Project Development Phase
Works Done on Each Sprints

Date	14.11.2022
Team ID	PNT2022TMID11899
Project Name	Inventory Management System for Retailers

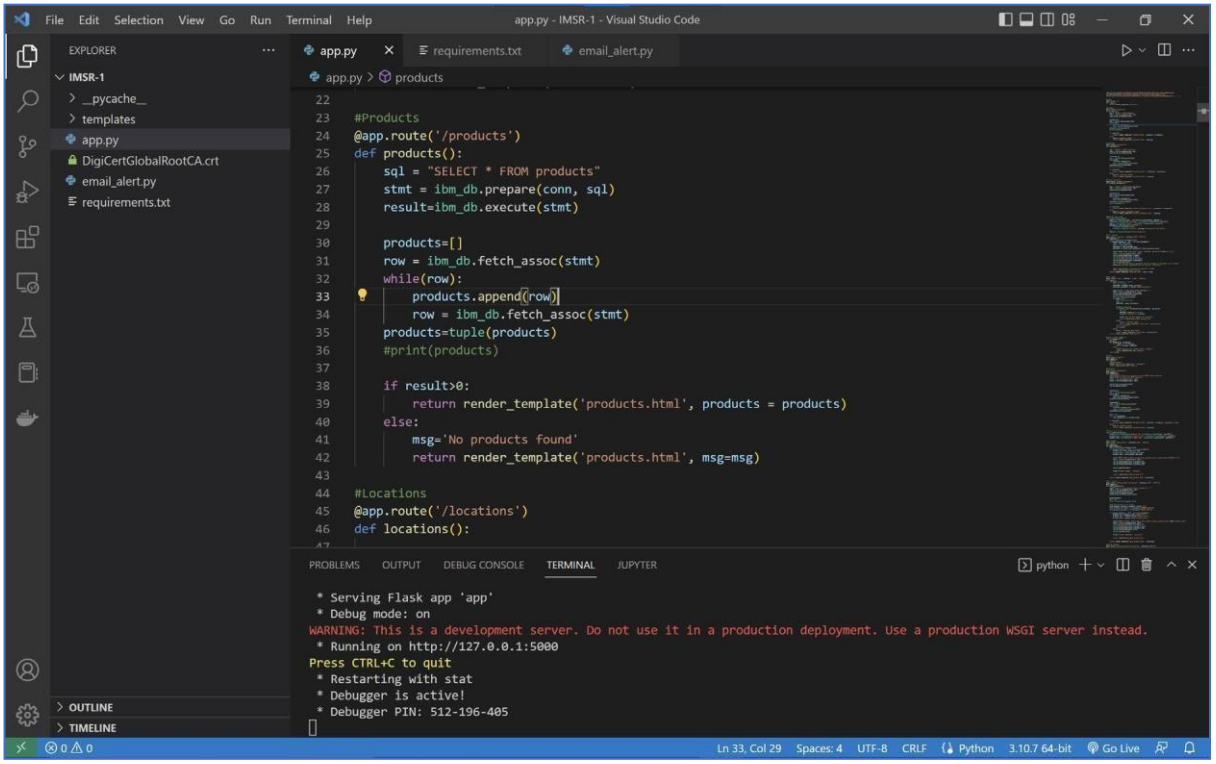
Introduction:

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.

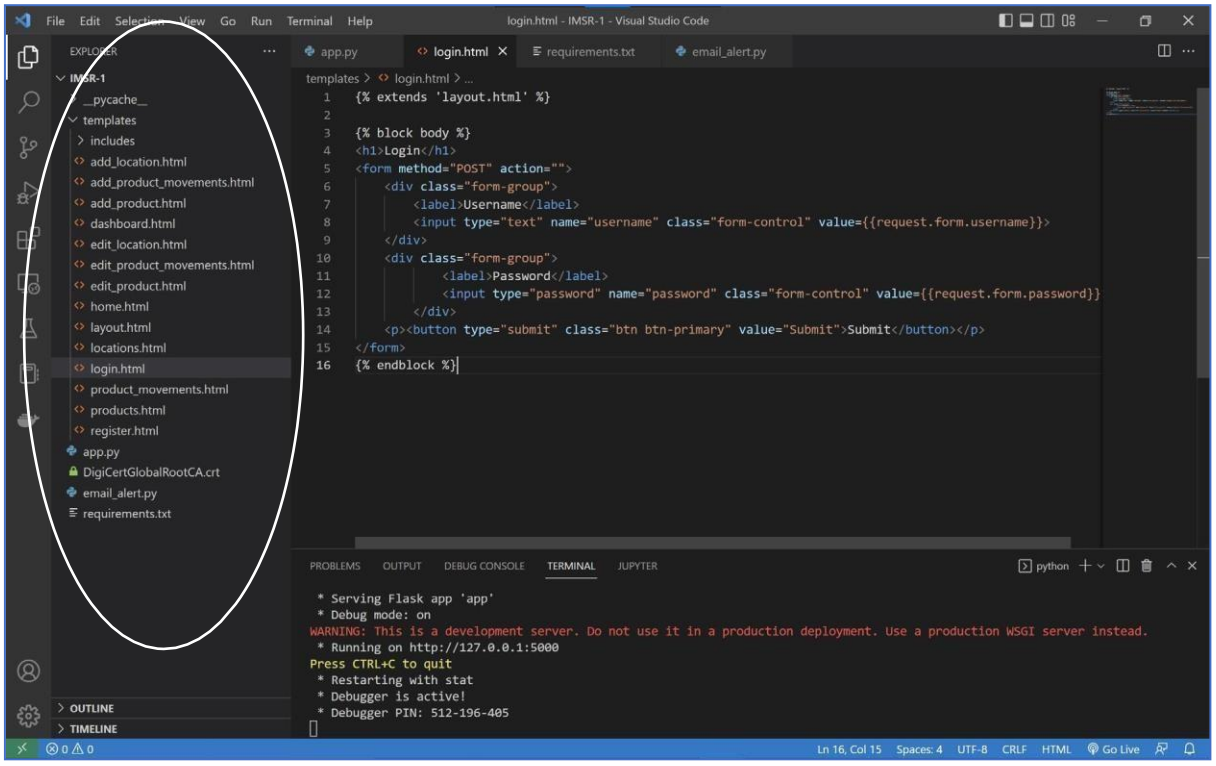
Example, (For Products page)



Sprint 2 – Frontend:

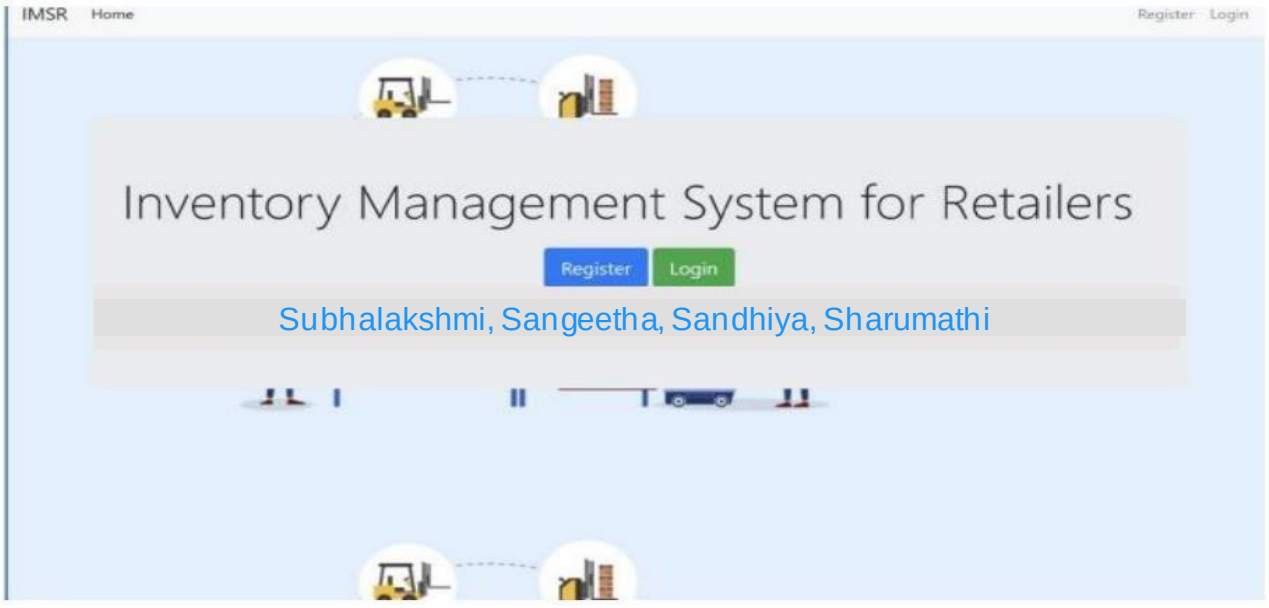
The frontend is written using HTML, CSS (using Bootstrap) and JavaScript for all the pages to which the routes created in Sprint 1.

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

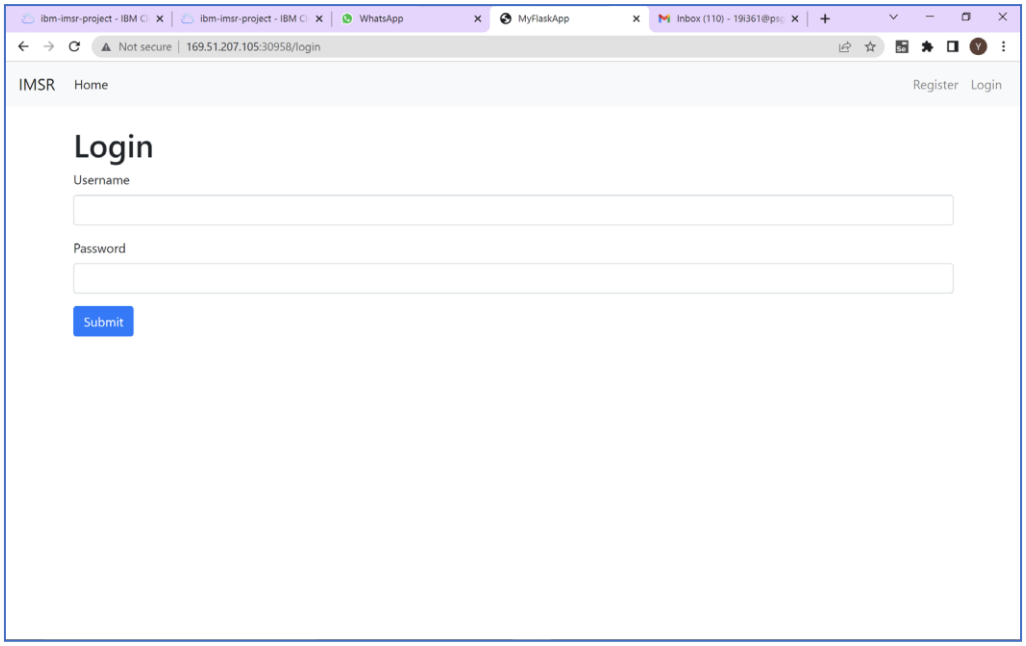


Sample FrontEnd Pages,

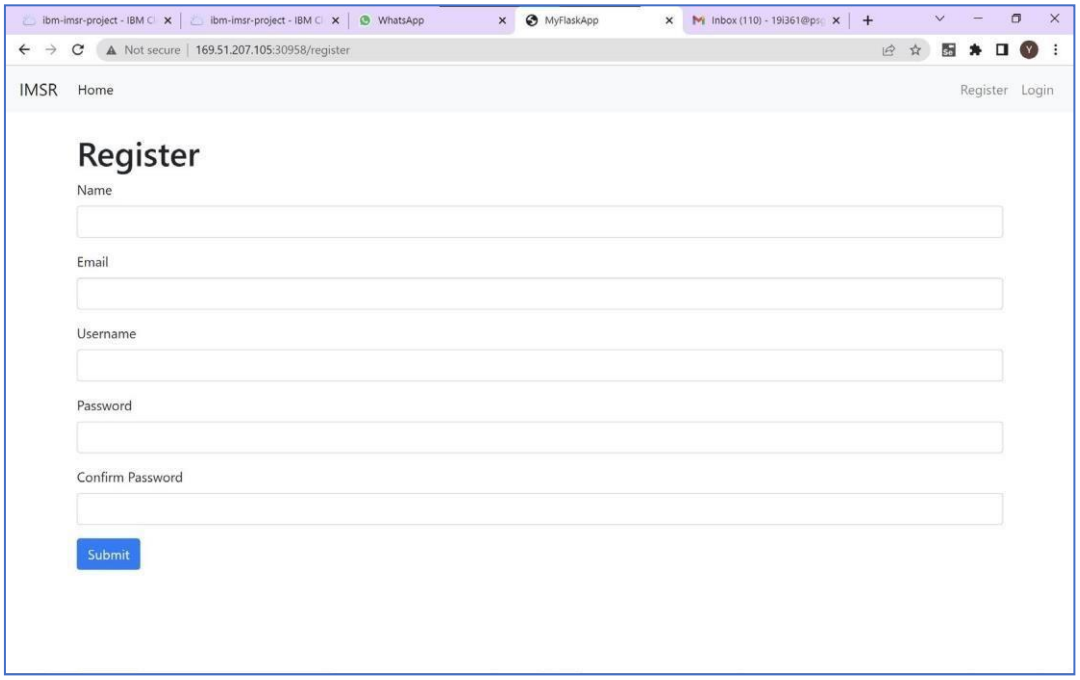
Home Page,



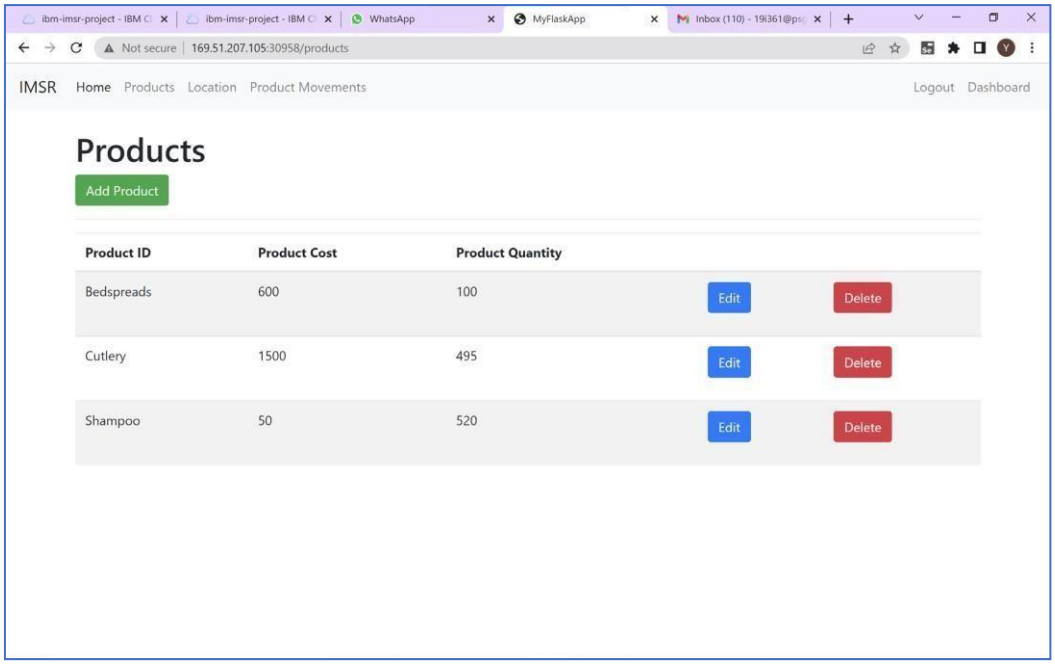
Login Page,



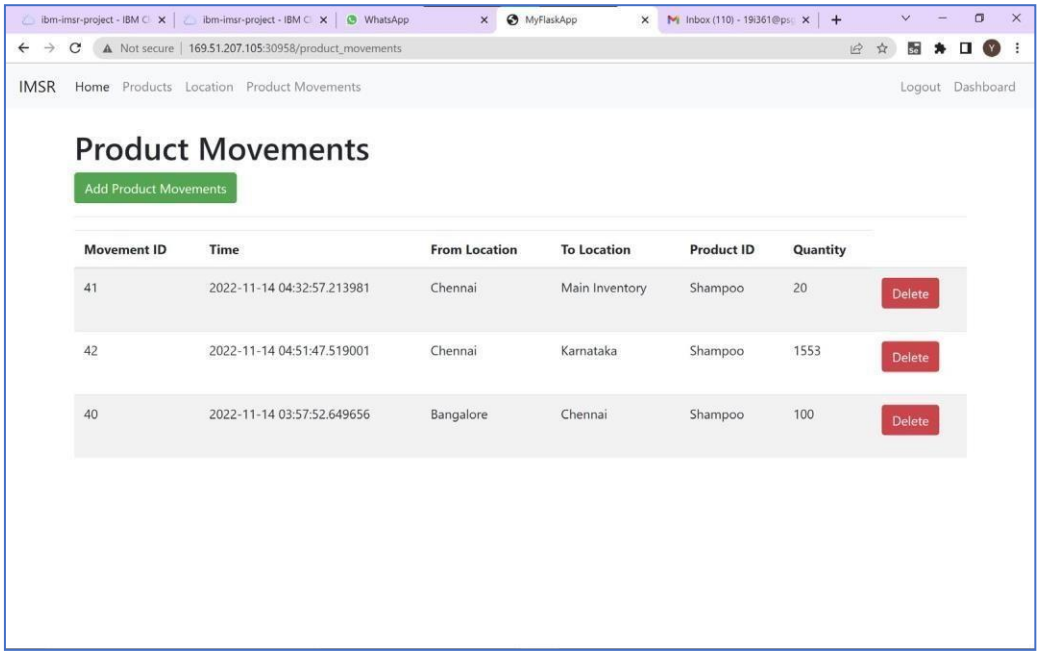
Register Page,



Products Page,



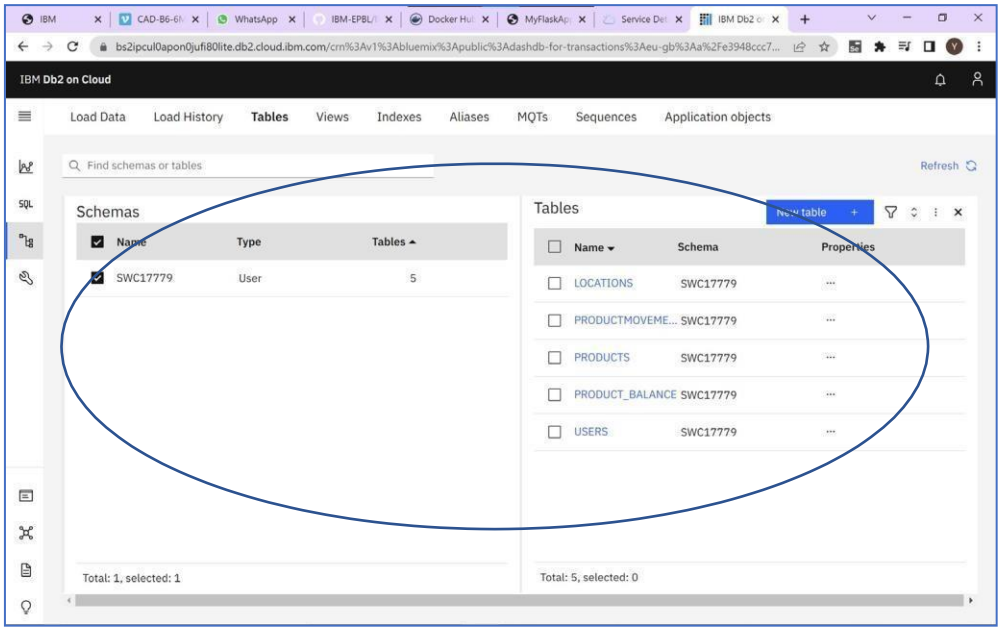
Product Movements Page,



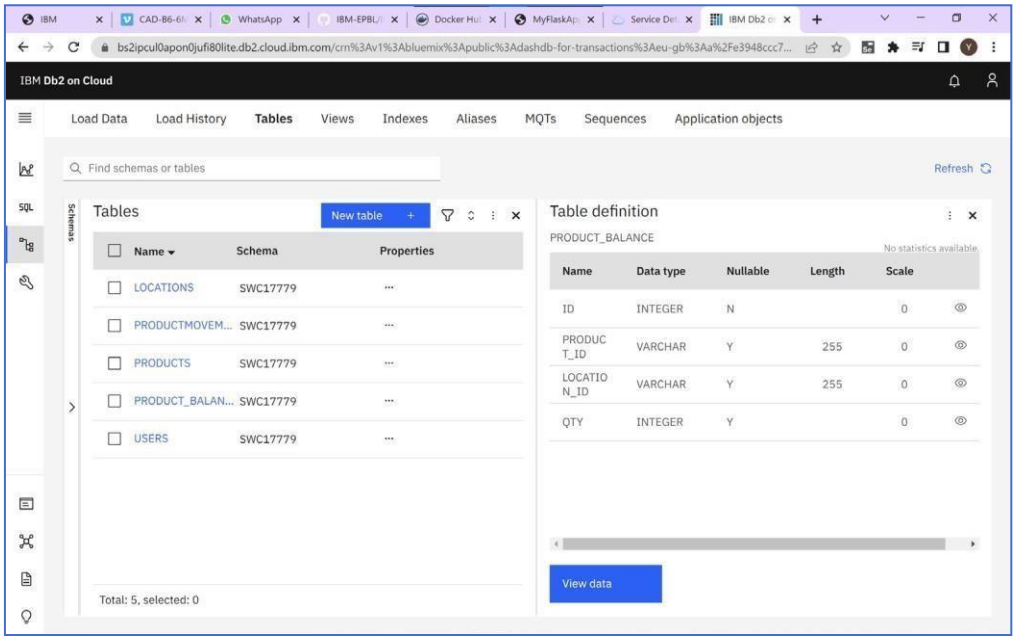
Sprint 3 - IBM Cloud Integration + Integration of SendGrid:

IBM Cloud Integration:

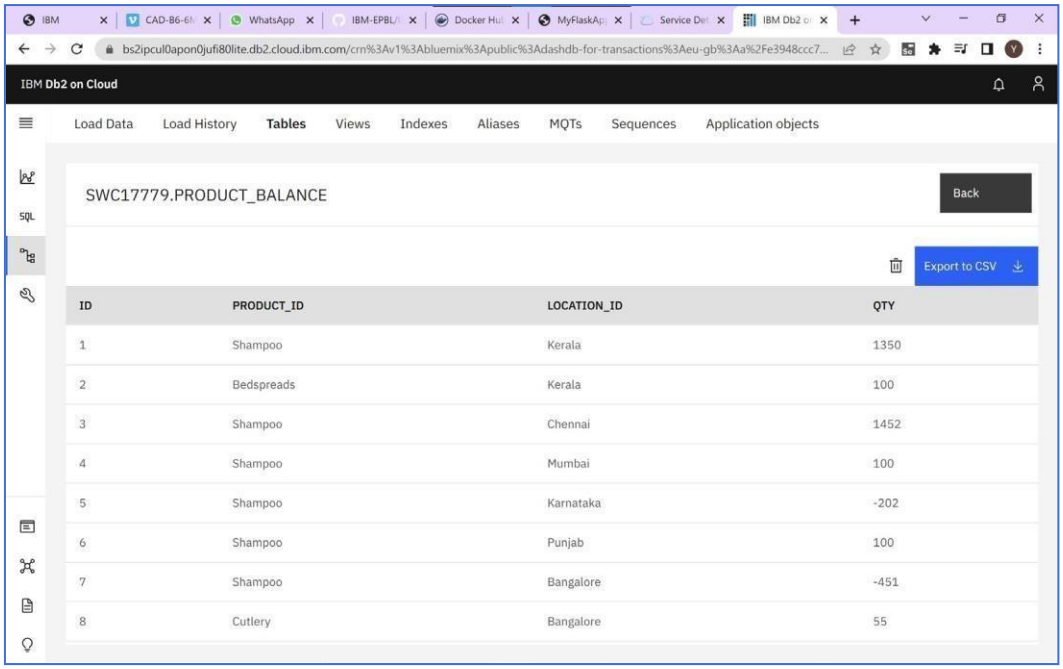
5 tables created for our project,



Schema of the particular table (For Example, Product_Balance)



Data of a particular table (For Example, Product_Balance)



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

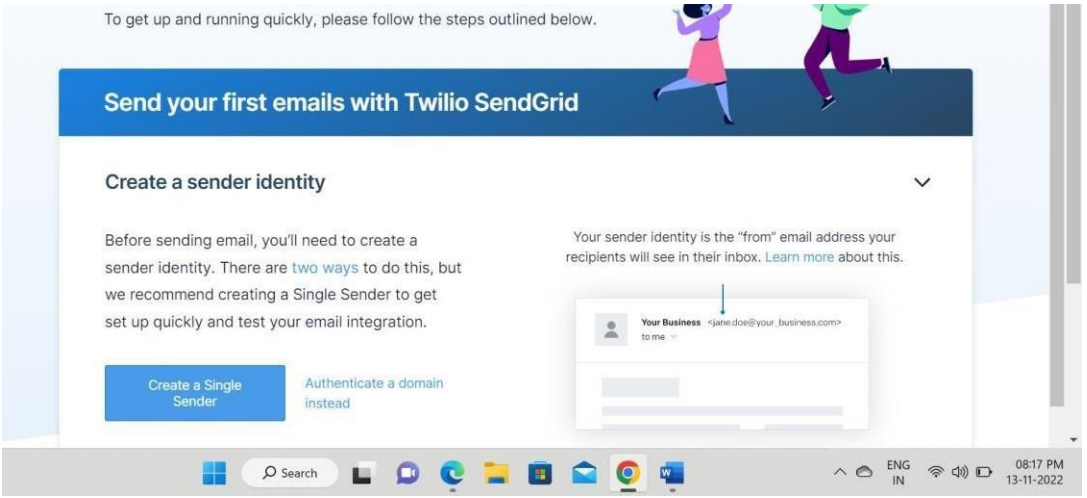
Code for Connection of IBM Database,

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

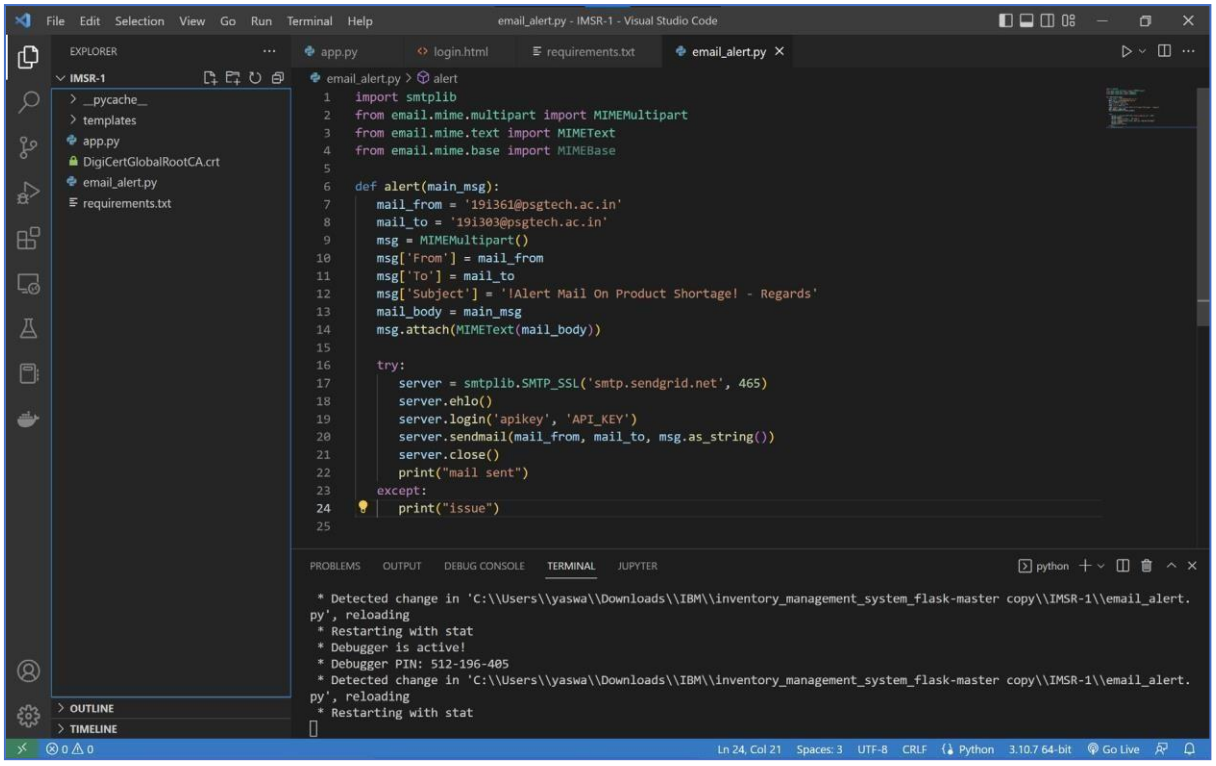
Note: DigiCertGlobalRootCA.crt should be downloaded and configured within the project folder.

SendGrid Integration:

Creation of SendGrid account,



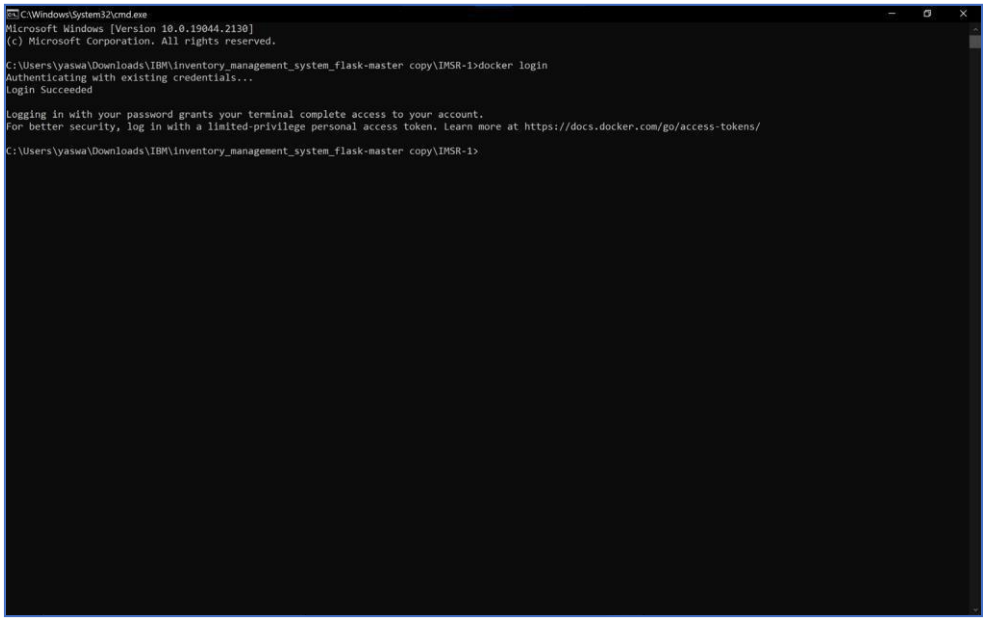
Code for email alert:



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.



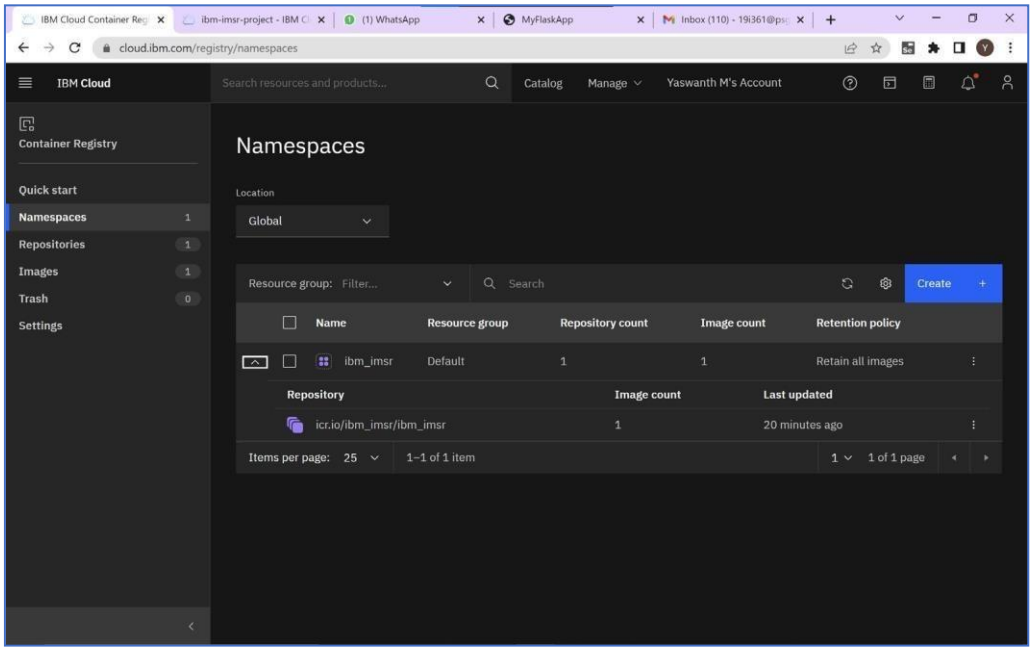
Building an image for our project,

```
  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):
```

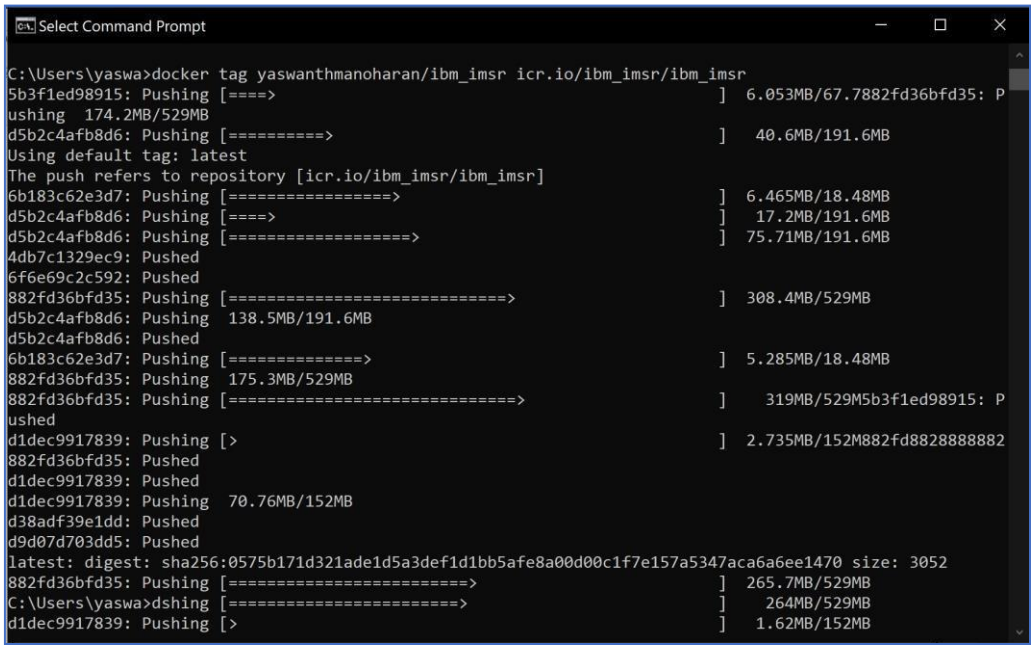
Create a valid Deployment.yaml file,

```
PS C:\Users\yaswa\Downloads\IBM\IMSR-1> kubectl apply -f deployment.yaml
deployment.apps/ibmimsr created
PS C:\Users\yaswa\Downloads\IBM\IMSR-1>
```

Create a namespace in IBM Container registry,

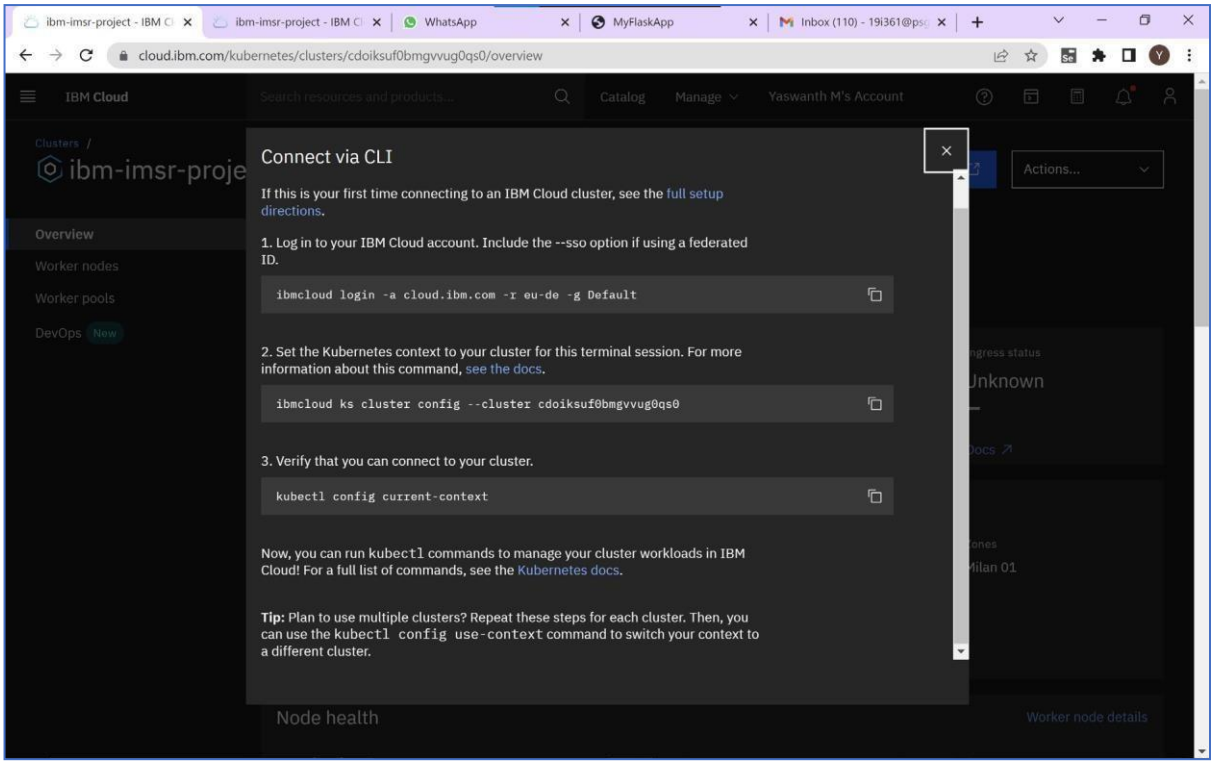


Pushing the project into IBM container Registry,



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

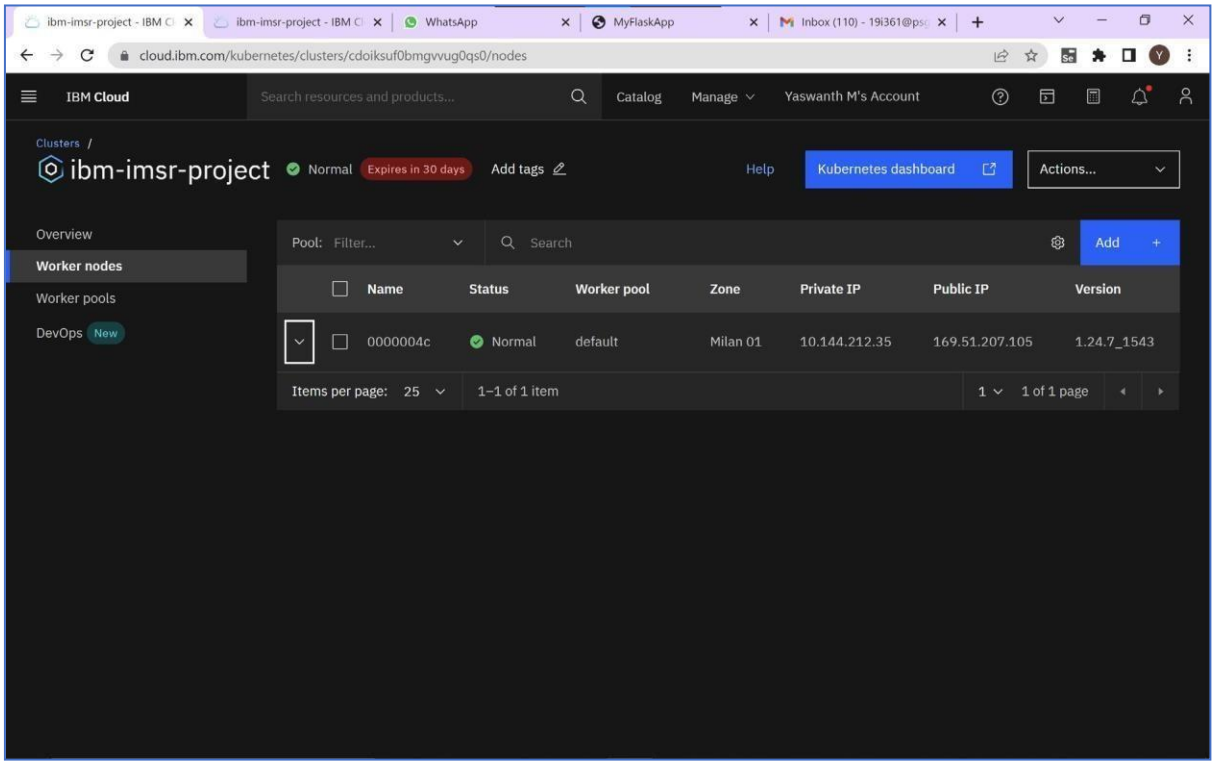
Then, Login into Kubernetes Cluster using the following commands,



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,

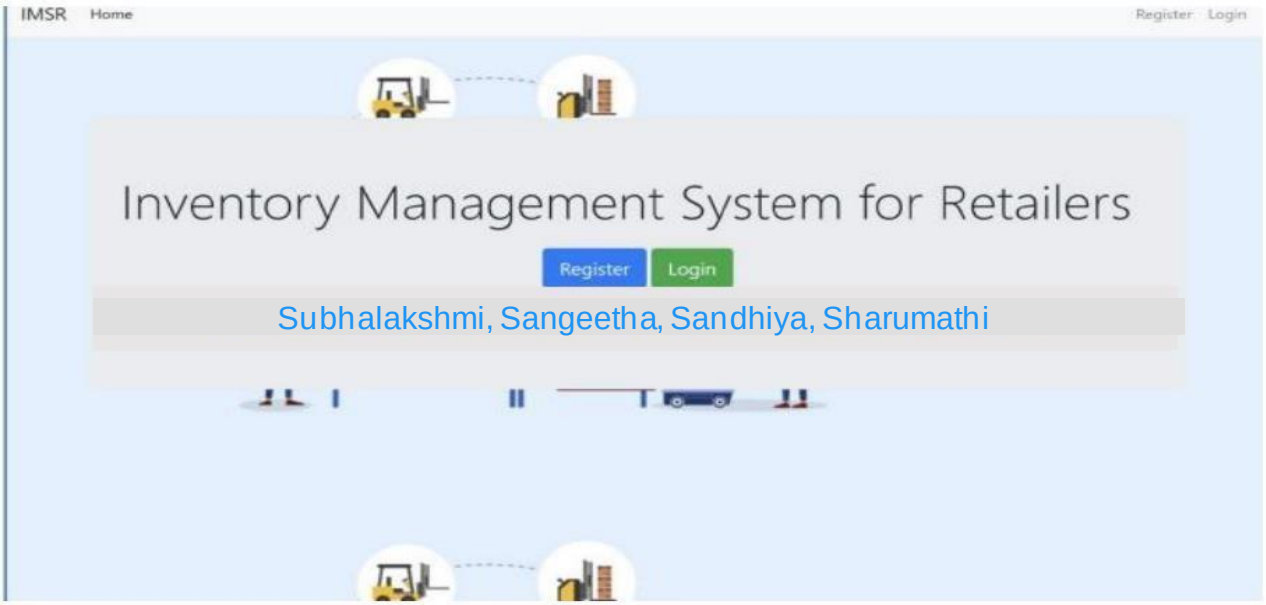


Thus we have the Public IP address and the Nodeport.

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!