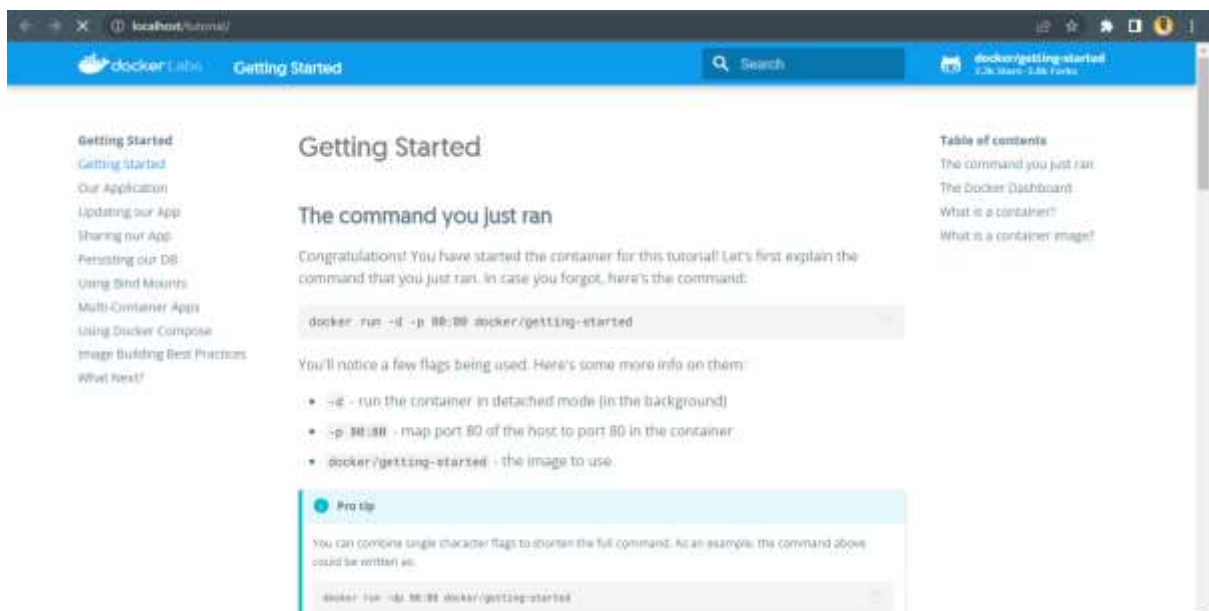


1. Pull an Image from the docker hub and run it in the docker playground.

Step 1: Installed docker and signed up with docker hub. Then, pulled the image (docker's get started) and run it in the docker playground from the command line interface of a computer.

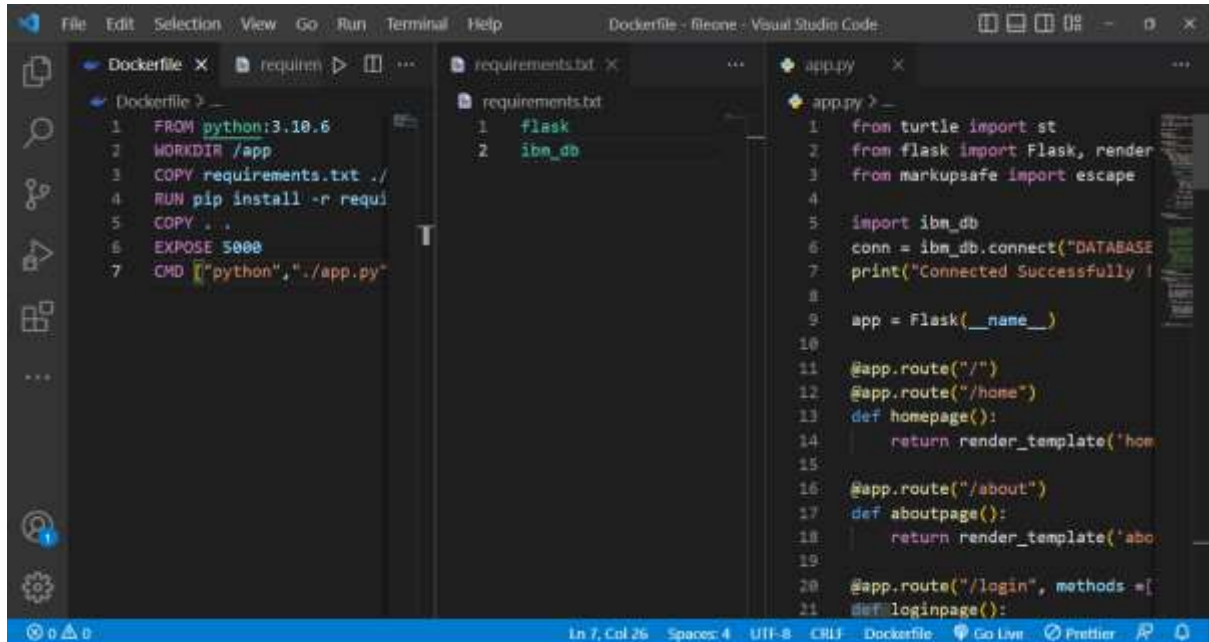


Step 2: Running the container locally.



2. Create a docker file for the job portal application and deploy it in the Docker desktop application.

Step 1: First, I created the docker file with requirements file. In which, flask and ibm_db are mentioned.



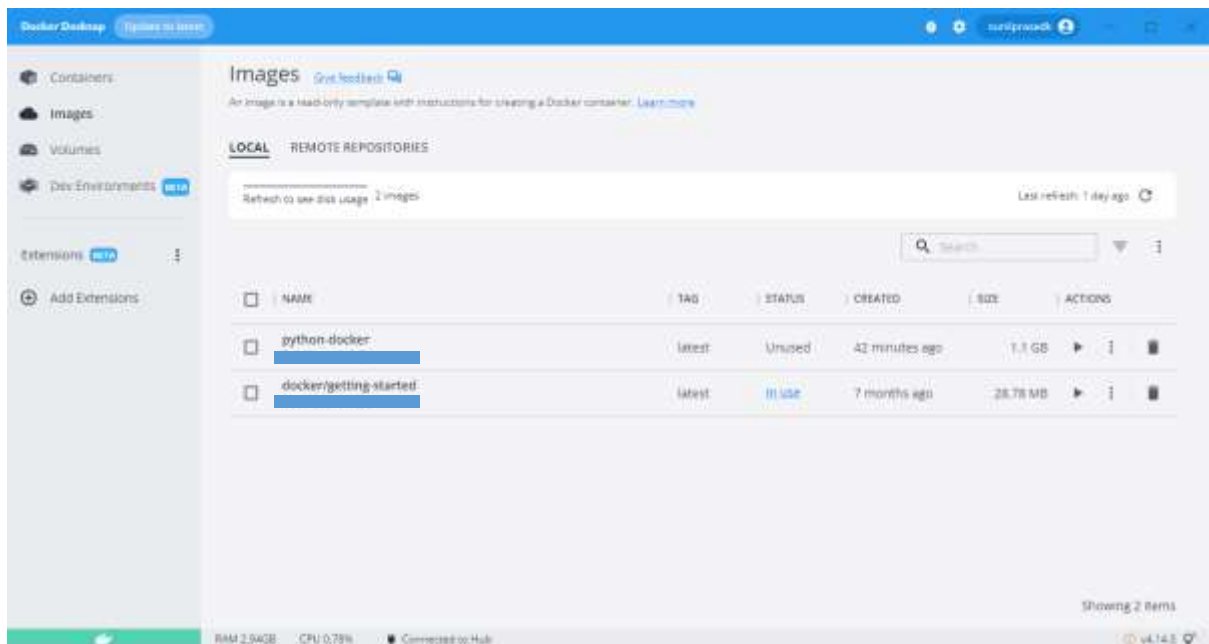
The screenshot shows a Visual Studio Code editor with three files open: Dockerfile, requirements.txt, and app.py. The Dockerfile contains instructions to build a Python image using python:3.10.6 as the base, set the working directory to /app, copy requirements.txt and the current directory, install dependencies with pip, and run the application with python ./app.py. The requirements.txt file lists flask and ibm_db. The app.py file is a Flask application that connects to a database, sets up routes for homepage, about, and login, and renders templates.

```
Dockerfile
1 FROM python:3.10.6
2 WORKDIR /app
3 COPY requirements.txt ./
4 RUN pip install -r requirements.txt
5 COPY . .
6 EXPOSE 5000
7 CMD ["python", "./app.py"]

requirements.txt
1 flask
2 ibm_db

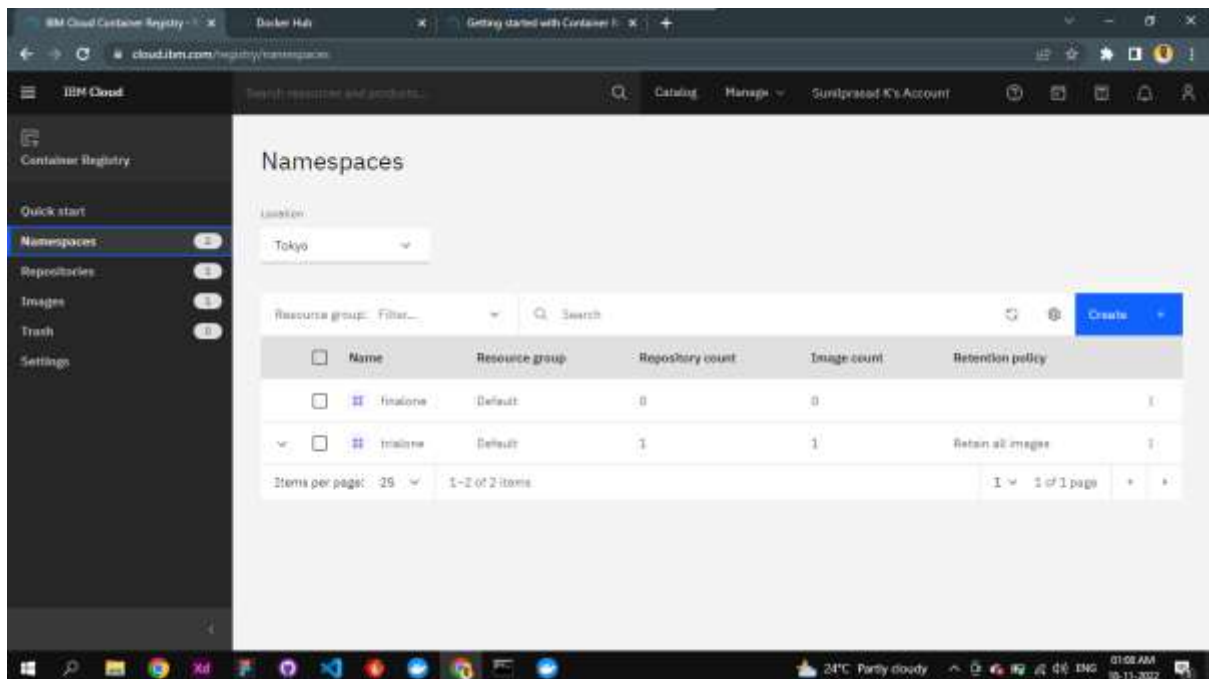
app.py
1 from turtle import st
2 from flask import Flask, render
3 from markupsafe import escape
4
5 import ibm_db
6 conn = ibm_db.connect("DATABASE
7 print("Connected Successfully !
8
9 app = Flask(__name__)
10
11 @app.route("/")
12 @app.route("/home")
13 def homepage():
14     return render_template('hom
15
16 @app.route("/about")
17 def aboutpage():
18     return render_template('abo
19
20 @app.route("/login", methods = [
21 def loginpage():
```

Step 2: Then, build the image and run the image container in the docker desktop application.

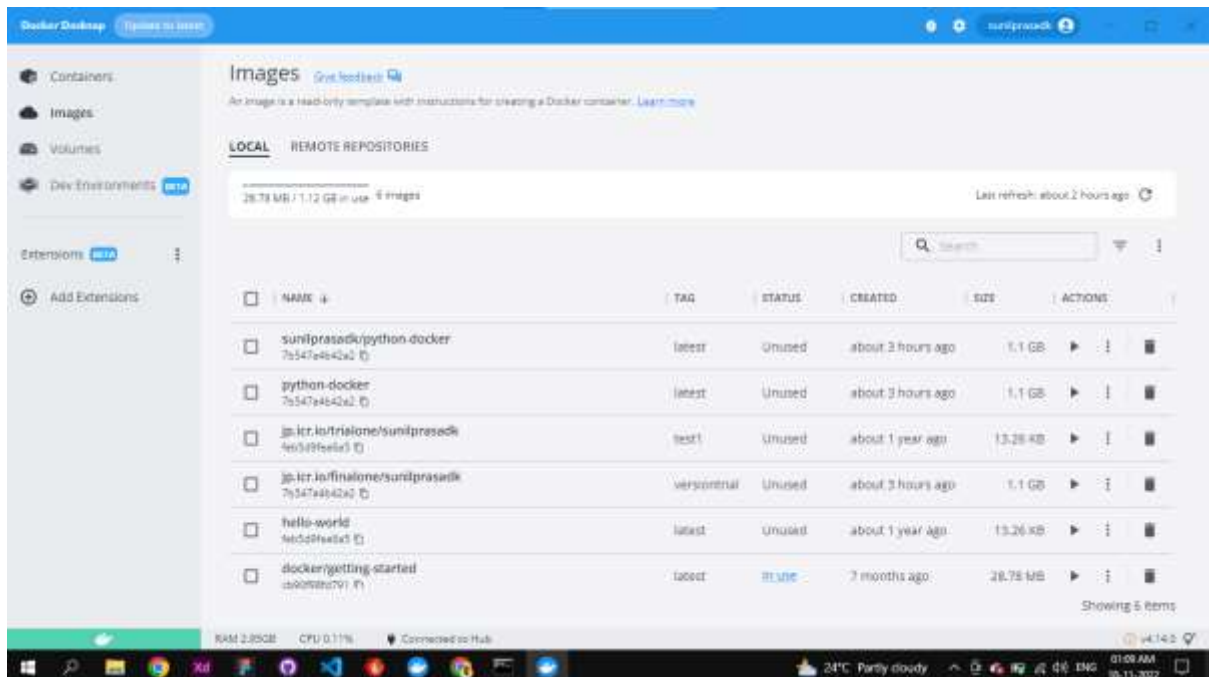


3. Create an IBM container registry and deploy the HelloWorld app or job portal app.

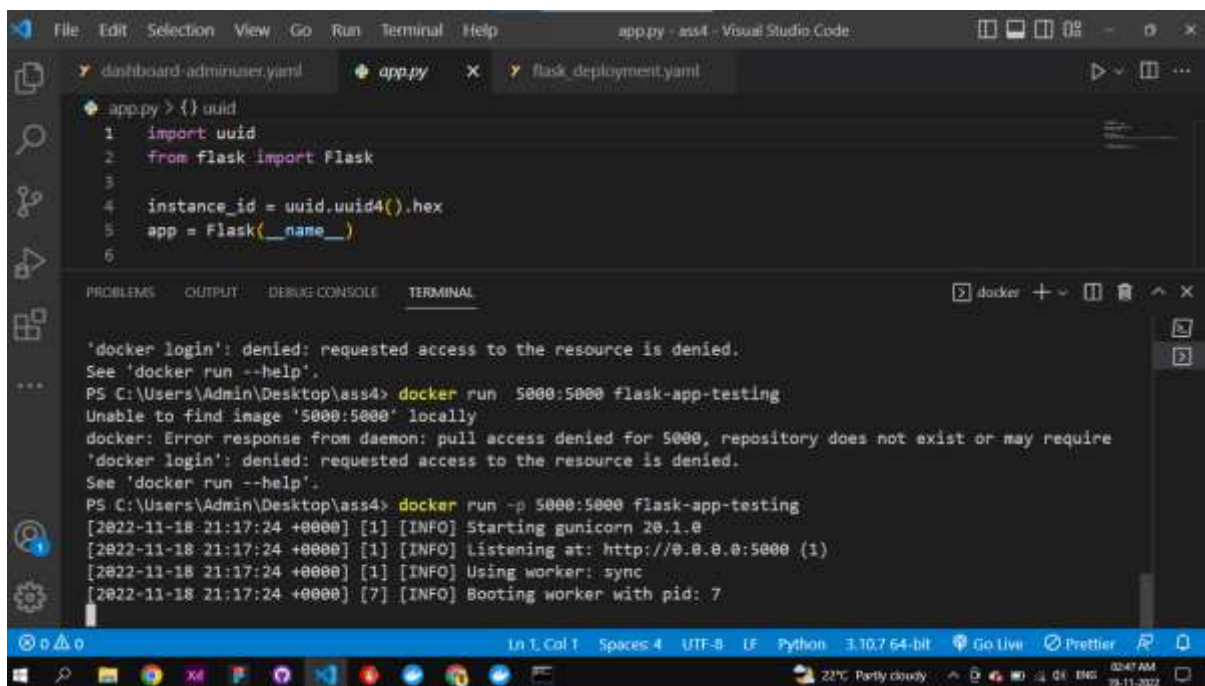
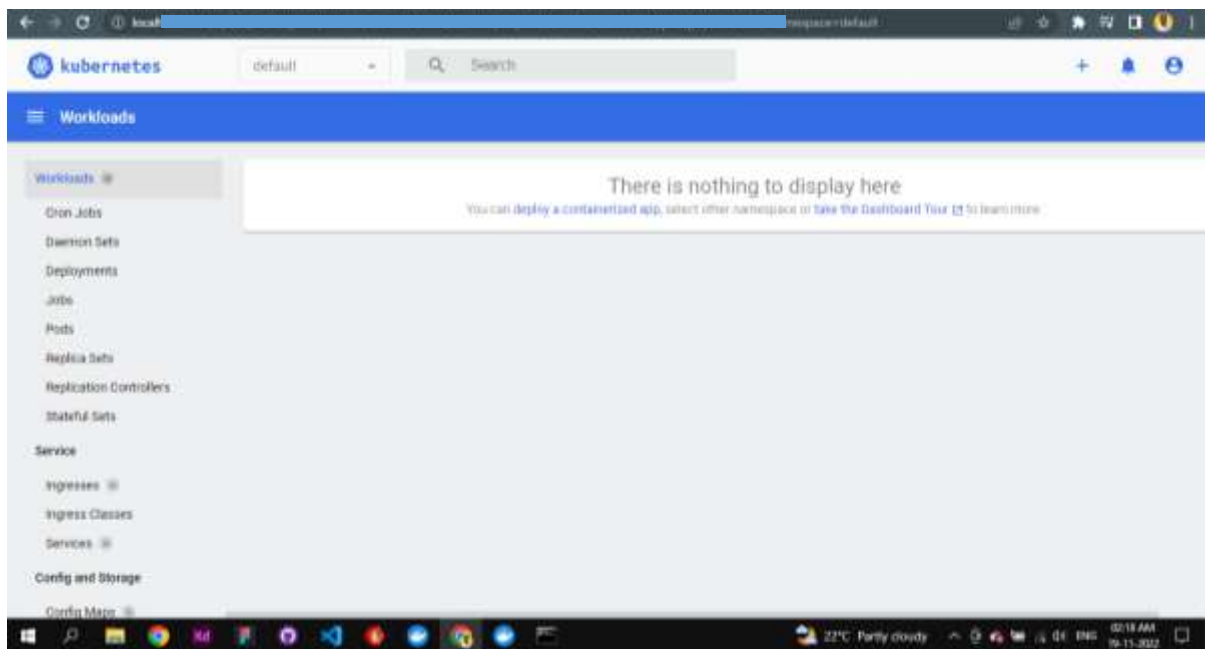
Step 1: First, I created the container registry in IBM cloud, then create the namespaces and perform the quick start commands to understand the operation.



Step 2: Finally, I tag the image which I want to send to the IBM cloud container registry. Then I push it to container registry.



3. Create a Kubernetes cluster in IBM cloud and deploy helloworld image or jobportal image and also expose the same app to run in nodeport.



kubernetes

All namespaces

Search

Workloads > Pods

Daemon Sets

Deployments

Jobs

Pods

Replica Sets

Replication Controllers

Stateful Sets

Service

Ingresses

Ingress Classes

Services

Config and Storage

Config Maps

Persistent Volume Claims

Secrets

Storage Classes

Pods

Name	Namespace	Images	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created
flash-app-8366b947b-w107	kubernetes-dashboard	flash-app-test:ng	app: flash-app pod-template-hash: 8366b947b	docker-desktop	Running	0	-	-	8 minutes ago
flash-app-8366b947b-vkz8	kubernetes-dashboard	flash-app-test:ng	app: flash-app pod-template-hash: 8366b947b	docker-desktop	Running	0	-	-	8 minutes ago
flash-app-8366b947b-9g9p	kubernetes-dashboard	flash-app-test:ng	app: flash-app pod-template-hash: 8366b947b	docker-desktop	Running	0	-	-	8 minutes ago
kubernetes-dashboard-66c8b7759-6027x	kubernetes-dashboard	kubernetes-dashboard:v2.6.1	kube-app: kubernetes-dashboard	docker-desktop	Running	9	-	-	an hour ago

22°C

Partly cloudy

03:00 AM

06-11-2022