

# Assignment Kubernetes / Docker

Team ID	PNT2022TMID40334
Project Name	Project - Skill/Job Recommender Application

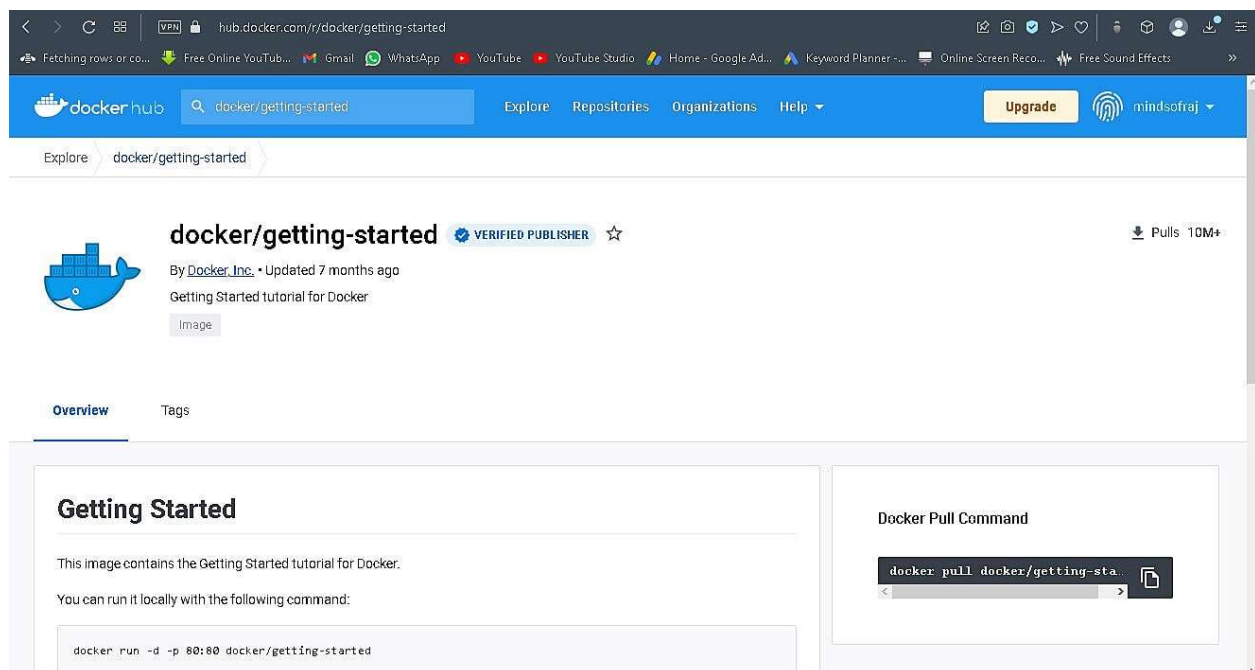
## Prerequisites :

- Download and Install the Docker Desktop for windows
- Login to the Docker Desktop

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

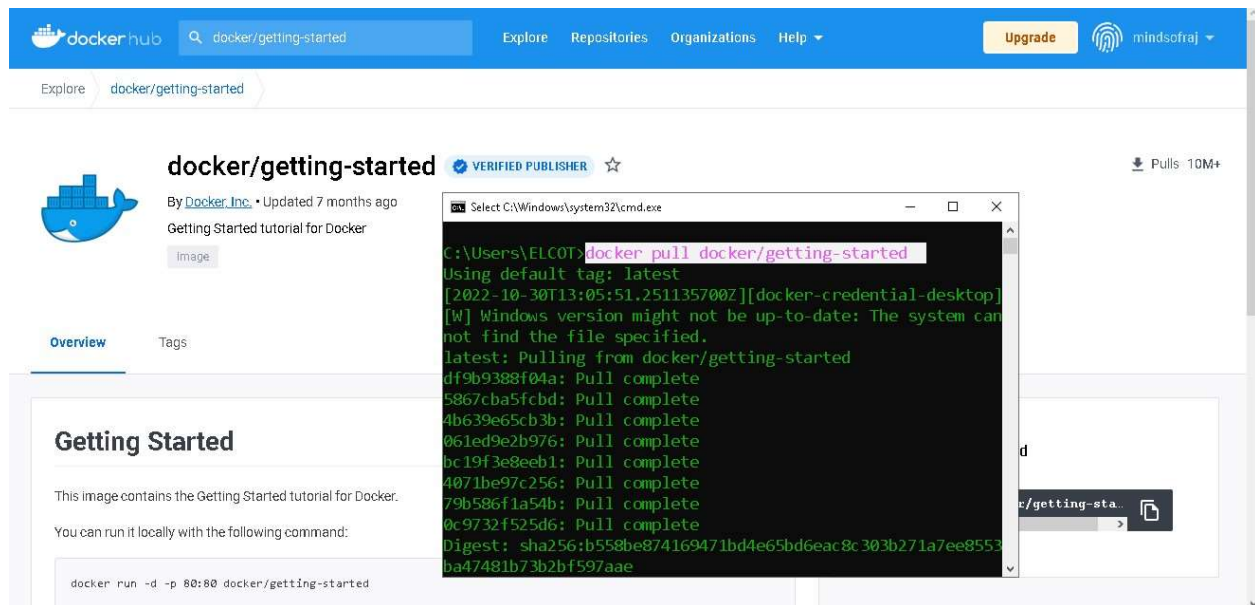
1.1. First We have to signup to the Docker Hub (<https://hub.docker.com>)

1.2. Search for the docker images

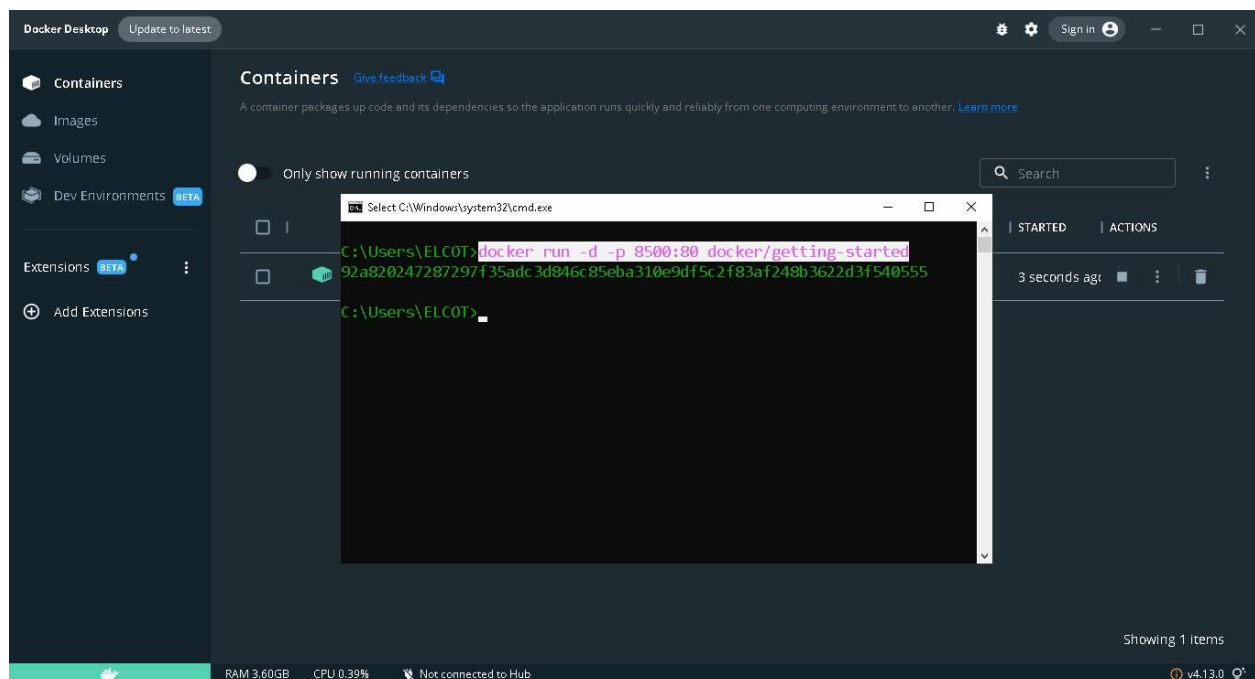


The screenshot shows the Docker Hub interface for the 'docker/getting-started' image. The page includes a search bar with 'docker/getting-started' entered, a navigation bar with 'Explore', 'Repositories', 'Organizations', and 'Help', and a user profile 'mindsofraz'. The main content area features the Docker logo, the image name 'docker/getting-started' with a 'VERIFIED PUBLISHER' badge, and a description: 'Getting Started tutorial for Docker'. Below this, there are tabs for 'Overview' and 'Tags'. The 'Overview' tab is active, showing a 'Getting Started' section with the text: 'This image contains the Getting Started tutorial for Docker. You can run it locally with the following command:'. The command is displayed in a code block: `docker run -d -p 80:80 docker/getting-started`. To the right, there is a 'Docker Pull Command' section with a code block showing: `docker pull docker/getting-started`.

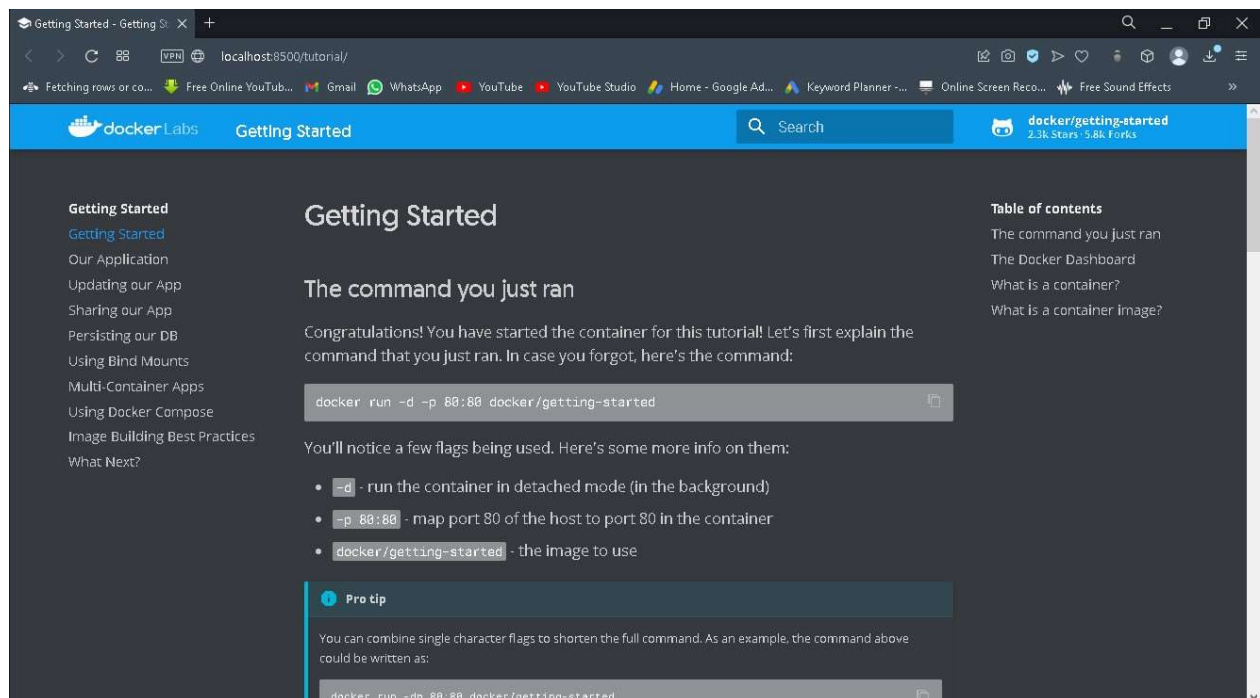
### 1.3. Run the pull command in command prompt



### 1.4. After Successfully downloading the docker image run it using the command prompt in a desired port.

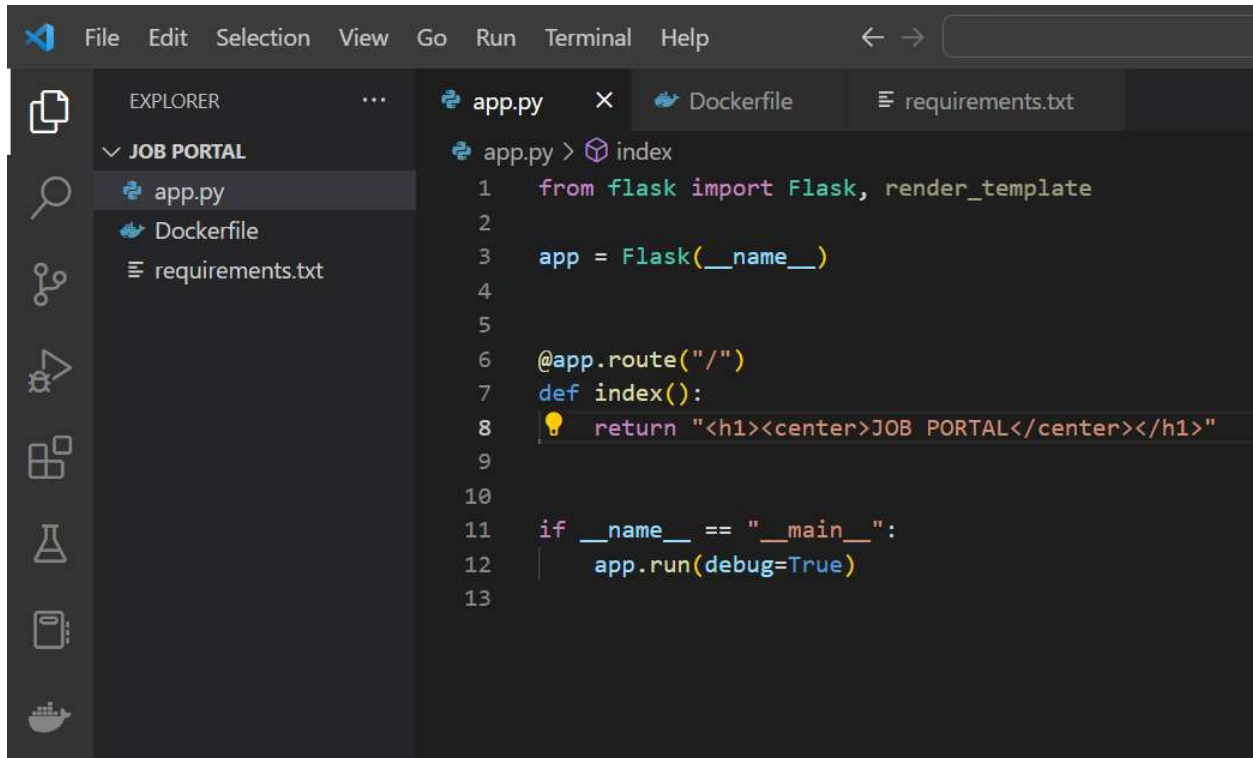


1.5. Then open the localhost inside the browser on the given port  
*https://localhost:8500/*



## 2. Create a docker file for the jobportal application and deploy it in Docker desktop application.

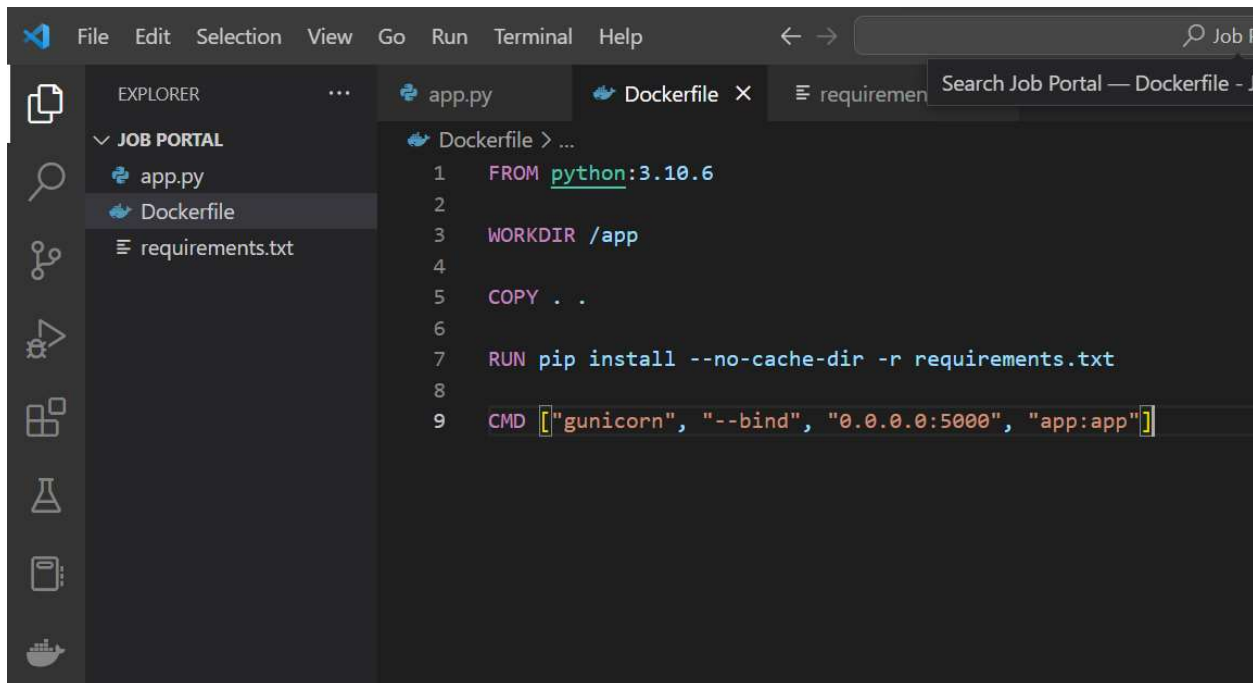
### 2.1. Create Job Portal Flask Application



The screenshot shows the Visual Studio Code interface with a project named 'JOB PORTAL'. The Explorer sidebar on the left shows the project structure with files 'app.py', 'Dockerfile', and 'requirements.txt'. The main editor window displays the 'app.py' file, which contains the following Python code:

```
1 from flask import Flask, render_template
2
3 app = Flask(__name__)
4
5
6 @app.route("/")
7 def index():
8     return "<h1><center>JOB PORTAL</center></h1>"
9
10
11 if __name__ == "__main__":
12     app.run(debug=True)
13
```

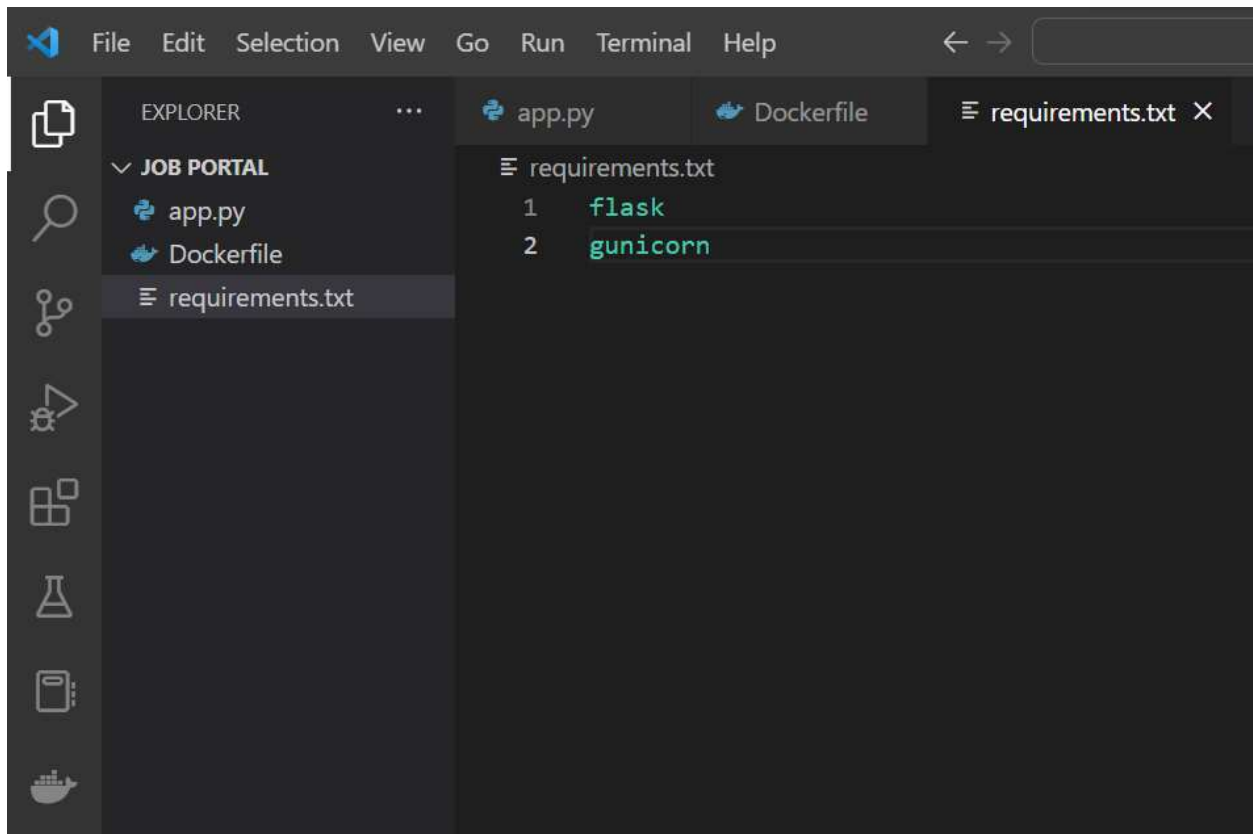
### 2.2. Create a Dockerfile



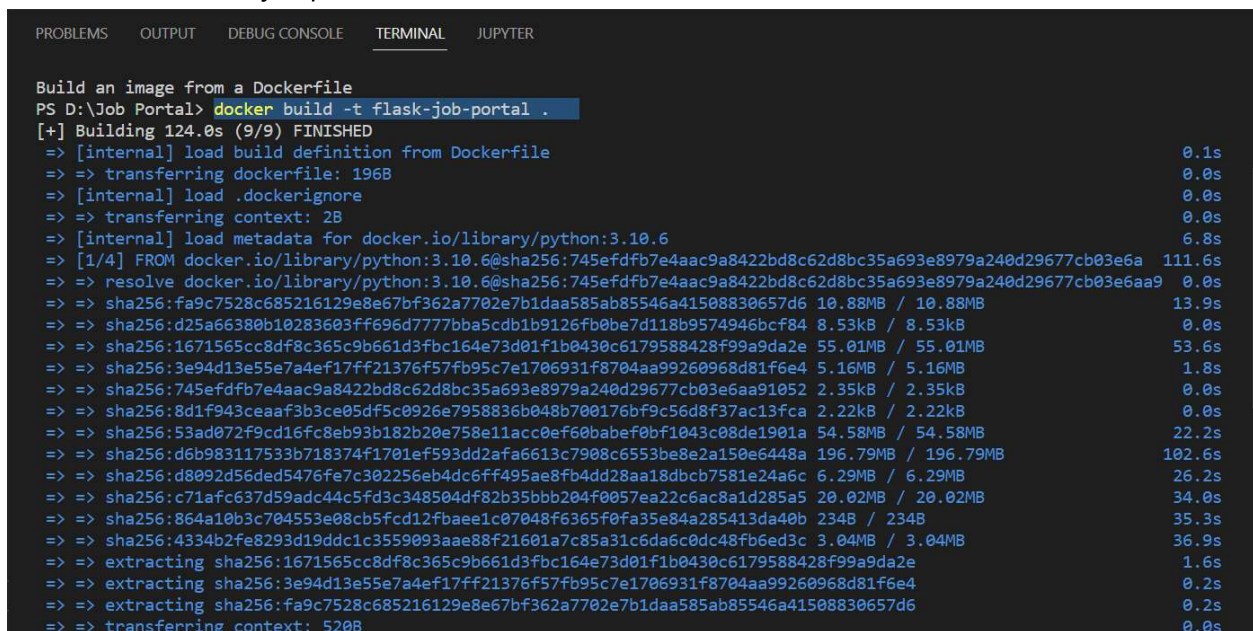
The screenshot shows the Visual Studio Code interface with the 'JOB PORTAL' project. The Explorer sidebar on the left shows the project structure with files 'app.py', 'Dockerfile', and 'requirements.txt'. The main editor window displays the 'Dockerfile' file, which contains the following Dockerfile instructions:

```
1 FROM python:3.10.6
2
3 WORKDIR /app
4
5 COPY . .
6
7 RUN pip install --no-cache-dir -r requirements.txt
8
9 CMD ["gunicorn", "--bind", "0.0.0.0:5000", "app:app"]
```

## 2.3. Create Requirements.txt File



## 2.4. Build the Docker Image Using the Docker `docker build -t flask-job-portal`.



## 2.5. Run the Docker Image using Docker Command

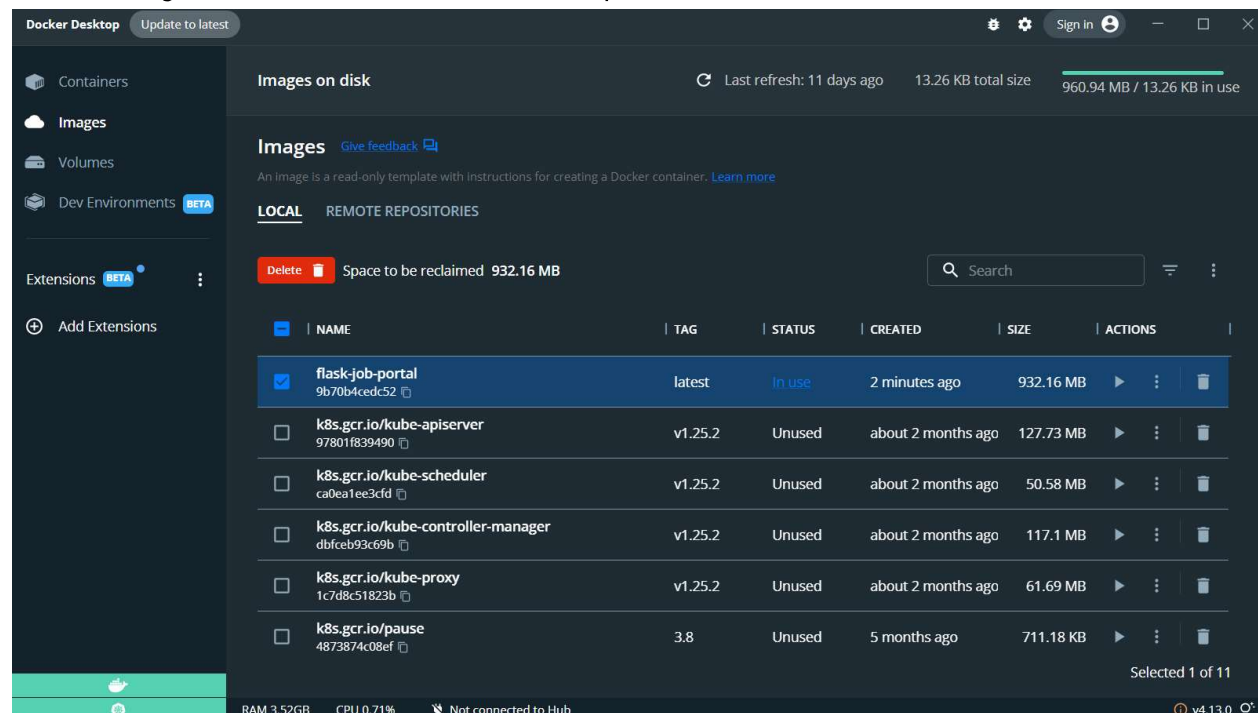
`docker run -d -p 5000:5000 flask-job-portal`

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

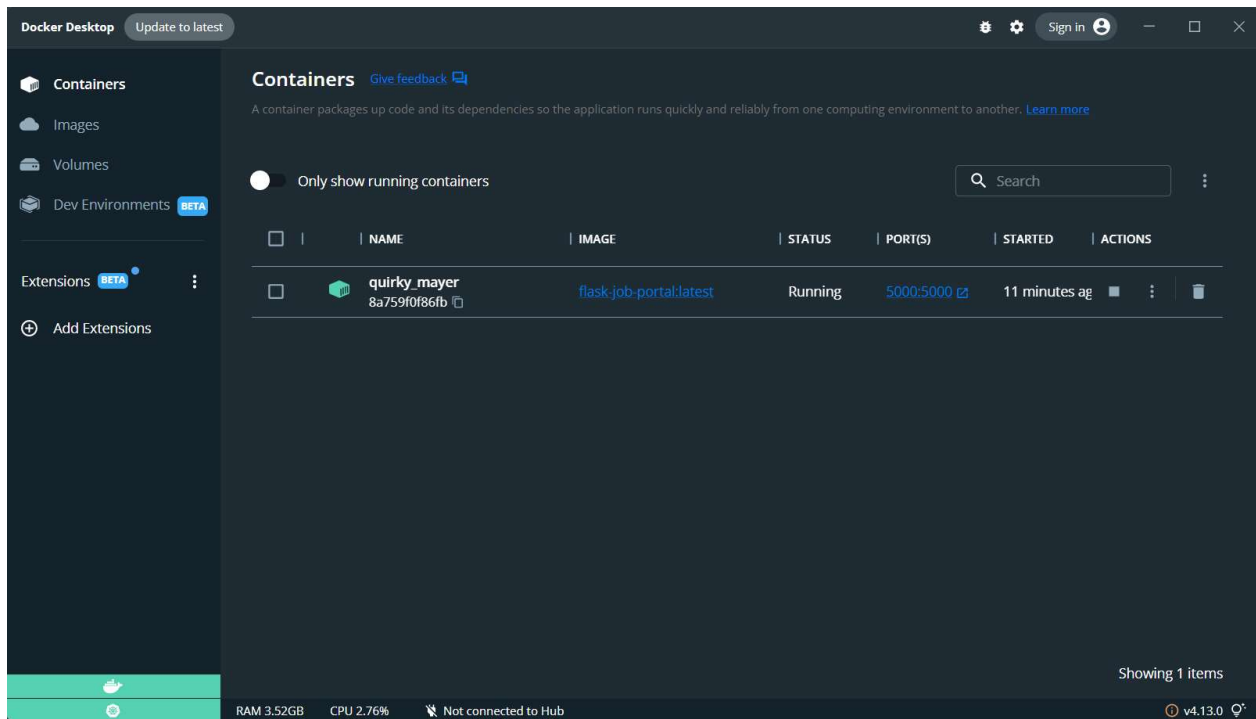
=> => transferring context: 520B                                0.0s
=> [2/4] WORKDIR /app                                           0.3s
=> [3/4] COPY . .                                               0.0s
=> [4/4] RUN pip install --no-cache-dir -r requirements.txt     5.1s
=> exporting to image                                           0.1s
=> => exporting layers                                           0.1s
=> => writing image sha256:9b70b4cedc527190e3ef430d3fbc1ab08316395b38f2b573a5b6e71bceaba47d  0.0s
=> => naming to docker.io/library/flask-job-portal              0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
PS D:\Job Portal> docker run -d -p 5000:5000 flask-job-portal
8a759f0f86fb24897300a09a2e694bc74e97352d606d7825f7736ab0816131e9
PS D:\Job Portal> |
```

## 2.6. An image is Created in the Docker desktop



## 2.7. A Container is created on the port 5000



## 2.8. Our app is running in the browser on the localhost <http://127.0.0.1:5000/>

