Assignment 4

Using Generative AI APIs
Due: 31st March 23 04:59 pm

Submission:
1. Github Repo Link
2. 10 min video recorded demo

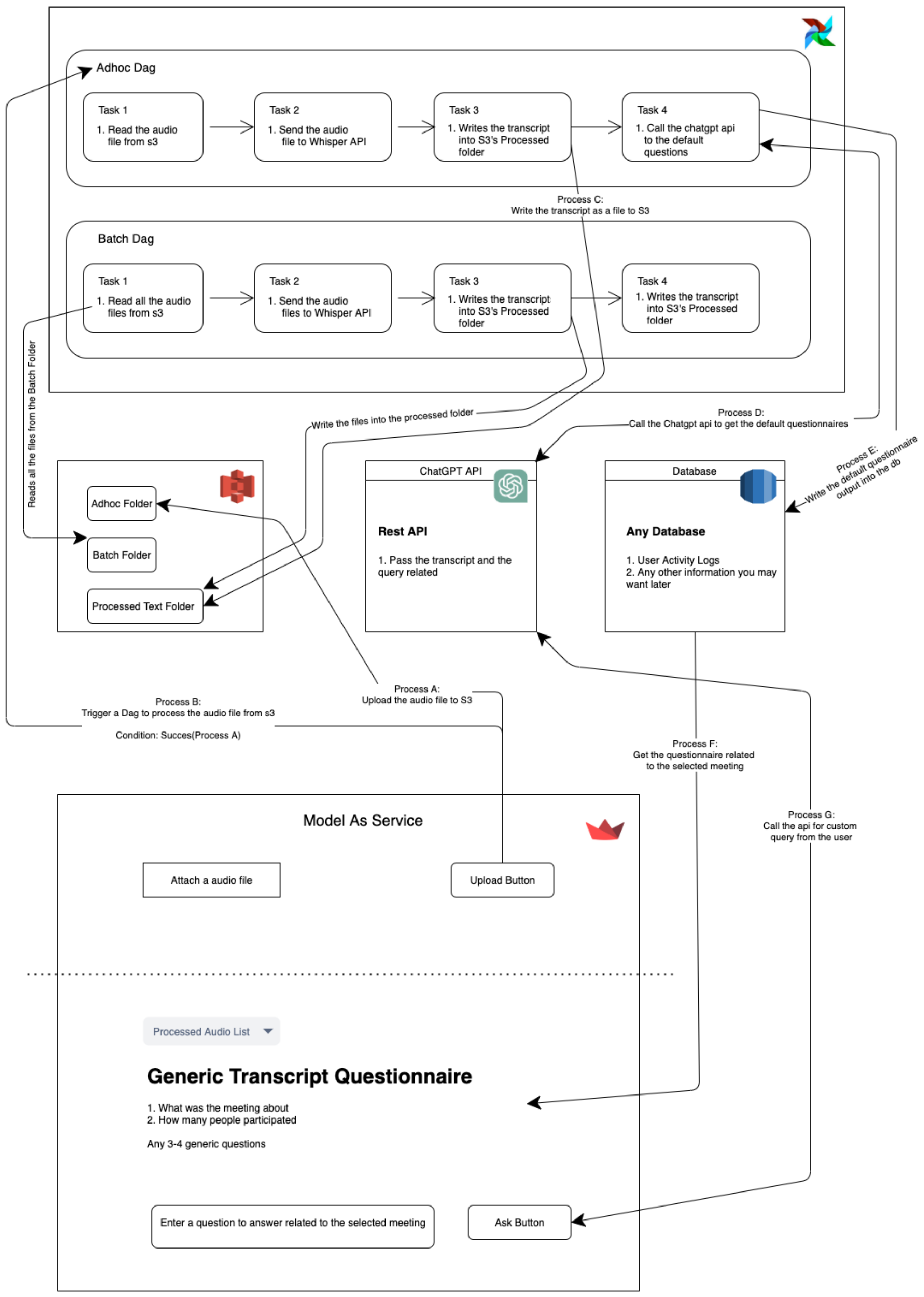
Project summary:
You intend to build a Meeting Intelligence Application. To test it, you will record 4, 10 min long meetings and use Whisper and GPT 3.5 APIs integrating with Streamlit and Airflow. A reference architecture is shown below. Note that Once you record a mp3 file of the meeting, you will use the Whisper API to convert to transcript. Review the transcript and comment on the quality of voice-to-text.

Use the transcripts with GPT 3.5 to build a Query engine, try out different tasks.
See **https://platform.openai.com/examples for inspiration.**
See below for a summarization example.
https://bigcodegen.medium.com/transcribing-youtube-video-using-whisper-for-gpt-3-text-summarization-ad80dfcba9ed
Implement a Streamlit app to illustrate the workflow
Implement and automate it with Airflow (See ref architecture below)

Grading Percentage:

1. Airflow - 40 points
2. Streamlit - 20 points
3. Deployment on cloud - 10 points
4. Architecture diagram - 10 points
5. Github Repo structure - 10 points
6. Codelsb documentation - 10 points

Process Workflow:


Adhoc Dag
Task 1
1. Read the audio file from s3
Task 2
1. Send the audio file to Whisper API
Task 3
1. Writes the transcript into S3's Processed folder
Task 4
1. Call the chatgpt api to the default questions
Process C:
Write the transcript as a file to S3
Batch Dag
Task 1
1. Read all the audio files from s3
Task 2
1. Send the audio files to Whisper API
Task 3
1. Writes the transcript into S3's Processed folder
Task 4
1. Writes the transcript into S3's Processed folder
Write the files into the processed folder
Process D:
Call the Chatgpt api to get the default questionnaires
Process E:
Write the default questionnaire output into the db
Reads all the files from the Batch Folder
Adhoc Folder
Batch Folder
Processed Text Folder
ChatGPT API
Rest API
1. Pass the transcript and the query related
Database
Any Database
1. User Activity Logs
2. Any other information you may want later
Process A:
Upload the audio file to S3
Process B:
Trigger a Dag to process the audio file from s3
Condition: Succes(Process A)
Process F:
Get the questionnaire related to the selected meeting
Process G:
Call the api for custom query from the user
Model As Service
Attach a audio file
Upload Button
Processed Audio List
Generic Transcript Questionnaire
1. What was the meeting about
2. How many people participated
Any 3-4 generic questions
Enter a question to answer related to the selected meeting
Ask Button

Whisper APIs

1. Documentation[1]
2. Pass on the audio media file into text (See https://whisper-openai.vercel.app/ or https://replicate.com/openai/whisper for examples)

Chat APIs

1. Documentation[2]
2. Pass on the questions along with the meeting transcript

Airflow

1. Airflow has 2 dags
    a. adhoc process - Can be triggered using REST API calls[3]
    b. batch process - scheduled using cron
2. The process related to converting audio file into text and answering the generic questionnaire

Streamlit

1. Upload a audio file to S3
2. Select a meeting from a list of processed meetings
3. A prompt to ask any question related to the selected meeting
4. Application design to be stateless, ie no data stored within the streamlit application

Other Deliverables

1. All application should be deployed on cloud and accessible to public (No localhost)
2. Links to Streamlit / Fastapi / Airflow / Codelab docs in the github README.md file
3. Use Github Issues to log a bug[4] / conversation[5] on your peer repository, following respective templates.
4. Fix for bug should be done using PR's and tagging the the issues[6]
5. Donot publish your virtual environments / API key on github.

Cloud Services:

Free to choose any cloud platform or services type

Examples

Airflow - Cloud Composer[7]

Database - RDS[8] or Cloud SQL[9]

Streamlit - Streamlit Cloud or Cloud Run[10]

---

[1] https://platform.openai.com/docs/api-reference/audio
[2] https://platform.openai.com/docs/api-reference/chat
[3] https://airflow.apache.org/docs/apache-airflow/stable/stable-rest-api-ref.html
[4] https://github.com/stevemao/github-issue-templates/blob/master/bugs-only/ISSUE_TEMPLATE.md
[5] https://github.com/stevemao/github-issue-templates/blob/master/conversational/ISSUE_TEMPLATE.md
[6] https://docs.github.com/en/issues/tracking-your-work-with-issues/linking-a-pull-request-to-an-issue
[7] https://cloud.google.com/composer
[8] https://aws.amazon.com/rds/
[9] https://cloud.google.com/sql
[10] https://cloud.google.com/run

Additional notes:

1. Required attestation and contribution declaration on the GitHub page:

Compute Engine - Run all services in a virtual machine self-managed

Presentation to cover the following:

1. Current architecture diagram
2. Demo of the application
3. S3 bucket design
4. Airflow Dags task
5. Whisper and ChatGPT api call processes

WE ATTEST THAT WE HAVEN'T USED ANY OTHER STUDENTS' WORK IN OUR ASSIGNMENT AND ABIDE BY THE POLICIES LISTED IN THE STUDENT HANDBOOK
Contribution:
- member1: 25%
- member2: 25%
- member3: 25%
- member4: 25%

2. Keep your repository private until the submission. Incase of plagiarism both the team would be equally held responsible.
3. Make sure you do not push anything to your GitHub after submission date even the readme.md. Work on a bug fix branch incase you want to.
4. Create a Codelab document describing everything you did. In your GitHub you should have a readme.md files which would tell what all things are there in this GitHub repository.
5. Incase you are unable to present in class, the recorded video would be used for grading.