K. J. Somaiya School of Engineering, Mumbai-77
(Somaiya Vidyavihar University)
Department of Computer Engineering

Batch: C-1	**Roll No.: 16010122323**
Experiment / assignment / tutorial No 04	
Grade: AA / AB / BB / BC / CC / CD /DD	
Signature of the Staff In-charge with date	

Title: Hosting website using Google Cloud Platform

Objective:

Learn and implement the process of deploying a website on **Google App Engine (GAE)** using **Google Cloud Platform (GCP)**.

Expected Outcome of Experiment:

CO	Outcome
CO4	Build cloud services and applications

Books/ Journals/ Websites referred:

1. Google Cloud Documentation - https://cloud.google.com/docs
2. Mozilla Developer Network (MDN) - https://developer.mozilla.org/
3. Google App Engine Hosting Guide - https://developer.mozilla.org/en-US/docs/Learn/Common_questions/How_do_y ou_host_your_website_on_Google_App_Engine
4. Stack Overflow discussions on GCP deployment issues
5. Google Cloud Storage Documentation - https://cloud.google.com/storage

Abstract:- This experiment explores the deployment of a static website on **Google App Engine (GAE)**, an integral service of Google Cloud Platform. The process involves setting up a project, uploading website files, and configuring the `app.yaml` file to specify runtime and URL mappings. The deployment is performed using the Google Cloud Shell and `gcloud` command-line tool. Additionally, troubleshooting errors such as unsupported Python runtimes and permission issues related to Cloud Storage buckets is covered. The experiment demonstrates cloud-based hosting for scalable web applications, enabling practical knowledge of GCP services and cloud deployment strategies.

Related Theory: -

1. **Google App Engine (GAE):**
   - A **Platform-as-a-Service (PaaS)** solution for deploying web applications.
   - Supports multiple programming languages such as Python, Java, and Node.js.
   - Provides auto-scaling and built-in security features.
2. **Google Cloud Platform (GCP):**
   - A suite of cloud computing services including compute, storage, networking, and AI.
   - Offers **Google Cloud Shell**, a browser-based terminal to interact with cloud resources.
3. **App Engine Deployment Process:**
   - **Project Creation:** Setting up an application on GCP using the App Engine service.
   - **Website Preparation:** Organizing website files and ensuring the presence of `index.html`.
   - **Configuration (app.yaml):** Defining runtime, handlers, and application settings.
   - **Deployment (gcloud app deploy):** Uploading and publishing the website.
   - **Accessing the Website:** Hosted under a GCP-provided subdomain (`project-id.appspot.com`).
4. **Common Errors and Fixes:**
   - **Invalid Python Runtime:** Update `app.yaml` to use `python311`.
   - **Permission Issues:** Assign necessary IAM roles (Storage Admin, Cloud Build Editor) to the service account.

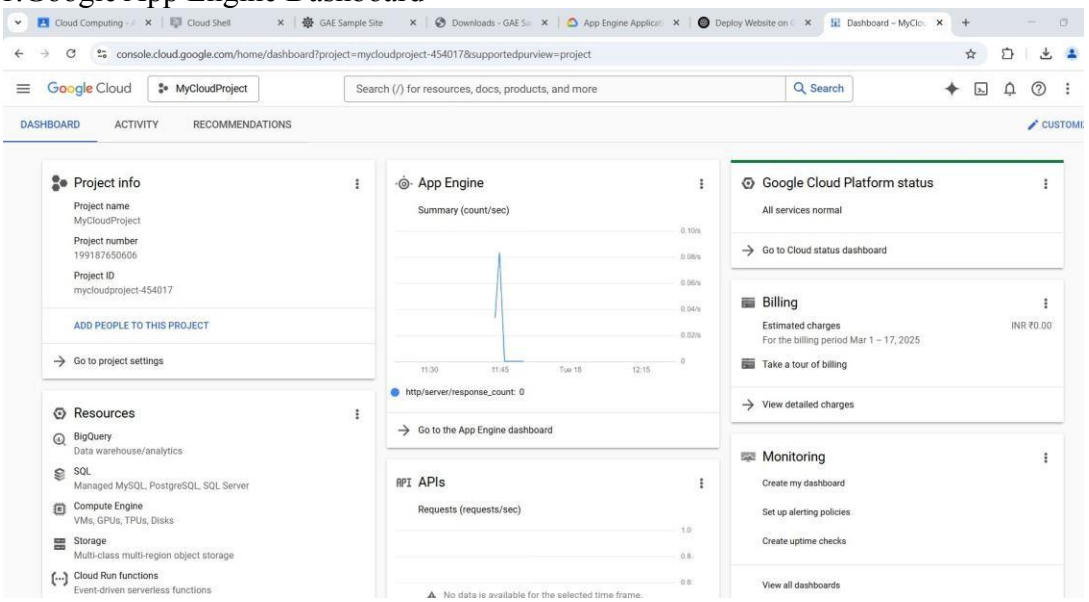
- **Missing Staging Bucket:** Manually create the storage bucket and provide access.

Implementation Details:

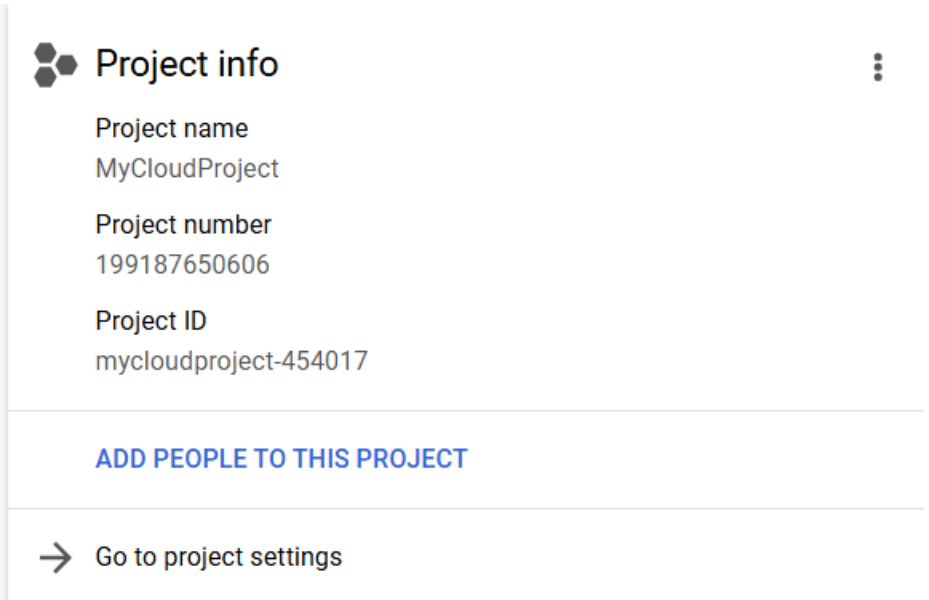
1. **Steps Followed and Options Explored:**
   - Created a **Google Cloud Project** and enabled App Engine.
   - Uploaded website files with an `index.html` as the main entry point.
   - Configured the `app.yaml` file with the correct runtime (`python311`).
   - Used **Google Cloud Shell** and `gcloud` CLI for deployment.
   - Explored different solutions for fixing **permission issues**, such as granting IAM roles to the service account.
   - Investigated and fixed **staging bucket errors** by manually creating the required storage bucket.
2. **Program Logic, Classes, and Methods Used:**
   - The deployment used **static file hosting** where `app.yaml` mapped URLs to static content.
   - No backend logic was required, but App Engine supports Flask or Django if needed.
   - Used `gcloud app deploy` to upload and manage the deployment.
3. **Importance of the Approach Followed:**
   - Utilizing **Google App Engine** simplifies web hosting without managing infrastructure.
   - Using **Google Cloud Shell** ensures a cloud-based environment for consistent deployment.
   - Applying **IAM roles correctly** prevents permission errors and ensures security.
   - Troubleshooting and manually configuring storage buckets provided a deeper understanding of GCP's resource dependencies.

Screenshots:

1.Google App Engine Dashboard



2.Created a new project with following project name and project ID

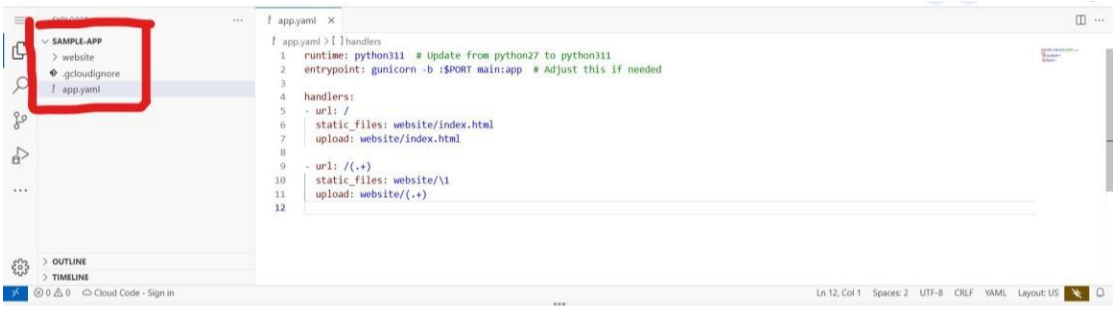


3.Open Google Cloud shell and set your project using ‘gcloud config set [PROJECT_ID]’

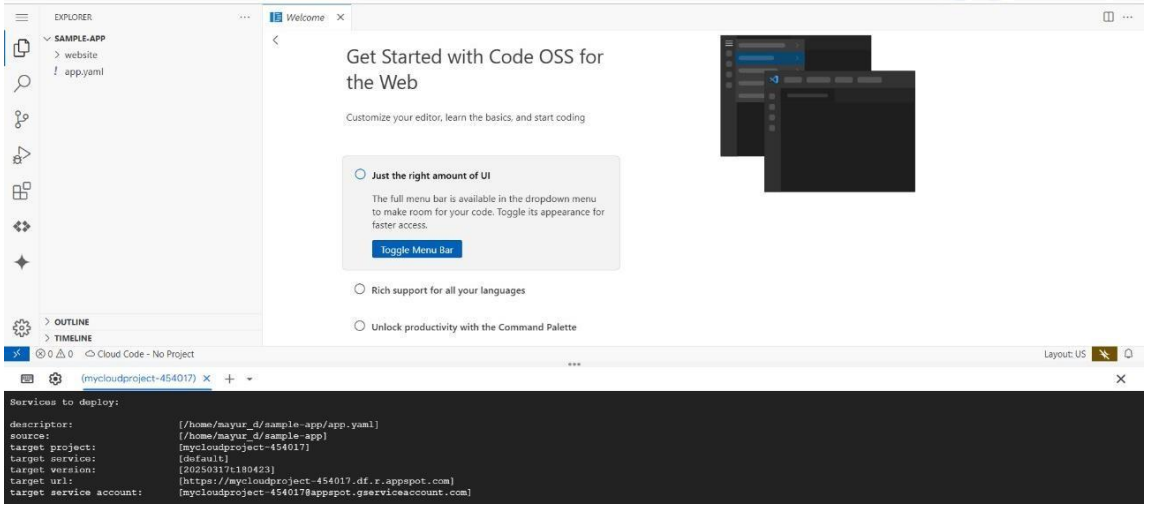
```
  gcloud config set --help
mayur_d@cloudshell:~$ ^C
mayur_d@cloudshell:~$ gcloud config list
[accessibility]
screen_reader = True
[component_manager]
disable_update_check = True
[compute]
gce_metadata_read_timeout_sec = 30
[core]
account =
disable_usage_reporting = False
[metrics]
environment = devshell

Your active configuration is: [cloudshell-15302]
mayur_d@cloudshell:~$ gcloud config set  mycloudproject-454017
```

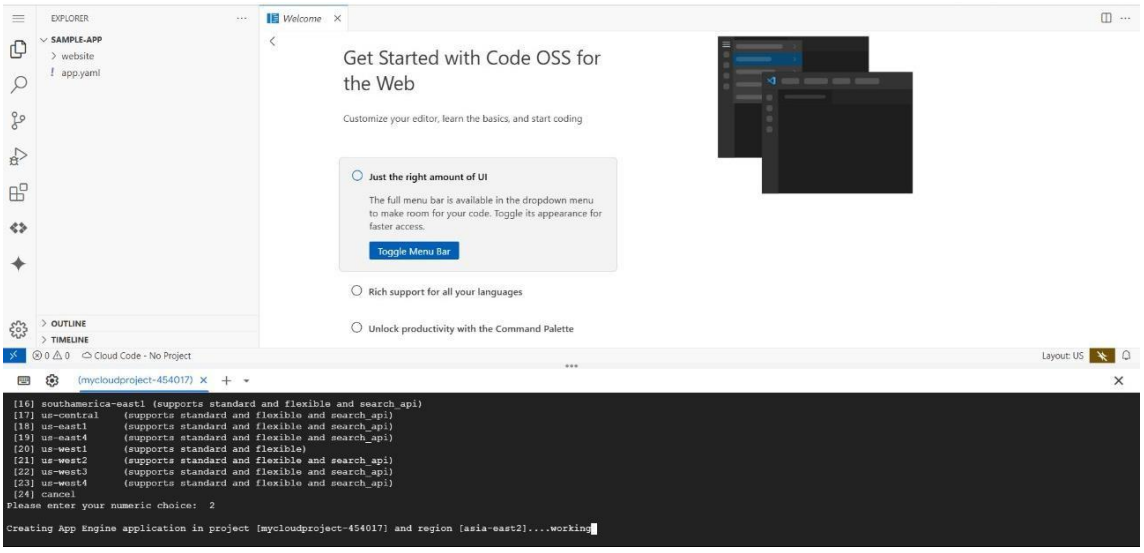
4.Nextly move the sample application folder in the google cloud shell



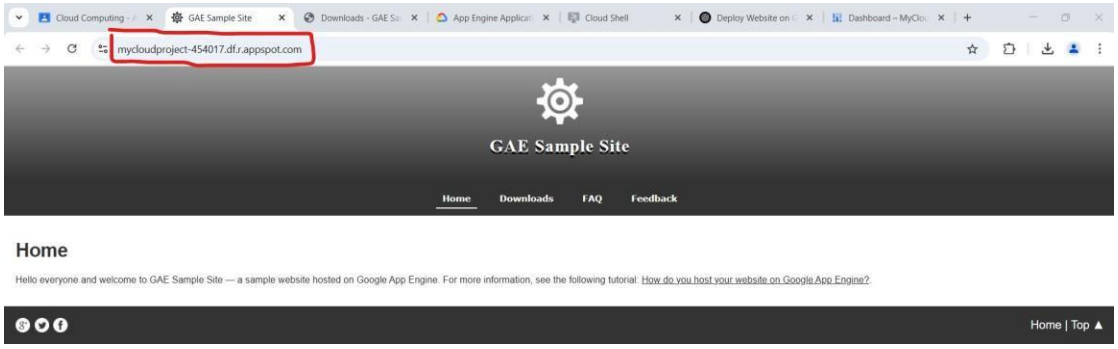
5.Then Deploy the application using ‘gcloud app deploy’ command



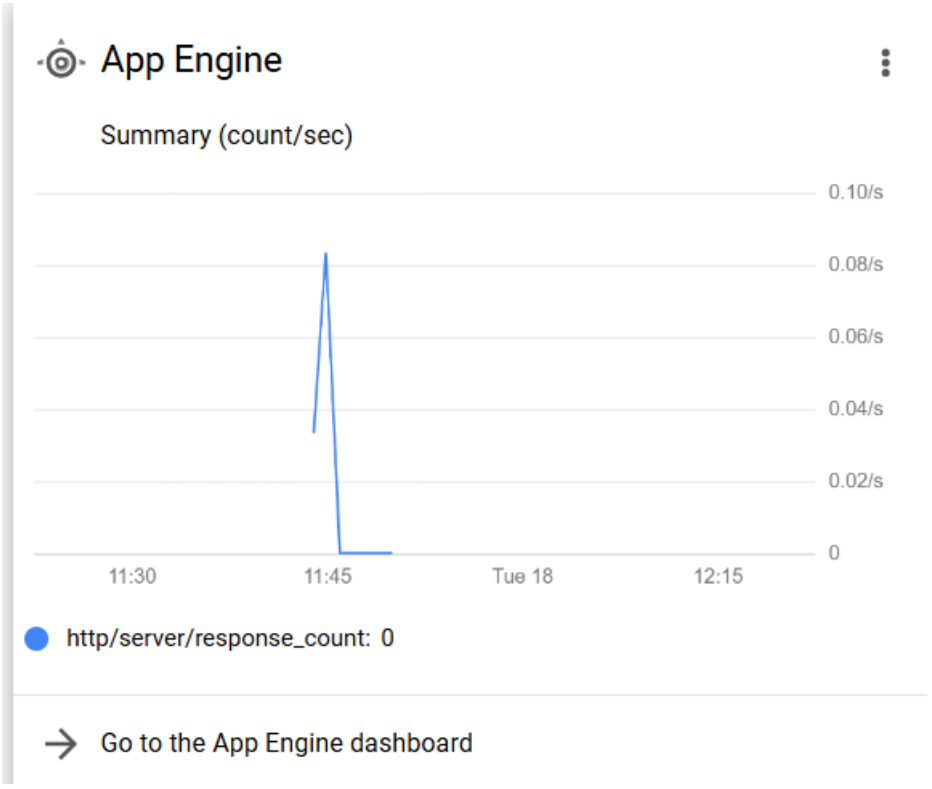
6. Choose the region of deployment using appropriate numeric value and type Y to confirm the deployment process.



7.The application is finally deployed and can be visited using the highlighted URL. The Highlighted URL consists of the project ID of our created project. In the cloud shell one can obtain the URL of deployed application using 'gcloud app browse' command.



8. Finally this is the summary for the deployed application available on google app engine dashboard.



Conclusion:- This experiment successfully demonstrates how to deploy a website using **Google App Engine**. It highlights key steps including **project setup, configuration, deployment, and troubleshooting**. By resolving issues like **runtime errors and permission constraints**, the deployment process is better understood. This experiment provides valuable insights into **cloud-based hosting**, enabling students to implement scalable web solutions using Google Cloud Platform