Final Report

Project Aim

The goal for creating addiction is to design a web platform for language learners to improve their auditory comprehension. OpenAI's chatGPT API is used to generate stories of varying levels of english, which are then read back to the user, sentence by sentence, using another API. The user then writes into the web application to verify their auditory comprehension of the sentences read out to them. By the end of the short story, the user will receive a score indicating their errors. The end result is a language learning tool that will assist auditory comprehension

Project Description

This project is a web-based application built with Express.js and other technologies that provides users with the ability to interact with OpenAI's GPT-3 model. Users can select a writing level (e.g., beginner, intermediate, advanced) and request the generation of a short story. The application sends the request to OpenAI's API, receives the generated content, and displays it word by word in a user-friendly manner. The words are then converted to audio using AWS Polly, and read back to the user so that they may type it back into the web application.

Functional Requirements:

User Story Generation:

- Users can select a genre (e.g., adventure, horror, fiction) and difficulty level (easy, medium, hard, expert).
- Users can initiate story generation by clicking a "Generate Story" button.
- The application uses OpenAI GPT-3 to generate a concise story based on the user's selections.
- The generated story is converted to audio using Amazon Polly.

Audio Playback:

- Users can listen to the generated audio story.
- Playback controls, such as play, pause, and volume adjustment, are available to the user.

Evaluation of Input

- Application will evaluate inputted text to give the user a score
- Score may be interpreted in a way to assist with comprehension of english language

Genre and Difficulty Filtering:

- The application provides a filtering mechanism to select stories based on genre and difficulty.
- Users can browse existing stories by genre and difficulty.

Non-Functional Requirements:

Scalability:

- The application should be able to handle a growing number of users and story requests.
- Efficient resource management and scalability considerations should be implemented.

Performance:

- The response time for story generation and audio playback should be fast and responsive.
- The application should be able to handle concurrent audio requests without significant delays.

Security:

- User data, including API keys for OpenAI and Amazon Polly, should be securely stored and transmitted.
- Proper authentication and authorization mechanisms should be in place to prevent unauthorized access.

User Stories:

- **As a user, I want to be able to select a genre and difficulty level for the story so that I can get a personalized audio story experience.**
- **As a user, I want to initiate story generation by clicking a "Generate Story" button, and I want the generated story to be played back as audio.**
- **As a user, I want to browse and filter existing stories by genre and difficulty, so I can explore different stories matching my preferences.**
- **As a user, I want to improve my auditory comprehension of the english language**
- **As a user, I want to receive an indicator, or a score, that will function as an indicator to track progress as I get better at understanding what is being read back to me**

Individual Role and Responsibilities

Ira Hudgin

- Server structure
- OpenAI connection and fetching
- Initial prototyping for API connections and text generation
- EJS templating
- Initial script.js construction
- Deliverable participation
- CI/CD Tools
- Test Cases
- API Documentation

Florjan Blakaj

- Server structure
- OpenAI connection and fetching
- Amazon Polly Connection and fetching
- Initial prototyping for API connections and text generation
- EJS templating
- Initial script.js construction
- Deliverable participation
- CI/CD Tools
- Test Cases
- Hosting
- API Documentation

Petar

- Hosting
- EJS templating