Efficient App Deployment with Dockerfile

ROOM 2

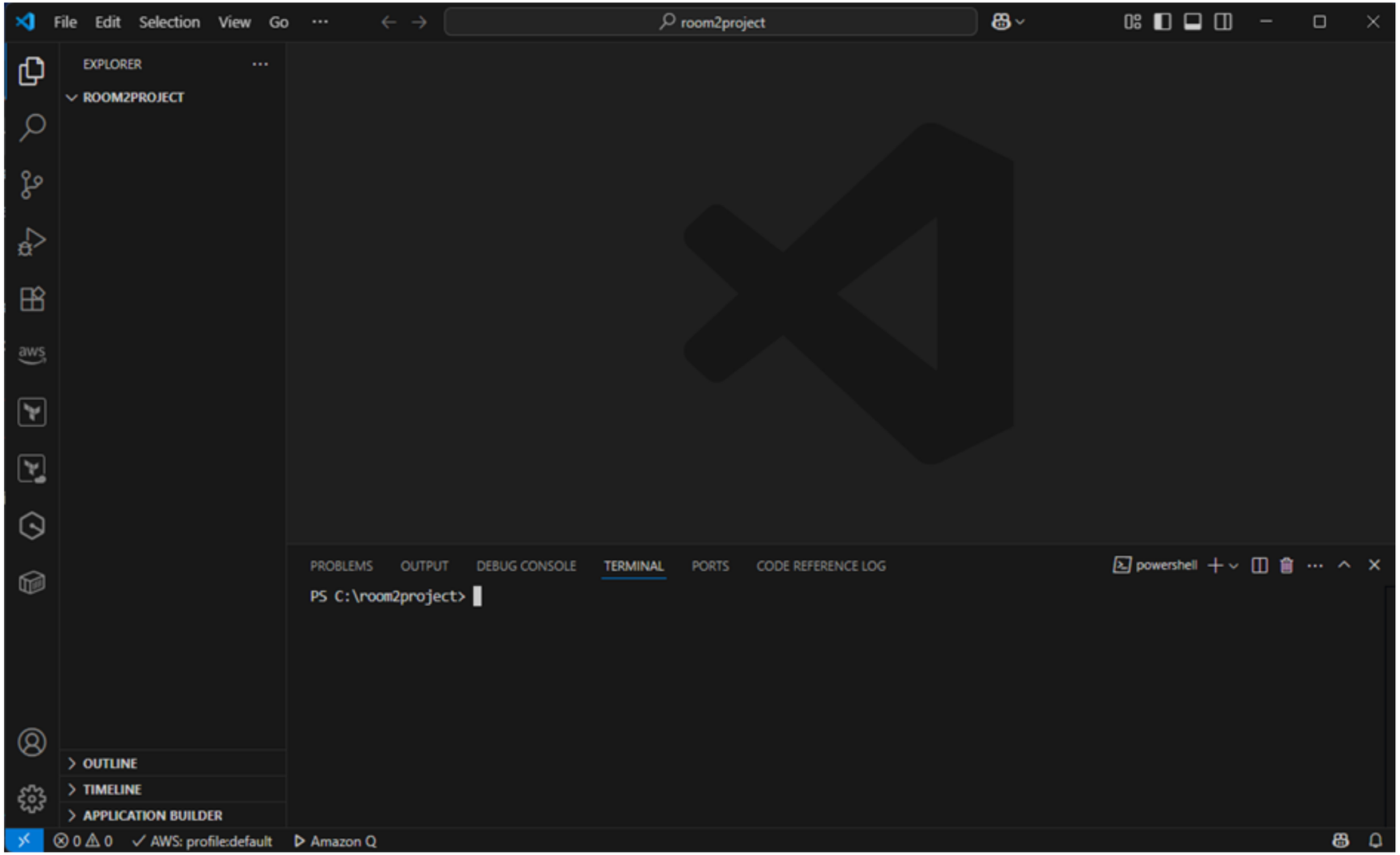
Project Overview

We embarked on a practical learning project to explore Docker fundamentals, container image creation, and scaling strategies. Through collaboration, hands-on work, and shared problem-solving, we deepened our technical understanding and team synergy.

Creating Project Directory

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    CODE REFERENCE LOG

PS C:\Users\oteng> mkdir C:\room2project


    Directory: C:\


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----          7/9/2025   9:11 AM                room2project


PS C:\Users\oteng> cd  C:\room2project
```

CD into Directory

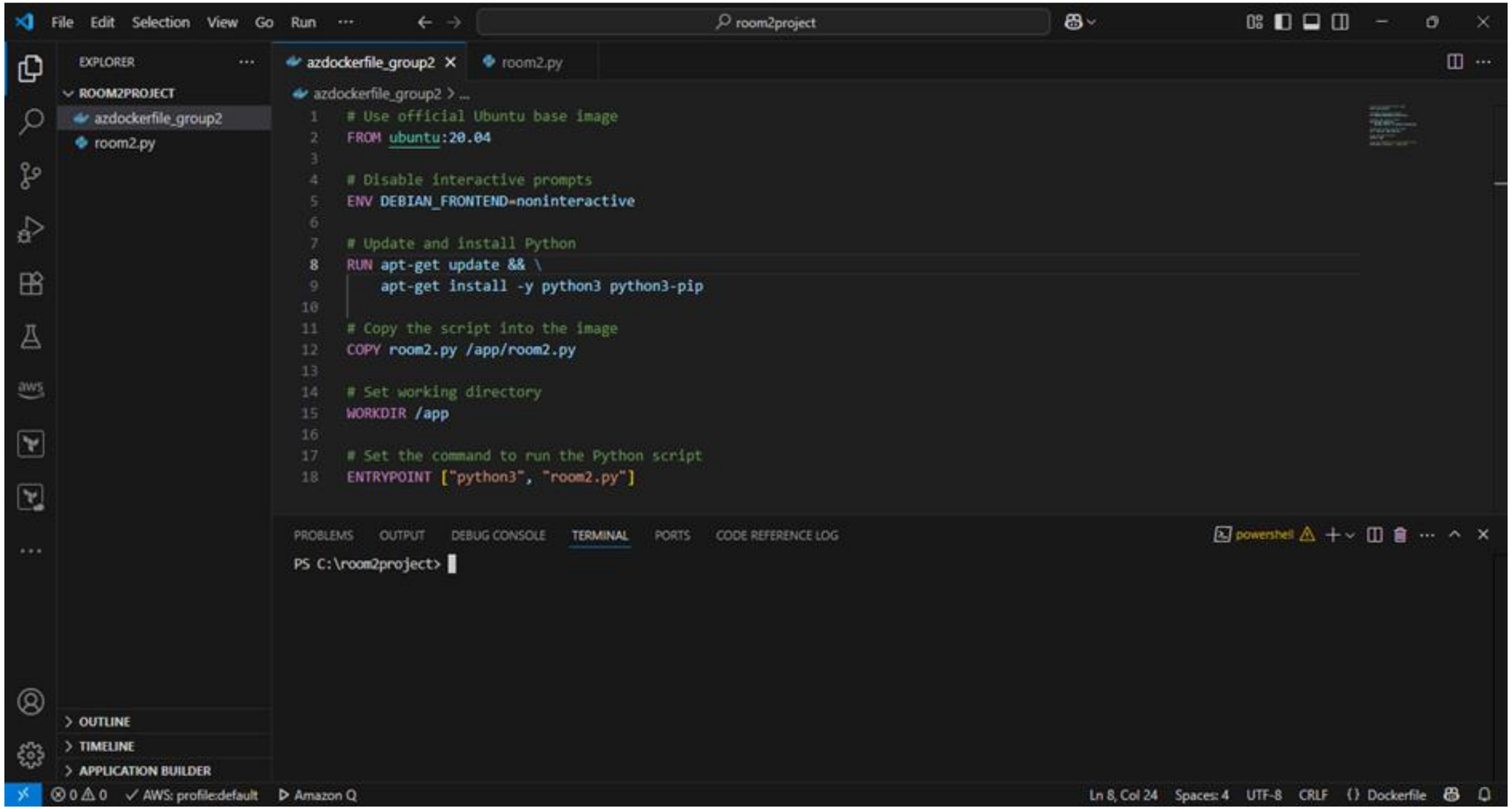


Report on CD into Directory

Created Our Project Directory
We started by organizing our workspace using the command below:

```
mkdir azdockerproject_groupname
cd azdockerproject_group2
```

Code for Docker File

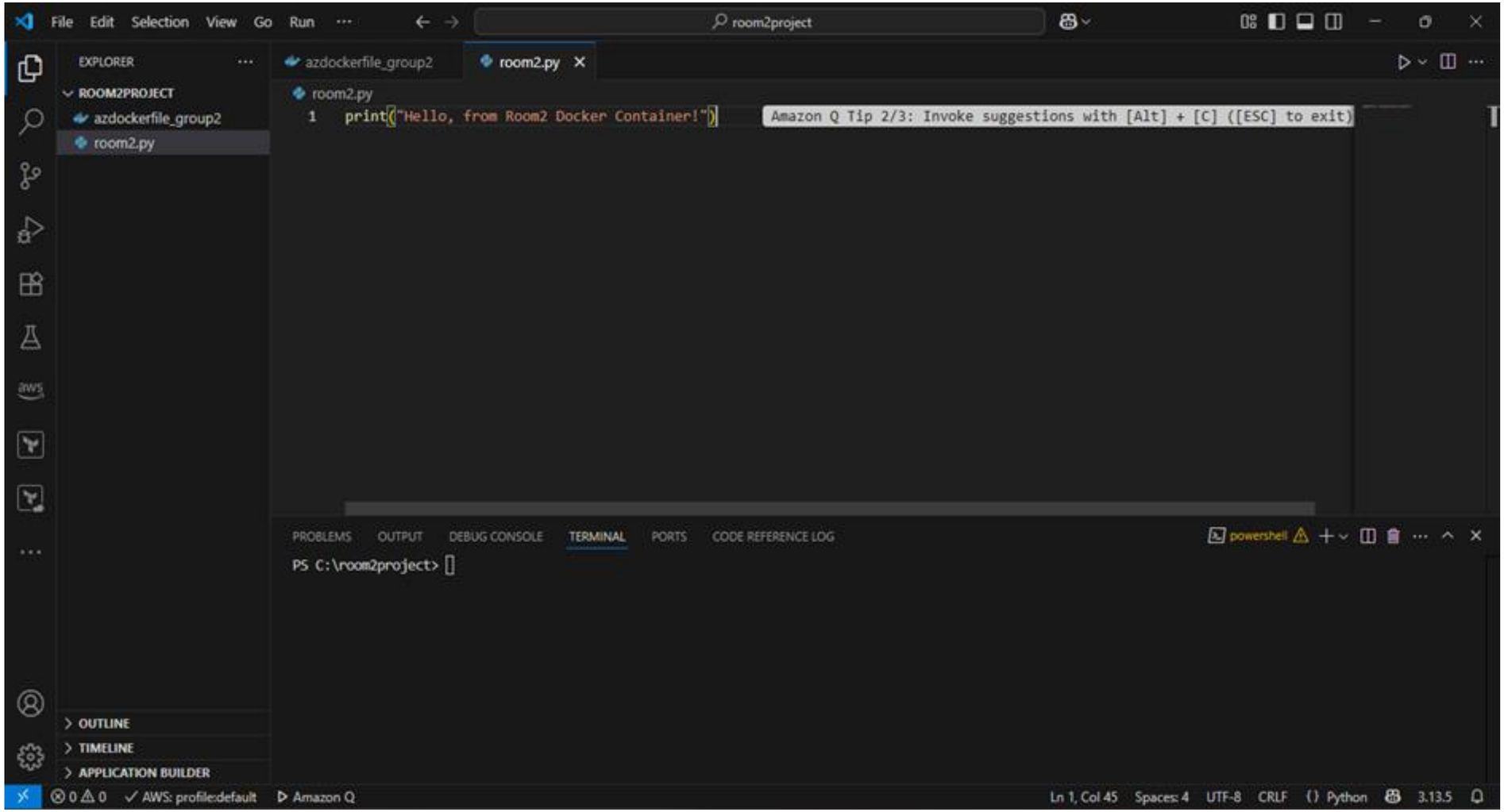


Report on Code for Docker File

We created azdockerfile_group2 using Ubuntu as our base image. We installed Python, added a simple hello.py script, and defined our entry point.

```
FROM ubuntu:20.04
ENV DEBIAN_FRONTEND=noninteractive
RUN apt-get update && apt-get install -y
python3  python3-pip
COPY room2.py /app/room2.py
WORKDIR /app
ENTRYPOINT ["python3", "room2.py"]
```

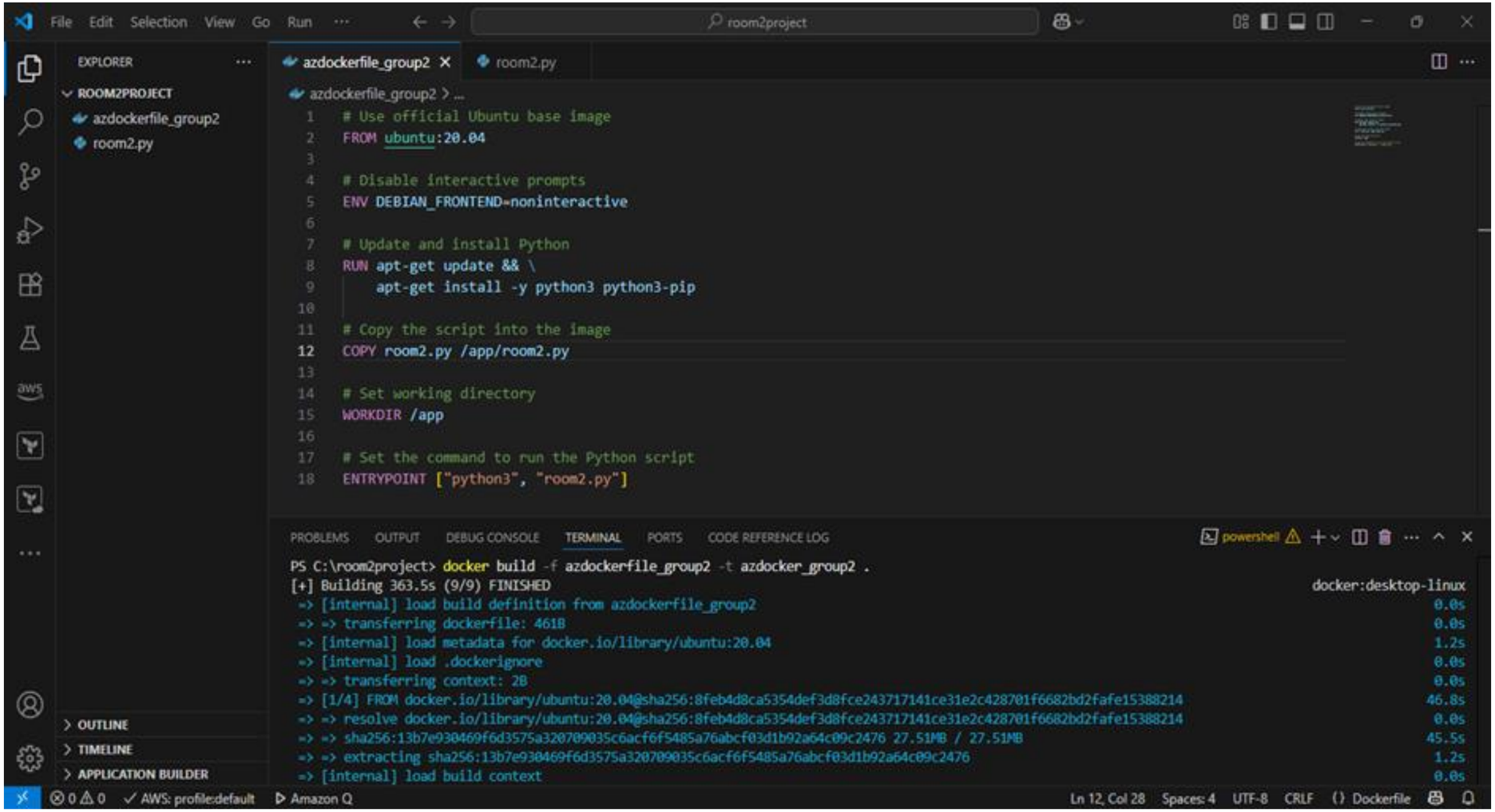
Code for Python File



Report on Python File

A simple python script that prints to screen "Hello from room2 Docker container!"

Building Docker Image



Report on Docker Image

We built our image and ran it to confirm functionality:

```
docker build –t azdockerimage_group2 –f azdockerfile_group2
docker run azdockerimage_group2
```

We saw our python script run successfully, printing: "Hello from the Docker container!"

Docker Container

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   CODE REFERENCE LOG                                  powershell

 => [4/4] WORKDIR /app                                                                                      0.1s
 => exporting to image                                                                                     21.3s
 => => exporting layers                                                                                    16.3s
 => => exporting manifest sha256:a0b2a5d888eb6e619726b2ff8c1ee0ab996308b61d96926ced8f4e1601350c14           0.0s
 => => exporting config sha256:d7a8d1618af519e2941e475496a6b8ecb64cfe97fbee2295e49cf9063716c73a             0.0s
 => => exporting attestation manifest sha256:79c39d855ee79cd181e061dd16607c481680913a16c152d8d72a5737f62628ec  0.0s
 => => exporting manifest list sha256:900510507659ee230b4bd1e590cc9002f16fa72c952afdd2fc5034a1fdd6c19c      0.0s
 => => naming to docker.io/library/azdocker_group2:latest                                                  0.0s
 => => unpacking to docker.io/library/azdocker_group2:latest                                               4.7s
PS C:\room2project> docker run azdocker_group2
Hello, from Room2 Docker Container!
PS C:\room2project>

OUTLINE
TIMELINE
APPLICATION BUILDER

0 ⚠ 0   ✓ AWS: profile:default   ▷ Amazon Q   Ln 12, Col 28   Spaces: 4   UTF-8   CRLF   {} Dockerfile
```

Report on Docker Container

Installed Python Inside a Running Container

To simulate image patching, we launched a container, manually installed Python, and committed it to a new image.

```
docker run -it ubuntu bash
apt-get update && apt-get install -y python3
docker commit
```

Building Ubuntu:latest

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   CODE REFERENCE LOG                    docker

PS C:\room2project> docker run -it ubuntu:latest bash
root@38430dfadb09:/# apt-get update && apt-get install -y python3
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:2 http://archive.ubuntu.com/ubuntu noble InRelease [256 kB]
Get:3 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [1123 kB]
Get:4 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:6 http://archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [331 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [23.0 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [1735 kB]
Get:9 http://archive.ubuntu.com/ubuntu noble/main amd64 Packages [1808 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1237 kB]
Get:11 http://archive.ubuntu.com/ubuntu noble/universe amd64 Packages [19.3 MB]
75% [11 Packages 16.4 MB/19.3 MB 85%]                                          602 kB/s 13s

Amazon Q   Ln 1, Col 45   Spaces: 4   UTF-8   CRLF   {} Python   3.13.5
```

Checking for Container; Ubuntu:latest

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS CODE REFERENCE LOG

```
PS C:\room2project> docker ps -a
CONTAINER ID   IMAGE            COMMAND              CREATED          STATUS                     PORTS     NAMES
38430dfadb09   ubuntu:latest    "bash"               4 minutes ago    Exited (0) 28 seconds ago            charming_lumiere
b05f6490658c   azdocker_group2  "python3 room2.py"   11 minutes ago   Exited (0) 11 minutes ago            gracious_goldwasser
ae34438cf528   alpine           "sh"                 17 hours ago     Exited (0) 17 hours ago              myapps
874da79c617e   alpine           "sh"                 17 hours ago     Exited (1) 17 hours ago              apps
975dc2f5e2e9   d219c9ca46b1     "python app.py"      17 hours ago     Exited (0) 17 hours ago              xenodochial_sammet
d28fb6b3eb70   d219c9ca46b1     "python app.py"      17 hours ago     Exited (0) 17 hours ago              elated_wozniak
f5f5c47dcca3   d219c9ca46b1     "python app.py"      18 hours ago     Exited (0) 18 hours ago              blissful_ardinghelli
eadaea643dad   hello-world      "/hello"             19 hours ago     Exited (0) 19 hours ago              objective_hawking
77a0b4d2d36b   ubuntu           "/bin/bash"          41 hours ago     Exited (130) 18 hours ago            keen_kepler
62486092fd75   hello-world      "/hello"             41 hours ago     Exited (0) 41 hours ago              eloquent_jepsen
eff26264bbea   ubuntu           "/bin/bash"          42 hours ago     Exited (127) 42 hours ago            gifted_mirzakhani
81a82968e9a0   hello-world      "/hello"             42 hours ago     Exited (0) 42 hours ago              priceless_bohr
d7785788e31c   hello-world      "/hello"             42 hours ago     Exited (0) 42 hours ago              stupefied_khayyam
4f70aeaf9098   hello-world      "/hello"             42 hours ago     Exited (0) 42 hours ago              dazzling_diffie
PS C:\room2project> docker commit 38430dfadb09 python_installed_image
```

Amazon Q Ln 1, Col 45 Spaces: 4 UTF-8 CRLF {} Python 3.13.5

Python_Installed_Image Created!

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   CODE REFERENCE LOG                                  powershell

62486092fd75   hello-world      "/hello"          41 hours ago    Exited (0) 41 hours ago          eloquent_jepsen
eff26264bbea   ubuntu           "/bin/bash"       42 hours ago    Exited (127) 42 hours ago        gifted_mirzakhani
81a82968e9a0   hello-world      "/hello"          42 hours ago    Exited (0) 42 hours ago          priceless_bohr
d7785788e31c   hello-world      "/hello"          42 hours ago    Exited (0) 42 hours ago          stupefied_khayyam
4f70aeaf9098   hello-world      "/hello"          42 hours ago    Exited (0) 42 hours ago          dazzling_diffie
PS C:\room2project> docker commit 38430dfadb09 python_installed_image
sha256:df4f1303141948f4ecb0a07ec995a1516a092a09c7ce651d66e35eda2cf7bc35
PS C:\room2project> docker images
REPOSITORY              TAG       IMAGE ID       CREATED          SIZE
python_installed_image  latest    df4f13031419   10 minutes ago   264MB
azdocker_group2         latest    900510507659   28 minutes ago   672MB
myapp                   text      d219c9ca46b1   18 hours ago     206MB
mypythonapp             latest    d219c9ca46b1   18 hours ago     206MB
ubuntu                  latest    440dcf6a5640   2 weeks ago      117MB
alpine                  latest    8a1f59ffb675   5 weeks ago      12.8MB
hello-world             latest    940c619fbd41   5 months ago     20.4kB
PS C:\room2project>

Amazon Q                                   Ln 1, Col 45   Spaces: 4   UTF-8   CRLF   {} Python   3.13.5
```

Report on Checking for Container; Ubuntu:latest

Opened a bash shell from an ubuntu docker image with the latestubuntu version available.

```
docker run –it ubuntu:latest bash
```

To simulate image patching, we launched a container, manually installed Python, and committed it to a new image.

```
docker run -it ubuntu bash
apt-get update && apt-get install -y python3
docker commit <container_id> patched-python-image
```

Auto-Scaling Discussion

Docker by itself does not support auto-scaling, but it can be integrated with orchestration tools like Kubernetes to achieve it.

Auto-scaling ensures your application can handle varying loads by automatically increasing or decreasing the number of containers based on resource usage (like CPU or memory).

The best and most widely used method is: Docker + Kubernetes + Horizontal Pod Autoscaler (HPA)

Challenges

Challenge	Impact	Solution
Dockerfile syntax errors	Delayed builds	Used CLI feedback and rebuilt with --no-cache
Python not executing	Container permissions	Added chmod to Dockerfile and verified execution path
Version coordination	Conflicts in edits	Adopted Git workflow for syncing changes
Scaling complexity	Conceptual confusion	Reviewed AWS docs and used visual aids for clarity

Conclusion

We built, we broke, we rebuilt and we did it together. From setting up directories to running containers, every step reinforced our technical capabilities and teamwork.

The experience was empowering, insightful, and a major milestone in our Docker journey.

Meet Our Best Team

- Clement Owusu Bempah
- Emmanuel Gyau
- Fahad Mohammed Gibrine
- Macleana Mensah Oteng
- Mariama Faisal
- Osman Abdul
- Samuel Kofi Asare Dwumah
- Violette Naa Adoley Allotey

Group 2

Thank You

For Your Attention