Experiment – 6 in Cloud Computing (PVP23)

A basic Flask application for Google App Engine (Standard environment) requires a root directory containing at least three files: **`main.py`**, **`app.yaml`**, and **`requirements.txt`**. For serving web pages and static assets, the standard Flask structure with `templates/` and `static/` folders is used.

Recommended Project Structure

```
your-project-folder/
├── main.py
├── app.yaml
├── requirements.txt
├── static/
│   ├── css/
│   ├── js/
│   └── img/
└── templates/
    ├── index.html
    └── base.html
```

Essential Files and Directories

main.py

- The application's entry point. App Engine expects to find a WSGI-compatible `app` object in this file by default. This file contains your core Flask application logic and routing.

- **Code Snippet (`main.py`)**:

```python
from flask import Flask, render_template
app = Flask(__name__)
@app.route("/")
def home():
    return render_template("index.html")

if __name__ == "__main__":
    # Used for local development
    # app.run(host="0.0.0.0", port=8080, debug=True)
    app.run(host="127.0.0.1", port=8080, debug=True)
```

app.yaml

The configuration file for Google App Engine. It tells App Engine which runtime to use and how to handle requests.

- **Code Snippet (`app.yaml`)**:

```yaml
runtime: python312 # Use a supported Python version, e.g., python312
instance_class: F1 # Smallest instance class, good for simple apps
automatic_scaling:
  min_idle_instances: 0 # Scales to zero when idle to save costs
```

- For serving static files efficiently, you can add handlers:

```yaml
handlers:
- url: /static
  static_dir: static/
- url: /.*
  script: auto # Routes everything else to the Flask app
```

requirements.txt

- This file lists all the Python dependencies your application needs. App Engine reads this file during deployment and installs the listed packages.

- **Code Snippet (`requirements.txt`)**:

```
Flask
# Add other dependencies like requests, PyMySQL, etc..
```

- **`templates/`**: A standard Flask directory where HTML templates (using Jinja2) are stored. The `render_template` function looks for files in this folder.

- **`static/`**: A standard Flask directory for static assets like CSS files, JavaScript files, and images.

Deployment

Once the structure is in place and the Google Cloud SDK is installed and configured, you can deploy your application by navigating to your project's root directory in the terminal and running the following command:

bash

```
gcloud app deploy
```

Installing the Google Cloud CLI

To install the Google Cloud CLI (formerly Cloud SDK) on Windows, the recommended method is to use the interactive installer, which simplifies the process of setting up prerequisites like Python.

Step-by-Step Installation Guide

1. **Download the installer:**

- Navigate to the official Google Cloud CLI download page.
- Download the installer file (`GoogleCloudCliInstaller.exe`) for your version of Windows (64-bit is most common).

2. **Run the installer:**

- Locate the downloaded file and double-click to run it.
- If prompted by User Account Control, click **Yes** to allow the application to make changes.

3. **Follow the installation wizard:**

- **Welcome Screen:** Click **Next** to begin the installation.
- **License Agreement:** Read and **accept** the license agreement.
- **Installation Location:** Choose an installation directory, or leave the default path as is.
- **Python Prerequisite:** The installer can bundle Python 3 automatically. It is recommended to leave the option to install the bundled Python checked, as the gcloud CLI requires a supported Python version (3.10-3.14).
- **Components and Shortcuts (Optional):** You can optionally choose to install beta commands or create desktop/start menu shortcuts.

4. **Complete the installation:**

- Click **Install** and wait for the process to complete. This may take a few minutes.
- Once the installation is complete, click **Next**, then **Finish**. By default, the installer will automatically open a terminal window and run `gcloud init` to configure your setup.

Configuration and Initialization

After the physical installation is complete, you need to configure the SDK to link it with your Google Cloud account and project.

1. **Run `gcloud init`:**

- If the installer didn't automatically run it, open a new command prompt or the **Google Cloud CLI Shell** from your Start menu and type `gcloud init`.

2. **Authorize your account:**

- The command prompt will ask you to log in. It will open a web browser to a Google sign-in page.
- Sign in with your Google account and grant the necessary permissions to access your Google Cloud resources.
- A message in the browser will confirm successful authentication, and you can close the window.

3. **Configure project and region:**

- Back in the command prompt, you will be prompted to select a Google Cloud project from a list of your available projects.
- You may also be asked to configure a default Compute Engine region and zone (this is optional but recommended).

Verification

To verify the installation, open a new terminal window and run:

```
gcloud version
```

This command should display the installed gcloud CLI version and other component details, confirming a successful installation. You can now use the `gcloud`, `gsutil`, and `bq` command-line tools to interact with Google Cloud services.

Execution

To execute a basic Flask application on Google App Engine (GAE), you need to set up your project files, test the application locally, configure a Google Cloud project, and then deploy using the `gcloud` CLI.

Prerequisites

- A Google Cloud project with billing enabled.
- The Google Cloud SDK (gcloud CLI) installed and initialized on your local machine (`gcloud init`).
- Python 3.8 or later installed locally.

Step-by-Step Guide

1. Create Your Flask Application Files

Create a project directory and add the following three files in the root folder:

- **`main.py`**: The Python application file.
- **`requirements.txt`**: A list of Python packages the application needs.
- **`app.yaml`**: The configuration file for Google App Engine.

`main.py`

This file contains your Flask application code. Google App Engine's standard environment expects an `app` variable in `main.py` by default:

python

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

@app.route("/")
def hello() -> str:
    """Return a friendly HTTP greeting."""
    return "Hello World! This is a Flask app on Google App Engine."

if __name__ == "__main__":
    # This is used when running locally only. When deploying to Google App
    # Engine, a webserver process such as
```

python

```
Gunicorn
```

python

```
 will serve the app.
    app.run(host="127.0.0.1", port=8080, debug=True)
```

`requirements.txt`

This file lists the Python packages required for your app to run:

```
Flask
```

You can create this file automatically by running `pip freeze > requirements.txt` in your activated virtual environment.

`app.yaml`

This file tells App Engine how to run your application. For a basic Python 3 app in the standard environment, you only need to specify the runtime:

yaml

```yaml
runtime: python3
```

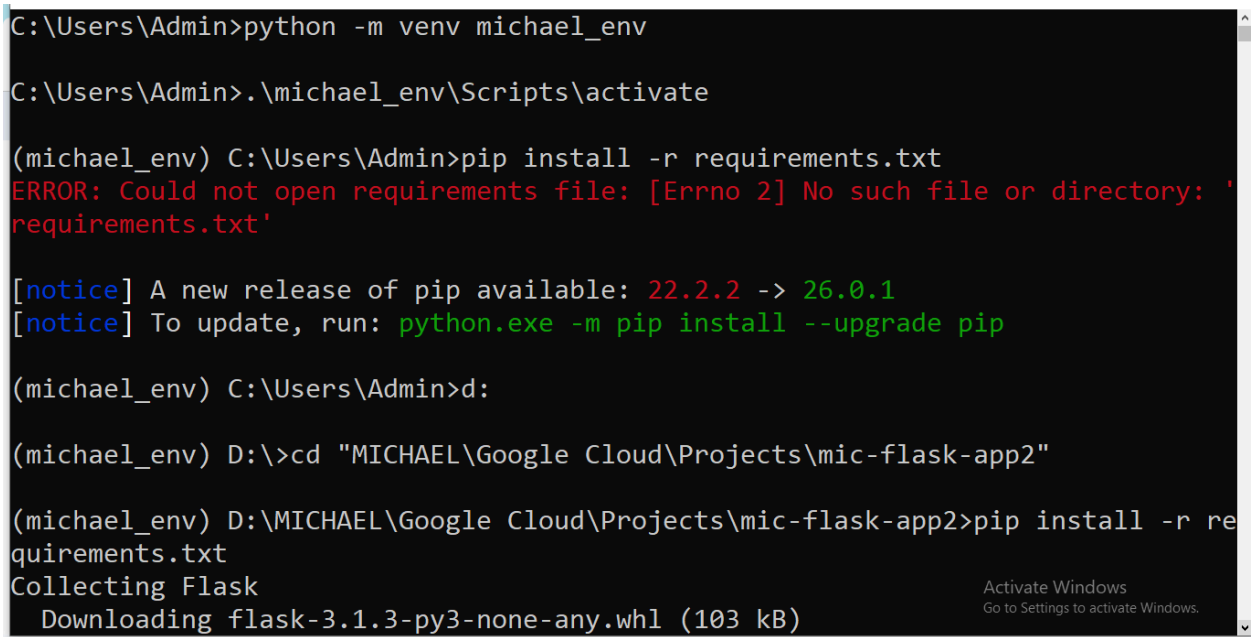
2. Test the Application Locally

Run your application locally to ensure it works before deploying:

1. **Install dependencies** in a virtual environment:

bash

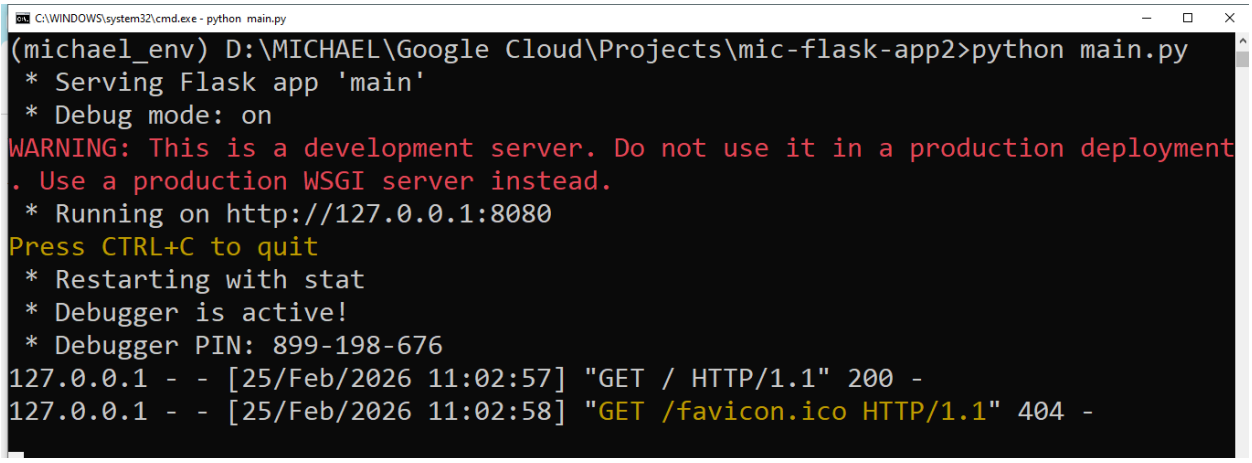
```bash
python3 -m venv env
source env/bin/activate  # On Windows use `.\env\Scripts\activate`
pip install -r requirements.txt
```



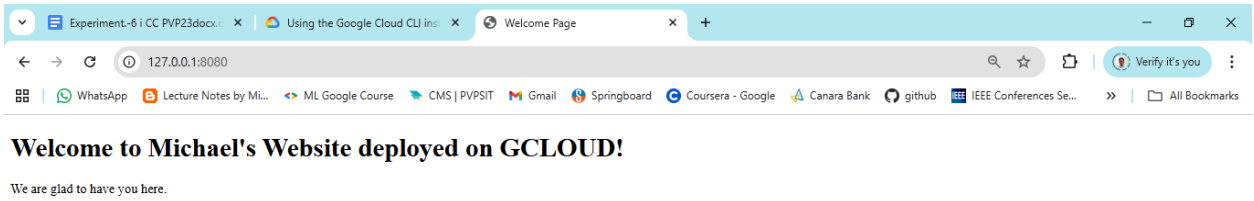
2. **Run the Flask app**:

bash

```bash
python main.py
```



3. **View the app** by opening `http://127.0.0.1:8080` in your web browser.



3. Deploy to Google App Engine

Once the app works locally, you can deploy it to GAE:

1. **Ensure you are authenticated** with the Google Cloud CLI:

bash

```
gcloud auth login
```

2. **Set the project ID** if not already set:

bash

```
gcloud config set project YOUR_PROJECT_ID
```

3. **Deploy your application** from your project's root directory:

bash

```
gcloud app deploy
```

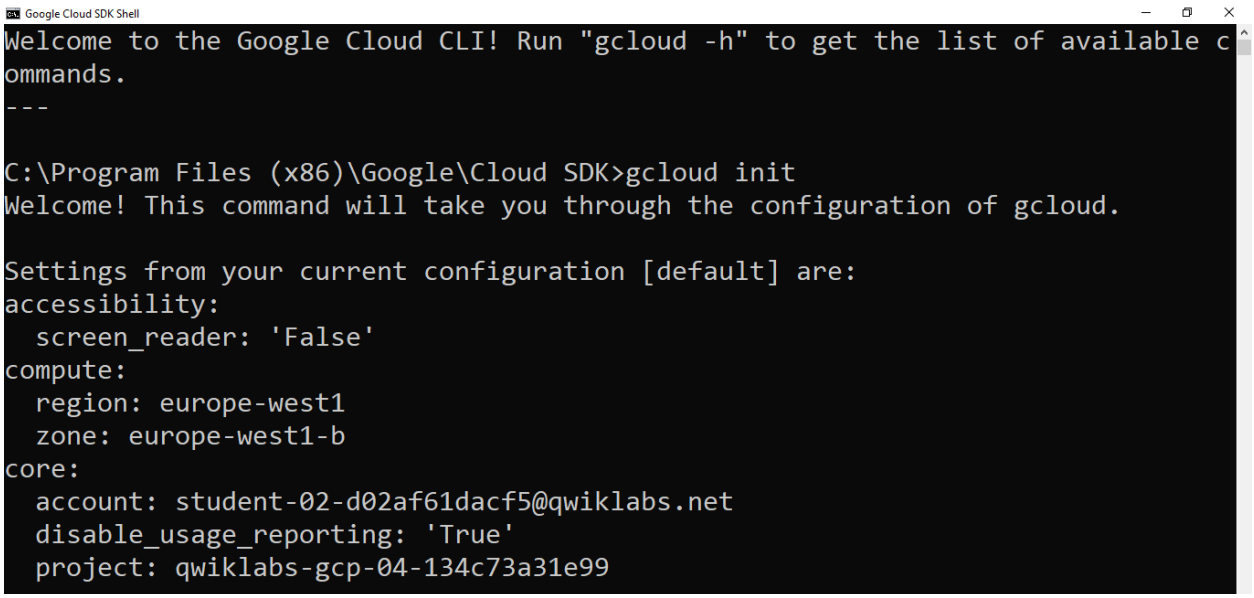
4. **Confirm the deployment** when prompted. The process will upload your files, build a container, and provision the necessary infrastructure.

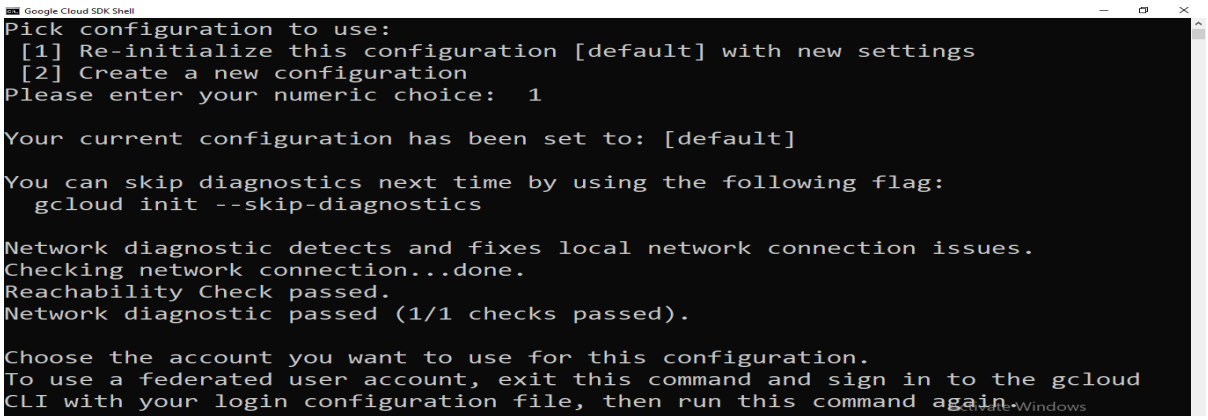
5. **View your live application** in a web browser using the command:

bash

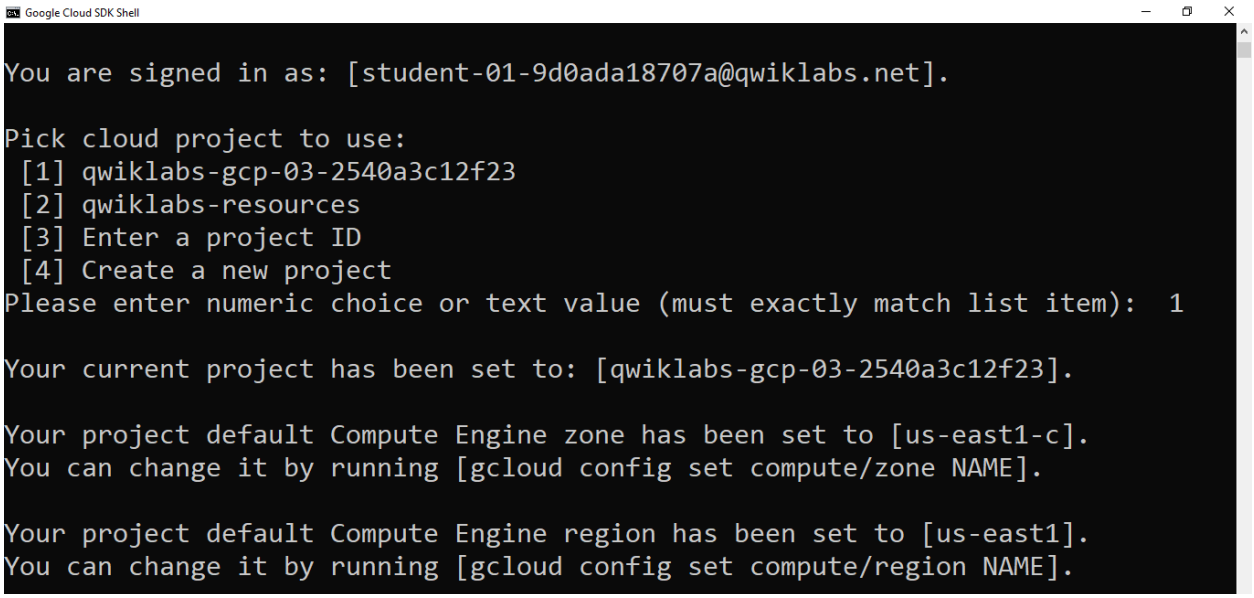
`gcloud app browse`

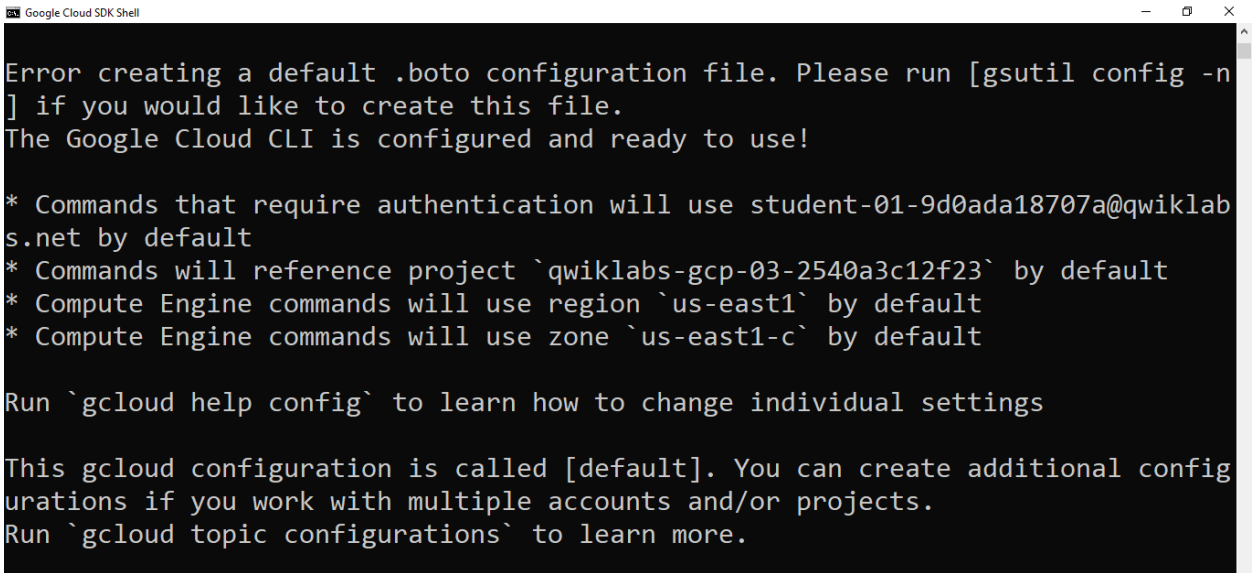
This command will output the URL of your deployed service (e.g., https://PROJECT_ID.REGION_ID.r.appspot.com).

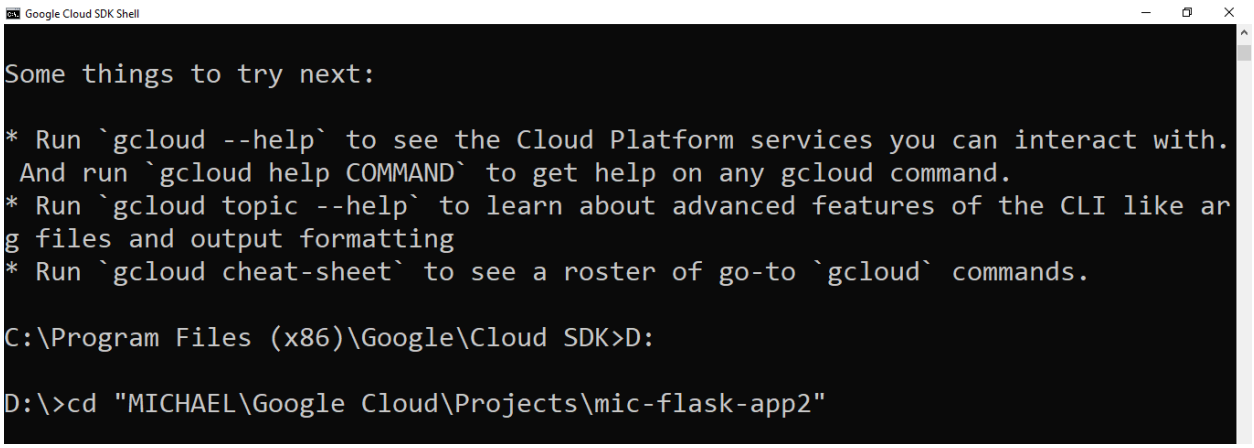



Google Cloud SDK Shell
Pick configuration to use:
[1] Re-initialize this configuration [default] with new settings
[2] Create a new configuration
Please enter your numeric choice: 1
Your current configuration has been set to: [default]
You can skip diagnostics next time by using the following flag:
gcloud init --skip-diagnostics
Network diagnostic detects and fixes local network connection issues.
Checking network connection...done.
Reachability Check passed.
Network diagnostic passed (1/1 checks passed).
Choose the account you want to use for this configuration.
To use a federated user account, exit this command and sign in to the gcloud
CLI with your login configuration file, then run this command again.


Google Cloud SDK Shell
Select an account:
[1] junction.fellowship@gmail.com
[2] michael.kona@gmail.com
[3] student-02-d02af61dacf5@qwiklabs.net
[4] Sign in with a new Google Account
[5] Skip this step
Please enter your numeric choice: 4
Your browser has been opened to visit:
https://accounts.google.com/o/oauth2/auth?response_type=code&client_id=3255
5940559.apps.googleusercontent.com&redirect_uri=http%3A%2F%2Flocalhost%3A8085%2
F&scope=openid+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+https%3
A%2F%2Fwww.googleapis.com%2Fauth%2Fcloud-platform+https%3A%2F%2Fwww.googleapis.
com%2Fauth%2Fappengine.admin+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fsqlservi
ce.login+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcompute+https%3A%2F%2Fwww.go
ogleapis.com%2Fauth%2Faccounts.reauth&state=Lzg0zUha06BRBVV5IqzBqfVZl4MiyD&acces
s_type=offline&code_challenge=vlc78JOtHPYSwRu70In-Be9opEWJtSzY4mI1EtgWVNg&code
_challenge_method=S256


Google Cloud SDK Shell
You are signed in as: [student-01-9d0ada18707a@qwiklabs.net].
Pick cloud project to use:
[1] qwiklabs-gcp-03-2540a3c12f23
[2] qwiklabs-resources
[3] Enter a project ID
[4] Create a new project
Please enter numeric choice or text value (must exactly match list item): 1
Your current project has been set to: [qwiklabs-gcp-03-2540a3c12f23].
Your project default Compute Engine zone has been set to [us-east1-c].
You can change it by running [gcloud config set compute/zone NAME].
Your project default Compute Engine region has been set to [us-east1].
You can change it by running [gcloud config set compute/region NAME].


Google Cloud SDK Shell
Error creating a default .boto configuration file. Please run [gsutil config -n
] if you would like to create this file.
The Google Cloud CLI is configured and ready to use!
* Commands that require authentication will use student-01-9d0ada18707a@qwiklab
s.net by default
* Commands will reference project `qwiklabs-gcp-03-2540a3c12f23` by default
* Compute Engine commands will use region `us-east1` by default
* Compute Engine commands will use zone `us-east1-c` by default
Run `gcloud help config` to learn how to change individual settings
This gcloud configuration is called [default]. You can create additional config
urations if you work with multiple accounts and/or projects.
Run `gcloud topic configurations` to learn more.


Google Cloud SDK Shell
Some things to try next:
* Run `gcloud --help` to see the Cloud Platform services you can interact with.
And run `gcloud help COMMAND` to get help on any gcloud command.
* Run `gcloud topic --help` to learn about advanced features of the CLI like ar
g files and output formatting
* Run `gcloud cheat-sheet` to see a roster of go-to `gcloud` commands.
C:\Program Files (x86)\Google\Cloud SDK>D:
D:\>cd "MICHAEL\Google Cloud\Projects\mic-flask-app2"

Google Cloud SDK Shell

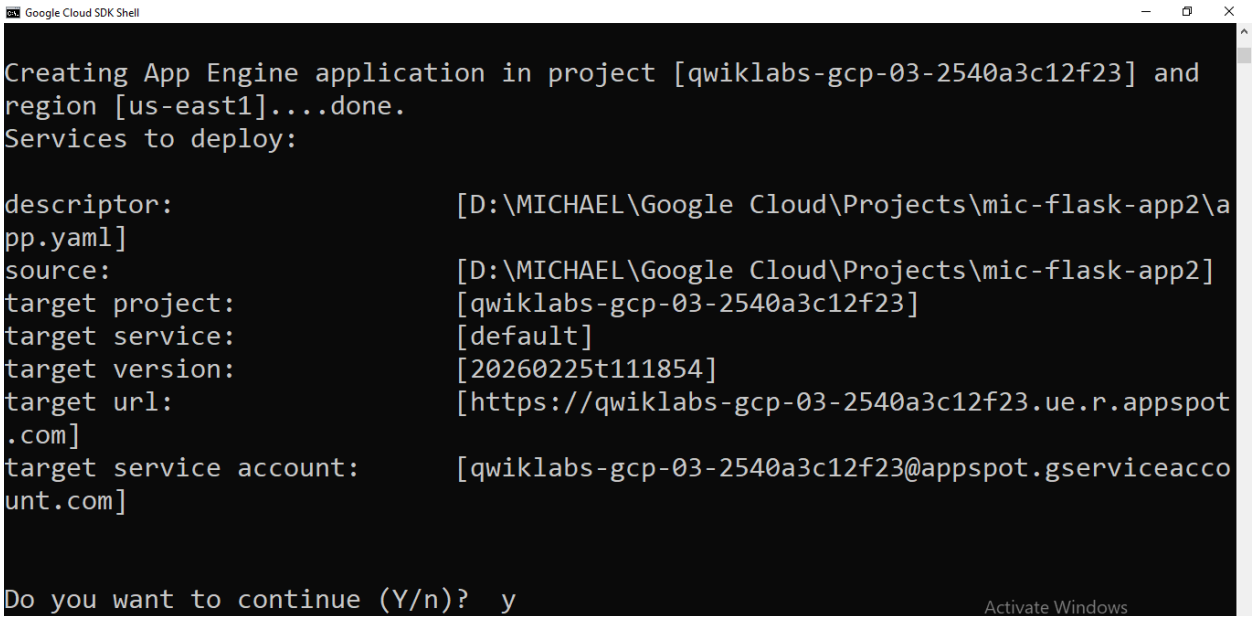
```
D:\>cd "MICHAEL\Google Cloud\Projects\mic-flask-app2"

D:\MICHAEL\Google Cloud\Projects\mic-flask-app2>gcloud app deploy
WARNING: You might be using automatic scaling for a standard environment deploy
ment, without providing a value for automatic_scaling.max_instances. Starting f
rom March, 2025, App Engine sets the automatic scaling maximum instances defaul
t for standard environment deployments to 20. This change doesn't impact existi
ng apps. To override the default, specify the new max_instances value in your a
pp.yaml file, and deploy a new version or redeploy over an existing version. Fo
r details on max_instances, see https://cloud.google.com/appengine/docs/standar
d/reference/app-yaml.md#scaling_elements.
```

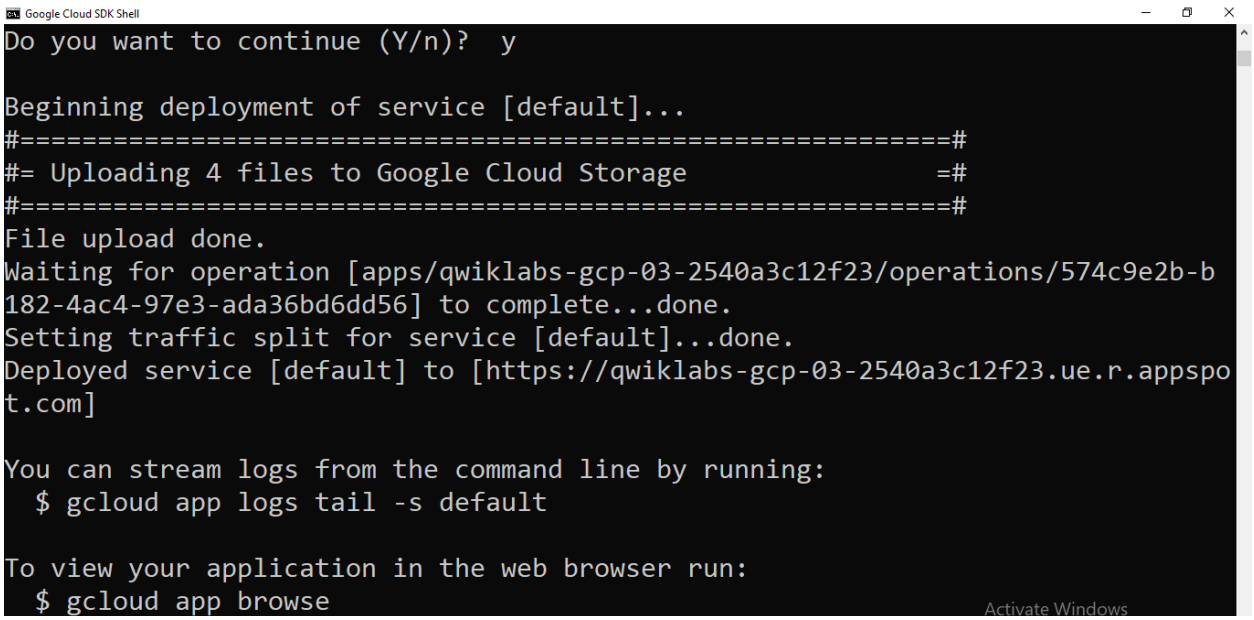
Google Cloud SDK Shell

```
You are creating an app for project [qwiklabs-gcp-03-2540a3c12f23].
WARNING: Creating an App Engine application for a project is irreversible and t
he region
cannot be changed. More information about regions is at
<https://cloud.google.com/appengine/docs/locations>.

WARNING: Starting from March, 2025, App Engine sets the automatic scaling maxim
um instances
default for standard environment deployments to 20. This change doesn't impact
existing apps. To override the default, specify the new max_instances value in
your
app.yaml file, and deploy a new version or redeploy over an existing version.
For more details on max_instances, see
<https://cloud.google.com/appengine/docs/standard/reference/app-yaml.md#scaling
_elements>.
```

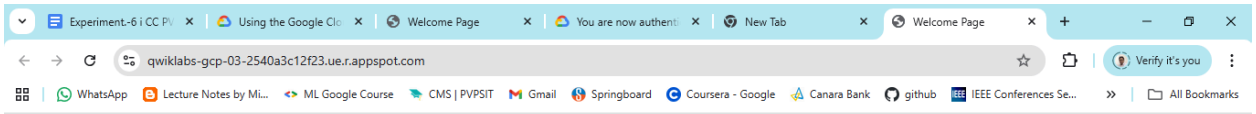
Google Cloud SDK Shell

```
NOTE: Cloud Run offers the most modern fully managed application hosting experi
ence
with lower minimum billable times and support for GPUs on demand for your AI/ML
 workloads.
Deploy code written in any programming language supported by App Engine on Clou
d Run.
Learn more at https://cloud.google.com/run/docs/quickstarts#build-and-deploy-a-
web-service

Please choose the region where you want your App Engine application located:

 [1] europe-west   (supports standard and flexible and search_api)
 [2] us-central    (supports standard and flexible and search_api)
 [3] us-east1      (supports standard and flexible and search_api)
 [4] cancel
Please enter your numeric choice:  3

Creating App Engine application in project [qwiklabs-gcp-03-2540a3c12f23] and
region [us-east1]....done.
```


Google Cloud SDK Shell
Creating App Engine application in project [qwiklabs-gcp-03-2540a3c12f23] and region [us-east1]....done.
Services to deploy:
descriptor: [D:\MICHAEL\Google Cloud\Projects\mic-flask-app2\app.yaml]
source: [D:\MICHAEL\Google Cloud\Projects\mic-flask-app2]
target project: [qwiklabs-gcp-03-2540a3c12f23]
target service: [default]
target version: [20260225t111854]
target url: [https://qwiklabs-gcp-03-2540a3c12f23.ue.r.appspot.com]
target service account: [qwiklabs-gcp-03-2540a3c12f23@appspot.gserviceaccount.com]
Do you want to continue (Y/n)? y
Activate Windows


Google Cloud SDK Shell
Do you want to continue (Y/n)? y
Beginning deployment of service [default]...
#===#
#= Uploading 4 files to Google Cloud Storage =#
#===#
File upload done.
Waiting for operation [apps/qwiklabs-gcp-03-2540a3c12f23/operations/574c9e2b-b182-4ac4-97e3-ada36bd6dd56] to complete...done.
Setting traffic split for service [default]...done.
Deployed service [default] to [https://qwiklabs-gcp-03-2540a3c12f23.ue.r.appspot.com]
You can stream logs from the command line by running:
$ gcloud app logs tail -s default
To view your application in the web browser run:
$ gcloud app browse
Activate Windows


Google Cloud SDK Shell
D:\MICHAEL\Google Cloud\Projects\mic-flask-app2>gcloud app browse
Opening [https://qwiklabs-gcp-03-2540a3c12f23.ue.r.appspot.com] in a new tab in your default browser.
D:\MICHAEL\Google Cloud\Projects\mic-flask-app2>



Welcome to Michael's Website deployed on GCLOUD!

We are glad to have you here.