Final Deliverables Report

Date	14.11.2022
Team ID	PNT2022TMID03188
Project Name	Inventory Management System for Retailers

Team members and their Contribution:

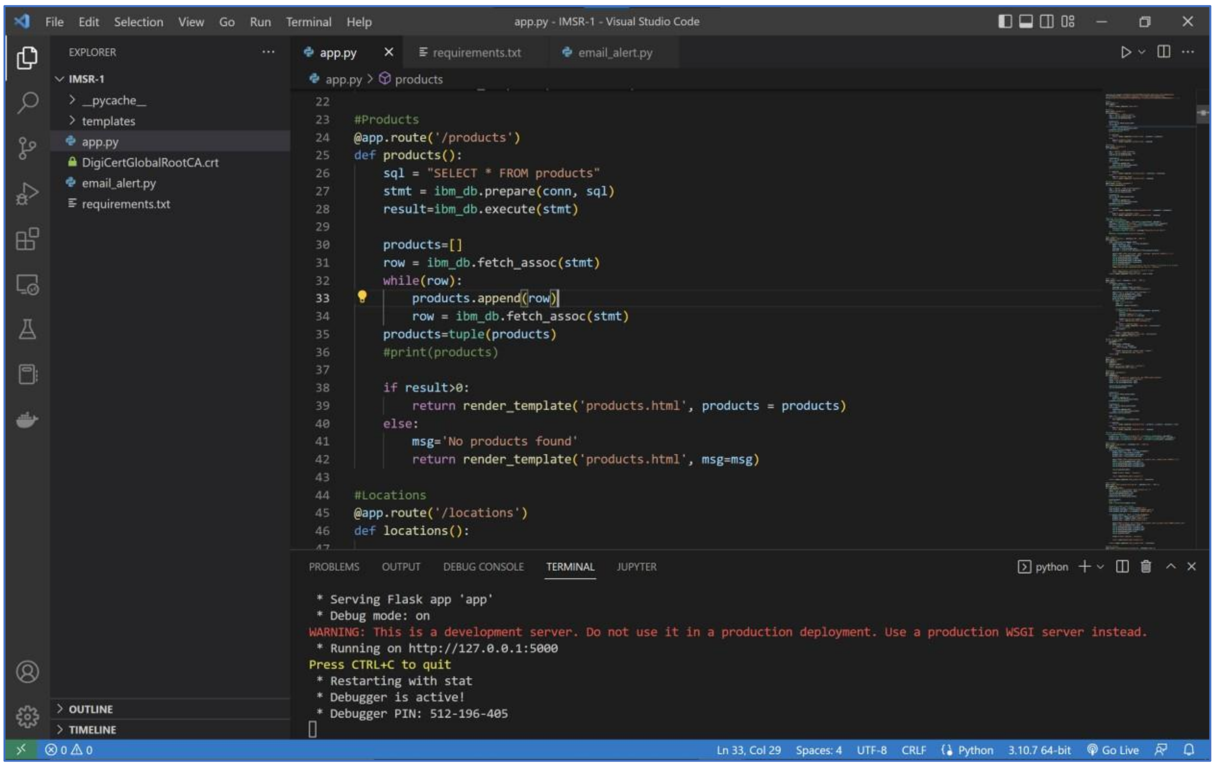
Name	**Roll no**	**Contribution**
SIDDARTH S	727719EUEC144	Frontend – 5 Pages, Integration of Sendgrid, Deployment of using docker and Kubernetes.
SRINITHI S	727719EUEC141	Frontend – 5 Pages, Documentation
SIBI CHAKRAVARTHY	727719EUEC142	Frontend – 4 Pages, Documentation
SIDDARTH G	727719EUEC143	Backend Fully (For all 14 Pages), Integration of IBM Cloud, Deployment of using docker and Kubernetes.

Introduction:

1. Sprint 1 – Backend
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:**

All the routes to each page and APIs are created.

Example, (For Products page)



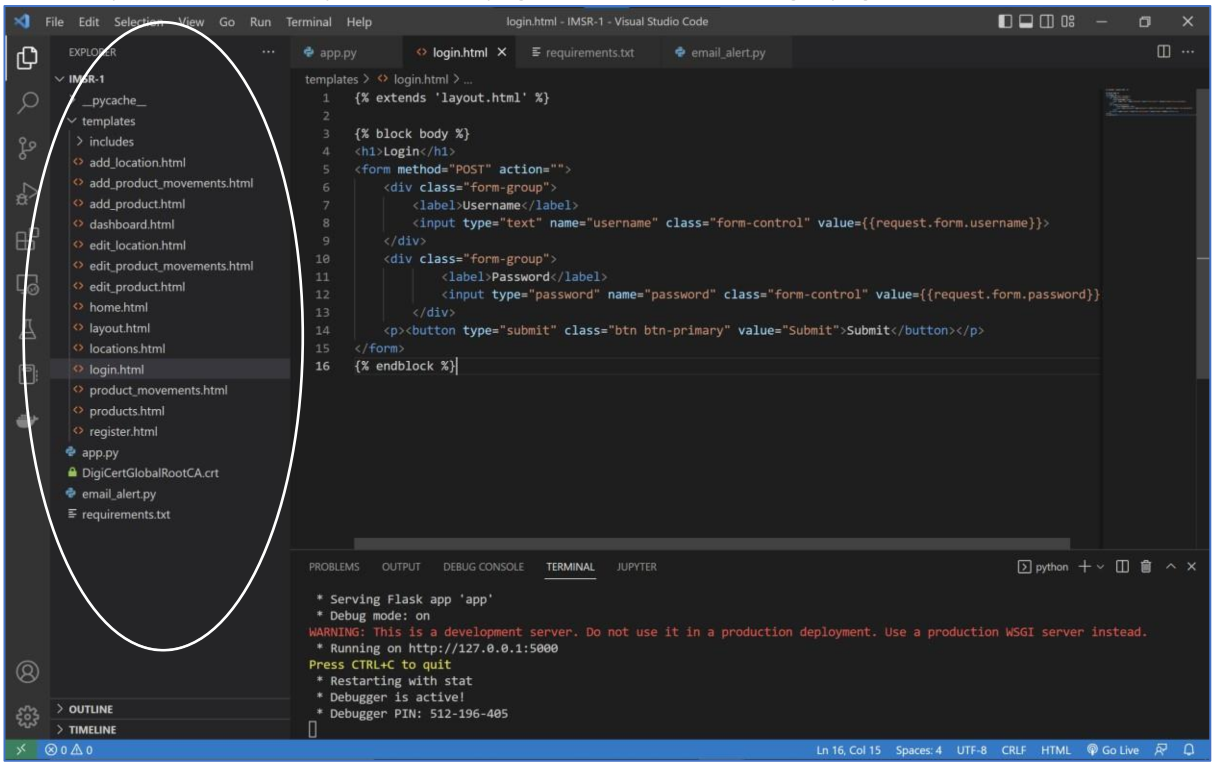
```python

#Products
@app.route('/products')
def products():
    sql = "SELECT * FROM products"
    stmt = ibm_db.prepare(conn, sql)
    result=ibm_db.execute(stmt)

    products=[]
    row = ibm_db.fetch_assoc(stmt)
    while(row):
        products.append(row)
        row = ibm_db.fetch_assoc(stmt)
    products=tuple(products)
    #print(products)

    if result>0:
        return render_template('products.html', products = products)
    else:
        msg='No products found'
        return render_template('products.html', msg=msg)

#Locations
@app.route('/locations')
def locations():
```

```
 * 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
```

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)

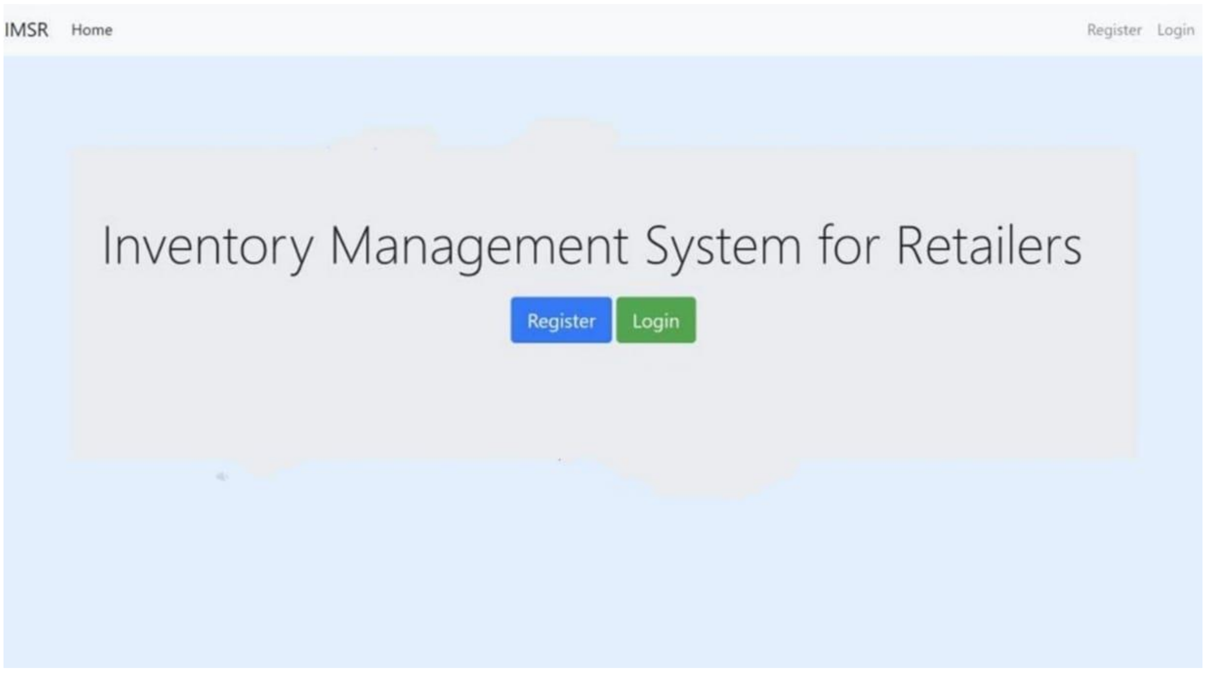


templates > login.html > ...

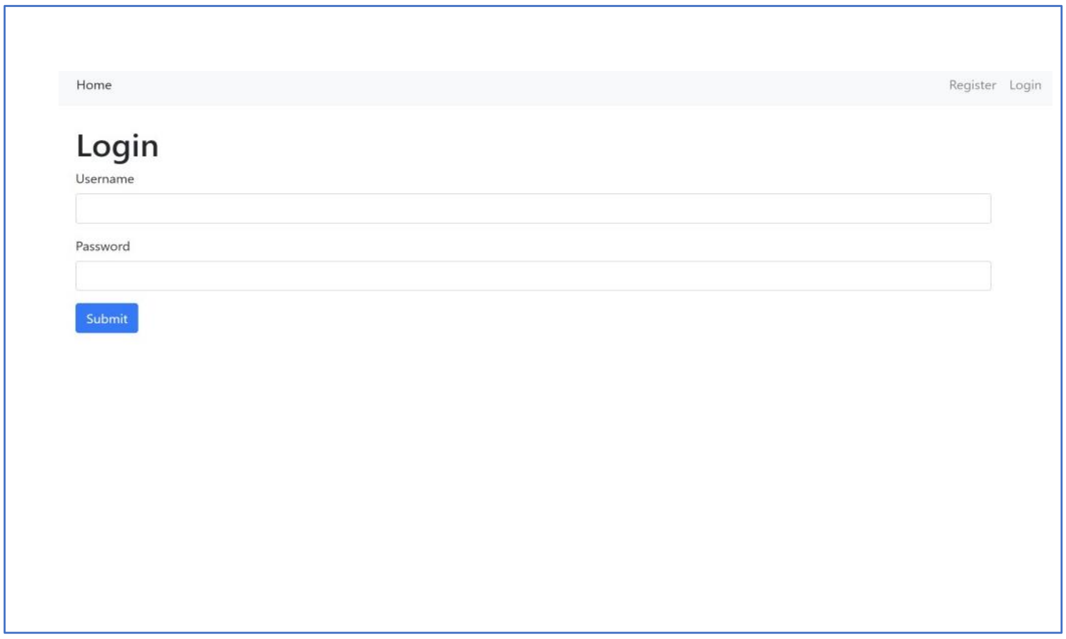
```html
{% extends 'layout.html' %}

{% block body %}
<h1>Login</h1>
<form method="POST" action="">
    <div class="form-group">
        <label>Username</label>
        <input type="text" name="username" class="form-control" value={{request.form.username}}>
    </div>
    <div class="form-group">
            <label>Password</label>
            <input type="password" name="password" class="form-control" value={{request.form.password}}
        </div>
    <p><button type="submit" class="btn btn-primary" value="Submit">Submit</button></p>
</form>
{% endblock %}
```

```
 * 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
```

Sample FrontEnd Pages,



Login Page,


Home Register Login

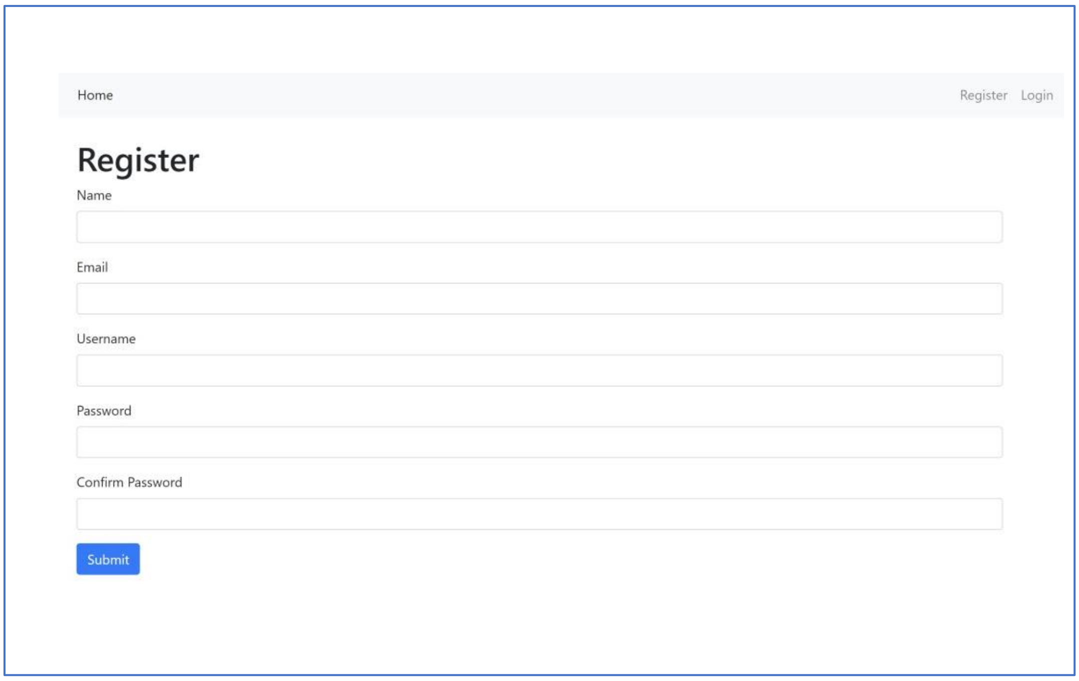
Login

Username

Password

Submit

Register Page,


Home Register Login

Register

Name

Email

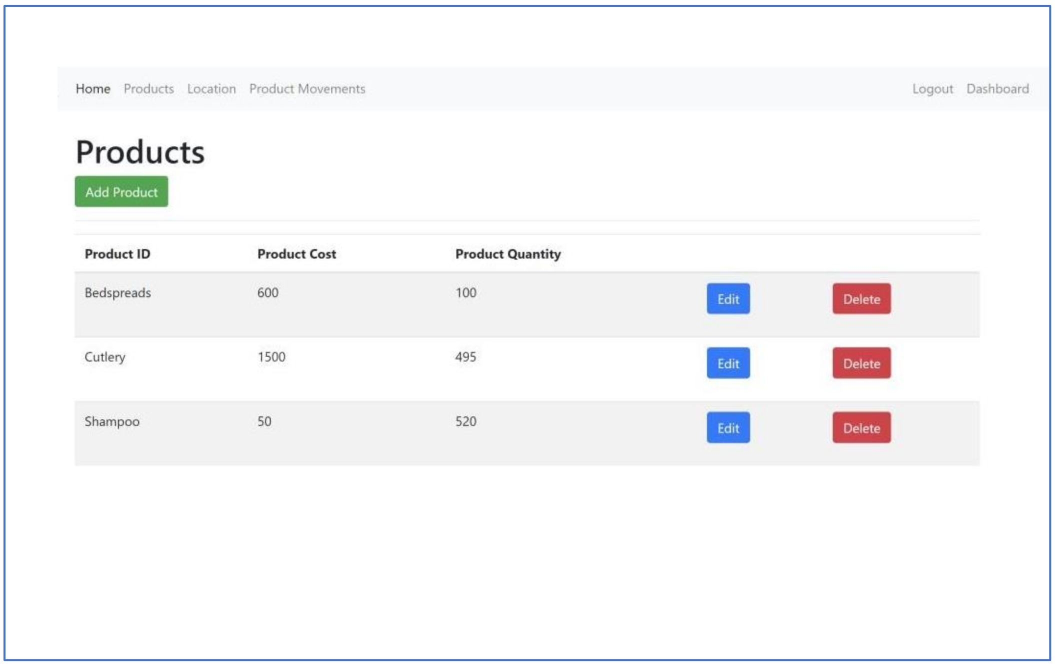
Username

Password

Confirm Password

Submit

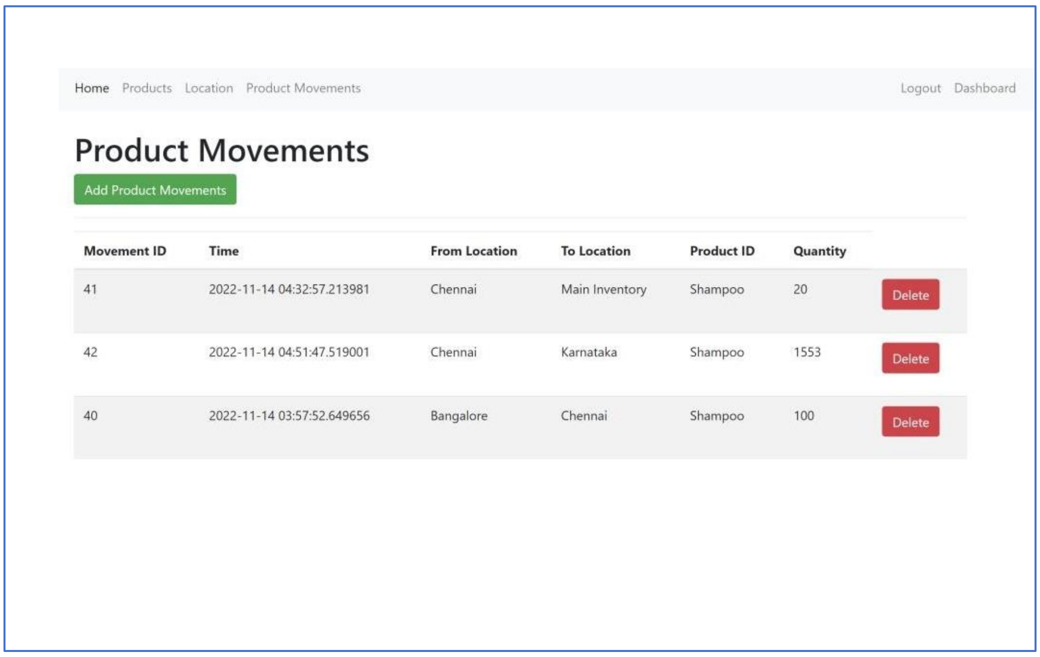
Products Page,



Products

Product ID	Product Cost	Product Quantity		
Bedspreads	600	100	Edit	Delete
Cutlery	1500	495	Edit	Delete
Shampoo	50	520	Edit	Delete

Product Movements Page,



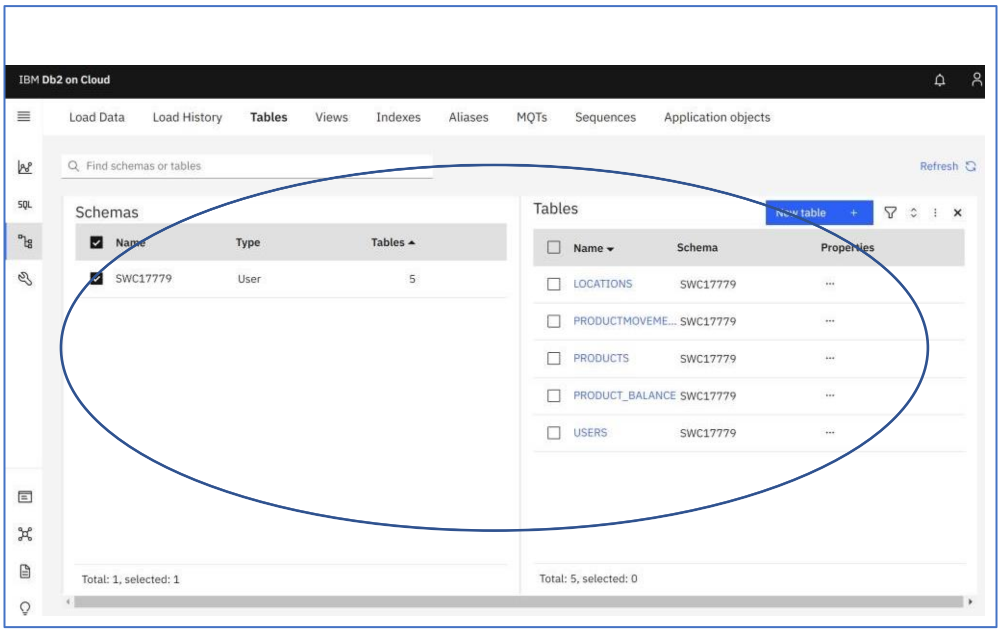
Product Movements

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,



Schemas

Name	Type	Tables
SWC17779	User	5

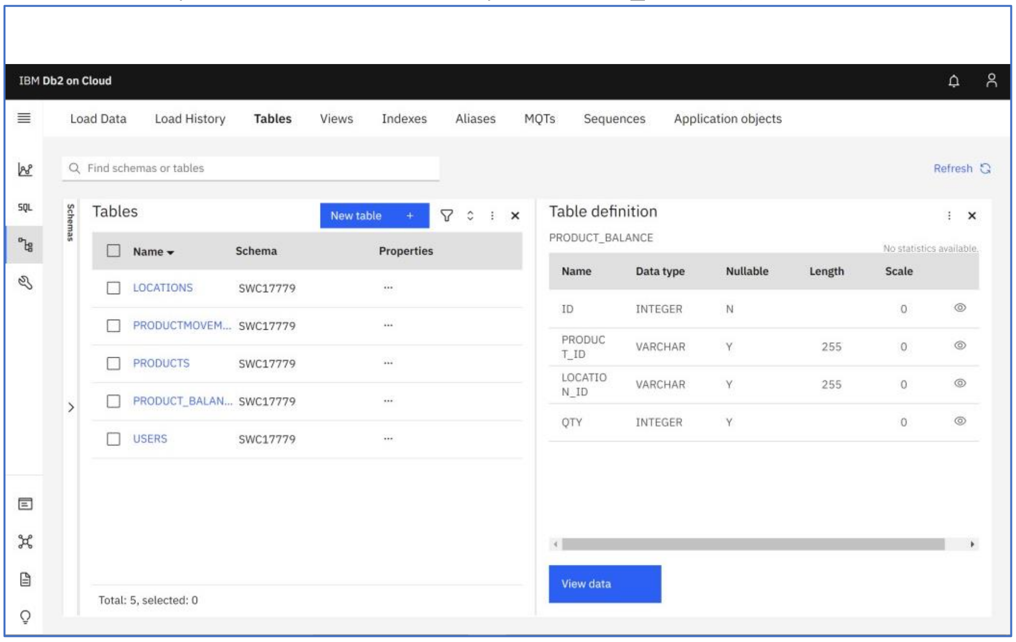
Total: 1, selected: 1

Tables

Name	Schema	Properties
LOCATIONS	SWC17779	···
PRODUCTMOVEME...	SWC17779	···
PRODUCTS	SWC17779	···
PRODUCT_BALANCE	SWC17779	···
USERS	SWC17779	···

Total: 5, selected: 0

Schema of the particular table (For Example, Product_Balance)



Tables

Name	Schema	Properties
LOCATIONS	SWC17779	···
PRODUCTMOVEM...	SWC17779	···
PRODUCTS	SWC17779	···
PRODUCT_BALAN...	SWC17779	···
USERS	SWC17779	···

Total: 5, selected: 0

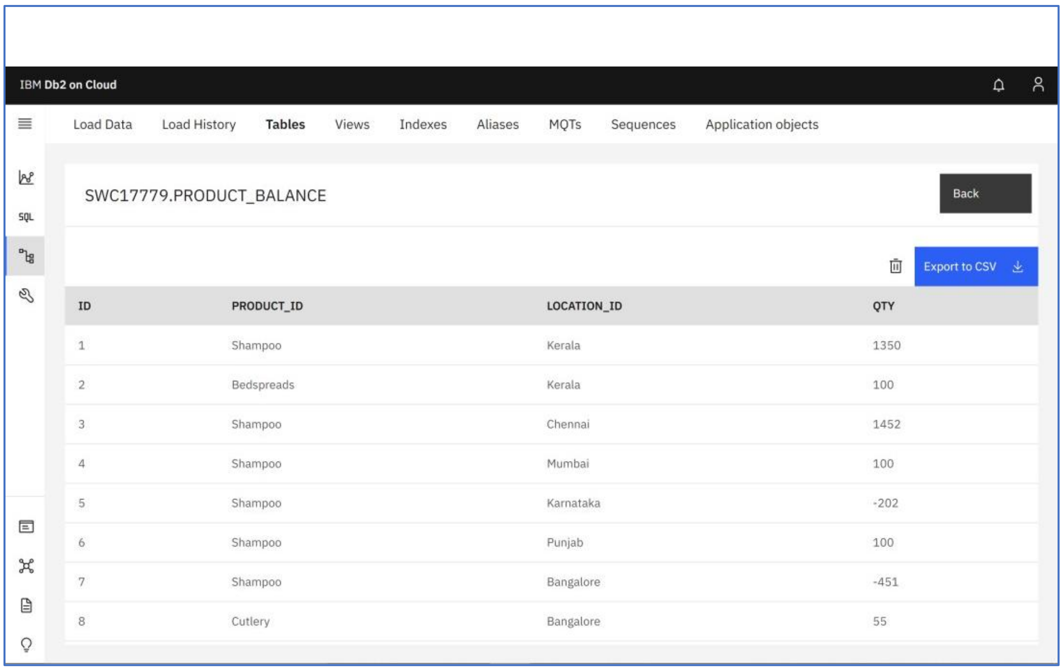
Table definition

PRODUCT_BALANCE

No statistics available.

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

Data of a particular table (For Example, Product_Balance)



SWC17779.PRODUCT_BALANCE

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

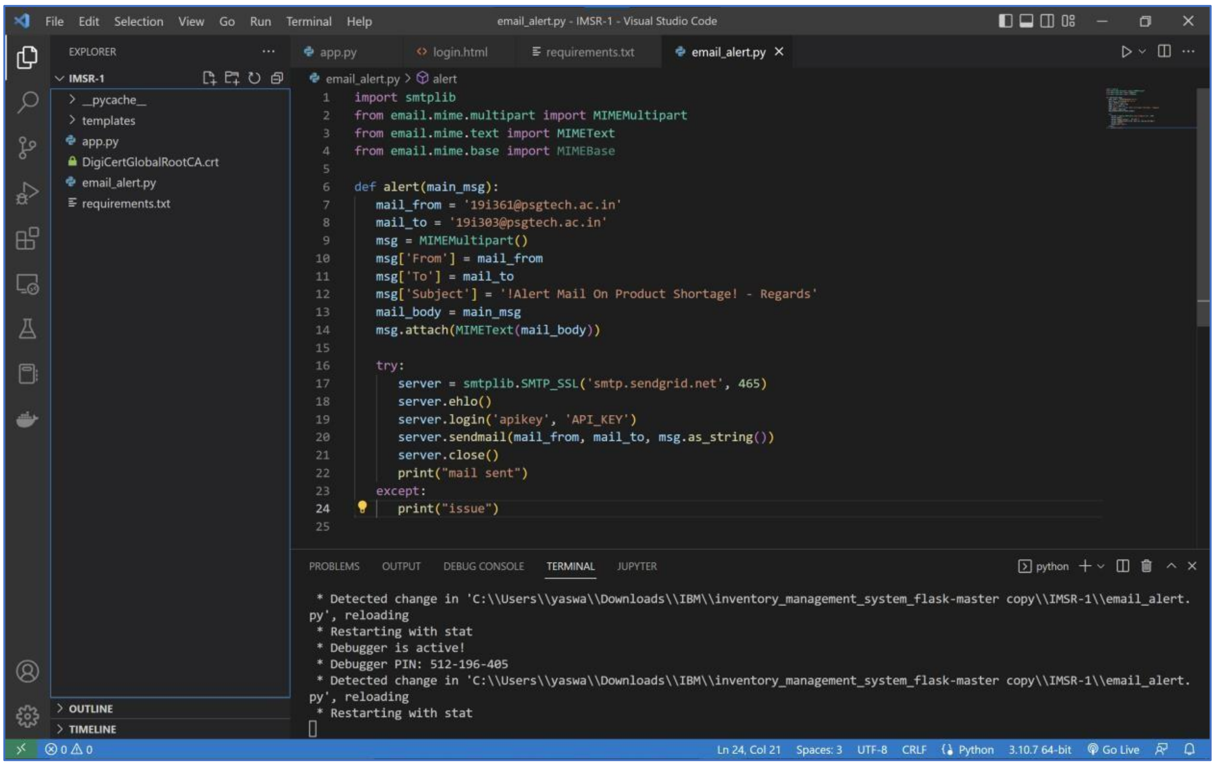
Code for Connection of IBM Database,

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

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

SendGrid Integration:

Code for email alert



```python
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.base import MIMEBase

def alert(main_msg):
   mail_from = '19i361@psgtech.ac.in'
   mail_to = '19i303@psgtech.ac.in'
   msg = MIMEMultipart()
   msg['From'] = mail_from
   msg['To'] = mail_to
   msg['Subject'] = '!Alert Mail On Product Shortage! - Regards'
   mail_body = main_msg
   msg.attach(MIMEText(mail_body))

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

```

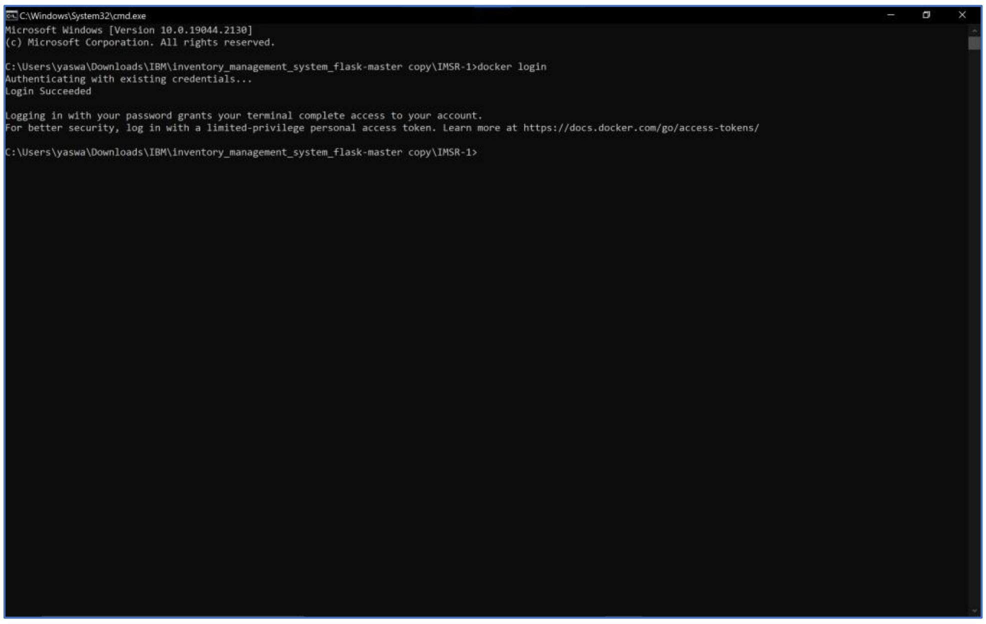
```
 * Detected change in 'C:\\Users\\yaswa\\Downloads\\IBM\\inventory_management_system_flask-master copy\\IMSR-1\\email_alert.
py', reloading
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 512-196-405
 * Detected change in 'C:\\Users\\yaswa\\Downloads\\IBM\\inventory_management_system_flask-master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

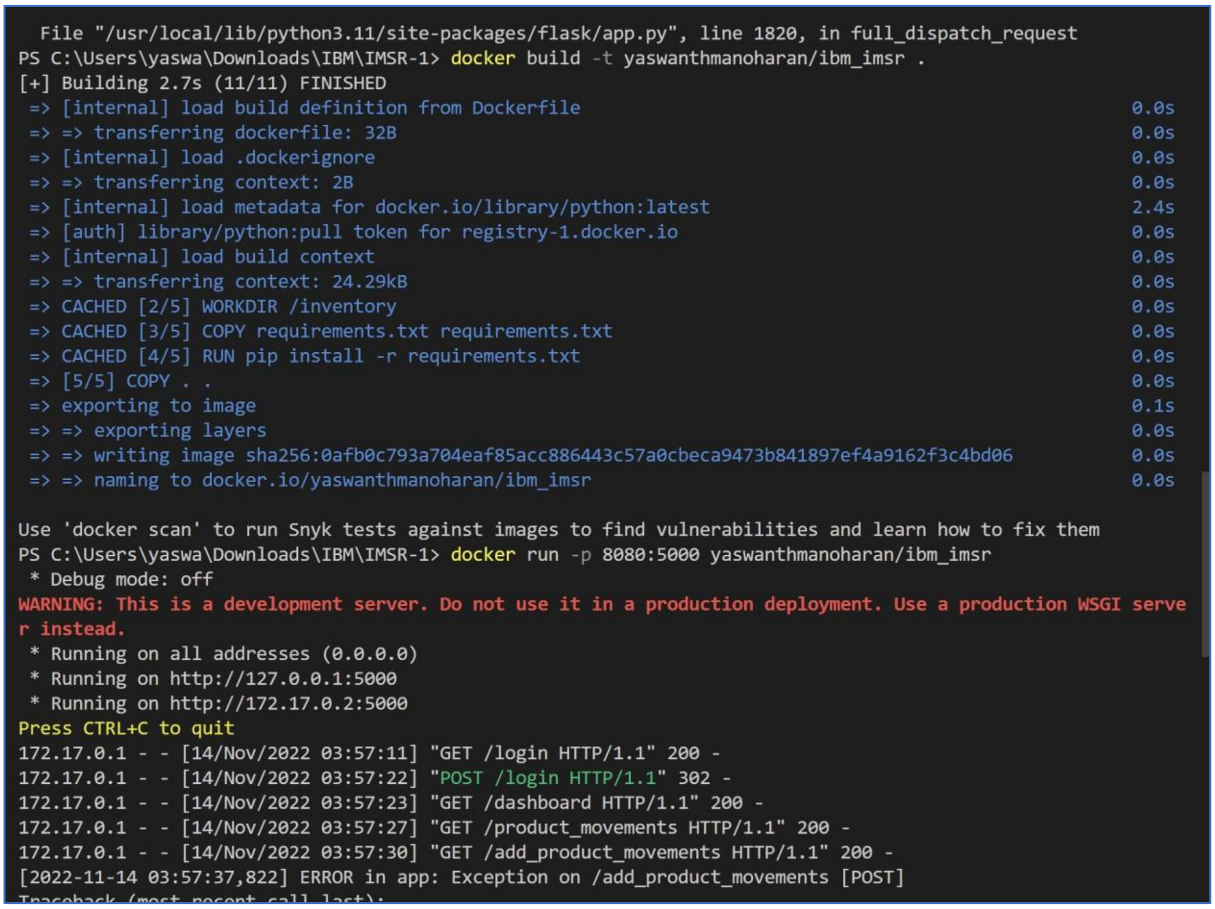
Note: Make sure to create a Dockerfile in the project folder.

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

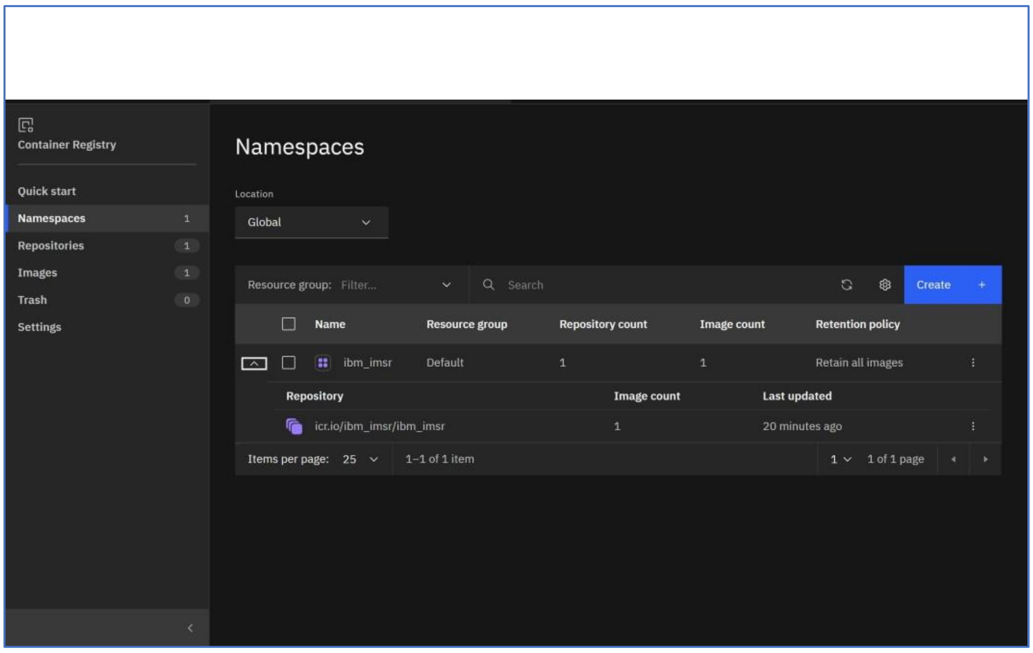


```
Microsoft Windows [Version 10.0.19044.2130]
(c) Microsoft Corporation. All rights reserved.

C:\Users\yaswa\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\yaswa\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):
```

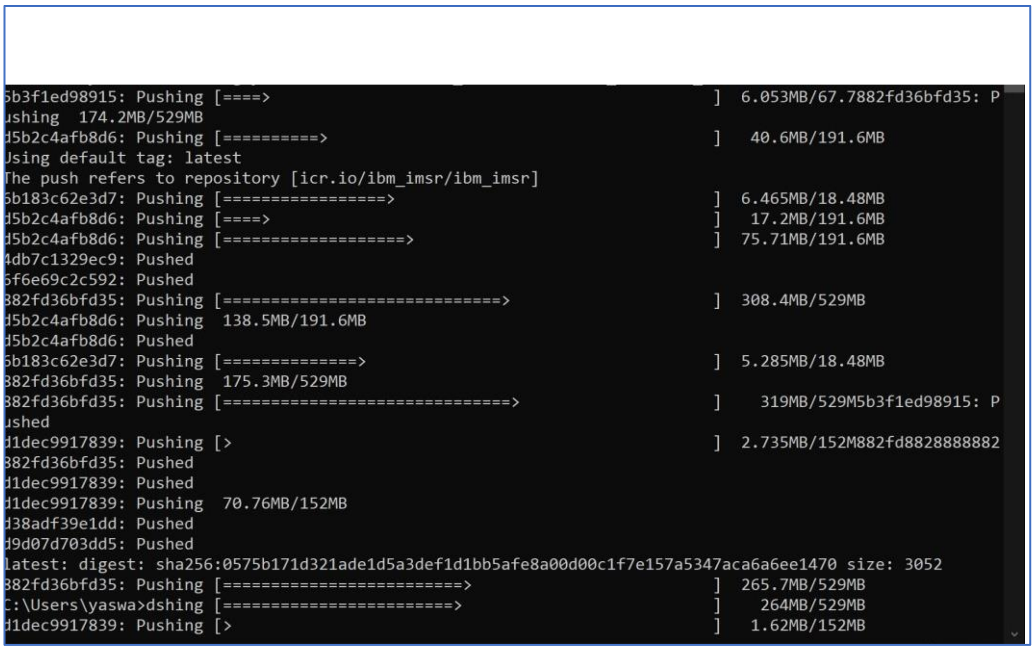


	Name	Resource group	Repository count	Image count	Retention policy
	ibm_imsr	Default	1	1	Retain all images

Repository	Image count	Last updated
icr.io/ibm_imsr/ibm_imsr	1	20 minutes ago

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

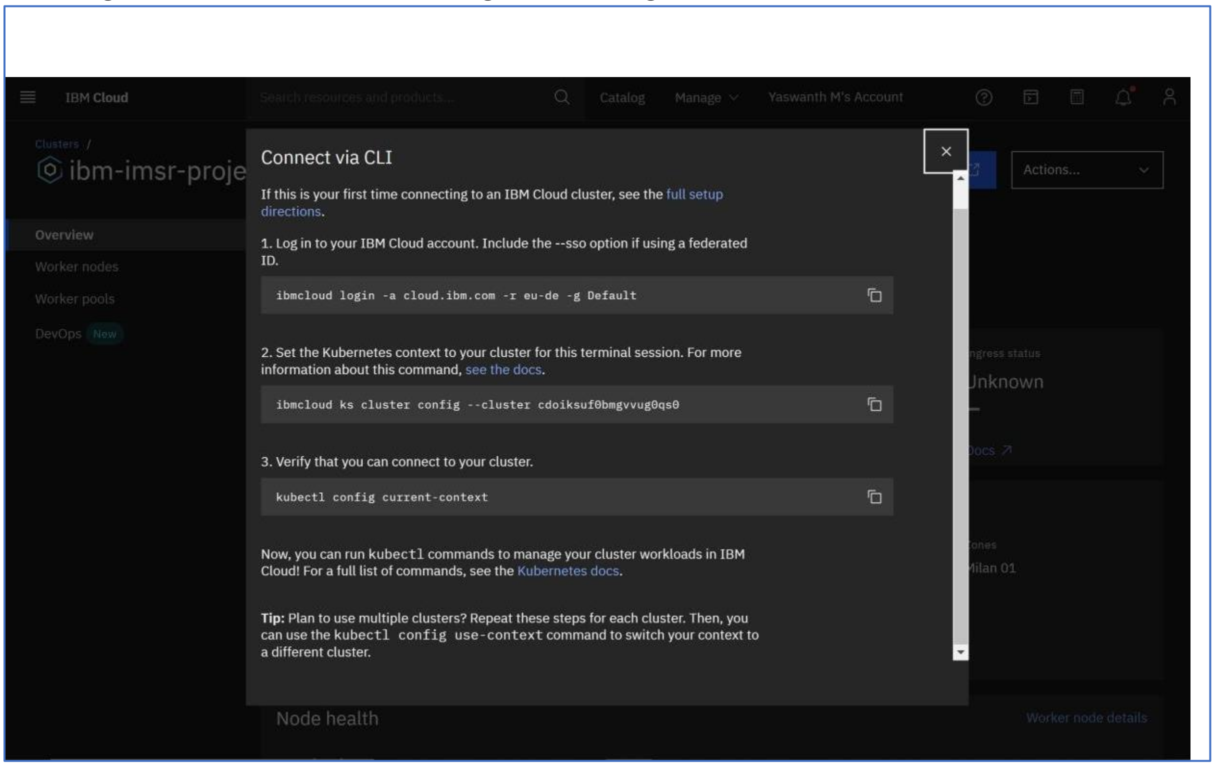
Pushing the project into IBM container Registry,



```
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_imsr/ibm_imsr]
5b183c62e3d7: Pushing [=================>                             ]  6.465MB/18.48MB
d5b2c4afb8d6: Pushing [====>                                          ]   17.2MB/191.6MB
d5b2c4afb8d6: Pushing [===================>                           ]  75.71MB/191.6MB
4db7c1329ec9: Pushed
5f6e69c2c592: Pushed
382fd36bfd35: Pushing [=============================>                 ]  308.4MB/529MB
d5b2c4afb8d6: Pushing  138.5MB/191.6MB
d5b2c4afb8d6: Pushed
5b183c62e3d7: Pushing [==============>                                ]  5.285MB/18.48MB
382fd36bfd35: Pushing  175.3MB/529MB
382fd36bfd35: Pushing [==============================>                ]    319MB/529M5b3f1ed98915: P
ushed
d1dec9917839: Pushing [>                                              ]  2.735MB/152M882fd8828888882
382fd36bfd35: Pushed
d1dec9917839: Pushed
d1dec9917839: Pushing  70.76MB/152MB
d38adf39e1dd: Pushed
d9d07d703dd5: Pushed
latest: digest: sha256:0575b171d321ade1d5a3def1d1bb5afe8a00d00c1f7e157a5347aca6a6ee1470 size: 3052
382fd36bfd35: Pushing [=========================>                     ]  265.7MB/529MB
C:\Users\yaswa>dshing [========================>                      ]    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,



Connect via CLI

If this is your first time connecting to an IBM Cloud cluster, see the full setup directions.

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 cdoiksuf0bmgvvug0qs0
```

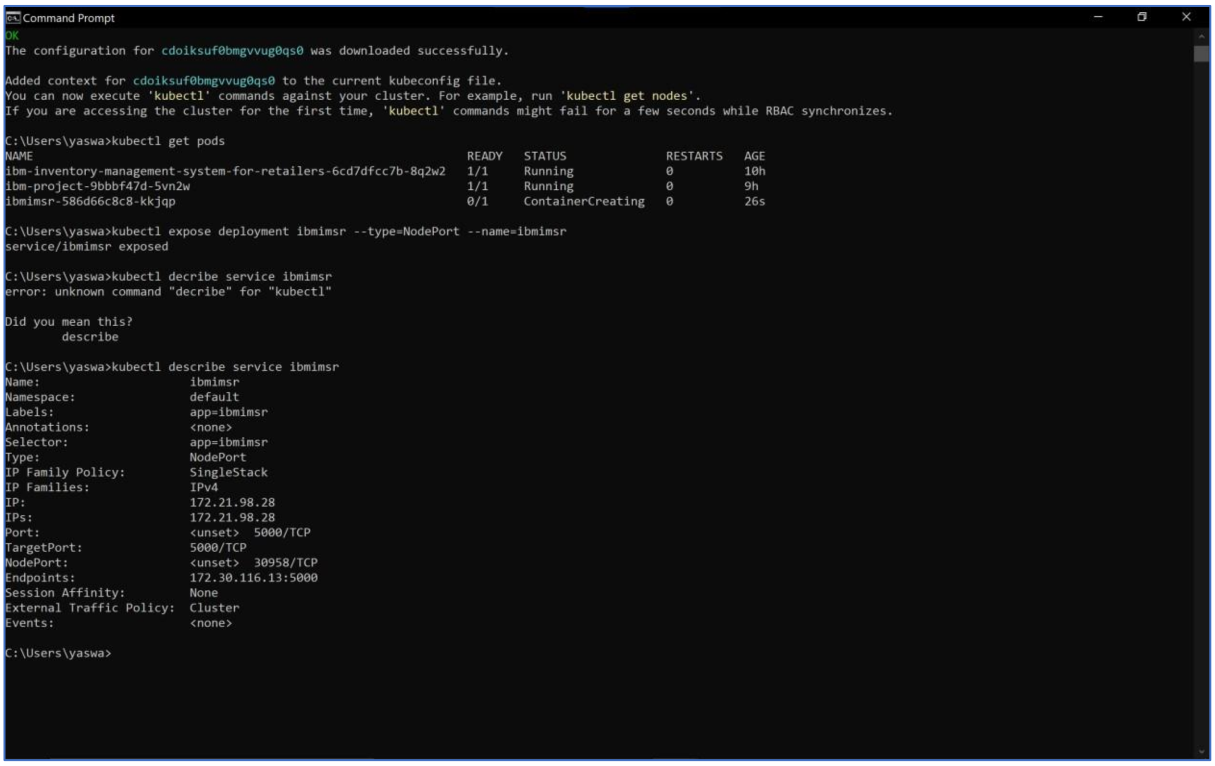
3. Verify that you can connect to your cluster.

```
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.

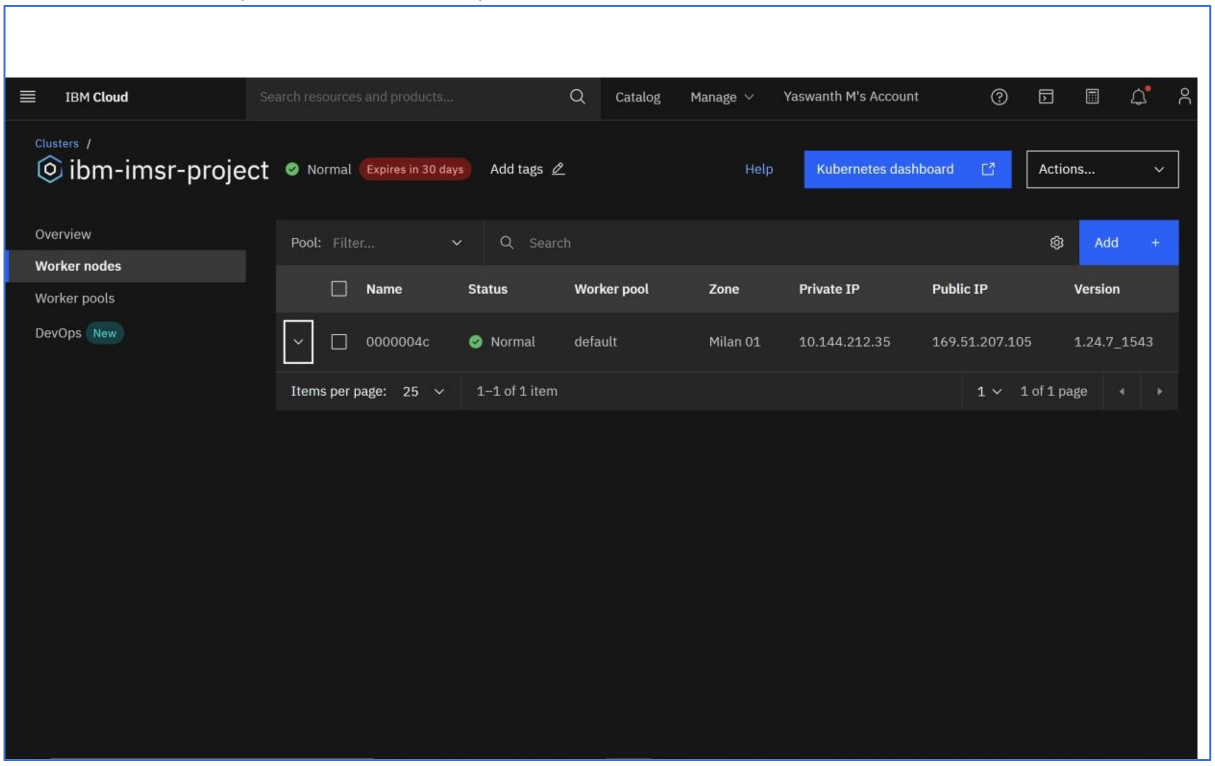
Tip: Plan to use multiple clusters? Repeat these steps for each cluster. Then, you can use the `kubectl config use-context` command to switch your context to a different cluster.

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



```
Command Prompt
OK
The configuration for cdoiksuf0bmgvvug0qs0 was downloaded successfully.

Added context for cdoiksuf0bmgvvug0qs0 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-9bbbf47d-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 decribe service ibmimsr
error: unknown command "decribe" 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:               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,



Name	Status	Worker pool	Zone	Private IP	Public IP	Version
0000004c	Normal	default	Milan 01	10.144.212.35	169.51.207.105	1.24.7_1543

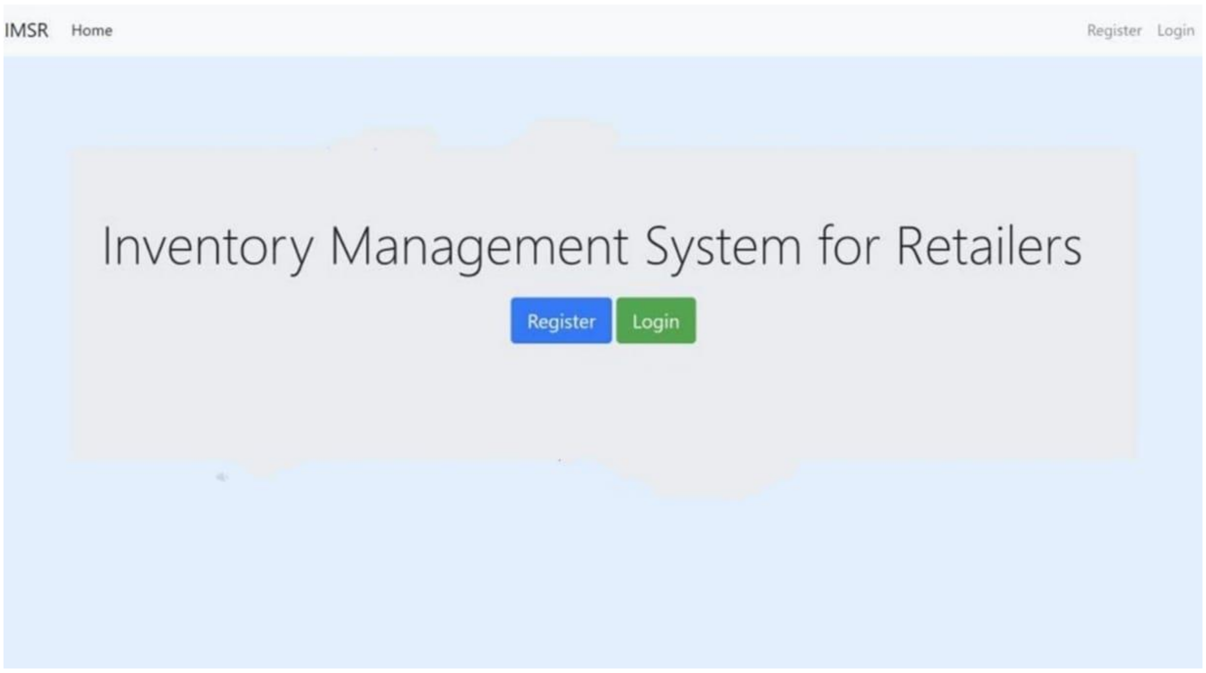
Items per page: 25 1–1 of 1 item 1 1 of 1 page

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!