

Assignment 3

Student: Islam Aip

Course: Cloud application development

Date:

Almaty, 2024

**Exercise 1: Managing APIs with Google Cloud Endpoints**

**Objective**: Deploy and manage an API using Google Cloud Endpoints.

**Instructions**:

1. **Setup**:
   * Ensure you have a Google Cloud account.
   * Install the Google Cloud SDK and gcloud command-line tool.
2. **Create a Project**:
   * Create a new project in the Google Cloud Console.
3. **Prepare the API**:
   * Create a simple REST API using Python Flask.

Example app.py:  
  
from flask import Flask, jsonify

app = Flask(\_\_name\_\_)

@app.route('/api/hello', methods=['GET'])

def hello():

    return jsonify({'message': 'Hello, World!'})

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(host='0.0.0.0', port=8080, debug=True)

1. **Create an OpenAPI Specification**:
   * Create an openapi.yaml file to define your API.

Example openapi.yaml:  
openapi: 3.0.0

info:

  title: Hello World API

  description: A simple API to say hello

  version: 1.0.0

paths:

  /api/hello:

    get:

      summary: Returns a hello message

      responses:

        '200':

          description: A hello message

          content:

            application/json:

              schema:

                type: object

                properties:

                  message:

                    type: string

                    example: Hello, World!

1. **Deploy the API to Google Cloud Endpoints**:

Create a new service and deploy your API.

Use the following commands to deploy the API configuration and service:  
  
gcloud endpoints services deploy openapi.yaml

gcloud app deploy

1. **Test the API**:
   * Once deployed, use the provided URL to test the API endpoint via a web browser or curl.

**Deliverables**:

* A deployed API on Google Cloud Endpoints.
* A screenshot of a successful API call response.

**Exercise 2: Google Cloud Databases**

**Objective**: Set up and interact with a Google Cloud SQL database.

**Instructions**:

1. **Setup**:
   * Ensure you have a Google Cloud account.
   * Install the Google Cloud SDK.
2. **Create a Cloud SQL Instance**:
   * Navigate to the Google Cloud Console and create a new Cloud SQL instance.
   * Choose MySQL, PostgreSQL, or SQL Server as the database type.
   * Configure the instance settings (region, machine type, etc.).
3. **Create a Database and Table**:
   * Connect to your Cloud SQL instance using the Cloud SQL client or mysql command-line tool.
   * Create a new database and a table with sample data.

CREATE DATABASE sample\_db;

USE sample\_db;

CREATE TABLE users (

  id INT AUTO\_INCREMENT PRIMARY KEY,

  name VARCHAR(100) NOT NULL,

  email VARCHAR(100) NOT NULL

);

INSERT INTO users (name, email) VALUES ('Alice', 'alice@example.com');

INSERT INTO users (name, email) VALUES ('Bob', 'bob@example.com');

1. **Connect to the Database**:
   * Create a connection to the Cloud SQL instance from a Python application.

Example connect.py:  
  
import mysql.connector

cnx = mysql.connector.connect(

    user='your-username',

    password='your-password',

    host='your-cloud-sql-instance-ip',

    database='sample\_db'

)

cursor = cnx.cursor()

cursor.execute('SELECT \* FROM users')

for row in cursor:

    print(row)

cursor.close()

cnx.close()

1. **Run the Connection Code**:

Execute the Python script to verify that you can retrieve data from the Cloud SQL instance.

**Deliverables**:

* A working Cloud SQL database with sample data.
* A Python script that successfully connects to and queries the database.

**Exercise 3: Integrating Machine Learning with Google Cloud**

**Objective**: Train and deploy a machine learning model using Google Cloud AI Platform.

**Instructions**:

1. **Setup**:
   * Ensure you have a Google Cloud account.
   * Install the Google Cloud SDK and TensorFlow.
2. **Create a Cloud Storage Bucket**:
   * Create a new Cloud Storage bucket to store your training data and model.
3. **Prepare Training Data**:
   * Upload sample training data to your Cloud Storage bucket. For example, use a dataset for classification or regression.
4. **Create a Training Script**:
   * Write a simple TensorFlow training script.

Example train.py:  
python  
Копировать код  
import tensorflow as tf

def create\_model():

    model = tf.keras.Sequential([

        tf.keras.layers.Dense(10, activation='relu', input\_shape=(784,)),

        tf.keras.layers.Dense(10, activation='softmax')

    ])

    model.compile(optimizer='adam', loss='sparse\_categorical\_crossentropy', metrics=['accuracy'])

    return model

def main():

    model = create\_model()

    train\_data = tf.data.Dataset.from\_tensor\_slices((X\_train, y\_train)).batch(32)

    model.fit(train\_data, epochs=5)

    model.save('gs://your-bucket/model')

if \_\_name\_\_ == '\_\_main\_\_':

    main()

1. **Train the Model**:
   * Submit a training job to Google Cloud AI Platform.

Use the following command to start training:  
bash  
Копировать код  
gcloud ai custom-jobs create --region=your-region --display-name=ml-job --python-package-uris=gs://your-bucket/train.py --python-module=train --container-image-uri=gcr.io/cloud-aiplatform/training/tf-cpu.2-4:latest

1. **Deploy the Model**:
   * Deploy the trained model to an AI Platform endpoint.

Use the following command:  
bash  
Копировать код  
gcloud ai models create your-model --region=your-region

gcloud ai versions create v1 --model=your-model --origin=gs://your-bucket/model --runtime-version=2.7 --python-version=3.8

1. **Test the Model**:
   * Use the deployed model endpoint to make predictions.

Example predict.py:  
python  
Копировать код  
from google.cloud import aiplatform

def predict():

    client = aiplatform.gapic.PredictionServiceClient()

    endpoint = client.endpoint\_path(project='your-project', location='your-region', endpoint='your-endpoint-id')

    instance = {'input': [/\* your data \*/]}

    response = client.predict(endpoint=endpoint, instances=[instance])

    print(response.predictions)

if \_\_name\_\_ == '\_\_main\_\_':

    predict()

**Deliverables**:

* A trained machine learning model deployed on Google Cloud AI Platform.
* A script that makes predictions using the deployed model.
* Report