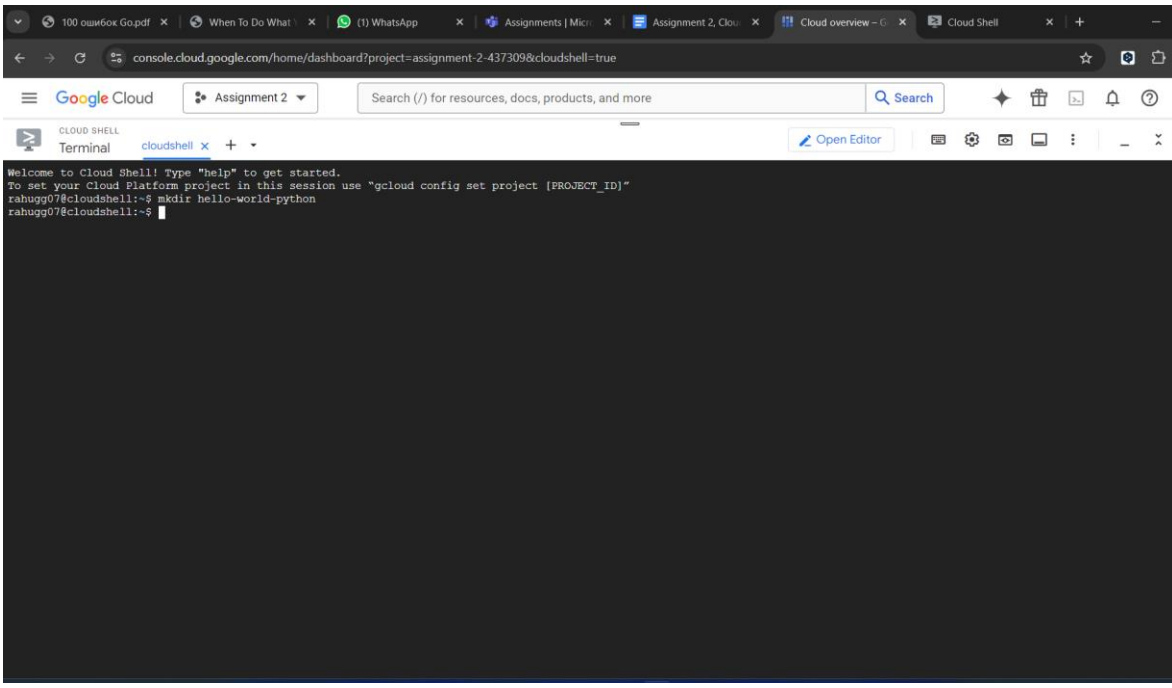
Assignment 2, Cloud Application Development

Exercise 1: Google App Engine

Objective: Deploy a simple web application on Google App Engine.

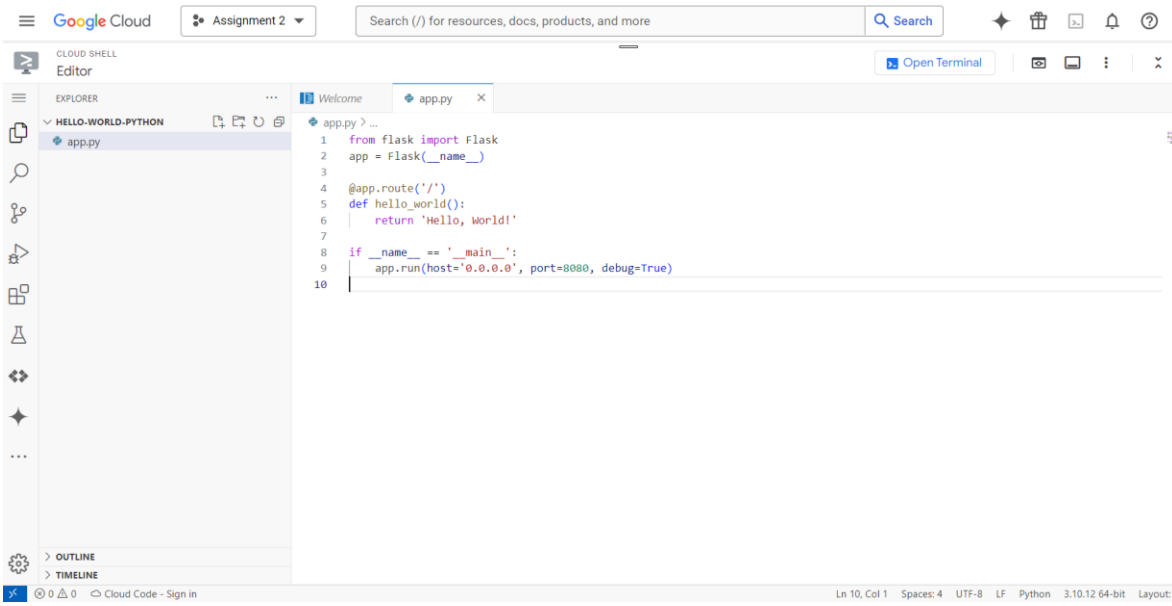
Instructions:

1. **Setup**:
    - Ensure you have a Google Cloud account.x`
    - Install the Google Cloud SDK on your local machine.
2. **Create a Project**:
    - Create a new project in the Google Cloud Console.
      I created a folder for my project assignment 2 via google cloud console



3. **Prepare the Application**:
    - Write a simple "Hello, World!" web application using Python (Flask).
      I created a file named app.py and wrote this code for printing the hello

world



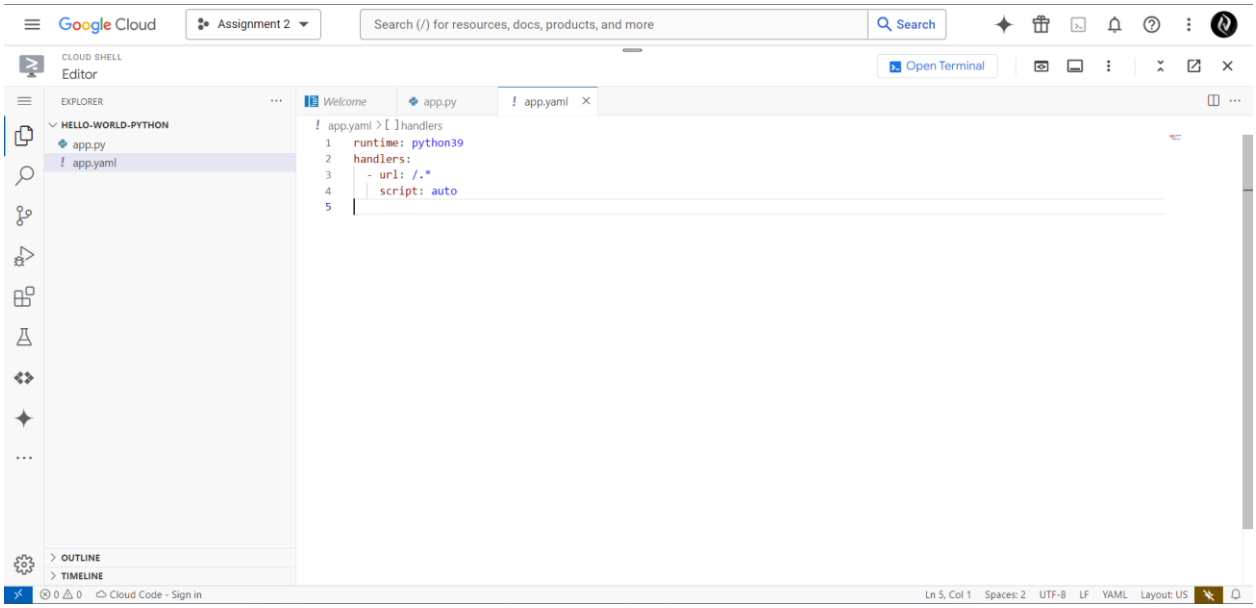
app.py:

```
from flask import Flask
app = Flask(__name__)


@app.route('/')
def hello_world():
    return 'Hello, World!'


if __name__ == '__main__':
    app.run(host='0.0.0.0', port=8080, debug=True)
```

-

4. **Create the App Engine Configuration**:

I created app.yaml and pasted the following code:



Create a app.yaml file with the following content:

runtime: python39

handlers:

- url: /.*

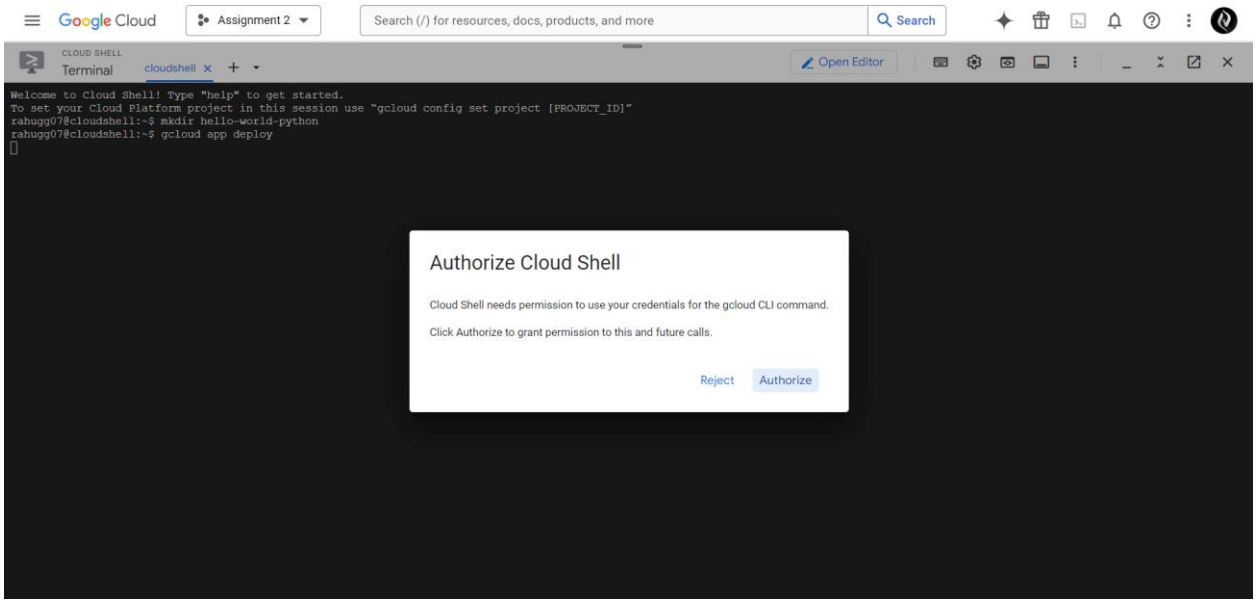
  script: auto

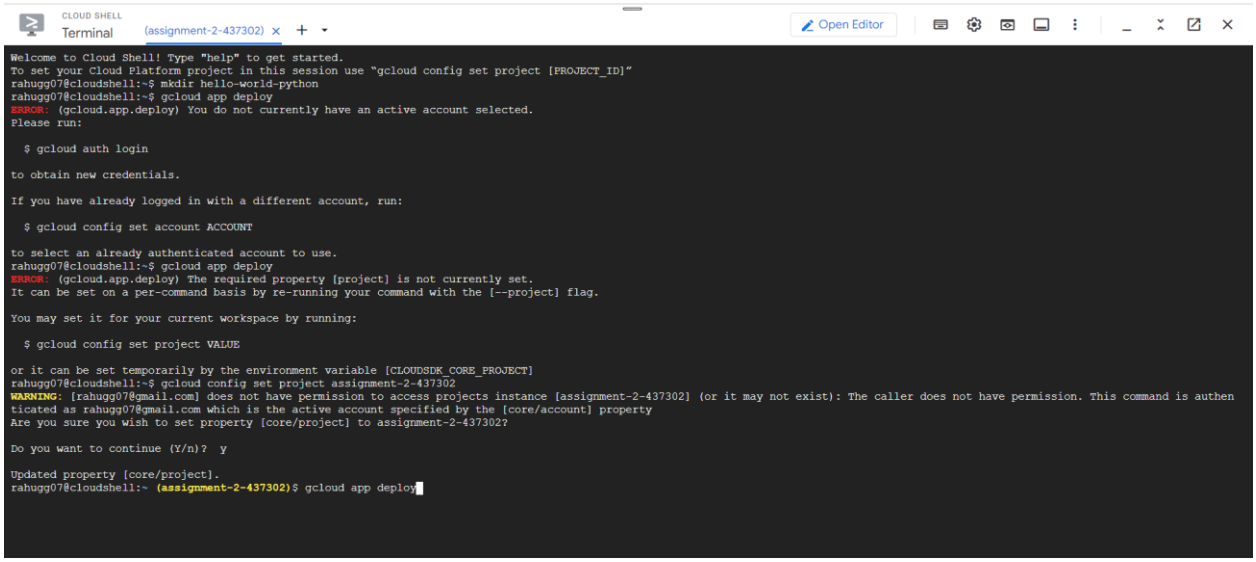
  ○

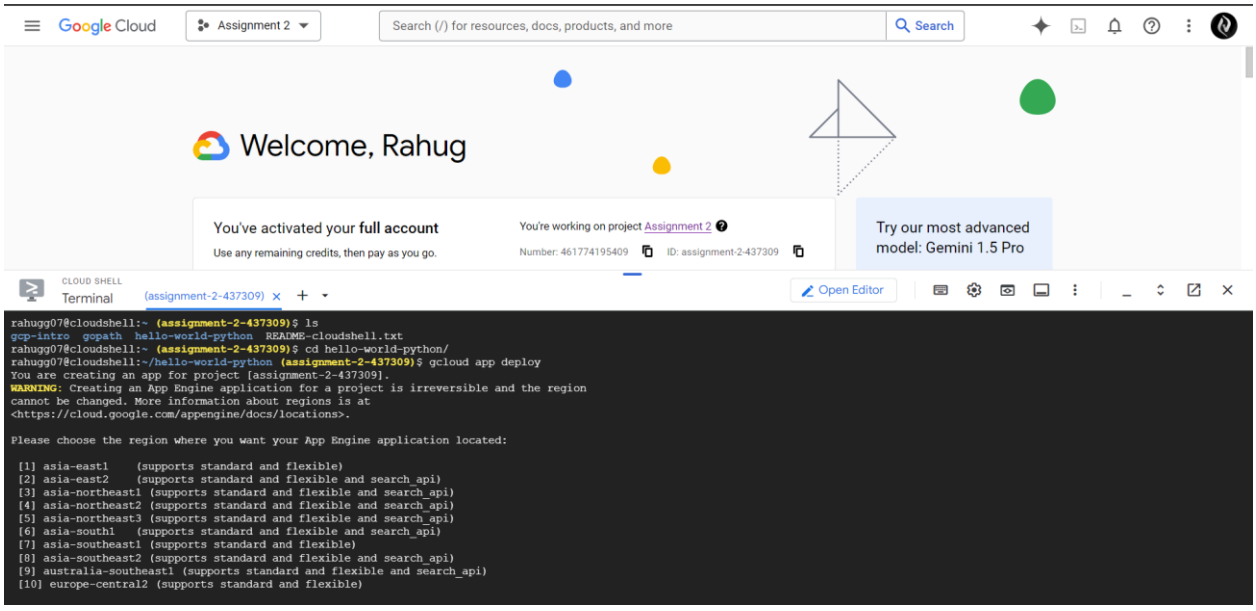
5. **Deploy the Application**:

Use the following command to deploy the application to Google App Engine:

I wrote the command, and the next step was indeed to set the default project which I did in the next screenshots, and also I activated billing account in order to deploy it

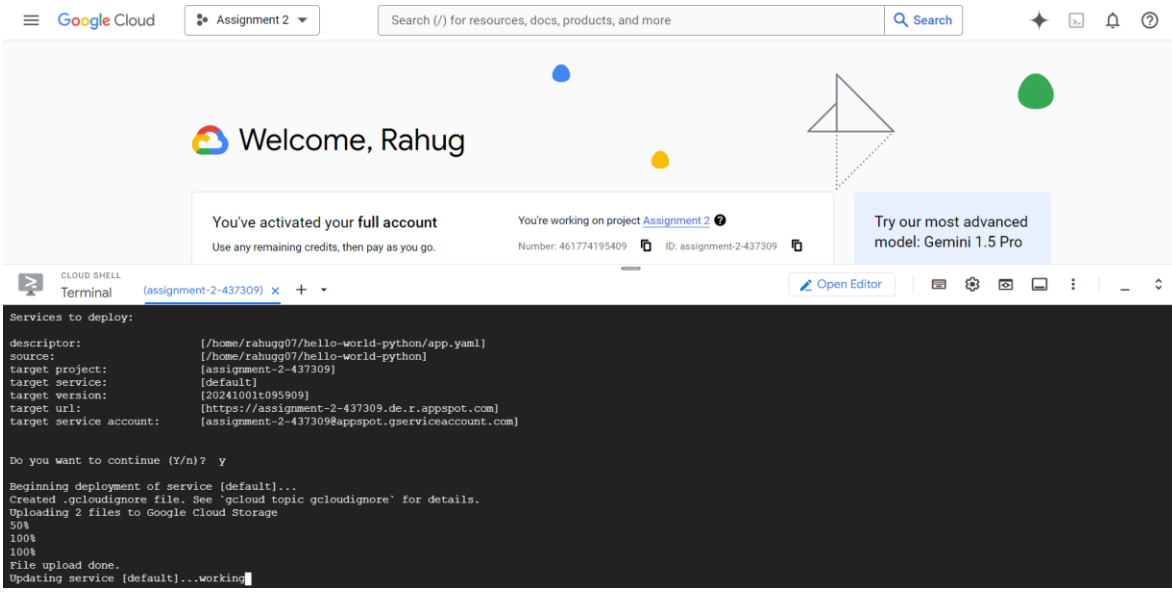


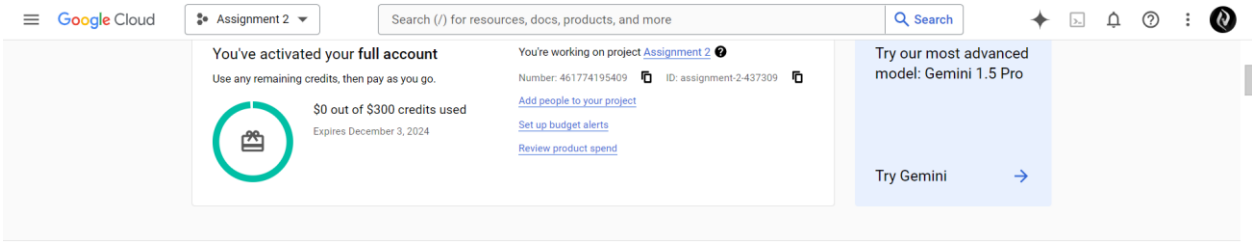


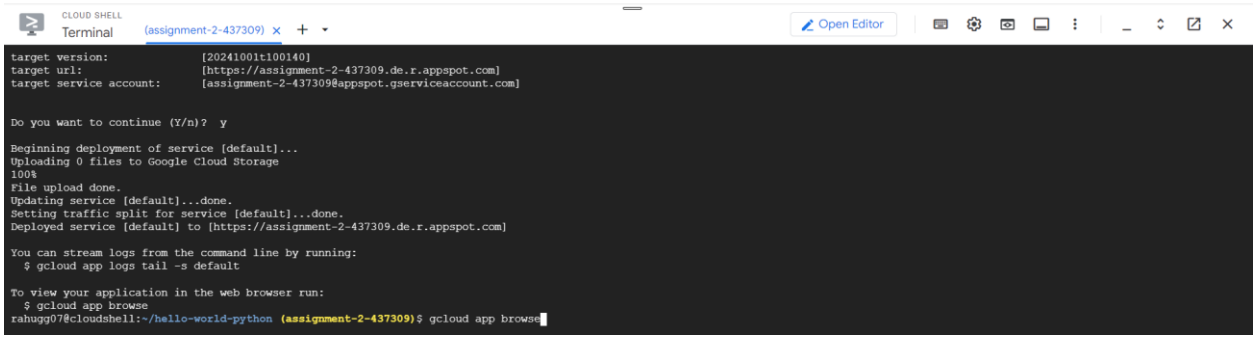


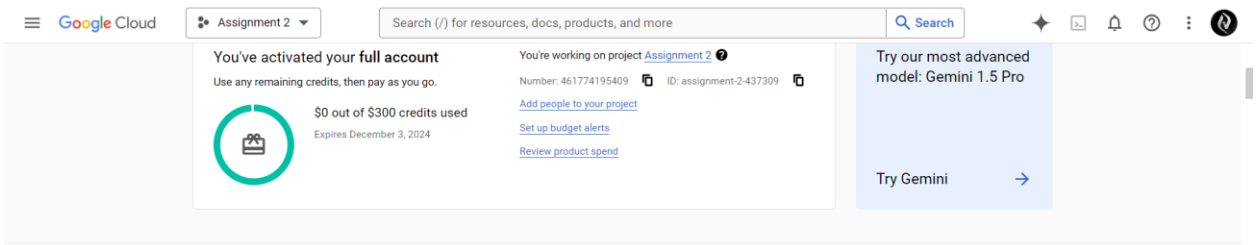
gcloud app deploy

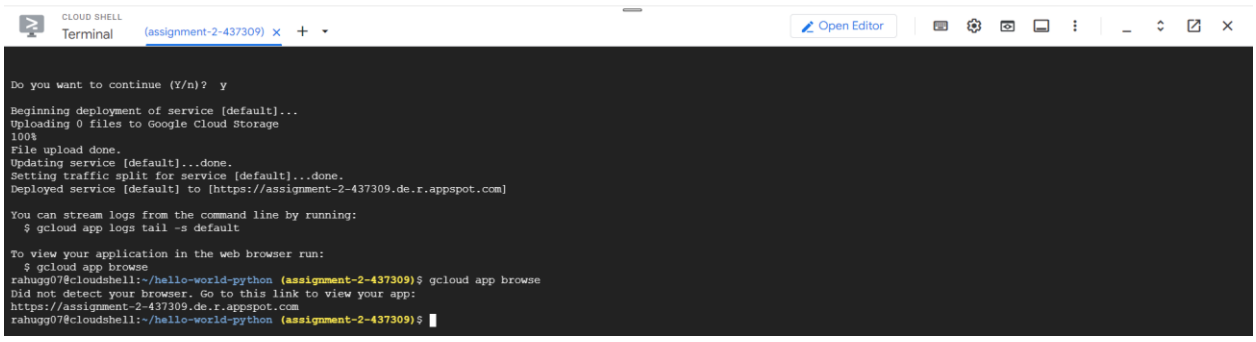
6. **Access the Application**:
    - Once deployed, access your application using the URL provided by Google App Engine.







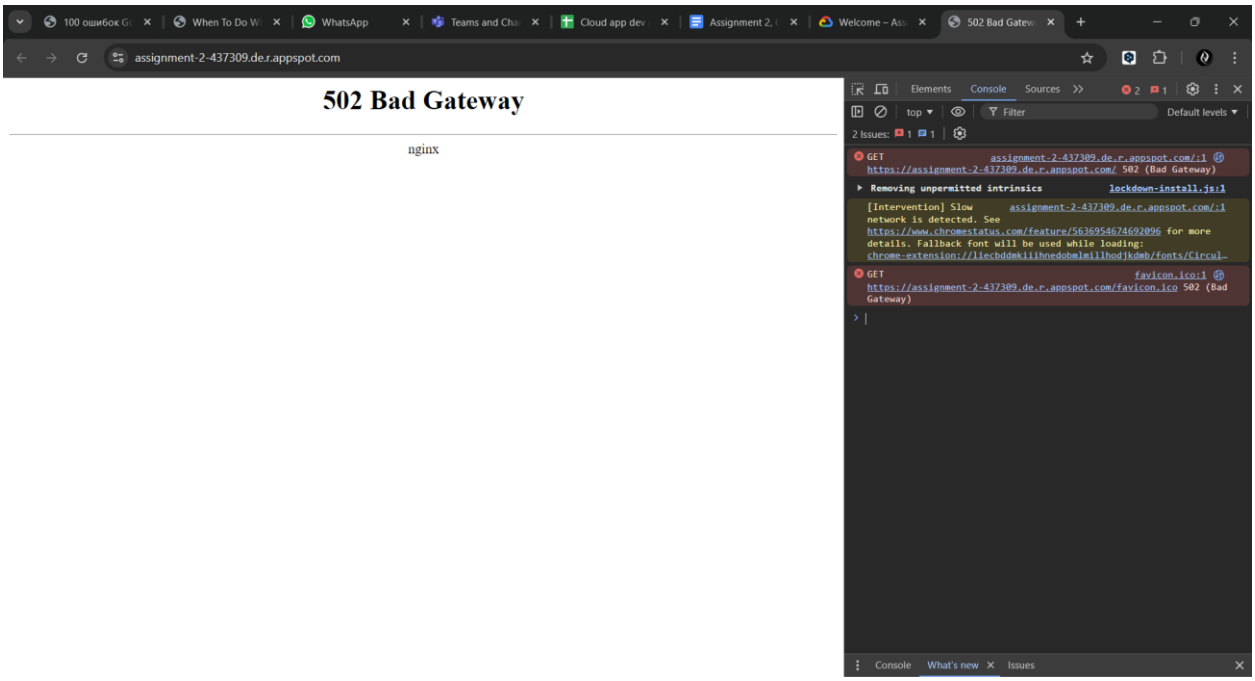




Deliverables:

- A deployed web application on Google App Engine.

- A screenshot of the running application.



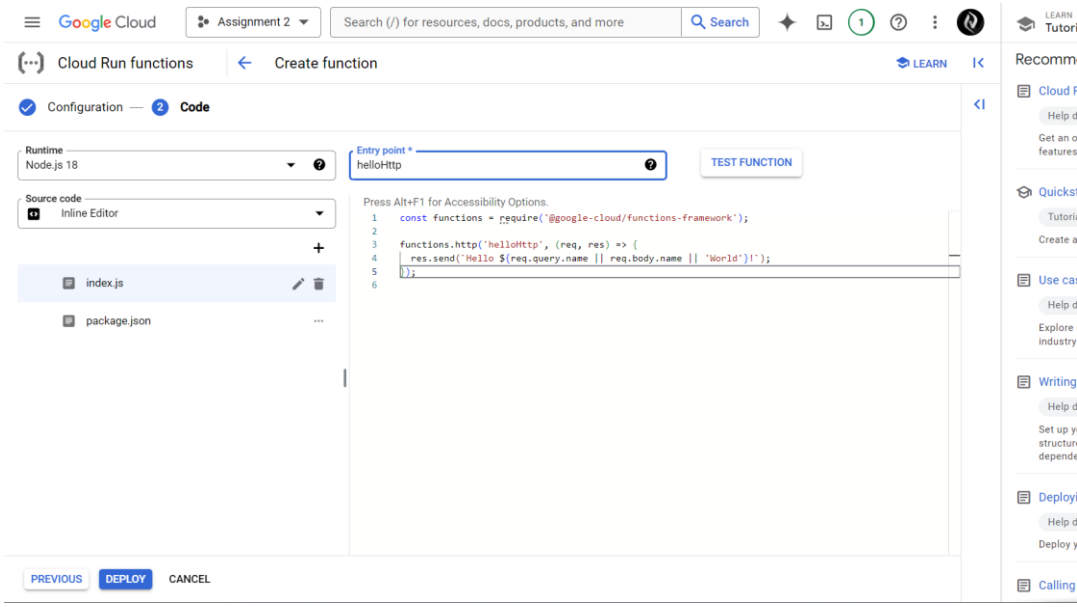
Exercise 2: Building with Google Cloud Functions

Objective: Create a Google Cloud Function that processes HTTP requests.

Instructions:

1. **Setup**:
    - Ensure you have a Google Cloud account.
    - Install the Google Cloud SDK on your local machine.
2. **Create a Function**:
    - Create a new Google Cloud Function using the following configuration:
        - **Name**: helloWorldFunction
        - **Trigger**: HTTP
        - **Runtime**: Node.js 18 (or another supported runtime)

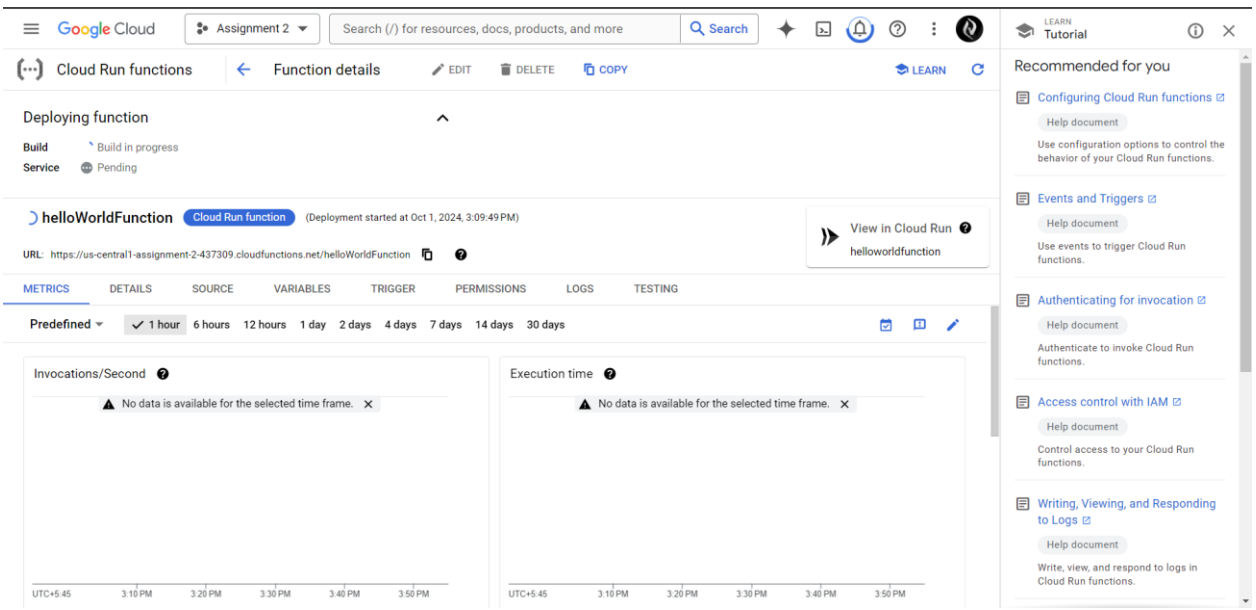
- **Entry Point**: helloWorld



3. **Write the Code**:
    - Write a simple function that returns "Hello, World!" when accessed via HTTP.

Example index.js:

```
exports.helloWorld = (req, res) => {
  res.send('Hello, World!');
};
```

- 

4. **Deploy the Function**:

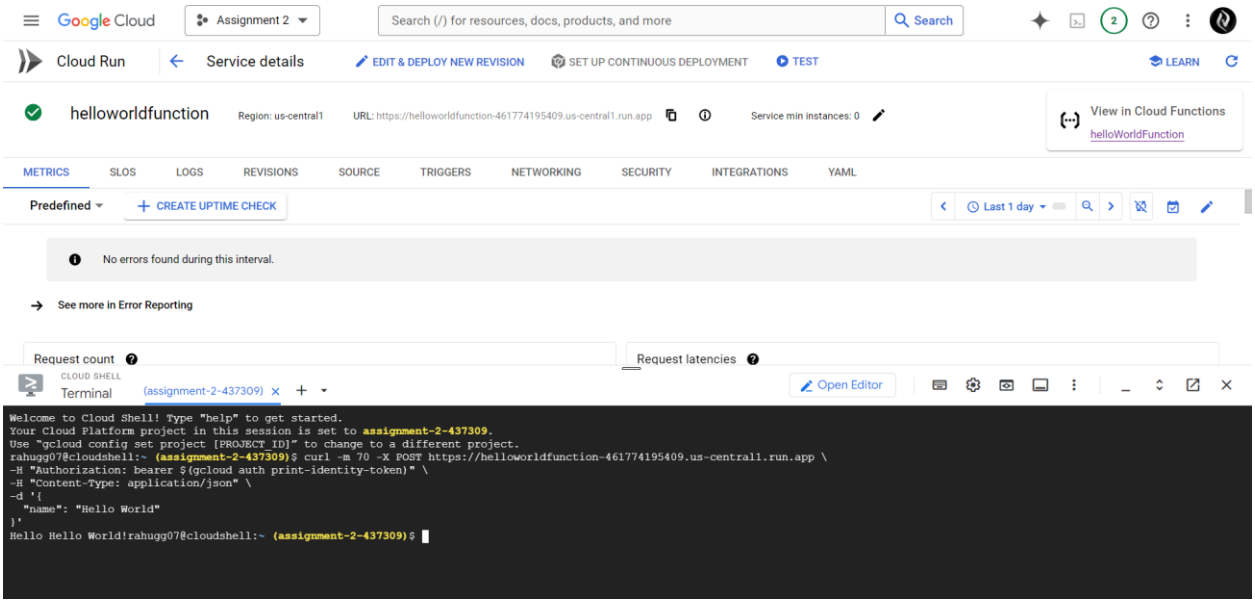
Use the following command to deploy the function:

gcloud functions deploy helloWorldFunction --runtime nodejs18 --trigger-http

5. **Invoke the Function**:
    - Once deployed, use the provided URL to test the function by accessing it via a web browser or curl.

Deliverables:

- A deployed Google Cloud Function.
- A screenshot showing the response from the function.



- **Text RESULT:** I created a cloud function helloworld and then configured the node.js file to be able to see the message hello world as a result via curl I got a response from my cloud function

---

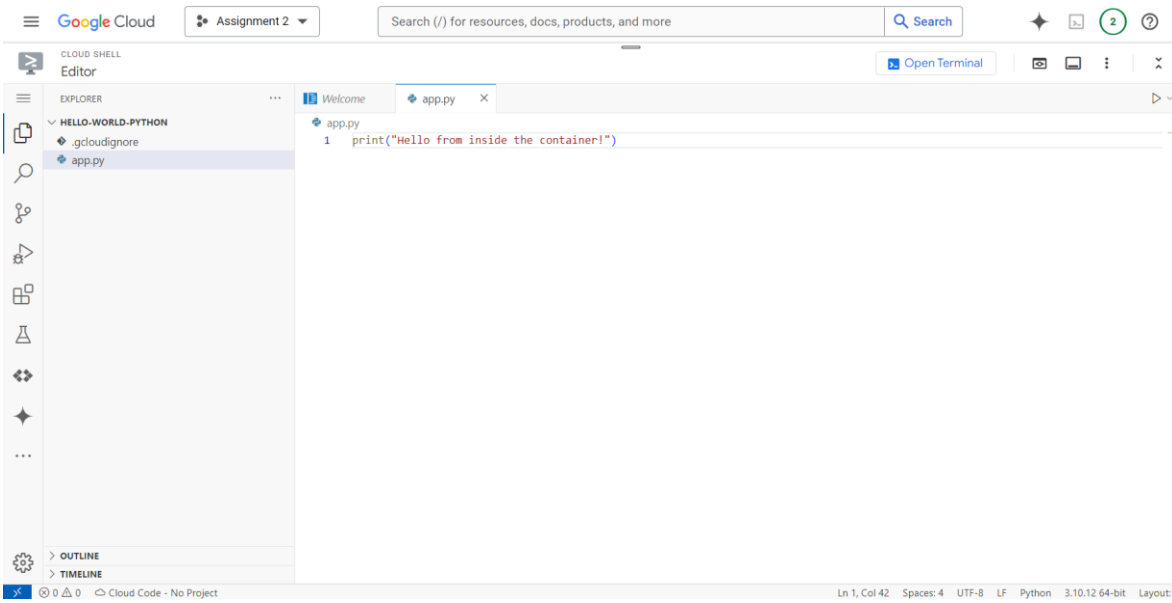
Exercise 3: Containerizing Applications

Objective: Containerize a simple application using Docker.

Instructions:

1. **Setup**:
    - Ensure Docker is installed on your local machine.
2. **Create a Simple Application**:

- Write a simple Python application.



Example app.py:

print("Hello from inside the container!")

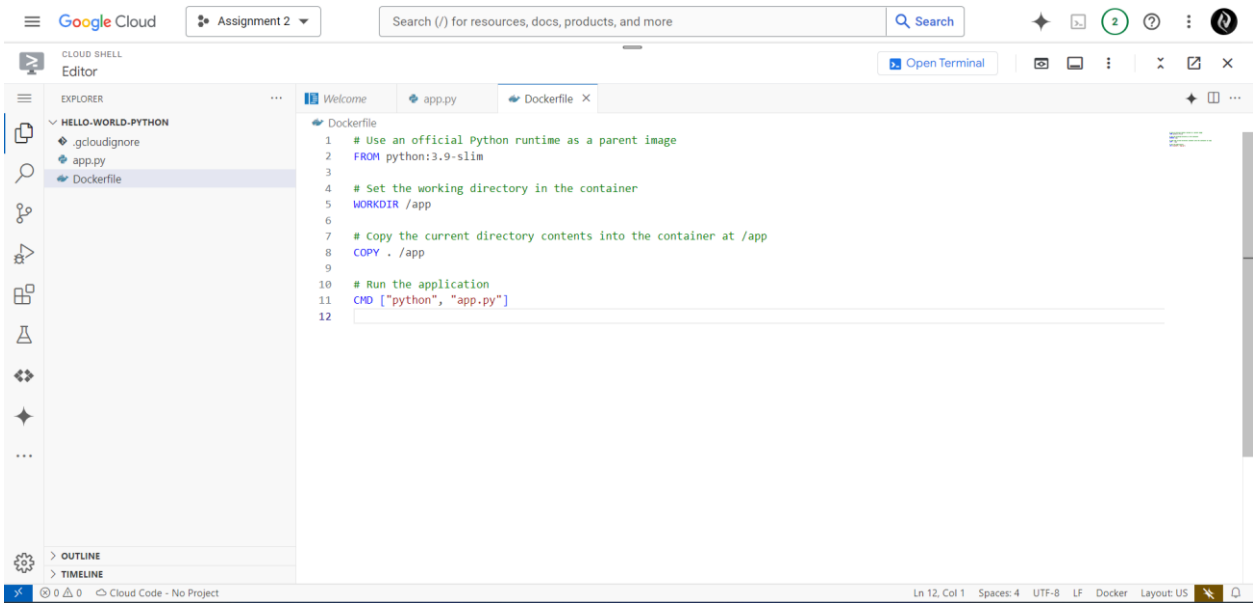
-

3. **Create a Dockerfile**:
    - Write a Dockerfile to containerize the application.

Example Dockerfile:

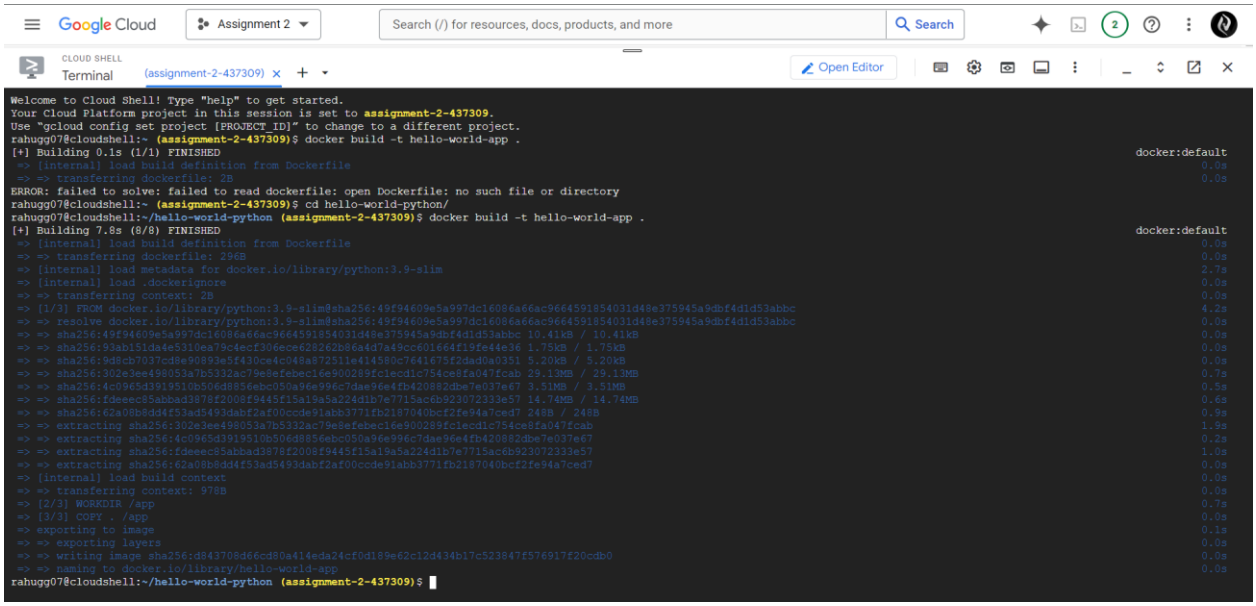
```
# Use an official Python runtime as a parent image
FROM python:3.9-slim

# Set the working directory in the container
WORKDIR /app

# Copy the current directory contents into the container at /app
COPY . /app

# Run the application
CMD ["python", "app.py"]
```

4. **Build the Docker Image**:



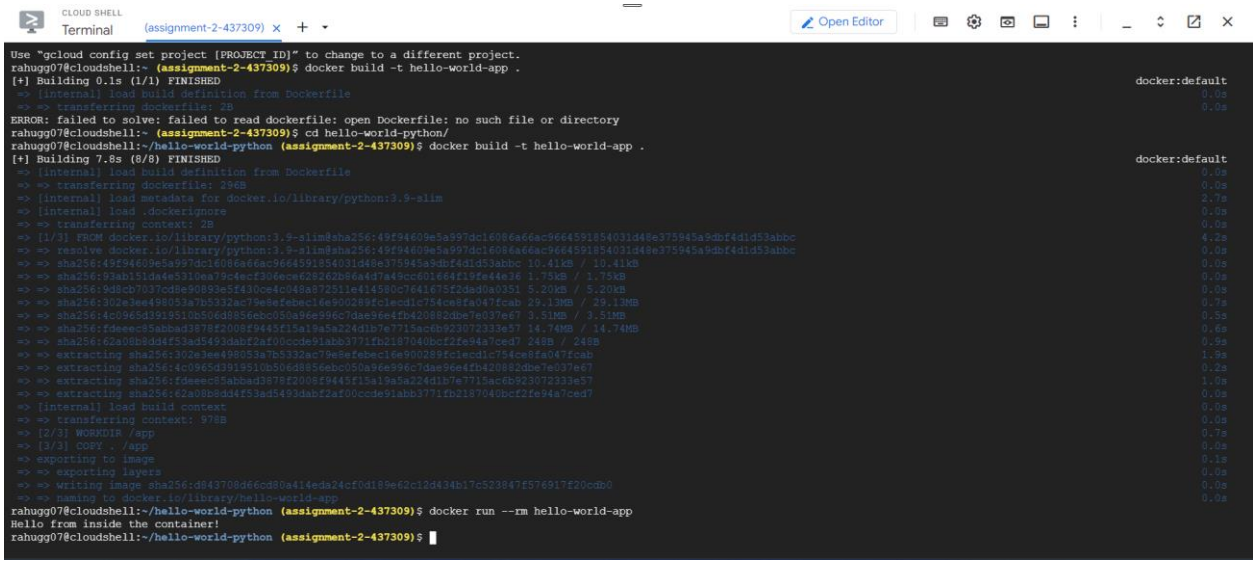
Build the Docker image using the following command:

docker build -t hello-world-app .



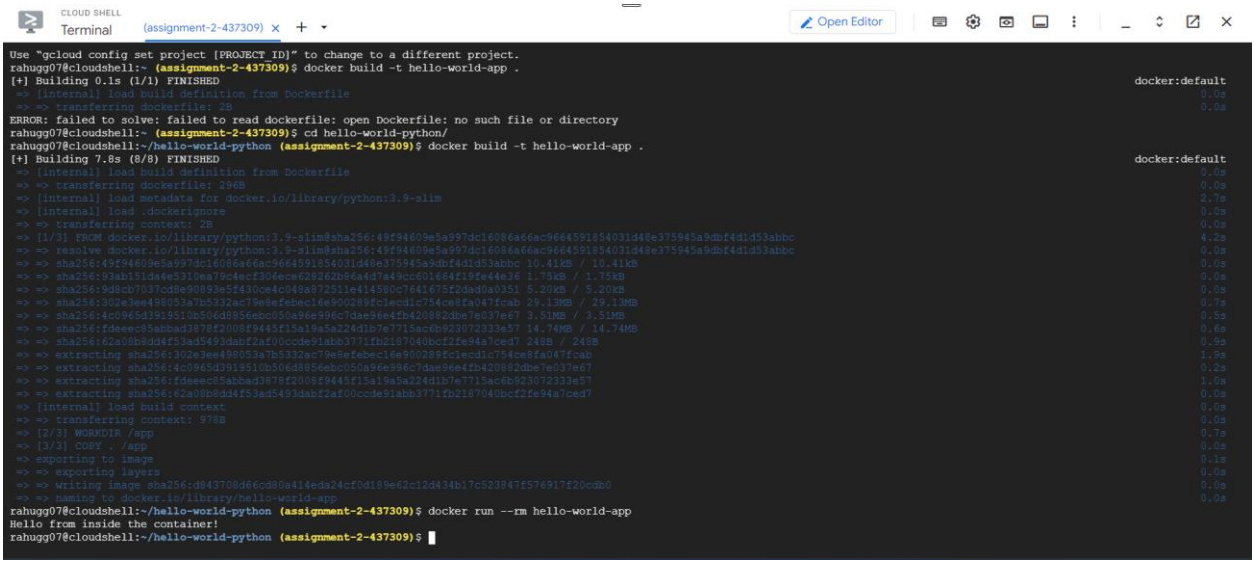
5. **Run the Docker Container**:

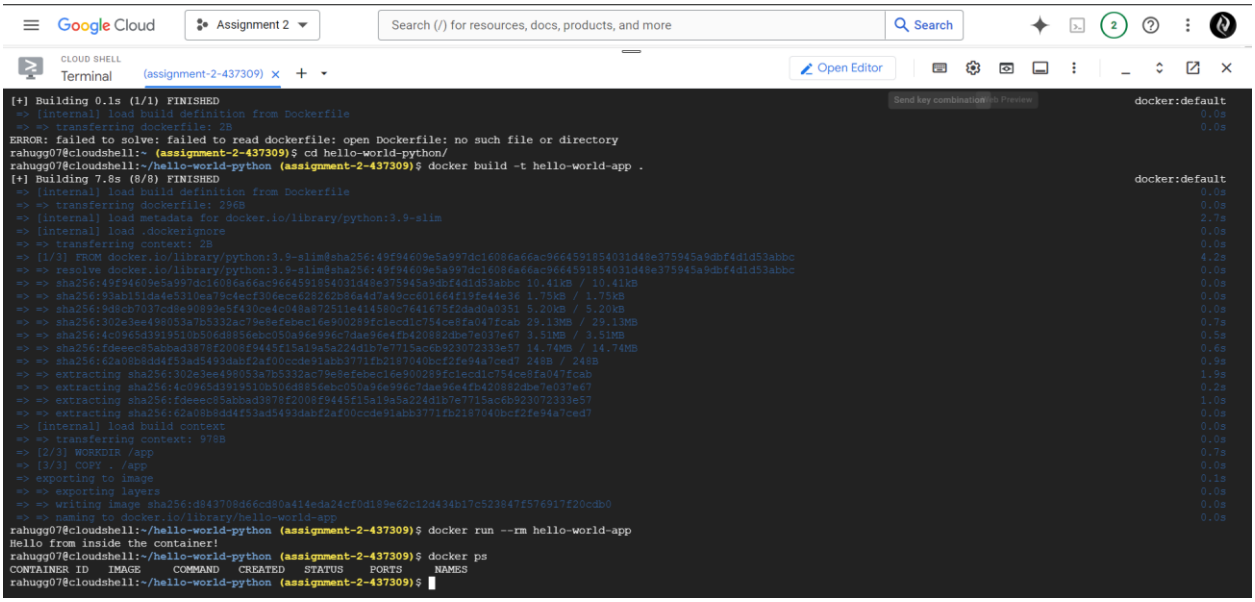
Run the container using the following command:
docker run --rm hello-world-app



Deliverables:

- A Docker image that runs a simple application.
- A screenshot of the container output showing "Hello from inside the container!"





Text RESULT: I created an app.py and wrote the docker file for configurations then via command docker and then I built docker image and ran it through the docker command as a result there is a message in the console “Hello from inside the container!”