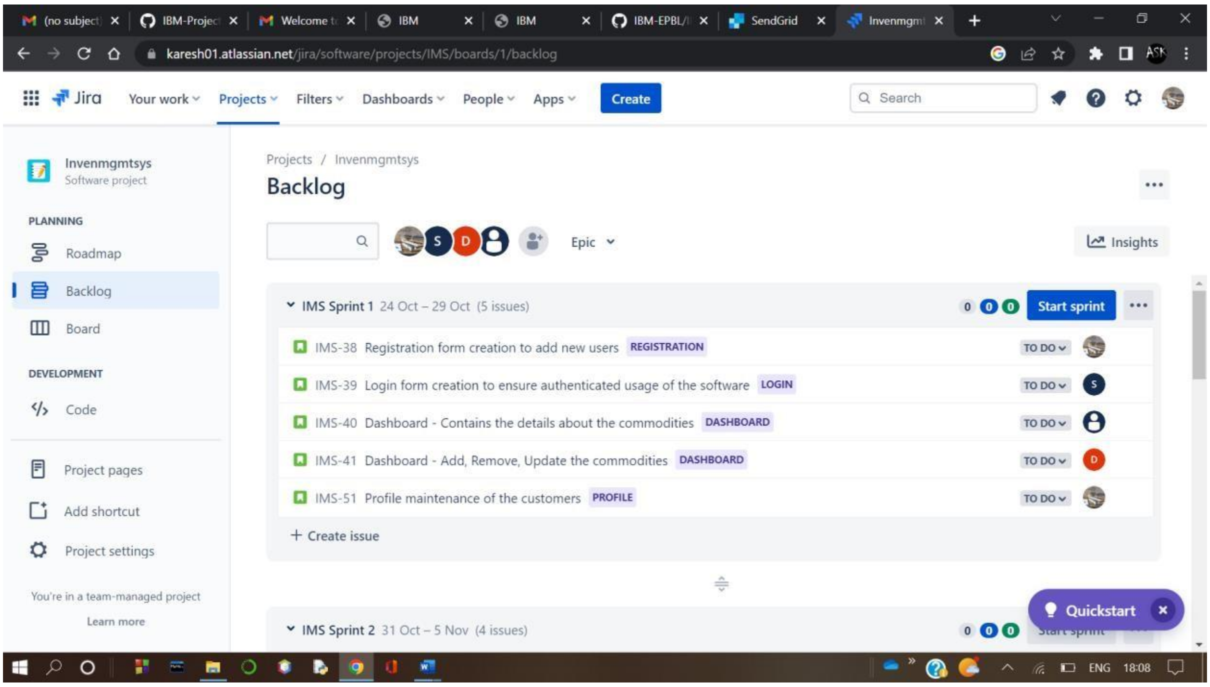
IMS-14 Bill

IMS-15 Chatbot

IMS-16 Containerization

IMS-17 Deployment

Creating the issue (Backlog) for the project:



Projects / Invenmgmtsys

Backlog

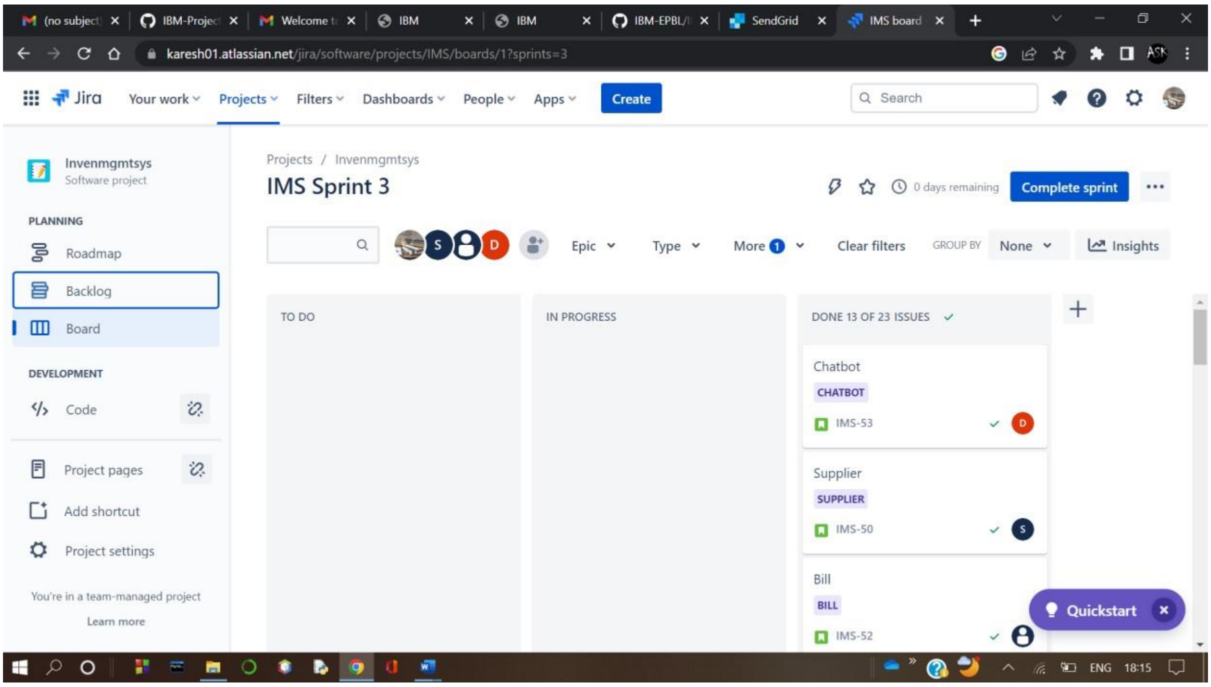
IMS Sprint 1 24 Oct – 29 Oct (5 issues)

IMS-38	Registration form creation to add new users	REGISTRATION	TO DO
IMS-39	Login form creation to ensure authenticated usage of the software	LOGIN	TO DO
IMS-40	Dashboard - Contains the details about the commodities	DASHBOARD	TO DO
IMS-41	Dashboard - Add, Remove, Update the commodities	DASHBOARD	TO DO
IMS-51	Profile maintenance of the customers	PROFILE	TO DO

+ Create issue

IMS Sprint 2 31 Oct – 5 Nov (4 issues)

Creating the board for the sprints of the project:



Projects / Invenmgmtsys

IMS Sprint 3

TO DO

IN PROGRESS

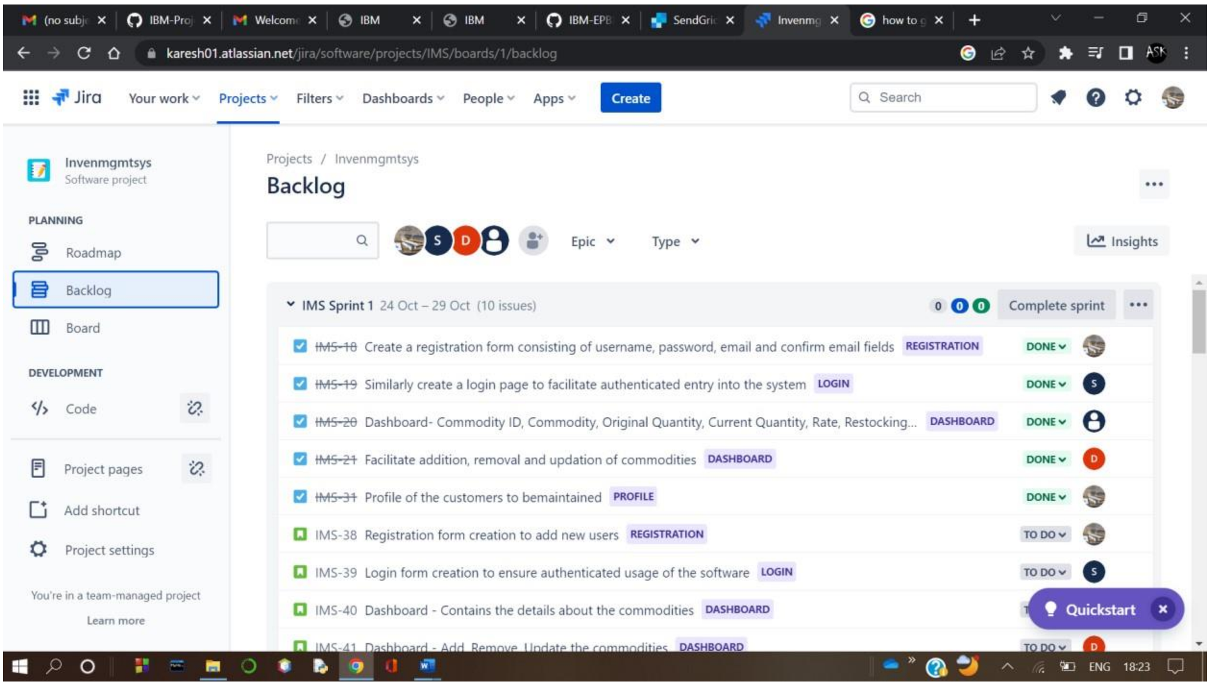
DONE 13 OF 23 ISSUES ✓

Chatbot
CHATBOT
IMS-53 ✓

Supplier
SUPPLIER
IMS-50 ✓

Bill
BILL
IMS-52 ✓

Completion of the assigned issue:



Backlog

IMS Sprint 1 24 Oct – 29 Oct (10 issues)

~~IMS-18~~	Create a registration form consisting of username, password, email and confirm email fields	REGISTRATION	DONE
~~IMS-19~~	Similarly create a login page to facilitate authenticated entry into the system	LOGIN	DONE
~~IMS-20~~	Dashboard- Commodity ID, Commodity, Original Quantity, Current Quantity, Rate, Restocking...	DASHBOARD	DONE
~~IMS-21~~	Facilitate addition, removal and updation of commodities	DASHBOARD	DONE
~~IMS-31~~	Profile of the customers to bemaintained	PROFILE	DONE
IMS-38	Registration form creation to add new users	REGISTRATION	TO DO
IMS-39	Login form creation to ensure authenticated usage of the software	LOGIN	TO DO
IMS-40	Dashboard - Contains the details about the commodities	DASHBOARD	
IMS-41	Dashboard - Add, Remove, Update the commodities	DASHBOARD	TO DO

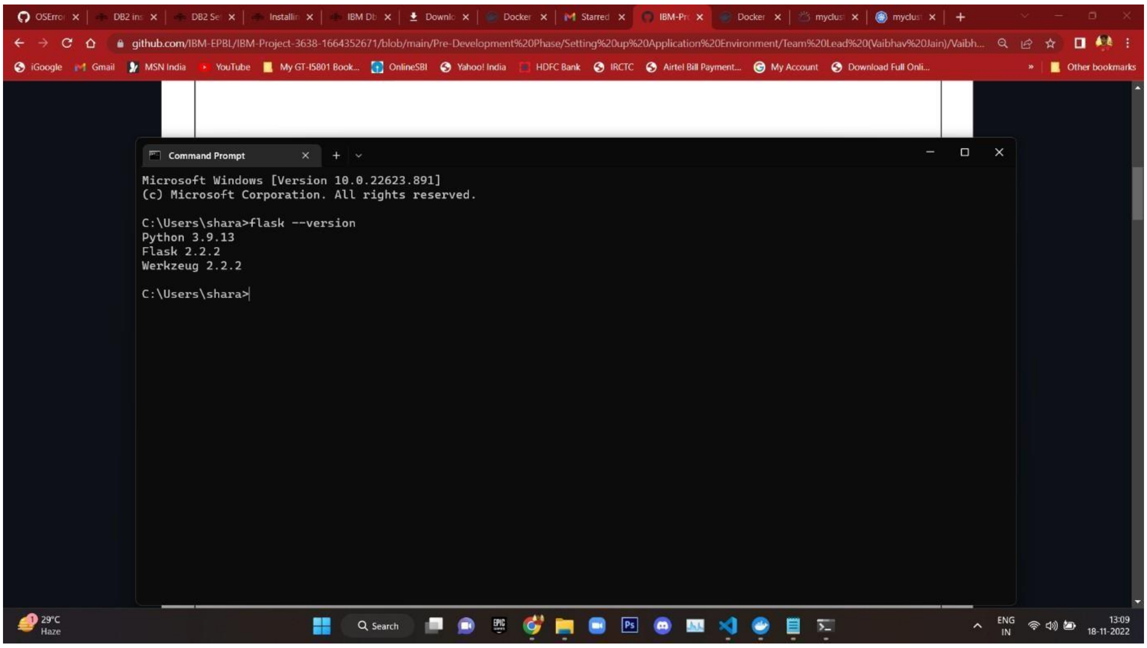
7. CODING AND SOLUTION

SETTING UP THE APPLICATION ENVIRONMENT:

TASK

- Create Flask Project
- Create IBM Cloud Account
- Install IBM Cloud CLI
- Docker CLI Installation
- Create An Account in SendGrid

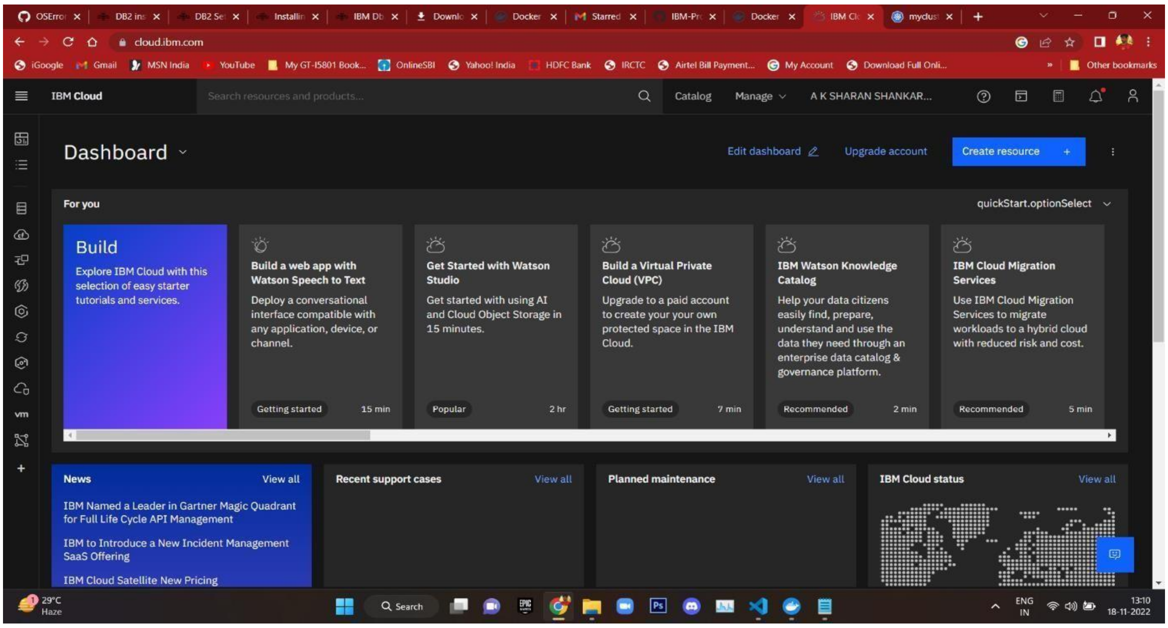
Create Flask Project



```
Microsoft Windows [Version 10.0.22623.891]
(c) Microsoft Corporation. All rights reserved.

C:\Users\shara>flask --version
Python 3.9.13
Flask 2.2.2
Werkzeug 2.2.2

C:\Users\shara>
```

github.com/IBM-EPBL/IBM-Project-3638-1664352671/blob/main/Pre-Development%20Phase/Setting%20up%20Application%20Environment/Team%20Lead%20(Vaibhav%20Jain)/Vaibh...
Command Prompt
29°C
Haze
Search
ENG
IN
13:09
18-11-2022

Create IBM Cloud Account



Dashboard

For you

Build
Explore IBM Cloud with this selection of easy starter tutorials and services.

Build a web app with Watson Speech to Text
Deploy a conversational interface compatible with any application, device, or channel.
Getting started 15 min

Get Started with Watson Studio
Get started with using AI and Cloud Object Storage in 15 minutes.
Popular 2 hr

Build a Virtual Private Cloud (VPC)
Upgrade to a paid account to create your your own protected space in the IBM Cloud.
Getting started 7 min

IBM Watson Knowledge Catalog
Help your data citizens easily find, prepare, understand and use the data they need through an enterprise data catalog & governance platform.
Recommended 2 min

IBM Cloud Migration Services
Use IBM Cloud Migration Services to migrate workloads to a hybrid cloud with reduced risk and cost.
Recommended 5 min

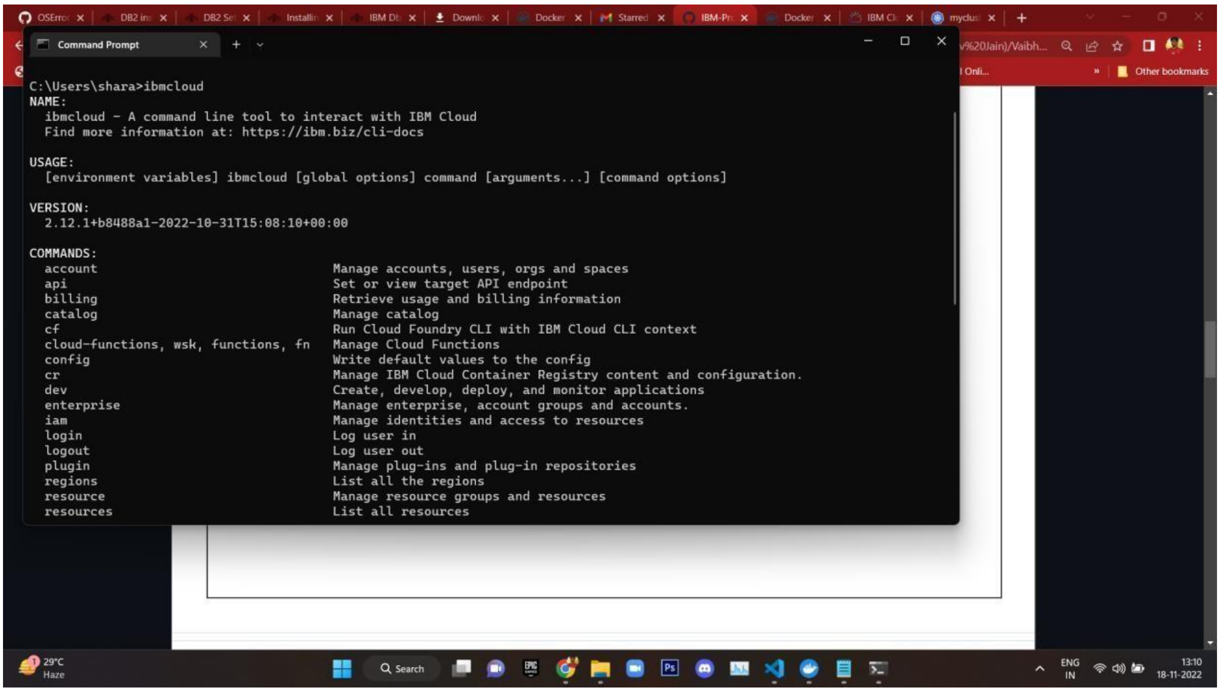
News
IBM Named a Leader in Gartner Magic Quadrant for Full Life Cycle API Management
IBM to Introduce a New Incident Management SaaS Offering
IBM Cloud Satellite New Pricing

Recent support cases

Planned maintenance

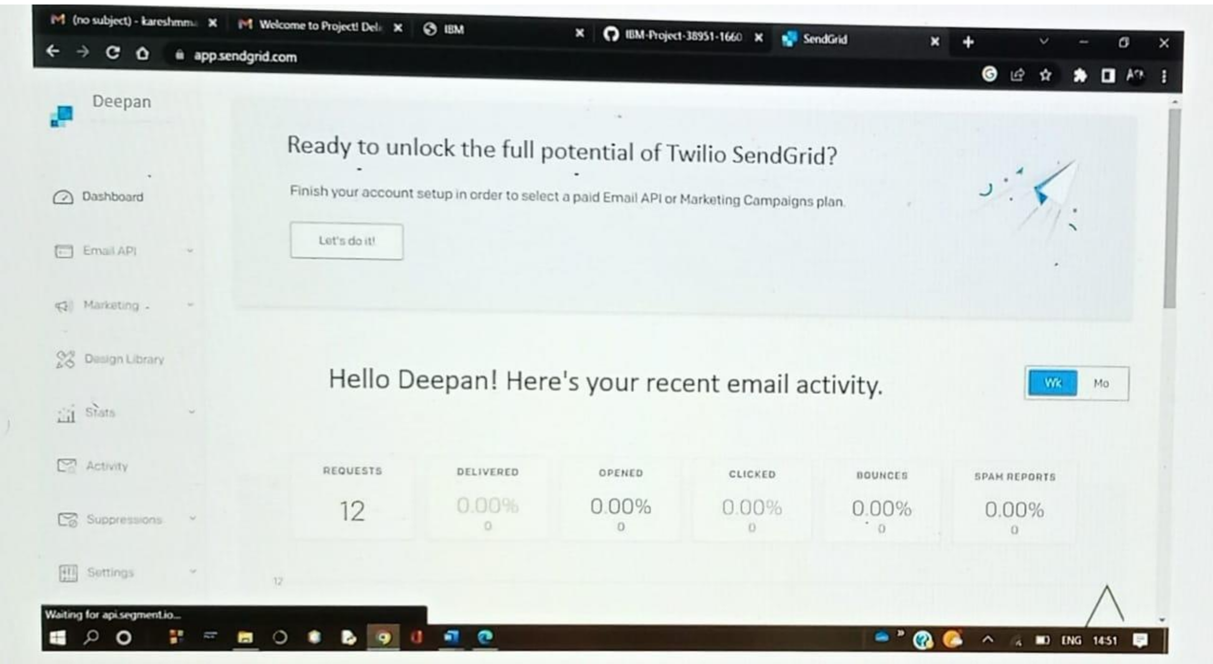
IBM Cloud status

Install IBM Cloud CLI



```
C:\Users\shara>ibmcloud
NAME:
  ibmcloud - A command line tool to interact with IBM Cloud
  Find more information at: https://ibm.biz/cli-docs

USAGE:
  [environment variables] ibmcloud [global options] command [arguments...] [command options]

VERSION:
  2.12.1+b8488a1-2022-10-31T15:08:10+00:00

COMMANDS:
  account                               Manage accounts, users, orgs and spaces
  api                                   Set or view target API endpoint
  billing                               Retrieve usage and billing information
  catalog                               Manage catalog
  cf                                    Run Cloud Foundry CLI with IBM Cloud CLI context
  cloud-functions, wsk, functions, fn   Manage Cloud Functions
  config                                Write default values to the config
  cr                                    Manage IBM Cloud Container Registry content and configuration.
  dev                                   Create, develop, deploy, and monitor applications
  enterprise                            Manage enterprise, account groups and accounts.
  iam                                   Manage identities and access to resources
  login                                 Log user in
  logout                                Log user out
  plugin                                Manage plug-ins and plug-in repositories
  regions                               List all the regions
  resource                              Manage resource groups and resources
  resources                             List all resources
```

Create An Account in SendGrid



Ready to unlock the full potential of Twilio SendGrid?

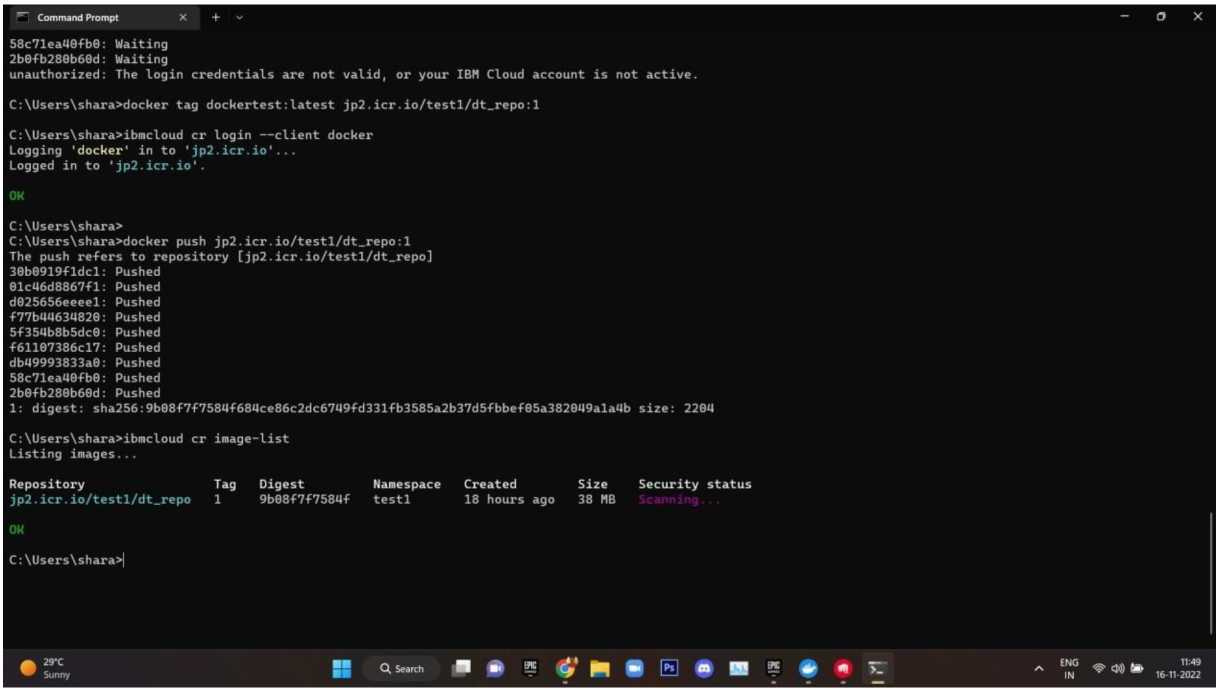
Finish your account setup in order to select a paid Email API or Marketing Campaigns plan.

Hello Deepan! Here's your recent email activity.

REQUESTS	DELIVERED	OPENED	CLICKED	BOUNCES	SPAM REPORTS
12	0.00% 0	0.00% 0	0.00% 0	0.00% 0	0.00% 0

Waiting for api.segment.io...

Docker CLI Installation



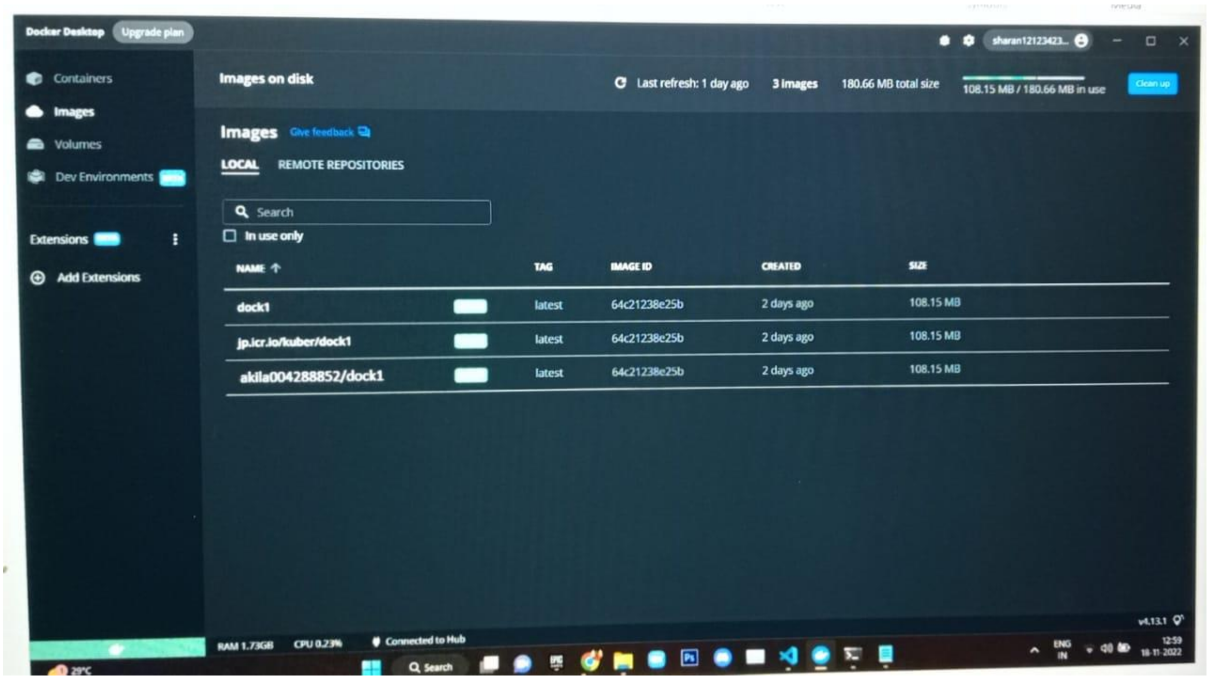
```
58c71ea40fb0: Waiting
2b0fb280b60d: Waiting
unauthorized: The login credentials are not valid, or your IBM Cloud account is not active.

C:\Users\shara>docker tag dockertest:latest jp2.icr.io/test1/dt_repo:1

C:\Users\shara>ibmcloud cr login --client docker
Logging 'docker' in to 'jp2.icr.io'...
Logged in to 'jp2.icr.io'.

OK

C:\Users\shara>
C:\Users\shara>docker push jp2.icr.io/test1/dt_repo:1
The push refers to repository [jp2.icr.io/test1/dt_repo]
30b0919f1dc1: Pushed
01c46d8867f1: Pushed
d025656eeee1: Pushed
f77b44634820: Pushed
5f354b8b5dc0: Pushed
f61107386c17: Pushed
db49993833a0: Pushed
58c71ea40fb0: Pushed
2b0fb280b60d: Pushed
1: digest: sha256:9b08f7f7584f684ce86c2dc6749fd331fb3585a2b37d5fbbef05a382049a1a4b size: 2204

C:\Users\shara>ibmcloud cr image-list
Listing images...

Repository               Tag   Digest         Namespace   Created         Size    Security status
jp2.icr.io/test1/dt_repo 1     9b08f7f7584f   test1       18 hours ago    38 MB   Scanning...

OK

C:\Users\shara>
```

DEPLOYMENT OF APPLICATION IN IBM CLOUD:

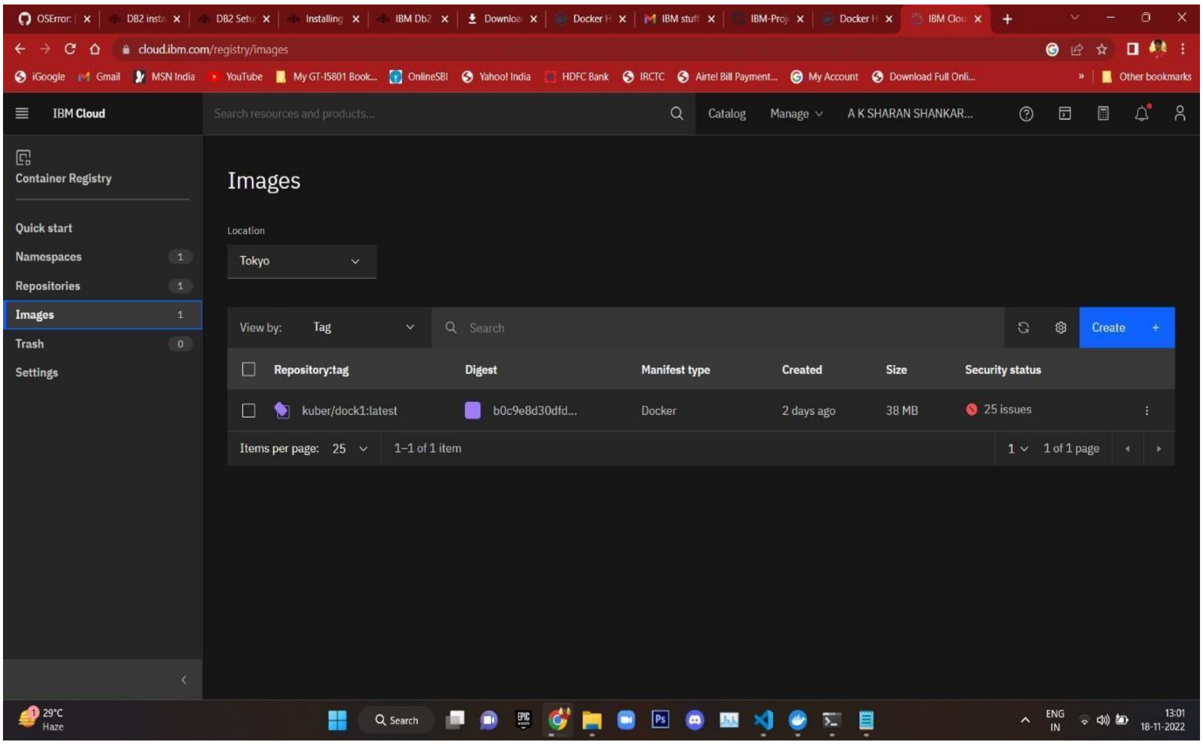
TASK:

- Containerise the app
- Upload image to IBM container registry
- Deployment in Kubernetes cluster

Containerise the app:



NAME ↑	TAG	IMAGE ID	CREATED	SIZE
dock1	latest	64c21238e25b	2 days ago	108.15 MB
jp.icr.io/kuber/dock1	latest	64c21238e25b	2 days ago	108.15 MB
akila004288852/dock1	latest	64c21238e25b	2 days ago	108.15 MB



Images

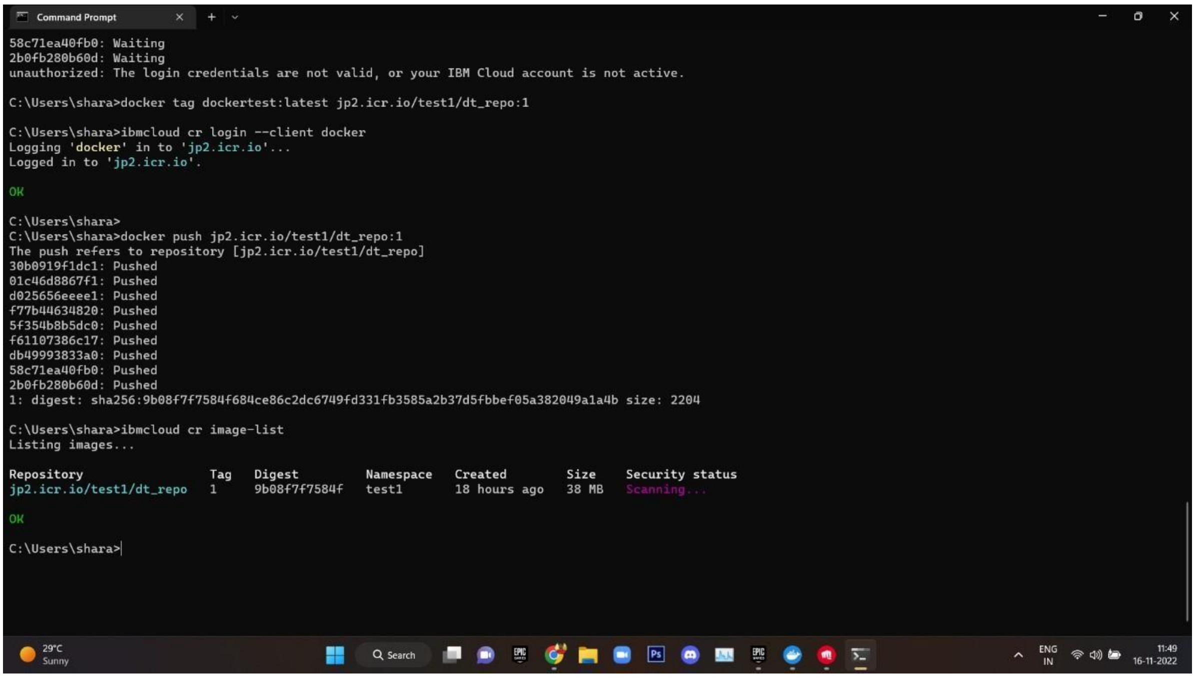
Location: Tokyo

View by: Tag

Repository:tag	Digest	Manifest type	Created	Size	Security status
kuber/dock1:latest	b0c9e8d30dfd...	Docker	2 days ago	38 MB	25 issues

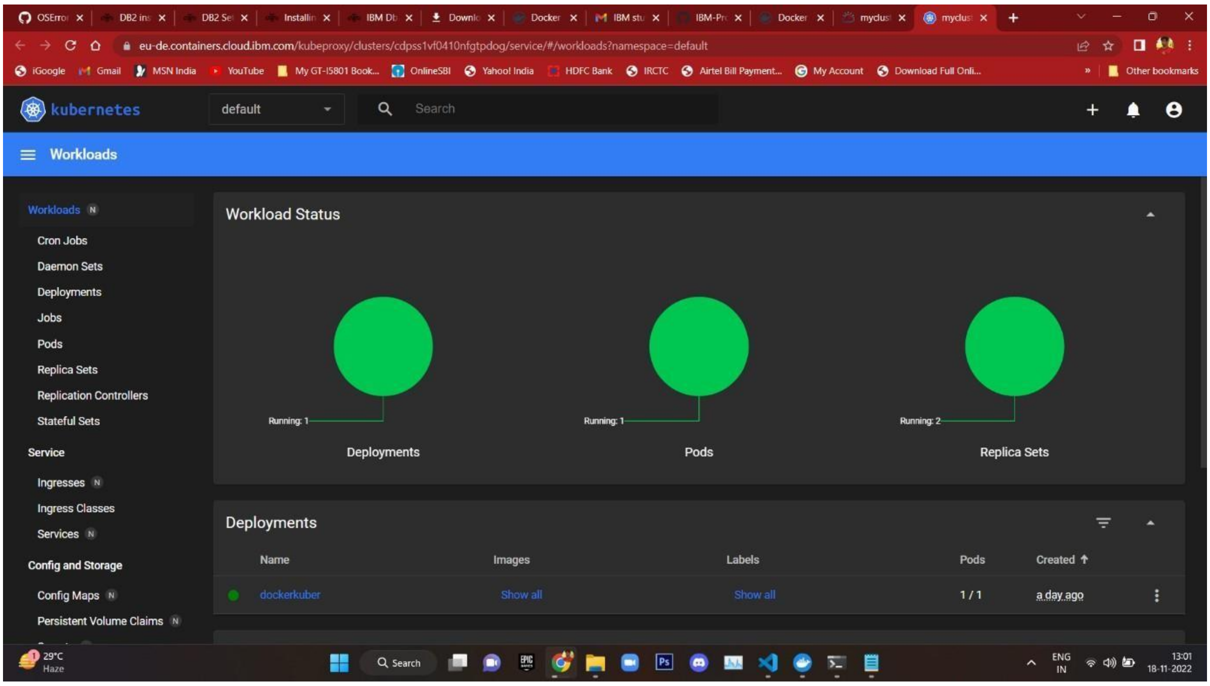
Items per page: 25 1–1 of 1 item 1 of 1 page

Upload image to IBM Container Registry:



```
58c71ea40fb0: Waiting
2b0fb280b60d: Waiting
unauthorized: The login credentials are not valid, or your IBM Cloud account is not active.

C:\Users\shara>docker tag dockertest:latest jp2.icr.io/test1/dt_repo:1

C:\Users\shara>ibmcloud cr login --client docker
Logging 'docker' in to 'jp2.icr.io'...
Logged in to 'jp2.icr.io'.

OK

C:\Users\shara>
C:\Users\shara>docker push jp2.icr.io/test1/dt_repo:1
The push refers to repository [jp2.icr.io/test1/dt_repo]
30b0919f1dc1: Pushed
01c46d8867f1: Pushed
d025656eeee1: Pushed
f77b44634820: Pushed
5f354b8b5dc0: Pushed
f61107386c17: Pushed
db49993833a0: Pushed
58c71ea40fb0: Pushed
2b0fb280b60d: Pushed
1: digest: sha256:9b08f7f7584f684ce86c2dc6749fd331fb3585a2b37d5fbbef05a382049a1a4b size: 2204

C:\Users\shara>ibmcloud cr image-list
Listing images...

Repository                Tag   Digest         Namespace   Created        Size    Security status
jp2.icr.io/test1/dt_repo  1     9b08f7f7584f   test1       18 hours ago   38 MB   Scanning...

OK

C:\Users\shara>
```

Deploy in Kubernetes Cluster:



Name	Images	Labels	Pods	Created
dockerkuber	Show all	Show all	1 / 1	a day ago

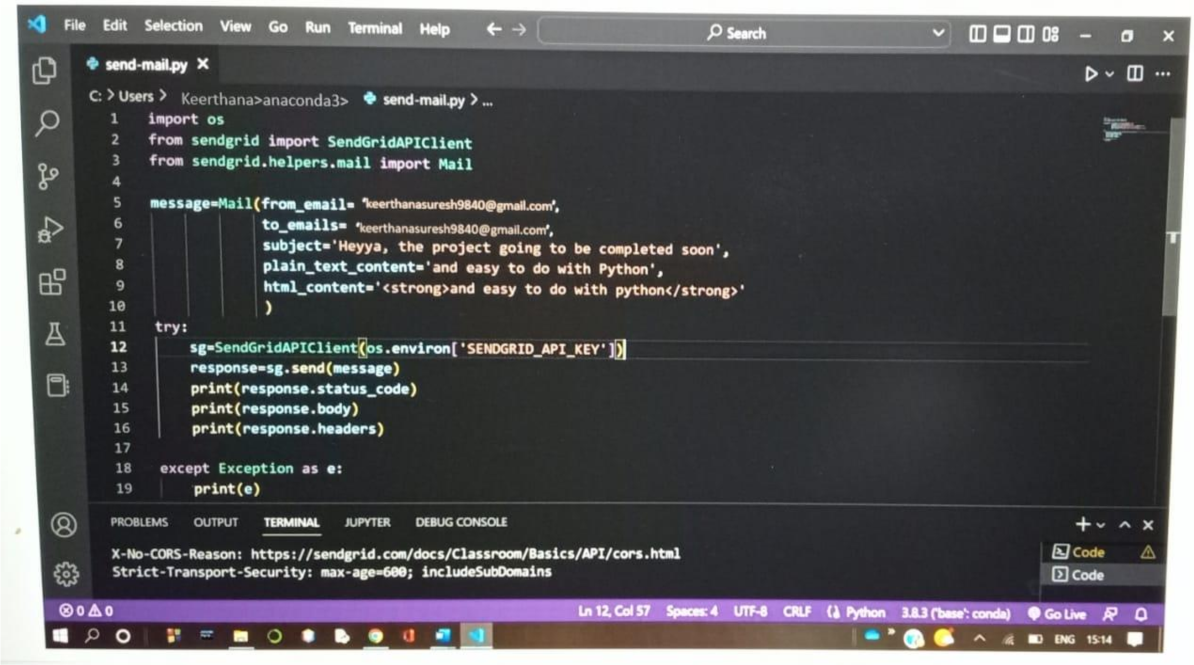
INTEGRATING SENDGRID SERVICE:

Here, SENDGRID_API_KEY must be substituted with unique API Key that is generated when you create an API Key in the SendGrid account.

SendGrid integration with Python code:

CONCLUSION:

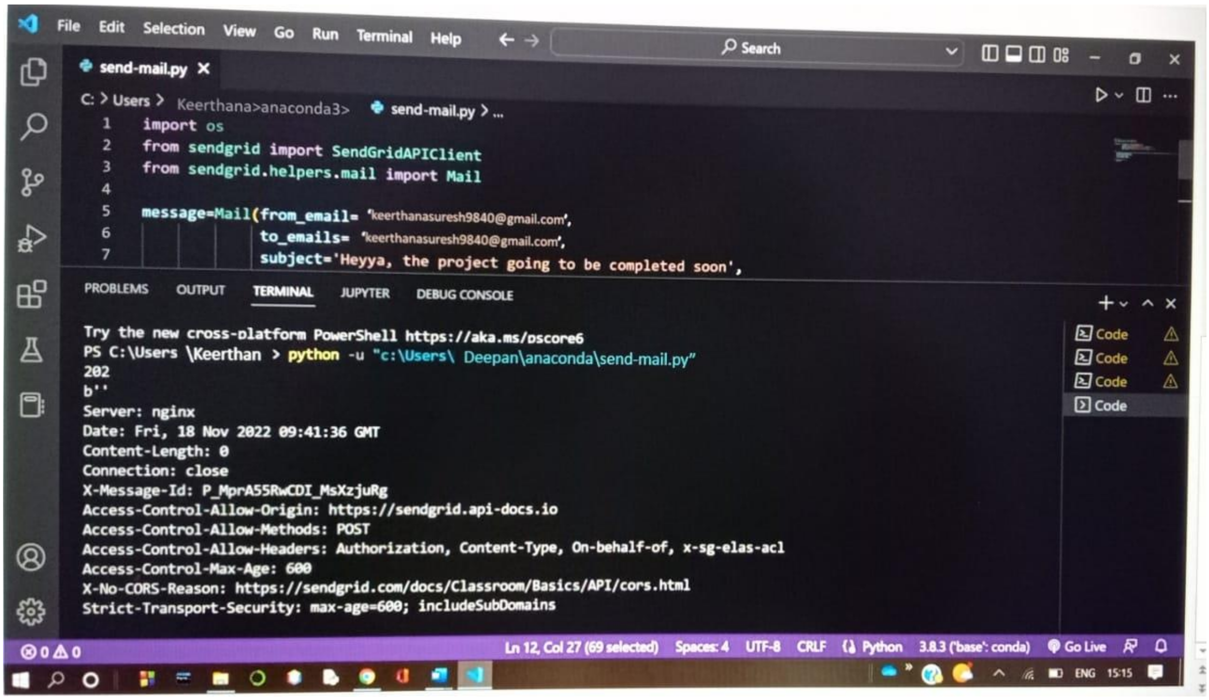
A manual pen and paper system must be replaced with an inventory management



```python
import os
from sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail

message=Mail(from_email= ‘keerthanasuresh9840@gmail.com’,
                to_emails= ‘keerthanasuresh9840@gmail.com’,
                subject='Heyya, the project going to be completed soon',
                plain_text_content='and easy to do with Python',
                html_content='<strong>and easy to do with python</strong>'
                )
 try:
     sg=SendGridAPIClient(os.environ['SENDGRID_API_KEY'])
     response=sg.send(message)
     print(response.status_code)
     print(response.body)
     print(response.headers)

 except Exception as e:
     print(e)
```

```
X-No-CORS-Reason: https://sendgrid.com/docs/Classroom/Basics/API/cors.html
Strict-Transport-Security: max-age=600; includeSubDomains
```

system. Its primary objective is to regulate how the products are moved and stored, with the added advantages of improved security and quicker handling. A software programme called an inventory management system is essential for keeping track of a certain retailer's stock levels. Additionally, it can offer insightful data to sales analytics. In the end, it serves as a company's lifeline because it drives profitability by creating sales. A company's overall success may be significantly impacted by how it manages its inventory.



```python
import os
from sendgrid import SendGridAPIClient
from sendgrid.helpers.mail import Mail

message=Mail(from_email= 'keerthanasuresh9840@gmail.com',
                to_emails= 'keerthanasuresh9840@gmail.com',
                subject='Heyya, the project going to be completed soon',
```

```
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users \Keerthan > python -u "c:\Users\ Deepan\anaconda\send-mail.py"
202
b''
Server: nginx
Date: Fri, 18 Nov 2022 09:41:36 GMT
Content-Length: 0
Connection: close
X-Message-Id: P_MprA55RwCDI_MsXzjuRg
Access-Control-Allow-Origin: https://sendgrid.api-docs.io
Access-Control-Allow-Methods: POST
Access-Control-Allow-Headers: Authorization, Content-Type, On-behalf-of, x-sg-elas-acl
Access-Control-Max-Age: 600
X-No-CORS-Reason: https://sendgrid.com/docs/Classroom/Basics/API/cors.html
Strict-Transport-Security: max-age=600; includeSubDomains
```

Note: For security reasons, the key is not exposed.

IMPLEMENTING WEB APPLICATION:

TASK

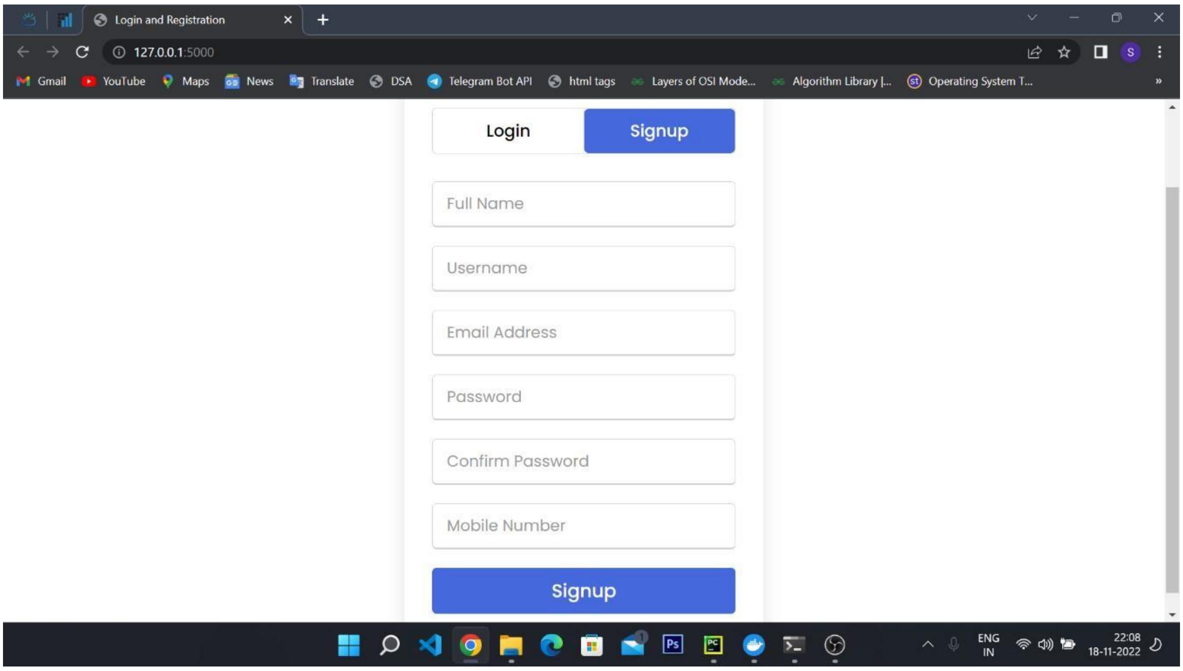
- **Create UI to interact with application**
  - Registration Page

- Login Page
- Display items in the Dashboard
- Adding items
- Removing items

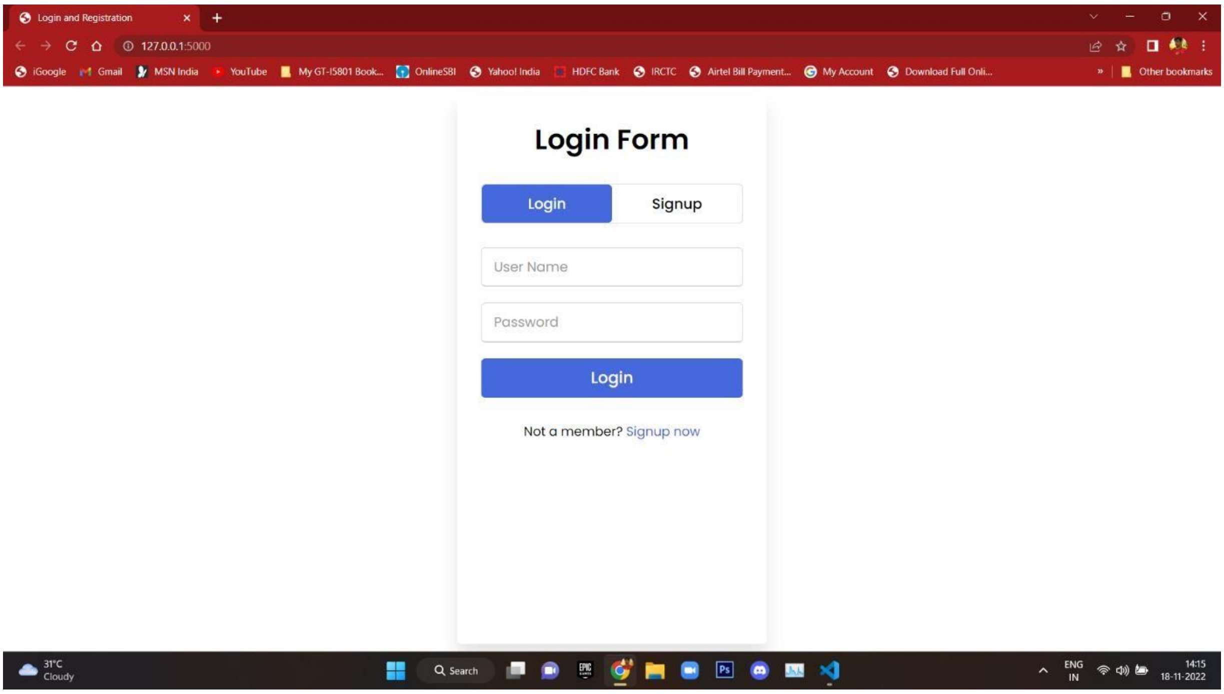
- **Create IBM DB2 and connect with Node.js**

Create UI to interact with application

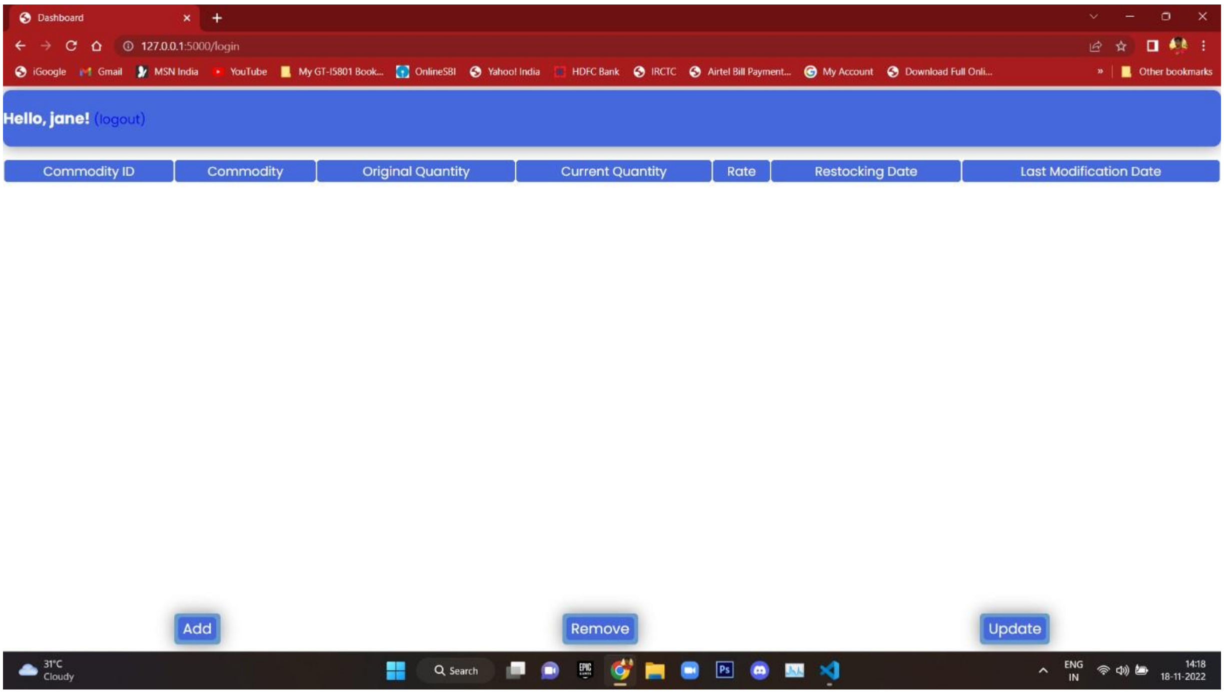
Registration Page



Login Page

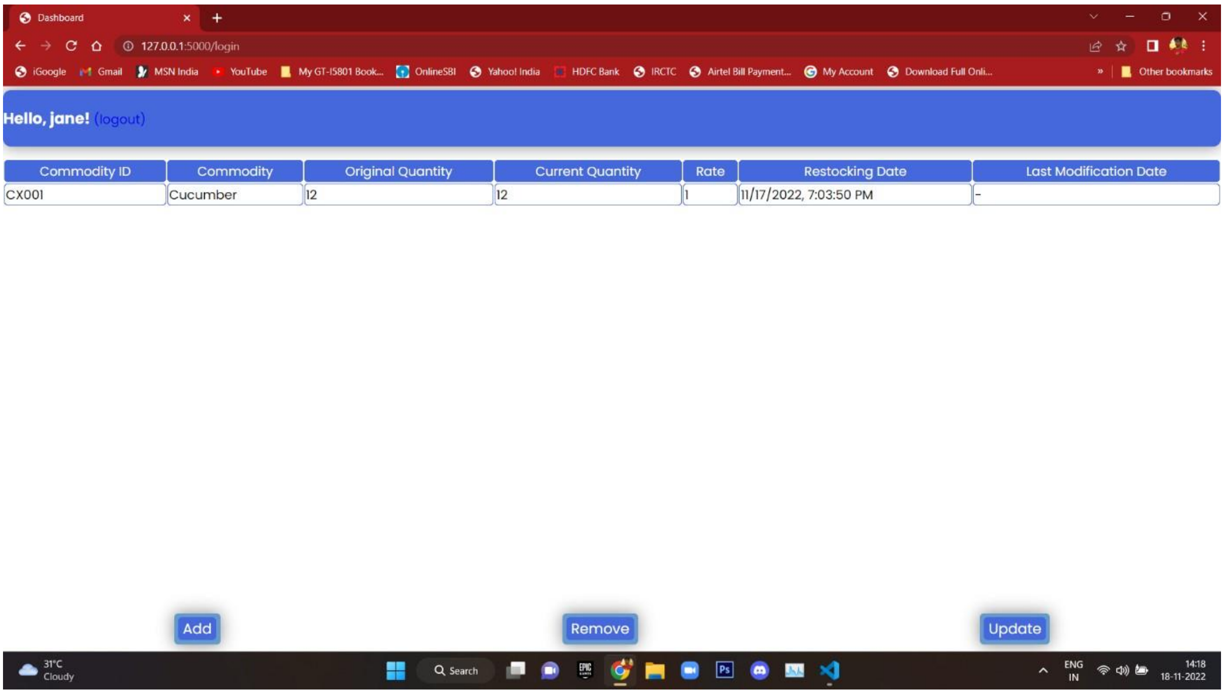


Display items in the Dashboard



Hello, jane! (logout)

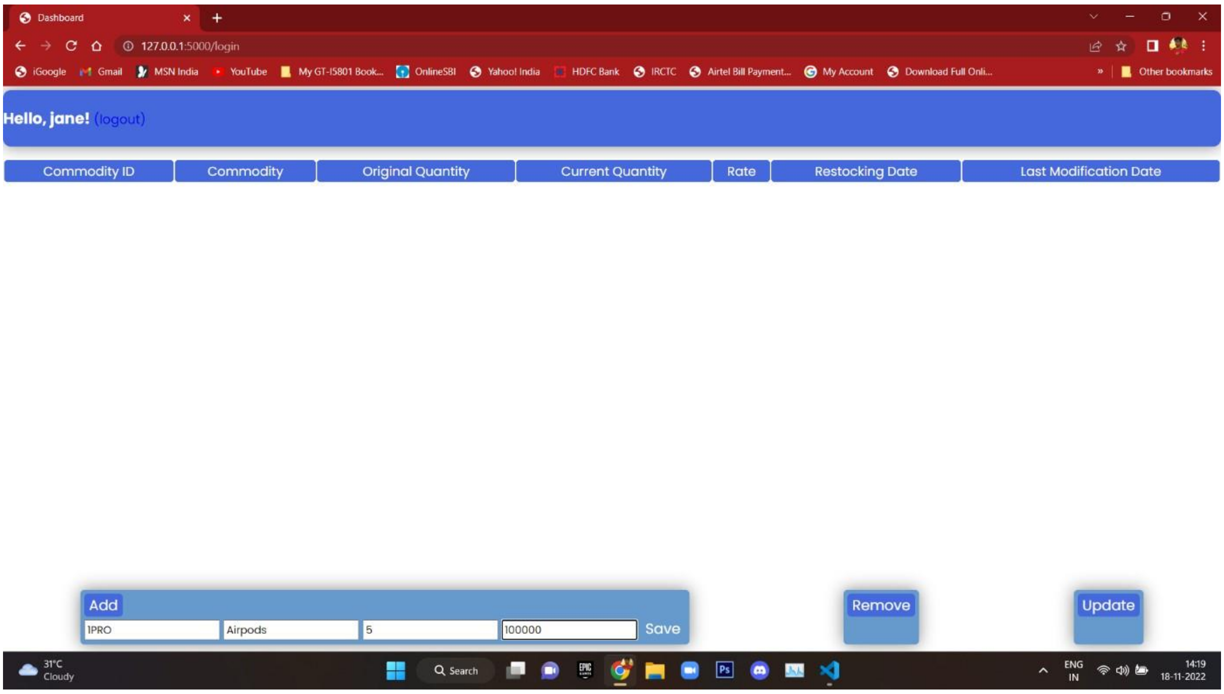
Commodity ID	Commodity	Original Quantity	Current Quantity	Rate	Restocking Date	Last Modification Date



Hello, jane! (logout)

Commodity ID	Commodity	Original Quantity	Current Quantity	Rate	Restocking Date	Last Modification Date
CX001	Cucumber	12	12	1	11/17/2022, 7:03:50 PM	-

Adding items

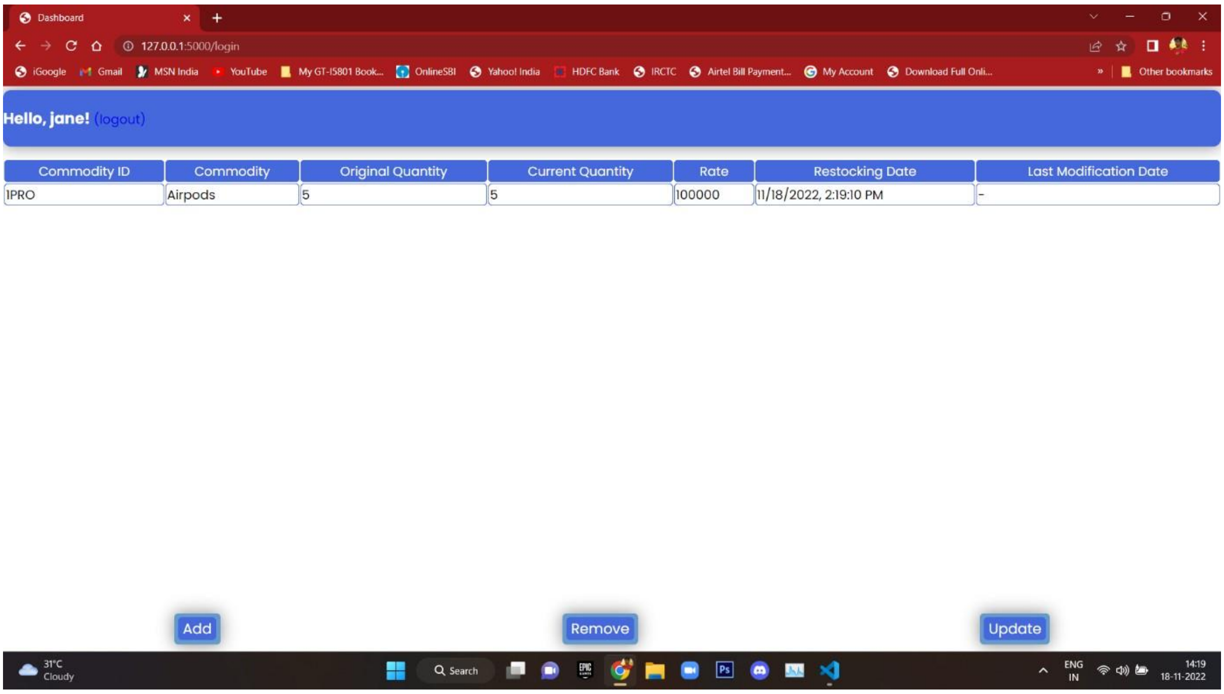


Hello, jane! (logout)

Commodity ID	Commodity	Original Quantity	Current Quantity	Rate	Restocking Date	Last Modification Date

Add

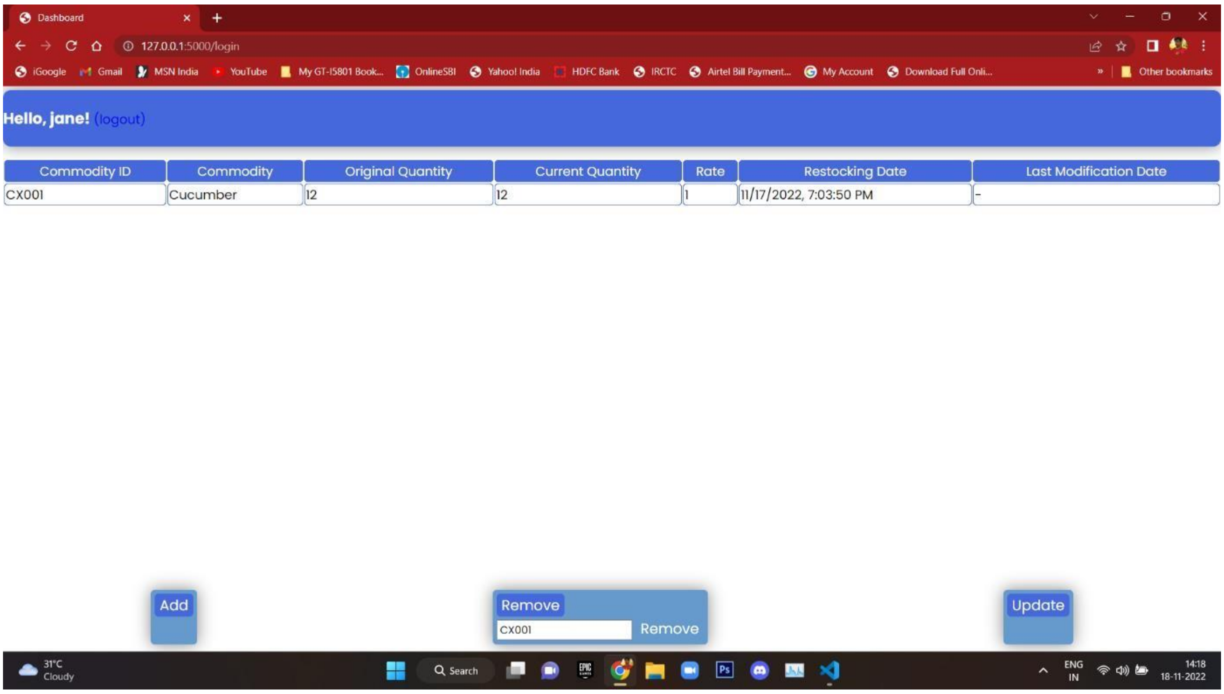
1PRO	Airpods	5	100000	Save



Hello, jane! (logout)

Commodity ID	Commodity	Original Quantity	Current Quantity	Rate	Restocking Date	Last Modification Date
IPRO	Airpods	5	5	100000	11/18/2022, 2:19:10 PM	-

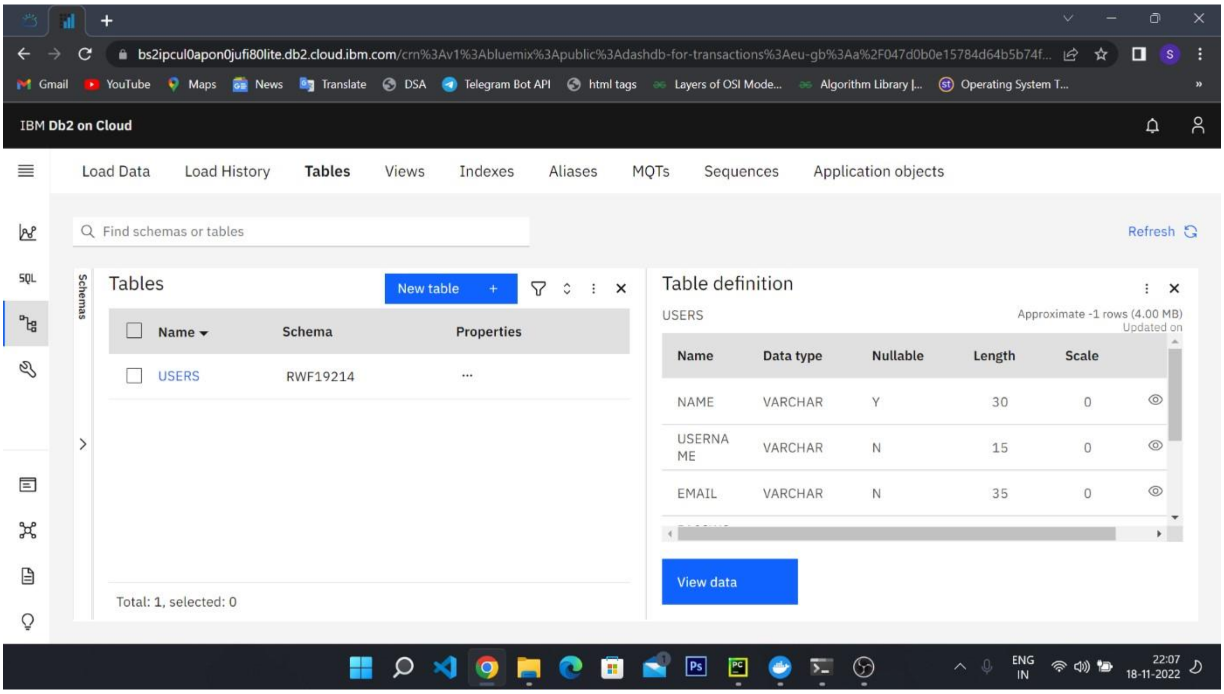
Removing Items



Hello, jane! (logout)

Commodity ID	Commodity	Original Quantity	Current Quantity	Rate	Restocking Date	Last Modification Date
CX001	Cucumber	12	12	1	11/17/2022, 7:03:50 PM	-

Create IBM DB2 and connect with Node.js



Tables

Name	Schema	Properties
USERS	RWF19214	...

Total: 1, selected: 0

Table definition

USERS

Approximate -1 rows (4.00 MB)

Name	Data type	Nullable	Length	Scale
NAME	VARCHAR	Y	30	0
USERNAME	VARCHAR	N	15	0
EMAIL	VARCHAR	N	35	0

8. TESTING

The following 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).

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	8	7	1	2	18
Duplicate	2	0	2	0	4
External	2	3	1	2	8
Fixed	12	1	5	17	35
Not Reproduced	0	0	1	0	1

Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	16	13	23	76

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	6	0	0	6
Client Application	55	0	0	55
Security	4	0	0	4
Outsource Shipping	3	0	0	3
Exception Reporting	8	0	0	8
Final Report Output	4	0	0	4
Version Control	2	0	0	2

9. ADVANTAGES AND DISADVANTAGES

Real-time inventory tracking makes it easier to manage your inventory and guarantees that you have enough stock on hand to complete orders. By avoiding overstocking, you can cut back on the expense and labour-intensive manual processes involved in maintaining it.

The advantages are:

- The advantages of using effective inventory management methods and software include increased agility and cost optimization.

- Reduces the need for huge working capital, increases cash flow, and gives you the money you need to pay employees, produce new products, or engage in any other business activity.
- To save time on inventory forecasting, you can automate your inventory management process.
- You may move data with ease and keep tabs on business activity by using a central data warehouse.
- Complete and correct customer orders in a timely manner.

The disadvantages are:

- The potential for a system crash is among the biggest issues with any computerised system.
- A valuable tool to use when looking for possible financial information or personal information of owners, vendors, or customers is an inventory system connected to point-of-sale devices and accounting.
- It is simple to omit time-consuming physical inventory audits when everything is automated which is necessary too.

10. RESULTS AND CONCLUSION

A manual pen and paper system must be replaced with an inventory management system. Its primary objective is to regulate how the products are moved and stored, with the added advantages of improved security and quicker handling. A software programme called an inventory management system is essential for keeping track of a certain retailer's stock levels. Additionally, it can offer insightful data to sales analytics. In the end, it serves as a company's lifeline because it drives profitability by creating sales. A company's overall success may be significantly impacted by how it manages its inventory.

Thus, a successful “Inventory management System for Retailers” using Python, SendGrid and IBM Cloud Services (IBM DB2, IBM Container registry, IBM Kubernetes) has been developed.

11. FUTURESCOPE:

- Inventory Management System will be seen as a strategic asset by successful businesses rather than a burdensome expense or an unavoidable evil.
- Effective inventory management will depend on collaboration with supply chain partners and a holistic approach to supply chain management.
- Decisions about the deployment of inventory will be significantly impacted by the changing nature of globalization.
- The main drivers for modifying supply chain and inventory strategies will be an increased emphasis on supply chain security and worries about the quality of inventory itself.

12. APPENDIX:

Source Code: main.py:

```
from flask import Flask, render_template, request, session, redirect, url_for

from database_handlers import * from ibm_db_dbi import IntegrityError,

ProgrammingError from json import loads, dumps


print('entering flask') app = Flask(__name__)

app.secret_key = 'temp string. tolerate this for now pls.'

FIELDS = ['cid', 'cname', 'oqty', 'cqty', 'rate', 'date']

@app.route('/') def

login_initial():

    """

    Handles the entrypoint to the website. Acts as a landing page.

    :return: Initial Login page
    """     return render_template('login.html',
err_msg='') @app.route('/login', methods =
['POST']) def login_check():
    """
    Handles the login process
    :return: Login page
    """

   # capture the username and the password entered by the user and remove trailing and leading
whitespaces            username  =  request.form['uname'].rstrip().rstrip()           entered_password  =
request.form['password']
    # get the expected password corresponding to the username
expected_password = fetch_password(username)     if not
expected_password:
        # if expected password is not found in the database, it means no such account is registered yet.
        return render_template('login.html', err_msg = 'Register an account first!')
elif entered_password != expected_password:
        # if the entered and expected password don't match, it means the password entered is wrong.
        return render_template('login.html', err_msg = 'Wrong username or password')
    else:
      # if no errors occur, log the user in and create a session.
session.update({'username': username})
        # fetch the inventory items corresponding to the user. It's stored in the form of a JSON as CLOB
tableContents = fetch_table_contents(username)
        # the value returned above will be of the form list[tuple[str/int/float, str/int/float, ...]]
if tableContents[0][0] is not None:
```

```
            # if the contents are actually present, convert the JSON to python Dict.
tableContents = loads(tableContents[0][0])
            # print(tableContents['cid'][1])
else:
            # if no content is present, pass an empty list to the template which will be evaluated as 'false'
by the
            # jinja2 template engine.
            tableContents = []
        return render_template('dashboard.html', greeting=f'Hello, {username}!',
tableContents=tableContents)     # unreachable code. It's there because I'm
paranoid.
  return 'You weren\'t supposed to be here :/'
@app.route('/signup', methods=['POST']) def
signup():
    """
    Handles the signup process.
    :return: Signup Webpage or Dashboard
    ""
    # capture the details passed from the form and remove trailing and leading whitespaces.
    fullName = request.form.get('fname').lstrip().rstrip()
username = request.form.get('uname').lstrip().rstrip()
email = request.form.get('email').lstrip().rstrip()     password
= request.form.get('password').lstrip().rstrip()     mobile =
request.form.get('mobile').lstrip().rstrip()
    try:
        # try to create account with the given details
create_account(fullName, username, email, password, mobile)     except
IntegrityError:
        # integrity error means either NOT NULL or UNIQUE constraint has been violated.
        # the former being highly unlikely because of the validation checks in the front end.
        return render_template('login.html', err_msg = 'The account exists already. Please log in.')
    else:
        # if no errors occur, log the user in and create a session.
        # session.update({'username': username})

 # redirect the user to the login procedure, code = 307 specifies to preserve the HTTP method used
originally
        # (POST in this case)         return
redirect(url_for('login_check'), code=307)
        # return render_template('dashboard.html', greeting = f'Welcome, {fullName.split()[0]}!
({username})')     # unreachable code     return 'You
weren\'t supposed to be here. Please go back'
@app.route('/check-username', methods = ['POST']) def
check_username():
    """
    Checks if the username is already present in the database or not.
    :return: 'Exists' if present, else return the passed email itself.
    """
    # get the passed email and remove leading and trailing whitespaces, if any.
    passed_username = request.form.get('username').lstrip().rstrip()
 # check if the email exists or not     exists =
check_username_existence(passed_username)
```

```
    if exists:
        return 'Exists', 403
return passed_username, 200
@app.route('/check-email', methods = ['POST']) def
check_email():
    """
    Checks if the email is already present in the database or not.
    :return: 'Exists' if present, else return the passed email itself.
"""

    # get the passed email and remove leading and trailing whitespaces, if any.
passed_email = request.form.get('email').lstrip().rstrip()
    # check if the email exists or not     exists =
check_email_existence(passed_email)
    if exists:
        return 'Exists', 403
return passed_email, 200
@app.route('/add-commodity', methods = ['POST']) def
add_commodity():     # print(request.form)     details =
request.form.to_dict()     tableContents =
fetch_table_contents(session.get('username'))     if
tableContents[0][0] is None:
        tableContents = {field: [] for field in FIELDS}
    else:
        tableContents = loads(tableContents[0][0])     if
details.get('cid') in list(tableContents.values())[0]:
        return 'Such an item already exists.', 403
for key, value in tableContents.items():
value.append(details.get(key))     details_json
= dumps(tableContents)
print(details_json)     print(details)    try:
        add_details_to_db(session.get('username'), details_json) except
    Exception as exception:
        print(exception)


return 'Something went wrong, try again', 400
    else:
        return 'Details Added.', 200
        # return redirect(url_for('login_check'), code = 307)
return 'add req rcvd' if __name__ == '__main__':
    app.static_folder = 'static'
app.run()
```

db_handlers.py import ibm_db_dbi as db print('[CONNECTING]')

```
print('[CONNECTED]') def create_account(fullName, username, email,
password, mobileNumber):
    cursor = conn.cursor()
    SQL = f"INSERT INTO users VALUES(?, ?, ?, ?, ?, NULL)"
cursor.execute(SQL, [fullName, username, email, password, mobileNumber])
print('[INSERTED]') cursor.close()     conn.commit() def
execute_generic_query(sql, *params):
    # return type is: [(1, 2, 3), (4, 5, 6), (7, 8, 9)]
cursor = conn.cursor()     if params:
        cursor.execute(sql, params) else:
        cursor.execute(sql)


conn.commit()
    try:
        rows = cursor.fetchall()
except db.ProgrammingError:
        rows = None
else:         if not
rows:
            print('Nothing returned by the query')
        else:
            for row in rows:
print(row)     finally:
        cursor.close()       return rows def
check_username_existence(value):
    SQL = f'SELECT password FROM users WHERE username LIKE \'{value}\''
result = execute_generic_query(SQL)
    print(result)    if not result:
return False return True def
check_email_existence(value):
    SQL = f'SELECT password FROM users WHERE email LIKE \'{value}\''
result = execute_generic_query(SQL)
```

```
    print(result) if
    not result:
    return False
    return True
    def
    fetch_passwor
    d(username):
    SQL = f'SELECT password FROM users WHERE username like \'{username}\''
password = execute_generic_query(SQL)     print(password)     if not password:
return password     return password[0][0] def fetch_table_contents(username):
    SQL = f'SELECT contents FROM users WHERE username = \'{username}\''
tableContents = execute_generic_query(SQL)     print('table contents are',
tableContents)     return tableContents def add_details_to_db(username,
details):
    SQL = f'UPDATE users SET contents = \'{details}\' WHERE username = \'{username}\''
execute_generic_query(SQL)
'''
def view_all_entries():
   execute_generic_query('SELECT * FROM users') def
view_names():
   execute_generic_query('SELECT name FROM users') def
view_usernames():
    execute_generic_query('SELECT username FROM users') def
view_passwords():
    execute_generic_query('SELECT password FROM users') def
view_emails():
    execute_generic_query('SELECT email FROM users')


def view_mobiles():

    execute_generic_query('SELECT mobile FROM users') def

view_table_layout():

    # You cannot describe a table for some reason.

    return

    # Useless junk below

cursor = conn.cursor()

    cursor.execute("SELECT * FROM SYSIBM.COLUMNS WHERE TABLE_NAME = 'users'''")

    rows = cursor.fetchall()

for row in rows:

print(row)     cursor.close()

'''

def main():

    # view_all_entries()

    # view_table_layout()

    '''

    functions = [

view_names,

view_usernames,

view_emails,

view_passwords,

view_mobiles

        ]

    for function in functions:
        print('*'*15)
function()

    '''
    print('[EXIT-MAIN]') if
__name__ == '__main__':
    main()
```

dashboard.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <style>         #nav {
width: 100vw;
border: 1px solid red;
text-align: center;
        }
        #inventory {            width:
100%;            border: 1px solid
black;
        }
        #inventory * {
border: 1px solid black;
```

```
        }

        #add-container {

display: none;

        }


 #update-container {

            display: none;

        }

    </style>

    <title>Dashboard</title>

</head>

<body>

    <nav id="nav">

        <span>{{ greeting }}</span>

    </nav>

    <div id="table-container">

        {% if tableContents %}

            <table id="inventory">

       <tr>

                    <th>Commodity ID</th>

                    <th>Commodity</th>

                    <th>Original Quantity</th>


 <th>Current Quantity</th>

                    <th>Rate</th>

                    <th>Modification Date</th>
```

```
                </tr>
                {% for index in range(tableContents[(tableContents.keys()|list)[0]]|length) %}
                    <tr onclick="update_qty()">
                        {% for attribute in tableContents.keys() %}
                            <td>{{ tableContents[attribute][index] }}</td>
                        {% endfor %}
                    </tr>
                {% endfor %}
            </table>
            {% else %}
                <p>No content</p>
        {% endif %}
    </div>
    <div id="modification-container">
        <button class="modification-btn" onclick="reveal_addition_wizard()">Add</button>
        <div id="add-container">
            <input type="text" placeholder="Commodity ID">
            <input type="text" placeholder="Commodity name">
            <input type="text" placeholder="Quantity">
            <input type="text" placeholder="Rate">
            <button onclick="verify_and_insert_to_table()" class="modification-btn">Save</button>
</div>
        <button class="modification-btn">Remove</button>
        <button lass="modification-btn" onclick="reveal_updation_wizard()">Update</button>
        <div id="update-container">
            <input type="text" placeholder="Commodity ID">
            <input type="text" placeholder="New Quantity">
            <button onclick="update_qty()" class="modification-btn">Save</button>
        </div>
    </div>     <script>        const cidFormat =
/^[a-zA-Z0-9_]+$/;         const cnameFormat =
/^[a-zA-Z0-9_ ]+$/;         const qtyFormat =
/^\d+$/;


     const rateFormat = /^\d+.?\d*$/;         const addBtn =
document.getElementsByClassName('modification-btn')[0];         const saveBtn =
document.getElementsByClassName('modification-btn')[1];         const removeBtn =
document.getElementsByClassName('modification-btn')[2];         const updateBtn =
document.getElementsByClassName('modification-btn')[3];         const table =
document.getElementById('inventory');              function reveal_updation_wizard()
{             document.getElementById('update-container').style.display = 'block';
        }
        function hide_updation_wizard() {             document.getElementById('update-
container').style.display = 'none';
        }
        function reveal_addition_wizard() {             document.getElementById('add-
container').style.display = 'block';
        }
        function hide_addition_wizard() {             document.getElementById('add-
container').style.display = 'none';
        }
        function verify_and_insert_to_table() {            let allGood = true;            let errString
= 'Enter ';            let cid = document.getElementById('add-
container').children[0].value.trim();            let cname = document.getElementById('add-
container').children[1].value.trim();            let qty = document.getElementById('add-
container').children[2].value.trim();            let rate = document.getElementById('add-
container').children[3].value.trim();            if(!cidFormat.test(cid)) { errString +=
'Commodity ID, '; allGood = false; }            if(!cnameFormat.test(cname)) { errString +=
'Commodity name, '; allGood = false; }  if(!qtyFormat.test(qty)) { errString += 'Quantity, ';
allGood = false; }            if(!rateFormat.test(rate)) { errString += 'Rate '; allGood = false; }
errString += 'correctly.';            if(!allGood) alert(errString);            else {                let
dateTime = new Date().toLocaleString();                let array = [cid, cname, qty, qty, rate,
dateTime];                addBtn.disabled = true;                removeBtn.disabled = true;
saveBtn.disabled = true;                updateBtn.disabled = true;                xhr = new
XMLHttpRequest();                xhr.open("POST", "/add-commodity", true);
xhr.onreadystatechange = () => {                    if (xhr.readyState ===
XMLHttpRequest.DONE) {                        alert(xhr.responseText);
addBtn.disabled = false;                        removeBtn.disabled = false;
saveBtn.disabled = false;                        updateBtn.disabled = false;
if(xhr.status === 200) {                        let row = table.insertRow(-1);
for(let cell = 0; cell < 6; ++cell) {                                let newCell = row.insertCell(cell);
newCell.innerHTML = array[cell];
                            }
                        }
                        hide_addition_wizard();
                    }
                }
```

```
            xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');

xhr.send(`cid=${cid}&cname=${cname}&oqty=${qty}&cqty=${qty}&rate=${rate}&date=${dateTi
me}`);
        }
    }
    function update_qty() {            let allGood = true;            let targetCommodity =
document.getElementById('update-container').children[0].value.trim();            let newQty =
document.getElementById('update-container').children[1].value.trim();
if(!cidFormat.test(targetCommodity) || !qtyFormat.test(newQty)) allGood = false;            if(!allGood)
alert('Detail(s) entered in wrong format.');
        else {
                }
            }
  </script>
</body>
</html>
```

login.html:

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
  <meta charset="utf-8">
  <title>Login and Registration</title>

  <link rel="stylesheet" href="{{ url_for('static', filename='login_style.css') }}">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
</head>

<body>

  <div class="wrapper">

    <div class="title-text">

      <div class="title login">

        Login Form

      </div>

      <div class="title signup">

        Signup Form

      </div>

    </div>

    <div class="form-container">

      <div class="slide-controls">

        <input type="radio" name="slide" id="login" checked>

 <input type="radio" name="slide" id="signup">

        <label for="login" class="slide login">Login</label>

        <label for="signup" class="slide signup">Signup</label>

        <div class="slider-tab"></div>

      </div>

      <div class="form-inner">

        <form action="/login" class="login" method="post" name="login-form">

          <div class="field">

            <input type="text" placeholder="User Name" required name="uname">

          </div>

          <div class="field">
```

```
<input type="password" placeholder="Password" required name="password">
          </div>
          <div class="pass-link">
            <a href="#">Forgot password?</a>
          </div>
          <div style="color: red; text-align: center;">
            {{ err_msg }}
          </div>
          <div class="field btn">
            <div class="btn-layer"></div>
            <input type="submit" value="Login">
          </div>
          <div class="signup-link">
            Not a member? <a href="">Signup now</a>
          </div>
        </form>
        <form action="/signup" class="signup" method="post" name="signup-form">
          <div class="field">
            <input type="text" placeholder="Full Name" required
name="fname" onfocusout="check_name()">
          </div>
          <div class="field">
            <input type="text" placeholder="Username" required
name="uname" onfocusout="check_username()">
          </div>
          <div class="field">
```

```
                <input type="text" placeholder="Email Address" required name="email"
onfocusout="check_email()">

              </div>

              <div class="field">

                <input type="password" placeholder="Password" required name="password"

onfocusout="check_password()">

              </div>

              <div class="field">

                <input type="password" placeholder="Confirm Password" required name="password-cnf"

onfocusout="check_password()">

              </div>

              <div class="field">

                <input type="tel" placeholder="Mobile Number" required name="mobile"

onfocusout="check_mobile()">

              </div>

              <div class="field btn">

                <div class="btn-layer"></div>

                <input type="submit" value="Signup" disabled name="signup-btn" style="cursor:
notallowed;"

                  onclick="check_all()">

              </div>

            </form>

          </div>

        </div>

   </div>    <script>       var flag1 = false, flag2 = false, flag3 = false, flag4

= false, flag5 = false;       const loginText = document.querySelector(".title-

text .login");
```

```
  const loginForm = document.querySelector("form.login");       const loginBtn
= document.querySelector("label.login");       const signupBtn =
document.querySelector("label.signup");       const signupLink =
document.querySelector("form .signup-link a");       const nameField =
document.forms['signup-form']['fname'];       const usernameField =
document.forms['signup-form']['uname'];       const emailField =
document.forms['signup-form']['email'];       const passwordField =
document.forms['signup-form']['password'];       const password_cnfField =
document.forms['signup-form']['password-cnf'];       const mobileField =
document.forms['signup-form']['mobile'];       const signupBtnActual =
document.forms['signup-form']['signup-btn'];       signupBtn.onclick = (() => {
loginForm.style.marginLeft = "-50%";          loginText.style.marginLeft = "-
50%";
      });
      loginBtn.onclick = (() => {
loginForm.style.marginLeft = "0%";
loginText.style.marginLeft = "0%";
      });
      signupLink.onclick = (() => {
signupBtn.click();          return
false;
      });
      function check_all() {
console.log('check_all');
check_name();          check_username();
```

```
check_password();        check_email();

check_mobile();        if (check_flags())

{            enable_signup_btn();

      }

      else disable_signup_btn();

    }

    function check_flags() {        return flag1 && flag2

&& flag3 && flag4 && flag5;

    }

    function enable_signup_btn() {

signupBtnActual.disabled = false;

signupBtnActual.style.cursor = 'pointer';

    }

    function disable_signup_btn() {

signupBtnActual.disabled = true;

signupBtnActual.style.cursor = 'not-allowed';

    }

    function check_username() {        if

(usernameField.value === '') return;        var xhr

= new XMLHttpRequest();

xhr.open("POST", "/check-username", true);

let usernameformat = /^[a-z][a-z0-9_\.]+$/;

let username = usernameField.value.trim();

let valid = usernameformat.test(username);

      if (valid) {
```

```
 xhr.onreadystatechange = () => {                if (xhr.readyState ===
XMLHttpRequest.DONE && xhr.status === 403) {
usernameField.value = 'Username Already Exists!';
usernameField.style.backgroundColor = '#FFC6B9';
                    valid = false;
               }
            }
            xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
xhr.send("username=" + username);
        }          if
(!valid)
            usernameField.style.backgroundColor = '#FFC6B9';
        else
            usernameField.style.backgroundColor = '#C5FFB9';        flag2 = valid;        if
(check_flags()) enable_signup_btn();          else disable_signup_btn();  }       function
check_email() {              if (emailField.value === '') return;             var xhr = new
XMLHttpRequest();                        let   mailformat   =   /^\w+([\.-]?\w+)*@\w+([\.-
]?\w+)*(\.\w{2,3})+$/;            let email = emailField.value.trim();           let valid =
mailformat.test(email);           if (valid) {              xhr.open("POST", "/check-email",
true);               xhr.onreadystatechange = () => {                  if (xhr.readyState ===
XMLHttpRequest.DONE  &&  xhr.status  ===  403)  {  emailField.value  =  'Email
Already Exists!';

                    valid = false;
emailField.style.backgroundColor = '#FFC6B9';
               }
            }
            xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');
xhr.send("email=" + email);
         }          if
(!valid)
            emailField.style.backgroundColor = '#FFC6B9';
         else
            emailField.style.backgroundColor = '#C5FFB9';
flag3 = valid;          if (check_flags())
enable_signup_btn();          else disable_signup_btn();
      }
 function check_name() {          let name
= nameField.value.trim();          if (name
=== '') return;
         let       nameformat       =       /^(?:((([^0-9_!¡?÷?¿/\\+=@#$%ˆ&*(){}|~<>;:[\]"’,\-.\s])){1,}(["’,\-
\.]){0,1}){2,}(([^0-9_!¡?÷?¿/\\+=@#$%ˆ&*(){}|~<>;:[\]"’,\-.          ]))*(([          ]+){0,1}(((([^0-
9_!¡?÷?¿/\\+=@#$%ˆ&*(){}|~<>;:[\]"’,\-\.\s])){1,})(["’\-,\.]){0,1}){2,}((([^0-
9_!¡?÷?¿/\\+=@#$%ˆ&*(){}|~<>;:[\]"’,\-\.\s])){2,})?)*)$/;
let valid = nameformat.test(name);
         if (valid)
            nameField.style.backgroundColor = '#C5FFB9';
         else
            nameField.style.backgroundColor = '#FFC6B9';

   flag1 = valid;          if (check_flags())
enable_signup_btn();          else
disable_signup_btn();
```

```
    }

    function check_mobile() {          let
mobile = mobileField.value.trim();          if
(mobile === '') return;          let
mobileformat = /^\d{10}$/;          let valid
= mobileformat.test(mobile);

      if (valid)

        mobileField.style.backgroundColor = '#C5FFB9';

      else

        mobileField.style.backgroundColor = '#FFC6B9';

flag5 = valid;          if (check_flags())

enable_signup_btn();          else disable_signup_btn();

    }

    function check_password() {          if (passwordField.value === '' &&

password_cnfField.value === '') return;          let valid = (passwordField.value

=== password_cnfField.value);

      if (!valid) {

        password_cnfField.style.backgroundColor = '#FFC6B9';

passwordField.style.backgroundColor = '#FFC6B9';

      }

else {

        password_cnfField.style.backgroundColor = '#C5FFB9';

passwordField.style.backgroundColor = '#C5FFB9';
```

```
     }          flag4 = valid;          if
(check_flags()) enable_signup_btn();
else disable_signup_btn();
      }
   </script>
</body>
</html>
```

login_style.css:

```
@import url('https://fonts.googleapis.com/css?family=Poppins:400,500,600,700&display=swap');
*{   margin: 0;   padding: 0;   box-
sizing: border-box;   font-family:
'Poppins', sans-serif;
} html,body{
  display: grid;   height: 100%;   width: 100%;   place-
items: center;   background: -webkit-linear-gradient(top,
#eacda3, #fff);
}
::selection{
background: #d6ae7b;
  color: #fff;
}

.wrapper{   overflow: hidden;   max-width:
390px;   background: #fff;   padding: 30px;
```

```
border-radius: 5px;   box-shadow: 0px 15px
20px rgba(0,0,0,0.1);
}
.wrapper .title-text{   display: flex;   width: 200%; }
.wrapper .title{   width: 50%;   font-size: 35px;   font-
weight: 600;   text-align: center;   transition: all 0.6s
cubic-bezier(0.68,-0.55,0.265,1.55);
}
.wrapper .slide-controls{
position: relative;   display: flex;
height: 50px;   width: 100%;
overflow: hidden;   margin:
30px 0 10px 0; justify-content:
space-between;   border: 1px
solid lightgrey;   border-radius:
5px;
}
.slide-controls .slide{
height: 100%;   width:
100%;   color: #fff;
font-size: 18px;   font-
weight: 500;   text-align:
center;   line-height:
48px;   cursor: pointer;
z-index: 1;   transition:
all 0.6s ease;
```

```
}

.slide-controls label.signup{

color: #000;

}

.slide-controls .slider-tab{

position: absolute;

height: 100%;   width:

50%;

  left: 0;   z-index: 0;   border-radius: 5px;   background: -

webkit-linear-gradient(left, #eacda3, #d6ae7b); transition: all

0.6s cubic-bezier(0.68,-0.55,0.265,1.55);


} input[type="radio"]{

display: none;

} #signup:checked ~ .slider-

tab{   left: 50%;

}

#signup:checked ~ label.signup{

  color: #fff;   cursor: default;

user-select: none; }

#signup:checked ~ label.login{

color: #000;

} #login:checked ~

label.signup{   color: #000;

}
```

```
#login:checked ~ label.login{

cursor: default;   user-select:

none; } .wrapper .form-

container{   width: 100%;

overflow: hidden; }

.form-container .form-inner{ display:

  flex;



  width: 200%;

}

.form-container .form-inner form{   width: 50%;

transition: all 0.6s cubic-bezier(0.68,-0.55,0.265,1.55);

}

.form-inner form .field{

height: 50px;   width:

100%;   margin-top:

20px;

}

.form-inner form .field input{

height: 100%;   width: 100%;

outline: none;   padding-left:

15px;   border-radius: 5px;

border: 1px solid lightgrey;

border-bottom-width: 2px;
```

```
font-size: 17px;   transition:
all 0.3s ease;
}
.form-inner form .field input:focus{   border-
color: #eacda3;
  /* box-shadow: inset 0 0 3px #fb6aae; */}
.form-inner form .field input::placeholder{
  color: #999;   transition: all 0.3s
ease; } form .field
input:focus::placeholder{   color:
#b3b3b3; }
.form-inner form .pass-link{   margin-
top: 5px;
}
.form-inner form .signup-link{
text-align: center;   margin-top:
30px;
}
.form-inner form .pass-link a,
.form-inner form .signup-link a{
color: #eacda3;   text-decoration:
none;
}
.form-inner form .pass-link a:hover,
.form-inner form .signup-link a:hover{
text-decoration: underline;} form .btn{
```

```
height: 50px;   width: 100%;   border-
radius: 5px;   position: relative;
overflow: hidden;

} form .btn .btn-layer{   height: 100%;   width: 300%;   position: absolute;   left: -
100%;   background: -webkit-linear-gradient(right, #d6ae7b, #eacda3, #d6ae7b,
#eacda3);   border-radius: 5px;   transition: all 0.4s ease;;} form .btn:hover .btn-
layer{
  left: 0; } form .btn
input[type="submit"]
{   height: 100%;
width: 100%;   z-
index: 1;   position:
relative;
background: none;
border: none;
color: #fff;
padding-left: 0;
border-radius: 5px;
font-size: 20px;
font-weight: 500;
cursor: pointer;
}
```

mail.py:

```
from sendgrid import SendGridAPIClient from
sendgrid.helpers.mail import Mail API_KEY = 'Your
API key here' def sendgrid_email(subject, content,
receiver):     message =
Mail(from_email='from@example.com',
                  to_emails=receiver,
subject=subject,
plain_text_content=content)
    sg = SendGridAPIClient(API_KEY)
response = sg.send(message)
print(response.status_code, response.body) def
main():
    SUBJECT = ''
    CONTENT = ''     RECEIVER = ''
sendgrid_email(SUBJECT, CONTENT, RECEIVER) if
__name__ == '__main__':
    main()
```

requirements.txt:

```
click==8.1.3
colorama==0.4.6
Flask==2.2.2 ibm-db==3.1.3
itsdangerous==2.1.2
Jinja2==3.1.2
MarkupSafe==2.1.1
PTable==0.9.2 python-http-
client==3.3.7
sendgrid==6.9.7 starkbank-
ecdsa==2.2.0
```

```
Werkzeug==2.2.2
```

login_script.js:

```
var flag1 = false, flag2 = false, flag3 = false, flag4 = false, flag5 = false;

const loginText = document.querySelector(".title-text .login"); const

loginForm = document.querySelector("form.login"); const loginBtn =

document.querySelector("label.login"); const signupBtn =

document.querySelector("label.signup"); const signupLink =

document.querySelector("form .signup-link a"); const nameField =

document.forms['signup-form']['fname']; const usernameField =

document.forms['signup-form']['uname']; const emailField =

document.forms['signup-form']['email']; const passwordField =

document.forms['signup-form']['password']; const password_cnfField =

document.forms['signup-form']['password-cnf']; const mobileField =

document.forms['signup-form']['mobile']; const signupBtnActual =

document.forms['signup-form']['signup-btn']; signupBtn.onclick = (() => {

loginForm.style.marginLeft = "-50%";     loginText.style.marginLeft = "-

50%";

}); loginBtn.onclick = (() => {

loginForm.style.marginLeft = "0%";

loginText.style.marginLeft = "0%";


}); signupLink.onclick = (()

=> {     signupBtn.click();

return false; }); function

check_all() {
```

```
console.log('check_all');
check_name();
check_username();
check_password();
check_email();
check_mobile();     if
(check_flags()) {
enable_signup_btn();
    }
    else disable_signup_btn();
} function check_flags() {     return flag1 && flag2 &&
flag3 && flag4 && flag5;
} function enable_signup_btn() {
signupBtnActual.disabled = false;
signupBtnActual.style.cursor = 'pointer';
} function disable_signup_btn() {
signupBtnActual.disabled = true;
signupBtnActual.style.cursor =
'not-allowed';

} function check_username() {     if
(usernameField.value === '') return;     var xhr
= new XMLHttpRequest();
xhr.open("POST", "/check-username", true);
let usernameformat = /^[a-z][a-z0-9_\.]+$/;
let username = usernameField.value.trim();

let valid = usernameformat.test(username);

    if (valid) {     xhr.onreadystatechange = () => {         if (xhr.readyState ===

XMLHttpRequest.DONE && xhr.status === 403) {

usernameField.value = 'Username Already Exists!';

usernameField.style.backgroundColor = '#FFC6B9';

                valid = false;

         }

    }

    xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');

xhr.send("username=" + username);

    }     if

(!valid)

    usernameField.style.backgroundColor = '#FFC6B9';

    else

    usernameField.style.backgroundColor = '#C5FFB9';

flag2 = valid;     if (check_flags())

enable_signup_btn(); else disable_signup_btn();


} function check_email() {     if (emailField.value === '') return;     var xhr =

new XMLHttpRequest();     let mailformat = /^\w+([\.-]?\w+)*@\w+([\.-

]?\w+)*(\.\w{2,3})+$/;     let email = emailField.value.trim();     let valid =

mailformat.test(email);     if (valid) {     xhr.open("POST", "/check-email",

true);     xhr.onreadystatechange = () => {         if (xhr.readyState ===

XMLHttpRequest.DONE && xhr.status === 403) {                 emailField.value
= 'Email Already Exists!';          valid = false;

emailField.style.backgroundColor = '#FFC6B9';

        }

    }

    xhr.setRequestHeader('Content-type', 'application/x-www-form-urlencoded');

xhr.send("email=" + email);

    }     if

(!valid)

    emailField.style.backgroundColor = '#FFC6B9';

    else

    emailField.style.backgroundColor = '#C5FFB9';

flag3 = valid;     if (check_flags())

enable_signup_btn();     else disable_signup_btn();

}

function check_name() {     let name

= nameField.value.trim();     if (name

=== '') return;

    let    nameformat    =    /^(?:((([^0-9_!¡?÷?¿/\\+=@#$%ˆ&*(){}|~<>;:[\]'’,\-.\s])){1,}(['’,\-
\.]){0,1}){2,}(([^0-9_!¡?÷?¿/\\+=@#$%ˆ&*(){}|~<>;:[\]'’,\-.    ]))*(([    ]+){0,1}(((([^0-
9_!¡?÷?¿/\\+=@#$%ˆ&*(){}|~<>;:[\]'’,\-\.\s])){1,})(['’\-,\.]){0,1}){2,}((([^0-
9_!¡?÷?¿/\\+=@#$%ˆ&*(){}|~<>;:[\]'’,\-\.\s])){2,})?)*)$/;

let valid = nameformat.test(name);

    if (valid)

    nameField.style.backgroundColor = '#C5FFB9';

    else
    nameField.style.backgroundColor = '#FFC6B9';
flag1 = valid;     if (check_flags())
enable_signup_btn();     else disable_signup_btn();
} function check_mobile() {     let
mobile = mobileField.value.trim();     if
(mobile === '') return;     let
mobileformat = /^\d{10}$/;     let valid
= mobileformat.test(mobile);     if
(valid)
    mobileField.style.backgroundColor = '#C5FFB9';
    else
    mobileField.style.backgroundColor = '#FFC6B9';
flag5 = valid;     if (check_flags())
enable_signup_btn(); else disable_signup_btn();

} function check_password() {     if (passwordField.value === '' &&
password_cnfField.value === '') return;     let valid = (passwordField.value
=== password_cnfField.value);
    if (!valid) {
    password_cnfField.style.backgroundColor = '#FFC6B9';
passwordField.style.backgroundColor = '#FFC6B9';
    }
else {
    password_cnfField.style.backgroundColor = '#C5FFB9';
passwordField.style.backgroundColor = '#C5FFB9';
```

```
    }     flag4 = valid;     if (check_flags())
enable_signup_btn();     else
disable_signup_btn();
}
```

dashboard_style.css:

```
@import url('https://fonts.googleapis.com/css2?family=Great+Vibes&display=swap');  /* great vibes
*/
@import url('https://fonts.googleapis.com/css2?family=Josefin+Slab&display=swap'); /* josefin */
@import url('https://fonts.googleapis.com/css?family=Poppins:400,500,600,700&display=swap');
* {     font-family: 'Poppins', sans-
serif;
} body {
margin: 0;
}
#nav {
    display: flex;     width: 100vw;     height:
10vh;     background-color: #4568dc;
align-items: center;     justify-content: left;
border-radius: 10px;     margin-top: 5px;
box-shadow: 0px 0px 20px 0px #888888;
}
#nav div {     /* border: 1px
solid red; */     width: 100%;
}
```

```
#nav div span {     font-
size: larger;     font-
weight: bold;     color:
white; } #nav div a {
text-decoration: none;
width: 100%;     text-
align: right; }
#nav div a:visited { color:
    white;

}
#table-container {
display: flex;     align-
items: center;     justify-
content: center;     margin-
top: 15px;
}
#inventory, #inventory * {
border: 1px solid #4568dc;
border-radius: 2px;
background-color: #4568dc;
font-weight: normal;
} #inventory {
border: none;
width: 100%;
border-radius: 5px;
```

```
    /* background-color: rgb(224, 224, 255); */     background-
color: white;
}
#inventory th {     color:
white;     border-radius:
5px; } #inventory td {
background-color: white;
border-radius: 5px;
}
.hidden {
display: none; }
.visible {
display: block;
}
#modification-container {
width: 100vw;     position:
absolute;     bottom: 0;
display: flex;     flex-direction:
row;     justify-content: space-
around;
    /* border: 1px solid red; */
} .modification-btn {
border: none;     background-
color: #4568dc;     color:
white;     border-radius: 5px;
```

```
transition: all 0.5s;     font-
size: large;
    /*box-shadow: 0px 0px 50px 10px #888888;*/
}
.modification-btn:hover {     background-
color: #6699cc;
}
.outer-container {
margin: 10px;     padding:
5px;
    /* border: 1px solid rgb(243, 212, 212); */
border-radius: 5px;     background-color:
#6699cc;     box-shadow: 0px 0px 20px 0px
#888888;
}
```

Dashboard_script.js:

```
const cidFormat = /^[a-zA-Z0-9_]+$/; const cnameFormat = /^[a-zA-Z0-
9_ ]+$/; const qtyFormat = /^\d+$/; const rateFormat = /^\d+.?\d*$/; const
btns = document.getElementsByTagName('button'); const table =
document.getElementById('inventory'); function hide(action, element_id
= '') {     action = action.trim();     element_id = element_id.trim();     const
idArray = ['add-container', 'update-container', 'remove-container'];
if(action === 'SELF-HIDE')
document.getElementById(element_id).style.display = 'none';     else
if(action === 'SELF-REVEAL')
```

```
document.getElementById(element_id).style.display = 'block';     else

if(action === 'OTHERS-HIDE')

        for(let id of idArray)

if(id != element_id)

document.getElementById(id)

.style.display = 'none';     else

if(action === 'OTHERS-

REVEAL')

        for(let id of idArray)                 if(id != element_id)

document.getElementById(id).style.display = 'block';

else if(action === 'ALL-HIDE')         for(let id of idArray)

document.getElementById(id).style.display = 'none';     else

if(action === 'ALL-REVEAL')

        for(let id of idArray)

document.getElementById(id).style.display = 'block';

} function deactivate_btns()

{     for (const btn of btns)

btn.disabled = true;

} function activate_btns()

{     for (const btn of

btns)         btn.disabled =

false;

} function verify_and_insert_to_table() {     let allGood = true;     let errString =

'Enter ';     let cid = document.getElementById('add-

container').children[0].value.trim();     let cname =

document.getElementById('add-container').children[1].value.trim();     let qty =
document.getElementById('add-container').children[2].value.trim();     let rate =
document.getElementById('add-container').children[3].value.trim();
if(!cidFormat.test(cid)) { errString += 'Commodity ID, '; allGood = false; }
if(!cnameFormat.test(cname)) { errString += 'Commodity name, '; allGood =
false; }     if(!qtyFormat.test(qty)) { errString += 'Quantity, '; allGood = false; }
if(!rateFormat.test(rate)) { errString += 'Rate '; allGood = false; }     errString +=
'correctly.';     if(!allGood) alert(errString);     else {         let dateTime = new
Date().toLocaleString();         let array = [cid, cname, qty, qty, rate, dateTime, '-'];
deactivate_btns();         xhr = new XMLHttpRequest();         xhr.open("POST",
"/add-commodity", false);                 xhr.setRequestHeader('Content-type',
'application/x-www-form-urlencoded');

hr.send(`cid=${cid}&cname=${cname}&oqty=${qty}&cqty=${qty}&rate=${rate}&date=${dateTim
e}&date_mdf=-`);
        if(xhr.status === 200) {
location.reload();
        }         else {
alert(xhr.responseText);
hide('ALL-HIDE');
console.log('heu');
activate_btns();
        }
    }
}
function update_qty() {     let allGood = true;     let targetCommodity =
document.getElementById('update-container').children[0].value.trim();     let newQty =
document.getElementById('update-container').children[1].value.trim();
```

```
if(!cidFormat.test(targetCommodity) || !qtyFormat.test(newQty)) allGood = false;     if(!allGood)
alert('Detail(s) entered in wrong format.');     else {         let dateTime = new
Date().toLocaleString();         deactivate_btns();         xhr = new XMLHttpRequest();
xhr.open("POST", "/update-commodity", false);             xhr.setRequestHeader('Content-type',
'application/x-www-form-urlencoded');
xhr.send(`cid=${targetCommodity}&qty=${newQty}&date=${dateTime}`);
        if(xhr.status === 200) {
location.reload();
        }         else {
alert(xhr.responseText);
hide('ALL-HIDE');
activate_btns();
        }
    }
    }
unction remove_item() {     let targetCommodity = document.getElementById('remove-
container').children[0].value.trim();     let allGood = true;     if(!cidFormat.test(targetCommodity))
allGood = false;

    if(!allGood) alert('Detail(s) entered in wrong format.');     else {
deactivate_btns();         xhr = new XMLHttpRequest();         xhr.open("POST",
"/remove-commodity", false);                 xhr.setRequestHeader('Content-type',
'application/x-www-form-urlencoded');         xhr.send(`cid=${targetCommodity}`);
if(xhr.status === 200) {             location.reload();
        }         else {
alert(xhr.responseText);
hide('ALL-HIDE');

activate_btns();

        }

    }

}
```

Github : https://github.com/IBM-EPBL/IBM-Project-51795-1660984505

Demo link : https://drive.google.com/file/d/1k_Sk_1ky9r0ykKX01R-1l62rNRV25rHV/view