

**Fedoseenko A.**

## **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.
  - Install the Google Cloud SDK on your local machine.
2. **Create a Project:**
  - Create a new project in the Google Cloud Console.

```
imac@MacBook-Pro-iMac google-cloud-sdk % gcloud projects create my-web-app-project-cloud --name="My Web App Project"

Create in progress for [https://cloudresourcemanager.googleapis.com/v1/projects/my-web-app-project-cloud].
Waiting for [operations/cp.7243476141897273552] to finish...done.
Enabling service [cloudapis.googleapis.com] on project [my-web-app-project-cloud]...
Operation "operations/acat.p2-884941616033-30d6cb41-7e73-45f5-a25b-2f68202ed7c6" finished successfully.

Updates are available for some Google Cloud CLI components. To install them,
please run:
  $ gcloud components update

To take a quick anonymous survey, run:
  $ gcloud survey

imac@MacBook-Pro-iMac google-cloud-sdk % gcloud config set project my-web-app-project-cloud

Updated property [core/project].
imac@MacBook-Pro-iMac google-cloud-sdk %
```

3. **Prepare the Application:**
  - Write a simple "Hello, World!" web application using Python (Flask).

Example `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)
```

```
UW PICO 5.09 File: app.py Modified
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:

Create a `app.yaml` file with the following content:

```
runtime: python39
handlers:
- url: /*
  script: auto
```

```
UW PICO 5.09 File: app.yaml Modified
runtime: python39
handlers:
- url: /*
  script: auto
```

```
imac@MacBook-Pro-iMac google-cloud-sdk % nano app.py
imac@MacBook-Pro-iMac google-cloud-sdk % nano app.yaml
```

○

#### 5. Deploy the Application:

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

```
gcloud app deploy
```

```

imac@MacBook-Pro-iMac google-cloud-sdk % nano app.yaml
imac@MacBook-Pro-iMac google-cloud-sdk % gcloud app deploy
You are creating an app for project [my-web-app-project-cloud].
WARNING: Creating an App Engine application for a project is irreversible and the region
cannot be changed. More information about regions is at
<https://cloud.google.com/appengine/docs/locations>.

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

[1] asia-east1      (supports standard and flexible)
[2] asia-east2      (supports standard and flexible and search_api)
[3] asia-northeast1 (supports standard and flexible and search_api)
[4] asia-northeast2 (supports standard and flexible and search_api)
[5] asia-northeast3 (supports standard and flexible and search_api)
[6] asia-south1     (supports standard and flexible and search_api)
[7] asia-southeast1 (supports standard and flexible)
[8] asia-southeast2 (supports standard and flexible and search_api)
[9] australia-southeast1 (supports standard and flexible and search_api)
[10] europe-central2 (supports standard and flexible)
[11] europe-west    (supports standard and flexible and search_api)
[12] europe-west2   (supports standard and flexible and search_api)
[13] europe-west3   (supports standard and flexible and search_api)
[14] europe-west6   (supports standard and flexible and search_api)
[15] northamerica-northeast1 (supports standard and flexible and search_api)
[16] southamerica-east1 (supports standard and flexible and search_api)
[17] us-central     (supports standard and flexible and search_api)
[18] us-east1       (supports standard and flexible and search_api)
[19] us-east4       (supports standard and flexible and search_api)
[20] us-west1       (supports standard and flexible)
[21] us-west2       (supports standard and flexible and search_api)
[22] us-west3       (supports standard and flexible and search_api)
[23] us-west4       (supports standard and flexible and search_api)
[24] cancel

Please enter your numeric choice: 10

Creating App Engine application in project [my-web-app-project-cloud] and region [europe-central2]...failed.
ERROR: (gcloud.app.deploy) PERMISSION_DENIED: Read access to project 'my-web-app-project-cloud' was denied: please che
ck billing account associated and retry. This command is authenticated as anna.knst23@gmail.com which is the active ac
count specified by the [core/account] property
imac@MacBook-Pro-iMac google-cloud-sdk %

```

## 6. Access the Application:

- Once deployed, access your application using the URL provided by Google App Engine.
- Expected output:

Deployed service to [https://my-web-app-project-cloud.lm.r.appspot.com]

```

imac@MacBook-Pro-iMac google-cloud-sdk % gcloud app deploy
Services to deploy:

descriptor:      [/Users/imac/Documents/google-cloud-sdk/app.yaml]
source:          [/Users/imac/Documents/google-cloud-sdk]
target project:  [my-web-app-project-cloud]
target service:  [default]
target version:  [20240927t092938]
target url:      [https://my-web-app-project-cloud.lm.r.appspot.com]
target service account: [my-web-app-project-cloud@appspot.gserviceaccount.com]

Do you want to continue (Y/n)? y

Beginning deployment of service [default]...

- Uploading 0 files to Google Cloud Storage

File upload done.
ERROR: (gcloud.app.deploy) INVALID_ARGUMENT: This deployment has too many files. New versions are limited to 10000 fil
es for this app.
- '@type': type.googleapis.com/google.rpc.BadRequest
  fieldViolations:
    - description: This deployment has too many files. New versions are limited to 10000
      files for this app.
      field: version.deployment.files[...]
imac@MacBook-Pro-iMac google-cloud-sdk %

```



Approximate output:

---

## 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!');  
};
```

○

```
imac@MacBook-Pro-iMac google-cloud-sdk % mkdir my-cloud-function  
[cd my-cloud-function
```

```
[imac@MacBook-Pro-iMac my-cloud-function % nano index.js
```

```
UW PICO 5.09 File: 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.

```

imac@MacBook-Pro-iMac my-cloud-function % gcloud functions deploy helloWorldFunction \
--runtime nodejs18 \
--trigger-http \
--entry-point helloWorld \
--allow-unauthenticated
In a future Cloud SDK release, new functions will be deployed as 2nd gen functions by default. This is equivalent to
currently deploying new with the --gen2 flag. Existing 1st gen functions will not be impacted and will continue to de
ploy as 1st gen functions.
You can preview this behavior in beta. Alternatively, you can disable this behavior by explicitly specifying the --no-
gen2 flag or by setting the functions/gen2 config property to 'off'.
To learn more about the differences between 1st gen and 2nd gen functions, visit:
https://cloud.google.com/functions/docs/concepts/version-comparison
Deploying function (may take a while - up to 2 minutes)...
For Cloud Build Logs, visit: https://console.cloud.google.com/cloud-build/builds;region=us-central1/62f1e9c1-1c00-47c4
-a9f4-6f2f5517e0be?project=884941616033
Deploying function (may take a while - up to 2 minutes)...done.
automaticUpdatePolicy: {}
availableMemoryMb: 256
buildId: 62f1e9c1-1c00-47c4-a9f4-6f2f5517e0be
buildName: projects/884941616033/locations/us-central1/builds/62f1e9c1-1c00-47c4-a9f4-6f2f5517e0be
buildServiceAccount: projects/my-web-app-project-cloud/serviceAccounts/884941616033-compute@developer.gserviceaccount.
com
dockerRegistry: ARTIFACT_REGISTRY
entryPoint: helloWorld
httpsTrigger:
  securityLevel: SECURE_OPTIONAL
  url: https://us-central1-my-web-app-project-cloud.cloudfunctions.net/helloWorldFunction
ingressSettings: ALLOW_ALL
labels:
  deployment-tool: cli-gcloud
name: projects/my-web-app-project-cloud/locations/us-central1/functions/helloWorldFunction
runtime: nodejs18
serviceAccountEmail: my-web-app-project-cloud@appspot.gserviceaccount.com
sourceUploadUrl: https://storage.googleapis.com/uploads-789528987352.us-central1.cloudfunctions.appspot.com/c1497b35-8
627-4db4-80b3-5dc9b9fa7070.zip
status: ACTIVE
timeout: 60s
updateTime: '2024-09-26T19:51:46.229Z'
versionId: '3'
imac@MacBook-Pro-iMac my-cloud-function %

```

- A screenshot showing the response from the function.

```

imac@MacBook-Pro-iMac my-cloud-function % curl https://us-central1-my-web-app-project-cloud.cloudfunctions.net/helloWo
rldFunction
Hello, World!%
imac@MacBook-Pro-iMac 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.

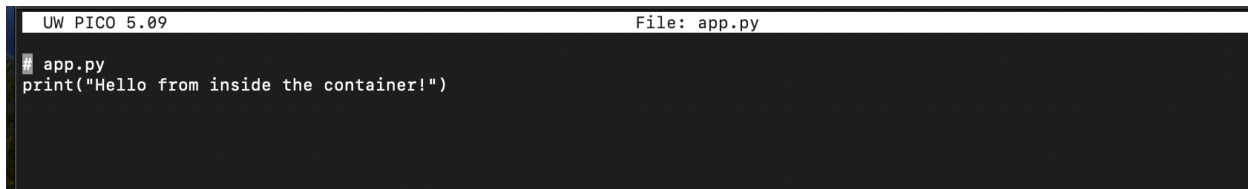


Example `app.py`:

```
print("Hello from inside the container!")
```

```
imac@MacBook-Pro-iMac bin % mkdir docker-python-app
cd docker-python-app

[imac@MacBook-Pro-iMac docker-python-app % nano app.py
[imac@MacBook-Pro-iMac docker-python-app % nano Dockerfile
```

A screenshot of a nano editor window. The title bar at the top shows "UW PICO 5.09" on the left and "File: app.py" on the right. The main editing area contains the following text:

```
# 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"]
```

```
# 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"]
```

**FROM python:3.9-slim:** This pulls the official lightweight Python 3.9 image as the base for your container.

**WORKDIR /app:** This sets the working directory in the container to **/app**.

**COPY . /app:** This copies the current directory's contents into the container's **/app** directory.

**CMD ["python", "app.py"]:** This specifies the command to run the **app.py** file when the container starts.

#### 4. Build the Docker Image:

Build the Docker image using the following command:

**docker build -t hello-world-app .**

```
[imac@MacBook-Pro-iMac docker-python-app % docker build -t hello-world-app .
[+] Building 293.9s (8/8) FINISHED                                docker:desktop-linux
=> [internal] load build definition from Dockerfile                0.0s
=> => transferring dockerfile: 297B                                0.0s
=> [internal] load metadata for docker.io/library/python:3.9-slim 7.4s
=> [internal] load .dockerignore                                  0.0s
=> => transferring context: 2B                                       0.0s
=> [1/3] FROM docker.io/library/python:3.9-slim@sha256:2851c06da1fdc3c451784beef8aa31d1a313d8e3fc122e4a18910 286.3s
=> => resolve docker.io/library/python:3.9-slim@sha256:2851c06da1fdc3c451784beef8aa31d1a313d8e3fc122e4a1891085 0.0s
=> => sha256:2851c06da1fdc3c451784beef8aa31d1a313d8e3fc122e4a1891085a104b7c7b 10.41kB / 10.41kB 0.0s
=> => sha256:17934128833e308455054ef8eb75d87e1760f4470a2ae77a27d488e0e5411d92 1.75kB / 1.75kB 0.0s
=> => sha256:c0e4ee20d94c77d442f8c8b0153a5e15052211ff874dc08a23e3b955d0689765 5.22kB / 5.22kB 0.0s
=> => sha256:92c3b3500be621c72c7ac6432a9d8f731f145f4a1535361ffd3a304e55f7ccda 29.16MB / 29.16MB 279.1s
=> => sha256:89790d4ca55c29720fc29c489ba4403f3bb6baa1a6d8b5d0e96c3d40521408c0 3.33MB / 3.33MB 35.4s
=> => sha256:04acf592cf1a560c87343c39e76e3372e54bed9bcd59bbc2c5289ff4eeb9e08 14.71MB / 14.71MB 285.7s
=> => sha256:c21204f5797c735b45941b864c2f521033d92bb12a150b4257afde4f5570a250 250B / 250B 56.2s
=> => extracting sha256:92c3b3500be621c72c7ac6432a9d8f731f145f4a1535361ffd3a304e55f7ccda 1.0s
=> => extracting sha256:89790d4ca55c29720fc29c489ba4403f3bb6baa1a6d8b5d0e96c3d40521408c0 0.1s
=> => extracting sha256:04acf592cf1a560c87343c39e76e3372e54bed9bcd59bbc2c5289ff4eeb9e08 0.5s
=> => extracting sha256:c21204f5797c735b45941b864c2f521033d92bb12a150b4257afde4f5570a250 0.0s
=> [internal] load build context                                  0.0s
=> => transferring context: 384B                                     0.0s
=> [2/3] WORKDIR /app                                           0.1s
=> [3/3] COPY . /app                                           0.0s
=> exporting to image                                           0.0s
=> => exporting layers                                           0.0s
=> => writing image sha256:70fc8b0068588784132e7672a794ea170699b711c324900a619f7084339fb48a 0.0s
=> => naming to docker.io/library/hello-world-app               0.0s
[imac@MacBook-Pro-iMac docker-python-app % ]
```



## 5. Run the Docker Container:

Run the container using the following command:

```
docker run --rm hello-world-app
```

```
imac@MacBook-Pro-iMac docker-python-app % docker run --rm hello-world-app
Hello from inside the container!
imac@MacBook-Pro-iMac docker-python-app %
```

### Deliverables:

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

```
imac@MacBook-Pro-iMac docker-python-app % docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
hello-world-app	latest	70fc8b006858	32 minutes ago	151MB

```
imac@MacBook-Pro-iMac docker-python-app %
```