Final Deliverables Report

Date	14.11.2022
Team ID	PNT2022TMID13127
Project Name	Inventory Management System for Retailers

Team members and their Contribution:

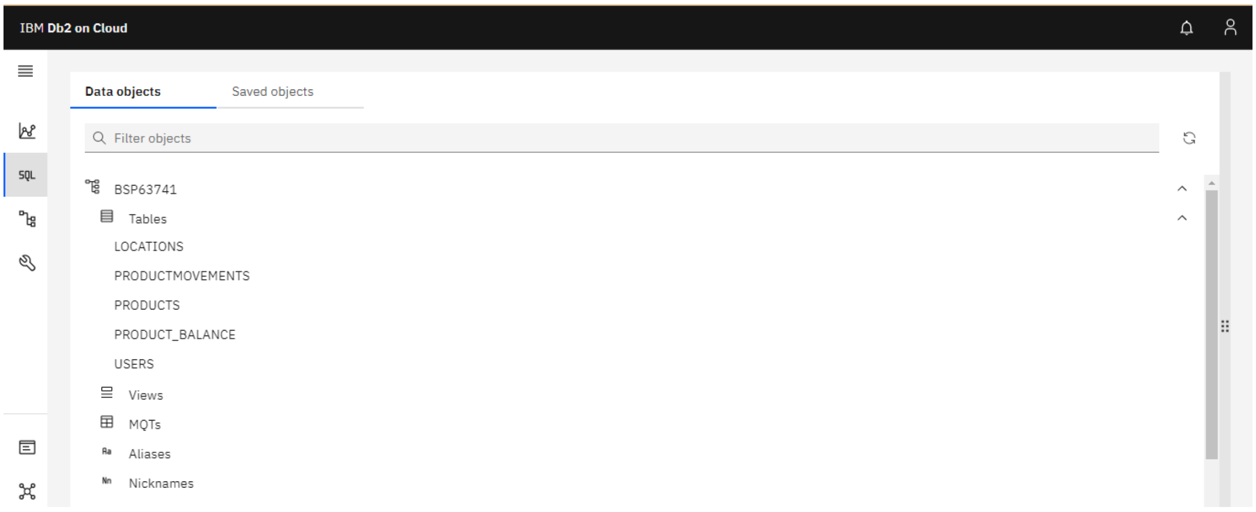
Name	Roll no	Contribution
Harisudhan T	782519I317	Backend, Integration of Sendgrid, Deployment of using docker and Kubernetes.
Jeevanantham V K	782519I323	Frontend, Deployment of using docker and Kubernetes.
Jhanani J	782519I324	Frontend, Backend, Documentation
Kavi Varshini S	782519I326	Backend, Integration of IBM Cloud.

Introduction:

1. Sprint 1 – Backend and Frontend
2. Sprint 2 – Frontend
3. Sprint 3 – IBM Cloud Integration + Integration of SendGrid
4. Sprint 4 – Deploying the application using Docker and Kubernetes

Sprint 1 – Backend and Frontend :

- Create Database and necessary Tables
- Insert Values into the Database
- Create Login Page and New Registration Page.
- Create Frontend Pages of the application



Database

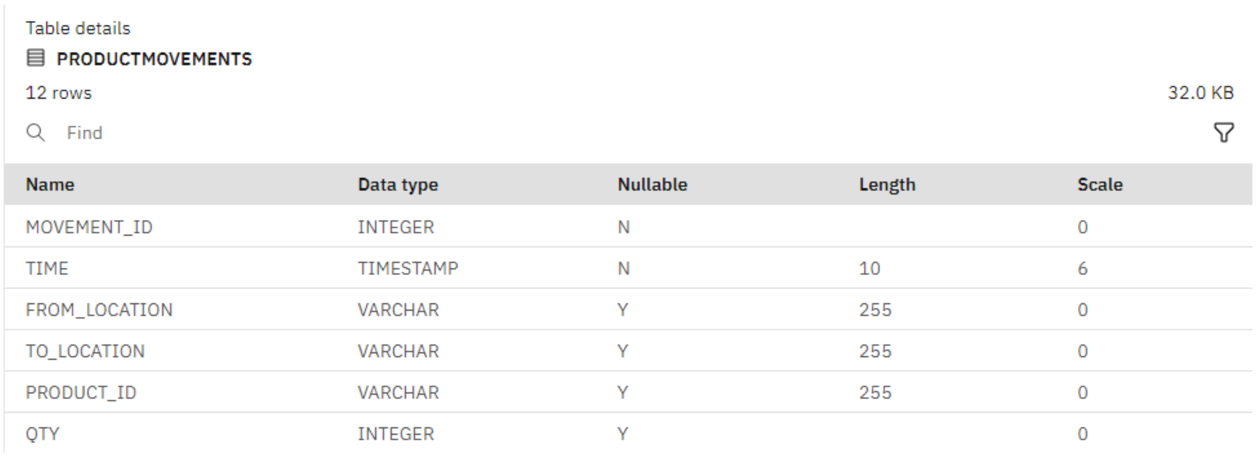


Table details
PRODUCTMOVEMENTS
12 rows 32.0 KB

Name	Data type	Nullable	Length	Scale
MOVEMENT_ID	INTEGER	N		0
TIME	TIMESTAMP	N	10	6
FROM_LOCATION	VARCHAR	Y	255	0
TO_LOCATION	VARCHAR	Y	255	0
PRODUCT_ID	VARCHAR	Y	255	0
QTY	INTEGER	Y		0

Product movements table

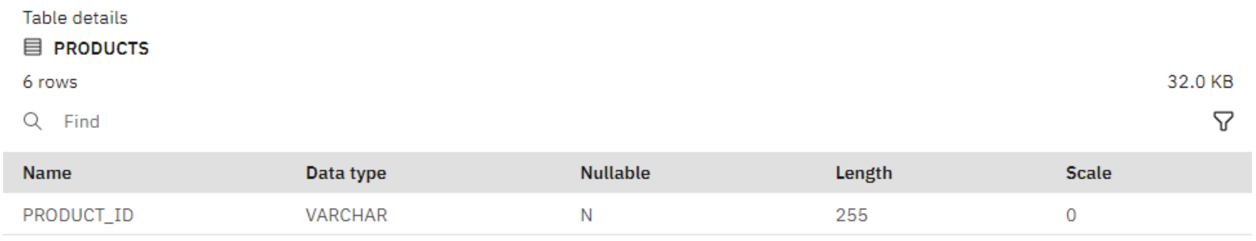


Table details
PRODUCTS
6 rows 32.0 KB

Name	Data type	Nullable	Length	Scale
PRODUCT_ID	VARCHAR	N	255	0

Products table

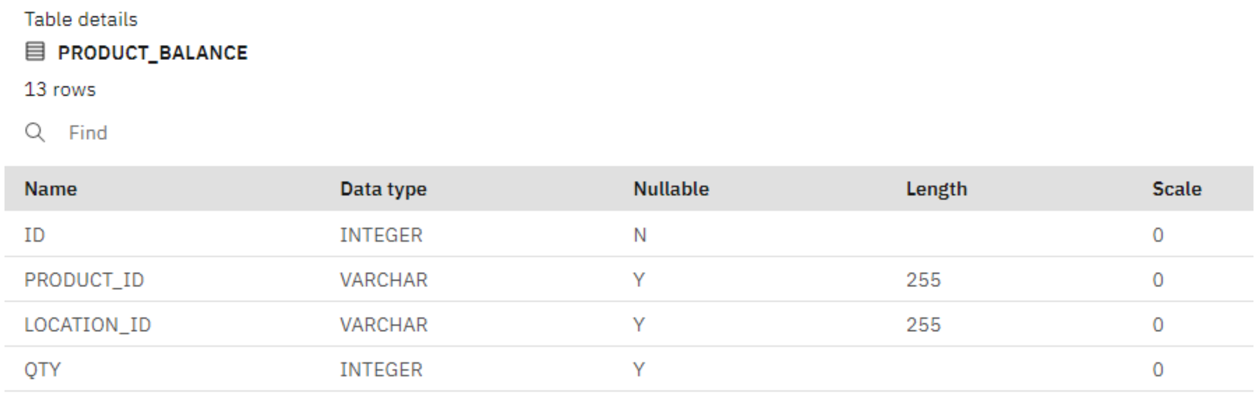


Table details

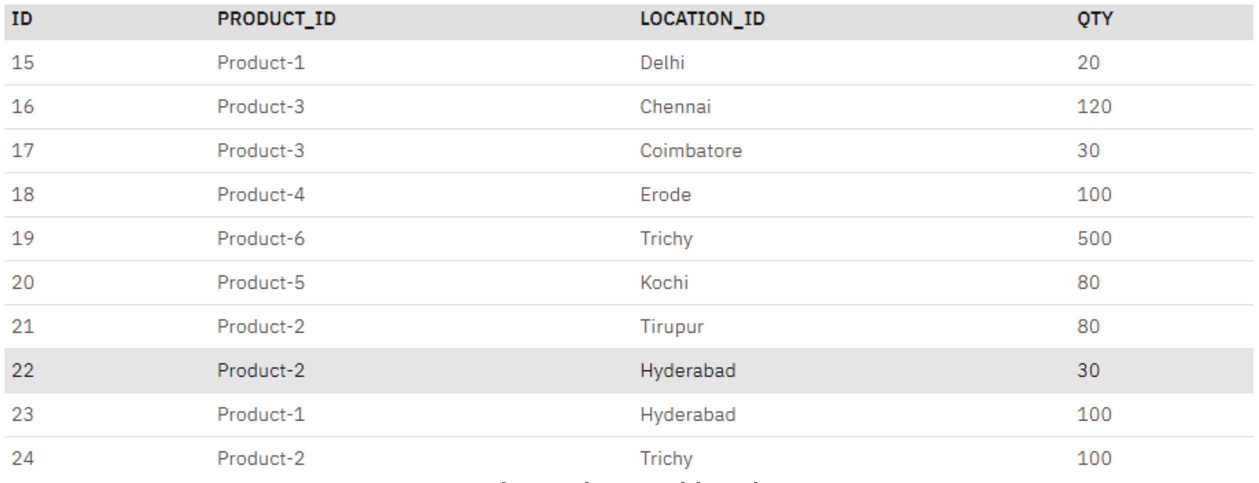
PRODUCT_BALANCE

13 rows

Find

Name	Data type	Nullable	Length	Scale
ID	INTEGER	N		0
PRODUCT_ID	VARCHAR	Y	255	0
LOCATION_ID	VARCHAR	Y	255	0
QTY	INTEGER	Y		0

Product Balance table



ID	PRODUCT_ID	LOCATION_ID	QTY
15	Product-1	Delhi	20
16	Product-3	Chennai	120
17	Product-3	Coimbatore	30
18	Product-4	Erode	100
19	Product-6	Trichy	500
20	Product-5	Kochi	80
21	Product-2	Tirupur	80
22	Product-2	Hyderabad	30
23	Product-1	Hyderabad	100
24	Product-2	Trichy	100

Product Balance table values

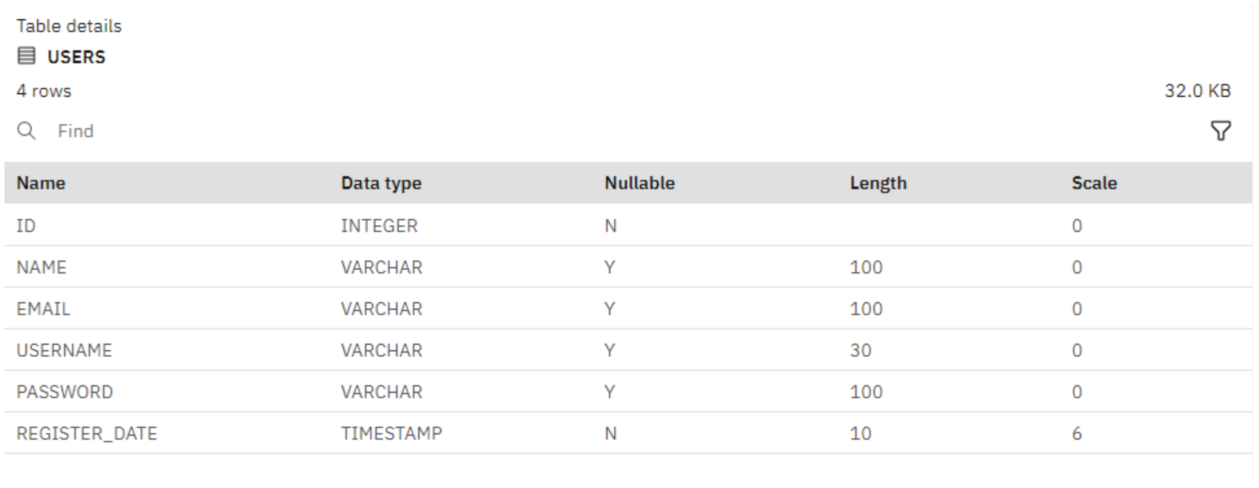


Table details

USERS

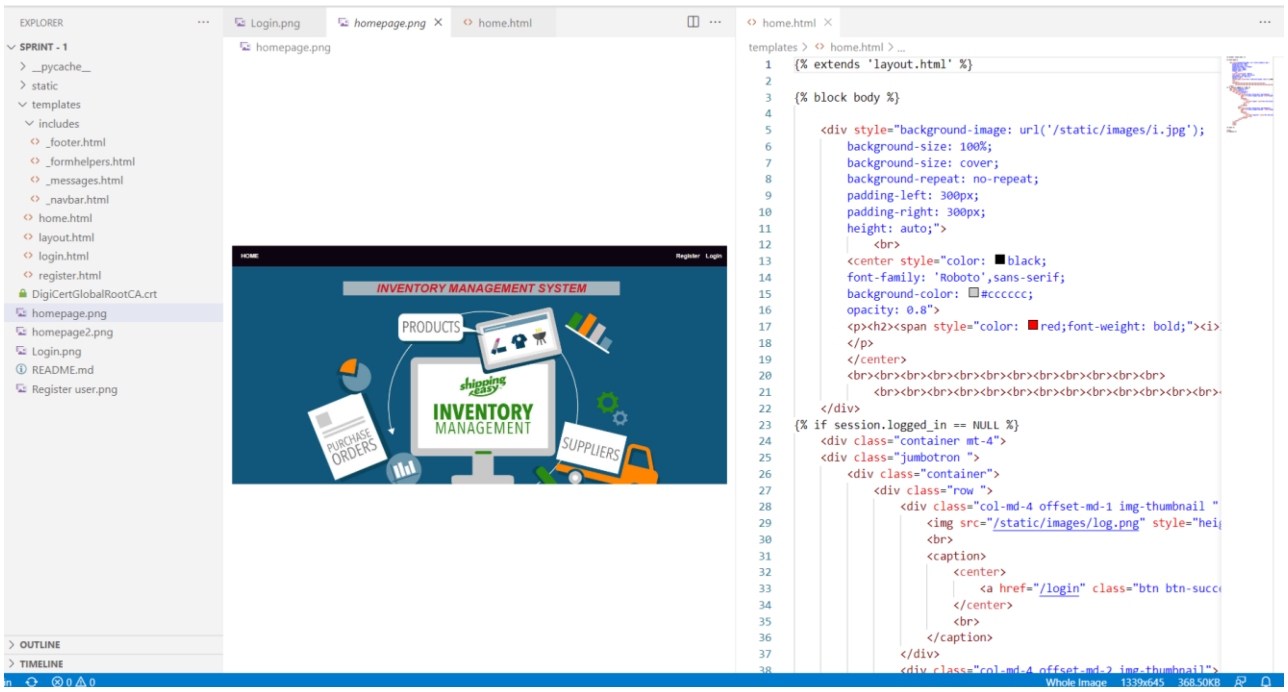
4 rows

32.0 KB

Find

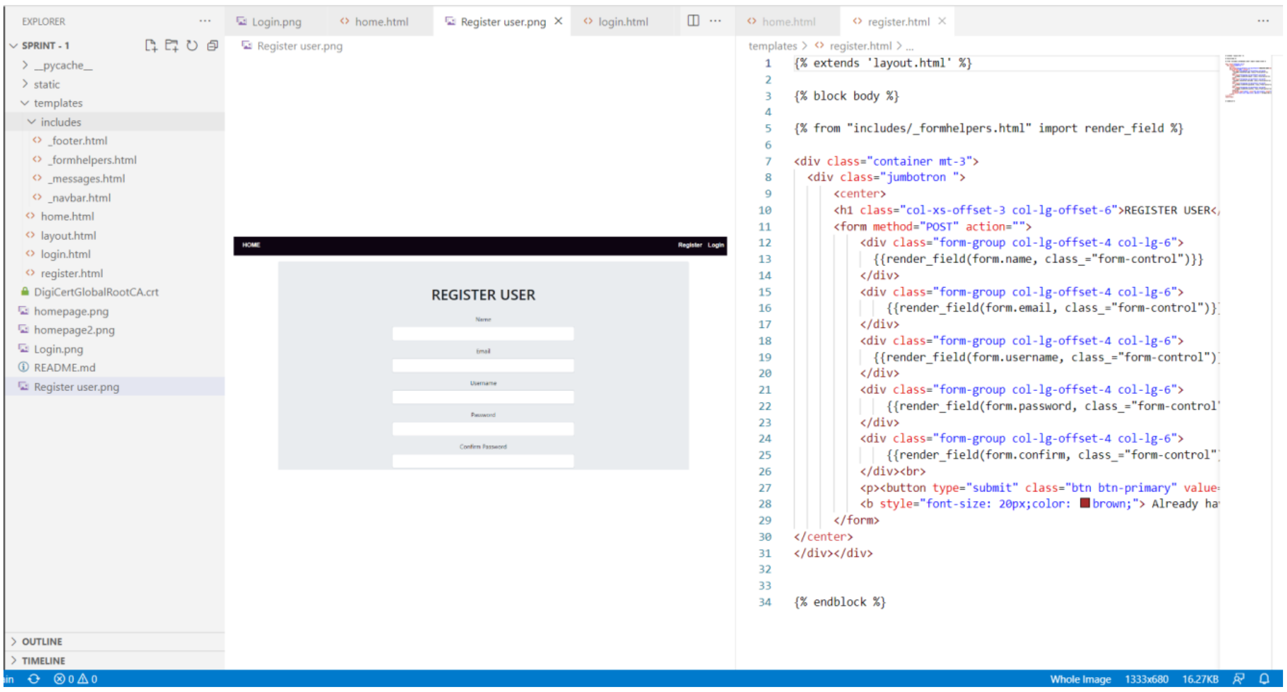
Name	Data type	Nullable	Length	Scale
ID	INTEGER	N		0
NAME	VARCHAR	Y	100	0
EMAIL	VARCHAR	Y	100	0
USERNAME	VARCHAR	Y	30	0
PASSWORD	VARCHAR	Y	100	0
REGISTER_DATE	TIMESTAMP	N	10	6

Users Table



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

{% block body %}

    <div style="background-image: url('/static/images/i.jpg');
        background-size: 100%;
        background-size: cover;
        background-repeat: no-repeat;
        padding-left: 300px;
        padding-right: 300px;
        height: auto;">
            <br>
        <center style="color: black;
        font-family: 'Roboto',sans-serif;
        background-color: #cccccc;
        opacity: 0.8">
        <p><h2><span style="color: red;font-weight: bold;"><i>
        </p>
        </center>
        <br><br><br><br><br><br><br><br><br><br><br><br>
            <br><br><br><br><br><br><br><br><br><br><br><br><br>
    </div>
{% if session.logged_in == NULL %}
    <div class="container mt-4">
    <div class="jumbotron ">
        <div class="container">
            <div class="row ">
                <div class="col-md-4 offset-md-1 img-thumbnail "
                    <img src="/static/images/log.png" style="hei
                    <br>
                    <caption>
                        <center>
                            <a href="/login" class="btn btn-succ
                        </center>
                        <br>
                    </caption>
                </div>
                <div class="col-md-4 offset-md-2 img-thumbnail">
```

Frontend of the Home Page

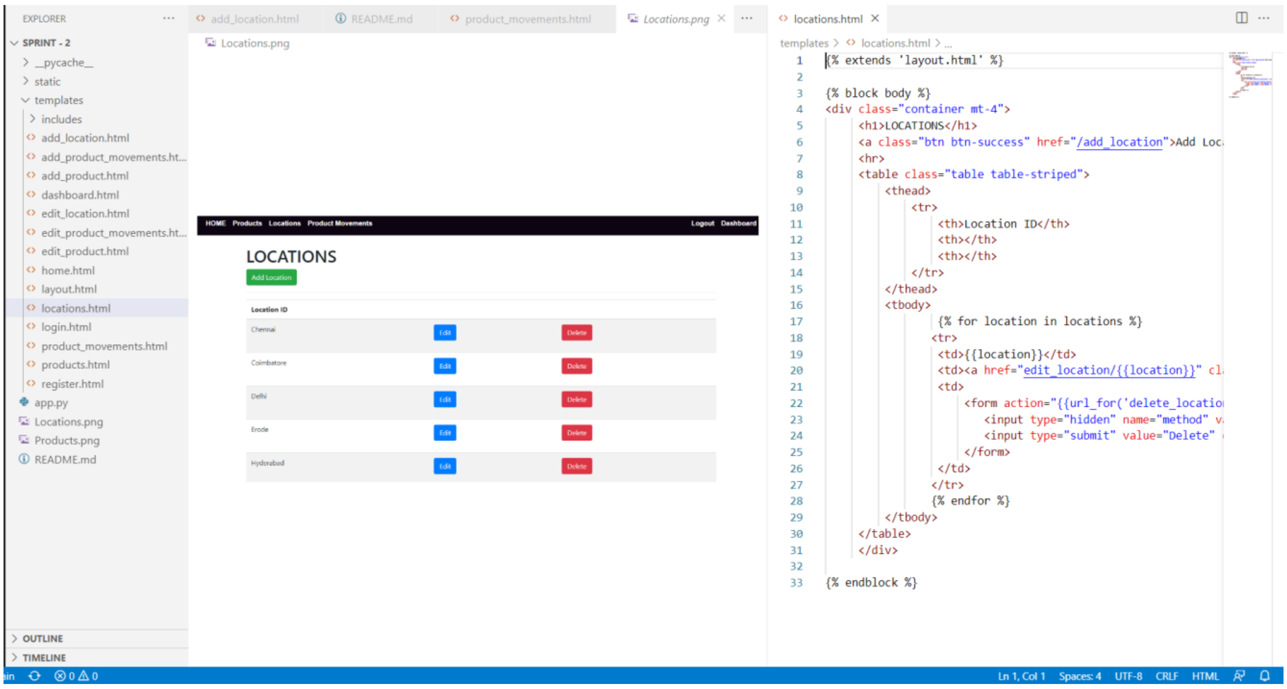


```
{% extends 'layout.html' %}

{% block body %}

{% from "includes/_formhelpers.html" import render_field %}

<div class="container mt-3">
  <div class="jumbotron ">
      <center>
      <h1 class="col-xs-offset-3 col-lg-offset-6">REGISTER USER</
      <form method="POST" action="">
          <div class="form-group col-lg-offset-4 col-lg-6">
            {{render_field(form.name, class_="form-control")}}
          </div>
          <div class="form-group col-lg-offset-4 col-lg-6">
              {{render_field(form.email, class_="form-control")}
          </div>
          <div class="form-group col-lg-offset-4 col-lg-6">
            {{render_field(form.username, class_="form-control")
          </div>
          <div class="form-group col-lg-offset-4 col-lg-6">
              {{render_field(form.password, class_="form-control'
          </div>
          <div class="form-group col-lg-offset-4 col-lg-6">
              {{render_field(form.confirm, class_="form-control"
          </div><br>
          <p><button type="submit" class="btn btn-primary" value=
          <b style="font-size: 20px;color: brown;"> Already ha
      </form>
</center>
</div></div>


{% endblock %}
```

Frontend of the Registration Page

Sprint 2 – Frontend:

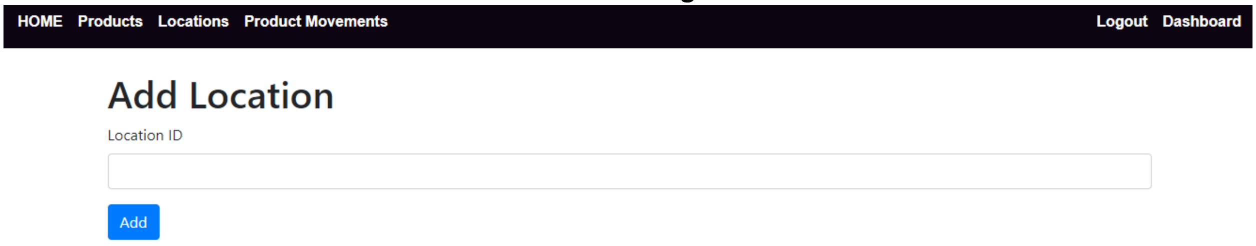
- Code the backend part to link the frontend pages created in Sprint 1.
- Create the main pages and functionalities of the application - Products Page, Locations Page, Product Movements Page and Dashboard.



Location ID		
Chennai	Edit	Delete
Coimbatore	Edit	Delete
Delhi	Edit	Delete
Erode	Edit	Delete
Hyderabad	Edit	Delete

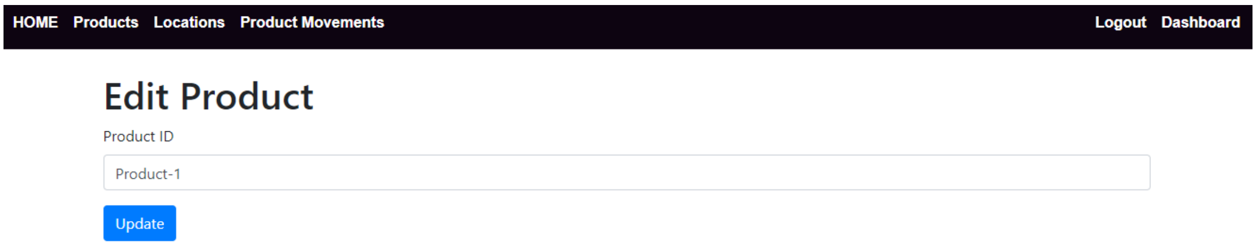
```
{% extends 'layout.html' %}

{% block body %}
<div class="container mt-4">
    <h1>LOCATIONS</h1>
    <a class="btn btn-success" href="/add_location">Add Loc
    <hr>
    <table class="table table-striped">
        <thead>
            <tr>
                <th>Location ID</th>
                <th></th>
                <th></th>
            </tr>
        </thead>
        <tbody>
                {% for location in locations %}
                <tr>
                <td>{{location}}</td>
                <td><a href="edit_location/{{location}}" cl
                <td>
                    <form action="{{url_for('delete_locatio
                        <input type="hidden" name="method" v
                        <input type="submit" value="Delete" 
                    </form>
                </td>
                </tr>
                {% endfor %}
        </tbody>
    </table>
    </div>

{% endblock %}
```

Location Page



Add Location

Location ID

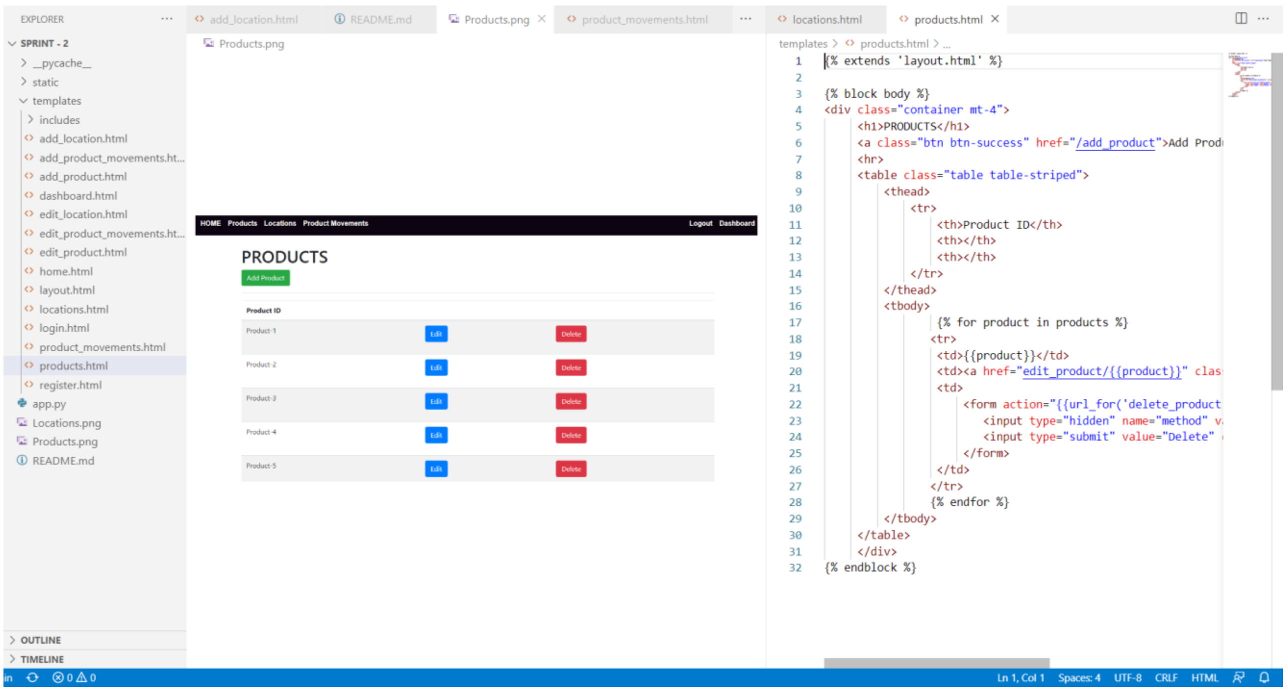


Edit Product

Product ID

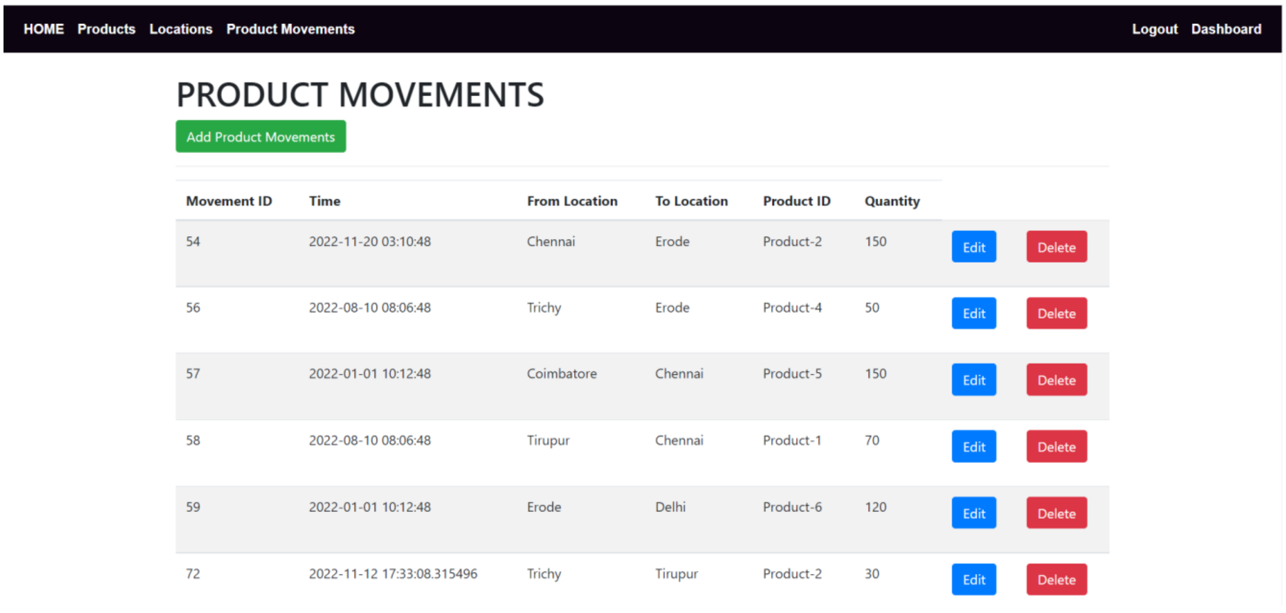
Product-1

To add and edit product



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

{% block body %}
<div class="container mt-4">
    <h1>PRODUCTS</h1>
    <a class="btn btn-success" href="/add_product">Add Prod
    <hr>
    <table class="table table-striped">
        <thead>
            <tr>
                <th>Product ID</th>
                <th></th>
                <th></th>
            </tr>
        </thead>
        <tbody>
                {% for product in products %}
              <tr>
                <td>{{product}}</td>
                <td><a href="edit_product/{{product}}" clas
                <td>
                    <form action="{{url_for('delete_product
                       <input type="hidden" name="method" v
                       <input type="submit" value="Delete"
                    </form>
                </td>
              </tr>
              {% endfor %}
        </tbody>
    </table>
    </div>
{% endblock %}
```

Products Page



PRODUCT MOVEMENTS

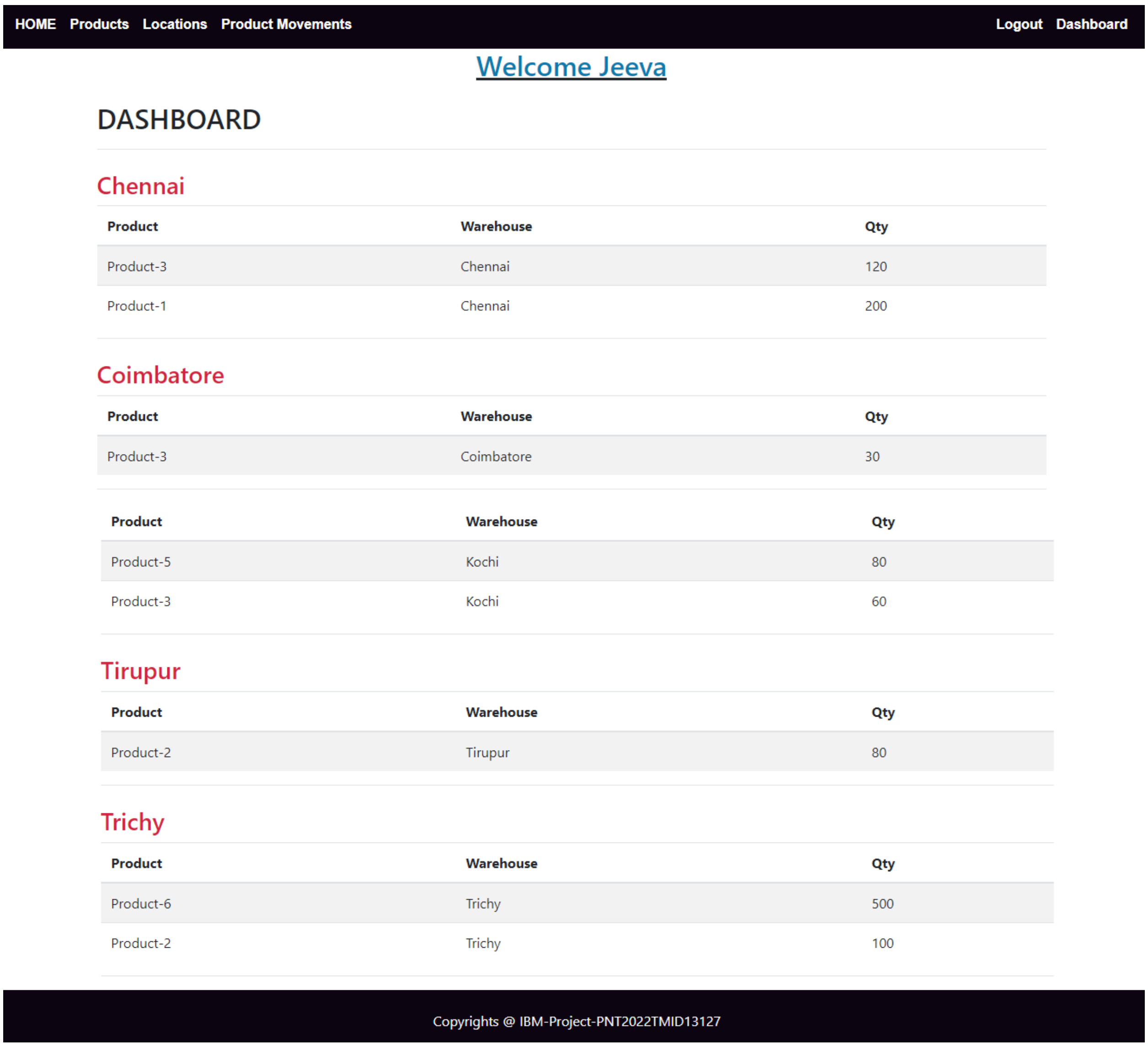
Movement ID	Time	From Location	To Location	Product ID	Quantity		
54	2022-11-20 03:10:48	Chennai	Erode	Product-2	150	Edit	Delete
56	2022-08-10 08:06:48	Trichy	Erode	Product-4	50	Edit	Delete
57	2022-01-01 10:12:48	Coimbatore	Chennai	Product-5	150	Edit	Delete
58	2022-08-10 08:06:48	Tirupur	Chennai	Product-1	70	Edit	Delete
59	2022-01-01 10:12:48	Erode	Delhi	Product-6	120	Edit	Delete
72	2022-11-12 17:33:08.315496	Trichy	Tirupur	Product-2	30	Edit	Delete

Product Movements Page

Sprint 3 - IBM Cloud Integration + Integration of SendGrid:

- Connect the pages with the cloud database
- Update Stocks in the dashboard when product movement occurs
- Integrating SendGrid Service
- Using Sendgrid to send mail to user if the stocks are less than the limit.

Dashboard



Welcome Jeeva

DASHBOARD

Chennai

Product	Warehouse	Qty
Product-3	Chennai	120
Product-1	Chennai	200

Coimbatore

Product	Warehouse	Qty
Product-3	Coimbatore	30

Product	Warehouse	Qty
Product-5	Kochi	80
Product-3	Kochi	60

Tirupur

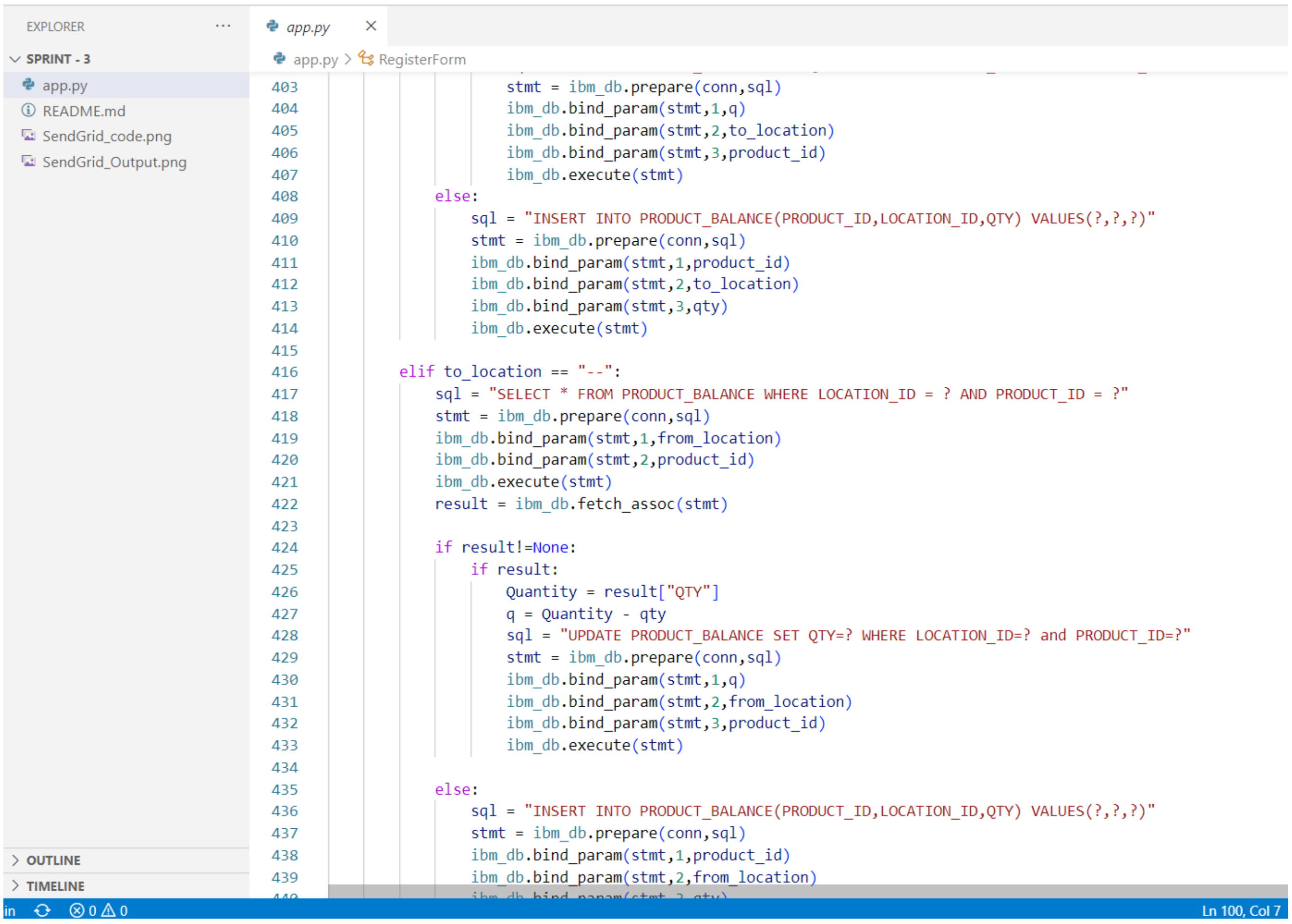
Product	Warehouse	Qty
Product-2	Tirupur	80

Trichy

Product	Warehouse	Qty
Product-6	Trichy	500
Product-2	Trichy	100

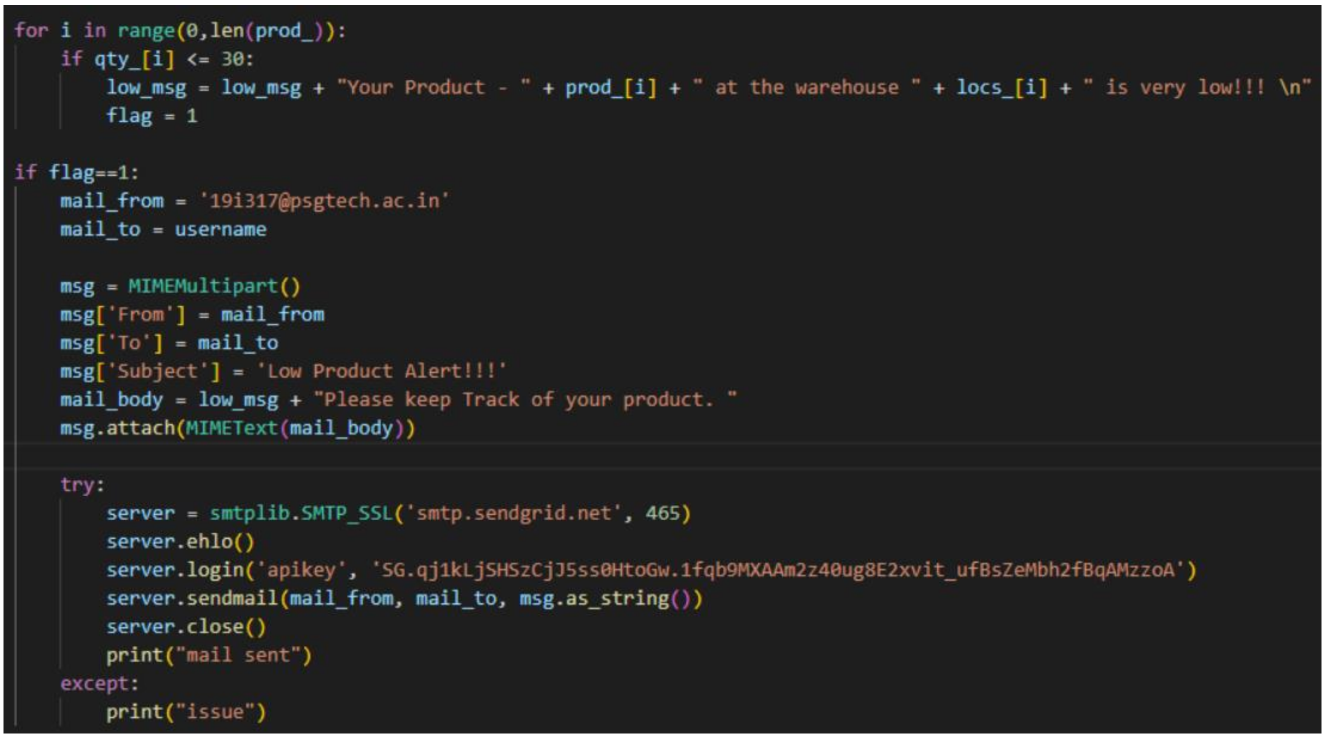
Copyrights @ IBM-Project-PNT2022TMID13127

Update stock in Dashboard



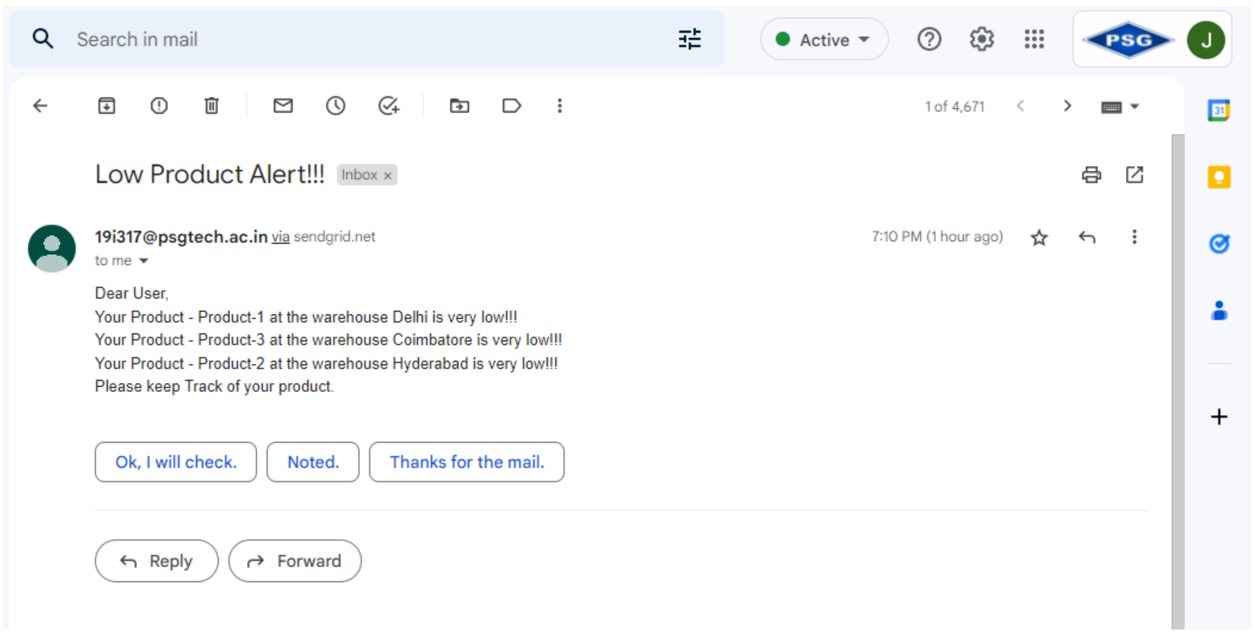
```
            stmt = ibm_db.prepare(conn,sql)
            ibm_db.bind_param(stmt,1,q)
            ibm_db.bind_param(stmt,2,to_location)
            ibm_db.bind_param(stmt,3,product_id)
            ibm_db.execute(stmt)
    else:
        sql = "INSERT INTO PRODUCT_BALANCE(PRODUCT_ID,LOCATION_ID,QTY) VALUES(?,?,?)"
        stmt = ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,product_id)
        ibm_db.bind_param(stmt,2,to_location)
        ibm_db.bind_param(stmt,3,qty)
        ibm_db.execute(stmt)

elif to_location == "--":
    sql = "SELECT * FROM PRODUCT_BALANCE WHERE LOCATION_ID = ? AND PRODUCT_ID = ?"
    stmt = ibm_db.prepare(conn,sql)
    ibm_db.bind_param(stmt,1,from_location)
    ibm_db.bind_param(stmt,2,product_id)
    ibm_db.execute(stmt)
    result = ibm_db.fetch_assoc(stmt)

    if result!=None:
        if result:
            Quantity = result["QTY"]
            q = Quantity - qty
            sql = "UPDATE PRODUCT_BALANCE SET QTY=? WHERE LOCATION_ID=? and PRODUCT_ID=?"
            stmt = ibm_db.prepare(conn,sql)
            ibm_db.bind_param(stmt,1,q)
            ibm_db.bind_param(stmt,2,from_location)
            ibm_db.bind_param(stmt,3,product_id)
            ibm_db.execute(stmt)

    else:
        sql = "INSERT INTO PRODUCT_BALANCE(PRODUCT_ID,LOCATION_ID,QTY) VALUES(?,?,?)"
        stmt = ibm_db.prepare(conn,sql)
        ibm_db.bind_param(stmt,1,product_id)
        ibm_db.bind_param(stmt,2,from_location)
```

Code for email alert:



```
for i in range(0,len(prod_)):
    if qty_[i] <= 30:
        low_msg = low_msg + "Your Product - " + prod_[i] + " at the warehouse " + locs_[i] + " is very low!!! \n"
        flag = 1

if flag==1:
    mail_from = '19i317@psgtech.ac.in'
    mail_to = username

    msg = MIMEMultipart()
    msg['From'] = mail_from
    msg['To'] = mail_to
    msg['Subject'] = 'Low Product Alert!!!'
    mail_body = low_msg + "Please keep Track of your product. "
    msg.attach(MIMEText(mail_body))

    try:
        server = smtplib.SMTP_SSL('smtp.sendgrid.net', 465)
        server.ehlo()
        server.login('apikey', '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')
        server.sendmail(mail_from, mail_to, msg.as_string())
        server.close()
        print("mail sent")
    except:
        print("issue")
```

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



Low Product Alert!!! Inbox

19i317@psgtech.ac.in via sendgrid.net
to me

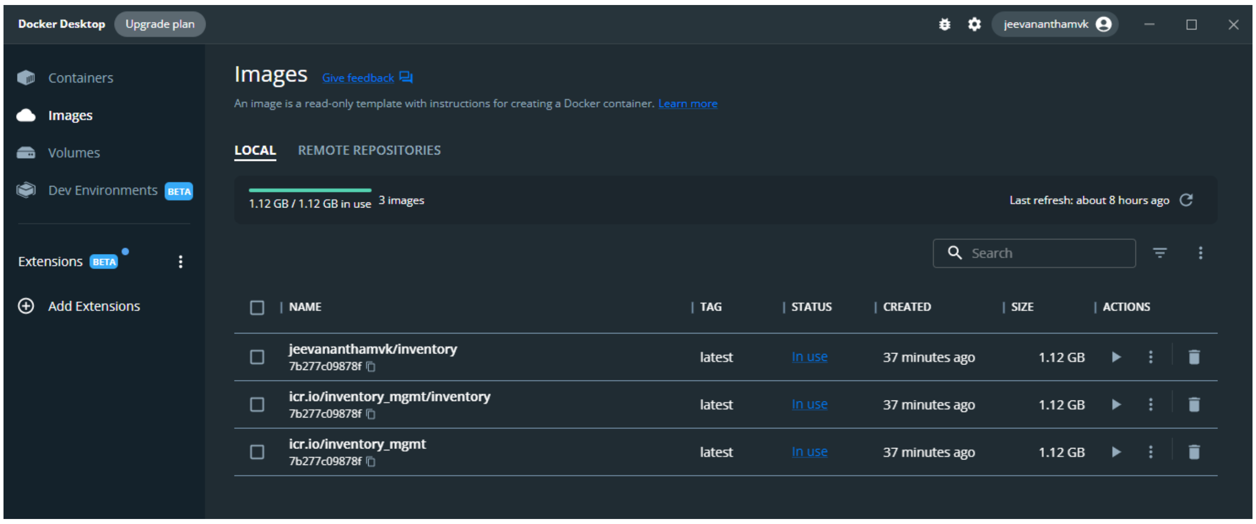
7:10 PM (1 hour ago)

Dear User,
Your Product - Product-1 at the warehouse Delhi is very low!!!
Your Product - Product-3 at the warehouse Coimbatore is very low!!!
Your Product - Product-2 at the warehouse Hyderabad is very low!!!
Please keep Track of your product.

Sprint 4 - Deploying the application using Docker and Kubernetes:

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

- Building an image for our project.



Images

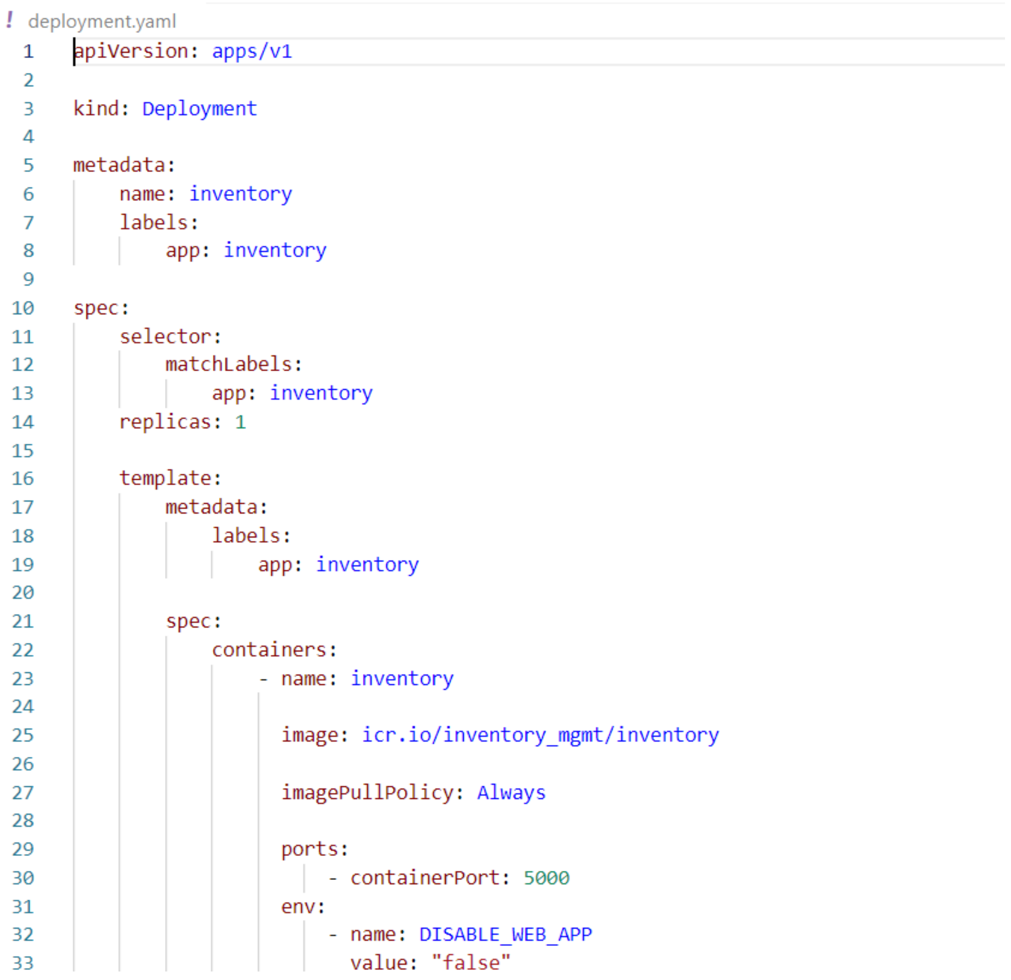
An image is a read-only template with instructions for creating a Docker container. Learn more

1.12 GB / 1.12 GB in use 3 images

Last refresh: about 8 hours ago

NAME	TAG	STATUS	CREATED	SIZE	ACTIONS
jeevananthamvk/inventory 7b277c09878f	latest	In use	37 minutes ago	1.12 GB	
icr.io/inventory_mgmt/inventory 7b277c09878f	latest	In use	37 minutes ago	1.12 GB	
icr.io/inventory_mgmt 7b277c09878f	latest	In use	37 minutes ago	1.12 GB	

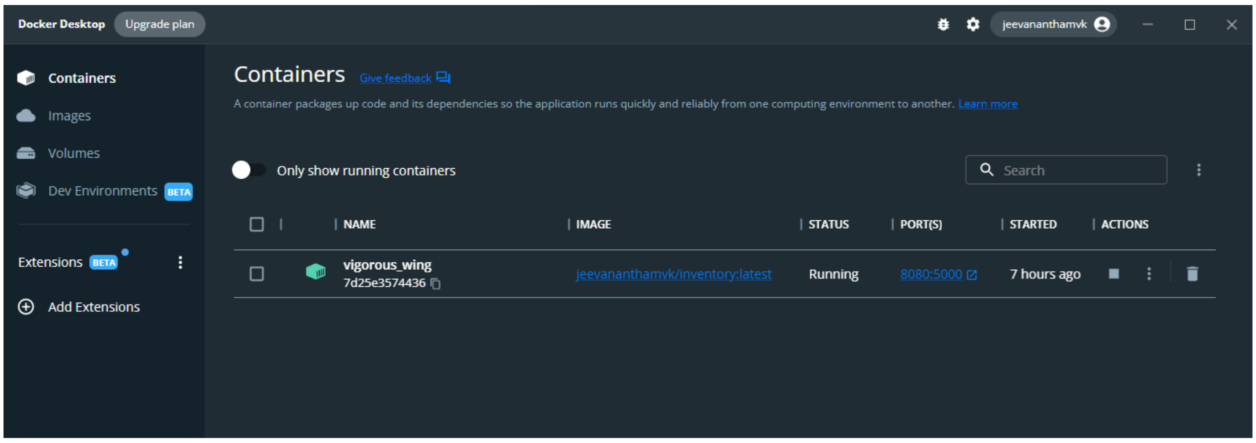
- Create a valid Deployment.yaml file.



deployment.yaml

```yaml
apiVersion: apps/v1

kind: Deployment

metadata:
    name: inventory
    labels:
        app: inventory

spec:
    selector:
        matchLabels:
            app: inventory
    replicas: 1

    template:
        metadata:
            labels:
                app: inventory

        spec:
            containers:
                - name: inventory

                  image: icr.io/inventory_mgmt/inventory

                  imagePullPolicy: Always

                  ports:
                    - containerPort: 5000
                  env:
                    - name: DISABLE_WEB_APP
                      value: "false"
```

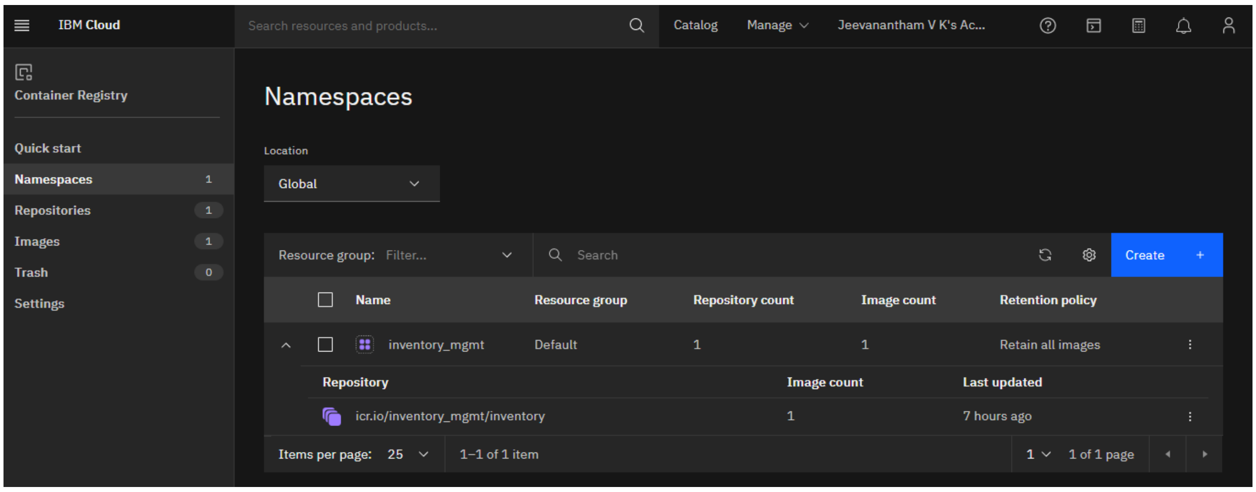
- Create a namespace in IBM Container registry and Push the project into IBM container Registry.



Containers

A container packages up code and its dependencies so the application runs quickly and reliably from one computing environment to another. Learn more

NAME	IMAGE	STATUS	PORT(S)	STARTED	ACTIONS
vigorous_wing 7d25e3574436	jeevananthamvk/inventory:latest	Running	8080:5000	7 hours ago	



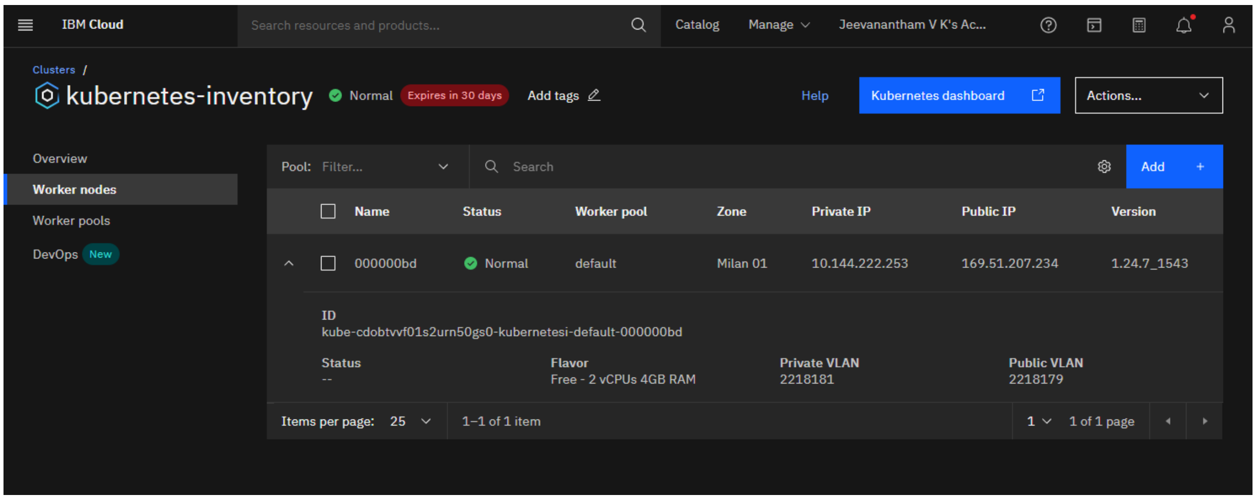
Namespaces

Name	Resource group	Repository count	Image count	Retention policy
inventory_mgmt	Default	1	1	Retain all images

Repository	Image count	Last updated
icr.io/inventory_mgmt/inventory	1	7 hours ago

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

- Create a Kubernetes Cluster in IBM Cloud and deploy work node. Then, Check for the public IP address in your IBM Kubernetes Cluster under Worker Node.



kubernetes-inventory Normal Expires in 30 days

Name	Status	Worker pool	Zone	Private IP	Public IP	Version
000000bd	Normal	default	Milan 01	10.144.222.253	169.51.207.234	1.24.7_1543

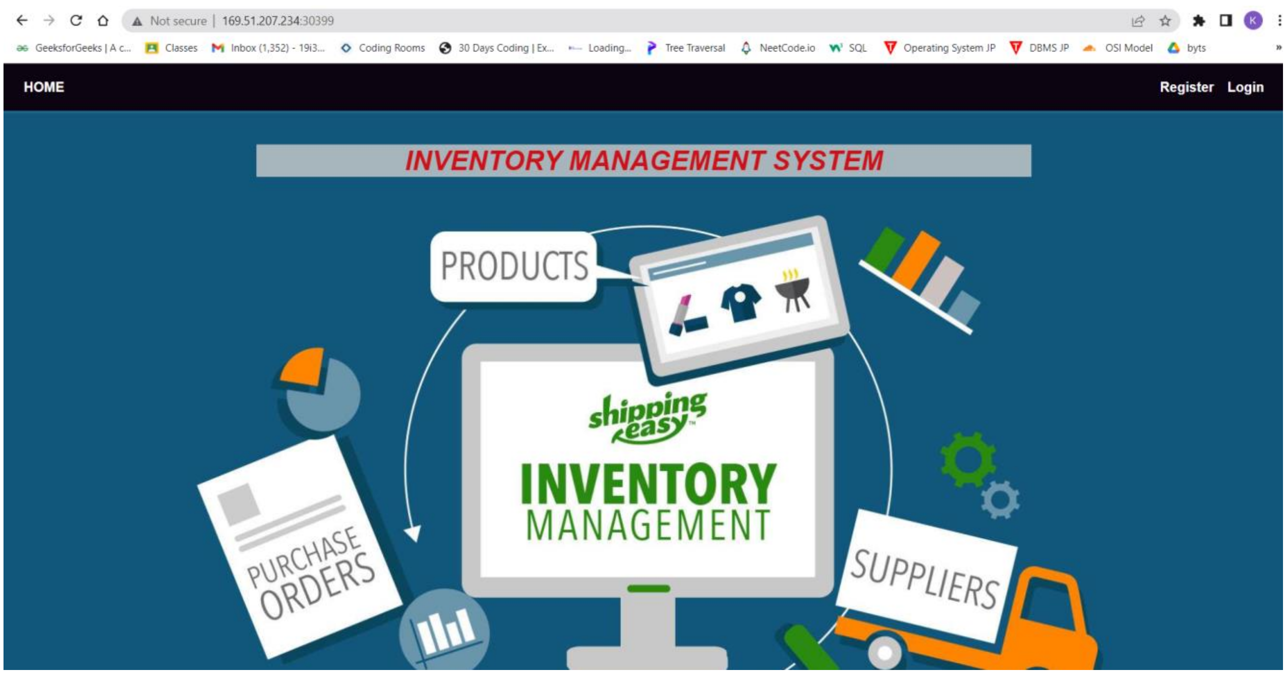
ID
kube-cdobtvvf01s2urn50gs0-kubernetesi-default-000000bd

Status	Flavor	Private VLAN	Public VLAN
--	Free - 2 vCPUs 4GB RAM	2218181	2218179

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

Thus, we have the Public IP address and the Nodeport.<Public_IP>:<NodePort> will help us to access Inventory management system application, i.e. **169.51.207.234:30399**

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).