

**Technical University of Moldova**

**Platformă educațională îmbunătățită prin modele istorice AI interactive**

**Educational platform enhanced by interactive historical AI models**

|  |  |
| --- | --- |
| **Student:** | **Al Haj Ahmed,**  **gr. FAF 211** |
| **Coordonator:** | **Braga Vasile,**  **PHD, univ. lect.** |

**Chisinau, 2025**

**MINISTRY OF EDUCATION AND RESEARCH**

**Technical University of Moldova**

**Faculty of Computers, Informatics, and Microelectronics**

**Department of Software Engineering and Automation**

**Approved**

**Head of the department:**

**Ion FIODOROV, Assoc. Prof., PhD**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**„ ” May, 2025**

**Educational platform enhanced by interactive historical AI models**

**Bachelor's Thesis**

|  |  |  |
| --- | --- | --- |
| **Student:** | **\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Al Haj Ahmed,**  **gr. FAF-211** |
| **Coordinator:** | **\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Braga Vasile,**  **PHD, univ. lect.** |
| **Adviser:** | **\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Istrati Daniela,**  **PHD, Asoc. prof.** |

**Chisinau, 2025**

**Technical University of Moldova**

**Faculty of Computers, Informatics, and Microelectronics**

**Department of Software Engineering and Automation**

**Software Engineering Study Programme**

**Approved**

**Head of the department:**

**Ion FIODOROV, Assoc. Prof., PhD**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**„ 01 ” November, 2025**

**PROJECT OUTLINE**

**for the Bachelor’s Thesis of the student:**

*Al Haj Ahmed*

*(name and surname of the student)*

1. **Title of the Bachelor's Thesis:** Educational platform enhanced by interactive historical AI models, **Confirmed by the Faculty Council Decision No.** 2 **from** 01.11.2024
2. **Deadline for submitting the Bachelor's Thesis:** 20.05.2025
3. **Initial data for developing the Bachelor's Thesis:** Develop an educational platform with historical figures powered by AI technology with 3d models, voice recognition and TTS.
4. **Contents of the Explanatory Report:**

Introduction

1. Domain Analysis
2. System Modelling And Design
3. System Implementation
4. Documentation
5. Cost Estimation

Conclusions

1. **Contents of the Graphical/Illustrative Component of the Bachelor's Thesis:**

* System architecture diagram (Client(Frontend) ↔ Flask API ↔ CharachtersDBs);
* Charts showing survey results regarding the clients recommendation and focus;
* Sequence diagram for Client interactions with the AI (teacher/student/school);
* Class diagrams describing the general structure of features in the system;
* Screenshot of the Front end design and AI response generation;

1. **Advisory Board:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name, surname** | **Type** | **Confirmation of Activity Completion** | |
| **Adviser’s signature & date** | **Student’s signature & date** |
| Istrati Daniela | *Quality control standards and technological standards* |  |  |

1. **Submission date of the Project Outline: 02.09.2024**

**Coordinator: Braga Vasile \_\_\_\_\_\_\_\_\_\_**

*signature*

**The task has been assigned for execution to the student** Al Haj Ahmed

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*02.09.2024*

*signature, date*

**SCHEDULED ACTIVITIES**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Project development stage** | **Deadline for completion the stage** | **Comments** |
|  | **Task development, task assignment** | *02.09.24– 15.10.24* | *10 %* |
|  | **Domain analysis** | *16.10.24– 27.11.24* | *15 %* |
|  | **Project management** | *28.11.24 – 15.12.24* | *15 %* |
|  | **System design** | *16.12.24 – 07.02.25* | *35 %* |
|  | **System implementation** | *08.02.25 – 29.04.25* | *10 %* |
|  | **System description** | *30.04.25 – 12.05.25* | *10 %* |
|  | **Project completion** | *13.05.25 – 20.05.25* | *5 %* |

|  |  |
| --- | --- |
| **Student:** | **Al Haj Ahmed(\_\_\_\_\_\_\_\_\_\_\_)** |
|  |  |
| **Bachelor’s Thesis Coordinator:** | **Braga Vasile (\_\_\_\_\_\_\_\_\_\_\_)** |

**TECHNICAL UNIVERSITY OF MOLDOVA**

**FACULTY OF COMPUTERS, INFORMATICS, AND MICROELECTRONICS**

**DEPARTMENT OF SOFTWARE ENGINEERING AND AUTOMATION**

**SOFTWARE ENGINEERING STUDY PROGRAMME**

**COORDINATOR'S APPROVAL FOR BACHELOR'S THESIS**

**Thesis title:** Educational platform enhanced by interactive historical AI models

**Student** Al Haj Ahmed, group FAF-211

**1. Relevance of the topic:** It helps modernize the teaching system with the use of LLM’s to aid our students with their studies.

**2. Project characteristics:** 3D interactive interface, TTS for responses, voice recognition and live detailed answers from the characters.

**3. Prototype analysis**: the prototype covers the base functionality which will have a 3d interface through which the user will be able to interact, modify, discuss and control the AI, detailed character design with accurate historical knowledge base, a context saving system using vertex database.

**4. Estimation of the results:** the results are according to the documentation and the demo runs as described, the responses from the characters correspond to the persona, the context recall is accurate, voice recognition is functional and TTS is working fine, the interactive 3d classroom design is present to the eye, and the interactive parts of the UI works as intended.

**5. Accuracy of the presented material:** Accurate, detailed and well organized.

**6. Quality of graphic materials:** well made, descriptive, organized and cover most of the implemented functionality in the system as well as the use cases, sequences and deployment.

**7. Practical value of the thesis:** the thesis can be implemented into the schooling system for educational aid or be deployed separately as a consultant service that can be monetized.

**8. Observations and recommendations:** some minor adjustments in the layout of the report.

**9. Student's characteristics and degree awarded:** Al Haj Ahmed demonstrated strong analytical abilities as well as high dedication, creativity, fast delivery, quality execution and deserves the title of Bachelor of Engineering. Based on the above, it follows that the bachelor's thesis may be admitted for defence, with the grade \_\_\_\_\_\_\_\_\_.

**The electronic version of the paper corresponds to the original submitted for public defense. The software solution has been completed and functions according to the specifications / The software solution has been partially completed, and the implemented functionalities work according to the requirements.**

**Bachelor's Thesis Coordinator \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** / Braga Vasile, *Lect. univ.*

*Signature*

.

**Both variants will be written by hand.**

**Declarație pe propria răspundere**

Subsemnata, Al Haj Ahmed, declar pe proprie răspundere, că lucrarea de față este rezultatul muncii mele, realizată pe baza propriilor cercetări și pe baza informațiilor obținute din surse care au fost citate și indicate, conform normelor etice, în note și în bibliografie.

Declar că lucrarea nu a mai fost prezentată sub această formă la nici o instituție de învățământ superior în vederea obținerii titlului de inginer licențiat

Semnătura autorului \_\_\_\_\_\_\_\_\_\_\_\_\_

**Personal statement of responsibility**

I, the undersigned, Al Haj Ahmed, hereby declare on my own responsibility that the present work is the result of my own efforts, carried out based on my own research and on information obtained from sources that have been cited and referenced, in accordance with ethical standards, in the notes and bibliography.  
I also declare that this work has not been previously submitted in this form to any higher education institution for the purpose of obtaining the Bachelor of Engineering degree.

Author's signature \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# REZUMAT

Acest proiect își propune să creeze o platformă care să-i ajute pe studenți și profesori să cerceteze, să predea și să învețe diverse subiecte mai ușor, prin a face figuri istorice să prindă viață pentru a ajuta, asista, chestionează și evaluează progresul elevilor și profesorilor deopotrivă, iar pe parcursul acestui raport va fi analizat domeniul de potrivire produs-piață, apoi sistemul va fi modelat, prezentat și procesul de implementare explicat și testat, iar platforma a fost adusă la prețul de utilizare, dacă a fost estimată și documentată. finalizare.

Proiectul analizează potrivirea produsului-piață prin cercetări cuprinzătoare în domeniu și sondaje ale utilizatorilor, confirmând cererea de instrumente educaționale interactive și adaptative. Utilizează o arhitectură modulară, bazată pe microservicii pentru scalabilitate și folosește instrumente open-source și terțe, precum Ollama, ChromaDB, ElevenLabs și Gooey AI pentru a optimiza performanța și costul. Implementarea include un front-end bazat pe web, personalizarea bazei de cunoștințe, generarea de chestionare și testarea acurateței răspunsurilor, proiectul abordează lacunele critice din educația tradițională, oferind un mediu distractiv, accesibil și bogat în context pentru învățarea istoriei și științei. Proiectul pune bazele unei schimbări transformatoare în modul în care cursanții se implică cu conținutul istoric prin AI.

Proiectul introduce o platformă educațională inovatoare care folosește inteligența artificială pentru a recrea personaje virtuale interactive, exacte din punct de vedere istoric, ale unor figuri renumite precum Albert Einstein și Nikola Tesla. Menit să sporească implicarea studenților și să ofere experiențe de învățare personalizate, sistemul permite utilizatorilor să converse în timp real cu personaje istorice bazate pe inteligență artificială care răspund pe baza ideologiilor, cunoștințelor și contextului lor. Construită folosind modele mari de limbaj (LLM), baze de date vectoriale și tehnologii integrate, cum ar fi Text-to-Speech (TTS), animație de sincronizare buzelor și modelare 3D, platforma oferă o experiență de învățare multimodală și captivantă.

# ABSTRACT

This project aims to create a platform to aid students and teachers to research, teach and learn various topics easier through making historical figures come to life to aid, assist, quiz and evaluate the progress of students and teachers alike and throughout this report the domain will be analyzed the product-market fit will be explored then the system will be modeled, presented and the implementation process explained and tested and the platform usage documented as well as the estimated costs of the project if it was brought to completion.

The project analyzes product-market fit through comprehensive domain research and user surveys, confirming the demand for interactive and adaptive educational tools. It employs a modular, microservice-based architecture for scalability and uses open-source and third-party tools like Ollama, ChromaDB, ElevenLabs, and Gooey AI to optimize performance and cost. The implementation includes a web-based frontend, knowledge base customization, quiz generation, and response accuracy testing the project addresses critical gaps in traditional education by providing a fun, accessible, and context-rich environment for learning history and science. The project sets the foundation for a transformative shift in how learners engage with historical content through AI.

The project introduces an innovative educational platform that leverages artificial intelligence to recreate historically accurate, interactive virtual personas of renowned figures such as Albert Einstein and Nikola Tesla. Aimed at enhancing student engagement and providing personalized learning experiences, the system allows users to converse in real time with AI-driven historical characters who respond based on their ideologies, knowledge, and context. Built using large language models (LLMs), vector databases, and integrated technologies such as Text-to-Speech (TTS), lip-sync animation, and 3D modeling, the platform delivers a multimodal and immersive learning experience.

Contents

[REZUMAT 1](#_Toc198470375)

[ABSTRACT 8](#_Toc198470376)

[INTRODUCTION 5](#_Toc198470377)

[1 Domain Analysis 6](#_Toc198470378)

[1.1 Domain Description 6](#_Toc198470379)

[1.2 Project Contest and Relevance 6](#_Toc198470380)

[1.3 Stakeholders 7](#_Toc198470381)

[1.4 Current State of Educational Technology 7](#_Toc198470382)

[1.4.1 AI in Education 8](#_Toc198470383)

[1.4.2 Glossary of Terms 8](#_Toc198470384)

[1.5 Survey Analysis 8](#_Toc198470385)

[1.5.1 AI Knowledge level 8](#_Toc198470386)

[1.5.2 Subject Interest 9](#_Toc198470387)

[1.5.3 Historical personas preferences 10](#_Toc198470388)

[1.5.4 Features importance 10](#_Toc198470389)

[1.5.5 Importance of Realism 10](#_Toc198470390)

[1.5.6 Survey Analysis Conclusions 11](#_Toc198470391)

[1.6 Project Objectives 11](#_Toc198470392)

[1.6 Existing Solutions and Competition 12](#_Toc198470393)

[1.6.1 Hello History 12](#_Toc198470394)

[1.6.2 People AI 12](#_Toc198470395)

[1.7 Requirements Specification 13](#_Toc198470396)

[1.7.1 Functional Requirements 13](#_Toc198470397)

[1.7.2 Non-Functional Requirements 13](#_Toc198470398)

[2 SYSTEM MODELING AND DESIGN 15](#_Toc198470399)

[2.1 Behavioral representation of the system 15](#_Toc198470400)

[2.1.1 General representation of the system 15](#_Toc198470401)

[2.1.4 Description of application usage scenarios 18](#_Toc198470402)

[2.2.1 Static Structure of the System 20](#_Toc198470403)

[2.2.2 Dependencies between system components 21](#_Toc198470404)

[2.2.3 Modeling the development tools environment 22](#_Toc198470405)

[3 SYSTEM Implementation 26](#_Toc198470406)

[3.1 Implementation Progress 27](#_Toc198470407)

[3.1.1 Frontend 27](#_Toc198470408)

[3.1.2 LLM prompt Engineering 31](#_Toc198470409)

[3.1.3 TTS (Text-to-Speech) 31](#_Toc198470410)

[3.1.4 Lip-sync technology 32](#_Toc198470411)

[3.1.5 Database design and connections 32](#_Toc198470412)

[3.2 System Testing 33](#_Toc198470413)

[4 Documentation 35](#_Toc198470414)

[4.1 Installation Guide 35](#_Toc198470415)

[4.2 How to use the platform 35](#_Toc198470416)

[5 Cost estimation 39](#_Toc198470417)

[5.1 Core Components 39](#_Toc198470418)

[5.1.1 AI Infrastructure 39](#_Toc198470419)

[5.1.2 3D Models & Animation 39](#_Toc198470420)

[5.1.3 Educational Platform Development 39](#_Toc198470421)

[5.2 Estimated Budget Tiers 40](#_Toc198470422)

[5.3 Optional/Recurring Costs 40](#_Toc198470423)

[Conclusions 41](#_Toc198470424)

[BIBLIOGRAFIE 42](#_Toc198470425)

[ANEXA A 44](#_Toc198470426)

[Survey Questionnaire A.1 44](#_Toc198470427)

[Chalkboard design A.2 46](#_Toc198470428)

[Teacher design and animations A.3 47](#_Toc198470429)

[Chat box design A.4 50](#_Toc198470430)

[Camera control and interactions A.5 51](#_Toc198470431)

# INTRODUCTION

The rapid evolution of AI (artificial intelligence) and machine learning has massively changed how people interact with historical knowledge, our project aims to use these advancements by creating virtual personas that will imitate various historical figures, these figures will be em- powered by the knowledge, history, ideology and contextual understanding of the represented figure, this offers for the use an immersive, interactive and fun experience while interacting with these figures , by simulating these great historical figures our app aims to make history more approachable and interesting for both student’s and researchers alike, this approach will not just enhance the traditional learning methods but also help accelerate different running researches as the researchers don’t have to waste as much money and time on looking for and learning the facts of the past.

The main goal of the project is to develop an intelligent system capable of interpreting and generating responses that reflect the ideology, thought process and knowledge of these figures, which will offer a new tool for education and historical and scientific research. Through the combination of AI-driven dialog, neural networks and large language models (LLM’s) users will be able to engage in meaningful conversations with the virtual personas in real time fostering a more profound understanding of historical events, ideologies, and contexts.

The traditional school system and the delivery of the curriculum to the student has seen little to no change at all and with our new tool we hope to boost the development of a more modern approach to learning which will be more appealing, fun and most importantly easy to both the teacher and student alike which will (we hope) further improve our society in the future.

Education is changing faster than ever before. With the rise of artificial intelligence and the rapid growth of digital tools, students and teachers alike are looking for new, more engaging ways to explore and understand complex subjects. That’s where this project comes in., this project was born out of a simple but powerful idea: what if we could talk to the greatest minds in history, not just read about them?

This platform is a platform that brings historical figures like Albert Einstein or Nikola Tesla to life using advanced AI technology. These aren’t just static avatars they can talk, respond to questions, adapt to your learning style, and even quiz you on what you've learned. Whether you're a student struggling with science or a history enthusiast curious about the mindset of revolutionary thinkers, this project offers a way to make learning more personal, interactive, and fun.

This report walks through the journey of designing, building, and testing the platform. It covers everything from understanding the educational challenges we aimed to solve, to analyzing user needs through surveys, to implementing the technology stack including LLMs, 3D models, voice interaction, and more. By the end, it becomes clear that the project isn’t just a school project it’s a step toward a smarter, more inspiring future for education.

# 1 Domain Analysis

The domain of the project lies at the intersection of artificial intelligence, education and his- tory specifically we are using the Natural language processing (NLP) of large language models (LLM’s) like ChatGPT to recreate the personalities and intellect of the historical figures. This virtual persona system will use AI techniques such as machine learning and neural networks to simulate how the figure might respond to different questions based on their knowledge base and characteristics.

1.1 Domain Description

**goal:** to create an AI system that allows users (students/teachers) to interact with virtual versions of historical figures to help with student's educational process;

**scope:** educational platform, e-learning, history classes, research, museums...;

**problem:** students are looking for more engaging and fun ways to learn instead of the normal schooling process, the lack of communication between teacher and student;

**solution:** AI-powered, interactive virtual personas of historical figures to teach, quiz, and inspire.

1.2 Project Contest and Relevance

In the last couple of years, the demand and use of a personalized and adaptive learning experience has increased tremendously, the traditional education system is still stuck in the past and often struggles to personalize the learning experience for each student’s needs and understanding which leads to the disengagement of students and overall lack of motivation, to solve this issue our app the LEApS project is designed to use the vast knowledge of the past and leverage it to provide a more interactive , personalized and fun experience for all students.

Key factors driving the relevance of LEApS include:

**growing need for personalized learning:** with the increasing demand of personalized learning, home schooling and self-teaching students and teachers are seeking tools that can help with this new way of learning, our app offers this possibility as the student can have conversations with our historical figures and dive deep into the subject of their liking at their own pace;

**limited access to quality education:** in many parts of the world students face challenges to get quality education, our app offers a solution as it can be accessed remotely from any place in the world offering the student the ability to dive deep into complex topics as they like;

**technological advancements:** with the rapid advancement of AI technologies, particularly in natural language processing and machine learning, schools can now offer more dynamic and adaptive tools to better the learning experience, our project capitalizes on these advancements bringing historical figures back to life through AI to help our new generation;

**engagement challenges:** modern learners, especially younger generations, require more engaging and interactive methods of learning. Traditional textbooks and lectures often fail to capture students’ attention. LEApS provide an immersive learning experience through virtual conversations with iconic figures, making history and science more engaging and relatable.

This project doesn’t only respond to current issues/gaps in the traditional education system but also provide a forward-looking solution that can adapt to all the technological advances to further push our schooling system, including virtual classrooms, hybrid learning model and self-teaching ideology students and teachers alike can use our personas as teachers, assistants and/or coaches.

1.3 Stakeholders

The stakeholders in the LEApS project include:

**students:** they can use the platform to aid their learning by interacting with the personas and engaging in deep discussions about complex topics at their own pace ensuring a personalized experience as well as monitor and generate quizzes for the student and help them monitor and advance their learning;

**teachers:** teachers can integrate the AI system into their curriculum providing more context, responding to questions, or helping them write tests and give advice like a virtual assistant;

**historians and researchers:** this group can benefit from the vast knowledge base of various experts on different subjects as they can obtain any historical data accurately quickly;

**AI and educational technologists:** professionals in AI development and educational technology are key to ensuring the success of the project, as they provide the necessary expertise for developing and maintaining the platform.

1.4 Current State of Educational Technology

In the current day and age, we see more and more students drive away from the normal schooling methods, the home schooling and online schooling rates raised considerably after the appearance of covid, AI Chatbots and detailed and fun educational platforms as can be observed from the **figure 1.1**.

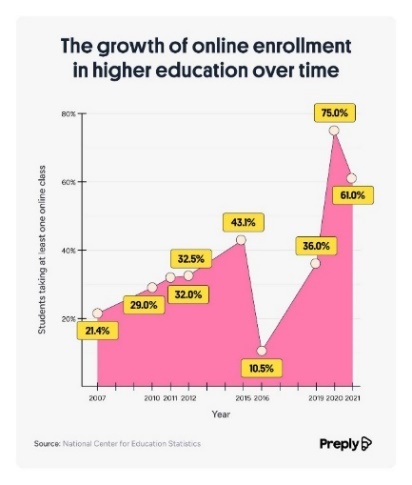


Figure 1.1 - Online enrollment

1.4.1 AI in Education

Current applications of AI in education primarily focus on:

**content recommendation:** AI systems suggest learning materials based on a student’s progress, preferences, and learning style;

**tutoring capabilities:** AI tutors provide basic guidance, explanations, and practice exercises, often personalized to the learner’s needs;

**limited personalization:** while there are some personalized learning platforms, few of them can simulate real-time interaction with complex historical personas.

1.4.2 Glossary of Terms

Here are some domain specific terms that will be used during the documentation of the project and their definition:

**virtual persona:** an AI model impersonating a historical figure;

**knowledge base**: the historical data specific to the character;

**dialogue system**: the system powering the natural language interaction with the customer;

**TTS (Text-to-Speech) and Lip-sync AI:** for voice recognition and realistic facial animations.

1.5 Survey Analysis

A Survey was conducted on a sample of UTM students about their virtual AI knowledge, personal preferences and feedback to determine the amount of traction the project would receive, the overall familiarity that the people have to such tech as well as their interest in the provided solution idea.

The goal of the survey was to ensure the focus of the project, the presence of need in the market, the feature specifications, market size, potential customers and pricing ranges, as the survey includes questions about subject interests, AI-familiarity, realism importance, character selection, privacy, kind of help needed and much more.

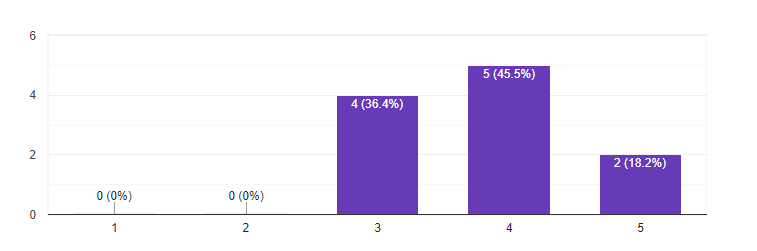
And according to the results that was analyzed we can observe the following:

1.5.1 AI Knowledge level

The appearance of the new LLM technology mainly chatGPT has raised the AI awareness level among the people and young students in particular as it can be very useful in automating mundane tasks

As more and more people are using AI’s nowadays the basic knowledge of their power and usefulness is becoming more and more apparent for people especially the younger generation after the appearance of the different tools that use the power of LLM’s either for generating essays responding to difficult questions or even solving homework and it will only rise with time ensuring the need and ask for systems such as this project.

As presented in **figure 1.2** you can see that the knowledge level of AI and its implementations are quite high indicating a high interest for such solutions as this one.

Figure 1.2 - AI familiarity

Key Insights:

the majority of people already have a very good understanding of AI and its capabilities;

this data should confirm that the people would respond positively to the integration of our tool into the education system.

1.5.2 Subject Interest

As can be deducted from the **figure 1.3**, people's interests are more focused on scientific objects, as we suspected from a mostly student audience

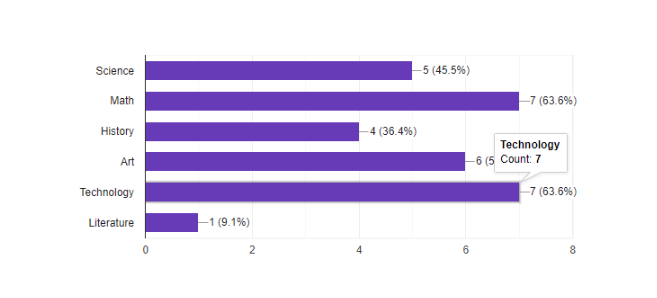


Figure 1.3 - Subject interest

Key Insights:

people would most likely want to meet and discuss with people of great knowledge or historical significance as we can see most votes go to math tech and art;

people imagine our app as a tool to help guide students with different hard technical subjects.

1.5.3 Historical personas preferences

Responses varied from artistic people like De Vinci to historical characters like Napoleon Bonaparte but most of the responded focused on scientific characters like Albert Einstein or Tesla.

Key Insights:

the app will focus on the people with the bigger scientific background for a better experience;

people ask for characters with very odd/complex personalities, thoughts and wisdom which further will help them grow their knowledge of the great people who changed the world.

1.5.4 Features importance

Most people saw that a deep knowledge of the character's subject and background as well as their ability to personalize the learning experience are the most important parts as presented in **figure 1.4**:

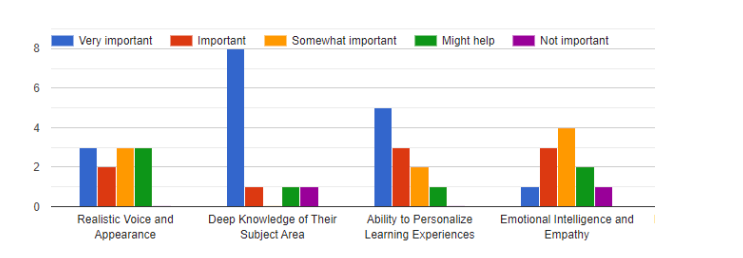


Figure 1.4 – Features

Key Insights:

the platform must focus more on the authenticity of the character as well as its knowledge to try and simulate as good of a replica as possible;

the app should focus as well to personalize every experience so that the AI adapts to the questions style and every user's desired answer format.

1.5.5 Importance of Realism

Realism plays a big role in the development of games, applications, movies or any software solutions as they help with the experience of the user and help to retain the user for much longer in the app or on the platform but for an educational platform it’s much better to keep the students focused on the tasks on hand and/or the subjects they’re studying rather than at the teacher himself so the 3d modeling feature was pushed further into the development cycle.

As for this project it aims to make the app as cost efficient and useful as possible

According to **figure 1.5** the majority of people saw that being the 3d model as realistic as possible isn't the main concern as they want a pleasant experience that both looks ok and performs well.

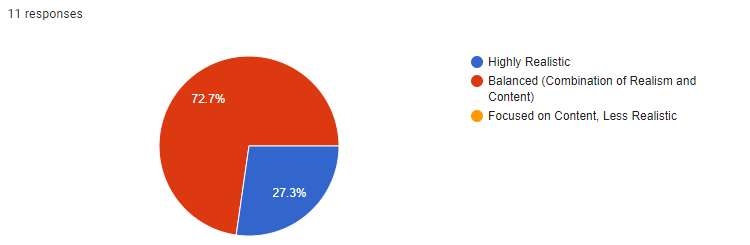


Figure 1.5 - Realism importance

Key Insights:

the app should be a balance between realism and performance;

ultra-realism isn't the focus of the platform as people want to explore the persona's mind as well.

1.5.6 Survey Analysis Conclusions

In conclusion, the platform should focus on personalization, Knowledge, interactivity, and support, while also offering a balance between traditional education and alternative pathways to meet the diverse needs of its users.

1.6 Project Objectives

The LEAps project is driven by several key objective aimed at transforming how the students, teachers and researchers interact with history and historical knowledge, these objectives aim to deliver a high quality, AI powered educational and coaching platform that enhances the learning experience and provides a unique, personalized, and fun approach to learning.

**Develop and launch an AI-Powered Interactive Learning Platform:** we hope to deliver (by the end of the year) at least 3 virtual personas powered by the knowledgebase corresponding to the historical figure it represents as well as have the 3D model for each character.

**Achieve High User Satisfaction:** we aim to maintain above 80% satisfaction rate which we will try to maintain through monitoring user feedback and having an interactive fun interface through which we introduce our gamification elements.

**Implement Personalized Learning Paths:** our platform will offer personalized experience that will correspond to their knowledge level, learning speed and performance metrics ensuring that every student gets the best guidance they need.

**Increase Student Engagement by 30%:** compared to traditional online learning plat- forms, LEApS aim to increase student engagement by 30%. This will be done by creating an intuitive, interactive platform that keeps the user engaged and having fun while learning.

**Achieve a 95 % Accuracy Rate in AI Responses:** the platform’s AI personas will aim to provide accurate, historically, and contextually appropriate responses to user queries. This accuracy will be critical in ensuring that users can rely on the virtual figures as credible sources of information.

These objectives reflect the project’s mission to advance education with the power of AI ensuring that teachers and students get a personalized, engaging, and fun experience, LEApS seek to create a platform that not only improves knowledge retention but also makes learning more enjoyable and accessible.

1.6 Existing Solutions and Competition

According to our research regarding similar solution in the market, most of the solution found were just simple assistant-like Chatbots that are given some kind of back knowledge without any accurate information or any visual or audio interface, the closest two that stood out were Hello history and People AI:

1.6.1 Hello History

Hello history is an app that provides the ability to chat with various AI powered historical figures and other great minds to be implemented in business and education. And the **figure 1.6** represents their website.

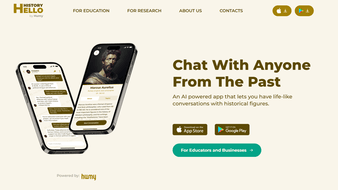


Figure 1.6 - Hello History platform

The main differences between their approach and this solution is the ability to converse and interact almost live with that character as the project aims to bring more life to the characters using the custom animations and lip-sync.

1.6.2 People AI

People AI is an online platform that aims to give people the ability to converse and discuss with people from the past and learn from their history and personal experiences, as well as many other historical figures celebrities and famous criminals which only gives the ability to interact with them via text and it’s a chat base system.

In the following **figure 1.7** is their website.

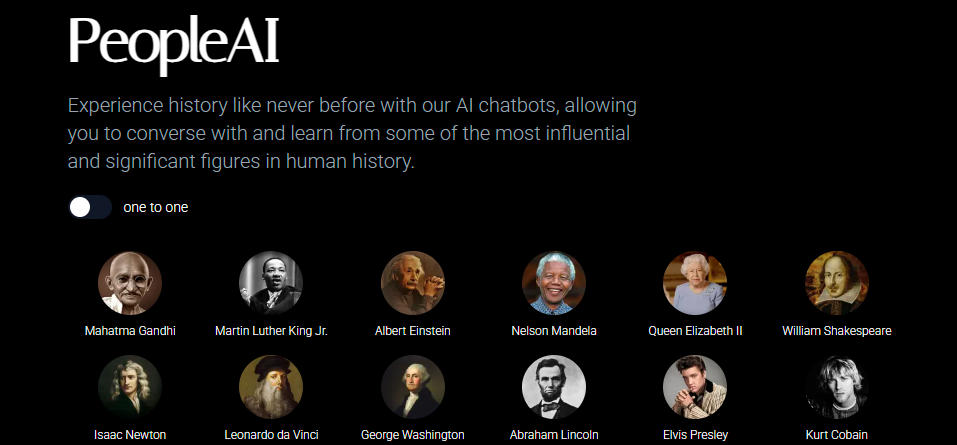


Figure 1.7 - People-AI platform

As for this platform the differences remain the same as they all provide the simple text style chat without any audio or animation as well as the lack of verification on their knowledge based as this solution aims for a near 100% accuracy regarding the characters past and experience as it will be gained from trusted Sources and will be available to check on the platform.

1.7 Requirements Specification

In this section the core functional and non-functional requirements will be specified and listed of the projects platform.

1.7.1 Functional Requirements

Historical Figure AI Characters:

system must support multiple AI characters (Einstein, Tesla, Picasso);

each character must maintain consistent personality and knowledge domain;

characters must provide historically accurate information.

User Interaction

support for text-based input and output;

voice interaction capabilities (speech-to-text and text-to-speech);

support for image and PDF document processing.

Content Processing

process and analyze uploaded images;

extract and summarize content from PDF documents;

generate contextual responses based on uploaded content.

1.7.2 Non-Functional Requirements

Performance Requirements:

AI response generation: < 2 seconds;

voice processing: < 1 second;

image/PDF processing: < 5 seconds.

Reliability:

system availability: 99.9%;

data persistence and backup;

error handling and recovery.

Scalability Requirements:

support for multiple concurrent users ;

efficient resource utilization;

horizontal scaling capability.

# 2 SYSTEM MODELING AND DESIGN

When designing a new system it usually include selecting the best software design pattern keeping in consideration scale and accessibility as well as fault tolerance and ease of implementation, In this project the **MVC** design pattern was selected and The **Microservices** architecture for scaling and accessibility.

An MVC design pattern is a design pattern in which we have a View with which the user interacts, a Model in which the data are stored and all the static knowledge and operations of the system and a Controller that routes the different requests from and to the View and Model.

**MVC** stands for **Model-View-Controller**, a software architectural pattern used to separate concerns in an application. It breaks down as follows:

**model**: handles the data and business logic (e.g., database access, data processing).

**view**: displays the data (UI elements like web pages or app screens).

**controller**: manages input (like user actions), updates the model, and refreshes the view.

Microservice architecture is a design approach in software development where a large application is built as a suite of small, independently deployable services. Each microservice is responsible for a specific business capability, such as user authentication, product catalog management, or order processing. These services run in their own process and often use lightweight communication methods like HTTP REST, gRPC, or messaging queues (e.g., RabbitMQ or Kafka) to interact with each other.

Why Microservices? Because the microservice architecture provides a loosely coupled easy to duplicate and expand service which allows us to scale horizontally securely without any issues and it insures that the platform stays up during maintenance/development.

2.1 Behavioral representation of the system

The behavioral representation of a system includes the dev’s imagination and picture of the system, the various use cases through which the platform’s features/functionalities can be tested as well as the relations between all the components, algorithms and services used in the project as well as the general planned structure of the project and its components.

2.1.1 General representation of the system

The general representation of the system will be presented using the following Use case diagrams in which the different function/features of the platform will be presented as well as how the user will interact with the platform and its functions and different cases through which the features of the app would be presented.

In the following diagrams would be presented the use cases of the app, different sequence diagrams as well as deployment and class diagrams which would cover every feature and functionality in detail.

Which represent some general cases through which the user in our case the teacher/student and how they would interact with the app/system and what are the different features that would be available to them as well as the interactions between some of the main components of said system and what sub features each one has which will make the system easier to imagine and implement clearing any confusion on what are the needed features for development.

In **figure 2.1** is represented the system and its general use case

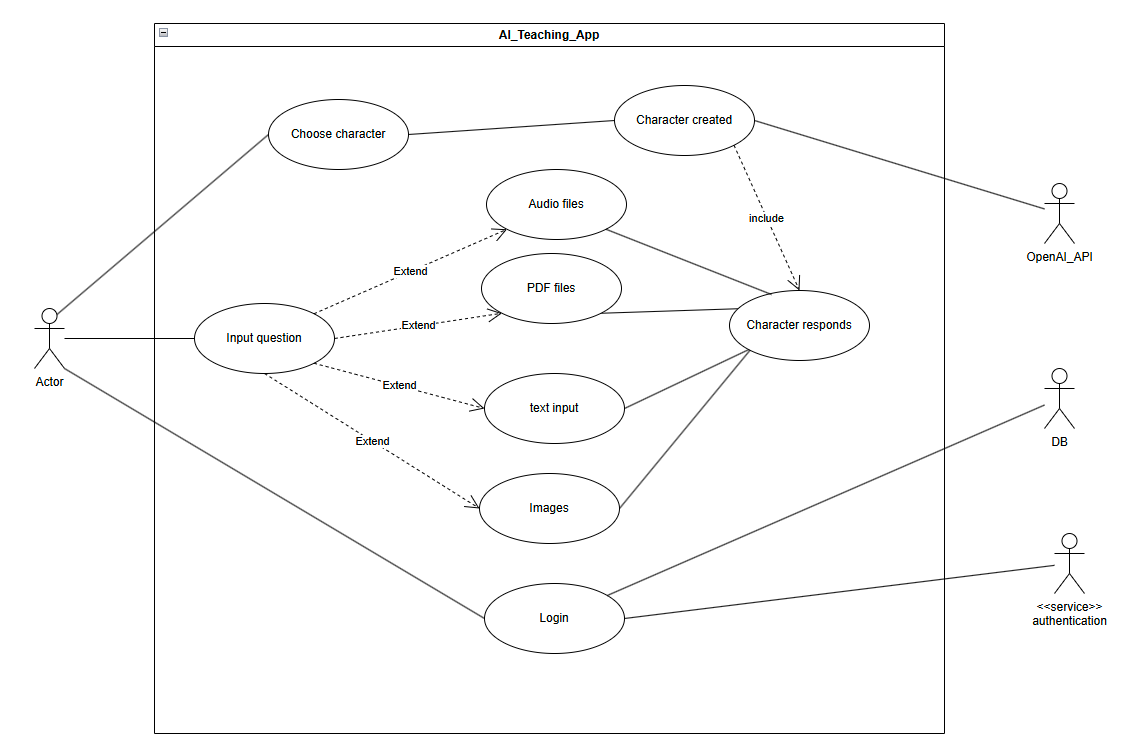


Figure 2.1 - General use case diagram

**Description:** this is a general representation of the client interacting with the various parts/aspect of the system where he can interact with the persona, change the character, and ask the persona different kind of questions in various forms (text, audio, or file).

In **figure 22** the teachers use case is represented and how and what features he would be able to interact with on the platform

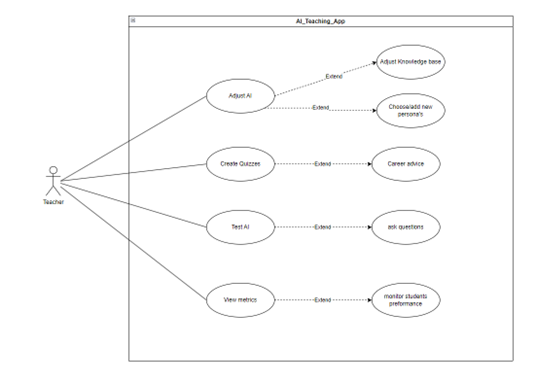


Figure 2.2 - Teacher Use Case diagram

**Description:** in this diagram we represent how the teacher would interact with our system as you can see, he can update the AIs knowledgebase, help it create more accurate tests/quizzes or view various metrics of the students using the app

The teacher will also be able to monitor students’ progress and answers as well as their overall performance to make sure to work on each individual student’s weaknesses.

In figure **2.3** the student’s point of view and use cases are represented

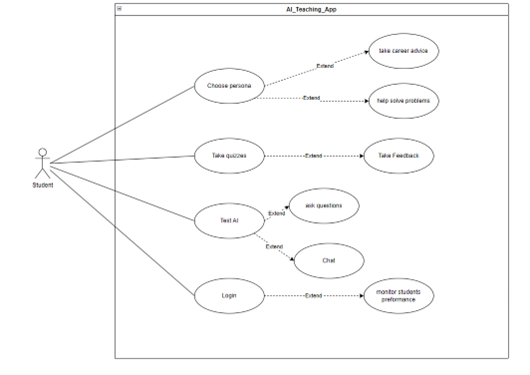


Figure 2.3 - Students Use Case diagram

**Description:** in this diagram you can see how the student would interact with the system as he can login to maintain his monitoring data, try out the quizzes/tests generated by the AI, test the AI knowledge through simple Q&A as well as choose what character he would like to see/talk to to help them with their career/studies.

**Figure 2.4** represents the schools functions and the features they’ll control and the various features they can interact with

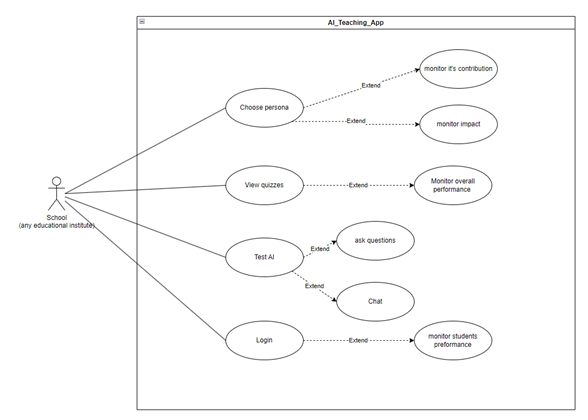


Figure 2.4 - Schools Use Case diagram

**Description:** in this diagram you can see how the school or educational institute would interact with the app as they can monitor the progress of whole classes through the graphing system, test the AI themselves, and choose what characters to add to the roster and other monitoring tools that would help them control the progress of the school.

2.1.4 Description of application usage scenarios

In the following Sequence diagrams will be presenting the various scenarios through which the platforms features/functionalities interactions with the user and other components of the system can be observed and the data exchange happening throughout.

In **figure 2.5** is represented the sequence of actions and the flow of data during a text input.

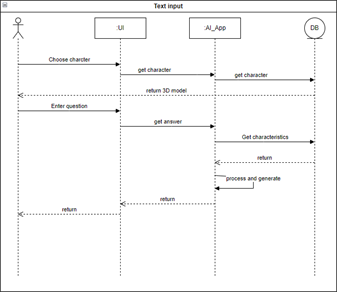


Figure 2.5 - Text Input Sequence Diagram

**Description:** in this diagram we can see how the user will proceed with asking the AI a question through text and how the result will be parsed and the answer generated and returned.

In **figure 2.6** is represented the sequence of calls during an update of the AIs knowledge base.

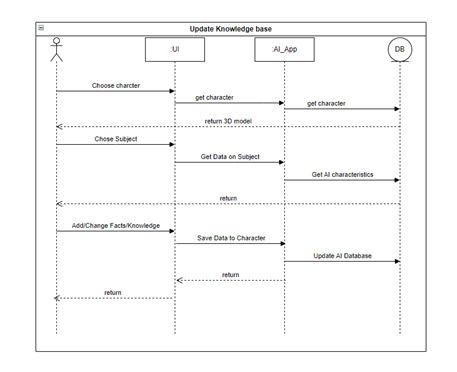


Figure 26 - Update Sequence diagram

**Description:** in this diagram you can see how the teacher or professor updates the knowledgebase of the AI through our interface using the admins UI through which he will be able to modify and monitor the information that each of the character will have and their accuracy as well as their strictness to the subject and showcase how the messages travel between the UI to the app and to and from the database ensuring safe and easy updates.

In **figure 2.7** is represented the sequence of function calls during a quiz generation or submission:

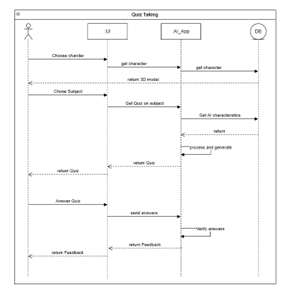


Figure 2.7 - Quiz Sequence diagram

**Description:** in this diagram we can see how the user will proceed to take one of the generated quizzes and monitor their results.

In **figure 2.8** is represented the sequence of calls during an audio file submission or an audio input case

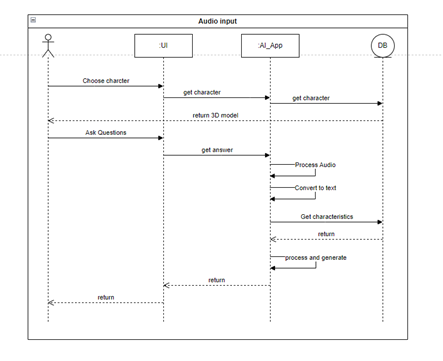


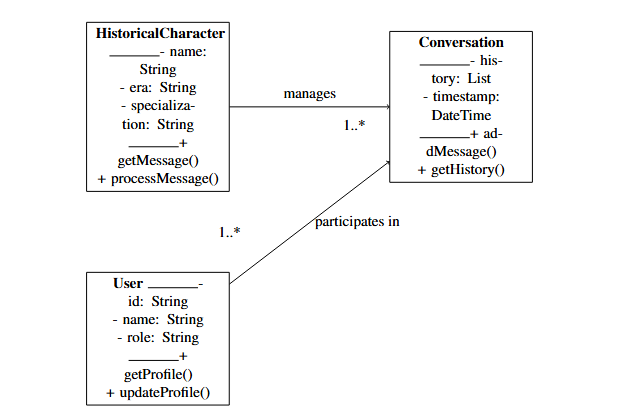
Figure 2.8 - Audio Input Sequence diagram

**Description:** in this diagram you can see how the app would go about processing the audio input, generating the response, and converting it through TTS all of this is going to be possible using Microsoft’s Azure for both the lip sync and the TTS which is a ai powered tool that has a big library of voices.

2.2.1 Static Structure of the System

In the following Class diagram will be represented a simplistic version of the static structure of the system and how it interacts on a data level and how the various components store, send and receive data across the platform:

In **figure 2.9** is represented the core system class relations, attributes and a simplified view of the components of the system

  
Figure 2.9 - Core System Classes

**Description:** in this diagram you can see the general class structure of the system as we intend to use a microservice approach to each of the services that the app will provide so we will have the AI machine learning engine that will be reviewing the historical data through an api as well as a user management service to personalize the experience as well as the conversational/tts and audio processing service and the 3d and lipsync models.

As can be seen from the class diagrams that there are 3 main components to the system the UI/frontend and the backend character design and knowledge base storage system as well as caching and the api that connects them and the 3rd party services used to generate the voices, voice recognition and animations.

In **figure 2.10** is represented the basic functionalities elements and how they interact and would be implemented:

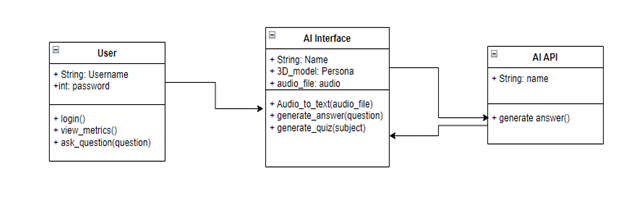


Figure 2.10 - Basic Functionality

**Description:** in this diagram we can see how the users will be defined as well as how they would interact with the AI persona and the flow of data from the AI through an API gateway and back to the interface as the requests that were sent by the user can be processed by the system and add all the prompts needed to control the response from the AI through prompt engineering.

2.2.2 Dependencies between system components

A component diagram is a visual way to show how the different parts of a software system are organized and how they interact. Each "component" represents a module or part of the system, like a login service or a database, and the diagram shows how these parts connect and depend on each other. It's like a blueprint that helps developers understand the structure of the system and how the pieces fit together.

In the following Component Diagrams shown in **figure 2.11** will be represented the dependency relations between component of the system and the general data flow throughout the system:

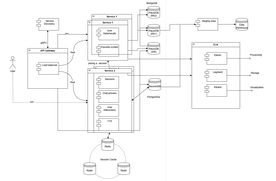


Figure .11 - System Architecture

**Description:** this will be our architecture that we will use, we will have a client-server architecture with microservices which ensures security and scalability we will have separate services for user management, session managements, Data logging and quizzes. As well as:

implement read replication for user data by setting up two dedicated read replicas to distribute and balance query load;

use a Two-Phase Commit (2PC) to ensure safe and atomic deletion of user data, removing all associated records and deregistering the user from any lobbies they participated in;

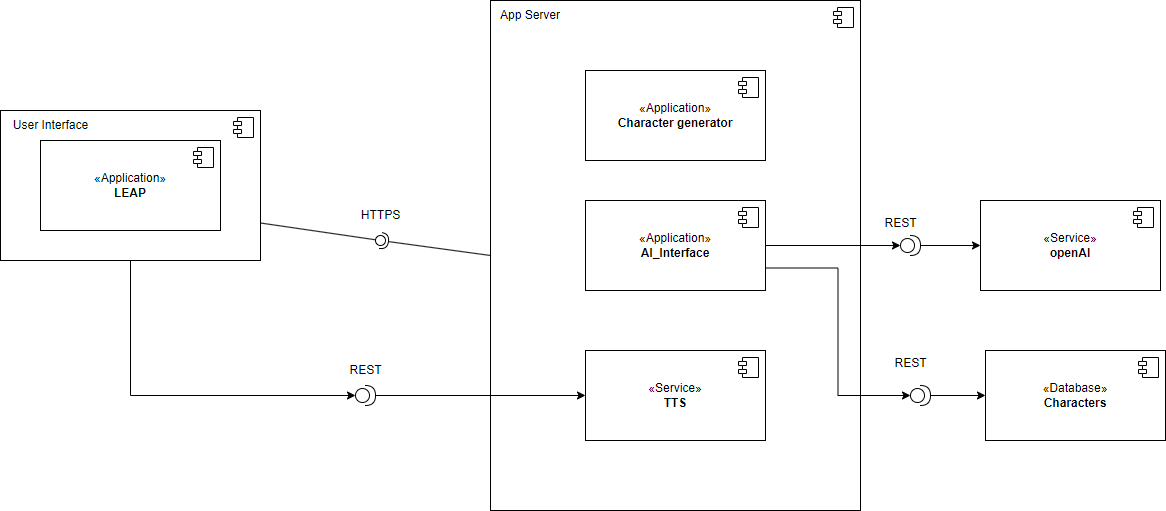
integrate a data warehouse to enable User data analysis like their rank, performance, log time and different user metrics;

deploy the ELK stack to centralize and process logs across all system instances, enhancing monitoring and troubleshooting capabilities;

deploy a Redis cluster to enable shared caching of Session data using consistent hashing.

Using such architecture will ensure that the system can withhold big traffic spikes as well as give the room to future development scaling as we can scale horizontally as needed by adding more processing power to run the models or even develop an in house version if needed.

In **figure 2.12** is represented a simple overview of the static components of the system:

  
Figure 2.12 - Simple overview

**Description:** in this diagram you can see a simpler view of the static components of the system and how they interact with each other.

2.2.3 Modeling the development tools environment

In the following deployment diagrams will be represented the development environment (the tools) that will host and run the platform and how the different layers of the system interact as well as the data flow between them:

In **figure .13** is shown an overview of the deployment process of the system

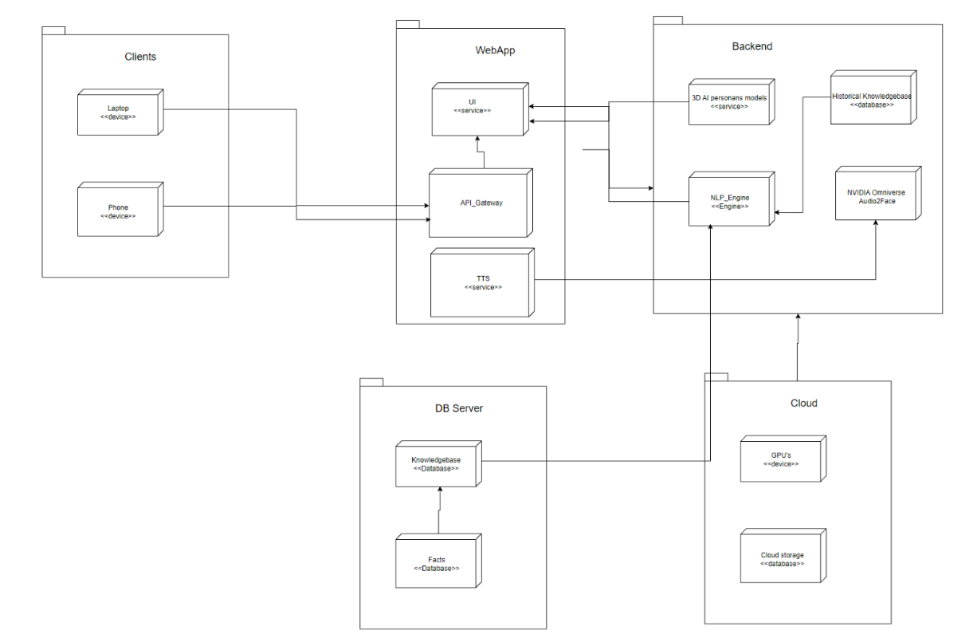


Figure 2.13 - General overview deployment diagram

**Description:** in this diagram you can see an overview of the structure of our app and how the different components will interact and be deployed, we will have a webapp and a mobile app interface where the users can interact with their personalized persona that is carefully created to their needs and accurately trained on its corresponding data set to emulate the desired character as well as a cloud service to store newly obtained data as well as external gpu computational power to render all those 3d models that we’ll have on the app.

In **figure 2.14** is shown the front end layer deployment elements and how they interact

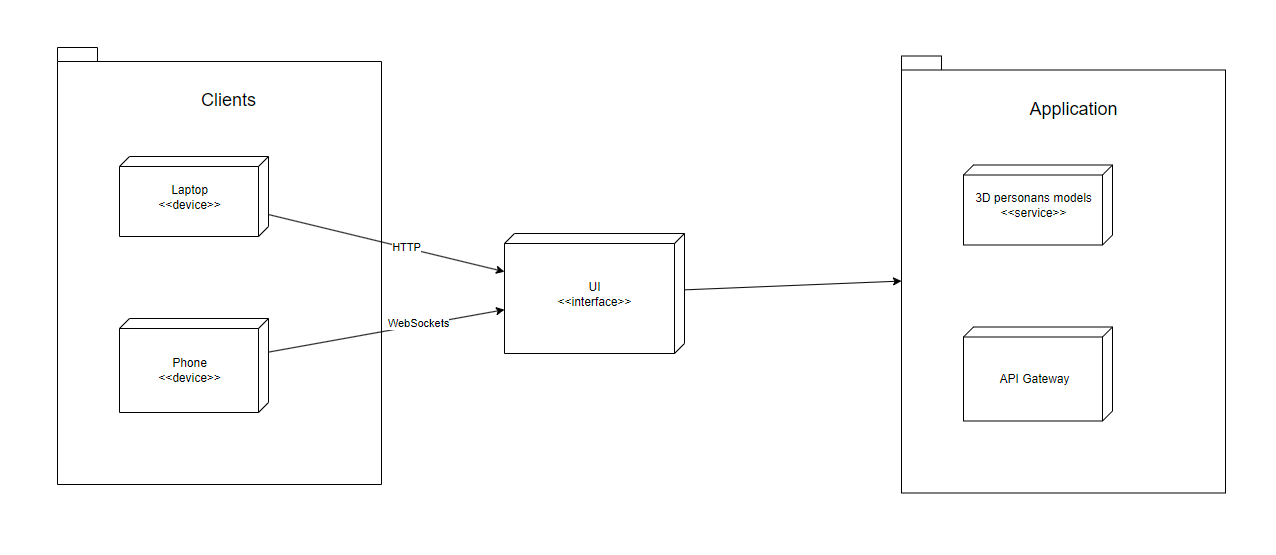
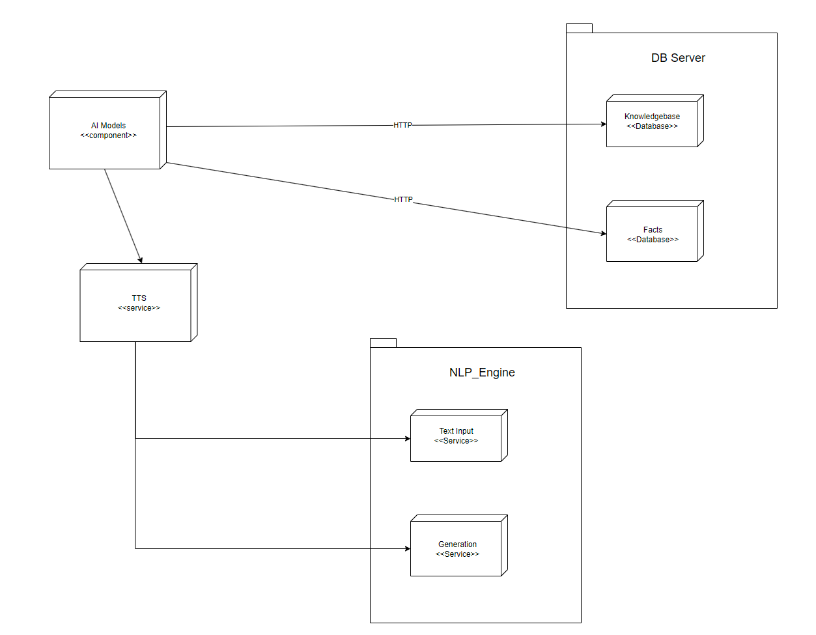


Figure 2.14 - Frontend Layer

**Description:** in the frontend layer we illustrate how the clients will interact with our app as we have a webapp for computer devices, and a mobile App for the mobile phones that will have our user interface through which the user can interact, choose or text with the personas.

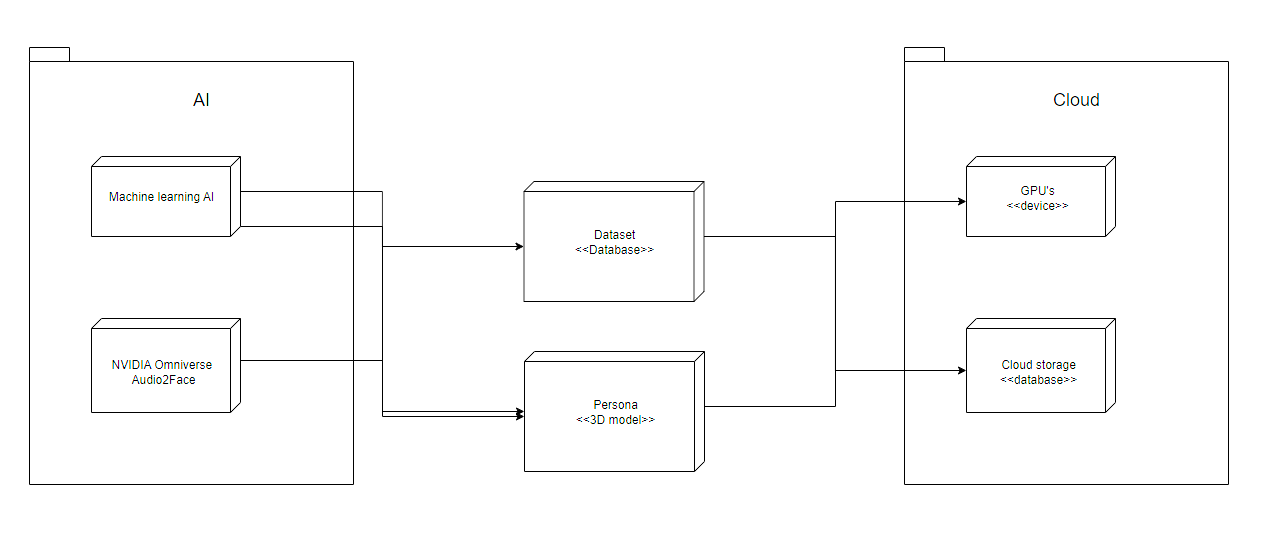
In **figure 2.15** is shown the back end layer deployment elements an how they interact:

  
Figure 2.15 - Backend Layer

**Description:** This diagram represents the backend structure of our project which consists of mainly 3 parts the AI 3d models, lip-sync technology and the NLP engine that will parse and generate responses for the user.

Next is the cloud services, the cloud services are services that are used instead of running the models locally as they can use too much process power on high traffic spicks so using cloud services ensures that the app is still running even with high traffic as the services and models will be run on big servers that are specialized with processing such big request quantities which assures stability for the platform and scalability although a bit costly but is worth it because of the security that such services provide.

In **figure 2.16** is represented the cloud service deployment and its connections to the systm:

Figure 2.16 - Cloud Services

**Description:** in this diagram you can see the additional cloud services that will be used in the project like the cloud storage that will hold all the data scrapped from any new News’s, discoveries etc... And the computational power of some GPUs that will help render the models.

# 3 SYSTEM Implementation

As for the LLM (Large Language Model) in this project was selected a local free version of **Ollama** that was run locally for ease of use, which can be substituted easily with an API key for OpenAI for example in the future for faster and more accurate responses.

As shown in **figure 3.1** the ollama LLM page and parameters from their original developer:

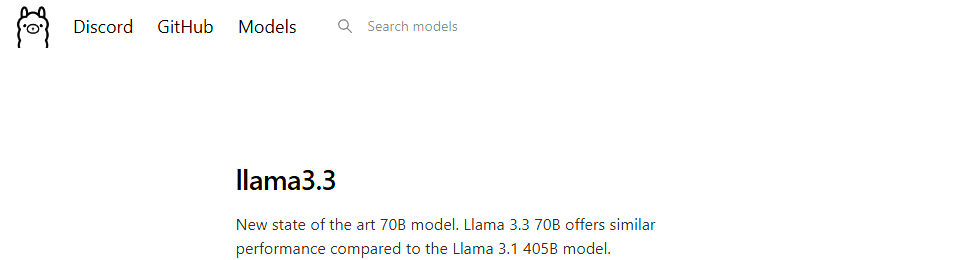


Figure 3.1 – Ollama LLM

The Prompt engineering and backend design was done in **nodeJS** for the app design as the knowledge base was mainly stored on a Victor database using **ChromaDB** as well as **Reddis** for the chat Caching as presented in **figure 3.2**.

Figure 3.2 – flask, Chroma DB

What is node.js?

Node.js is a powerful, open-source runtime environment that lets you run JavaScript code outside of a web browser. It was created to help developers build fast and scalable applications, especially for tasks that involve handling lots of input/output operations, like web servers, APIs, or real-time apps.

One of the things that makes Node.js so popular is its **non-blocking, event-driven** architecture. This means it can handle many tasks at the same time without slowing down. For example, if you're building a chat app, Node.js can manage thousands of messages being sent and received at once without crashing or freezing.

Node.js also has a huge collection of ready-made tools and libraries, thanks to its package manager called **npm (Node Package Manager)**. Developers can easily add features like file uploading, database connections, or real-time notifications by installing these packages, which saves a lot of time.

The Frontend was made using **HTML/CSS** and **JavaScript** for the interaction.

For Voice recognition and TTS azura was used.

What is azura?

Asure is a platform focused on AI**-**poweredvirtualavatars and human-computer interaction. It specializes in real-time lip-syncing, text-to-speech (TTS**)**, and voice recognition technologies, aiming to make digital avatars more realistic and interactive. It is commonly used in virtual meetings, digital twins, education, entertainment, and customer service.

3.1 Implementation Progress

In this section the implementation process, feature designs and overall design will be explained as well as the technologies used, researched and any third party websites, services or apps used in the system as well as the planned features, the structure of the project and the implemented part so far, as well as all the planned features and timeline for the overall implementation of them.

In the following sections will be represented the features implemented in detail and every aspect and part of the front end, animations and interactions in more detail as well as the backend, knowledge base and character design be it in 3d or the prompts/data/knowledgebase powering the personas as well as the API design the data storage and searching mechanisms as well as the service design and deployment and some unit tests and overall system calls and general QA.

3.1.1 Frontend

In this section will be presented the Frontend elements and their breakdown as well as the styling and overall design choices.

Because of the tight deadlines and the size of the project the frontend was simplified as the focus was directed towards finishing the core functionality first which led to the front end to be simplified.

For the front end was selected a 3D environment approach to mimic the real classroom through a virtual game like class in which the user will start in a disk faced by the persona the chalkboard and the responses on said chalkboard as you interact with the persona.

In **figure 3.3** is a presented the representation of the classroom in which the user should be able to interact, text and speak with the AI as well as choose the persona, control the conversation and classes

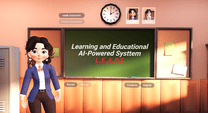


Figure 3.3 Frontend Design

As can be observed in the picture this is classroom in which the user starts which is a 3d environment with full 3d camera control which implementation is in **Anexa A.5** to insure gamification elements as well as emersion and retention.

In this project there would be a selection of 3 classes History, Art and language and each of the classes will have a corresponding classroom design as presented in the **figures 3.3, 3.4:**



Figure 3.1.4 History classroom and Art class

As presented in the previous pictures we aim to make the users experience as close to real schools as possible with the ability to change classes interact with the teachers, ask questions and control different aspects of the response it will create a safe, fun and interactive experience for the user ensuring higher retention rates and traffic.

Next lets breakdown each element of the UI:

First the **chat box**:

As presented in **figure 3.4 and Anexa A.4** the chat box is the main way to communicate with the AI through text and according to the class it will respond accordingly to the subject and selected persona, for this example the class will be in language and the AI will breakdown said sentence translate it and speak it out loud in Japanese.

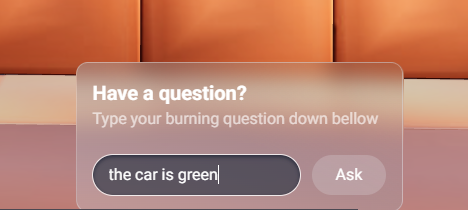


Figure 3.4 Chat box

**Description**: for the chat box was selected a simple clear design with a text interface that will stay stationary at the center bottom of the screen through which the user will be able to interact, ask different question in different subject to selected person in selected classrooms.

Next is the **chalkboard** in which as presented in **figure 3.5** and in Anexa **A.2 chalkboard design** the user will be greeted with the projects name waiting for the user to give a query to the AI?



Figure 3.5 the Chalkboard

After the user inputs a question or gives a statement the response will be displayed on the chalkboard as if the teacher wrote it there in the meanwhile the persona will be speaking the response out load with some simple animations to increase interactivity, as for this example the selected classroom will be languages and the AI will translate, breakdown and display any given sentence as presented in **figure 3.6**



Figure 3.6 Response example

In this example our query was the sentence “the car is green” and as can be seen the response is presented in the English and Japanese languages as well as a grammar breakdown of the sentence in the formal form as well as pronunciation if needed.

And in case of an error or a request failure an error message will be displayed on the board and said by the Ai as presented in the **figure 3.7**:

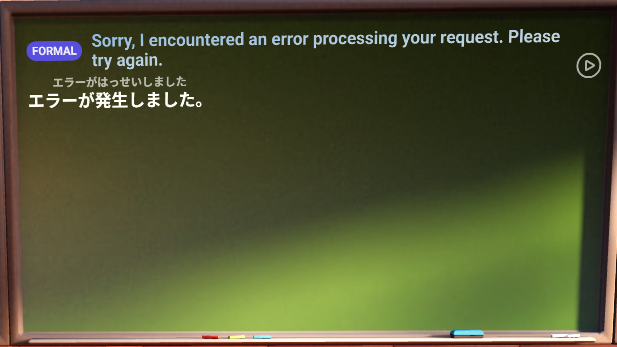


Figure 3.7 error handling

Next is the board control and interactions, the user will be able “in this example” to modify the said speech from formal to casual or he can change the sentence to be presented from furigana to English and vice versa as presented in **figure 3.8** the same sentence from the figure 3.1.5 but the furigana spelling breakdown is added:

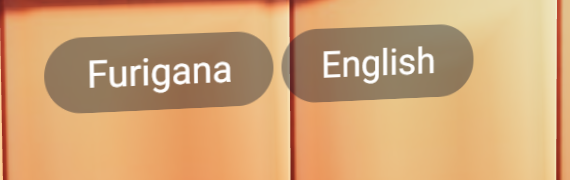


Figure 3.8 Furigana add

Using these buttons will show/hide each of the English translation or the furigana breakdown.

The user will be able to change classes as well which will change the way the persona responds according to the selected classroom.

The user will be able to choose persona that will respond to you which will change the style it responds the kind of questions and knowledge they have and interaction style as presented in the **figure 3.9**:

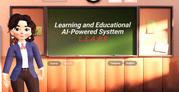
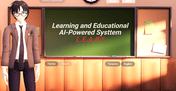




Figure 3.9 persona selection

As it can be seen from the previous pictures this creates a sense of freedom for the user to play around and test different combination and characters.

Lastly the 3D character, for the 3D character we have mainly 3 animations that loop in continue which are “thinking”, “idle” and “talking” which play when a certain action happens, when he responds its in the talking state, otherwise its in its idle state and before generating the responds its in the “thinking state”.

as shown in the **figure 3.10 and Anexa A.3**:



Figure 3.10 character animation states

3.1.2 LLM prompt Engineering

So far a simple system prompt was only given to the LLM to control its behavior as the main focus was to get the TTS and voice generation features working and here’s a sample of the response generation code:

try {  
 const chatCompletion = await axios.post(  
 "https://openrouter.ai/api/v1/chat/completions",  
 {  
 messages: [  
 {  
 role: "system",  
 content: `You are a Japanese language teacher.   
Your student asks you how to say something from english to japanese.  
You should respond with:   
- english: the english version ex: "Do you live in Japan?"

Code snippet 3.1 LLM Prompts

As can be observed from the code snippet that the AI acts according to the system prompt and after which every text is sent through the API key to generate the response within the predetermined limits.

3.1.3 TTS (Text-to-Speech)

For the text to speech azure was used which is a 3d party platform that specializes in generating audio for content creation, commentaries and AI assistants and we just send the text to through the API and run the response audio file using the custom voice created to match the virtual character.

*// Get audio data with visemes*const result = await new Promise((resolve, reject) => {  
 *console*.log('Starting speech synthesis...');  
 synthesizer.speakTextAsync(  
 text,  
 (result) => {  
 if (!result) {  
 reject(new Error("No result received from speech synthesis"));  
 return;  
 }  
 *console*.log('Speech synthesis completed, got audio data:', !!result.audioData);

Code snippet 3.2 TTS

This code generates the audio file with its visemes when we talk about “visemes,” we’re referring to the visual counterparts of phonemes—the basic sound units in language. While phonemes deal with the sounds we hear, visemes capture the way our mouths and faces look when we produce those sounds. Essentially, each viseme corresponds to a specific mouth shape or facial expression that accompanies a sound.

In case of an error or an undetected voice an error will be raised.

3.1.4 Lip-sync technology

So far in the project was implemented a simple animation sequence using CSS that animates a line to the voice to mimic a call and here’s the code for that animation:

playAudioWithLipSync(audioUrl) {  
*// Create audio waveform bars*const numBars = 20;  
for (let i = 0; i < numBars; i++) {  
 const bar = *document*.createElement('div');  
 bar.classList.add('waveform-bar');  
 lipSyncContainer.appendChild(bar);}

And the CSS container:

*/\* Audio Waveform Container \*/*#lip-sync-container {  
 width: 200px;  
 height: 60px;  
 background-color: #dee2e6;  
 border-radius: 10px;  
 display: flex;  
 justify-content: space-around;  
 align-items: center;  
 margin-top: 20px;  
 padding: 0 10px;  
 overflow: hidden;  
}

Code snippet 3.3 voice animations

3.1.5 Database design and connections

For the Knowledgebase of the characters was used a vector database called Chroma for ease of access and query as well for caching and maintaining on topic accuracy and history for the chat and here’s a sample of the implementation:

model = OllamaLLM(model="llama3.1:8b")  
embeddings = HuggingFaceEmbeddings(model\_name="sentence-transformers/all-MiniLM-L6-v2")  
prompt = ChatPromptTemplate.from\_template(template)  
chain = prompt | modeluser\_contexts = {}

async def handle\_message(update: Update, context: CallbackContext):  
 user\_id = update.message.chat\_id  
 user\_input = update.message.text  
 if user\_id not in user\_contexts:  
 user\_contexts[user\_id] = ""  
 *# Retrieve relevant company info* search\_results = vector\_store.similarity\_search(user\_input, k=1)  
 search\_results\_2 = pdf\_store.similarity\_search(user\_input, k=1)  
 retrieved\_info = search\_results[0].page\_content if search\_results else "No relevant company info found."  
 document\_info = "\n\n---\n\n".join([doc.page\_content for doc in search\_results\_2]) if search\_results\_2 else "No relevant document info found."

Code snippet 3.4 Database implementation

for Chaining huggingface was used to store the context, embedding’s and system prompt as for the recall similarity search was used to find context for a certain response if needed.

3.2 System Testing

As for testing this early in development only unit test and mainly manual testing throughout the development cycle was used.

The unit tests cover the majority of the functionalities created as well as every new feature added throughout the development ensuring quality and performance and here are some examples:

Requests test:

def test\_index\_route(self):

response = self.client.get('/')

self.assertEqual(response.status\_code, 200)

@patch('ollama.chat')

def test\_chat\_endpoint(self, mock\_ollama\_chat):

"""Test the chat endpoint with a mock response from Ollama."""

*# Mock the Ollama response*

mock\_ollama\_chat.return\_value = {

'message': {'content': 'A mock response from Einstein.'}}

*# Send a POST request to the chat endpoint*

response = self.client.post('/chat',

data=json.dumps({'message': 'What is relativity?'}),

content\_type='application/json')

*# Check the response*

data = json.loads(response.data)

self.assertEqual(response.status\_code, 200)

self.assertIn('response', data)

self.assertEqual(data['response'], 'A mock response from Einstein.')

mock\_ollama\_chat.assert\_called\_once()

Code Snippet 3.5 Request tests

Response generation:

In this snippet can be observed a section of the tests for the accuracy of responses and the transcription and temp file cleanup

*# Check the response*

data = json.loads(response.data)

self.assertEqual(response.status\_code, 200)

self.assertEqual(data['user\_message'], 'What is energy?')

self.assertEqual(data['einstein\_response'], 'Energy equals mass times the speed of light squared.')

self.assertEqual(data['voice\_path'], '/static/voice\_response.mp3')

*# Verify the transcription was called*

mock\_model.transcribe.assert\_called\_once\_with('/tmp/test\_audio.wav')

*# Verify the temp file was cleaned up*

mock\_unlink.assert\_called\_once\_with('/tmp/test\_audio.wav')

Code snippet 3.6 Response generation

Load tests:

In load test the performance of the server is tested and how big of a load it can tolerate so the test overloades the system with various number of requests tests its response limit and how many request can it handle without breaking to make sure the app runs accordingly although of the big number of requests as the model can reach its token limit fast.

In code snippet 3.7 is presented an example of the load test done on the system:

"""Stop the Flask server after tests."""

if hasattr(cls, 'flask\_process'):

# Send SIGTERM to the Flask process

cls.flask\_process.terminate()

# Give it some time to shut down gracefully

try:

cls.flask\_process.wait(timeout=5)

except subprocess.TimeoutExpired:

# If it doesn't shut down, force kill

cls.flask\_process.kill()

# Get output for debugging

stdout, stderr = cls.flask\_process.communicate()

if stderr:

print(f"Flask server stderr: {stderr.decode('utf-8')}")

Code snippet 3.7 Load tests

next are integration test, Integration tests are a type of software testing that checks how different pieces or modules of a system work together. Instead of testing parts of your code in isolation (like unit tests do), integration tests make sure that the components interact correctly when combined.

Integration test example:

*# Step 2: Convert the text to speech*

with patch('builtins.open', mock\_open()) as mock\_file:

tts\_response = self.client.post(

'/text-to-speech',

data=json.dumps({'text': chat\_data['response']}),

content\_type='application/json'

)

*# Check TTS response*

self.assertEqual(tts\_response.status\_code, 200)

tts\_data = json.loads(tts\_response.data)

self.assertIn('audio\_path', tts\_data)

self.assertEqual(tts\_data['audio\_path'], '/static/einstein\_response.mp3')

Code Snippet 3.8 Integration tests

Testing has played a crucial role in ensuring the LEAPS Teaching Platform is reliable, functional, and user-friendly. By implementing a combination of unit tests, integration tests, and user acceptance testing, we’ve been able to catch bugs early, confirm that different parts of the system work well together, and validate that the platform meets the needs of its users. This thorough approach not only improves the overall quality of the application but also builds confidence in its performance and stability as we continue to grow and add new features. Testing will remain an essential part of the development cycle to maintain the platform's integrity and support a seamless learning experience.

# 4 DOCUMENTATIONS

In this chapter will be explained the process of installing and running the application on your local machine as well as the documentation of each and every button and function.

4.1 Installation Guide

This guide will help you set up the platform and run it locally:

Prerequisites

NodeJS latest version;

Ollama (for AI responses);

Azure API key (optional, for high-quality voice synthesis).

Installation steps:

Install the requirements

pip install -r requirements.txt

**Setup environment variables** Create an .env file in the project root with the following variables:

Azure\_API\_KEY=your\_elevenlabs\_api\_key

EINSTEIN\_VOICE\_ID=your\_elevenlabs\_voice\_id

Note: The Azure API key is optional. If not provided, the app will fall back to browser-based text-to-speech.

**Set up Ollama** Install Ollama from ollama.ai and pull the necessary model:

ollama pull llama3.1:8b

Run the application

npm run dev

Troubleshooting:

Node version mismatch/issues, try pip install nodejs

Firewall access: insure that the app is an exception in the firewall in the windows defender settings

microphone Access: if you're having trouble with microphone access, ensure your browser has permission to use the microphone and that your microphone is properly connected.

4.2 How to use the platform

In this section will be demonstrated every interaction possible with the interface and explained what it does and how to use the app:

This section provides a complete guide on how to use the platform effectively. It covers everything from navigating the interface to interacting with avatars, and explains the function of every button, menu, and feature available in the app. Whether you're a first-time user or looking to explore advanced features, this documentation will walk you through all possible interactions, including voice input, text commands, avatar controls, and real-time customization tools. Use this section as your go-to reference to ensure a smooth and intuitive user experience.

First when you access the page you will be greeted with a loading screen in order to render all the 3d elements as presented in **figure 4.1:**

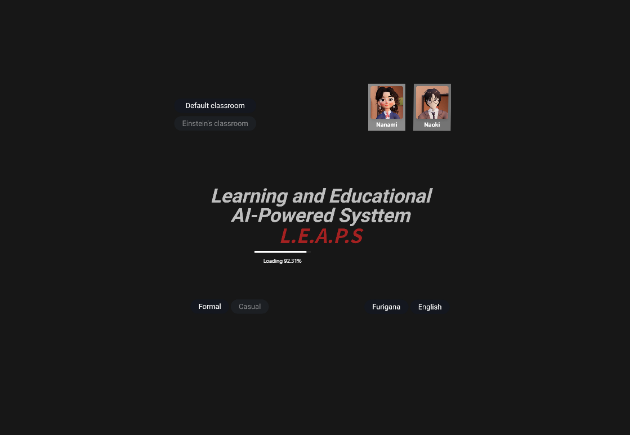


Figure 4.1 loading screen

After which you will be seated in your seat at the default classroom which will cover language as presented in **figure 4.2**



Figure 4.2 classroom

The user can zoom in and out in the class, look around, interact and ask the teacher through the chat box situated at the center bottom of the screen as presented in **figure 4.3**:

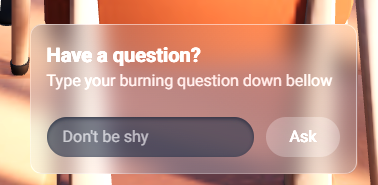


Figure 4.2 chat box

Through this box you can ask the teacher any question and according to the selected class and character the teacher will answer you in text form on the board and in audio directly to the user.

Next are interactions the user can change the classroom at any moment through the buttons situated above the left up corner of the chalkboard through which he will be sent to the according classroom with the teacher which in its way will change the way the ai responds and interacts like it’s presented in the **figurer 4.3**:

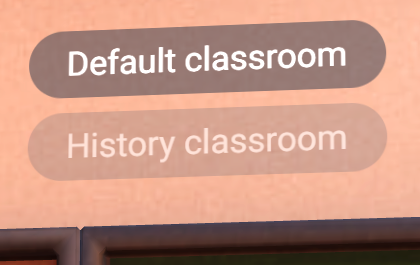
\

Figure 4.3 classroom choice

Next the user can choose which character to interact speak or learn from which will change the subject covered or the style in which the character speaks, functions and responds, these controls will be situated on the upper right corner of the chalkboard through which the user can choose as presented in **figure 4.4**:



Figure 4.4 Teacher selection

Next the user can control the form of the response and modify certain things, in this example in the language classroom he can modify the translation to be either formal or casual using the button situated under the lower left corner of the chalkboard like presented in the **figure 4.1.5** which will change the sentence entirely:

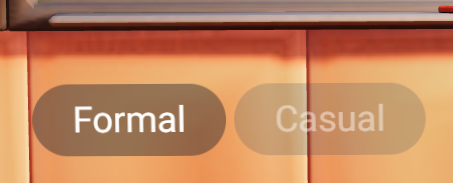


Figure 4.5 response format

Lastly the user will be able to add details and or hide them after the answer is generated in this case he will be able to show the English translation, show the furigana form of the text or hide it for learning purposes using the button under the lower right corner of the board as presented in the **figure 4.6**:

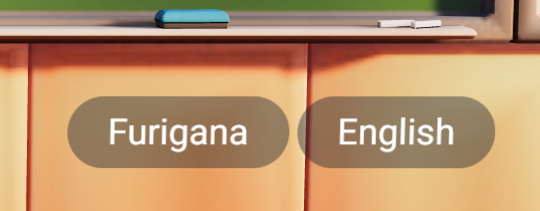


Figure 4.6 details addition

As for the responses they will be spoken by the teacher as well as printed on the chalkboard with the possibility for rereading if needed and detailed explanations of any certain word or phrase as presented in the **figure 4.7**:



Figure 4.7 the chalkboard

# 5 Cost estimation

In this chapter the estimation of the full scale project will be calculated and presented in detail as well as the financial breakdown of the maintenance costs as well as the recurring costs to run the system for a period of time, the solutions and replacements and options that can be chosen as well.

5.1 Core Components

Here is represented the core implementation and marketing cost breakdown:

5.1.1 AI Infrastructure

LLM API Usage:

* 1. $1,000 – $10,000/month (depending on usage volume and model complexity);
  2. option: self-hosting via Ollama can reduce cost but increases hardware and maintenance requirements.

In the demo of this project a locally run ollama LLM version was run to minimize the costs.

Text-to-Speech & Lip Sync (ElevenLabs + NVIDIA Audio2Face):

* 1. TTS API: $100 – $1,000/month
  2. audio2Face: free (GPU required, integration labor not included)

voice input recognition (Whisper, Google Speech):

* 1. $0.006 – $0.02/minute;
  2. self-hosted: free.

5.1.2 3D Models & Animation

3D Model Creation (custom avatars/personas):

* 1. $1,000 – $5,000 per character;
  2. rigging & facial blendshapes: $500 – $1,500 per character.

Lip Sync Integration (with Unity/Unreal + Audio2Face/Ready Player Me):

* 1. Development: $5,000 – $15,000;
  2. Animation software licenses (if needed): $500 – $2,000.

Educational Platform Development:

5.1.3 Educational Platform Development

Frontend & Backend Development (Web/App):

* 1. $15,000 – $40,000 (MVP);
  2. Features: user accounts, chat interface, quiz module, analytics, etc..

AI Feature Development (Quiz generation, student feedback, advice system):

* 1. $5,000 – $20,000 depending on depth.

Storage, database, traffic (e.g., AWS, GCP):

* 1. $200 – $1,000/month.

5.2 Estimated Budget Tiers

Here is represented the estimation of 3 tiers of the application according to the level of complexity and hosting of the elements of the project.

Basic MVP: Basic features, few AI characters, cloud APIs…..$35K-$60K;

Standard: Multiple characters, LLM APIs, real-time lip sync, full quiz module…..$60K-$150K;

Full-Scale: Self-hosted LLMs, full animation, real-time avatar interaction, advanced analytics…$150K-$300K.

5.3 Optional/Recurring Costs

**maintenance & support**: $10K – $50K/year;

**content development (Lessons, quiz templates)**: $5K – $20K/year;

**licensing (Unity/Unreal/Assets)**: up to $5K/year;

**marketing & deployment**: $10K – $30K.

# Conclusions

In conclusion it can be observed that there exists a market and an interest for the solution as the students and studies conducted show definitive results that the old/normal educational system is failing and that there’s a rising need for new tools and solution to help people cope with the rapidly expanding technologies and online studies as well as the core functional and non-functional requirements that should be in the project.

After designing and presenting the various use cases, scenarios and component relations of the system we can say that there’s a clear and visible way to implement and make the solution without any confusions or mistakes.

We can monitor the progress of the project implementation and development step by step ensuring maintaining communication throughout the whole process.

The project successfully demonstrates the potential of artificial intelligence to revolutionize the educational landscape by bridging the gap between historical knowledge and modern learning needs. Through the development of an AI-powered platform that brings historical figures to life, this work offers an engaging, interactive, and personalized learning experience for students, teachers, and researchers alike. By simulating conversations with figures such as Einstein or Tesla, the system enables users to gain insights into complex topics in a more humanized and intuitive way.

The research and survey analysis confirmed a strong market demand and educational need for such innovative tools. Students expressed a clear preference for personalized, interactive, and subject-focused experiences, while teachers and institutions saw value in the potential to improve engagement, performance tracking, and curriculum support. Furthermore, the platform was designed with cost efficiency and scalability in mind, using microservice architecture and leveraging modern technologies such as Ollama LLM, Azura for TTS, and ChromaDB for intelligent knowledge retrieval.

From domain analysis and system modeling to implementation and testing, each phase was conducted with attention to functionality, performance, and user experience. The use of modern frameworks, 3D modeling, and prompt engineering ensured that the personas are not only informative but contextually aware and historically accurate.

In conclusion, the platform is more than a technical solution it is a forward-thinking educational innovation. It addresses the shortcomings of traditional learning methods and sets the stage for more immersive, intelligent, and accessible education. With further development and investment, it holds the potential to transform how history is taught and how students around the world engage with knowledge. The project lays a strong foundation for future enhancements and broader deployment, marking a significant step toward AI-integrated education.

# BIBLIOGRAFIE

[1] “Speech-to-Text documentation | Cloud Speech-to-Text Documentation,” Google Cloud. Available: https://cloud.google.com/speech-to-text/docs. [Accessed: Apr. 29, 2025]

[2] “Documentation.” Available: https://www.ffmpeg.org/documentation.html. [Accessed: Apr. 29, 2025]

[3] “FFmpeg: Main Page.” Available: https://www.ffmpeg.org/doxygen/trunk/index.html. [Accessed: Apr. 29, 2025]

[4] “Introduction | ElevenLabs Documentation.” Available: https://elevenlabs.io/docs/api-reference/introduction. [Accessed: Apr. 29, 2025]

[5] “Python 3.13 documentation,” Python documentation. Available: https://docs.python.org/3/. [Accessed: Apr. 29, 2025]

[6] “Welcome to Flask — Flask Documentation (3.1.x).” Available: https://flask.palletsprojects.com/en/stable/. [Accessed: Apr. 29, 2025]

[7] “llama3.3.” Available: https://ollama.com/llama3.3. [Accessed: Apr. 29, 2025]

[8] “CSS: Cascading Style Sheets | MDN,” Mar. 22, 2025. Available: https://developer.mozilla.org/en-US/docs/Web/CSS. [Accessed: Apr. 29, 2025]

[9] M. Tanner, “The comprehensive guide to documenting microservices,” vFunction, Jan. 23, 2025. Available: https://vfunction.com/blog/guide-on-documenting-microservices/. [Accessed: Apr. 29, 2025]

[10] Rick-Anderson, “ASP.NET MVC.” Available: https://learn.microsoft.com/en-us/aspnet/mvc/. [Accessed: Apr. 29, 2025]

[11] “JavaScript | MDN,” Apr. 03, 2025. Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript. [Accessed: Apr. 29, 2025]

[12] “- YouTube.” Available: https://www.youtube.com/watch?v=w5unVTO7mLQ. [Accessed: Apr. 29, 2025]

[13] “Chroma Docs,” Chroma Docs. Available: https://docs.trychroma.com. [Accessed: Apr. 29, 2025]

[14] “Azure AI Speech | Microsoft Azure.” Available: https://azure.microsoft.com/en-us/products/ai-services/ai-speech. [Accessed: May 17, 2025].

# ANEXA A

Survey Questionnaire A.1

What is your role?

Student

Educator

Parent/Guardian

Educational Administrator

What is your level of familiarity with AI technology?

Not Familiar

1

2

3

4

5

Very Familiar

Which subject areas are you most interested in (select all that apply)?

Science

Math

History

Art

Technology

Other...

What type of interactions would you prefer with these virtual personas? (Select all that apply)

Conversational Q&A

Lectures and Presentations

Problem solving Asistant

Debates on Various Topics

Creative Workshops (e.g., brainstorming ideas)

Personalized Learning Path Recommendations...

Which historical figure or expert would you most like to see as a virtual persona? Why?

Answer ...

How important is it for these personas to reflect the authentic personality and style of the original figures?

Extremely Important

Somewhat Important

Not Important

Would you prefer virtual personas to be more realistic (closer to human behavior) or more abstract (emphasizing knowledge and teaching content over realism)?

Highly Realistic

Balanced (Combination of Realism and Content)

Focused on Content, Less Realistic

What concerns do you have about the use of AI in teaching and learning environments? (Select all that apply)

Privacy and Data Security

Potential Bias in AI Responses

Over-reliance on Technology

Authenticity and Accuracy of Information

Other...

Would you feel comfortable with a virtual persona having access to personal learning data to improve recommendations and support?

Yes, I’m Comfortable

Comfortable, but with Limitations

Not Comfortable

How should the creators of these personas ensure they remain ethical and unbiased in their teachings? (e.g., Regular Monitoring, Community Feedback, etc.)

Answer...

What would make you want to use or recommend AI teaching personas?

Answer...

Is there anything else you’d like to share about this concept?

Answer...

Chalkboard design A.2

import { *teachers*, *useAITeacher* } from "@/hooks/useAITeacher";  
  
export const BoardSettings = () => {  
 const furigana = *useAITeacher*((state) => state.furigana);  
 const setFurigana = *useAITeacher*((state) => state.setFurigana);  
  
 const english = *useAITeacher*((state) => state.english);  
 const setEnglish = *useAITeacher*((state) => state.setEnglish);  
  
 const teacher = *useAITeacher*((state) => state.teacher);  
 const setTeacher = *useAITeacher*((state) => state.setTeacher);  
  
 const speech = *useAITeacher*((state) => state.speech);  
 const setSpeech = *useAITeacher*((state) => state.setSpeech);  
  
 const classroom = *useAITeacher*((state) => state.classroom);  
 const setClassroom = *useAITeacher*((state) => state.setClassroom);  
  
 return (  
 <>  
 <div className="absolute right-0 bottom-full flex flex-row gap-10 mb-20">  
 {*teachers*.map((sensei, idx) => (  
 <div  
 key={idx}  
 className={`p-3 transition-colors duration-500 ${  
 teacher === sensei ? "bg-white/50" : "bg-white/40"  
 }`}  
 >  
 <div onClick={() => setTeacher(sensei)}>  
 <img  
 src={`/teachers/${sensei}.jpg`}  
 alt={sensei}  
 className="object-cover w-40 h-40 rounded-lg shadow-lg hover:scale-105 transition-transform duration-300"  
 />  
 </div>  
 <h2 className="text-3xl font-bold mt-3 text-center text-white">{sensei}</h2>  
 </div>  
 ))}  
 </div>  
 <div className="absolute left-0 bottom-full flex flex-col gap-4 mb-20 -ml-20">  
 <button  
 className={` ${  
 classroom === "default"  
 ? "text-white bg-slate-900/40 "  
 : "text-white/45 bg-slate-700/20 "  
 } py-4 px-10 text-4xl rounded-full transition-colors duration-500 backdrop-blur-md hover:bg-slate-800/50`}  
 onClick={() => setClassroom("default")}  
 >  
 **Default classroom** </button>  
 <button  
 className={` ${  
 classroom === "history"  
 ? "text-white bg-slate-900/40 "  
 : "text-white/45 bg-slate-700/20 "  
 } py-4 px-10 text-4xl rounded-full transition-colors duration-500 backdrop-blur-md hover:bg-slate-800/50`}  
 onClick={() => setClassroom("history")}  
 >  
 **History classroom** </button>  
 </div>  
 <div className="absolute left-0 top-full flex flex-row gap-2 mt-20">  
 <button  
 className={` ${  
 speech === "formal"  
 ? "text-white bg-slate-900/40 "  
 : "text-white/45 bg-slate-700/20 "  
 } py-4 px-10 text-4xl rounded-full transition-colors duration-500 backdrop-blur-md hover:bg-slate-800/50`}  
 onClick={() => setSpeech("formal")}  
 >  
 **Formal** </button>  
 <button  
 className={` ${  
 speech === "casual"  
 ? "text-white bg-slate-900/40 "  
 : "text-white/45 bg-slate-700/20 "  
 } py-4 px-10 text-4xl rounded-full transition-colors duration-500 backdrop-blur-md hover:bg-slate-800/50`}  
 onClick={() => setSpeech("casual")}  
 >  
 **Casual** </button>  
 </div>  
 <div className="absolute right-0 top-full flex flex-row gap-2 mt-20">  
 <button  
 className={` ${  
 furigana  
 ? "text-white bg-slate-900/40 "  
 : "text-white/45 bg-slate-700/20 "  
 } py-4 px-10 text-4xl rounded-full transition-colors duration-500 backdrop-blur-md hover:bg-slate-800/50`}  
 onClick={() => setFurigana(!furigana)}  
 >  
 **Furigana** </button>  
 <button  
 className={`${  
 english  
 ? "text-white bg-slate-900/40 "  
 : "text-white/45 bg-slate-700/20 "  
 } py-4 px-10 text-4xl rounded-full transition-colors duration-500 backdrop-blur-md hover:bg-slate-800/50`}  
 onClick={() => setEnglish(!english)}  
 >  
 **English** </button>  
 </div>  
 </>  
 );  
};

Teacher design and animations A.3

import { *teachers*, *useAITeacher* } from "@/hooks/useAITeacher";  
import { Html, useAnimations, useGLTF } from "@react-three/drei";  
import { useFrame } from "@react-three/fiber";  
import { useEffect, useRef, useState } from "react";  
import { *MathUtils*, MeshStandardMaterial } from "three";  
import { randInt } from "three/src/math/MathUtils";  
  
const ANIMATION\_FADE\_TIME = 0.5;  
  
export function Teacher({ teacher, ...props }) {  
 const group = useRef();  
 const { scene } = useGLTF(`/models/Teacher\_${teacher}.glb`);  
 useEffect(() => {  
 scene.traverse((child) => {  
 if (child.material) {  
 child.material = new MeshStandardMaterial({  
 map: child.material.map,  
 });  
 }  
 });  
 }, [scene]);  
  
 *// Load animations FIRST* const { animations } = useGLTF(`/models/animations\_${teacher}.glb`);  
 *// THEN use them* const { actions, mixer } = useAnimations(animations, group);  
 const [animation, setAnimation] = useState("Idle");  
  
 *// Read state needed for lip sync AND loading state* const loading = *useAITeacher*((state) => state.loading);  
 const currentMessageIndex = *useAITeacher*((state) => state.currentMessage);  
 const playingAudio = *useAITeacher*((state) => state.playingAudio);  
 const playingVisemes = *useAITeacher*((state) => state.playingVisemes);  
  
 *// Imported from r3f-virtual-girlfriend project* const [blink, setBlink] = useState(false);  
  
 useEffect(() => {  
 let blinkTimeout;  
 const nextBlink = () => {  
 blinkTimeout = setTimeout(() => {  
 setBlink(true);  
 setTimeout(() => {  
 setBlink(false);  
 nextBlink();  
 }, 100);  
 }, randInt(1000, 5000));  
 };  
 nextBlink();  
 return () => clearTimeout(blinkTimeout);  
 }, []);  
  
 useEffect(() => {  
 if (loading) {  
 setAnimation("Thinking");  
 } else if (currentMessageIndex !== null && playingAudio) {  
 *// Check if audio is playing for Talking animation* setAnimation(randInt(0, 1) ? "Talking" : "Talking2");  
 } else {  
 setAnimation("Idle");  
 }  
 }, [loading, currentMessageIndex, playingAudio]);  
  
 useFrame(({ camera }) => {  
 *// Smile* lerpMorphTarget("mouthSmile", 0.2, 0.5);  
 *// Blinking* lerpMorphTarget("eye\_close", blink ? 1 : 0, 0.5);  
  
 *// Talking / Lip Sync* for (let i = 0; i <= 21; i++) {  
 lerpMorphTarget(i, 0, 0.1); *// reset morph targets* }  
  
 if (  
 currentMessageIndex !== null &&  
 playingVisemes?.length > 0 &&  
 playingAudio  
 ) {  
 for (let i = playingVisemes.length - 1; i >= 0; i--) {  
 const viseme = playingVisemes[i];  
 if (playingAudio.currentTime \* 1000 >= viseme[0]) {  
 lerpMorphTarget(viseme[1], 1, 0.2);  
 break;  
 }  
 }  
 if (  
 actions[animation] &&  
 actions[animation].time >  
 actions[animation].getClip().duration - ANIMATION\_FADE\_TIME  
 ) {  
 setAnimation((animation) =>  
 animation === "Talking" ? "Talking2" : "Talking"  
 );  
 }  
 }  
 });  
  
 useEffect(() => {  
 actions[animation]  
 ?.reset()  
 .fadeIn(mixer.time > 0 ? ANIMATION\_FADE\_TIME : 0)  
 .play();  
 return () => {  
 actions[animation]?.fadeOut(ANIMATION\_FADE\_TIME);  
 };  
 }, [animation, actions]);  
  
 const lerpMorphTarget = (target, value, speed = 0.1) => {  
 scene.traverse((child) => {  
 if (child.isSkinnedMesh && child.morphTargetInfluences) {  
 if (target >= 0 && target < child.morphTargetInfluences.length) {  
 child.morphTargetInfluences[target] = *MathUtils*.lerp(  
 child.morphTargetInfluences[target],  
 value,  
 speed  
 );  
 }  
 }  
 });  
 };  
  
 const [thinkingText, setThinkingText] = useState(".");  
  
 useEffect(() => {  
 if (loading) { *// Use loading state here as well for the thinking indicator text* const interval = setInterval(() => {  
 setThinkingText((thinkingText) => {  
 if (thinkingText.length === 3) {  
 return ".";  
 }  
 return thinkingText + ".";  
 });  
 }, 500);  
 return () => clearInterval(interval);  
 }  
 }, [loading]); *// Use loading dependency* return (  
 <group {...props} dispose={null} ref={group}>  
 {loading && ( *// Show thinking indicator only when loading* <Html position-y={teacher === "Nanami" ? 1.6 : 1.8}>  
 <div className="flex justify-center items-center -translate-x-1/2">  
 <span className="relative flex h-8 w-8 items-center justify-center">  
 <span className="animate-ping absolute inline-flex h-full w-full rounded-full bg-white opacity-75"></span>  
 <span className="relative inline-flex items-center justify-center duration-75 rounded-full h-8 w-8 bg-white/80">  
 {thinkingText}  
 </span>  
 </span>  
 </div>  
 </Html>  
 )}  
 <primitive object={scene} />  
 </group>  
 );  
}  
  
*teachers*.forEach((teacher) => {  
 useGLTF.preload(`/models/Teacher\_${teacher}.glb`);  
 useGLTF.preload(`/models/animations\_${teacher}.glb`);  
});

Chat box design A.4

import { *useAITeacher* } from "@/hooks/useAITeacher";  
import { useState } from "react";  
  
export const TypingBox = () => {  
 const askAI = *useAITeacher*((state) => state.askAI);  
 const loading = *useAITeacher*((state) => state.loading);  
 const [question, setQuestion] = useState("");  
  
 const ask = () => {  
 askAI(question);  
 setQuestion("");  
 };  
 return (  
 <div className="z-10 max-w-[600px] flex space-y-6 flex-col bg-gradient-to-tr from-slate-300/30 via-gray-400/30 to-slate-600-400/30 p-4 backdrop-blur-md rounded-xl border-slate-100/30 border">  
 <div>  
 <h2 className="text-white font-bold text-xl">  
 **Have a question?** </h2>  
 <p className="text-white/65">  
 **Type your burning question down bellow** </p>  
 </div>  
  
 {loading ? (  
 <div className="flex justify-center items-center">  
 <span className="relative flex h-4 w-4">  
 <span className="animate-ping absolute inline-flex h-full w-full rounded-full bg-white opacity-75"></span>  
 <span className="relative inline-flex rounded-full h-4 w-4 bg-white"></span>  
 </span>  
 </div>  
 ) : (  
 <div className="gap-3 flex">  
 <input  
 className="focus:outline focus:outline-white/80 flex-grow bg-slate-800/60 p-2 px-4 rounded-full text-white placeholder:text-white/50 shadow-inner shadow-slate-900/60"  
 placeholder="Don't be shy"  
 value={question}  
 onChange={(e) => setQuestion(e.target.value)}  
 onKeyDown={(e) => {  
 if (e.key === "Enter") {  
 ask();  
 }  
 }}  
 />  
 <button  
 className="bg-slate-100/20 p-2 px-6 rounded-full text-white"  
 onClick={ask}  
 >  
 **Ask** </button>  
 </div>  
 )}  
 </div>  
 );  
};

Camera control and interactions A.5

"use client";  
import { *useAITeacher* } from "@/hooks/useAITeacher";  
import {  
 CameraControls,  
 Environment,  
 Float,  
 Gltf,  
 Html,  
 Loader,  
 useGLTF,  
} from "@react-three/drei";  
import { Canvas } from "@react-three/fiber";  
import { Leva, button, useControls } from "leva";  
import { Suspense, useEffect, useRef } from "react";  
import { degToRad } from "three/src/math/MathUtils";  
import { BoardSettings } from "./BoardSettings";  
import { MessagesList } from "./MessagesList";  
import { Teacher } from "./Teacher";  
import { TypingBox } from "./TypingBox";  
  
const itemPlacement = {  
 default: {  
 classroom: {  
 position: [0.2, -1.7, -2],  
 },  
 teacher: {  
 position: [-1, -1.7, -3],  
 },  
 board: {  
 position: [0.45, 0.382, -6],  
 },  
 },  
 history: {  
 classroom: {  
 position: [0.3, -10, 1],  
 rotation: [0, degToRad(90), 0],  
 scale: 0.4,  
 },  
 teacher: { position: [-0.3, -1.5, -5], scale: 1.2 },  
 board: { position: [2.5, 0, -8] },  
  
 },  
};  
  
export const Experience = () => {  
 const teacher = *useAITeacher*((state) => state.teacher);  
 const classroom = *useAITeacher*((state) => state.classroom);  
  
 return (  
 <>  
 <div className="z-10 md:justify-center fixed bottom-4 left-4 right-4 flex gap-3 flex-wrap justify-stretch">  
 <TypingBox />  
 </div>  
 <Leva hidden />  
 <Loader />  
 <Canvas  
 camera={{  
 position: [0, 0, 0.0001],  
 }}  
 >  
 <CameraManager />  
  
 <Suspense>  
 <Float speed={0.5} floatIntensity={0.2} rotationIntensity={0.1}>  
 <Html  
 transform  
 {...itemPlacement[classroom].board}  
 distanceFactor={1}  
 >  
 <MessagesList />  
 <BoardSettings />  
 </Html>  
 <Environment preset="sunset" />  
 <ambientLight intensity={0.8} color="pink" />  
  
 <Gltf  
 src={`/models/classroom\_${classroom}.glb`}  
 {...itemPlacement[classroom].classroom}  
 />  
 <Teacher  
 teacher={teacher}  
 key={teacher}  
 {...itemPlacement[classroom].teacher}  
 scale={1.5}  
 rotation-y={degToRad(20)}  
 />  
 </Float>  
 </Suspense>  
 </Canvas>  
 </>  
 );  
};  
  
const CAMERA\_POSITIONS = {  
 default: [0, 6.123233995736766e-21, 0.0001],  
 loading: [  
 0.00002621880610890309, 0.00000515037441056466, 0.00009636414192870058,  
 ],  
 speaking: [0, -1.6481333940859815e-7, 0.00009999846226827279],  
};  
  
const CAMERA\_ZOOMS = {  
 default: 1,  
 loading: 1.3,  
 speaking: 2.1204819420055387,  
};  
  
const CameraManager = () => {  
 const controls = useRef();  
 const loading = *useAITeacher*((state) => state.loading);  
 const currentMessage = *useAITeacher*((state) => state.currentMessage);  
  
 useEffect(() => {  
 if (loading) {  
 controls.current?.setPosition(...CAMERA\_POSITIONS.loading, true);  
 controls.current?.zoomTo(CAMERA\_ZOOMS.loading, true);  
 } else if (currentMessage) {  
 controls.current?.setPosition(...CAMERA\_POSITIONS.speaking, true);  
 controls.current?.zoomTo(CAMERA\_ZOOMS.speaking, true);  
 }  
 }, [loading]);  
  
 useControls("Helper", {  
 getCameraPosition: button(() => {  
 const position = controls.current.getPosition();  
 const zoom = controls.current.camera.zoom;  
 *console*.log([...position], zoom);  
 return (  
 <CameraControls  
 ref={controls}  
 minZoom={1}  
 maxZoom={3}  
 polarRotateSpeed={-0.3} *// REVERSE FOR NATURAL EFFECT* azimuthRotateSpeed={-0.3} *// REVERSE FOR NATURAL EFFECT* mouseButtons={{  
 left: 1, *//ACTION.ROTATE* wheel: 16, *//ACTION.ZOOM* }}  
 touches={{  
 one: 32, *//ACTION.TOUCH\_ROTATE* two: 512, *//ACTION.TOUCH\_ZOOM*  
useGLTF.preload("/models/classroom\_default.glb");  
useGLTF.preload("/models/classroom\_alternative.glb");  
useGLTF.preload("/models/classroom\_history.glb")