APRIL 18

HuddleAI

Asynchronous Standup Summaries for Modern Dev Teams

Product Owner
Sean Andre Membrido

Scrum Master
Aditya Sarin

Contributor
Parul Datta

Contributor
Paolo Pedroso

Contributor
Pranav Sathianathan

Why HuddleAI?

The Problem

Daily standups are disruptive, repetitive, and often inefficient – especially for remote teams

Main Points

1) Lost dev time spent in repetitive meetings
2) Time zone conflicts hinder participation
3) Silent contributors attend without giving input
4) No persistent record of what was discussed

Who this helps?

– Remote/Hybrid Dev Teams

Stay aligned across time zones without relying on daily sync calls. HuddleAI ensures every team member gets a summary – even if they miss the meeting.

– Engineers w/ async workflows

Avoid context-switching and maintain focus. Engineers can get concise updates without interrupting their coding flow.

– Project Managers

Instantly see progress, blockers, and who’s doing what – without chasing updates. HuddleAI acts like your automated note-taker.

02

Project Scope

Project Scope

Build an AI–powered assistant that summarizes daily standup meetings and delivers updates directly to Slack

Sprint 1

User Stories (Goal: Set up audio/video processing pipeline and basic transcription.)

As a user, I can:

- I want to upload a meeting recording (audio/video) so HuddleAI can process it.
- I want to see a raw transcript of the meeting to verify accuracy.

Spikes

- Research speech-to-text tools (Whisper vs. Google STT: accuracy, diarization support).
- Evaluate diarization libraries.

Infrastructure Tasks

- Set up backend API for file uploads
- Configure cloud storage for recordings
- Build basic transcript output (JSON/text file)

Sprint 2

User Stories (Goal: Summarize Meetings and Post to Slack)

As a user, I can:

- receive a structured summary (updates, blockers, action items) of a meeting I upload.
- see who said what in the summary (speaker attribution).
- receive the summary automatically in a designated Slack channel.

Spikes

- Refine prompt engineering for extracting key standup elements using GPT.
- Investigate Slack API features: threaded messages, formatting, and reactions.

Infrastructure Tasks

- Implement Slack webhook or bot token integration.
- Develop summary formatting logic and message sending module in backend.
- Add basic error logging and retry mechanisms for Slack delivery failures.

Sprint 3

User Stories (Goal: Polish and Expand User Experience)

As a user, I can:

- receive well-formatted summaries (Markdown, bullet points, etc.) in Slack.
- access a web interface to view past summaries. (stretch)
- tag summaries with topics (e.g., “Frontend,” “DevOps”). (stretch)

Spikes

- Explore methods for topic/tag classification (e.g., using GPT or keyword heuristics).
- Test file storage and summary retrieval with Firebase or PostgreSQL.
- Evaluate feasibility of persistent storage and filtering features.

Infrastructure Tasks

- Implement UI wireframes for the dashboard (React, if time permits).
- Add DB schema and integration layer for storing summaries.
- Improve summary formatting consistency and error handling.

Sprint 4

User Stories(Goal: Finalize and Polish Product for Delivery)

- use a /huddle summary Slack command to retrieve the latest meeting summary. (stretch)
- receive weekly digest summaries from HuddleAI via Slack. (stretch)
- As a team, we can test and verify that all MVP features work reliably across different inputs.

Spikes

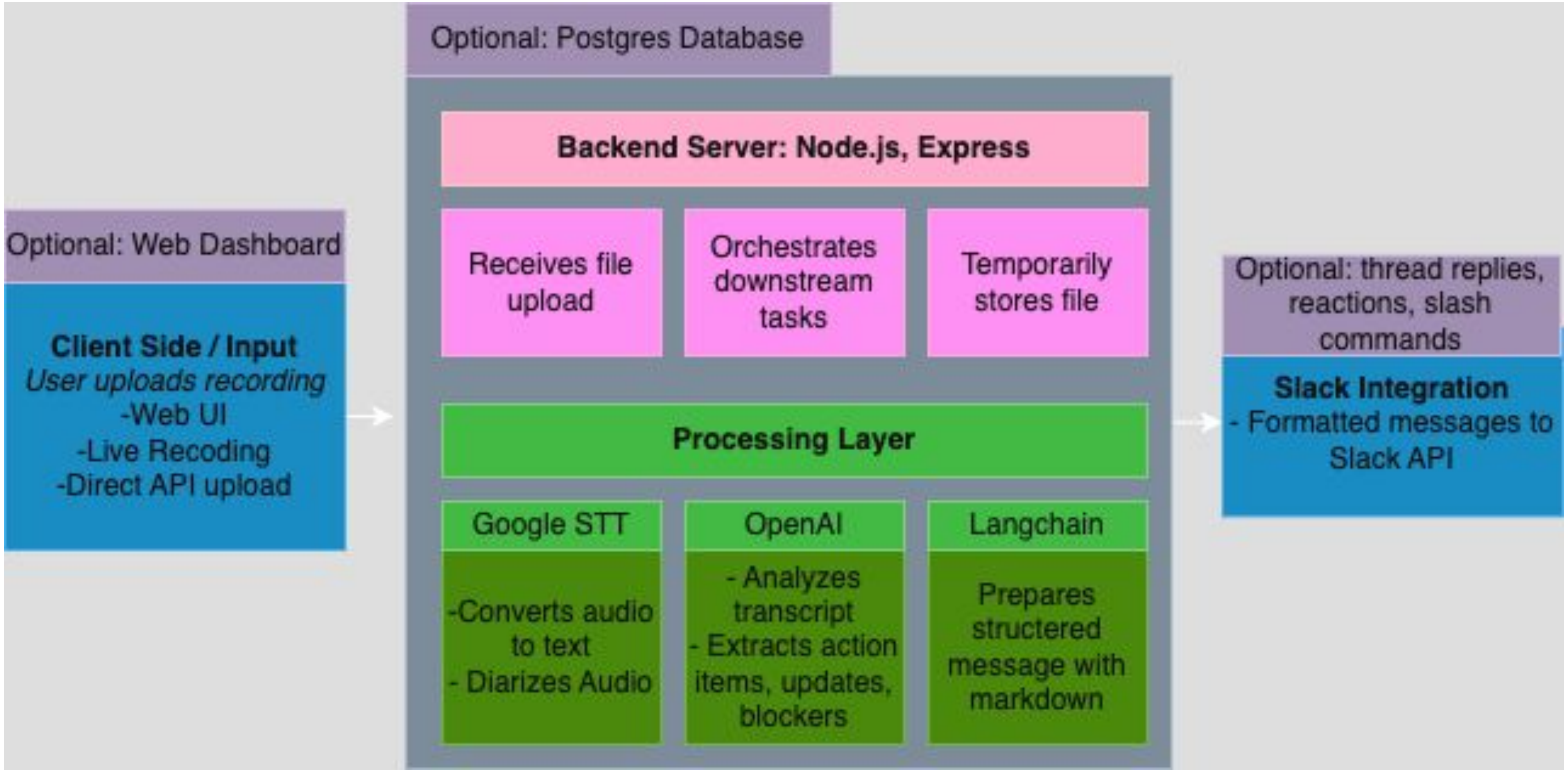
- Explore Slack command implementation and permissions setup.
- Research best practices for asynchronous notifications and digest generation.
- Conduct user feedback sessions to identify improvements and UX tweaks.

Infrastructure Tasks

- Add automated testing and CI checks for core features.
- Finalize deployment pipeline (e.g., Vercel/Render/Heroku for backend).
- Perform final bug fixes, cleanup unused code, and write project documentation.

01

Architecture



Optional: Web Dashboard

Client Side / Input
User uploads recording
-Web UI
-Live Recoding
-Direct API upload

Optional: Postgres Database

Backend Server: Node.js, Express

Receives file upload

Orchestrates downstream tasks

Temporarily stores file

Processing Layer

Google STT

-Converts audio to text
- Diarizes Audio

OpenAI

- Analyzes transcript
- Extracts action items, updates, blockers

Langchain

Prepares structered message with markdown

Optional: thread replies, reactions, slash commands

Slack Integration
- Formatted messages to Slack API

Technologies

Backend

- Python (Flask or FastAPI)
- Open AI Whisper
- Firebase Storage or Firestore

Front-End

- React
- Next.js
- Tailwind

Integrations

- Slack API
- OpenAI API

Challenges/Risks

- Speech to Text Accuracy
  - Low transcription quality from poor audio, overlapping speakers, or accents
- Speaker Diarization
  - Difficulty accurately assigning statements to the right person
- LLM Prompt Reliability
  - AI summaries may be inconsistent, vague, or too verbose
- API Limitations
  - Message formatting, rate limits, etc.

HuddleAI

Minimum Viable Product

- At minimum, HuddleAI needs to be a functional tool that can accept a recording of a standup meeting, process it to identify key information, and deliver a structured summary to a Slack channel.
- Success means engineering teams can stay informed about standup content without attending live meetings, saving time while maintaining team alignment.

Thank You!

Questions?