**Implementation guidelines:**

Google has confirmed the details for our dedicated sandbox environment.

Here's what we need to know:

1. **Pre-Configured Projects & Credits:** Each team will receive a pre-created Google Cloud project, complete with owner access assigned to your team's email IDs. This project comes pre-loaded with limited credits, providing us with the resources needed to leverage a wide array of Google Cloud services.
2. **Ready to Go:** As soon as you receive access to your project, you're all set to dive in and begin building! Follow the provided instructions to kickstart your development.

Make note of below details from given project

Project Name :

Project Id :

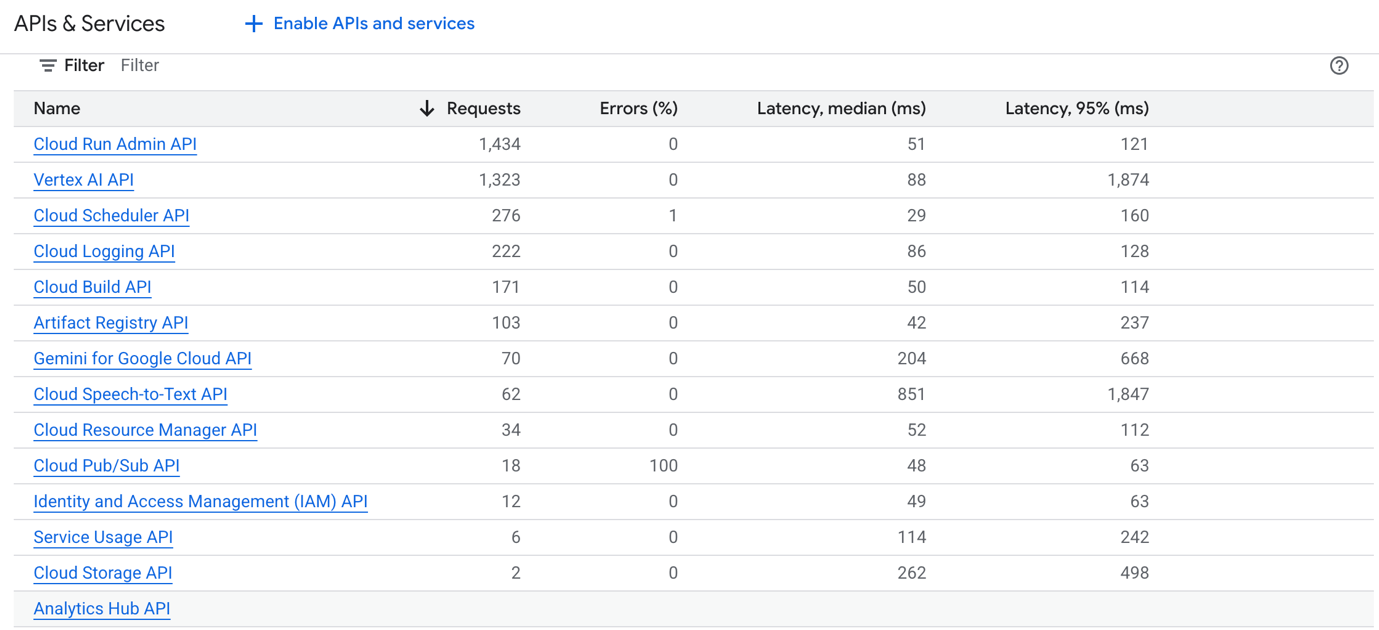
Project Number :

Project Region:

Cloud Storage Bucket names :

Cloud Storage Bucket locations:

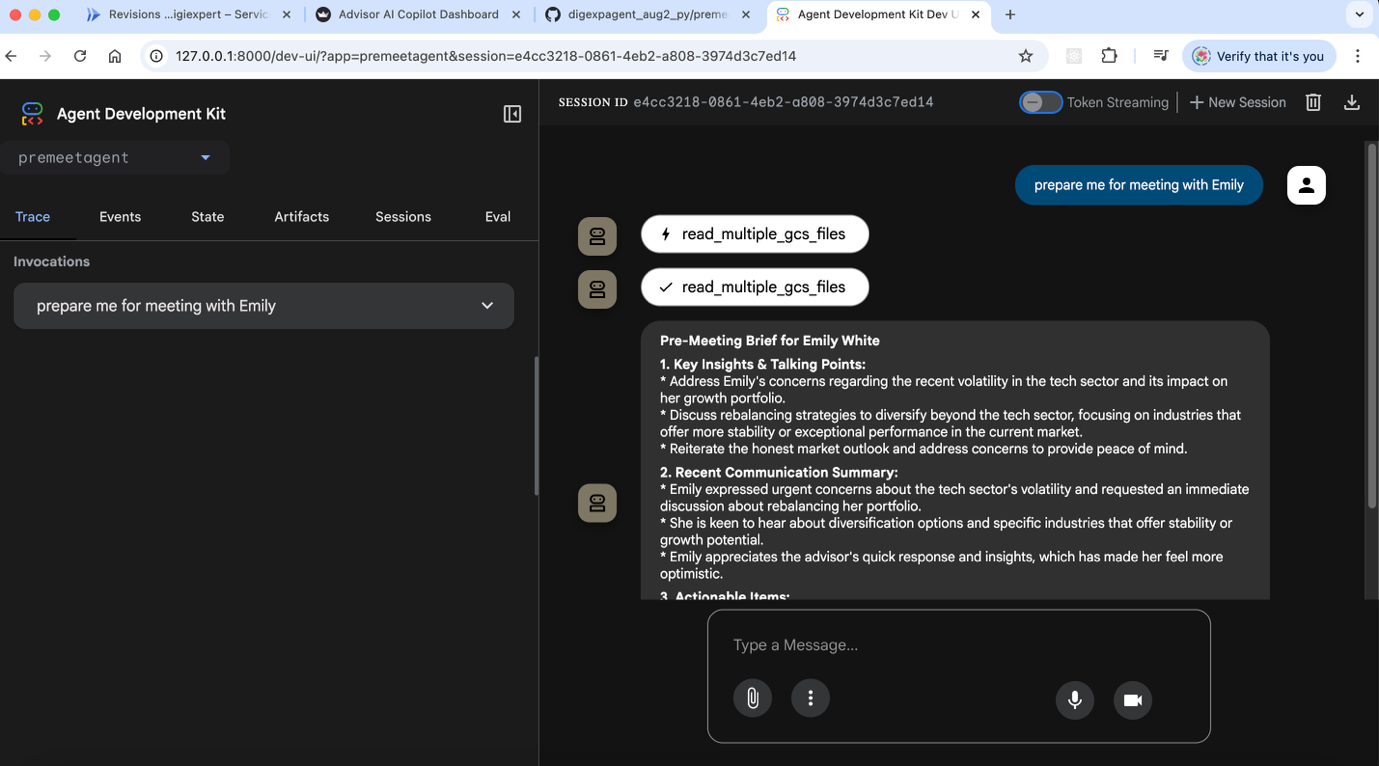
Proceed to enable below API’s from APIs & Services in GCP



**AI Agent Creation and Deployment**

Follow these steps for creation, local testing, and deployment to Agent Engine for all 4 agents:

* + **Example Python Script for Agents:** A sample Python script to create agents can be found here: <https://github.com/sivaprasadanakarla/digexpagent_aug2_py/blob/master/premeetagent/agent.py>
  + **Local Testing:** Test your agent locally using the command: adk web If there are no errors, adk web will launch the agents in a UI mode for interaction and testing.



* + **Deployment to Agent Engine:** Once local testing is successful, deploy the agent to Agent Engine using the following command:

adk deploy agent\_engine alwaysonagent --project digitalexperts-467610 --region us-central1 --staging\_bucket gs://digexbuck1 --display\_name alwaysonagent --description alwaysonagent

Upon successful deployment, you will receive an agent reference number.

* + **Testing Deployed Agent in Cloud:** You can test the agent deployed in the cloud using this sample Python script: <https://github.com/sivaprasadanakarla/digexpagent_aug2_py/blob/master/premeetagent/test_deploy.py>

## **Prompts created for Agents (**to be tested thoroughly during Hackathon**)**

**Pre-Meet Agent:**

## 

**In-Meet Agent:**



**Post-Meet Agent:**



**Always-on Agent:**



## **Dashboard Application Deployment**

The dashboard application, built with Streamlit, can be deployed to Google Cloud Run using Docker. A sample project and code are available at:

<https://github.com/sivaprasadanakarla/uidigiexpert>

Follow these steps to build and deploy Docker image:

1. **Configure Docker for Google Cloud:**

gcloud auth configure-docker

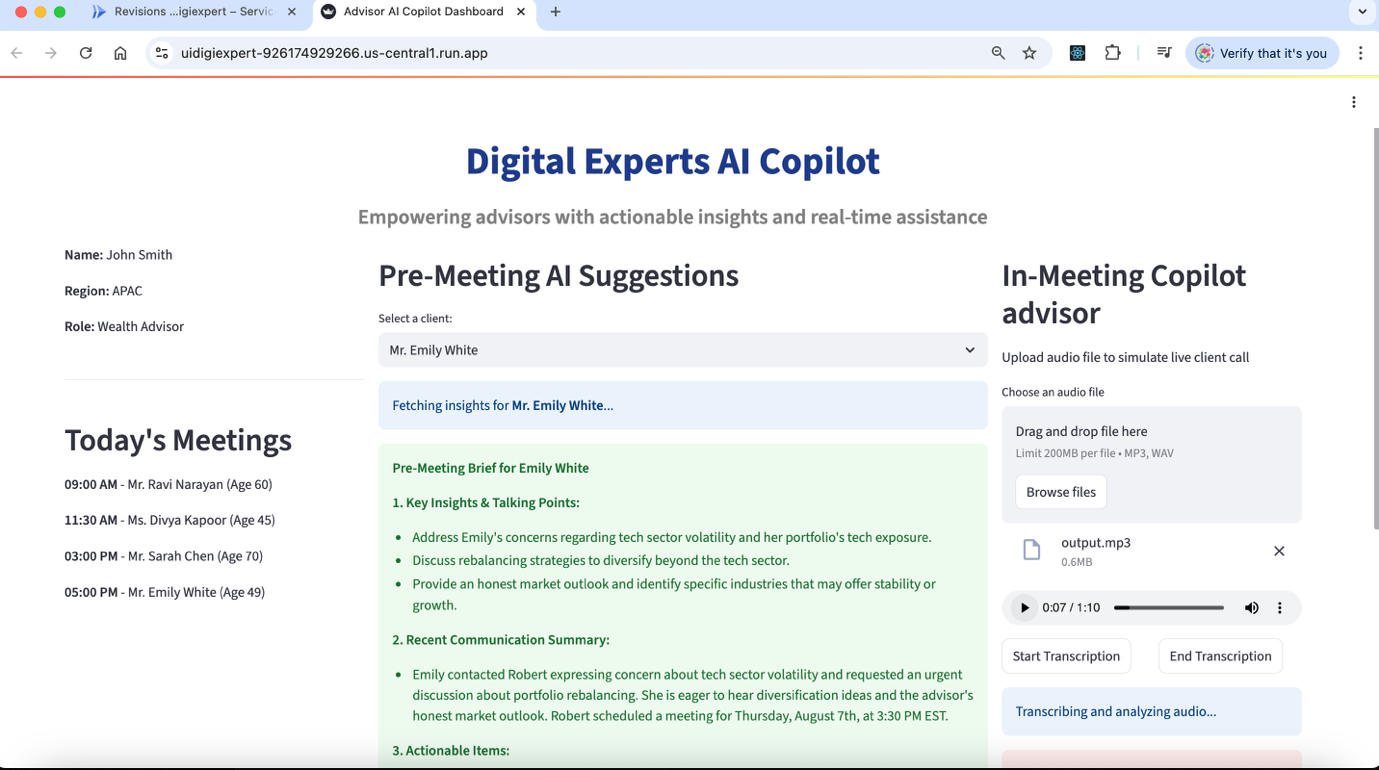
1. **Build the Docker Image:**

docker build --tag gcr.io/digitalexperts-467610/uidigiexpert:latest .

1. **Push the Docker Image to Google Cloud Artifact Registry:**

docker push gcr.io/digitalexperts-467610/uidigiexpert:latest

Once the Docker image is successfully pushed, proceed to deploy this image to Google Cloud Run. Upon successful deployment, your dashboard application will be live and accessible.



**Always-On Scheduler Service:**

This service automates the process of scanning Cloud Storage, applying custom business rules, and sending timely notifications, with full history tracking for easy dashboard viewing.

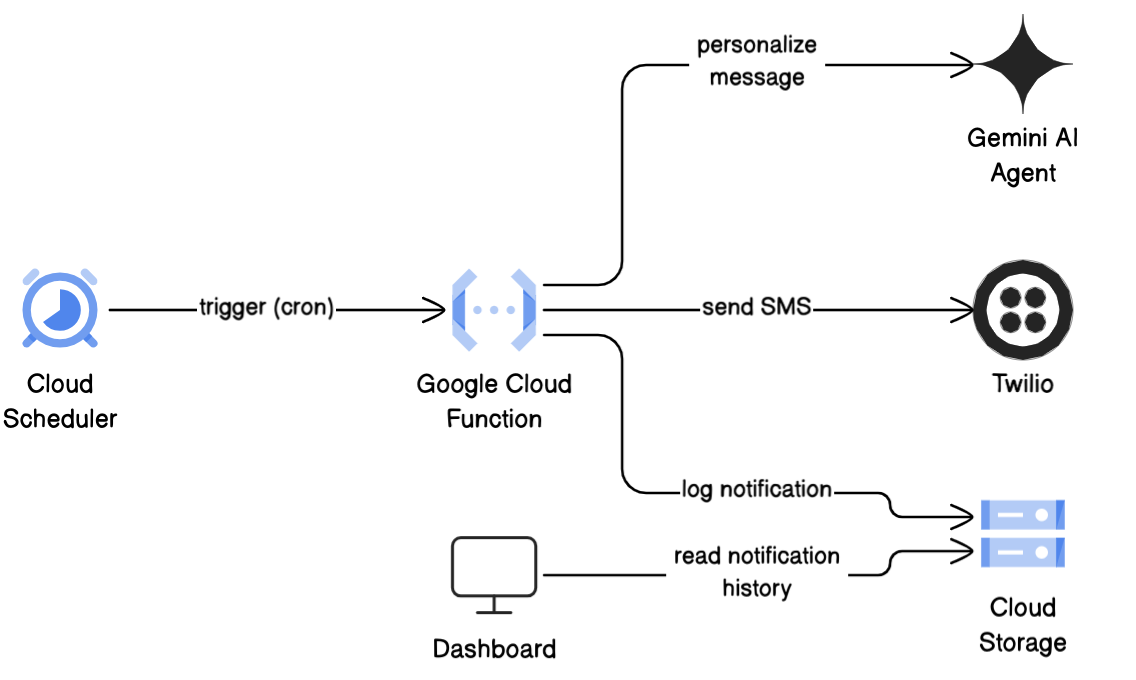
**Here's the streamlined process:**

1. **Cloud Function :** A serverless function (e.g., Python) regularly scans our specified Cloud Storage buckets. It runs unique business logic to identify conditions that require attention (e.g., kyc expiry, birthday etc).

Sample code for cloud function is available at [**https://github.com/sivaprasadanakarla/aocloudfunction**](https://github.com/sivaprasadanakarla/aocloudfunction)

1. **Cloud Scheduler :** This service acts as a reliable cron job, triggering the Cloud Function daily (or at our chosen frequency) to ensure continuous monitoring without manual intervention.
2. **Twilio :** When the Cloud Function's business logic identifies a notification trigger, it securely uses Twilio to send out SMS to the designated recipients.
3. **Cloud Storage :** Every notification sent (or attempted) is logged and stored in a dedicated Cloud Storage bucket. This creates a comprehensive history of all service activities.
4. **Dashboard :** reads this notification history from Cloud Storage, allowing you to easily view and track all alerts from the last 7 days, providing clear oversight .

This setup ensures your critical data is always monitored, and relevant stakeholders are promptly informed



## Twilio Setup

To integrate Twilio for sending notifications, follow these steps:

1. **Create a Free Twilio Account:** Sign up for a free account on the Twilio website.
2. **Generate a Twilio Phone Number:** Obtain a Twilio phone number from your account dashboard. This number will be used as the sender for your SMS notifications.
3. **Capture Account SID and Auth Token:** Locate your Account SID and Auth Token in your Twilio dashboard. These credentials are essential for authenticating our requests when invoking the Twilio SMS service from Google Cloud Function.