Voice-Based Sentiment Analysis for Customer Feedback

Checkpoint 2: Review Rationale, Research Output, and Design Documents

1. Rationale

Our project leverages AWS cloud services for sentiment analysis due to their scalability, security, and efficiency. The chosen services align with our project's objectives for the following reasons:

- **AWS Transcribe**: This service efficiently converts voice feedback into accurate text as it is essential for capturing customer insights directly from audio submissions.
- **AWS Comprehend**: Comprehend's powerful sentiment analysis capabilities allow us to assess customer emotions (Positive, Neutral, or Negative) with minimal customization.
- **AWS Polly**: Polly enables us to generate clear and engaging audio summaries from sentiment analysis results, enhancing the accessibility and usability of insights for business managers.
- **AWS Lambda and S3**: Lambda's serverless architecture ensures cost-effective computing, while S3 provides reliable storage for audio feedback files.

This combination of AWS services ensures that our system is robust, scalable, and capable of efficiently processing large volumes of voice feedback.

2. Research Output Review

Our team has made significant progress in the following areas:

- **System Design Research:** Conducted extensive research on AI-driven sentiment analysis models, identifying AWS Comprehend as the most suitable service for our project.
- **Voice Feedback Handling:** Explored various audio file formats and optimized transcription settings within AWS Transcribe for improved accuracy.
- **Dashboard Design Exploration:** Researched user-friendly dashboard interfaces to display sentiment trends, ensuring actionable insights for businesses.

Individual Contributions

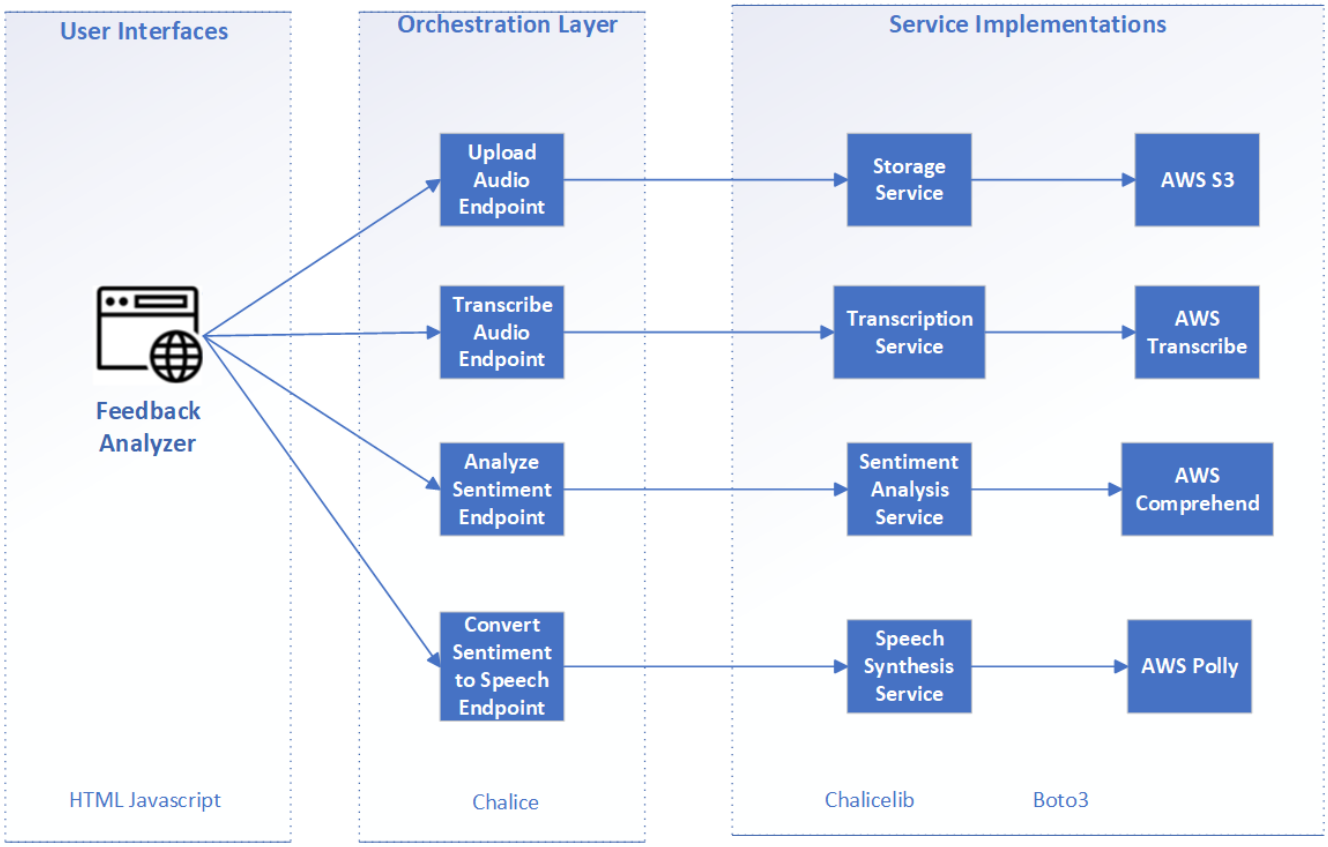
- **Adarsh Sindhu Prabhan:** Focused on integrating AWS Transcribe for voice-to-text conversion and researched data storage solutions using AWS S3.
- **Yajushi Garg:** Investigated AWS Comprehend's capabilities and configured sentiment analysis pipelines.
- **Vineet Sharma:** Designed preliminary wireframes for the user interface and dashboard visualization.
- **Kabir Rooprai:** Explored AWS Polly for generating audio summaries and integrated speech output logic.
- **Juwon Ajana:** Researched API integration strategies for connecting frontend and backend systems using AWS Lambda.

3. Design Documents Review

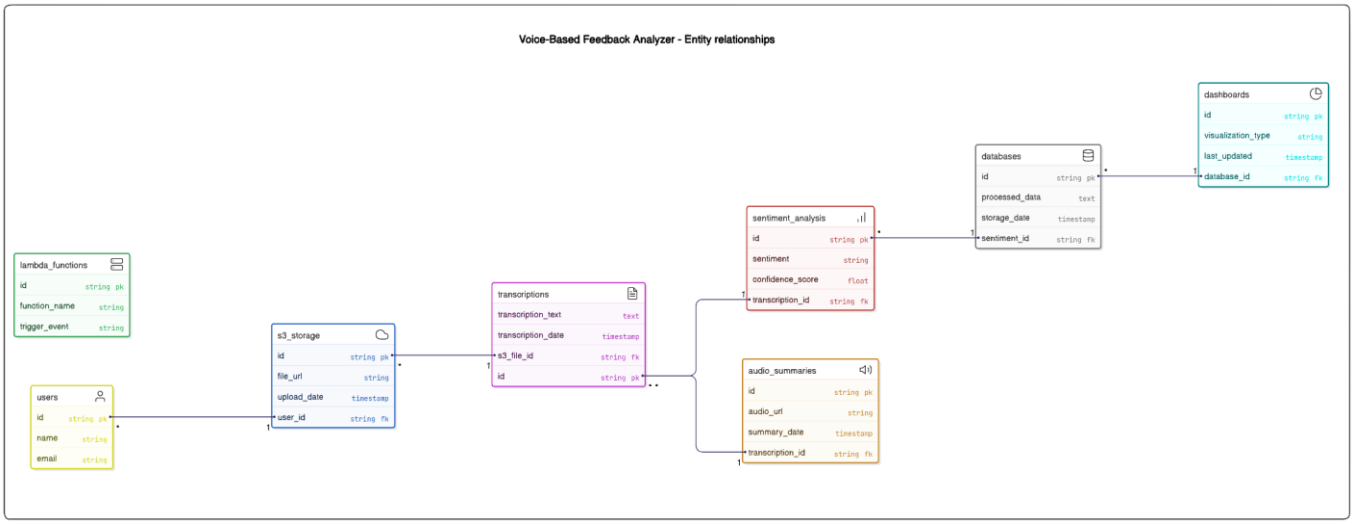
Our design documents include:

- **System Architecture Diagram:** Outlining interactions between AWS Transcribe, Comprehend, Polly, Lambda, and S3.
- **Component Diagrams:** Detailing system components such as data input, processing pipelines, and output visualization.
- **Workflow Plan:** Mapping the flow of data from user input through analysis to dashboard presentation.
- **Implementation Strategy:** Stepwise plan covering backend integration, frontend development, and AWS service configuration.

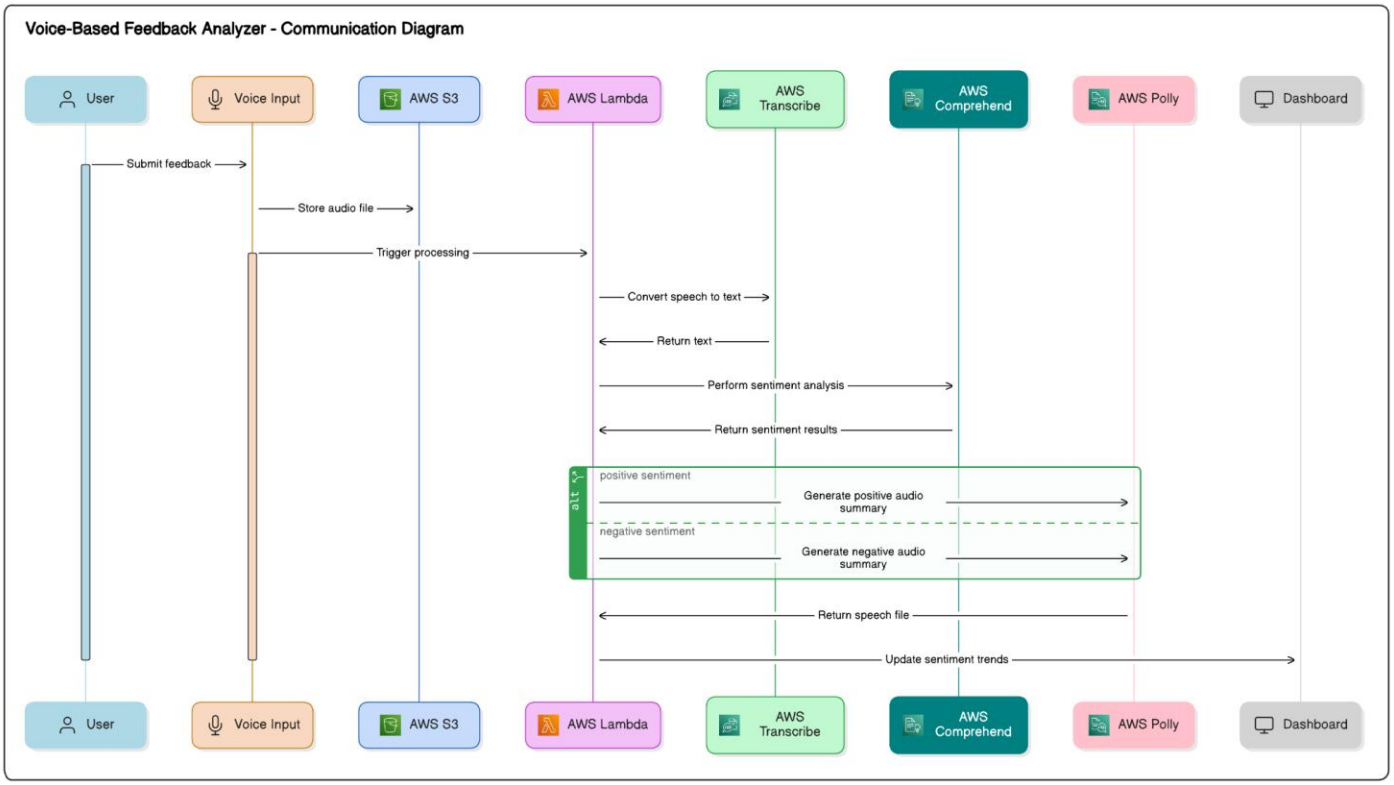
Architecture Diagram



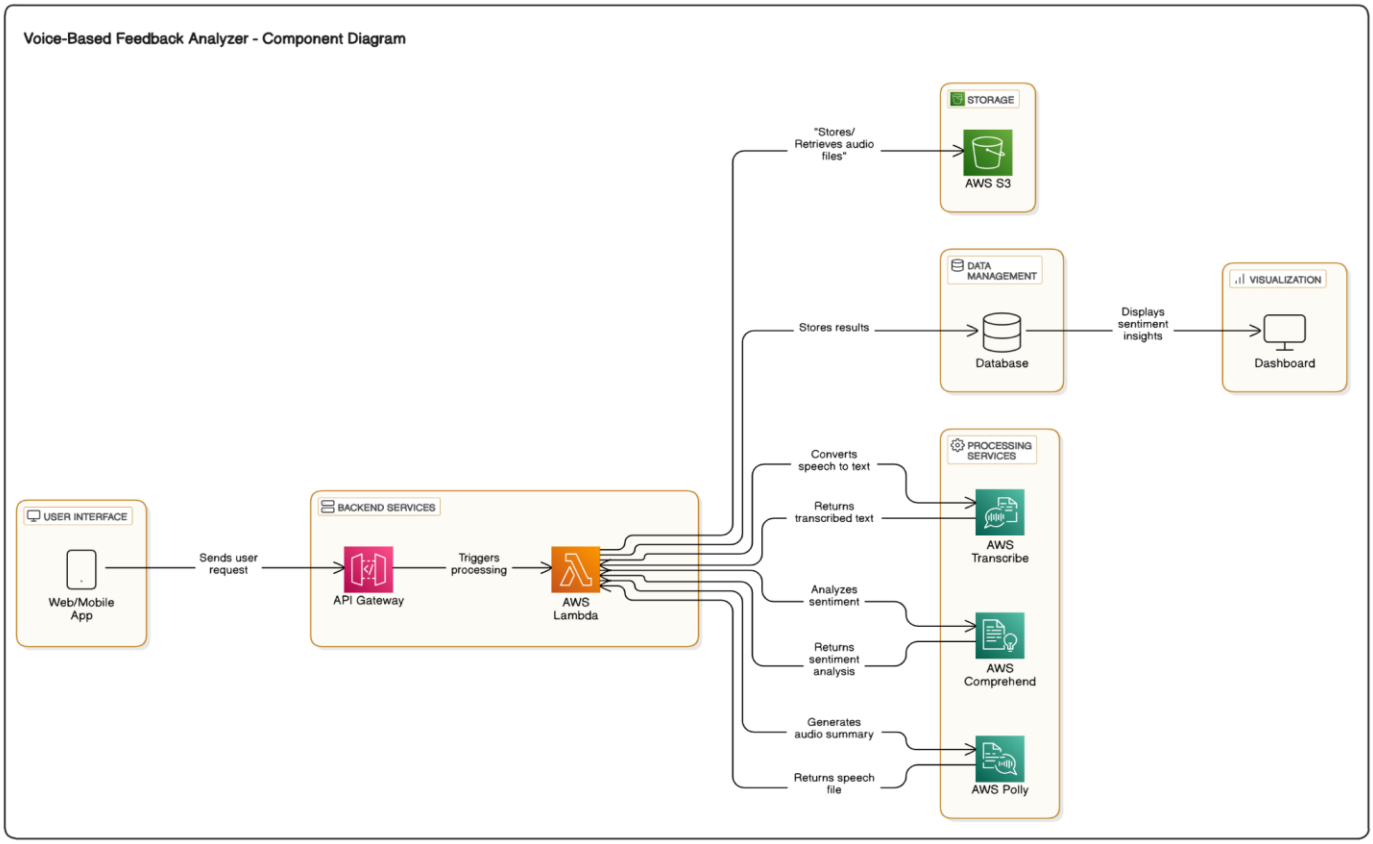
Entity Relationship Diagram



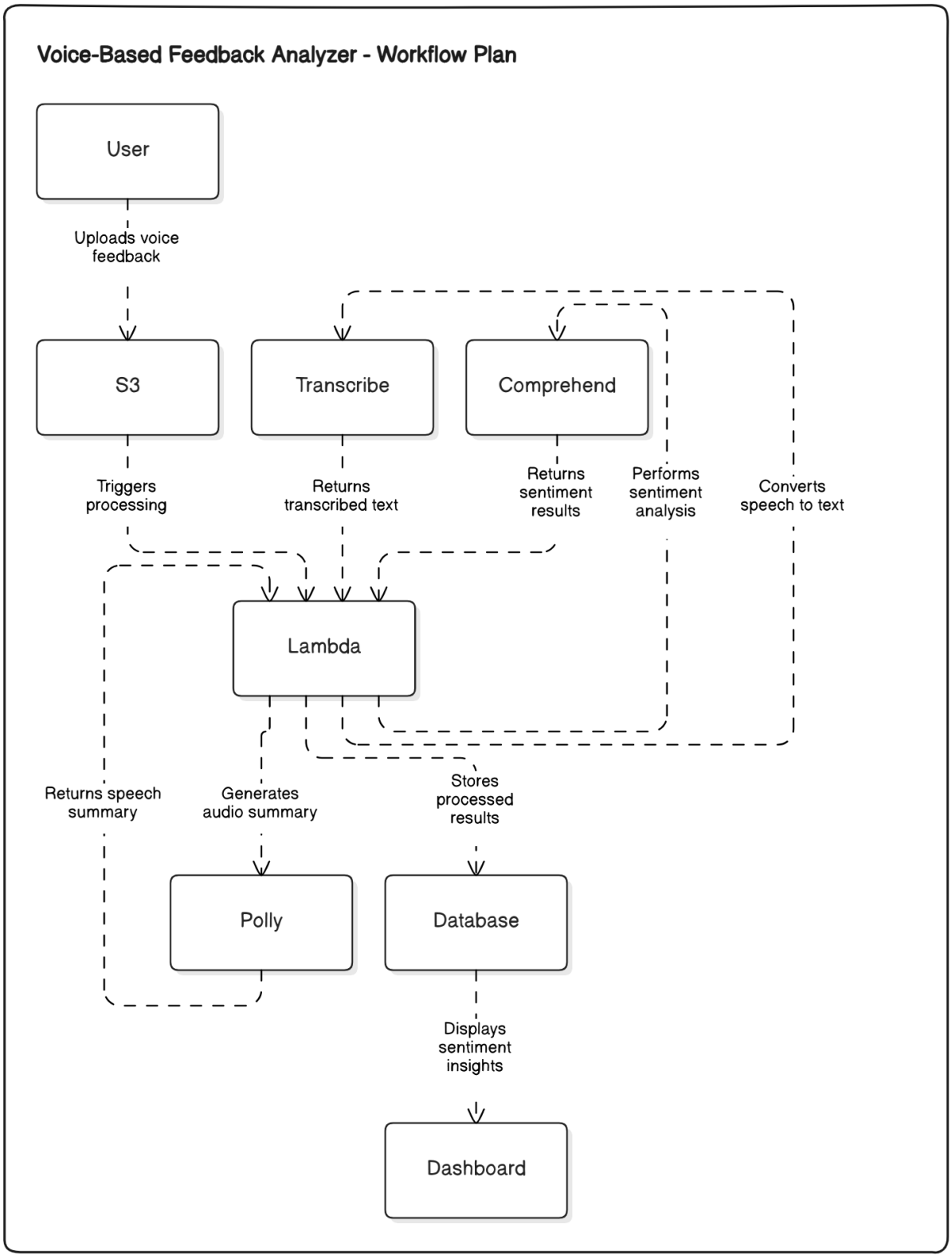
Communication Diagram



Component Diagram



Workflow Plan



Implementation Strategy

Phase 1: Backend Integration

1. Set Up AWS Services

- Create an S3 bucket for storing voice recordings.
- Configure AWS Transcribe for speech-to-text conversion.
- Enable AWS Comprehend for sentiment analysis.
- Set up AWS Polly for text-to-speech synthesis.
- Deploy AWS Lambda functions to orchestrate data flow between services.

2. Implement Serverless API with AWS Chalice

- Develop RESTful API endpoints for:
    - Uploading voice feedback to S3.
    - Triggering AWS Transcribe for transcription.
    - Processing sentiment analysis using AWS Comprehend.
    - Generating audio summaries using AWS Polly.
- Use Boto3 SDK to interact with AWS services.

3. Database Setup

- Choose DynamoDB or RDS (PostgreSQL/MySQL) to store processed sentiment results.
- Implement schema for storing user feedback, sentiment scores, and timestamps.

Phase 2: Frontend Development

1. Web & Mobile Application Development

- Create a user-friendly UI with HTML, JavaScript, and React (optional for dynamic UI).
- Integrate a voice recording feature for users to submit feedback.
- Implement API calls to interact with the backend.

2. Dashboard Development

- Design an interactive dashboard to visualize sentiment trends.
- Use Chart.js for graphical representation of sentiment data.
- Provide audio playback feature for sentiment summaries.

Phase 3: AWS Service Configuration & Deployment

1. IAM Roles & Security

- Set up IAM roles with least privilege access for Lambda functions.
- Implement CORS policies for secure API access.
- Enable AWS CloudWatch for logging and monitoring API activity.

2. API Gateway Deployment

- Deploy AWS Chalice application via API Gateway.
- Test API endpoints using Postman or curl.

Phase 4: Testing & Optimization

1. Unit & Integration Testing

- Write unit tests for backend services using pytest.
- Perform API integration testing using Postman/Newman.

2. Load & Performance Testing

- Simulate high-traffic scenarios using AWS Load Testing Tools.
- Optimize Lambda function execution time for cost efficiency.

3. Final Deployment & Monitoring

- Deploy the application to a production environment.
- Monitor API performance and user feedback using AWS CloudWatch & X-Ray.
- Continuously improve the system based on business insights.

Next Steps

- Finalizing the frontend design and dashboard implementation.
- Conducting initial testing of the integrated AWS services.
- Refining the workflow to improve data processing efficiency.

Voice-Based Sentiment Analysis for Customer Feedback

GROUP #2

Juwon Ajana

Yajushi Garg

Adarsh Sindhu Prabhan

Kabir Rooprai

Vineet Sharma

With the

guidance of Prof. Merlin James

Project Agenda

Our presentation will cover the following topics.

1. Project Overview
2. AWS Rationale
3. Implementation Strategy
4. System Architecture
5. Workflow Process
6. Live Demo

Project Overview

We developed a system that analyzes customer sentiment. It converts voice recordings to text. It analyzes sentiment as positive, neutral, or negative. It generates audio summaries and visualizes sentiment trends. Our target users are business managers seeking insights.

Voice-to-Text

Converts voice recordings to text.

Sentiment Analysis

Analyzes sentiment.

Audio Summaries

Generates audio summaries.

Rationale for AWS Services

We chose AWS services for their accuracy and power. AWS Transcribe converts voice to text. AWS Comprehend analyzes sentiment with minimal customization. AWS Polly performs text-to-speech. AWS S3 offers reliable storage.

AWS Transcribe

AWS Comprehend

AWS Polly

Implementation Strategy

Backend

The backend uses AWS Chalice API for serverless computing.

- REST API implemented with AWS Chalice.
- AWS S3 stores customer feedback data.

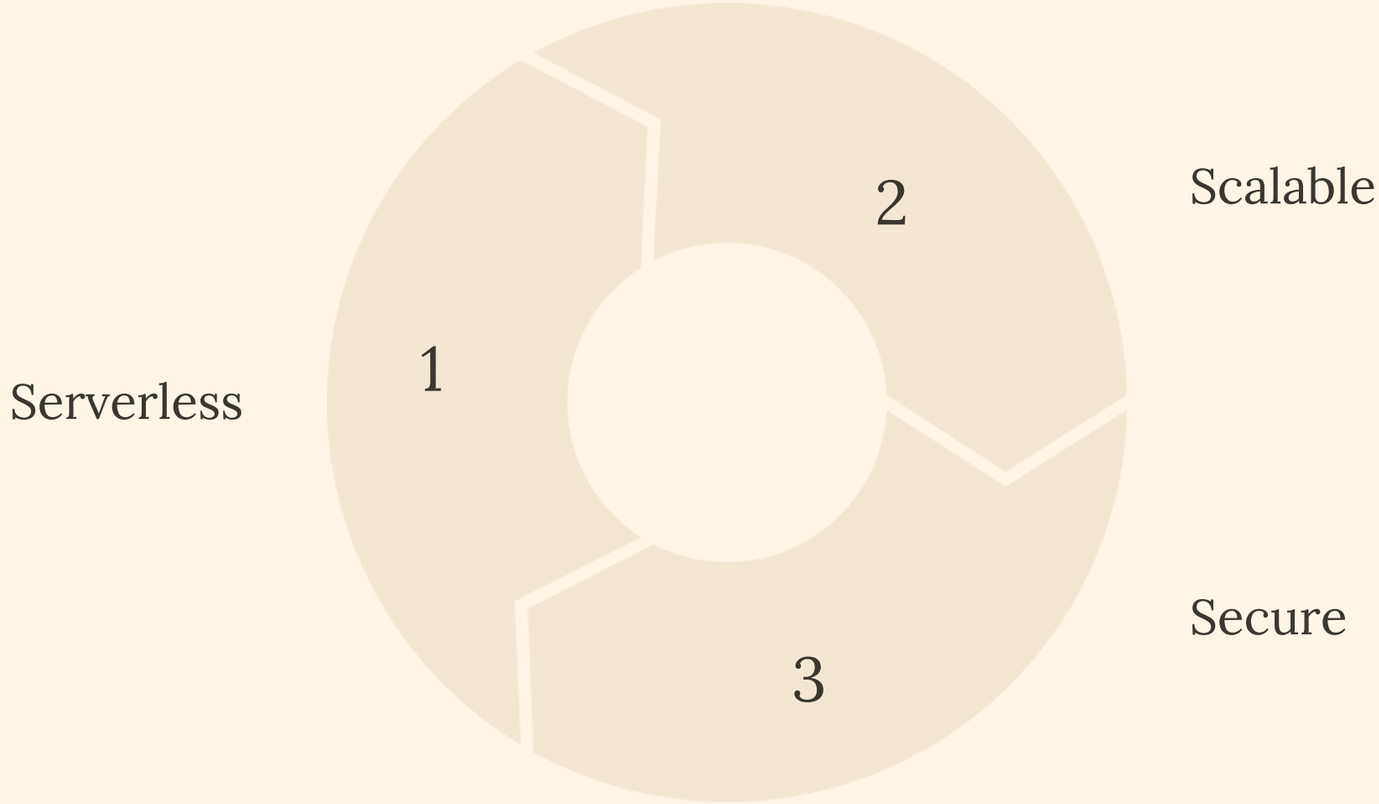
Frontend

The frontend is a web application with voice input capabilities.

- API integration for sentiment analysis and audio playback.
- Chart.js library for visualizing sentiment trends with graphs.

System Architecture

The system integrates AWS cloud services. It uses a serverless backend architecture. It features a scalable processing pipeline. It has secure data storage. The system provides real-time dashboard visualization.



Workflow Process

The workflow involves several steps for automated sentiment analysis.

Submit Feedback

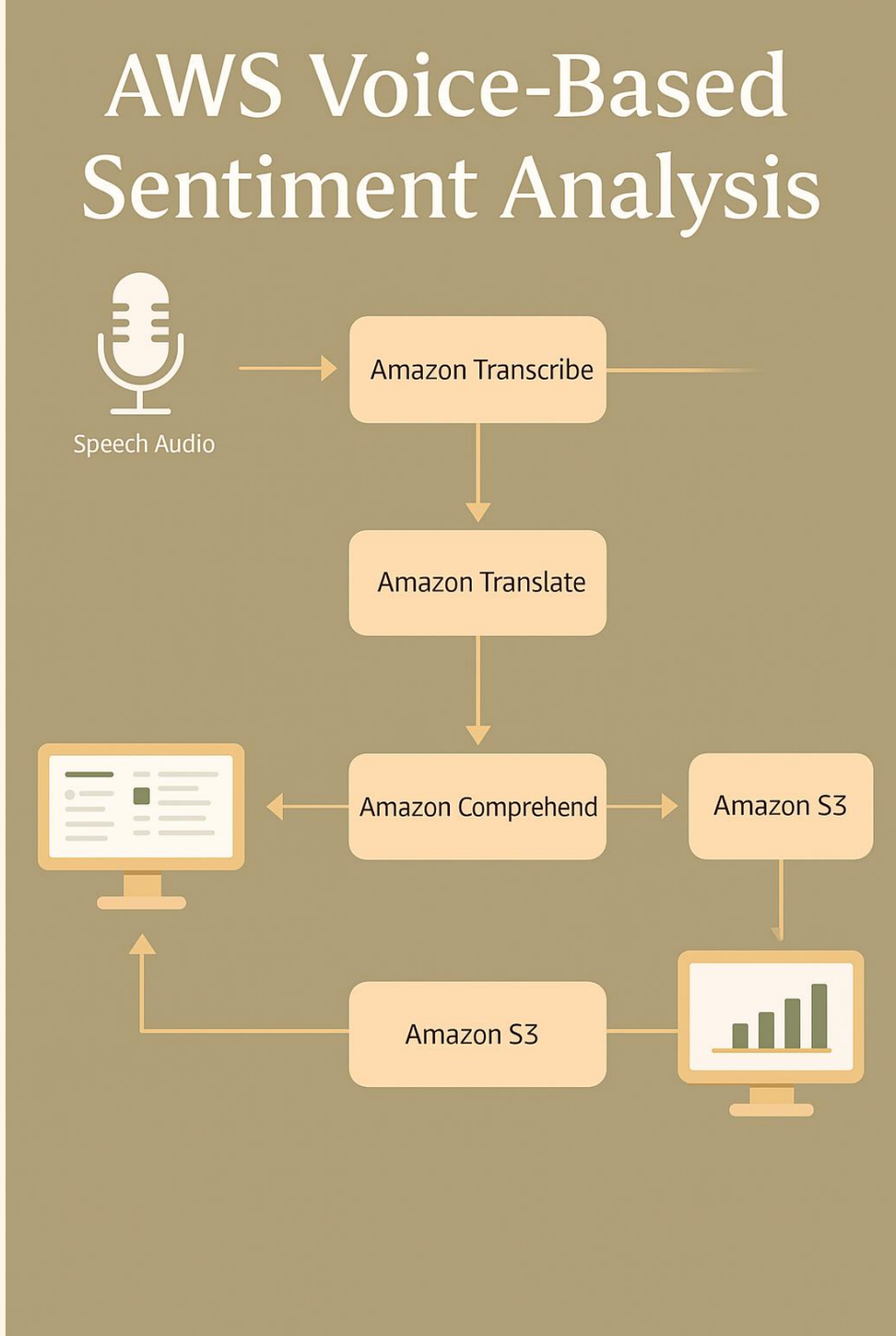
Customer submits voice feedback.

Store in S3

Audio stored in S3.

Transcribe

AWS Transcribe converts speech to text.



Live Demo

Voice Input

The interface allows users to submit voice recordings for analysis.

Sentiment Output

Sentiment is categorized as positive, neutral, or negative.

Visual Dashboard

Visualize sentiment trends with interactive graphs.

Audio Summary

Listen to generated audio summaries of the sentiment analysis.

Conclusion

Cloud-Native System

Successfully developed a cloud-native feedback analysis system.

AWS Leverage

Leveraged AWS for automation, scalability, and performance.

Ready for Scaling

Ready for deployment and scaling with real-world feedback.