PROJECT PLAN DOCUMENT

Project Number	2
Project Title	Speech2Speech Buffer
Document	DASS project Plan Document
Creation Date	14-02-2025
Created By	Akash, Varun
Client	**Satish kathirisetti , Product Labs, IIITH Nikunj Mangukiya (Mentor).**

Project Description: Real-Time Audio Processing System Using RabbitMQ Queues

- This project involves creating a real-time audio processing backend system using RabbitMQ for queue-based communication and using Fast API for efficiency. We also need to ensure reliability and minimize disruptions, a buffering mechanism is implemented to maintain continuous audio delivery, even if system failures occur.

System Flow:

1. **Input Audio Capture**:
   a. A listener speaks into a microphone, and their audio is captured.
   b. The audio data is divided into manageable chunks for processing.
   c. Each chunk is sent to an initial RabbitMQ queue for further handling.

2. **Audio Chunk Processing Service**:
   a. A Python service (using FastAPI) monitors the RabbitMQ queue for new audio chunks.

    b. Upon receiving a chunk(eg: Sent to ASR_output queue), Python server will read the chunk from the ASR_output queue and write it to MT_input queue.

    c. Now from the MT_input queue we must process the data by sending it to Machine Translator. For Phase-1 we are using just another python server which will just transfer it to MT_output (with some latencies for experiencing real-time).
    d. For Phase-2 we will write a new server code which will send the MT_input queue to Machine translator and take the processed data from Machine Translator and write it to MT_output queue.

3. **Translation Buffer**:
    a. Translated chunks (after processed by 'speech to text') are stored in a buffer.
    b. The system waits until 4-5 translated chunks are accumulated in the buffer before playback begins to the listener, ensuring smoother and coherent audio delivery.

Key Areas We will focus on:

Queue Management:

1. We will use RabbitMQ for efficient and reliable message delivery between components. And implement Proper acknowledgment and retry mechanisms which will minimize the data loss.

Latency Handling:

1. We will design the system to minimize latency while balancing processing time and playback smoothness (Delivering the translated text back to the listener with minimal waiting time, so it feels almost real-time.).
2. Introducing a small buffer of 4-5 chunks avoids interruptions and allows for error correction or adjustments.
3. We will include FastAPI for high performance and low latency. Since It is asynchronous it can handle many requests at once without getting blocked so which will make it ideal for our real-time application.

Scalability:

- **Managing Higher Input Loads**: If the amount of audio data being sent to the system increases, we will try to add extra servers to handle this larger data without crashes or delays.

Error and Data Loss Mitigation:

1. Using persistent queues in RabbitMQ to prevent data loss in case of a system crash.
2. Implement retries for failed message processing.
3. Monitor queue statuses and implement alerts for anomalies.
4. We will use FastAPI integrationsince it provides features to handle errors and exceptions gracefully.

Team Communication s Meetings

Meeting Days and Time:

- **Wednesdays**: 2:00 PM to 5:00 PM
- **Sundays**: 2:00 PM to 5:00 PM
- These time slots we will meet and do project together, share updates.
- Meetings are hybrid.

Team Communication Process

1. Review progress.
2. Discuss tasks for the week.
3. Work through coding, debugging, system integration.
4. Discuss new ideas, improvements, changes to the project.

Development Tools

<u>System requirements:</u>

- Operating System : Platform independent(Can be Mac, Linux or Windows).

- Development Environment – VS code, GitHub
- Programming language:

1. Backend – Python.
2. EMQ – Rabbit MQ.

- Python Libraries Used :

  1. Pika
  2. FastAPI
  3. Asyncio
  4. Regex(re)
  5. Json
  6. Logging
  7. Collections
  8. Os (file handling)
  9.requests

-

Milestone Schedule (Tentative)

Milestone	Due Date	Release	Deliverable
Learning Technologies	Week 1-2	R1	Research on RabbitMQ , Fast API,Git and revisiting Python.
Backend Setup and Environment Configuration	Week 3	R1	Implement RabbitMQ queues, Develop python server(basic code) without Fast API.
Implement Queue Mechanism	Week 4	R1	Develop backend logic with Fast API to capture and queue JSON chunks and a portion of Edge Cases and Error Handling
Testing and Debugging (Phase 1)	Week 5	R1	Work on Network Latency(Timeouts) and Work on code modularity and update all docs related to R1
Getting started with Phase-2	Week 6	R2	Working on Requirments and Design docs for Phase-2
Communicate between EMQ and Server(MT,TTS)	Week 7	R2	Implement backend logic (new python server including API to sandbox) to send and receive responses from a Prototype of a Sandbox
End-to-End Speech Processing Integration	Week 8	**R2**	Integrate ASR, MT, and TTS APIs with the real sandbox for speech-to-text, translation, and

			text-to-speech processing, including model fetching and authentication.
Real-Time Speech to Speech Translation with RabbitMQ	Week 9	**R2**	Integrate ASR, MT, TTS APIs with RabbitMQ for real-time processing, handle different wpm cases and multiple packets, manage edge cases, modify “generateaudio” code to push audio files for ASR input queue.
Performance Optimization and Robustness	Week 10	R2	Implement chunking for large payloads, add timeout mechanism for API requests, introduce rate-limiting to control API calls per minute, test on different environment.
Integrating Streaming API and optimizing ASR	Week 11	R2	Integrate Streaming API, change and optimize ASR Processing,Support parallel requests and conduct multi-system testing
Testing and Chunking	Week-12	R2	Write Unit tests,Integrated testing and end-2-end testing and update chunking logic
Optimizing and Documentation	Week 13	R2	Revise file naming and testcase files,Update SRS,Project Plan,Product design,Project synopsis,Instructions documents and code review from client,work on R2 presentation, demo video

Team Members role on the Project for Week 3,4 s 5.

A. Develop and execute test cases to stress test the system.
   - Responsible: Arya, Karthik
   - Deadline: End of Phase-1

B. Document all tests thoroughly, including results and observations.
   - Responsible: Varun, Akash
   - Deadline: End of Phase-1

C. Ensure robust error handling mechanisms are in place.
   - Responsible: Arya, Varun
   - Deadline: End of Phase-1

D. Set up RabbitMQ on a different machine for testing.
   - Responsible: Prashant
   - Deadline: End of Phase-1

Team Members role on the Project for Week 6, 7, 8, 9 & 13.

A. Prepare detailed requirements and design documents for Phase-2.
   - Responsible: Team
   - Deadline: End of week 6.

B. Write Generateaudio code with different WPM.
   - Responsible: Akash,Prashant
   - Deadline: End of week 6.

C. Integrate and test ASR, MT and TTS APIs with the sandbox.
   - Responsible: Varun,Akash
   - Deadline: End of week 8.

D. Integrate APIs with RabbitMQ and handle edge cases.
   - Responsible: Varun,Arya
   - Deadline: End of week 9.

E. Implement Chunking, timeout handling, Rate-limiting for API request.
   - Responsible: Karthik,Arya
   - Deadline: End of week 10.

F. Error handling and ensuring robustness.
   - Responsible: Karthik,Prashant

- Deadline: End of week 10.

G. Integrate Streaming API's

- Responsible: Varun,Akash
- Deadline: End of week 12.

H. Change and optimize ASR processing

- Responsible: Karthik,Arya
- Deadline: End of week 12.

I. Conduct Multi-system Testing

- Responsible: Prashant
- Deadline: End of week 11.

j. Perform UnitTesting

- Responsible: Varun, Akash
- Deadline: End of week 12.

k. Perform IntegratedTesting

- Responsible: Arya,Varun
- Deadline: End of week 12.

L. Perform End-2-End Testing

- Responsible: Karthik,Prashant
- Deadline: End of week 12.

M. Update Chunking Logic

- Responsible: Varun
- Deadline: End of week 12.

N. Update all documents

- Responsible: varun,Prashant
- Deadline: End of week 13.

O. Update File names and code review from client

- Responsible: Varun,Akash
- Deadline: End of week 13.

P. Work on R2 Presentation and Demo video

- Responsible: Karthik,Arya
- Deadline: End of week 13.