

## Inventory Managment System for Retailers

TEAM ID : PNT2022TMID03188

**Team Leader :** SRINITHI S

**Team member :** SIDDARTH S

**Team member :** SIDDARTH G

**Team member :** SIBICHAKRAVARTHY

## 1. ABSTRACT

Inventory Management System is important to ensure quality control in businesses that handle transactions revolving around consumer goods. Without proper inventory control, a large retail store may run out of stock on an important item. A good Inventory Management System will alert the retailer when it is time to reorder. Inventory Management System is also an important means of automatically tracking large shipments. For example, if a business orders ten pairs of socks for retail resale, but only receives nine pairs, this will be obvious upon inspecting the contents of the package, and error is not likely. On the other hand, say a wholesaler orders 100,000 pairs of socks and 10,000 are missing. Manually counting each pair of socks is likely to result in error. An automated Inventory Management System helps to minimize the risk of error. In retail stores, an Inventory Management System also helps track theft of retail merchandise, providing valuable information about store profits and the need for theft-prevention systems. Automated Inventory Management System work by scanning a barcode either on the item. A barcode scanner is used to read the barcode, and the information encoded by the barcode is read by the machine. This information is then tracked by a central computer system. For example, a purchase order may contain a list of items to be pulled for packing and shipping. The Inventory Management System can serve a variety of functions in this case. It can help a worker locate the items on the order list in the warehouse, it can encode shipping information like tracking numbers and delivery addresses, and it can remove these purchased items from the inventory tally to keep an accurate count of in-stock items. All of this data works in tandem to provide businesses with real-time inventory tracking information.

Inventory Management System make it simple to locate and analyze inventory information in real-time with a simple database search.

## 2. INTRODUCTION

The Inventory Management System is no different from any other information system in that there are factors that make it successful. The basis for this report is premised on the five components as outlined in our book. These five critical components are hardware, software, data, procedures and people. As these factors are discussed throughout the next several sections it becomes evident that they are contingent upon one another, and frankly will not function efficiently without the other. The Inventory Management System is no different from any other information system in that there are factors that make it successful. The basis for this report is premised on the five components as outlined in our book. These five critical

components are hardware, software, data, procedures and people. As these factors are discussed throughout the next several sections it becomes evident that they are contingent upon one another, and frankly will not function efficiently without the other.

### 3. LITERATURE SURVEY

Dave Piasecki .[1] (2001) He focused on various model of inventory to calculating optimum purchase quantity which used the EOQ method. He points out that many companies are not using EOQ model because of poor results resulted from inaccurate data input. He says that EOQ is an accounting formula which determines point at which combination of ordering costs and stock inventory costs are the least. He highlights that EOQ method would not conflict with the JIT approach. He further elaborates the EOQ model formula that includes parameters like yearly usage on unit, order cost and carrying cost. Finally, he proposes several steps to follow in implementing the EOQ model. Now this literature limitation is as it does not elaborate further association among EOQ and JIT. It does not associate stock turns with EOQ so fails for mention profit gain with associated stock is calculated. Sambasiva Rao. K [2] (2002) According his investigation on Materials Managing in Public Sector Ship Building Industry evaluates. Output of materials managing and identifies some problems faced by materials managing in the heavy engineering industry. This investigation method involves the 68 documentary evidence and survey of expert opinion. He evaluates the existing purchase systems and lead time involved on procurement of stock item and advised the long

lead time shall be reduced. His research points at additional stock in terms on months production cost in all the engineering units. He also highlights some of the problems in the area on materials managing such as delay in customer part on supplying own stock item, existence and disposal of surplus and non-moving items, excessive lead times and excessive dependence on imports. He claims that administrative and procurement lead times for organization are on the higher side according to peculiar nature of industry. He suggests liberalized purchase procedures, increased capital powers to the personnel, Opening up of liaison offices in various countries to reduce the lead time. Gaur, Fisher and Raman [3] (2005) In their study examined firm-level inventory behavior among retailing companies. They took a sample on 311 public-listed retail firms for years 1987–2000 for investigate relationship on stock turnover about gross margin, capital intensity , sales surprise. All observed that stock aggregate turnover for retailing company was positively related to capital intensity with sales surprise while inversely related gross margins. S. Singh [4] (2006) Analysed stock control exercises on single fertilizer company named IFFCO. He statistically examined stock level according consumption, sales as well as other variables along growth on these variables with inventory patterns. He concluded increments in components of stocks lead to increment in the proportion on stock in current assets. The special attention was made in stores with spares for calculate excess purchases resulting Pradeep singh (2008) In his study made an attempt to investigate stock with working capital managing Indian Farmers Fertilizer Cooperative Limited (IFFCO) / National Fertilizer Limited (NFL). He concluded that overall position of the working fund of IFFCO / NFL is satisfactory. But there arises need for improvement in stocking as situation of IFFCO. Although stock were not

properly utilized as well as maintained by IFFCO during investigation period. Also managing organization of NFL surely try to properly utilize stock with try to care stock according to requirements. So that liquidity will not interrupt. Capkun, Hameri and Weiss [5] (2009) Statistically analysed the association among stock levels with fund situation in manufacturing companies using capital information on large sample on USbased production units over a 26-year period, during, 1980 to 2005. According to them a significant relationship existed between inventory performance along with the performance of its components and profitability. Gaur and Bhattacharya [6] (2011) Attempted to study the linkage between the performance of the components of inventory such as raw material, work in progress and finished goods and financial performance of Indian manufacturing firms. The study revealed that finished goods inventory as inversely associated with business performance while raw material inventory and work in progress did not have much effect on same. They emphasised that instead of focusing on total inventory, an attempt should be made to concentrate on individual components of inventory so as to adequately manage the same. They concluded that managers not paying heed to inventory performance may become weak in combating competitors. Eneje et al [7] (2012) He researched the changes of raw stock inventory management system with margin of beer company in Nigeria during data from 1989 to 2008 which had gathered for analysis from the annual reports of the sampled brewery firms. Measures of profitability were examined and related to proxies for raw materials inventory management by brewers. The Ordinary Least Squares (OLS) stated in the form of a multiple regression model was applied in the analysis. Research analysed that local variable raw stock inventory managing system design such a way to capturing

changes of efficient management of raw stock inventory on behalf of company in terms of their margin is significantly strong and positive and influences the profitability of the brewery firms in Nigeria. They concluded that efficient management of raw material inventory is a major factor to be contained with by Nigerian brewers in enhancing or boosting their profitability. Nyabwanga and Ojera[8] (2012) Their research concentrate relationship among inventory management with business performance of smallscale enterprises (SSEs), in Kisii Municipality, Kisii County,Kenya. They used a cross-sectional survey study based on a small sample size of 79 SSEs. The study inferred that inventory comprised the maximum portion of working capital, and improper management of working capital was one of the major reasons of SSE failures. The empirical results disclosed that a positive significant relationship existed between business performance and inventory management practices with inventory budgeting having the maximum influence on business performance ensued by shelf-space management. The study suggested that by following effective inventory management practices business performance can be enhanced.n loss of profit. Sahari, Tinggi and Kadri [9]( 2012) They focused on association among the inventory management system and company performance corresponding to fund capability. Therefore according to that reason they looked 82 sample construction company in Malaysia during period of 2006– 2010. Using the regression and correlation analysis methods, they deduced that inventory management is positively correlated with firm performance. In addition, the results indicate that there is a positive link between inventory management and capital intensity. Soni [10] (2012) Made an in depth study of practices followed in regard to inventory management in the engineering goods industry in Punjab. The analysis used a

sample of 11 companies for a period five years, that is, 2004–2009 and was done using panel data set. The adequate and timely flow of inventory determines the success of an industry. She concluded that size of inventory enhanced marginally over the period as compared to a hike in current assets and net working capital. Inventories constituted half of the working capital which was due to overstocking of inventory as a result of low inventory turnover especially for finished goods and raw materials. Rise in sales and favourable market conditions lead to a rise in inventory levels. It was also inferred that sales increased more as compared to inventory. Lwini et al [11] (2013) A survey conducted on all the eight (8) sugar manufacturing firms in Kenya established that there is generally positive correlation between each of inventory management practices. Specific performance indicators were proved to depend on the level of inventory management practices. They established that Return on Equity had a strong correlation with lean inventory system and strategic supplier partnerships. As such, they concluded that the performance of sugar firms could therefore be stated as being a function of their inventory management practices. Panigrahi [12] (2013) According to his analysis inventory management practices used by Indian cement firms and their effects must be on working fund efficiency. The study also investigated the relationship between profitability and inventory conversion days. The study, using a sample of the top five cement companies of India over a period of 10 years from 2001 to 2010, concluded there must be exist inverse relationship among conversion period of inventory and profit margin. Madishetti and Kibona [13] (2013) Found that a well designed and executed inventory management contributes positively to a small or medium-sized enterprises (SMEs) profitability. They studied the association between inventory conversion period and



profitability and the impact of inventory management on SMEs profitability. They took a sample of 26 Tanzanian SMEs, and used the data from financial statements for the period 2006–2011. Regression analysis was adopted to determine the impact of inventory conversion period over gross operating profit. The results cleared out that significant negative linear relationship occurred between inventory conversion period and profitability.

Srinivas Rao Kasisomayajula [14] (2014) His research title based on the "Inventory Management in Commercial Vehicle Industry In India". There were five sample firms had preferred for study. The study concluded that all the units in the commercial vehicle industry have significant relationship between Inventory and Sales. Proper management of inventory is important to maintain and improve the health of an organization. Efficient management of inventories will improve the profitability of the organization.

Edwin Sitienei and Florence Memba [15] (2015) Conducted a study on Effect of Inventory Management on profitability of Cement Manufacturing Companies in Kenya. The study concluded that Gross profit margin is negatively correlated with the inventory conversion period, Increase in sales, which denotes the firm size enriches the firm's inventory levels, which pushes profits upwards due to optimal inventory levels. It is also noted that firms inventory systems must maintain an appropriate inventory levels to enhance profitability and reduce the inventory costs associated with holding excessive stock in warehouses

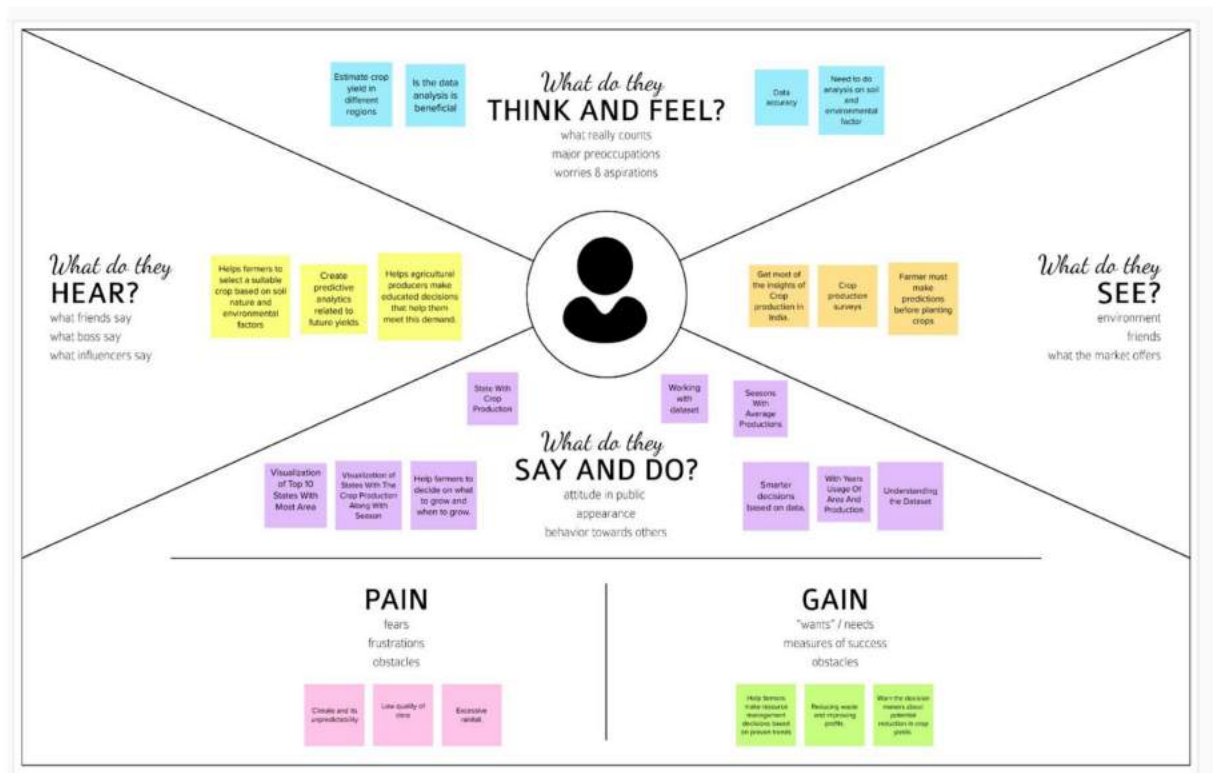
## 4. IDEATION AND PROPOSED SOLUTION

Proposed solution is a IBM cloud based application is deployed with which any customer can create their own account. They login and view the stocks, add new stocks, sales of each product groups, and even can email notification if no stocks available.

**Solution Architecture:** Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- Retailors login into the application and can manage stocks based on purchase rate of each product • Products can be purchased by Customers by creating their account. Whenever a new customer is traced the details are fetched, processed and recorded into the IBM DB2 Database by Backend servers dockerized.
  - Customers can login to their account and can purchase products and previously carted items as the sessions are restored from IBM DB2
  - The reason we have used IBM DB2 at backend for storage is it is highly available and can be scaled up by adding additional nodes to the cluster with minimal operational downtime
  - The fast selling and products which are to be soon out of stock is predicted using a machine learning model
  - Retailors gets a regular notification of products that are soon to be out-ofstock and that are of no stock available. Hence enabling better restocking of unavailable products
- Solution Architecture Diagram: Fig1:** Fig1 represents the solution architecture of inventory management application for retailors deployed in IBM Cloud - User account maintenance - Stock monitored by Retailor(supplier) - API request processed by docker containers - Data stored in IBM DB2 - Notification of stocks using SendGrid Email - Stock behavior analyzing using Machine learning

## EMPATHY MAP



## 5 . REQUIREMENT ANALYSIS

### **IBM Cognos Analytics**

A collection of business intelligence tools called IBM Cognos Analytics is offered both on-premises and in the cloud. The main emphasis is on descriptive analytics, which uses dashboards, expert reporting, and self-service data exploration to help users understand the information in your data. In this study, we analysed the crop yield data using IBM cognos data analytics.

Following are important features of IBM Cognos:

1. **Get Connected** - Connect your data effortlessly Import data from CSV files and spreadsheets. Connect to cloud or on-premises data sources, including SQL databases, Google BigQuery, Amazon, Redshift, and more.
2. **Prepare your data** – Prepare and connect data automatically Save time cleaning your data with AI-assisted data preparation. Clean and prep data from multiple sources, add calculated fields, join data, and create new tables.
3. **Build visualizations** - Create dynamic dashboards easily Quickly create compelling, interactive dashboards. Drag and drop data to create auto-generated visualizations, drill down for more detail, and share using email or Slack.

4. **Identify Patterns – Uncover hidden patterns** Ask the AI assistant a question in plain language, and see the answer in visualization. Use time series modelling to predict seasonal trends.
5. **Generate Personalised Reports – Create and deliver personalized reports** Keep your stakeholders up-to-date, automatically. Create and share dynamic personalized, multi-page reports in the formats your stakeholders want.
6. **Gain Insights - Make confident data decisions** Get deeper insights without a data science background. Validate what you know, identify what you don't with statistically accurate time-series forecasting and pinpoint patterns to consider.
7. **Stay Connected – Go Mobile** Stay connected on the go with the new mobile app. Access data and get alerts right from your phone.

#### B. System Architecture

India is one of the top countries for agricultural production, making it one of the most significant sources of income. As part of this project, we will analyse some significant visualisations, build a dashboard, and then use this information to gain the majority of our understanding of crop output in India.

IBM® Cognos® Analytics integrates reporting, modeling, analysis, dashboards, stories, and event management so that you can understand your organization data, and make effective business decisions.

After the software is installed and configured, administrators set up security and manage data sources. You can get started yourself by uploading local files and applying visualizations in dashboards or stories. For enterprise-level data, modelers are next in the workflow. After data modules and packages are available, report authors can then create reports for business users and analysts. Administrators maintain the system on an ongoing basis.

Whether you're an analyst, report author, data modeler, or an administrator, you start by signing in to the Welcome portal from your desktop or mobile device. There are coach marks in the user interface to help you discover what's where.

## REQUIREMENT ANALYSIS

4.1 Functional requirement Following are the functional requirements of the proposed solution. FR No. Functional Requirement (Epic) Sub Requirement (Story / Sub-Task) FR-1 User Registration Registration through Form Registration through Gmail Registration through LinkedIn FR-2 User Confirmation Confirmation via Email Confirmation via OTP FR-3 Sign in Sign in to the application by LinkedIn/Gmail, Username and Password. FR-4 Dashboard Can view the product details and offers.

FR-5 Booking The required products are selected and booked. FR-6 Shipping To track the delivery details of the selected product. FR-7 Restocking Ordering more products when the stock is low. 4.2 Non-Functional requirements Following are the non-functional requirements of the proposed solution.

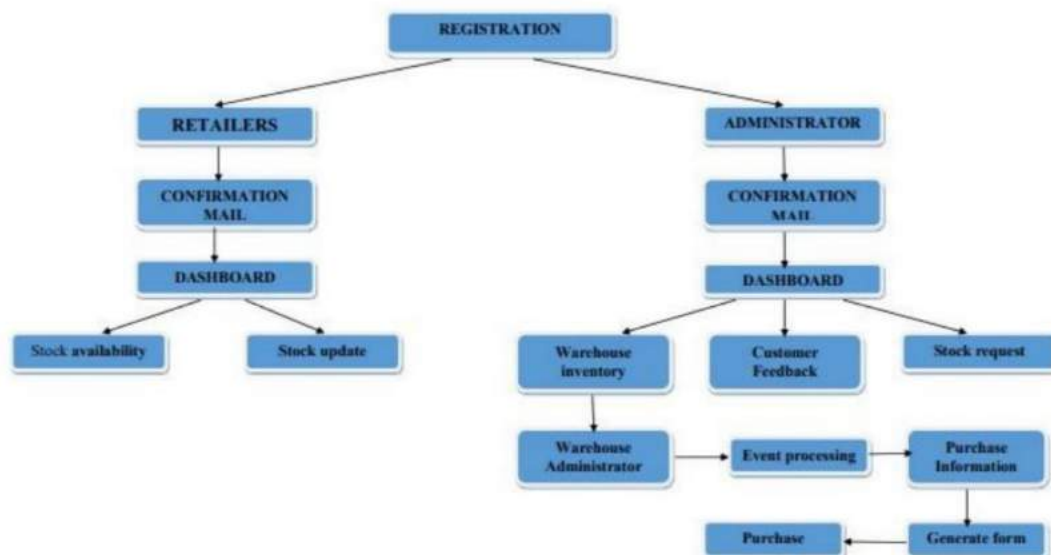
FR No. Non-Functional Requirement Description NFR-1 Usability

- ✓ Creating a learning curve into the site's design and development.
- ✓ Having a user-friendly, straightforward website. Beautiful-looking website.
- ✓ Making the website responsive for consumers on both desktops and mobile devices. NFR-2 Security
- ✓ Strong security is necessary to prevent hackers from accessing the accounts or data of authorized users. To demonstrate authentication and authorization, log in systems is utilized.
- ✓ Utilizing OTP can improve security.
- ✓ Cookies-based security mechanism for user authentication and enhanced website user
- ✓ experience NFR-3 Reliability
- ✓ When the website is active, it should be able to manage the necessary number of users without slowing or causing any inconvenience to the user.
- ✓ While running the apps, there should be few mistakes.
- ✓ It should be accessible even during disasters. NFR-4 Performance
- ✓ This has the advantage of cutting down the time needed for aisle and product searches, among other conveniences.
- ✓ It decreases expenses, saves time during restocking, and forecasts the top-selling goods.
- ✓ Due to the business's streamlined management system, it is more productive
- ✓ and profitable. NFR - 5 Availability
- ✓ To provide high availability of database servers and performances, this employs IBM
- ✓ DB2. NFR - 6 Scalability
- ✓ Due to DB2's excellent scalability, coding can be created and developed quickly, and new features can be added without much difficulty.

✓ High -scalability IBM Container is utilised in the Docker registry. ✓ Any new functionality can be added by reusing the code.

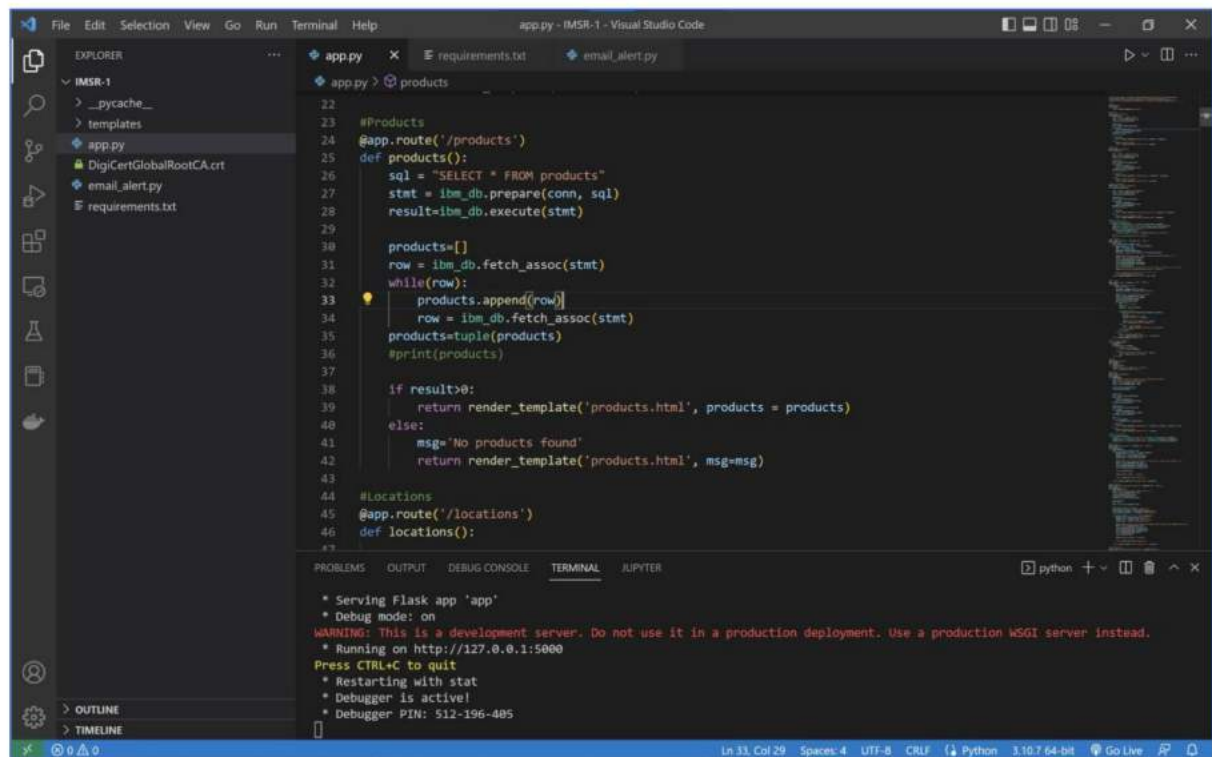
## Data Flow Diagrams

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.





Example, (For Products page)



```
22
23 #Products
24 @app.route('/products')
25 def products():
26     sql = "SELECT * FROM products"
27     stmt = ibm_db.prepare(conn, sql)
28     result=ibm_db.execute(stmt)
29
30     products=[]
31     row = ibm_db.fetch_assoc(stmt)
32     while(row):
33         products.append(row)
34         row = ibm_db.fetch_assoc(stmt)
35     products=tuple(products)
36     #print(products)
37
38     if result>0:
39         return render_template('products.html', products = products)
40     else:
41         msg="No products found"
42         return render_template('products.html', msg=msg)
43
44 #Locations
45 @app.route('/locations')
46 def locations():
47
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

python + - - ^ x

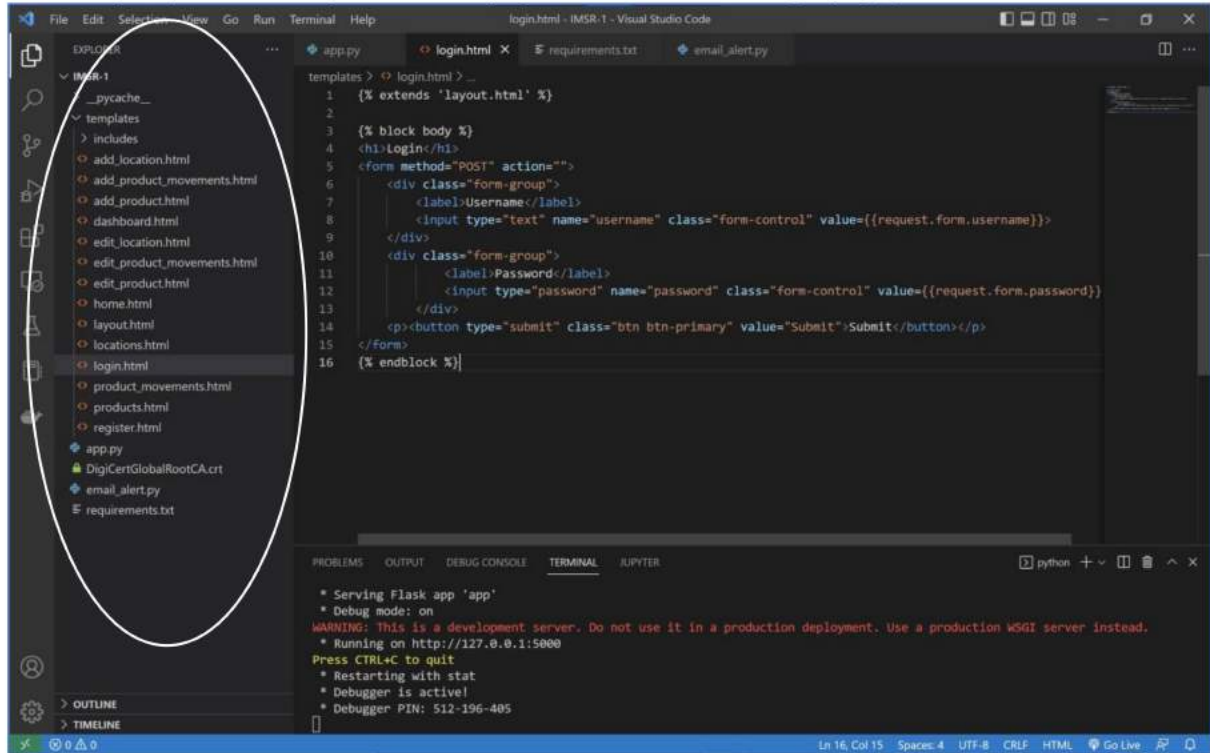
```
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 512-196-405
```

Ln 33, Col 29 Spaces: 4 UTF-8 CRLF Python 3.10.7 64-bit Go Live

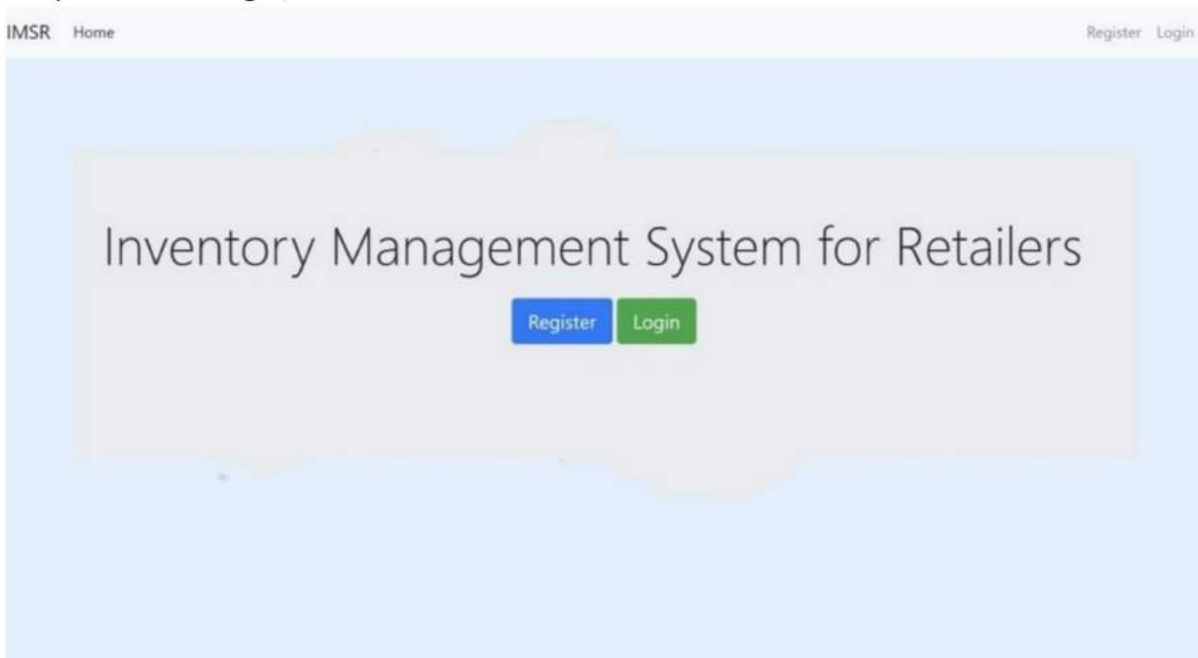
**Sprint 2 – Frontend:**

The frontend is written using HTML, CSS (using Bootstrap) and JavaScript for all the pages to which the routes created in Sprint 1.

For Example, (The Hierarchy of different pages and the code for login page)



Sample FrontEnd Pages,



Login Page,

[Home](#)[Register](#)[Login](#)

## Login

Username

Password

[Submit](#)

### Register Page,

[Home](#)[Register](#)[Login](#)

## Register

Name

Email

Username

Password

Confirm Password

[Submit](#)

### Products Page,

[Home](#)

[Products](#)

[Location](#)

[Product Movements](#)

[Logout](#)

[Dashboard](#)

# Products

Add Product

Product ID	Product Cost	Product Quantity		
Bedspreads	600	100	<a href="#">Edit</a>	<a href="#">Delete</a>
Cutlery	1500	495	<a href="#">Edit</a>	<a href="#">Delete</a>
Shampoo	50	520	<a href="#">Edit</a>	<a href="#">Delete</a>

## Product Movements Page,

[Home](#)
[Products](#)
[Location](#)
[Product Movements](#)

[Logout](#)
[Dashboard](#)

# Product Movements

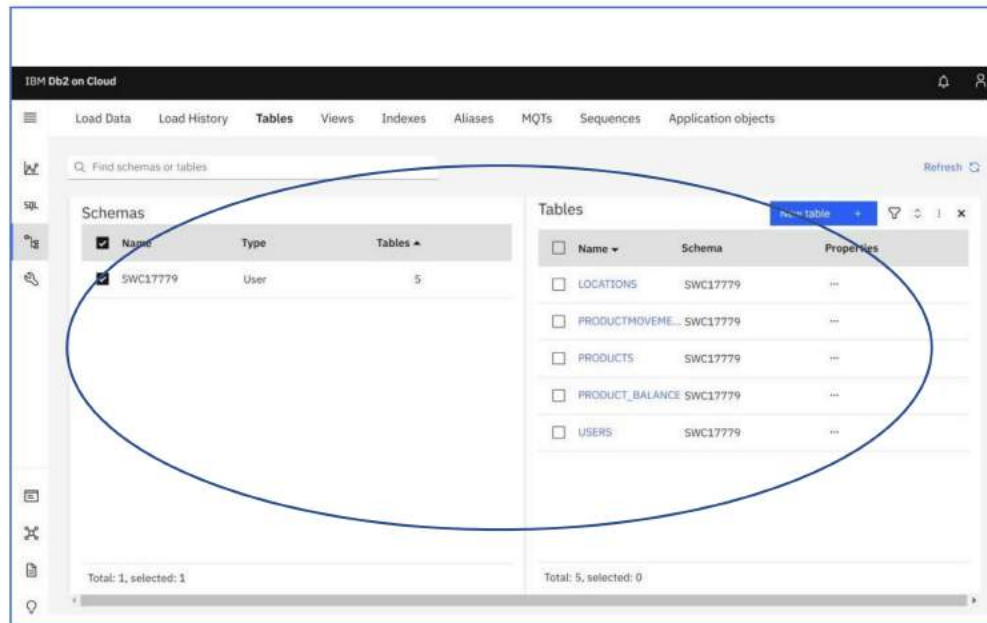
Add Product Movements

Movement ID	Time	From Location	To Location	Product ID	Quantity	
41	2022-11-14 04:32:57.213981	Chennai	Main Inventory	Shampoo	20	Delete
42	2022-11-14 04:51:47.519001	Chennai	Karnataka	Shampoo	1553	Delete
40	2022-11-14 03:57:52.649656	Bangalore	Chennai	Shampoo	100	Delete

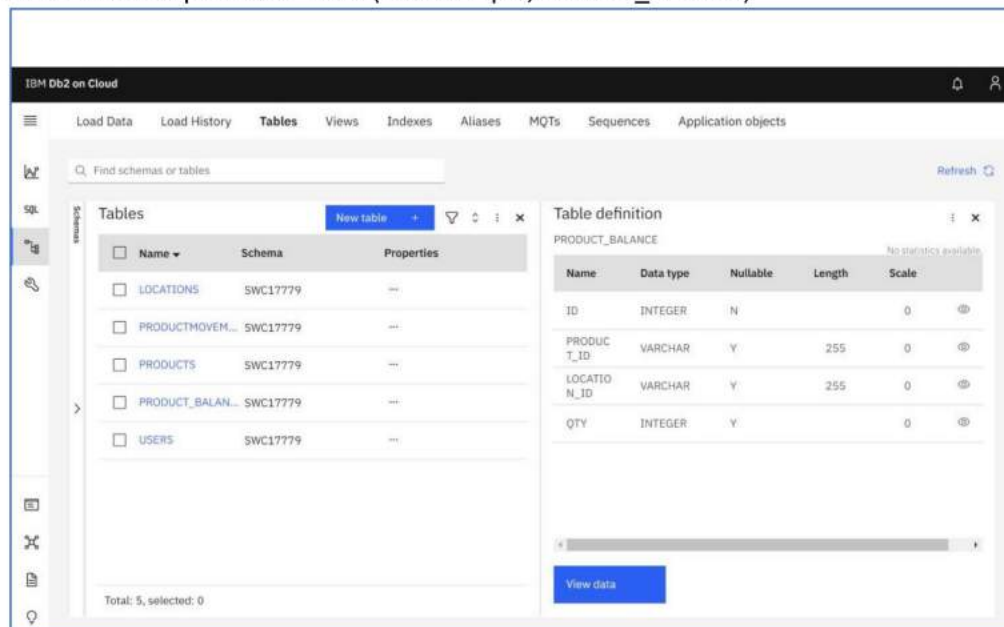
## Sprint 3 - IBM Cloud Integration + Integration of SendGrid:

### IBM Cloud Integration:

5 tables created for our project,



Schema of the particular table (For Example, Product\_Balance)



Data of a particular table (For Example, Product\_Balance)

The screenshot shows the IBM Db2 on Cloud web interface. The top navigation bar includes 'Load Data', 'Load History', 'Tables', 'Views', 'Indexes', 'Aliases', 'MQTs', 'Sequences', and 'Application objects'. The 'Tables' tab is selected, and the table 'SWC17779.PRODUCT\_BALANCE' is displayed. A 'Back' button is in the top right. Below the table name, there is a trash icon and an 'Export to CSV' button. The table has four columns: ID, PRODUCT\_ID, LOCATION\_ID, and QTY. It contains 8 rows of data.

ID	PRODUCT_ID	LOCATION_ID	QTY
1	Shampoo	Kerala	1350
2	Bedspreads	Kerala	100
3	Shampoo	Chennai	1452
4	Shampoo	Mumbai	100
5	Shampoo	Karnataka	-202
6	Shampoo	Punjab	100
7	Shampoo	Bangalore	-451
8	Cutlery	Bangalore	55

Code for Connection of IBM Database,

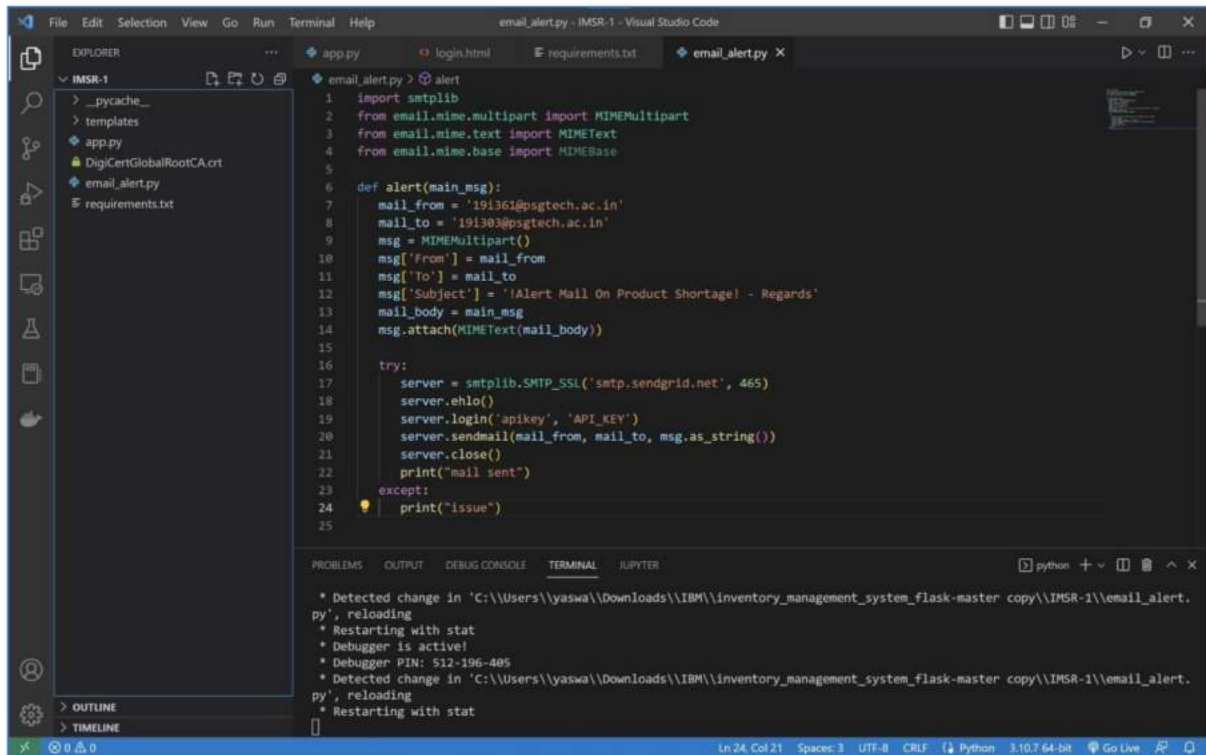
```
conn=ibm_db.connect("DATABASE=bludb;HOSTNAME=55fbc997-9266-4331-afd3-888b05e734c0.bs2io90l08kqb1od8lcg.databases.appdomain.cloud;PORT=;SECURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=;PWD=','','')

```

**Note:** DigiCertGlobalRootCA.crt should be downloaded and configured within the project folder.

**SendGrid Integration:**

## Code for email alert



The screenshot shows the Visual Studio Code interface with the file explorer on the left displaying the project structure for 'IMSR-1'. The main editor window shows the code for 'email\_alert.py'. The code imports 'smtplib' and 'email.mime' modules, and defines an 'alert' function that constructs an email message and sends it via SendGrid. The terminal at the bottom shows the output of the application, including messages about detected changes and restarts.

```
1 import smtplib
2 from email.mime.multipart import MIMEMultipart
3 from email.mime.text import MIMEText
4 from email.mime.base import MIMEBase
5
6 def alert(main_msg):
7     mail_from = '191361@psgtech.ac.in'
8     mail_to = '191303@psgtech.ac.in'
9     msg = MIMEMultipart()
10    msg['From'] = mail_from
11    msg['To'] = mail_to
12    msg['Subject'] = 'Alert Mail On Product Shortage! - Regards'
13    mail_body = main_msg
14    msg.attach(MIMEText(mail_body))
15
16    try:
17        server = smtplib.SMTP_SSL('smtp.sendgrid.net', 465)
18        server.ehlo()
19        server.login('apikey', 'API_KEY')
20        server.sendmail(mail_from, mail_to, msg.as_string())
21        server.close()
22        print("mail sent")
23    except:
24        print("issue")
25
```

Terminal Output:

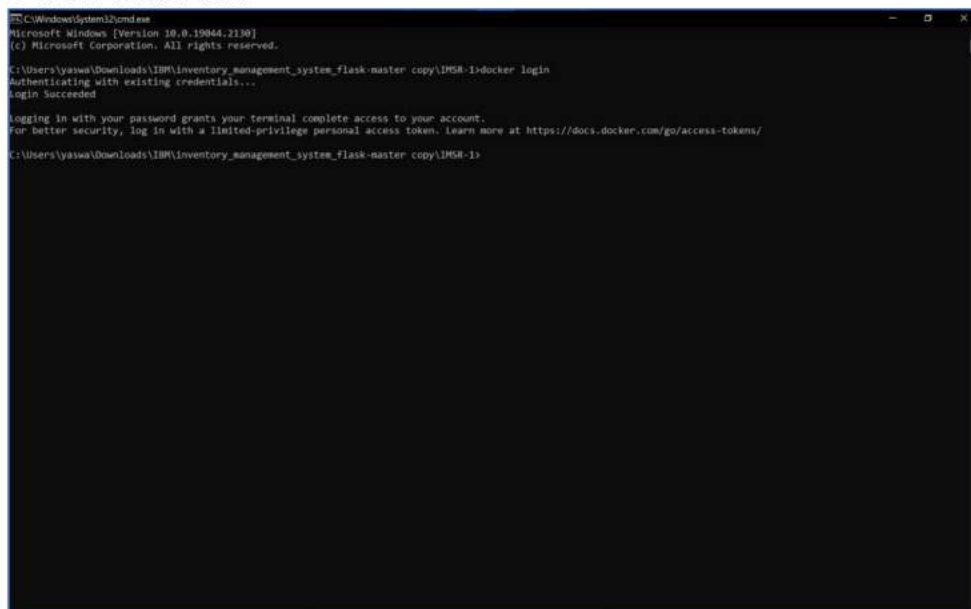
```
* Detected change in 'C:\Users\yasma\Downloads\IBM\inventory_management_system_flask-master copy\IMSR-1\email_alert.py', reloading
* Restarting with stat
* Debugger is active!
* Debugger PIN: 512-196-405
* Detected change in 'C:\Users\yasma\Downloads\IBM\inventory_management_system_flask-master copy\IMSR-1\email_alert.py', reloading
* Restarting with stat
```

Email Received on Shortage of materials at particular warehouse or Main Inventory:

## Sprint 4 (Deploying the application using Docker and Kubernetes):

**Note:** Make sure to create a Dockerfile in the project folder.

Login into DockerHub in Project Folder using command prompt. This connects local docker desktop to cloud docker hub.



The screenshot shows a Windows command prompt window with the following text:

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.

C:\Users\yasma\Downloads\IBM\inventory_management_system_flask-master copy\IMSR-1>docker login
Authenticating with existing credentials...
Login Succeeded

Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/

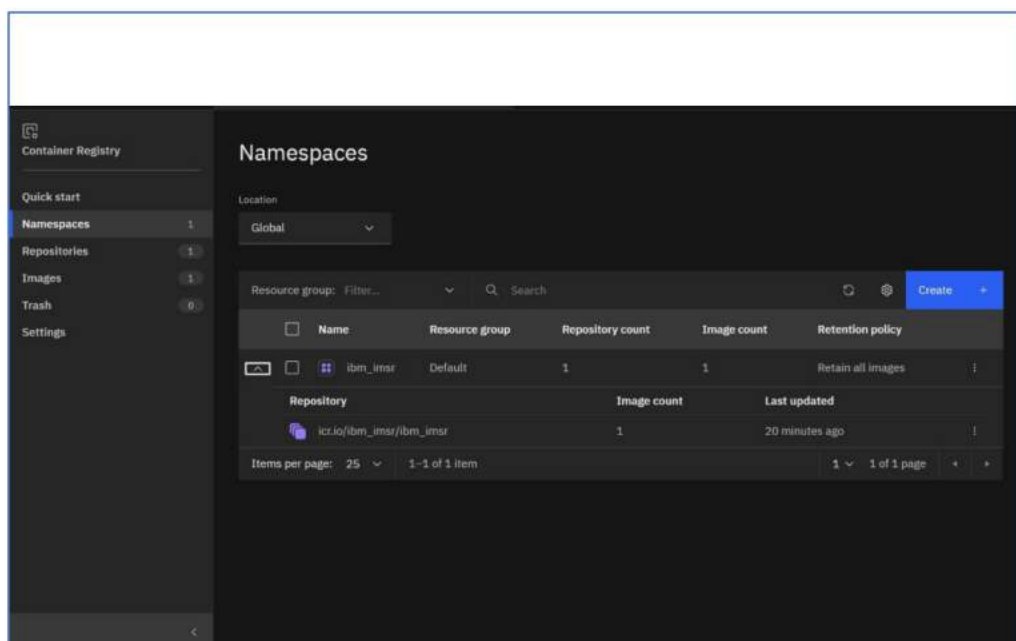
C:\Users\yasma\Downloads\IBM\inventory_management_system_flask-master copy\IMSR-1>
```



Building an image for our project,

```
File "/usr/local/lib/python3.11/site-packages/flask/app.py", line 1820, in full_dispatch_request
PS C:\Users\yaswa\Downloads\IBM\IMSR-1> docker build -t yaswanthmanoharan/ibm_imsr .
[+] Building 2.7s (11/11) FINISHED
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 32B                                              0.0s
=> [internal] load .dockerignore                                                0.0s
=> => transferring context: 2B                                                  0.0s
=> [internal] load metadata for docker.io/library/python:latest                2.4s
=> [auth] library/python:pull token for registry-1.docker.io                  0.0s
=> [internal] load build context                                                0.0s
=> => transferring context: 24.29kB                                             0.0s
=> CACHED [2/5] WORKDIR /inventory                                              0.0s
=> CACHED [3/5] COPY requirements.txt requirements.txt                          0.0s
=> CACHED [4/5] RUN pip install -r requirements.txt                            0.0s
=> [5/5] COPY . .                                                              0.0s
=> exporting to image                                                          0.1s
=> => exporting layers                                                         0.0s
=> => writing image sha256:0afb0c793a704eaf85acc886443c57a0cbeca9473b841897ef4a9162f3c4bd06 0.0s
=> => naming to docker.io/yaswanthmanoharan/ibm_imsr                          0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
PS C:\Users\yaswa\Downloads\IBM\IMSR-1> docker run -p 8080:5000 yaswanthmanoharan/ibm_imsr
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI serve
r instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://172.17.0.2:5000
Press CTRL+C to quit
172.17.0.1 - - [14/Nov/2022 03:57:11] "GET /login HTTP/1.1" 200 -
172.17.0.1 - - [14/Nov/2022 03:57:22] "POST /login HTTP/1.1" 302 -
172.17.0.1 - - [14/Nov/2022 03:57:23] "GET /dashboard HTTP/1.1" 200 -
172.17.0.1 - - [14/Nov/2022 03:57:27] "GET /product_movements HTTP/1.1" 200 -
172.17.0.1 - - [14/Nov/2022 03:57:30] "GET /add_product_movements HTTP/1.1" 200 -
[2022-11-14 03:57:37,822] ERROR in app: Exception on /add_product_movements [POST]
Traceback (most recent call last):
```



Pushing the project into IBM container Registry,



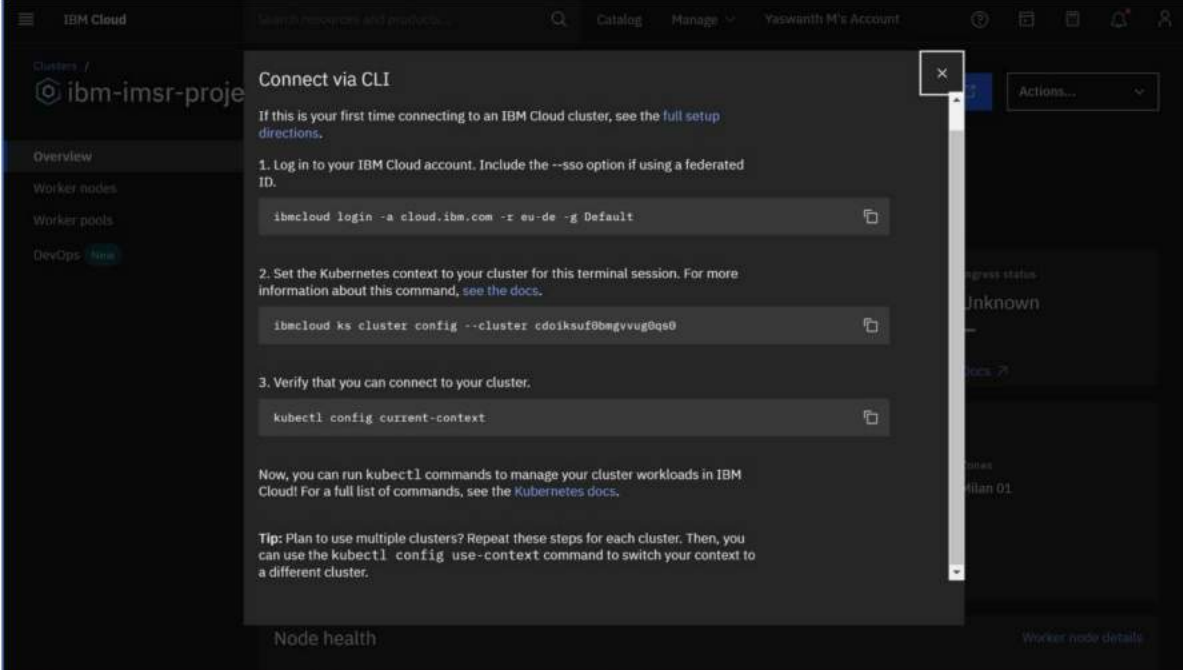
```

5b3f1ed98915: Pushing [====>] 6.053MB/67.7882fd36bfd35: P
ushing 174.2MB/529MB
d5b2c4afb8d6: Pushing [=====>] 40.6MB/191.6MB
Using default tag: latest
The push refers to repository [icr.io/ibm-imrsr/ibm-imrsr]
5b183c62e3d7: Pushing [=====>] 6.465MB/18.48MB
d5b2c4afb8d6: Pushing [====>] 17.2MB/191.6MB
d5b2c4afb8d6: Pushing [=====>] 75.71MB/191.6MB
4db7c1329ec9: Pushed
6f6e69c2c592: Pushed
882fd36bfd35: Pushing [=====>] 308.4MB/529MB
d5b2c4afb8d6: Pushing 138.5MB/191.6MB
d5b2c4afb8d6: Pushed
5b183c62e3d7: Pushing [=====>] 5.285MB/18.48MB
882fd36bfd35: Pushing 175.3MB/529MB
882fd36bfd35: Pushing [=====>] 319MB/529M5b3f1ed98915: P
ushed
d1dec9917839: Pushing [>] 2.735MB/152M882fd8888882
882fd36bfd35: Pushed
d1dec9917839: Pushed
d1dec9917839: Pushing 70.76MB/152MB
d38adf39e1dd: Pushed
d9d07d703dd5: Pushed
latest: digest: sha256:0575b171d321ade1d5a3def1d1bb5afe8a00d00c1f7e157a5347aca6a6ee1470 size: 3052
882fd36bfd35: Pushing [=====>] 265.7MB/529MB
C:\Users\yaswa>shing [=====>] 264MB/529MB
d1dec9917839: Pushing [>] 1.62MB/152MB

```

**Note:** Create a Kubernetes Cluster in IBM Cloud and wait for the work node to get fully deployed.

Then, Login into Kubernetes Cluster using the following commands,



The screenshot shows the IBM Cloud console interface. On the left, there is a sidebar with navigation links: 'Clusters / ibm-imrsr-proje', 'Overview', 'Worker nodes', 'Worker pools', and 'DevOps / New'. The main panel displays the 'Connect via CLI' section for a cluster. It includes instructions for logging in, setting the Kubernetes context, and verifying the connection. The instructions are as follows:

1. Log in to your IBM Cloud account. Include the `--sso` option if using a federated ID.  
`ibmcloud login -a cloud.ibm.com -r eu-de -g Default`
2. Set the Kubernetes context to your cluster for this terminal session. For more information about this command, [see the docs](#).  
`ibmcloud ks cluster config --cluster cdoiksuf0bmvgvug0qs0`
3. Verify that you can connect to your cluster.  
`kubectl config current-context`

Now, you can run `kubectl` commands to manage your cluster workloads in IBM Cloud! For a full list of commands, see the [Kubernetes docs](#).

**Tip:** Plan to use multiple clusters? Repeat these steps for each cluster. Then, you can use the `kubectl config use-context` command to switch your context to a different cluster.

At the bottom of the panel, there are sections for 'Node health' and 'Worker node details'.

Expose your application using the following command and check for the port number using the next command.

```
Command Prompt
The configuration for cdoiksuf0bmgyvug0qs0 was downloaded successfully.
Added context for cdoiksuf0bmgyvug0qs0 to the current kubeconfig file.
You can now execute 'kubectl' commands against your cluster. For example, run 'kubectl get nodes'.
If you are accessing the cluster for the first time, 'kubectl' commands might fail for a few seconds while RBAC synchronizes.

C:\Users\yaswa>kubectl get pods
NAME                                READY   STATUS             RESTARTS   AGE
ibm-inventory-management-system-for-retailers-6cd7dfcc7b-8q2w2  1/1     Running            0          10h
ibm-project-9bbb47d-5vn2w          1/1     Running            0          9h
ibmimsr-586d66c8c8-kkjqp          0/1     ContainerCreating  0          26s

C:\Users\yaswa>kubectl expose deployment ibmimsr --type=NodePort --name=ibmimsr
service/ibmimsr exposed

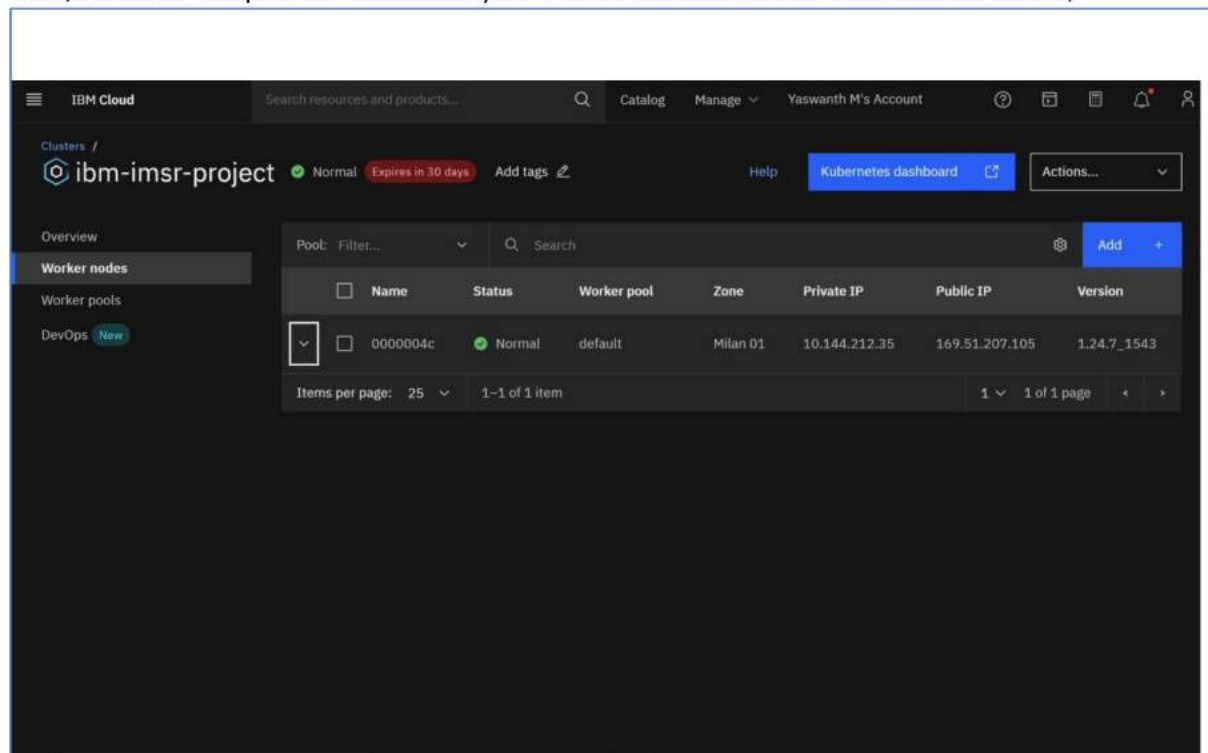
C:\Users\yaswa>kubectl describe service ibmimsr
error: unknown command "describe" for "kubectl"

Did you mean this?
    describe

C:\Users\yaswa>kubectl describe service ibmimsr
Name:                ibmimsr
Namespace:           default
Labels:              app=ibmimsr
Annotations:          <none>
Selector:             app=ibmimsr
Type:                NodePort
IP Family Policy:     SingleStack
IP Families:          IPv4
IP:                  172.21.98.28
IPs:                  172.21.98.28
Port:                 <unset> 5000/TCP
TargetPort:           <unset> 5000/TCP
NodePort:             <unset> 30958/TCP
Endpoints:            172.30.116.13:5000
Session Affinity:     None
External Traffic Policy: Cluster
Events:               <none>

C:\Users\yaswa>
```

Then, Check for the public IP address in your IBM Kubernetes Cluster under Worker Node,

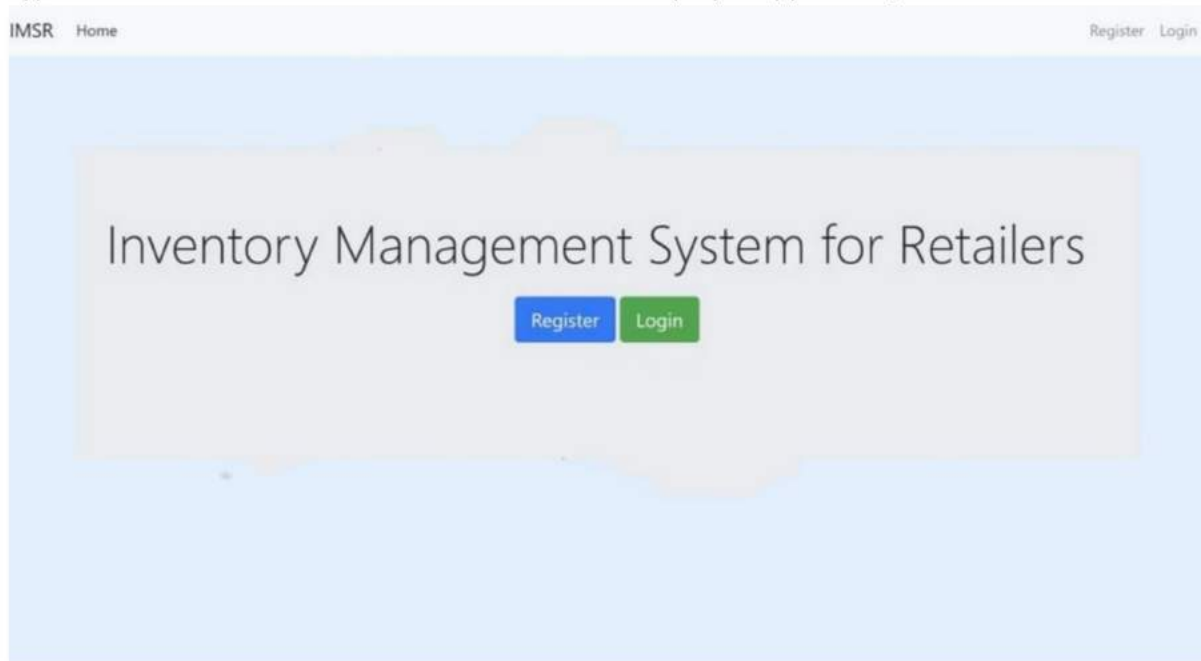


Thus we have the Public IP address and the Nodeport.

Now just type in this format - <Public\_IP>:<NodePort>

For our Inventory management system application it is, **169.51.207.105:30958**

Type this in the browser and click enter to access the deployed application,



**Result:**

Thus In this way We developed a “Inventory management System for Retailers” using Python, Sendgrid and IBM Cloud Services (IBM DB2, IBM Container registry, IBM Kubernetes).

# Thank You!