Report

Virtualization - Docker Containerization for Web Application

The purpose of this lab was to demonstrate the process of containerizing a web application using Docker. The lab involved creating a Dockerfile to define the environment and dependencies, building a Docker image, and running a container based on that image. The specific example used was a simple Python web application using the Flask framework.

- Set up the Flask Application:
    - Create a project directory.
    - Create a simple Flask application (app.py).
- Create Dockerfile and requirements.txt:
    - Create a Dockerfile specifying the Python runtime and application setup.
    - Create a requirements.txt file listing Flask as a dependency.
- Build and Run the Docker Container:
    - Build the Docker image using the Dockerfile.
    - Run a container based on the built image.
- Access the Web Application:
    - Open a web browser and visit http://localhost:5000 to see the running application.

This lab provided hands-on experience in containerizing a web application using Docker. The Dockerfile defined the environment, dependencies, and application setup. Building the Docker image and running a

container demonstrated the ease of deployment and scalability that Docker offers. The ability to access the web application through a browser highlighted the isolation and consistency achieved through containerization.

The successful completion of this lab marks a significant milestone in understanding and implementing Docker containerization. By containerizing a simple web application, we have experienced firsthand the efficiency, consistency, and scalability that Docker brings to the deployment process. This technology has become integral to modern software development, and the skills acquired in this lab lay the groundwork for navigating the evolving landscape of container orchestration and deployment strategies.

Docker containers are widely adopted in industry practices due to their ability to encapsulate applications and their dependencies. The advantages of isolation, portability, and scalability demonstrated in this lab directly translate to real-world scenarios. Organizations globally leverage Docker to streamline their development pipelines, ensuring that applications run consistently across various environments and can be easily scaled to meet changing demands.

In comparing Docker containerization with traditional hosting methods, the benefits of Docker become evident. The isolation and portability provided by Docker ensure that applications can be packaged with their dependencies, reducing conflicts and versioning issues. The consistent environment across different servers simplifies the deployment process, making it reproducible and reliable. The scalability and load balancing features, coupled with straightforward deployment and rollback processes, showcase the agility Docker brings to software delivery.

Docker containers offer several advantages over traditional hosting methods:

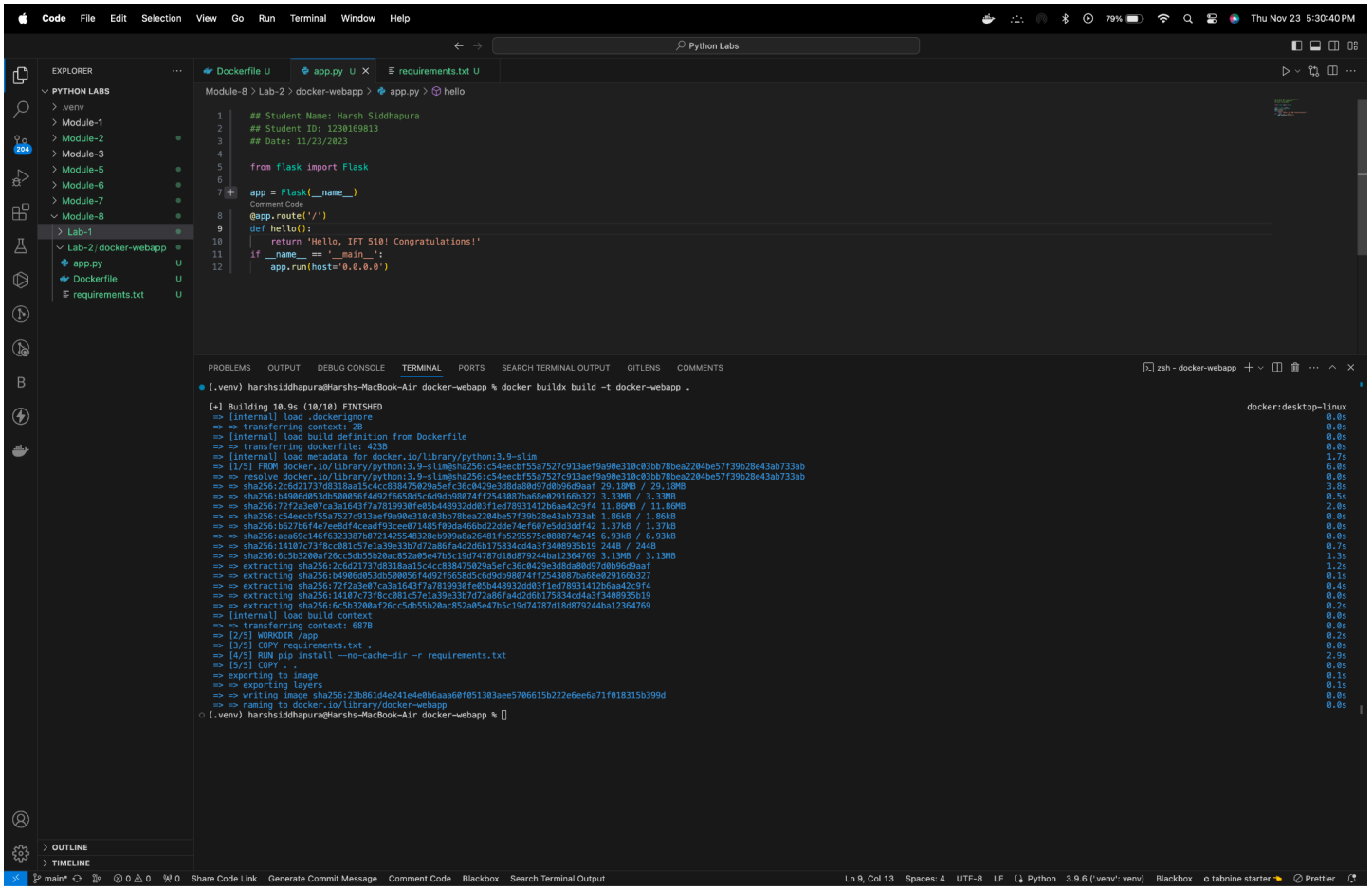
- Isolation and Portability: Docker ensures isolated environments, avoiding conflicts between applications.

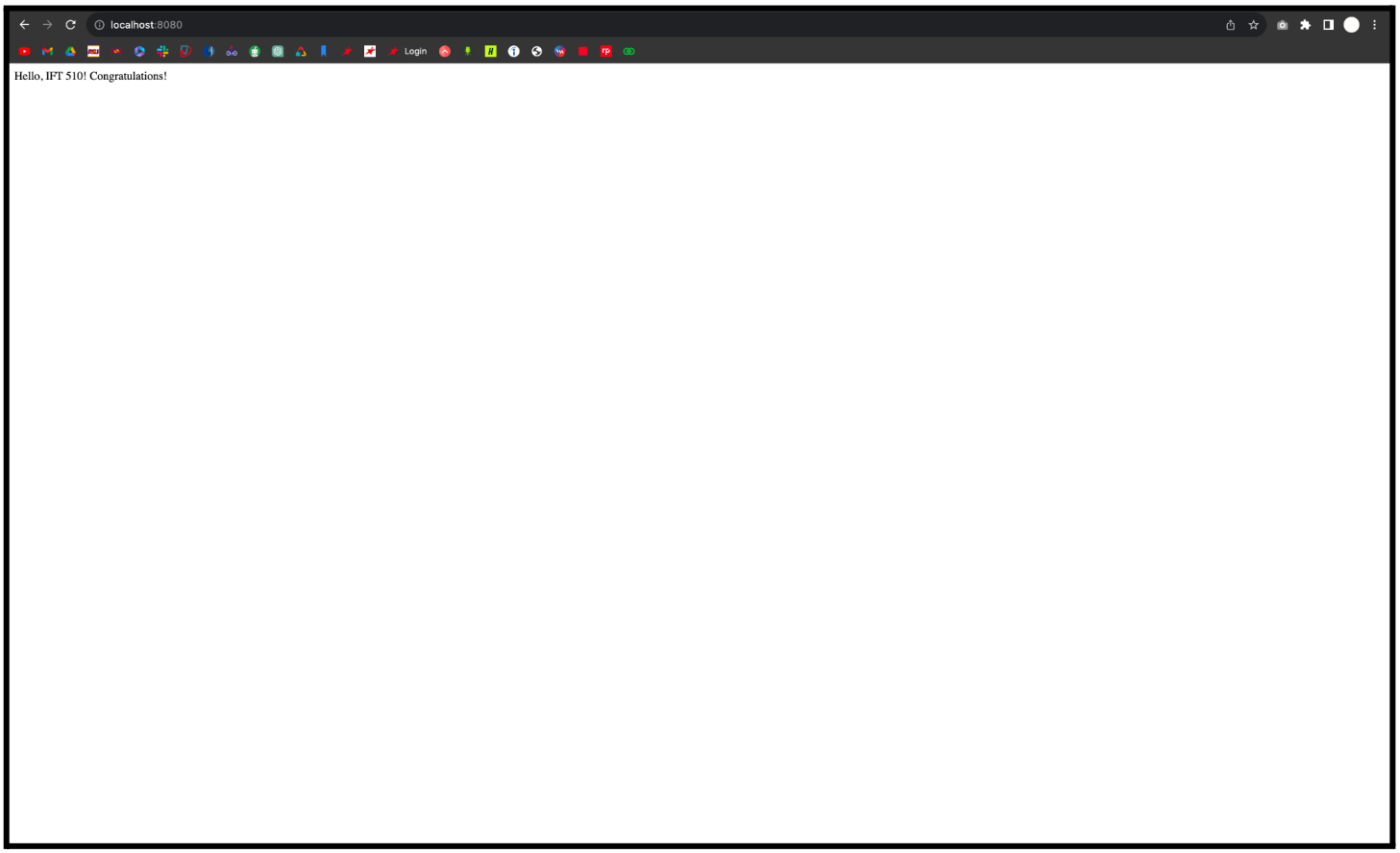
- Environment Consistency: Docker's Dockerfile ensures consistent deployments across different environments.
- Scalability and Load Balancing: Docker facilitates horizontal scaling and load balancing with container orchestration tools.
- Deployment and Rollback: Docker simplifies deployment and rollback processes through image-based updates.
- Infrastructure Utilization: Docker enables efficient resource utilization by running multiple containers on a single host.

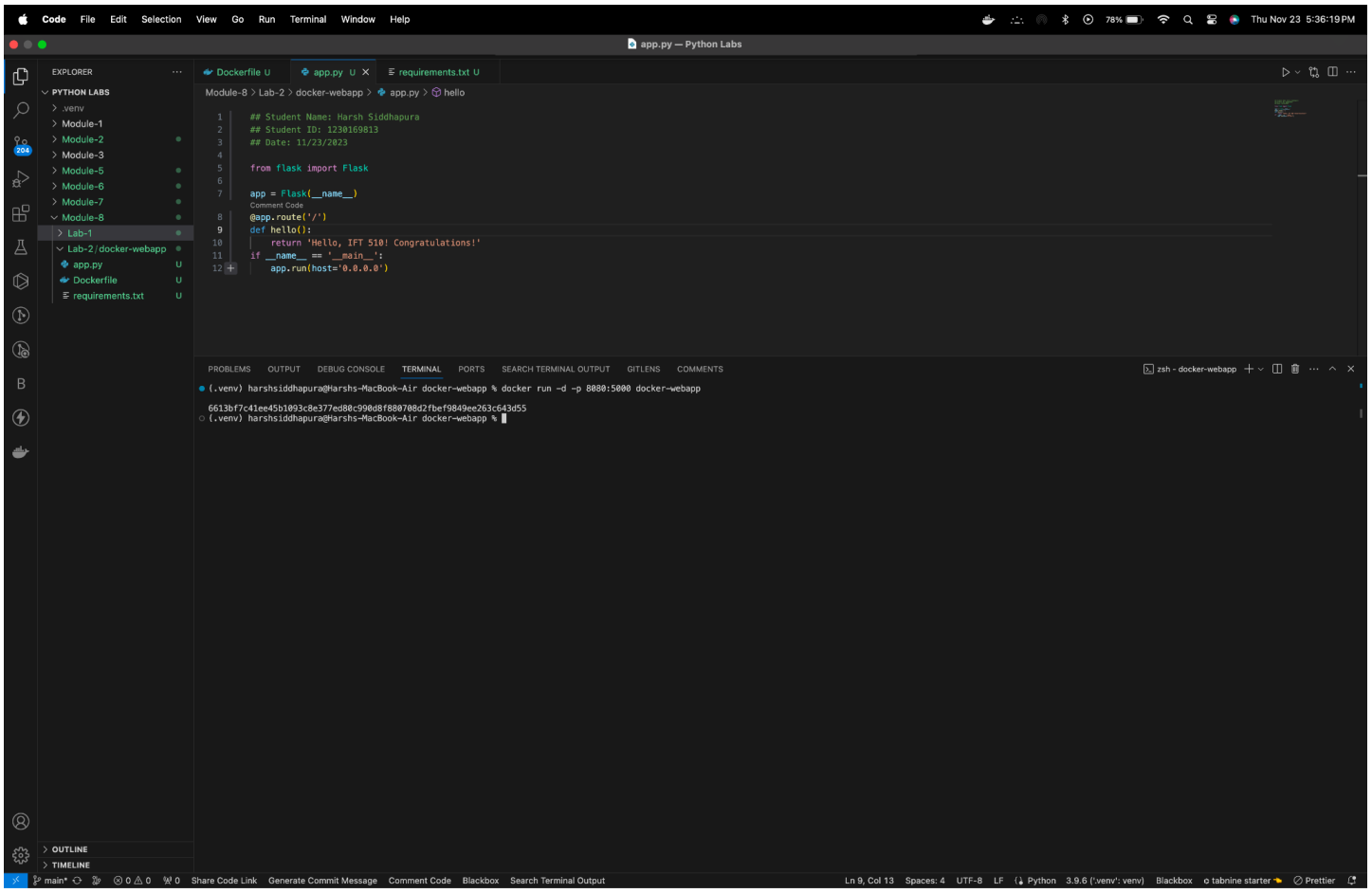
Overall, Docker containers provide a more streamlined, scalable, and portable approach to deploying web applications compared to traditional hosting methods. Understanding and implementing Docker containerization is crucial for modern software development and deployment. Further exploration of container orchestration tools like Docker Swarm or Kubernetes would enhance skills in managing containerized applications at scale. Additionally, exploring Docker Compose for defining multi-container applications would be beneficial for complex deployments. Containerization with Docker is a transformative approach to software deployment, and the skills acquired here serve as a foundation for future endeavors. The ability to containerize applications is a valuable asset in the rapidly evolving landscape of software development and infrastructure management.

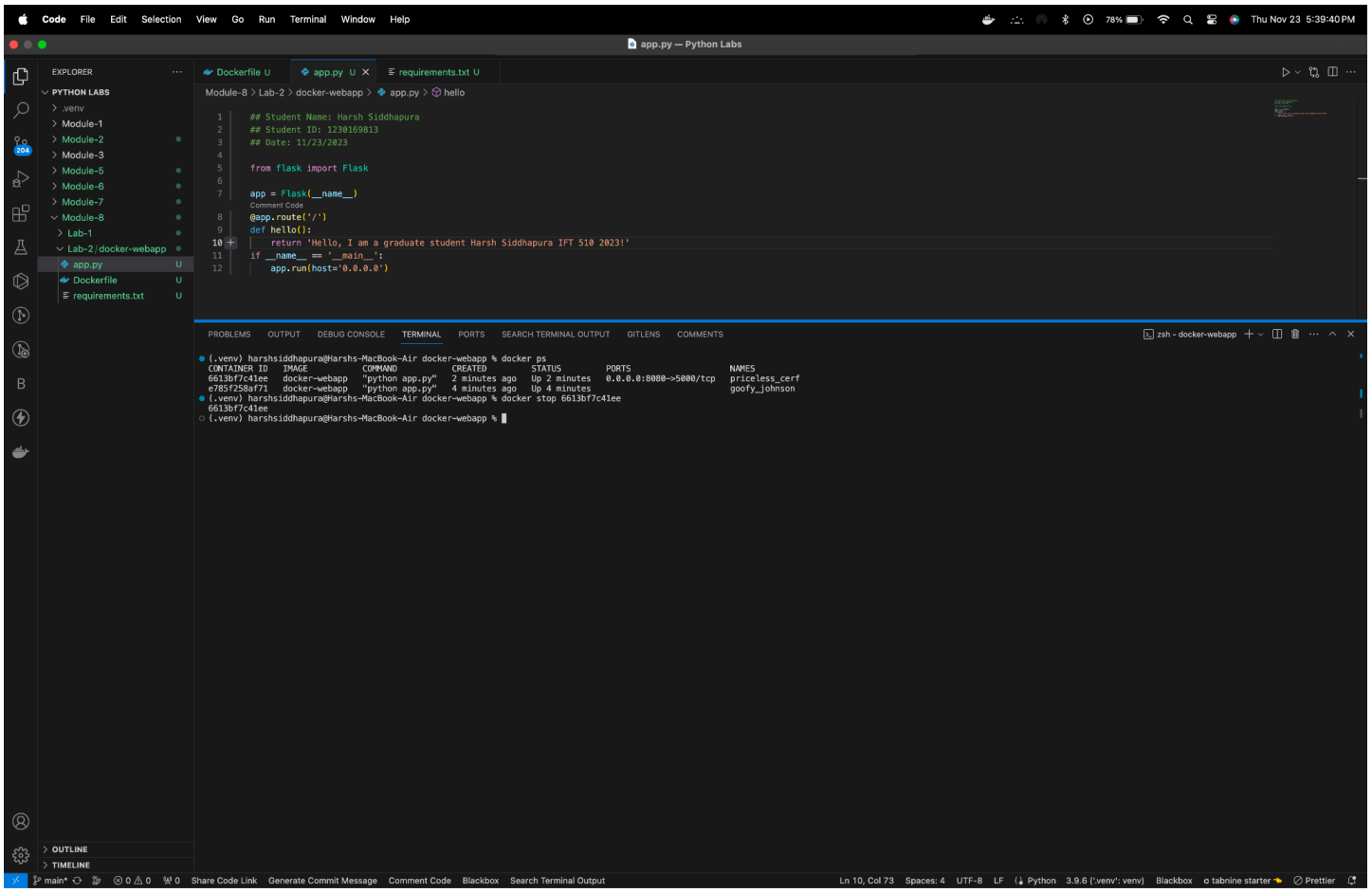
This lab provides a glimpse into the world of DevOps, where the synergy between development and operations optimizes the software delivery lifecycle.This lab builds on fundamental concepts of Docker and containerization. Acknowledgments to the educational resources and documentation that contributed to the understanding and successful completion of the lab. In conclusion, this lab equips individuals with essential skills in containerization, providing a foundation for more advanced topics in DevOps and container orchestration.

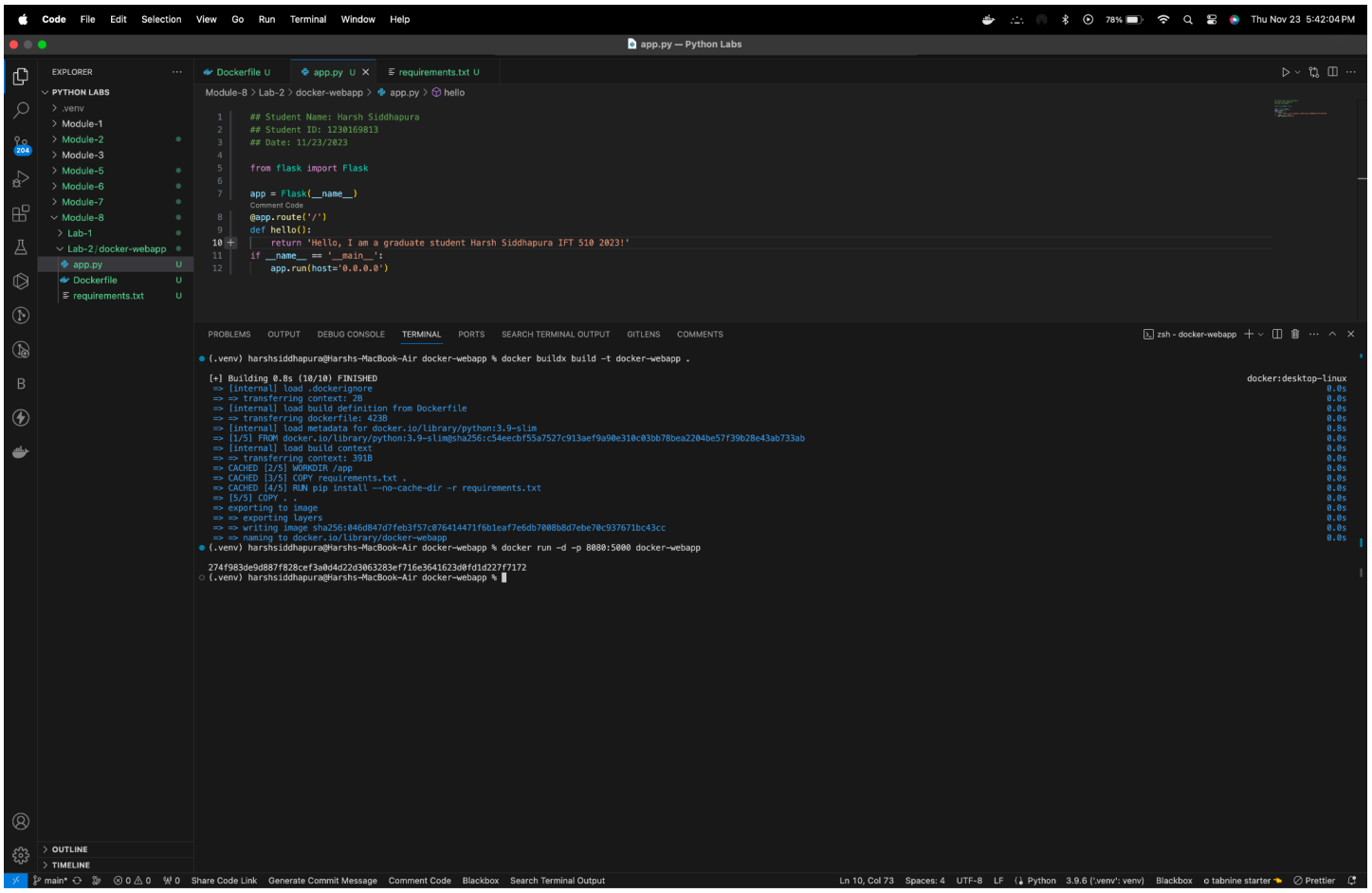
Screenshot

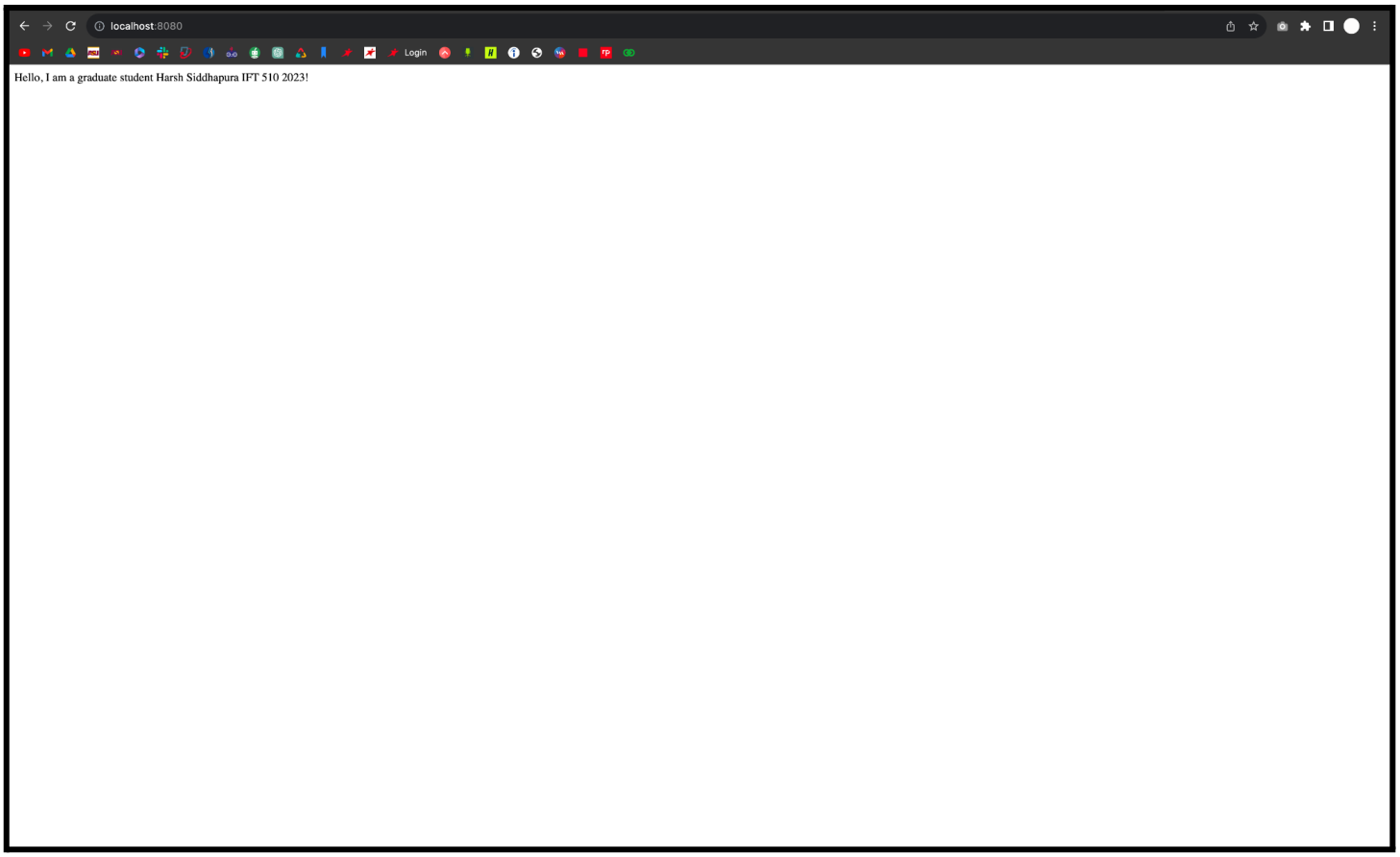

Code File Edit Selection View Go Run Terminal Window Help
Thu Nov 23 5:30:40 PM
Python Labs
EXPLORER
PYTHON LABS
.venv
Module-1
Module-2
Module-3
Module-5
Module-6
Module-7
Module-8
Lab-1
Lab-2/docker-webapp
app.py
Dockerfile
requirements.txt
Dockerfile U
app.py U
requirements.txt U
Module-8 > Lab-2 > docker-webapp > app.py > hello
Student Name: Harsh Siddhapura
Student ID: 1230169813
Date: 11/23/2023
from flask import Flask
app = Flask(__name__)
Comment Code
@app.route('/')
def hello():
return 'Hello, IFT 510! Congratulations!'
if __name__ == '__main__':
app.run(host='0.0.0.0')
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH TERMINAL OUTPUT GITLENS COMMENTS
zsh - docker-webapp
(.venv) harshsiddhapura@Harshs-MacBook-Air docker-webapp % docker buildx build -t docker-webapp .
[+] Building 10.9s (10/10) FINISHED
docker:desktop-linux
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 423B
=> [internal] load metadata for docker.io/library/python:3.9-slim
=> [1/5] FROM docker.io/library/python:3.9-slim@sha256:c54eecbf55a7527c913aef9a90e310c03bb78bea2204be57f39b28e43ab733ab
=> => resolve docker.io/library/python:3.9-slim@sha256:c54eecbf55a7527c913aef9a90e310c03bb78bea2204be57f39b28e43ab733ab
=> => sha256:2c6d21737d8318aa15c4cc838475029a5efc36c0429e3d0da80d970db96d9aaf 29.10MB / 29.10MB
=> => sha256:b4906d053db500056f4d92f6658d5c6d9db90874ff2543087ba68e029166b327 3.33MB / 3.33MB
=> => sha256:72f2a3e07ca3a1643f7a7819930fe05b448932dd03f1ed78931412b6aa42c9f4 11.86MB / 11.86MB
=> => sha256:c54eecbf55a7527c913aef9a90e310c03bb78bea2204be57f39b28e43ab733ab 1.86kB / 1.86kB
=> => sha256:b627b6f4e7ee8df4ceaaf93ced71485f09da466bd22dde74ef687e5dd3ddf42 1.37kB / 1.37kB
=> => sha256:aea69c146f6323387b872142554832beb909a8a26481fb5295575c008874e745 6.93kB / 6.93kB
=> => sha256:14107c73f8cc081c57e1a39e33b7d72a86fa4d2d6b175834cd4a3f3408935b19 244B / 244B
=> => sha256:6c5b3200af26cc5db55b20ac852a05e47b5c19d74787d18d879244ba12364769 3.13MB / 3.13MB
=> => extracting sha256:2c6d21737d8318aa15c4cc838475029a5efc36c0429e3d0da80d970db96d9aaf
=> => extracting sha256:b4906d053db500056f4d92f6658d5c6d9db90874ff2543087ba68e029166b327
=> => extracting sha256:72f2a3e07ca3a1643f7a7819930fe05b448932dd03f1ed78931412b6aa42c9f4
=> => extracting sha256:14107c73f8cc081c57e1a39e33b7d72a86fa4d2d6b175834cd4a3f3408935b19
=> => extracting sha256:6c5b3200af26cc5db55b20ac852a05e47b5c19d74787d18d879244ba12364769
=> [internal] load build context
=> => transferring context: 687B
=> [2/5] WORKDIR /app
=> [3/5] COPY requirements.txt .
=> [4/5] RUN pip install --no-cache-dir -r requirements.txt
=> [5/5] COPY . .
=> exporting to image
=> => exporting layers
=> => writing image sha256:23b861d4e241e4e0b6aaa60f051303aee5706615b222e6ee6a71f018315b399d
=> => naming to docker.io/library/docker-webapp
(.venv) harshsiddhapura@Harshs-MacBook-Air docker-webapp %
OUTLINE
TIMELINE
main*
Share Code Link
Generate Commit Message
Comment Code
Blackbox
Search Terminal Output
Ln 9, Col 13
Spaces: 4
UTF-8
LF
Python
3.9.6 ('.venv': venv)
Blackbox
tabnine starter
Prettier


localhost:8080
Login
Hello, IFT 510! Congratulations!


app.py — Python Labs
Dockerfile U
app.py U
requirements.txt U
Module-8 > Lab-2 > docker-webapp > app.py > hello
Student Name: Harsh Siddhapura
Student ID: 1230169813
Date: 11/23/2023
from flask import Flask
app = Flask(__name__)
@app.route('/')
def hello():
return 'Hello, IFT 510! Congratulations!'
if __name__ == '__main__':
app.run(host='0.0.0.0')
(.venv) harshsiddhapura@Harshs-MacBook-Air docker-webapp % docker run -d -p 8080:5000 docker-webapp
6613bf7c41ee45b1033c8e377ed80c990d8f880708d2fbef9849ee263c643d55
(.venv) harshsiddhapura@Harshs-MacBook-Air docker-webapp %


app.py — Python Labs
Dockerfile U
app.py U
requirements.txt U
Module-8 > Lab-2 > docker-webapp > app.py > hello
Student Name: Harsh Siddhapura
Student ID: 1230169813
Date: 11/23/2023
from flask import Flask
app = Flask(__name__)
@app.route('/')
def hello():
return 'Hello, I am a graduate student Harsh Siddhapura IFT 510 2023!'
if __name__ == '__main__':
app.run(host='0.0.0.0')
(.venv) harshsiddhapura@Harshs-MacBook-Air docker-webapp % docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
6613bf7c41ee docker-webapp "python app.py" 2 minutes ago Up 2 minutes 0.0.0.0:8080->5000/tcp priceless_cerf
e785f258af71 docker-webapp "python app.py" 4 minutes ago Up 4 minutes goofy_johnson
(.venv) harshsiddhapura@Harshs-MacBook-Air docker-webapp % docker stop 6613bf7c41ee
6613bf7c41ee
(.venv) harshsiddhapura@Harshs-MacBook-Air docker-webapp %


Code File Edit Selection View Go Run Terminal Window Help
Thu Nov 23 5:42:04 PM
app.py — Python Labs
EXPLORER
PYTHON LABS
.venv
Module-1
Module-2
Module-3
Module-5
Module-6
Module-7
Module-8
Lab-1
Lab-2/docker-webapp
app.py
Dockerfile
requirements.txt
Dockerfile U
app.py U
requirements.txt U
Module-8 > Lab-2 > docker-webapp > app.py > hello
Student Name: Harsh Siddhapura
Student ID: 1230169813
Date: 11/23/2023
from flask import Flask
app = Flask(__name__)
Comment Code
@app.route('/')
def hello():
return 'Hello, I am a graduate student Harsh Siddhapura IFT 510 2023!'
if __name__ == '__main__':
app.run(host='0.0.0.0')
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH TERMINAL OUTPUT GITLENS COMMENTS
zsh - docker-webapp
(.venv) harshsiddhapura@Harshs-MacBook-Air docker-webapp % docker buildx build -t docker-webapp .
[+] Building 0.8s (10/10) FINISHED
docker:desktop-linux
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 423B
=> [internal] load metadata for docker.io/library/python:3.9-slim
=> [1/5] FROM docker.io/library/python:3.9-slim@sha256:c54eecbf55a7527c913aef9a90e310c03bb78bea2204be57f39b28e43ab733ab
=> [internal] load build context
=> => transferring context: 391B
=> CACHED [2/5] WORKDIR /app
=> CACHED [3/5] COPY requirements.txt .
=> CACHED [4/5] RUN pip install --no-cache-dir -r requirements.txt
=> [5/5] COPY . .
=> exporting to image
=> => exporting layers
=> => writing image sha256:046d847d7feb3f57c0764144471f6b1eaf7e6db7008b8d7ebe70c937671bc43cc
=> => naming to docker.io/library/docker-webapp
(.venv) harshsiddhapura@Harshs-MacBook-Air docker-webapp % docker run -d -p 8080:5000 docker-webapp
274f983de9d887f828cef3a0d4d22d3063283ef716e3641623d0fd1d227f7172
(.venv) harshsiddhapura@Harshs-MacBook-Air docker-webapp %
OUTLINE
TIMELINE
main* 0 0 0 Share Code Link Generate Commit Message Comment Code Blackbox Search Terminal Output
Ln 10, Col 73 Spaces: 4 UTF-8 LF Python 3.9.6 ('.venv': venv) Blackbox tabnine starter Prettier


localhost:8080
Login
Hello, I am a graduate student Harsh Siddhapura IFT 510 2023!