POLITECNICO
MILANO 1863

TEACHme

Leveraging cutting edge AI/LLM technology to empower second language learners.

Andrea Federici (10621924)

Alireza Yahyanejad (10886993)

Mahdi Valadan (10903871)

Paolo Pertino (10729600)

Pietro Moroni (10625198)

July 2024

Table of Contents

Table of Figures

The Project

Motivation

Second language acquisition (SLA) is a complex process influenced by several factors, including motivation, learning environment, and practice opportunities. Traditional SLA methods often lack sufficient opportunities for interactive conversation practice, which is crucial for fluency development.

TEACHme aims to address these challenges by providing accessible and cost-effective conversational practice opportunities. The system will utilize Large Language Models (LLMs) embedded in embodied agents to facilitate language learning, specifically targeting Italian native speakers learning English.

Defining the requirements

The application would position itself in a face-to-face education environment, serving as an additional tool at the teacher's disposal to help their students practice their English conversation skills.

The scope of TEACHme doesn't involve a platform for teacher-student matchmaking, assuming that the rapport between the two parties is going to be established outside of the application.

This decision was made for several reasons, including simplicity and the desire to keep the system as flexible as possible to give more space to each teacher's didactic style and beliefs.

Stakeholders

We identified the stakeholders of the system as the two user categories – teachers and students – and the first step in designing the solution was to define a set of goals and requirements the application would need to satisfy to represent a valid and relevant prototype for them.

Value proposition

For students

The project's value proposition for the student was identified in providing them with a tool to practice conversation, while also being able to receive feedback on their mistakes or inaccuracies.

- Students should be able to practice and improve their conversational skills through feedback.
- Students should be able to enrich their vocabulary and improve their pronunciation through context-relevant insights.

- Students should be able to track and assess their progress.

For teachers

The goals for teachers were defined to ensure that the application would remain relevant in the context of an educational process that will also exist outside of the system.

- Teachers should be able to track their students' progress and visualize the conversations they are engaging with on the platform.
- Teachers should be able to customize the register, topic and level of difficulty of conversation agents to ensure they are tailored to the students needs.

Multidisciplinarity of the project

In the context of this project report, it's important to highlight the different disciplines and sets of skills that were implied in developing it.

Web development technologies and techniques

To design and develop the codebase behind TEACHme, a firm understanding of web development techniques and infrastructure was needed.

We needed to decide on what technology stack and architectural communication style we wanted to adopt and what consequences our choices would imply. Key decisions, which will be discussed more in detail later on in the document, included:

- Choosing between a single monolithic codebase or splitting the frontend and backend into standalone architectural components.
- Deciding whether to develop a backend application server or deploy the logic on serverless functions with a different set of dependencies.
- Designing the system in a modular way to avoid vendor lock-in with third-party services.

LLM technology and prompt engineering

Large language models (LLMs) are a recent emerging technology that stemmed from the machine learning field of natural language processing. In order to be able to harness this technology's full potential it's crucial to understand its inner workings. This understanding is also fundamental to developing effective *prompt engineering* strategies to combat the inherent black-box nature of the models.

A fundamental decision that will be discussed later revolved around whether the project should rely on an ad hoc, fine-tuned model or whether prompt engineering would be enough to achieve relevant results through the use of third-party models.

Second language education

It's easy for computer engineers who love writing code to forget the actual domain they are addressing and the real-world requirements such code will need to fulfill. In designing an application able to stay true to its value proposition and provide solid educational support to both teachers and students, we needed to understand where that value could be extracted from and how to best present it to the users.

This involved recognizing the importance of maintaining student focus during the conversations and providing valuable challenges and insights that are relevant to such activities. We needed to reframe the technical perspective into an education-oriented one and treasure the valuable insights provided by the tutors.

Building TEACHme

Designing the system architecture

To enable faster development times and incur less concurrent development-related conflicts, we decided to split the codebase into two architectural components: a front-end web server and a back-end application server. This approach also leveraged the team's skill sets, which ranged from web development to machine learning applications.

The architecture of the system would not be complete without a third tier dedicated to data storage, which in our case will reside on the MongoDB Atlas platform.

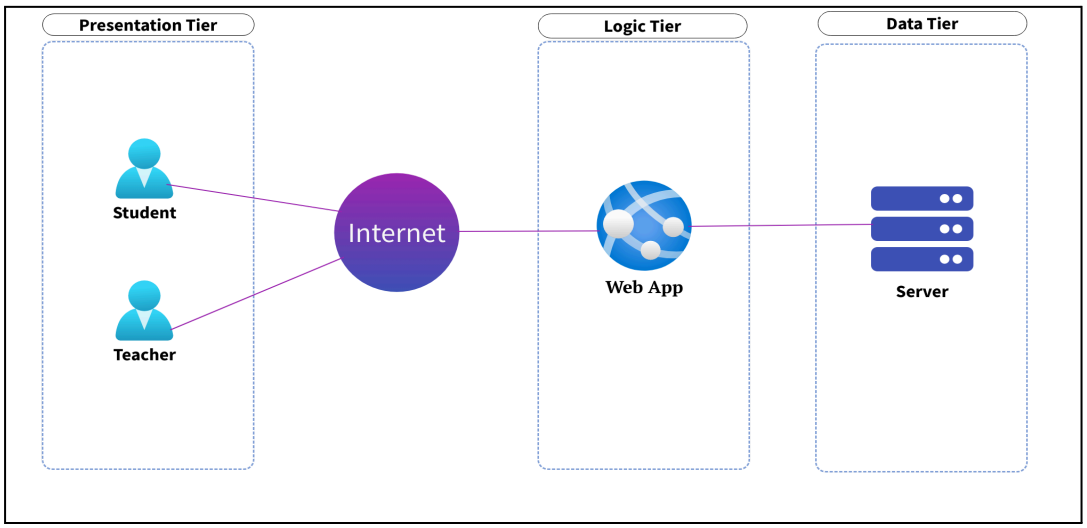

Figure 1. three tier architecture

Frontend web application

The frontend was developed using Javascript-based modern web technologies, specifically the Node.js environment and the React.js[13] framework Next.js.

Next.js

Next.js is a powerful opinionated framework that handles many of the traditional hurdles in web server development, and some of the features that make it an optimal choice in the context of developing a modern web app are:

- the easy to-use file-based routing system aimed at simplifying the process of setting up the web server aspect of the application;
- the support for Server Side Rendering (SSR) and Static Site Generation (SSG), which are crucial to provide a fast performance and responsive experience;
- the possibility to harness React to develop a component-based architecture for clean, modular code and reusable components.

Tailwind CSS

Tailwind CSS is a utility-first CSS framework designed to streamline the web design process by providing a comprehensive set of utility classes. These classes can be directly applied to HTML elements to achieve responsive and modern designs without writing custom CSS. Tailwind's utility-first approach promotes a consistent design system, making it easier to build and maintain complex UIs. Additionally, its flexibility and customizability allow developers to create unique designs while keeping the codebase clean and manageable.

Speech-to-text with Web Speech API

The Web Speech API is a browser-based service that provides real-time speech recognition functionalities. This enables TEACHme to seamlessly convert spoken language into text.

The Web Speech API offers several advantages, including:

- Continuous speech recognition: this allows the students to speak naturally without frequent interruptions, which helps in creating an immersive and more realistic conversation experience.
- Browser compatibility: as a browser-based service, it is compatible with most devices, making the platform more widely available.
- Reduced latency: being already integrated into the browser, it results in minimal delay compared to third-party speech-to-text services. With other services, we would need to send the entire audio file over the internet and then wait for a response after the server has fully processed the audio file.

Text-to-speech with Microsoft Azure API

Microsoft Azure's Text-to-Speech (TTS) API, part of Azure Cognitive Services, provides advanced capabilities for converting text into natural sounding speech. Here are some of its key features and benefits:

1. Natural and Human-like Voices:
    - Offers a wide range of voice options, including neural voices that sound more natural and human-like.

- Supports multiple languages and dialects, providing localized accents and pronunciations.

2. Customization:
    - Allows customization of voice models to suit specific needs, such as brand voices or unique pronunciations.
    - Users can adjust parameters like pitch, speed, and volume for more personalized output.
3. Versatility:
    - Integrates easily with various applications, including chatbots, virtual assistants, and interactive voice response (IVR) systems.
    - Supports a range of file formats for audio output, such as MP3, WAV, and OGG.
4. Accessibility:
    - Enhances accessibility for users with visual impairments or reading difficulties by providing an audio alternative to text.
    - Can be used in educational tools, reading apps, and more to improve learning and information consumption.
5. Real-time Synthesis:
    - Capable of generating speech in real-time, making it suitable for applications requiring immediate feedback.
    - Provides low latency and high performance, ensuring a seamless user experience.

Viseme-powered Lip Sync

A viseme represents the position of the face and mouth when saying a word. It is the visual equivalent of a phoneme, which is the basic acoustic unit from which a word is formed. Visemes are the basic visual building blocks of speech. Using visemes provided alongside the synthetic text-to-speech audio data by Microsoft's Azure Speech service[14] helps achieve natural mouth animations for the conversational agent, improving the visual appeal and realism of the experience. In conjunction with the Canvas API native to HTML, this also helped create high-performance lip syncing animations.

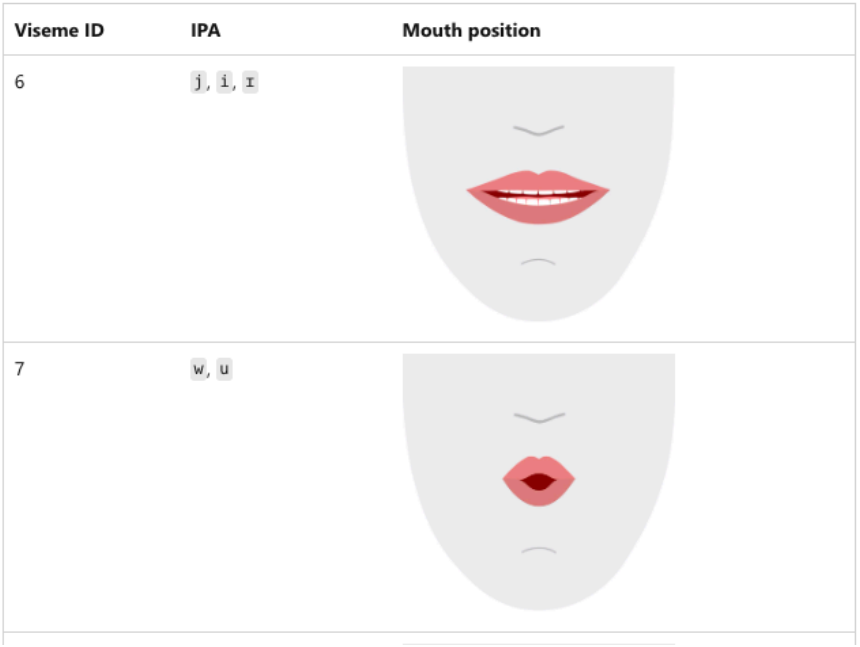

Figure 2. Example of International Phonetic Alphabet (IPA) symbols and their corresponding viseme [4].

Structure of the Application

The application features the following pages:

- A **Sign-up page** and a **Sign-in page** for users to create their accounts and log into the application, choosing a username and using their email and password.
- A **Teacher Dashboard** on which, upon logging into their account, teachers are presented with an overview of all conversations and their respective students. Each conversation is accompanied by its topic, difficulty, level, and the student's email addresses.
- A **Create New Content page**, exclusive to teachers, allows users to create new content by selecting the user level, difficulty, student, duration, and topic. Upon clicking the "Create" button, the new content is generated.
- A **Manage Students page**, where teachers can manage their students, including adding or removing them from their roster, within this section.
- A **Student Dashboard** within which all conversations and feedback are readily available. Upon logging into their account, students gain access to this section of the application, where they can view the topic, difficulty, and level associated with each conversation and feedback.
- A **Conversation page**, on which students can engage with topics previously created by their teacher. By clicking the "Start" button, students can begin conversing with the chatbot about the selected topic. They can also end the conversation at any time by clicking the "End Conversation" button.

- A **Feedback page**, within which students can review the details of their conversation, including overall feedback and specific details. The "Conversation Detail" section displays the topic, difficulty, user level, and teacher information. Students can access overall feedback and delve into conversation specifics such as content, feedback, synonyms, pronunciation, and AI responses.

Backend application server

The first step in building the backend of the application was deciding between coding a “proper” server, or using a serverless approach instead. Despite the convenience and scalability of the latter in many circumstances, we opted for the former: keeping the connection to the database alive – opposed to reconnecting to it on every request – and the ability to store conversation status in memory to avoid having to re-download the entire “chat” history for each ongoing conversation every time a message was sent and needed processing – on top of our early approach to speech recognition which involved streaming chunks of audio and “re-assembling” them in the backend – were pivotal decision points that made us pick the server approach.

We opted for the Flask[3] web application framework for Python because of its simplicity and modularity, the large community support, and available resources.

Third party services

Integrating third-party services was essential for equipping TEACHme with cutting-edge technologies for conversational AI and text-to-speech conversion. Here’s an in-depth look at the third-party services we integrated and how they contribute to TEACHme.

Conversational AI with OpenAI’s ChatGPT API

For generating intelligent and content-aware responses, we used OpenAI’s ChatGPT API[5][8]. This API utilizes advanced natural language processing capabilities to understand and create human-like responses based on the user’s input. This allows our embodied agents to engage in meaningful and relevant conversations with students.

Key benefits of using the ChatGPT API include:

- Dynamic conversations: the API produces diverse and contextually relevant responses, enhancing the engagement and educational value of conversations.
- Adaptive learning: by fine-tuning the prompts provided to the API, we can tailor the conversations to match the student’s proficiency level and learning objectives.

Harnessing the ChatGPT API also allowed us to direct our energy to designing the system and the experience rather than developing and training our own Large Language Model, which would have taken longer times, superior hardware and considerably larger funds than we would have been able to invest in the project.

Note that the entire chatbot logic has been constructed using the Langchain library. Langchain is a robust library specifically designed for building applications that rely on Large Language Models (LLMs). It serves as a generic interface for almost all LLMs, providing a centralized development environment for creating LLM applications and integrating them

with external data sources and software workflows. Langchain's modular approach allows comparing different prompts dynamically and various foundational models without needing to rewrite code. This flexibility and support for a variety of LLM providers enable us to develop a versatile application, avoiding dependency on a single service provider (e.g., OpenAI), and allowing easy interchangeability with other services by modifying just a few lines of code.

Text-to-speech with Microsoft Azure

To enable the virtual agent to speak to the student, we incorporated Microsoft Azure's text-to-speech API[1] into the application. This service converts the text generated by the ChatGPT API into natural-sounding speech.

Advantages of using Microsoft Azure's text-to-speech service:

- Natural sounding voices: the service offers a range of different voices, allowing us to select the most engaging and appropriate one for our users.
- Customization: the ability to customize the voice, intonation, and speaking style helps in making the response more human-like.
- Visemes for Lip Sync: on top of the synthetic audio data, the service also provides viseme information, making it easier to give life to our conversational agent via mouth shape animations.

Understanding the educational requirements

As already stated, understanding the functional requirements alone was not sufficient to ensure the project would be a meaningful prototype for what can be done with this technology in the education space. Therefore, we needed to sit down and take measures to align the technology we were using and the scope of the project itself.

The first thing we needed to take into consideration was the substantial difference between a text-based and a real-time chat: while the former usually unfolds in a succession of brief messages, that even belonging to the same train of thoughts might not be coherent from a syntactical point of view, the latter needs longer and more organic sentences to feel natural.

A text-based chat also often features noticeable delays between exchanges, while in a spoken conversation we want a continuous flow with as little latency as possible.

To fully exploit the educational potential of this tool, we need to ensure that – granted a slight uncanny valley feeling – the user can get immersed in the conversation and keep their momentum both while speaking to their virtual counterpart and when waiting for and listening to their response or receiving feedback.

Additionally, we needed to ensure that the tool aids users in expressing themselves and enhancing their conversational skills. Although the system accepts quick, closed-ended questions and answers, it tries to discourage this approach. Instead, the chatbot aims to maintain the conversation active by incorporating both closed and open-ended questions related to the topic of discussion.

It is also worth noticing that LLMs often provide users with complete and lengthy responses. While this can be useful in many scenarios, it is important to control this tendency during conversations. A balance of fairly long explanations, brief responses, and quick follow-up questions is essential for a natural interaction, rather than only detailed answers. We have not fully achieved this balance yet, indicating that there is room for improvement in the system.

Finally, another challenging part of the design of TEACHme is to provide proper feedback, or to make sure that feedback given serves the purpose of helping a user to learn English properly. In essence, effective feedback should be constructive and memorable; the user should have every reason to actively use the suggestions to improve language skills rather than simply acknowledge them and then forget. For example, feedback need not be just pointing out some grammatical errors but should explain the rule of the correction with a few illustrative examples. You should use 'went' instead of 'gone' in this sentence" is not as useful as saying, "Remember, 'went' is the past tense of 'go,' while 'gone' is the past participle. So you say, 'I went to the store yesterday,' not 'I went to the store yesterday.' Keep practicing it so it sticks."

Besides, feedback should be relevant to the user level and learning pace, neither overwhelming nor too simplistic. TEACHme wants feedback to be part of the learning process itself by focusing on actionable, clear, and contextually relevant advice.

Design and development challenges

Making the conversation “real time”

As already discussed, reducing latency in the conversational exchange was crucial and played a significant role in the final design of our infrastructure. For example, imagine if the user spoke for 30 seconds and then had to wait for their audio to be uploaded to the server, transcribed, processed, and then for the audio file of the response to be rendered and downloaded. This lengthy experience would probably serve not in helping them practice their conversational skills but rather in testing their patience (still a useful skill in the real world, but not in the scope of the project).

To tackle this challenge, we explored several different options that we’ll go through one by one.

Our first approach

Our first approach involved uploading the entire audio recording to be processed by the Google Cloud speech-to-text service. While this method was very straightforward, it resulted in significant and unpredictable latency. Since the entire file was being uploaded all at once, the size of this operation could reach tens of megabytes at a time. This resulted in a delay that ranged from 10 to 20 seconds, counting from the moment the user stopped speaking to the moment the agent started to reply. This lag hindered too much the real-time conversational experience.

Streaming the audio to reduce upload times

To address the upload time issue, we transitioned to streaming audio chunks rather than uploading the entire audio file at once. This was achieved by using Flask’s WebSocket technology. By sending smaller segments of audio data continuously, we aimed to reduce the time it took to transfer the audio to the STT service. This approach helped in reducing the delay, but it was still not enough for the conversation to feel human-like, as we still hovered around 5-15 seconds of waiting time.

Real time speech to text to minimize latency

Although we managed to cut down on the upload time a lot, we realized that the bottleneck of the process was the upload of the audio file to Google’s Speech Recognition server. We therefore searched for alternative ways to convert the speech data into text.

We found a solution in the Web Speech Recognition API, a browser-based service that provides real-time speech recognition capabilities. Since this approach lies within the browser, it works locally and there is no need to upload the audio file to external servers. In this way, we effectively eliminated all delays associated with uploading, server processing, and downloading. This resulted in a much smoother, real-time conversational experience for TEACHme users.

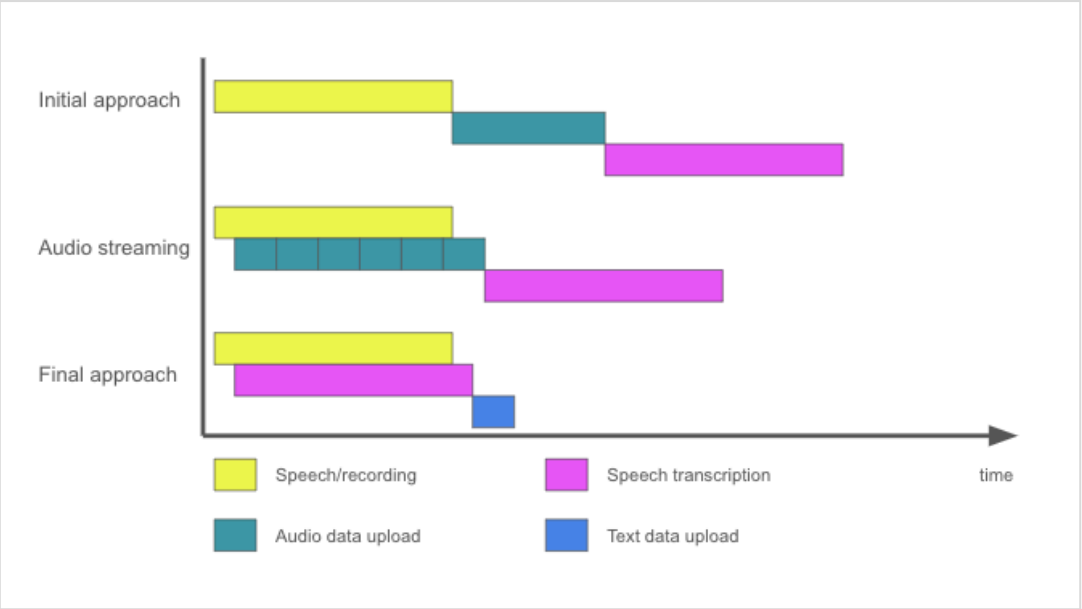

Figure 3. The graph shows the time approximately taken by different operations in the speech-to-text pipeline of the application and the improvements made by adopting the browser's SpeechRecognition API.

A non-intrusive feedback experience

As an educational tool, the application needed to provide feedback to the user, highlighting any syntactic mistakes or erroneous use of words but also providing challenges suggestions such as testing the users pronunciation of certain words and potential synonyms for the words they used. The modality behind how this feedback is presented to the user is critical in guaranteeing an immersive experience throughout the activity.

In absence of a computational model to mathematically score the performance of the conversation and a natural language processing model to extract the most relevant words to find synonyms or address the pronunciation of, we opted to harness prompt engineering to use our LLM of choice (i.e. ChatGPT) to generate the feedback for us, instructing it on what exactly was needed, how to format the output and providing relevant context such as the conversation transcript and other information about the activity.

Live feedback box and callouts

The initial plan for the conversation feedback was to display a "live" feedback box next to the main chat view with the conversation agent's avatar. This way we would be able to stream feedback to the user for every "message" (every exchange) they sent in real-time.

Looking more into it we realized though that this approach clashed with the very nature of the interaction. Exposing the student to feedback in the form of additional text to read during the conversation would result in a distraction damaging to the value of the experience.

This realization made us temporarily consider showing the user non-intrusive feedback reaction icons following their messages to symbolize whether they had made no mistakes if their message was a little confusing, or whether they had made a lot of mistakes: these icons would later be accompanied by more comprehensive feedback and explanations after

the activity was completed, so that users could go back and understand what exactly went wrong at any point during the conversation.

Providing feedback *ex-post*

Eventually, we realized that providing any form of visual feedback would be hindering the experience in general and we adopted a more empathetic approach: even just showing the player a negative icon could make them feel self-aware and perform worse during what remains of the activity, and therefore lower their self-confidence.

We landed on the final approach of the feedback system: for every message the user sent, after the agent's response, we now process it and generate feedback in the backend, storing everything in the database. After the conversation is over the user is then able to access a new feedback page where, for every activity they took part in, we show the feedback received for every message, with fun challenges (i.e. pronunciation and synonyms) and an overall recap of their performance.

Content moderation

Content moderation is a crucial component of TEACHme. Interactions within the platform must be safe and conducive to learning. The integration of AI-driven conversational agents requires careful management to ensure that the conversations remain appropriate even if the student tries to change the conversation or discuss content outside the scope of the application. To achieve this, TEACHme employs a combination of robust guidelines and advanced prompt engineering strategies when leveraging the ChatGPT API.

Prompt for ChatGPT API

The ChatGPT API is prompted with very specific instructions that are meant to guide its interactions with the users. These instructions are designed to align with the educational goals of TEACHme, while preventing conversations from heading into inappropriate or sensitive territories. Here is the moderation prompt (also known and referenced as system prompt) used:

"You are a conversation partner helping users practice and improve their English conversational skills. Your goal is to engage users in conversations to enhance their listening and speaking abilities and boost their confidence in using the language.

The user level is {user_level} and the conversation difficulty is {conversation_difficulty}. Tailor your responses to match the specified conversation difficulty in the following ways:

- Vocabulary:
    - If the conversation difficulty is set to 'easy', use simple, common words. Do not use rarely known words and context-specific jargon.
    - If the conversation difficulty is set to 'medium', use moderately complex words and introduce some commonly used phrases and idioms.
    - If the conversation difficulty is set to 'challenging', use advanced vocabulary, including less common words and context-specific jargon.

- Grammar and Syntax:
    - If the conversation difficulty is set to 'easy', use straightforward sentence structures (e.g., simple and compound sentences). Do not use complex sentences, hard idiomatic expressions, and syntactic constructions.
    - If the conversation difficulty is set to 'medium', use a mix of simple, compound, and some complex sentences, with appropriate use of conjunctions and transitional phrases.
    - If the conversation difficulty is set to 'challenging', use more complex sentence structures, such as compound-complex sentences, and incorporate varied grammatical constructions and advanced punctuation.
- Engagement:
    - If the conversation difficulty is set to 'easy', focus on maintaining a clear and concise conversation.
    - If the conversation difficulty is set to 'medium', engage with more detailed explanations and occasional follow-up questions to encourage deeper conversation.
    - If the conversation difficulty is set to 'challenging', stimulate critical thinking with probing questions, detailed explanations, and nuanced discussions.

Ensure your responses are always contextually appropriate and help the user progress in their understanding and use of English.

The conversation topic is {conversation_topic}.

Do not allow the user to completely change the topic of the conversation, and always steer the conversation back to the original topic, that is {conversation_topic}.

You have to respond in an engaging, informative, concise, and appropriate manner.

Maintain a relevant conversation but allow for natural digressions.

Encourage the user to continue the conversation.

Be very concise and natural in your responses, as if you are discussing with a friend.

Mixup open-ended questions with closed-ended questions.

Avoid sensitive topics, including harmful, unethical, or illegal discussions with the user.

If the user starts talking about negative feelings or private issues you must avoid providing advice or any kind of follow-up questions. You must not talk or listen to these topics. Just say that you are there to help the user practice their English skills.”

Key aspects of content moderation

- Topic adherence and steering: the AI is instructed to keep the conversation centered around the original topic, allowing for natural digressions but preventing the user from

changing the subject entirely. This ensures that the conversation remains adherent to the learning objectives proposed by the teacher.

- Engagement and appropriateness: the AI responds in an engaging, informative, concise, and appropriate manner, tailored to the user's English level and the specified conversation difficulty. This ensures that the learning experience is both challenging and supportive.

- Handling sensitive topics: the conversational agent must avoid any discussion on sensitive topics, including harmful, unethical, or illegal subjects. If the user attempts to talk about any of these topics, the AI is instructed to steer the conversation back to practicing English skills without engaging in or acknowledging the proposed sensitive subjects.

- Emotional and private issues: when the user expresses negative feelings or private issues, the AI does not provide advice or follow-up questions. Instead, it redirects the conversation to the primary subject, maintaining a supportive yet neutral stance.

It is important to note that, in the context of content moderation, we regard teachers as trusted users of our system. This means we assume they will not inject any harmful or illegal content as topics for discussion.

Constitutional chain

A constitutional chain[2] is a chain of safety checks applied to the output a model would give back to the user. This output is checked against user-defined principles (so-called constitutional principles that all together form the constitutional chain) and if they are not respected, the output is modified in order to comply with them. Using the constitutional chains is good but there are a few challenges:

- They increase the latency of the response since the model answers are checked against all the registered principles and every time they are adjusted to satisfy them (checking the text against a principle means scanning the text using a GPT with the principle embedded in a particular prompt).

- The principles must be carefully crafted. For example, if the principle is "Avoid illegal, harmful or unethical discussions with the user." and the human is discussing about his favorite films, if he mentions war/mafia films, the initial answer of the bot is the correct one (e.g. "I agree, the Godfather is a true masterpiece that delves into the Corleone family, one of the most brutal mafia family." that per se is not bad, but since it contains mafia related content it gets adjusted.

  Another example is:

  [human] "What is good to see in Rome in your opinion?

  [bot-initial-answer] "The ruins of the Colosseum, the place where gladiators were used to fight to the death."

  [bot-corrected] "The beautiful architecture of the Colosseum, which is a symbol of Rome's rich history.").

Ultimately, due to the significant increase in response latency intrinsic to the approach, we decided to abandon this strategy and focus more on building a stronger system prompt.

Market Analysis

The language learning market is vast and diverse, encompassing various tools and platforms designed to meet the needs of learners worldwide. TEACHme enters this landscape with a unique value proposition that addresses specific challenges in second language acquisition, particularly for Italian native speakers learning English.

Traditional methods of language learning are increasingly being replaced by digital solutions that offer greater flexibility, interactivity, and often lower costs.

TEACHme's Unique value proposition

One of TEACHme's greatest strengths is the integration of AI-driven conversational agents within a teacher-guided framework. This hybrid approach combines the availability and scalability of AI with the personalized guidance and expertise of human teachers.

By leveraging AI technology, TEACHme can provide learners with immediate, on-demand practice opportunities, ensuring that students can engage in conversational practice anytime and anywhere.

The AI-driven agents are capable of generating varied and contextually appropriate responses, making interactions dynamic and engaging, mimicking real-life scenarios.

Furthermore, the agents can adapt to the student's progress, providing a customized learning experience tailored to each student's needs.

All of this is combined with the crucial role played by the teachers. Teachers can monitor the student's progress and direct the learning experience to address individual challenges and goals. Teachers can assign tasks, set conversation topics, and adjust difficulty levels to match each student's pace and objectives.

This synergy between AI efficiency and human expertise sets TEACHme apart, making it a powerful tool for effective, personalized language learning education.

Similar emerging tools

Recently, tools and applications similar to TEACHme started appearing on the market. AI-powered language learning platforms that allow users to practice their speaking skills. The defining difference between these applications and TEACHme is that they target solo-learning individuals, aiding them in a self-taught journey rather than supporting a teacher-student relationship.

Loora

Loora[6] is a language learning platform similar to TEACHme. It leverages AI to facilitate conversational practice and language acquisition. Differently from TEACHme, Loora is

mainly text-based, with support for user input through audio: in fact it presents the user with a “traditional looking” text chat interface that focuses less on visual engagement and immersivity. Also, Loora provides a robust platform for solo learners to practice and improve their language skills independently, allowing a more flexible and self-paced study, while TEACHme is designed to be used in conjunction with a teacher and therefore provides a distinct edge for those seeking a more guided and personalized learning experience.

Praktika

Praktika[12] is a language learning app that combines features present both in Loora and TEACHme, harnessing visual AI conversation agents to allow a self-guided practice of one’s English speaking skills. Differently from Loora, Praktika really emphasizes its focus on engagement and immersivity, and it uses video-game engine technology to present fully animated 3D avatars for the user to interact with.

ChatGPT

“GPT-4o is a step towards much more natural human-computer interaction [...]. It can respond to audio inputs in as little as 232 milliseconds, with an average of 320 milliseconds, which is similar to human response time in a conversation. It matches GPT-4 Turbo performance on text in English and code, with significant improvement on text in non-English languages”.

We can’t talk about the market in AI-powered language learning applications without mentioning the platform that is enabling most of them. On top of providing the APIs that are being employed in building this new generation of interactive and educational chatbots, the latest release of the ChatGPT app, featuring the latest version of OpenAI’s Large Language Model, has also been speculated to potentially provide language learning and practicing capabilities. In the demo where GPT-4o was presented, the hosts were featured interacting with the virtual assistant in multiple languages, prompting it to translate their conversation in real-time.

Testing TEACHme

We eventually ran a small pilot study where we could observe real people interacting with the system in order to assess the usability of the application and potential areas for improvement. We mostly ran the study on our friends and family, but we were able to involve people with different levels of proficiency in English.

Structure of the test

For every person we tested TEACHme with, we had a precise routine and set of questions we would ask them. After collecting demographic data on them (age, sex, occupation and proficiency with technological systems like our app), the test plan was structured like this:

- Let the user register and log in to the application;

- Assign them three conversations from a chosen pool of topics and difficulties, and have them perform the activities;
- After every conversation have the user check out the ex-post feedback and ask them if they thought the feedback was relevant in identifying their shortcomings and whether the synonym and pronunciation challenges seemed accurate and interesting;
- After the three conversations, we asked the users to rate them in terms of what they thought the difficulty was for each of them;
- We also asked them for general feedback on the application, such as how easy it was to navigate, if they encountered any bugs, whether the conversation agent felt natural and convincing enough, and if they had any suggestions for improvement.

The conversations had to be 5 minutes long, and the topics we chose were “hobbies”, “favorite food” and “last summer holiday” because we deemed them accessible and relatable enough to work with anyone. The difficulty associated with each topic/conversation was randomized for every user and they were served following no specific order to avoid creating biases.

Results of the test

The overall feedback for the TEACHme application was highly positive, particularly regarding the user interface, design, and interactive robot, which users found enjoyable. The ChatGPT conversation agents demonstrated excellent adaptability and confidence, maintaining a balanced conversation flow through effective prompt engineering.

However, some areas for improvement were identified. One was with the speech-to-text system, which occasionally cut off users mid-sentence due to detecting pauses that were too long. Although it is easily fixable by introducing an artificial timer, rather than relying on the browser's built-in features. Nonetheless, the LLM was good enough at maintaining the conversation by using context from previous messages.

Another issue arose with the handling of Italian words, such as food or location names. Since the browser's internal speech detection was set to English, it sometimes missed Italian spellings. Despite this, ChatGPT was often able to correct these errors (e.g., identifying "pasta cacio e pepe" from "pasta catcher pepe"), with some errors only noticed during user feedback revision.

The questions asked by the conversation agent were most of the time relevant and open enough to guarantee a natural flow of conversation, even though we noticed that when the user was not as open to elaborating the topic or their answers further they could get somewhat repetitive; another thing that was noticed was the prevalence of American culture in the model’s training data (e.g. bringing up cheddar and mac-and-cheese as food topics) which is not a problem per se, but could contribute to a decrease in relatability for conversations: one user suggested we let the students contribute optional background and cultural information to their profiles, so that it could be used in the system prompt for steering the responses of the conversation agent towards potentially more relatable experiences for the user.

Some users also suggest improvements for the feedback page such as more specific and detailed overall feedback and having the option of the robot verbally delivering the overall feedback instead of reading it themselves.

Contributions

Front-end

Yahyanejad and Valadan worked on the design, development, and documentation of the frontend of the system supporting the TEACHme application. The frontend team's collaborative efforts were marked by regular weekly meetings, rigorous testing procedures, and a shared commitment to equitable task distribution. These practices were instrumental in fostering effective communication, ensuring quality assurance, and maintaining a balanced workload throughout the project.

Here is a detailed overview of the contributions of each single member:

- **Valadan:** setting up the frontend project environment, structuring and configuring the frontend project, managing routing and navigation, creating pages and components, implementing styling, handling state management, integrating data fetching APIs, integrating text-to-speech API, integrating speech-to-text API, optimizing performance, testing, design document, project report, and pilot study.
- **Yahyanejad:** Convert Script to Audio with Azure Text-to-Speech, Generate Viseme Data, Render Mouth Animations with Canvas, Style for avatar, Synchronizing Audio and Animation, Enhancing Realism, Optimization, project report, pilot study analysis, Testing and Iteration, Design Document, integrating data fetching APIs.

Back-end

Federici, Moroni, and Pertino worked on the design, development, and documentation of the backend system supporting the TEACHme application.
The team maintained cohesive collaboration throughout the entire project, ensuring an equal distribution of the workload necessary for the development of the application.

Here is a detailed overview of the contributions of each single member:

- **Federici**: database integration, routing, code documentation, errors logging, testing, project report, pilot study summary, backend API documentation
- **Moroni**: infrastructure, routing, components integration, experiments on technologies for optimizing the speech-to-text pipeline, testing, database integration, challenges, project report
- **Pertino**: database integration, chatbot, prompts, code documentation, API documentation, docker, testing, content moderation, Github documentation, and Wiki

References

[1] Microsoft Azure, *Azure SpeechSDK*. (2024). [Online]. Disponibile su: https://learn.microsoft.com/en-us/azure/ai-services/speech-service/speech-sdk

[2] Y. Bai *et al.*, «Constitutional AI: Harmlessness from AI Feedback», 15 dicembre 2022, *arXiv*: arXiv:2212.08073. Consultato: 17 luglio 2024. [Online]. Disponibile su: http://arxiv.org/abs/2212.08073

[3] Pallets Projects, *Flask*. (2024). [Online]. Disponibile su: https://flask.palletsprojects.com/en/3.0.x/

[4] Microsoft Azure, «Get Facial Positions with Visemes». 2024. [Online]. Disponibile su: https://learn.microsoft.com/en-us/azure/ai-services/speech-service/how-to-speech-synthesis-viseme?tabs=visemeid&pivots=programming-language-csharp

[5] OpenAI, *GPT-4o*. (2024). [Online]. Disponibile su: https://openai.com/index/hello-gpt-4o/

[6] Loora ltd, *Loora*. (2024). [Online]. Disponibile su: https://www.loora.ai/

[7] Vercel, *NextJS*. (2024). [Online]. Disponibile su: https://nextjs.org/

[8] OpenAI, *OpenAI*. (2024). [Online]. Disponibile su: https://openai.com/

[9] DesignCode, «Original mockup». [Online]. Disponibile su: https://www.figma.com/community/file/1116248614926294639/web-app-ui-design

[10] A. Hall, «Perspective Taking in Language Learning and Teaching», *Forum on Public Policy Online*, vol. 2006, fasc. 1, p. 1, 2006.

[11] A. Wheelock, «Phonological Difficulties Encountered by Italian Learners of English: An Error Analysis», *Hawaii Pacific University TESOL Working Paper Series*, vol. 14, pp. 41–61, 2016.

[12] Praktika.ai Company, *Praktika*. (2024). [Online]. Disponibile su: https://praktika.ai/

[13] Meta, *ReactJS*. (2024). [Online]. Disponibile su: https://react.dev/

[14] Amazon Web Services, «Visemes Documentation». 2024. [Online]. Disponibile su: https://docs.aws.amazon.com/polly/latest/dg/viseme.html

Annexes

Pilot Study Report — TEACHme

Andrea Federici (andrea3.federici@mail.polimi.it)
Alireza Yahyanejad (alireza.yahyanejad@mail.polimi.it)
Mahdi Valadan (mohammadmahdi.valadan@mail.polimi.it)
Paolo Pertino (paolo.pertino@mail.polimi.it)
Pietro Moroni (pietroguglielmo.moroni@mail.polimi.it)

July 2024

1 Introduction

The purpose of this pilot study was to evaluate the TEACHme application in terms of usability, conversational feedback relevance, and overall user experience. The study aimed to identify strengths and areas for improvement in order to refine the application.

2 Methodology

2.1 Participants

The pilot study included four participants selected to represent a diverse range of demographics and technical proficiencies. The participants varied in age and included both male and female genders. Occupations among the participants included a bank intern, students, and a physical education teacher in a primary school. Technical proficiency levels ranged from moderately proficient to very proficient, ensuring a broad spectrum of user experiences with the TEACHme application. A detailed description of each participant, including specific characteristics, will be illustrated in a later section.

2.2 Procedure

Participants signed up, logged in, and were assigned to three different conversation scenarios. Feedback was collected after each conversation, focusing on conversation feedback relevance, pronunciation and synonym suggestions, difficulty level, and general usability.

Each participant had the following conversations with the related difficulty levels:

- **Participant 1:**
  - Conversation 1: Last summer holidays (Difficulty: easy)
  - Conversation 2: Hobbies (Difficulty: medium)
  - Conversation 3: Favorite food (Difficulty: hard)
- **Participant 2:**
  - Conversation 1: Last summer holidays (Difficulty: hard)
  - Conversation 2: Hobbies (Difficulty: easy)
  - Conversation 3: Favorite food (Difficulty: medium)
- **Participant 3:**
  - Conversation 1: Last summer holidays (Difficulty: hard)
  - Conversation 2: Hobbies (Difficulty: easy)
  - Conversation 3: Favorite food (Difficulty: medium)
- **Participant 4:**
  - Conversation 1: Last summer holidays (Difficulty: medium)
  - Conversation 2: Hobbies (Difficulty: hard)
  - Conversation 3: Favorite food (Difficulty: easy)
- **Participant 5:**
  - Conversation 1: Last summer holidays (Difficulty: hard)
  - Conversation 2: Hobbies (Difficulty: medium)
  - Conversation 3: Favorite food (Difficulty: easy)

The duration was set to 5 minutes for each conversation.

3 Data Collection

3.1 Demographic Data

The participants' demographic information was collected to provide context for their feedback.

3.2 Usability Feedback

Participants were asked to rate the ease of navigation, report any issues or bugs, and provide overall usability feedback.

3.3 Conversational Feedback

Participants' responses were collected on the relevance of the conversation feedback and the accuracy of pronunciation/synonym suggestions.

3.4 Difficulty Levels

Participants were asked to assign difficulty levels to each conversation (easy, medium, hard).

3.5 Naturalness of Conversations

Participants provided feedback on whether conversations felt natural and human-like.

4 Results

4.1 Participant Demographics

- **Participant 1:**
  - Age: 23 years
  - Gender: Female
  - Occupation: Intern in a bank in Switzerland
  - Technical Proficiency: 3
- **Participant 2:**
  - Age: 25 years
  - Gender: Male
  - Occupation: Student
  - Technical Proficiency: 3/4
- **Participant 3:**
  - Age: 23 years
  - Gender: Male
  - Occupation: Student
  - Technical Proficiency: 4
- **Participant 4:**
  - Age: 25 years
  - Gender: Male

- Occupation: Student and physical education teacher in a primary school
- Technical Proficiency: 3/4

- **Participant 5:**
  - Age: 27 years
  - Gender: Female
  - Occupation: Student
  - Technical Proficiency: 4

4.2 Usability Feedback Analysis

- **Participant 1:**
  - **Ease of navigation:** Rated 3-4. The process to sign up, log in, and navigate the home page was clear. However, there were issues understanding some buttons on the conversation page. Adding instructions could improve clarity.
  - **Issues reported:** The participant experienced interruptions when thinking about what to say. Sometimes, the overall feedback was not immediately visible on the feedback page and required reloading.

- **Participant 2:**
  - **Ease of navigation:** Rated 4. Fairly easy to navigate, with no major issues.
  - **Issues reported:** The microphone would occasionally cut off, urging the participant to think quickly. Pronunciation of Italian location names was problematic.

- **Participant 3:**
  - **Ease of navigation:** Rated 5. Easy to navigate, but some buttons were in unintuitive places.
  - **Issues reported:** The participant experienced significant microphone issues, with frequent interruptions.

- **Participant 4:**
  - **Ease of navigation:** Rated 4-5. The application was easy to navigate but there was some confusion about how to use the buttons on the conversation screen.
  - **Issues reported:** While thinking, the application would sometimes send the message before the participant finished speaking.

- **Participant 5:**

- **Ease of navigation:** Rated 5.
- **Issues reported:** In general, everything was good, and I think that since the program is newly implemented, there may be some mistakes, but this program did not have too many problems.

4.3 Conversational Feedback Analysis

- **Participant 1:**
  - **Relevance of feedback:** The feedback was informative and relevant, helping the participant discover new ways to say things.
  - **Pronunciation and synonym suggestions:** Synonym suggestions were accurate and helpful. Pronunciation challenges were sometimes triggered by non-English words, reducing their informativeness.
- **Participant 2:**
  - **Relevance of feedback:** The feedback was somewhat generic but useful overall.
  - **Pronunciation and synonym suggestions:** Some suggestions were accurate and helpful, while others were less so.
- **Participant 3:**
  - **Relevance of feedback:** Feedback was skewed by microphone issues, leading to frustration.
  - **Pronunciation and synonym suggestions:** Generally accurate, with some suggestions more interesting than others.
- **Participant 4:**
  - **Relevance of feedback:** The feedback was relevant and useful for improving English speaking skills.
  - **Pronunciation and synonym suggestions:** Synonym suggestions were accurate, but some were difficult to understand and it was not clear how to use them in a sentence.
- **Participant 5:**
  - **Relevance of feedback:** The feedback was good. But in my opinion, overall feedback is too much and can be written in a more concise way.
  - **Pronunciation and synonym suggestions:** Overall it was good. Sometimes it didn't show anything about Pronunciation.

4.4 Difficulty Level Analysis

- **Participant 1:**
  - Easy conversation: Rated as Easy
  - Medium conversation: Rated as Hard
  - Hard conversation: Rated as Medium
- **Participant 2:**
  - Easy conversation: Rated as Medium
  - Medium conversation: Rated as Easy
  - Hard conversation: Rated as Hard
- **Participant 3:**
  - Easy conversation: Rated as Easy
  - Medium conversation: Rated as Medium
  - Hard conversation: Rated as Hard
- **Participant 4:**
  - Easy conversation: Rated as Easy
  - Medium conversation: Rated as Hard
  - Hard conversation: Rated as Medium
- **Participant 5:**
  - Easy conversation: Rated as Easy
  - Medium conversation: Rated as Medium
  - Hard conversation: Rated as Medium

4.5 Naturalness and Human-Like Quality

- **Participant 1:**
  - The conversation was engaging and stimulated continuous discussion. However, the participant noted that the agent did not naturally integrate personal experiences or feelings into the conversation unless explicitly prompted.
- **Participant 2:**
  - The conversation felt natural overall.
- **Participant 3:**

  - The conversation was mostly human-like, but safety principles were too strict, and some answers felt empty.

- **Participant 4:**
  - The conversation felt quite natural. However, the participant sometimes preferred shorter replies from the agent.

- **Participant 5:**
  - Sometimes I expected a shorter answer to the question I asked and vice versa.

4.6 Additional Comments and Suggestions for Improvement

- **Participant 1:**
  - Overall, the participant liked the application and suggested adding functionalities for self-study, improving UI clarity, and making the agent's responses more natural.

- **Participant 2:**
  - Fix bugs and improve the interface.

- **Participant 3:**
  - Fix bugs, improve the interface, and make the conversation feel more natural by addressing the strictness of safety principles.

- **Participant 4:**
  - Improve the time the agent waits while the user is thinking and consider allowing longer thinking times.

- **Participant 5:**
  - Improve the overall feedback and pronunciations.

5 Discussion

5.1 Key Findings

- Positive feedback on usability and conversational relevance.
- In Overall users like the design and the robot.

5.2 Strengths and Weaknesses

- **Strengths:** Ease of use, engaging conversations, accurate synonym suggestions.
- **Weaknesses:** UI clarity issues, issues in pronunciations, occasional bugs, and the need for more natural conversational integration.

5.3 Participant Feedback

The participants suggested improvements in UI clarity, pronunciations, more natural conversation integration, and functionalities for self-study.

6 Conclusion

6.1 Summary of Findings

The overall feedback for the TEACHme application was highly positive, particularly regarding the user interface, design, and interactive robot, which users found enjoyable. The ChatGPT conversation agents demonstrated excellent adaptability and confidence, maintaining a balanced conversation flow through effective prompt engineering.

6.2 Recommendations

- Improve UI clarity by adding instructions or on-hover descriptions.
- Address minor bugs, such as interruptions during conversation and delayed feedback visibility.
- Enhance the naturalness of conversations by integrating personal thoughts and experiences into the agent's responses.
- Add functionalities for self-study to allow students to practice independently.
- Adjust the time the agent waits while the user thinks to allow for longer thinking times.
- improvements for the feedback page such as more specific and detailed overall feedback and having the option of the robot verbally delivering the overall feedback instead of reading it themselves.

7 Appendices

7.1 Appendix A: Participant Responses

7.1.1 Participant 1

- Age: 23
- Gender: Female
- Occupation: Intern in a bank in Switzerland
- Technical proficiency: 3 (Moderately Proficient)
- Was the conversation feedback relevant in identifying your shortcomings?: Informative and relevant.
- Were pronunciation/synonym suggestions accurate?: Synonym suggestions are accurate; pronunciation challenges are sometimes triggered by non-English words.
- How easy was it to navigate the application (1 - Very difficult, 5 - Very easy): 3-4
- Did you encounter any issues or bugs?: Interruptions when thinking, delayed feedback visibility.
- Did the conversation feel natural and human-like to you?: Engaging but lacked integration of personal experiences.
- Any additional comments or suggestions for improvement?: Add functionalities for self-study, improve UI clarity, and enhance conversation naturalness.
- Difficulty matching (ground truth — predicted by the user):
  - Last summer holidays: Easy — Easy
  - Hobbies: Medium — Hard
  - Favorite food: Hard — Medium

7.1.2 Participant 2

- Age: 25
- Gender: Male
- Occupation: Student
- Technical proficiency: 3/4 (Moderately to Very Proficient)
- Was the conversation feedback relevant in identifying your shortcomings?: Somewhat generic but useful overall.

- Were pronunciation/synonym suggestions accurate?: Mixed accuracy, generally aware of suggestions.
- How easy was it to navigate the application (1 - Very difficult, 5 - Very easy): 4
- Did you encounter any issues or bugs?: Microphone issues, pronunciation problems with Italian names.
- Did the conversation feel natural and human-like to you?: Felt natural overall.
- Any additional comments or suggestions for improvement?: Fix bugs, and improve the interface.
- Difficulty matching (ground truth — predicted by the user):
  - Last summer holidays: Hard — Hard
  - Hobbies: Easy — Medium
  - Favorite food: Medium — Easy

7.1.3 Participant 3

- Age: 23
- Gender: Male
- Occupation: Student
- Technical proficiency: 4 (Very Proficient)
- Was the conversation feedback relevant in identifying your shortcomings?: Feedback skewed by microphone issues.
- Were pronunciation/synonym suggestions accurate?: Generally accurate, some more interesting than others.
- How easy was it to navigate the application (1 - Very difficult, 5 - Very easy): 5
- Did you encounter any issues or bugs?: Significant microphone issues, and frequent interruptions.
- Did the conversation feel natural and human-like to you?: Mostly human-like but with strict safety principles and some empty responses.
- Any additional comments or suggestions for improvement?: Fix bugs, improve the interface, and make conversations more natural.
- Difficulty matching (ground truth — predicted by the user):
  - Last summer holidays: Hard — Hard
  - Hobbies: Easy — Easy
  - Favorite food: Medium — Medium

7.1.4 Participant 4

- Age: 25
- Gender: Male
- Occupation: Student and physical education teacher in a primary school
- Technical proficiency: 3/4 (Moderately to Very Proficient)
- Was the conversation feedback relevant in identifying your shortcomings?: Relevant and useful for improving English speaking.
- Were pronunciation/synonym suggestions accurate?: Synonym suggestions were accurate, but some were difficult to understand and use in a sentence.
- How easy was it to navigate the application (1 - Very difficult, 5 - Very easy): 4-5
- Did you encounter any issues or bugs?: While thinking, the application would sometimes send the message before the participant finished speaking.
- Did the conversation feel natural and human-like to you?: The conversation felt quite natural. However, the participant sometimes preferred shorter replies from the agent.
- Any additional comments or suggestions for improvement?: Improve the time the agent waits while the user is thinking and consider allowing longer thinking times.
- Difficulty matching (Conversation scenario: ground truth — predicted by the user):
  - Favorite food: Easy — Easy
  - Last summer holidays: Medium — Hard
  - Hobbies: Hard — Medium

7.1.5 Participant 5

- Age: 27
- Gender: Female
- Occupation: Student
- Technical proficiency: 4
- Were pronunciation/synonym suggestions accurate?: The application is nice to use. Avatar can help with learning.

- How easy was it to navigate the application (1 - Very difficult, 5 - Very easy): 5
- Did you encounter any issues or bugs?: In general, everything was good, and the participant thinks that since the program is newly implemented there could have been some issues, but no breaking bugs were found.
- Did the conversation feel natural and human-like to you? Sometimes the participant expected a shorter answer or vice versa.
- Any additional comments or suggestions for improvement? Improve the overall feedback and pronunciation challenge.
- Difficulty matching (Conversation scenario: ground truth — predicted by the user):
  - Favorite food: Easy — Easy
  - Last summer holidays: Hard — Medium
  - Hobbies: Medium — Medium

Front-end Design Document

1. Introduction

1.1 Project Overview

Second language acquisition poses several challenges due to cost and accessibility barriers associated with conversational practice, such as hiring native English speakers for practicing speaking skills. This project aims to leverage Large Language Models (LLMs) embedded in embodied agents to enhance conversational practice for Italian speakers learning English.

1.2 Objectives

- Develop a scalable and cost-effective solution for conversational practice in second language acquisition.
- Integrate LLMs into embodied agents to provide realistic and interactive language practice sessions.
- Create a user-friendly application interface using Next.js for seamless user experience.

1.3 Scope

The scope of this project includes:

- Development of a front-end application using Next.js.
- Integration of LLMs to facilitate conversation practice.

2. Choosing Technologies

Each technology was chosen for its unique strengths and capabilities that contribute to the overall success of the project.

2.1 Next.js as Frontend Framework

Rationale: Next.js is a React-based framework known for its component-based architecture and ease of use, which facilitates the development of complex user interfaces. It supports Server-Side Rendering (SSR), improving performance and SEO, which is beneficial for web applications that require fast load times and high visibility. Additionally, Next.js supports Static Site Generation (SSG), enabling the creation of static pages that can be pre-rendered for even better performance and scalability.

Benefits: Improved performance and SEO are achieved through SSR. Development is eased with React's component-based approach. Scalability and flexibility are ensured with support for both SSR and SSG.

2.2 ChatGPT for script creation

Rationale: ChatGPT, powered by OpenAI's GPT-4 architecture, excels in generating human-like text based on input prompts, making it ideal for creating coherent and contextually appropriate scripts. It offers flexibility, as it can be fine-tuned and adapted to various styles and tones, making it versatile for different types of content. Additionally, automating script generation reduces the time and effort required to produce high-quality scripts, enabling rapid content creation and iteration.

Benefits: The use of ChatGPT results in high-quality, contextually relevant scripts. It provides customizable outputs to suit project-specific needs. Furthermore, it offers a scalable solution for generating large volumes of content.

2.3 Web Speech API for generating audio

Rationale: The Web Speech API enables seamless integration of speech recognition and synthesis capabilities into web applications, enhancing user interaction through natural, hands-free communication. It supports multiple languages and dialects, allowing for personalized and localized user experiences. The API's event handling and customization features provide developers with precise control over speech processes, ensuring reliable and responsive application behavior.

Benefits: Using the Web Speech API enhances user accessibility and engagement by providing voice-driven interaction. It is cost-effective and easy to implement, requiring no additional hardware or software. The API supports real-time transcription and interactive voice response systems, making it suitable for various applications, from customer service to accessibility tools. Additionally, it integrates smoothly with existing web technologies, offering flexibility and ease of use for developers.

2.4 Azure Text-to-Speech for audio synthesis

Rationale: Azure Text-to-Speech provides natural-sounding, high-fidelity speech synthesis, enhancing the overall quality of the audio output. The service offers various voices and languages, allowing for tailored audio that matches the desired character and tone of the content. Additionally, Azure's robust API makes it easy to integrate with other components of the project, streamlining the workflow.

Benefits: The use of Azure Text-to-Speech results in professional and realistic audio output. It offers a wide range of voice options and languages. Furthermore, it enables seamless integration with existing systems and services.

2.5 Viseme for phoneme frame data generation

Rationale: Viseme provides detailed phoneme frame data, which is crucial for synchronizing mouth movements with spoken audio, ensuring accurate lip-sync. Utilizing viseme data helps achieve more precise and natural mouth animations, improving the visual appeal and realism of the animations. Additionally, automating the generation of phoneme frame data reduces manual animation efforts and increases consistency across different content pieces.

Benefits: Using viseme data enhances lip-sync accuracy. It improves animation quality and realism. Furthermore, it saves time by automating the generation of animation data.

2.5 Canvas for rendering mouth animations

Rationale: The HTML5 Canvas API is highly versatile and can be used for a wide range of graphics and animations, including complex mouth movements. Canvas provides efficient rendering capabilities, ensuring smooth animations even for detailed mouth movements. Additionally, as a standard web technology, Canvas is widely supported across different browsers and platforms, ensuring broad accessibility.

Benefits: The use of the HTML5 Canvas API results in high-performance, smooth animations. It offers broad compatibility and support. Furthermore, it provides flexibility for creating detailed and dynamic animations.

2.6 Tailwind CSS

Rationale: Tailwind CSS is a highly customizable, low-level CSS framework that provides a wide array of utility classes to build responsive and visually appealing web designs directly in your markup. Unlike traditional CSS frameworks, Tailwind CSS promotes a utility-first approach, which allows developers to rapidly prototype and build consistent, scalable, and maintainable UI components. The framework's modular and functional design enables fine-grained control over styling, ensuring that design specifications are met precisely without the need for extensive custom CSS.

Benefits: Using Tailwind CSS streamlines the development process by reducing the time spent on writing and maintaining custom styles. It offers a consistent design system through utility classes, which leads to more uniform and predictable layouts. The framework is highly responsive, making it easy to build mobile-first designs that adapt seamlessly to different screen sizes. Tailwind CSS also promotes better collaboration between developers and designers by providing a shared language of utilities. Furthermore, its integration capabilities with modern JavaScript frameworks and build tools enhance productivity and project scalability.

3. Detailed Design

3.1 Application Pages

3.1.1 Student Dashboard

Within the student dashboard, all conversations and feedback are readily available. Upon logging into their account, students gain access to this section of the application, where they can view the topic, difficulty, and level associated with each conversation and feedback.

3.1.2 Teacher Dashboard

Upon logging into their account, teachers are presented with an overview of all conversations and their respective students. Each conversation is accompanied by its topic, difficulty, level, and the students' email addresses.

3.1.3 Sign-up Page

Both students and teachers can easily create their accounts through the sign-up page by providing their name, email, and password.

3.1.4 Login Page

Users can log in to their accounts by entering their credentials. Incorrect username or password entries will prevent access to the account.

3.1.5 Creating New Content

Exclusive to teachers, this section allows them to create new content by selecting the user level, difficulty, student, duration, and topic. Upon clicking the "Create" button, the new content is generated.

3.1.6 Manage Students

Teachers can manage their students, including adding or removing them from their roster, within this section.

3.1.7 Conversation Page

In the conversation page, students can engage with topics previously created by their teacher. By clicking the "Start" button, students can begin conversing with the chatbot about the selected topic. They can also end the conversation at any time by clicking the "End Conversation" button.

3.1.8 Feedback Page

Within this section, students can review the details of their conversation, including overall feedback and specific details. The "Conversation Detail" section displays the topic, difficulty, user level, and teacher information. Students can access overall feedback and delve into conversation specifics such as content, feedback, synonyms, pronunciation, and AI responses.

3.2 LLM Integration

3.2.1 Conversation Flow

The system initiates with an initial greeting and topic selection tailored to user preferences. It ensures conversational coherence through context-aware responses from the LLM. Additionally, it dynamically adjusts conversation complexity based on the user's proficiency level.

3.2.2 Error Handling

The system handles API errors gracefully, providing appropriate user feedback. It implements fallback mechanisms to address failed LLM responses, ensuring a seamless user experience even in the face of unexpected errors.

4. Implementation Plan

4.1 Development Phases

- Phase 1: Initial setup and basic front-end development
- Phase 2: Back-end development and API integration
- Phase 3: LLM integration and testing
- Phase 4: User interface refinement and final testing
- Phase 5: Deployment and user feedback collection

5. User Interface Design

5.1 Mockup

5.1.1 Student Panel

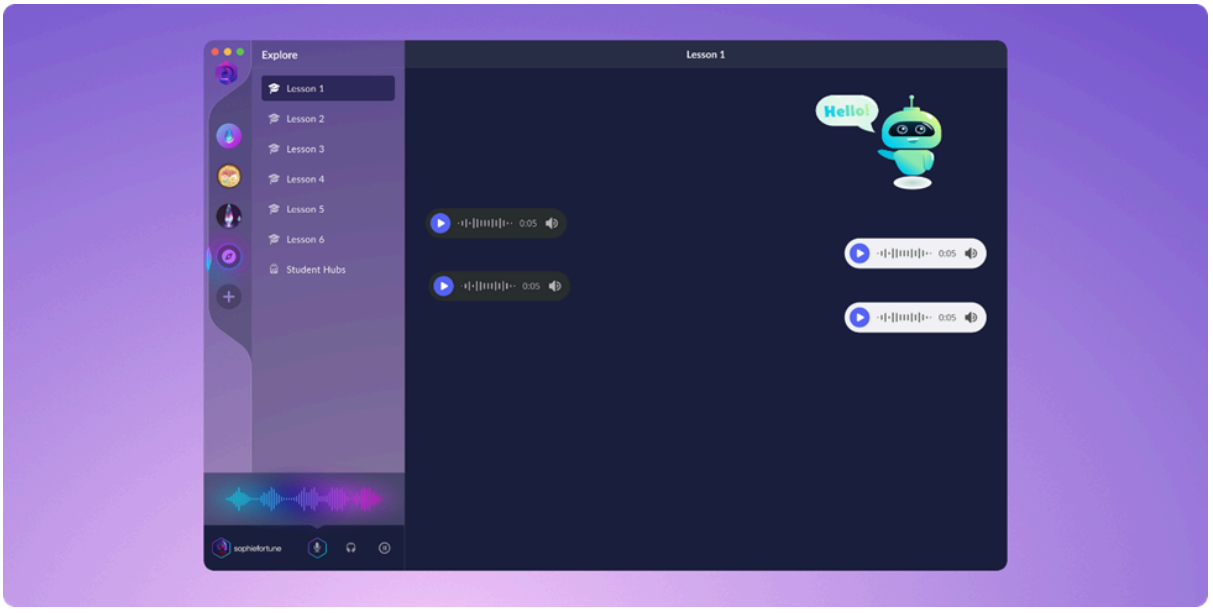

Figure 4. Mockup - Student Panel

5.1.2 Teacher Panel

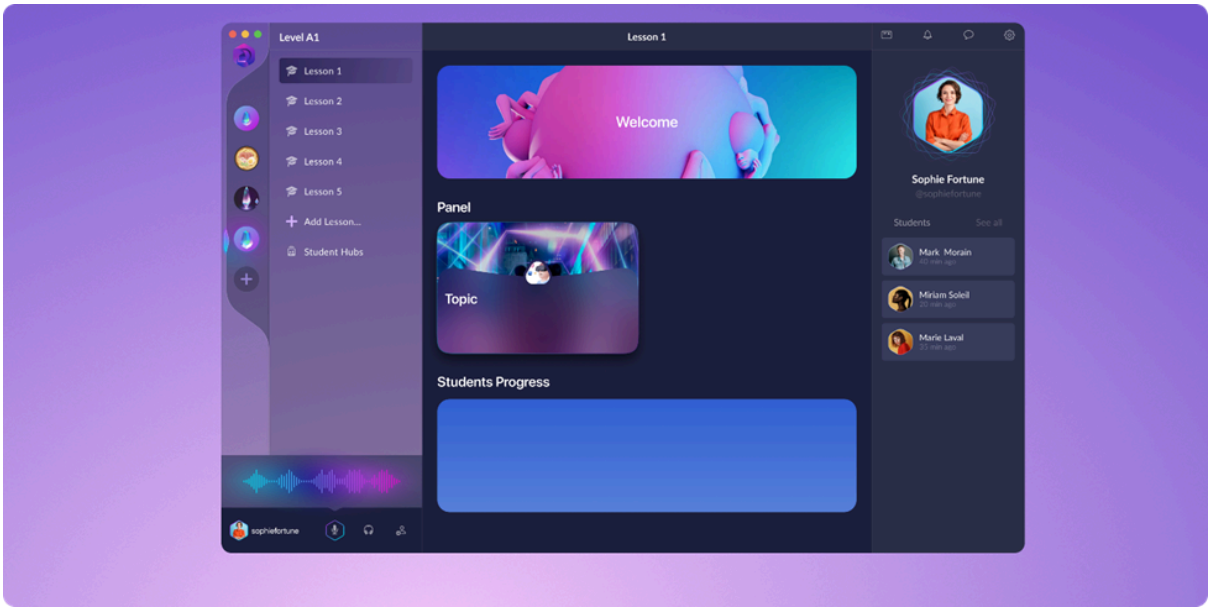

Figure 5. Mockup - Teacher Panel

5.1.3 Teacher Panel – Add Student

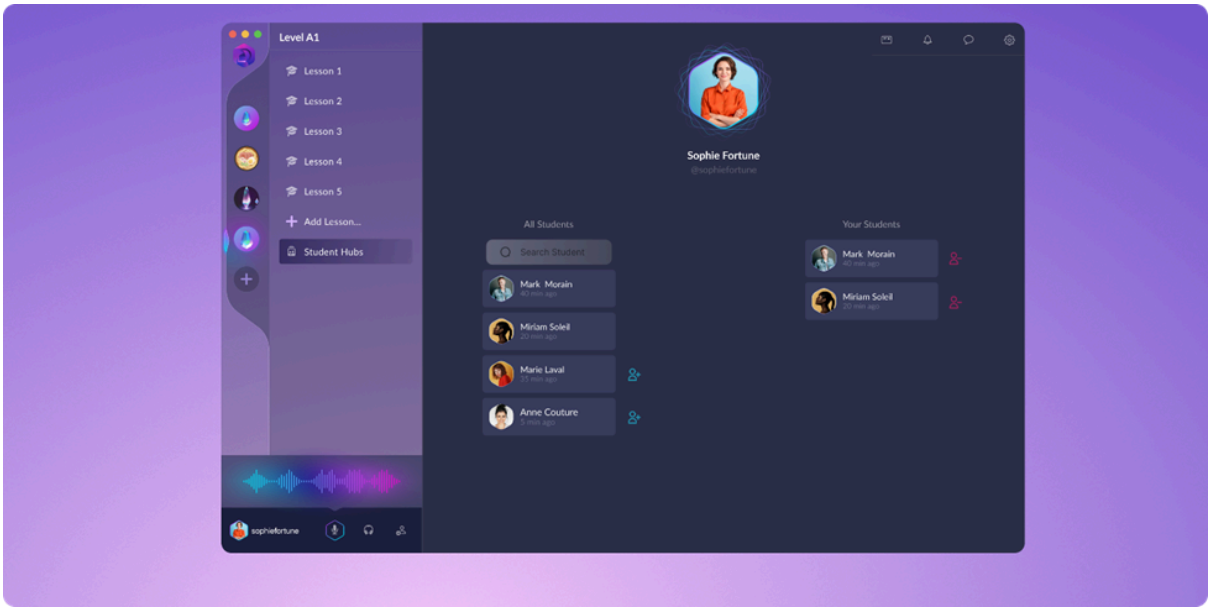

Figure 6. Mockup - Teacher Panel - Add Student

5.2 Real Application

5.2.1 Student Dashboard

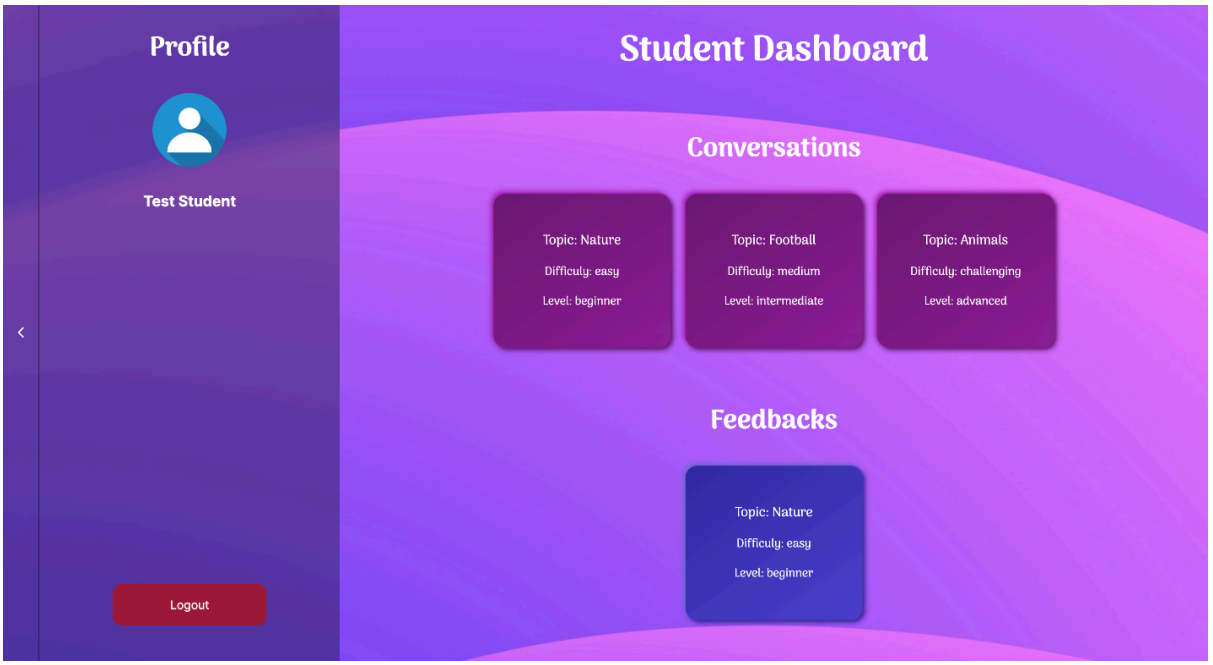

Figure 7. Student Dashboard

5.2.2 Teacher Dashboard

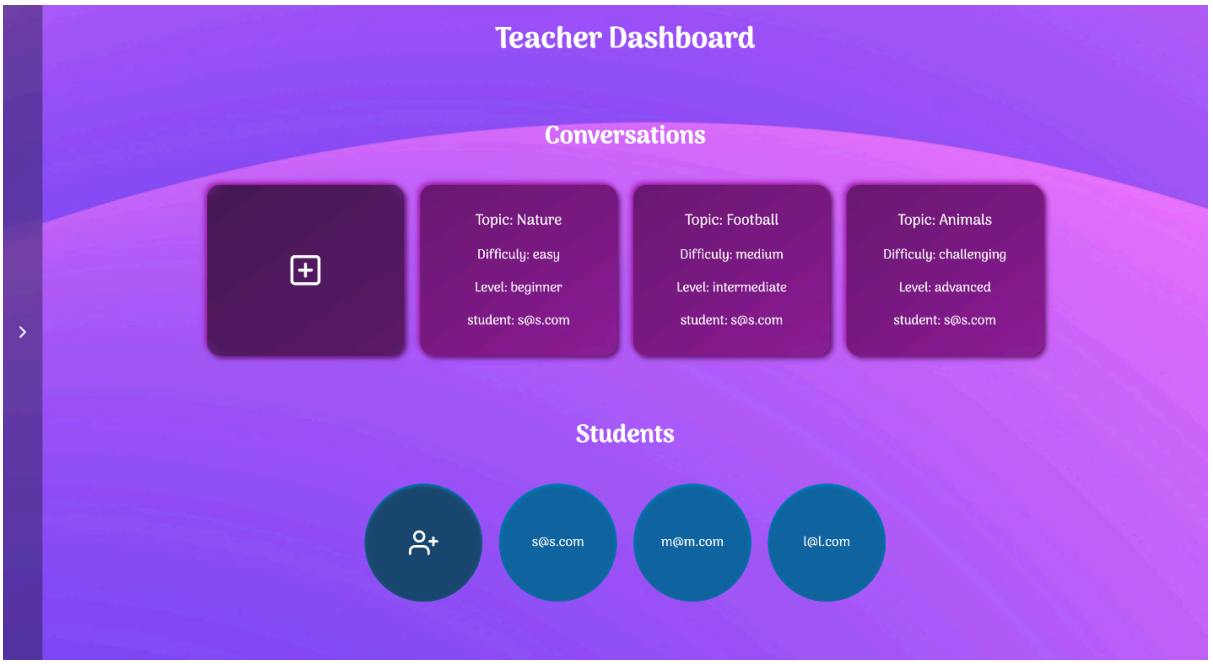

Figure 8. Teacher Dashboard - part 1

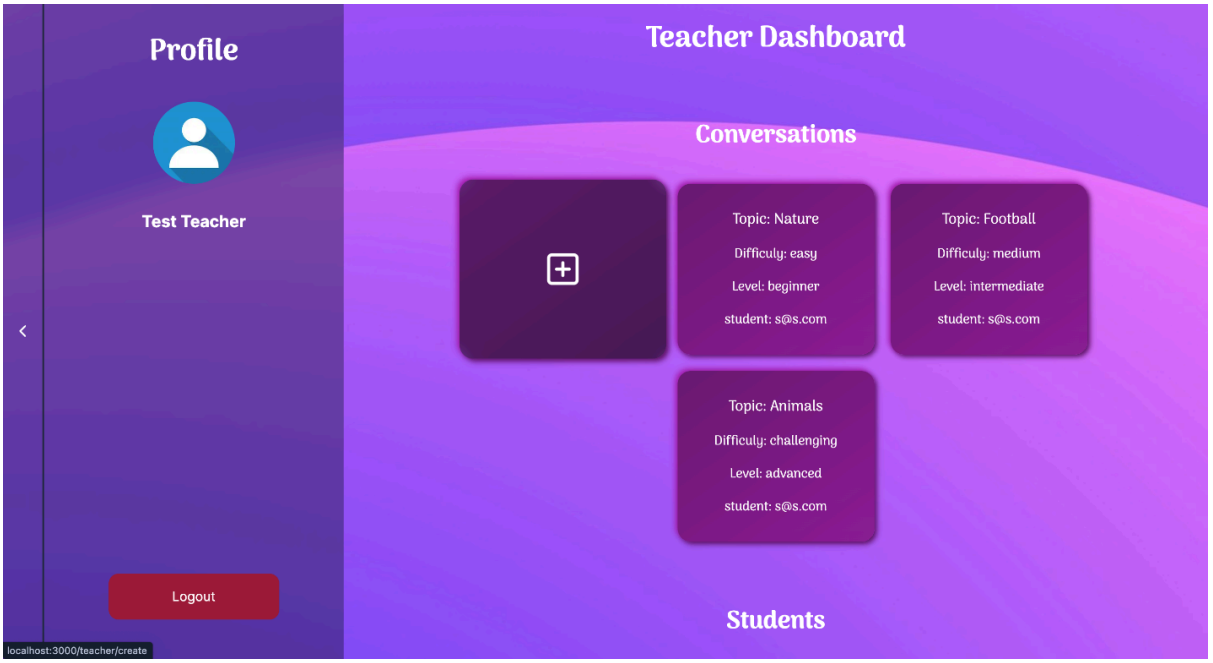

Figure 9. Teacher Dashboard - part 2

5.2.3 Sign-up Page

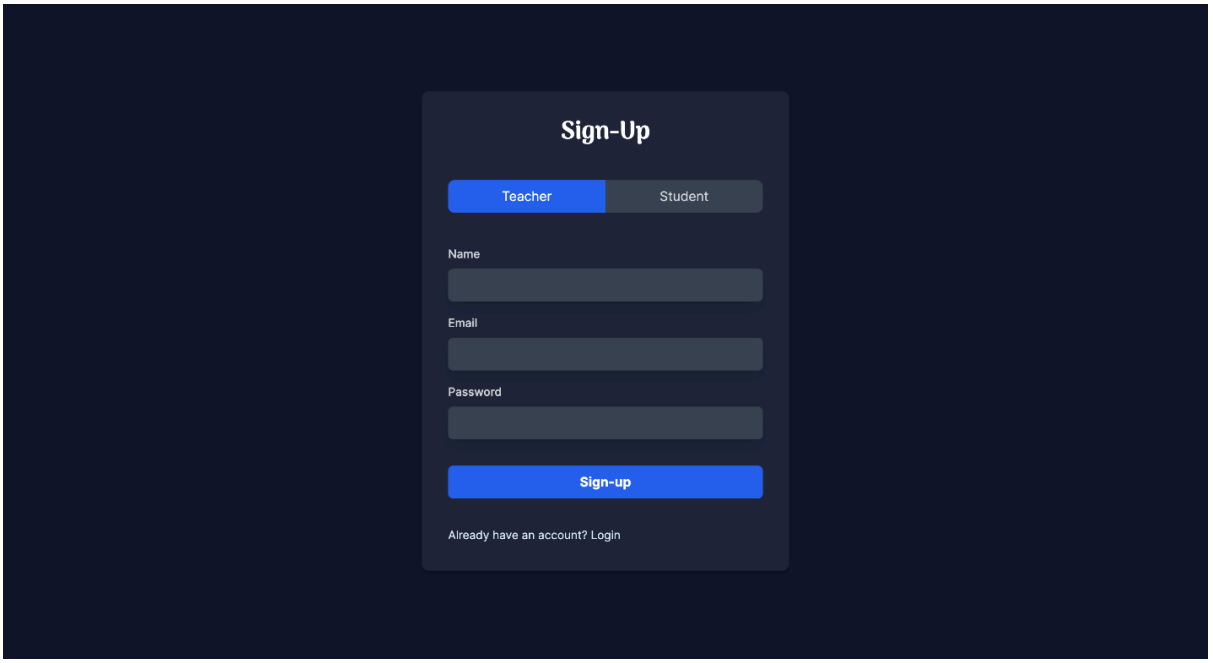

Figure 10. Sign-up Page

5.2.4 Login Page

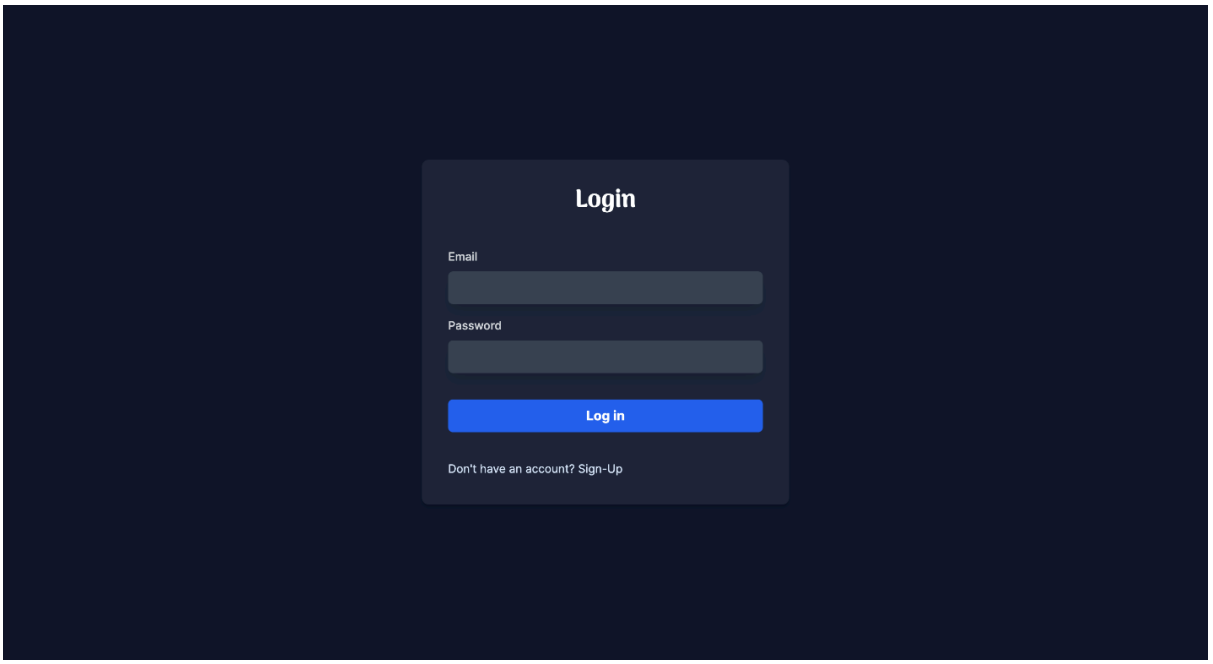

Figure 11. Login Page

5.2.5 Creating New Content

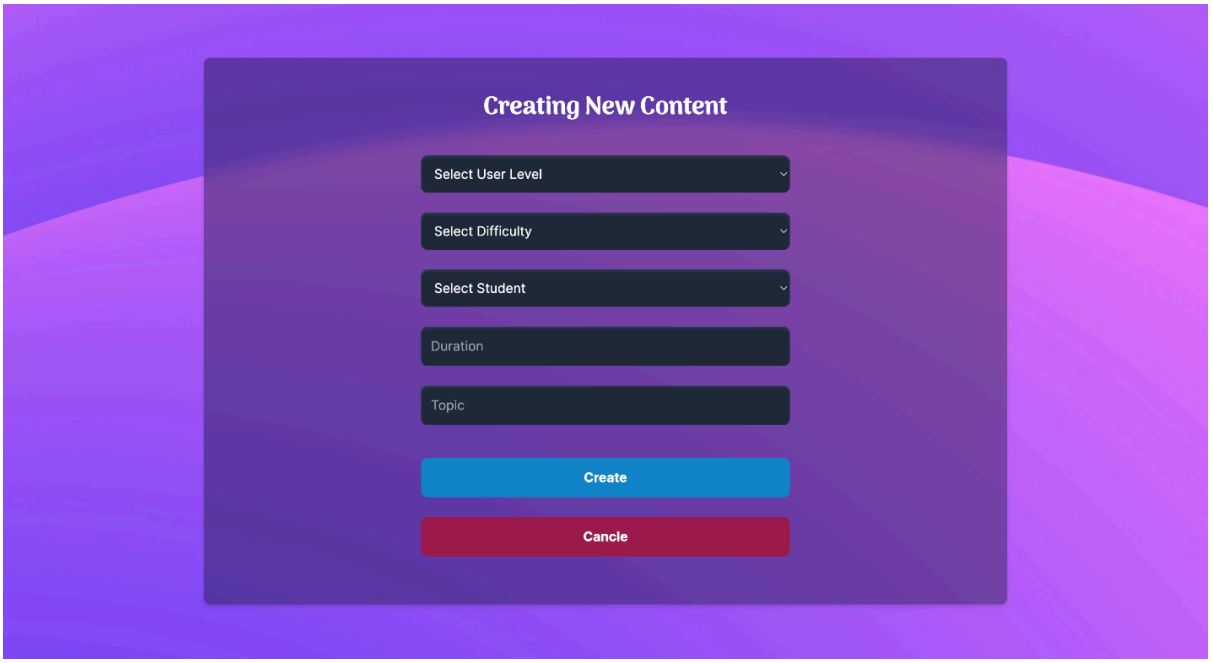

Figure 12. Creating New Content

5.2.6 Manage Students

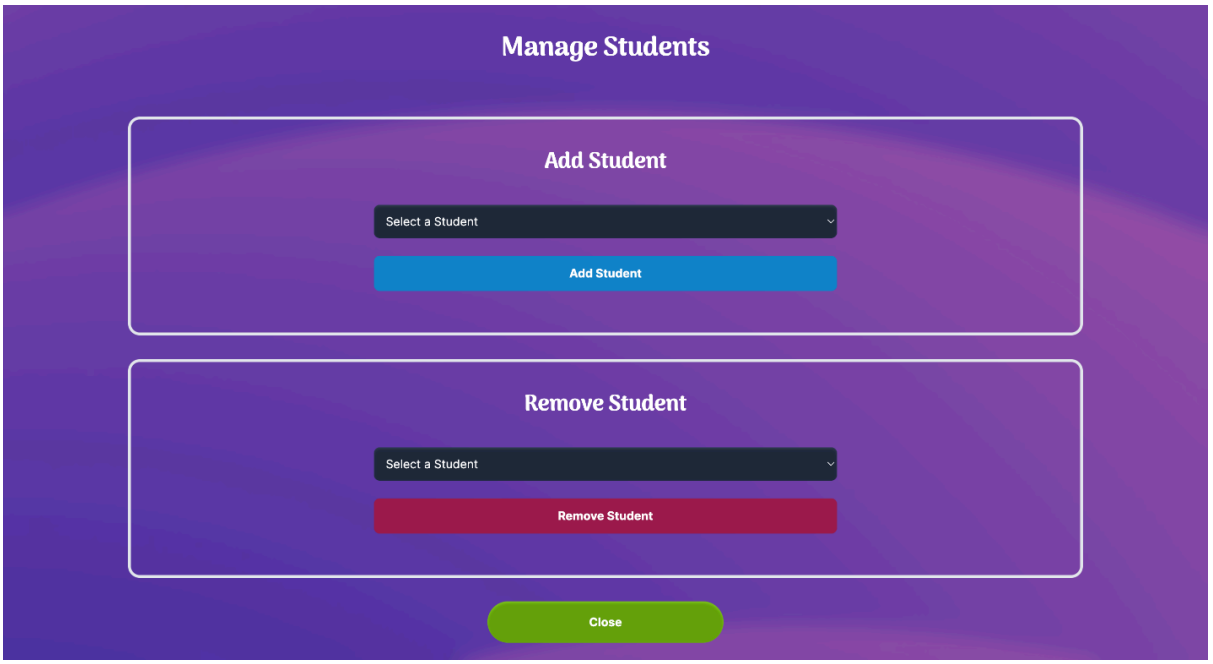

Figure 13. Manage Students

5.2.7 Conversation Page

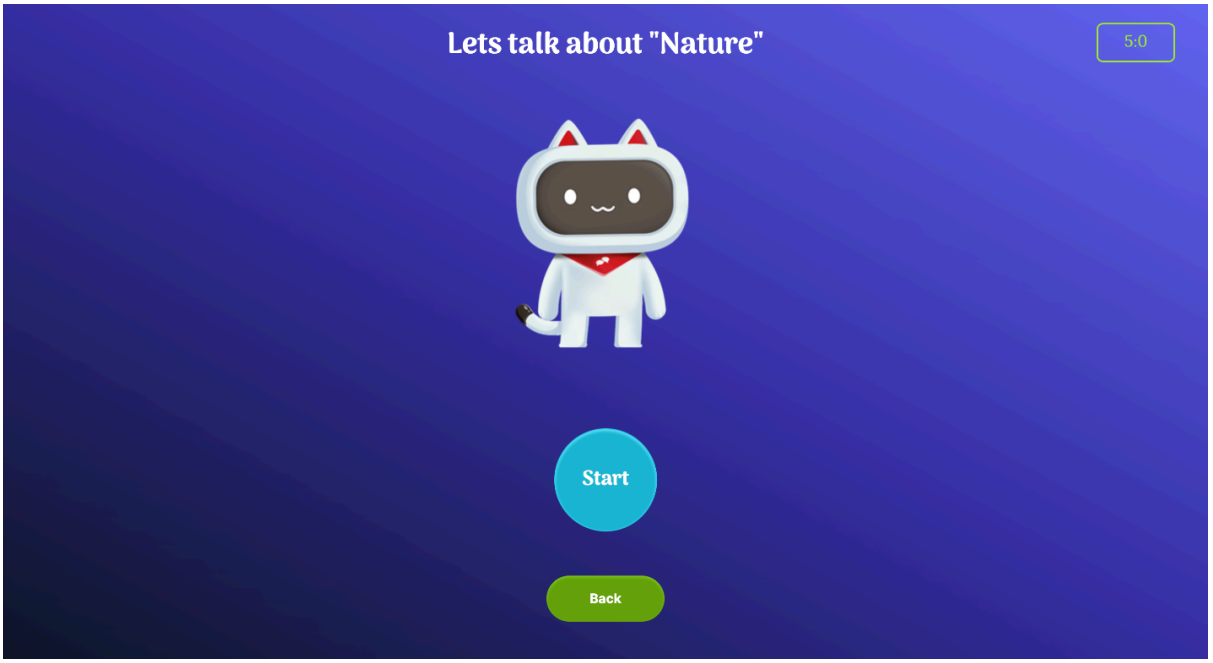

Figure 14. Conversation Page - part 1

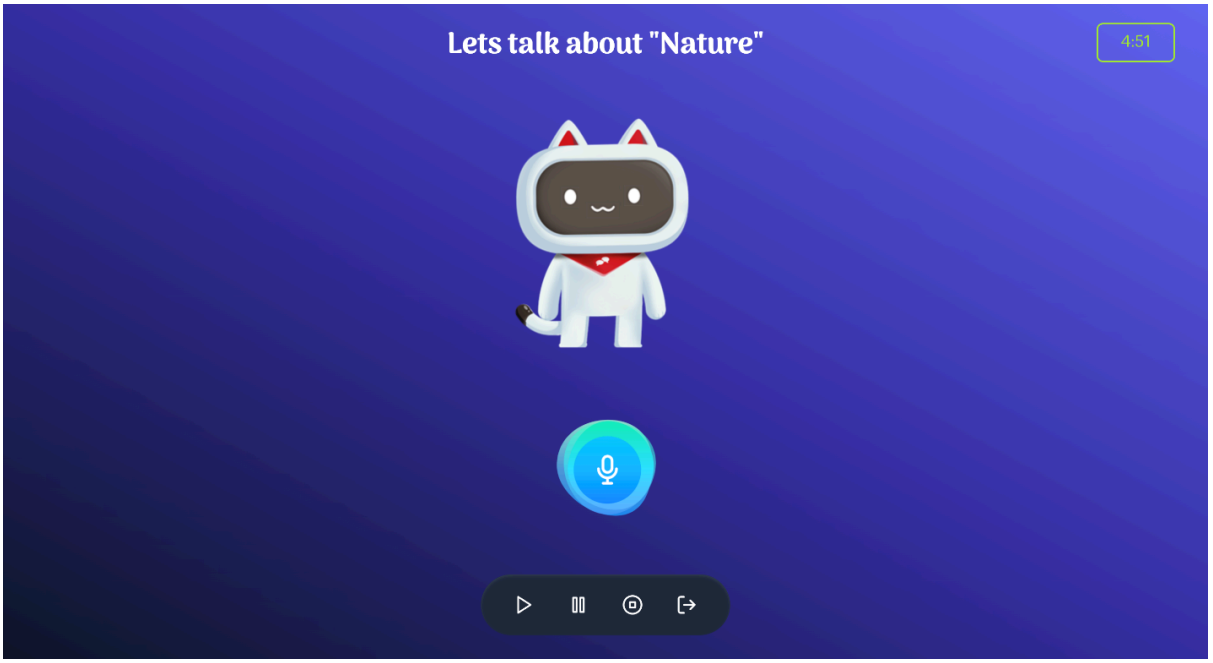

Figure 15. Conversation Page - part 2

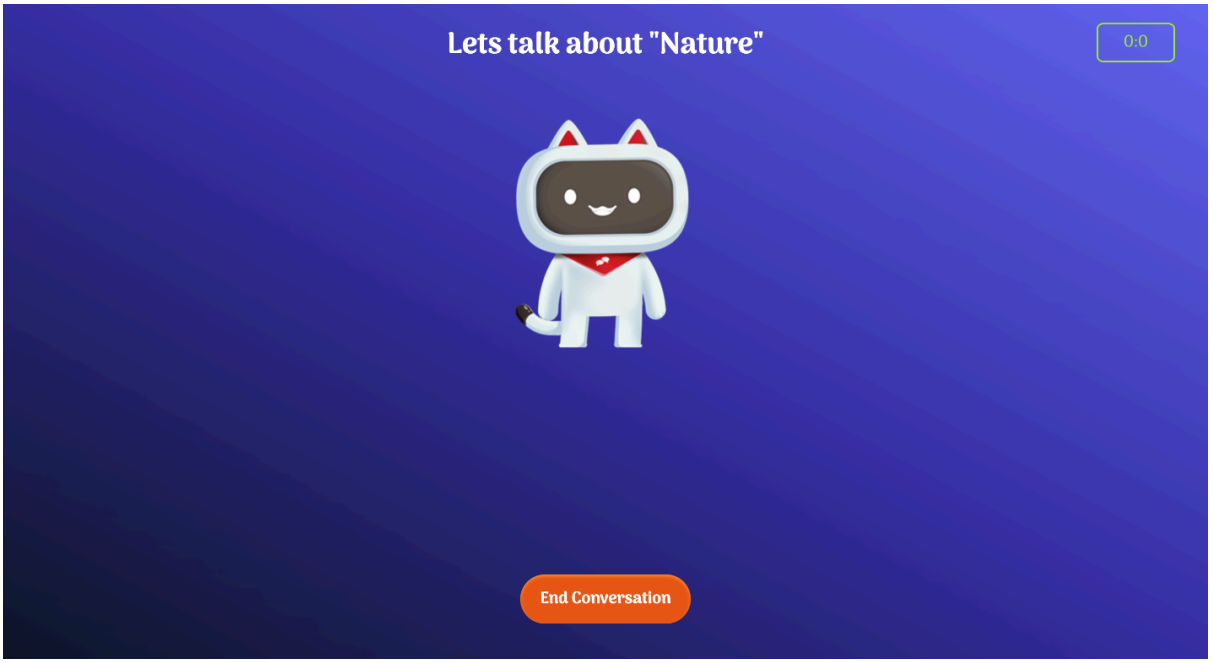

Figure 16. Conversation Page - part 3

5.2.8 Feedback Page

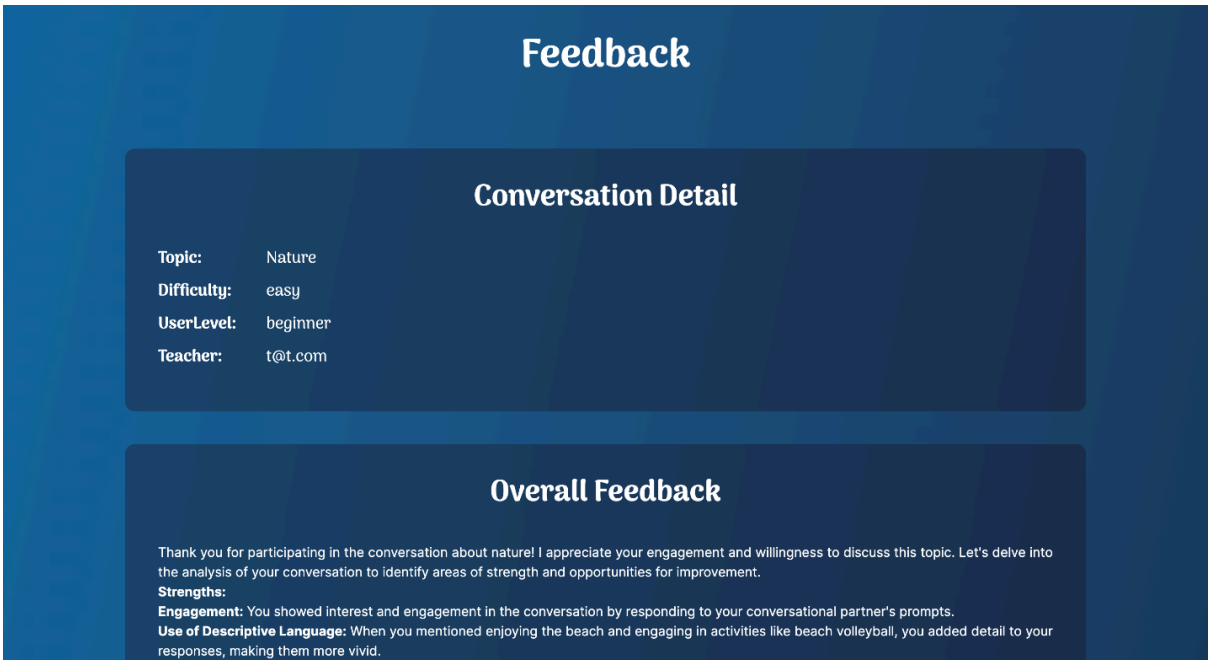

Figure 17. Feedback Page - part 1

Figure 18. Feedback Page - part 2

Figure 19. Feedback Page - part 3

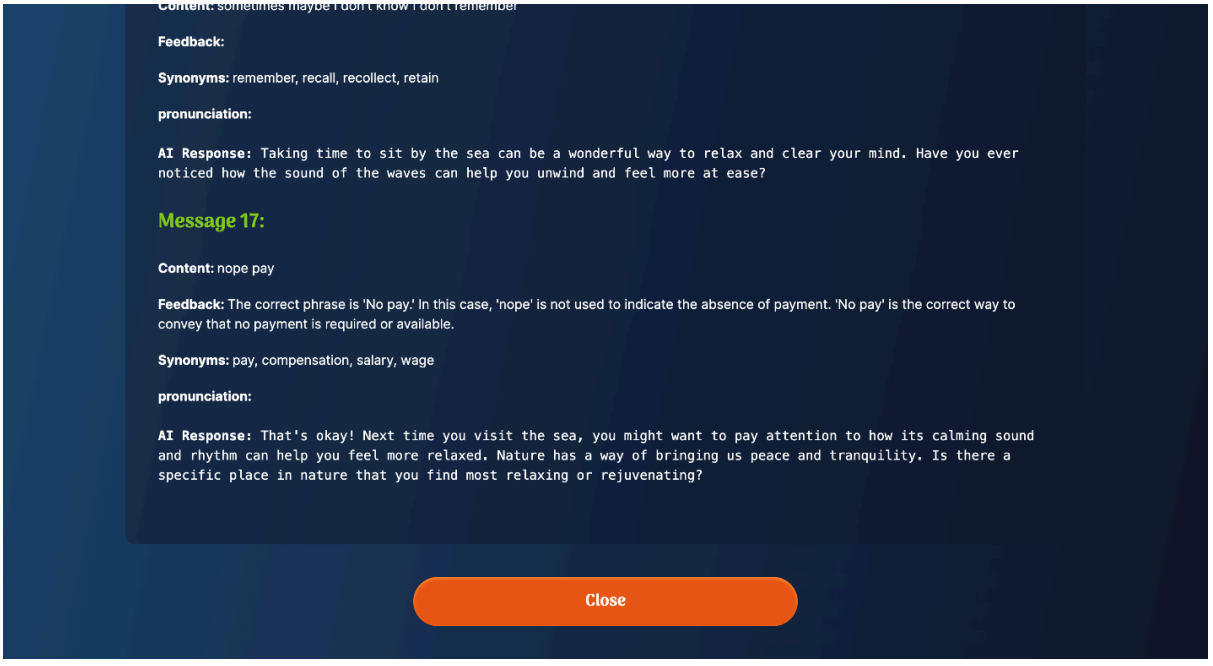

Figure 20. Feedback Page - part 4