**Title of Thesis: Virtual AI Assistant for Educational Institutions**

**Abstract:**

The rapid advancement of artificial intelligence (AI) tempers with new ways to enhance pedagogical interactions. This thesis proposes an AI-Virtual Assistant for educational environment creation and utilization. The major objectives behind it are to raise the overall efficiency of schools and colleges, increase student involvement and reduce administrative workload.

Consequently, the demand for digital transformation in education has been growing based on the requirement for improved learning experiences and efficient management, which provide the background of this study. This assistant will help higher education providers in being able to monitor attendance, manage schedules and notify students in a much timelier manner in what the university are hoping will lead to their students feeling safer on campus.

The methodology includes several technologies: facial recognition using OpenCV and geolocation in order to track absence automatically, indoor navigator with Unity and ARCore, timetable management. Also, a PyTorch-based chatbot interface that can answer all kinds of questions regarding academics, campus facilities etc.

Therefore, the proposed system, Educational Assistant, alleviates the problem of data entry and also eliminates cases of delayed arrival of students while offering efficient tracking of attendance records effectively. In the numbers there is a clear representation of better and increased satisfaction and affection of the students as well as carrying a preamble that AI can transform the educational management.

**Keywords:** Artificial Intelligence**,** Educational Assistant**,** Student Involvement**,** Timetable Management**,** Indoor Navigation**,** Absence Tracking**,** Face Recognition**,** Chatbot Interface**,** Digital Transformation**,** Campus Management, OpenCV, PyTorch, ARCore, Unity.

# Table of Contents

1. Introduction...................................................................................................................1
   1. Background..............................................................................................................1
   2. Objectives and Scope...............................................................................................1
   3. Significance ………………………………………………………………….........2
2. Literature Review...........................................................................................................3

2.1 Related Work............................................................................................................3 2.2 Technologies Used...................................................................................................4

1. System Architecture and Design....................................................................................9

3.1 Overall System Architecture................................................................................... 9 3.2 Timetable Integration.............................................................................................10 3.3 Indoor Navigation and Wayfinding........................................................................12 3.4 Automated Absence Tracking................................................................................13 3.5 Chatbot Interface....................................................................................................16

1. Implementation of the Program....................................................................................18

4.1 Development Tools and Environment....................................................................18 4.2 Implementing Timetable Integration......................................................................18 4.3 Building Indoor Navigation and Wayfinding.........................................................21 4.4 Automating Absence Tracking...............................................................................25 4.5 Creating the Chatbot Interface................................................................................28

1. Challenges and Future Work........................................................................................36

5.1 Development Challenges………............................................................................36 5.2 Lessons Learned………….....................................................................................38

1. Conclusion....................................................................................................................40

6.1 Conclusion..............................................................................................................40 6.2 Future Work..………..……………………………………………………………41 6.3 Final Thoughts........................................................................................................42

1. References....................................................................................................................43
2. Appendix…………………………………………………………………….……….45

List of Figures

Figure 3.1 ………………………………………………………………………………12

Figure 3.2 ………………………………………………………………………………13

Figure 3.3 ………………………………………………………………………………16

Figure 3.4 ………………………………………………………………………………18

Figure 4.1 ………………………………………………………………………………23

Figure 4.2 ………………………………………………………………………………32

Figure 4.3 ………………………………………………………………………………36

Figure 4.4 ………………………………………………………………………………37

List of Appendices

Appendix A…………………………………………………………………………………...45

Appendix B…………………………………………………………………………………...48

Appendix C…………………………………………………………………………………...53

**Introduction**

* 1. Background

There are significant changes in educational management due to new technologies in learning delivery and the fact that there is need to have efficient management systems in place. The key drivers that are driving this “Perceived Pressure for Change” include the following: Inadequate and inefficient management of student’s schedules, large campus structures and systems, and attendance systems. They are usually tedious, labor intensive with a huge chance of making errors and a lot of administrative work that put pressure on educational institutions.  
  
In light of these challenges, Artificial Intelligence (AI) holds a realm of value in this work. To this extent, the use of AI in education has the potential to facilitate a number of procedures, increase the extent of related precision, and offer customized approaches to the learners. This project is centered towards creating an AI Virtual Assistant that will change the way students schedule their programs and navigating through infrastructural facilities as well as keep record of attendant students.

1.2 Objectives and Scope

The primary objectives and scope of this project include:

Timetable Integration:

* Read the information from the timetable such as the program of study, day, lesson time and course.
* Send alerts to the students especially when there are lessons to be taught, the subject to be taught, the room and the time the lesson is to commence.

Indoor Navigation and Wayfinding:

* Develop an indoor navigation system using Unity and AR Core.
* Guide and support the students in navigating in the campus so that they can get to the right places on time.

Automated Absence Tracking:

* Geolocation services should be used to confirm students’ presence in the classrooms.
* Use face recognition technology to authenticate students.
* Make certain that only students in the classroom are treated as “present.”

Chatbot Interface:

* The chatbot should be trained by all the rules and policies of the CIT University.
* Make it easier for students to ask questions related to internships, Erasmus programs, friendship, love etc.
* Chatbot must support the students as their best friend

1.3 Significance

The significance of this project is manifold. With the technological advancements, the possibilities to revolutionize the whole educational system are numerous. Below are listed the reasons why this project is very important for the future of education globally:  
  
**1. Operational Efficiency:** By helping to manage the schedule, guiding through the halls, and tracking attendance, the AI Virtual Assistant lightens the organizational load and removes opportunities for errors.

**2. Enhanced Student Experience:** Through the assistant, individual and timely information is offered to the students to enable them effectively timetable and attend to other academic obligations. In addition, the support given to the student, both emotional and academic, can help them achieve their goals more easily.

**3. Punctuality and Presence:** Regular gentle reminders and efficient use of attendance also make sure that the students attend their classes and are always on time to make such a culture a throve.

**4. Technological Integration:** It demonstrates the ability of AI, geolocation, and facial recognition to be incorporated into applications into common use, establish precedents for employing advanced technologies into technological incorporations for educational software.

**5. Scalability and Adaptability:** The system can also be easily implemented to fit other organizations and institutions of learning hence is very useful in many learning institutions.

**Chapter 2: Literature Review**

2.1 Related Work

Development of an AI Virtual Assistant for educational purposes cuts across numerous fields such as augmented reality, artificial intelligence and mobile computing. This part provides a review of the previous works on timetable management, indoor navigation, class reminders, automated absence tracking and chatbot interfaces that form the basis for the innovations given in this project.

2.1.1 Timetable Management System

Time management is the ability to effectively and advantageously distribute one’s time, especially in the course of working hours. Success means accomplishing things fast as possible, which is an essential factor in this definition. Sometimes busy at work doesn’t translate into productivity or goal achievement. Many people get trapped in simply performing duties or engaging in activities at work rather than generating outcomes or directing them towards a goal.

Through good time management, people are able to do what they need to do and concentrate their efforts on important tasks that matter most. This leads to increased productivity. Time management also plays an important role of reducing stress since it gives an individual a sense of control as well as predictability. For example, organization timetable and planned strategies can help with it. As such, it helps people make informed decisions regarding their obligations; hence promoting positive work-life balance.

The time table planning within the traditional method entailed the undertaking of deep planning/updating. This one is characterized by communication with all the players, and does take a lot of time. Automated Timetable Management System such as Fedena and Genius EDU, use various algorithms to manage timetables. In general, their function is great. They provide ease of planning a day for students and they can view their class schedules. Also, it is easy to create and control schedules for several departments and courses with our University Time Table Generator Software. However, they lack real-time notifications and personalized schedules, which are addressed by the AI Virtual Assistant in this project.

2.1.2 Indoor Navigation Systems

A solid indoor navigation system allows all the actors inside a university to navigate and find what they are looking for. It is interesting to note that indoor navigation has recently shifted to the limelight, particularly with the use of smartphones and augmented reality. Previous work, including the work of Mapsted or Situm, tracked user position by using Wi-Fi signal, Bluetooth beacon, and ARCore. These system assists users in finding their way in indoor spaces but needs additional infrastructures in its implementation. Pokémon Go is used as an example of how we will implement our project using Unity and ARCore for effective and straightforward navigation that does not require extra hardware. Also, the ZXing library which is written in Java will be used in order to read QR codes and navigate directly to the desired facility. A special thing about this is that the user can request the chatbot to go to a class, and in response the chatbot will provide the QR code to the user.

2.1.3 Automated Absence Tracking

The conventional systems for taking attendance are faced with challenges such as increased need for human intervention and more time wasted on collecting counts manually. The use of attendance sheets and signatures has drawbacks that lead to inefficiency, unproductiveness, and wastage of time through impersonation or misplacement. Many universities have tried to solve this issue by using methods such as biometrics, but the result have not been that positive. On an article by “The Guardian” magazine [(Guardian staff reporter, 2018),](https://www.theguardian.com/higher-education-network/2018/apr/13/why-are-students-faking-attendance-they-feel-cheated-by-the-system) this phenomenon is explained in more details.

This issue is successfully solved by this program in the most outstanding way. Firstly, to continue with taking the absence, the user must be connected to the Wi-Fi of the campus. After this, the face recognition process will start. For the final step, geological coordination verification is applied. Each class is defined as a rectangle by geological coordinate, and the class and time where student have lesson is defined in the timetable. By providing these high steps of security, this program successfully solves the issue of attendance tracking.

2.1.4 Chatbot Interface in Education

Chatbots are often implemented in education for provision of quick assistance and support to learners. AI-Chatbots are capable, smart and automated programs which make an attempt to imitate human-like interaction with the users, based on the concepts of artificial intelligence, machine learning and natural language processing, in order to comprehend the scope and purpose of the user inquiries. They can converse in human tongue (as far as technology does this) and also learn the lessons from the conversation. In contrast to traditional software applications that do not use artificial intelligence, AI chatbots are capable of formulating replies to any question that is asked; the form of the question is unimportant. Of course, creating tools of this kind is costly and requires much time, yet higher education institutions cast money into it.

Many universities around the world like [Rogue Community College](https://chatbots.org/university/rogue_community_college/) in US or German Research Center for Artificial Intelligence, have integrated chatbot interfaces in their websites. What differs the chatbot that we are using from the other ones is that except from being able to answer any question related to the policies or rules of the campus, it can also be the student’s best friend. The chatbot is trained with multiple books on friendship, love and psychology. Hence, it is ready to support any student.

2.2 Technologies used

The creation of the AI Virtual Assistant depends on using a mix of different advanced technologies. This part gives an overview of the major technologies used in the project, highlighting what each does towards achieving the system's performance.

2.2.1 Flask Framework

Flasks main appeal is its simplicity and adaptability. Its core is deliberately kept small and easy to grasp making it a welcoming starting point, for beginners while still offering the capabilities for developers. The modular structure of Flask enables developers to incorporate the components they require ensuring that their applications remain lightweight and efficient.

A notable feature of Flask is its method of handling URL routing. Developers can assign URLs to Python functions using decorators resulting in code that's clear and easy to follow. This straightforward approach simplifies the task of determining how web applications respond to requests. The versatile templating system supports inheritance facilitating the maintenance and expansion of templates throughout the application.

Flask excels in managing HTTP requests and responses. The framework provides a request object for accessing request data, such as form inputs, JSON payloads and query parameters. On the response side Flask offers customizable response objects that empower developers to manage the information sent back to clients, including status codes and headers. This degree of control plays a role, in developing secure web applications. Flask framework works on the back end to perform the operations of the requests made to the server and also delivers the web pages and API endpoints invoked by the mobile and Web applications. For these reasons, Flask framework is the best alternative.

2.2.2 OpenCV and SSIM

OpenCV is an open-source computer vision and machine learning software library. Open Computer Vision framework will help to build a universal structure for the computer vision applications and thrust the utilization of the machine perception for the commercial products. As a business, OpenCV’s use is quite easy since it is an Apache 2 licensed product which makes it possible to use and alter the code. In fact, many packages are offered to perform the fundamental of machine learning, deep learning, and computer vision. According to the current state of the literature, the module most suitable for such complex problems is computer vision. OpenCV has the advantage of being supported by different programming languages and various platforms.

While, SSIM stands for the Structural Similarity Index and is a “perceptual” measurement of image quality loss resulting from a transformation process as data compression or from transmission losses. It is a full reference metric and it uses two images from the same image capture but one being the reference image and the other being processed image. Depending on the requirement of the further image usage the processed image is generally in compressed form. For instance, it can be gotten by storing a reference image and opening it in a JPEG format at any compression level. A significant fact worth noting is that SSIM is more famous in the video/image-processing industry but it remains highly applicable to still images.

In conclusion, the project uses OpenCV and SSIM for the face recognition. This way accurate verification can be ensured for absence tracking.

2.2.3 XAMPP

XAMPP is a community Cross-Platform web server solution kit container, basically built and distributed by Apache Friends, composed primarily of the Apache HTTP Server solution, MariaDB database, and scripts written in PH and Perl Interpreter. Since most actual deploys to web servers use most of the components of XAMPP, it can create an easy path from a test server in the lab to a live server. It can understand student details, time table, attendance and it can manage and store these records more accurately.

However, there are some suggestions to not use XAMPP for production even why it is very stable and barely goes down. Because of this reason, the project is built with keeping in mind scalability and flexibility. It is very ease to switch to database platforms like Google Firebase if XAMPP is not suitable for continuing that task.

2.2.4 Unity and ARCore with ZXing Library

Unity is a powerful and expansible tool that has turned into the pillar of game development and other interactive 3D applications including the indoor navigation system. Hence, when coupled with the features of AR Core and with the integration of the ZXing library, the application offers a complete solution to develop robust frame-based navigation applications. It means that Unity, AR Core, and ZXing library can be used to create highly intelligent indoor navigation services.  
These techniques show that Unity is a very powerful game engine since it provides a set of tools for 3D visualization, navigation and building user interfaces. Importing and creating accurate high polygon models is possible in Unity, and these models are important to accurately model the inside of the home. These models can be sculpted in external software programs like Blender or Autodesk Maya and easily transferred to Unity. Through its advanced real-time rendering and subsequent visualization capabilities, Unity guarantees a seamless and attractive navigation interface.  
There are many interactive elements of the indoor navigation system in Unity but the primary one is the NavMesh. Walkable areas for agents like animated characters or guides can be created by using NavMesh. The process of NavMesh baking assigns walkable surfaces which also creates a navigation mesh for the movement. It also allows dynamic obstacles to be defined in the system due to the flexibility of layout of the places.  
Another feature is that there is pathfinding in game, where the company uses A\* algorithm, whereas pathfinding can also be managed and created with C#Script. This particular adaptability also applies to Unity’s event system that can be used for creating more interactivity components such as automatic moving doors and informative kiosks.

This new capability is introduced in Unity through the help of Google’s AR Core framework that adds augmented reality to indoor navigation applications. AR Core allows placing digital hints on the physical space, which makes users have an enchanted and easy navigation experience. It is particularly helpful in the context where conventional signs may not suffice since the environment is better navigated with maxims such as this one.  
  
By using the AR Core API, Unity can access features of motion tracking, environmental understanding, as well as light estimation. Tracking helps the application to capture movements of the user in the real world and his / her orientation as well. The environmental understanding lets the application recognize the flat surfaces, such as the floor and tables, which are critical for placing virtual markers and guides. Light estimation makes virtual objects appear to fit properly in the real just like real objects by modifying the luminosity of virtual objects according to the lighting in the scene.  
Internally, AR Core works with the Unity game engine through the SDK called AR Foundation, which also supports other platforms aimed at augmented reality. This means that developers can sustain enough applications that support AR on multiple platforms so that the applications penetrate various devices, and the users experience similar results.  
ZXing (Zebra Crossing) is one of the most useful open-source libraries for integrating capability of scanning bar codes such as QR codes. Therefore, incorporating ZXing into Unity hugely enhances indoor navigation solutions and allowing people to acquire information easily through QR codes. Indoor application of QR codes can be deployed in a systematic manner where the users can be offered historical or some general information about the place, routes or other relevant data or offers that can be availed with the help of the code.  
  
For instance, in a museum where there are scripts next to specific artifacts, one can use their cellphone and take a picture of the code and be directed to watch videos and more content related to the exhibition. As applied in the retailing domain, it can make it possible to offer information regarding products, specials or discounts as well as direct a consumer to the position of some product in the shop. The ZXing library supports most commonly used barcodes these days and can be implemented into Unity applications with C# scripting.

Indoor navigation applications can be hence implemented on multiple types of devices which include Android, iOS, Windows, as well as, AR/VR headsets by Unity. This keeps the application versatile so that the user can have their navigation aids accessed on the device used at any given point in time.

Another vital facet about indoor navigation is the integration with sensors. It can then communicate with hardware sensors like GPS, IMUs, Bluetooth beacons to enhance positioning accuracy. These sensors could be integrated in the car using custom plugins since the interface can be made to be highly receptive and conscious.  
To expand on this, first and foremost, Unity is a widely used game engine with an enormous community and a vast array of resources, which makes it ideal for indoor navigation development. More so, the Unity Asset Store possesses numerous sensible premade assets, scripts, and plugins that may facilitate reel development. Also, to this, Unity offers nice documentation and hundreds of tutorials, which makes it easy for developers, beginners in special.  
  
When using Unity and its features, augmented reality of AR Core, and barcode scanning of the ZXing library, it becomes possible to develop indoor navigation systems that are as engaging and usable as this one at best. All these technologies collectively make up a solution that effectively solves for the issues of indoor navigation while at the same time it favors the users with an impressive and interactive wayfinding. Whether it is for the retail location, the museum, office space, or the hospital, the incorporation of such tools guarantees the novelty and efficiency of these indoor navigation applications.

2.2.5 PyTorch and Voiceflow

Artificial intelligence is a rapidly transforming area where chatbots have an established track of application that is there. By which businesses can then communicate with customers and thusly carry on their operations more smoothly. PyTorch is, by its design as a solid backing for a library of machine learning, set apart from other tools for creating such dialogue agents. Facebook’s AI Research lab has taken up the work with PyTorch, which was the company’s initiative, by designing it with a set of unique features including dynamic computation, simplex, and activation of a deep learning model.

One of the key features of PyTorch is dynamic computation graphs in it. These are not fixed and formed before the program is run as in the case of static graphs, dynamic graphs are formed in real-time. This feature of flexibility allows developers to change the neural network's architecture in the course of the model's development, thus, it is particularly appropriate for chatbot models which are developed through a series of iterations.

Pytorch with its dynamic computational graphs is very important to chatbot development. One main characteristic of this function is that programmers can experiment with a number of different neurons networking architecture which is critical in NLP (natural language processing) tasks where there are different objectives to be accomplished including for example having the right model structure and app.

On the other hand, an absolutely innovative way of creating chatbots is Voiceflow. It requires absolutely no code at all. You just have to provide some files in order to train the model and everything else is ready.

The great thing is that a comparison can be made to both ways the chatbot is created. This is why two ways of creating a chatbot are used in this project.

2.2.6 Geolocation Services

Through geolocation services, users are not only guided through their decision-making process, but also have become part of people’s daily routine. They use location data and, in turn, they offer almost live data so that people can plan their lives more effectively and businesses can operate more efficiently. In the following essay, the basic theories of geolocation services, technologies that exist to support such services, and the endless possibilities of applying them are described, including such creative examples as the use of indoor positioning for attendance.   
In the broadest sense, we can define geolocation services as having to do with trying to find out where things are in the world. This involves several key technologies like this that we have used in our project:

This involves several key technologies like this that we have used in our project:   
Global Positioning System (GPS): This last mode should be the most familiar to most people today as GPS uses satellite technology to pinpoint positions on the globe. It is very often and accurate for basically navigating through your everyday life (from getting directions to go out or hiking).   
This means that with the increasing development of advancement geolocation technologies will even become more enhanced and much widely used. Further advancements in positioning systems, as well as the growing use and application of integrated Artificial Intelligence for predictive analytics, will open new solutions for future developments enhancing the overall quality of life, as well as business improvements.   
Geolocation services are one of the basic blocks of current socio-technological processes, which are a starting point or a part of many applications that use location as an optimizing factor. Ranging from route determination, social services, contact tracing, indoor navigation for attendance and precision agriculture, these related services are built upon advanced positioning systems, mapping solutions, and strains of location- based services. Thus, the expansion of the global geolocation service market is obvious: with the introduction of new IT solutions, their application will become more versatile and demand more extensive implementation in different industries.

**Chapter 3: System Architecture and Design**

3.1 Overall System Architecture

In the case of an AI Virtual Assistant for the educational purposes, to achieve the purpose of making the system easily scalable, maintainable, and easily integrateable, a modular architecture is used. The system architecture may include few major subsystems that serve different purposes within the framework of the system. This part of the work gives a brief description of the general layout of the entire system and informs the reader about the active communication between these parts. of growth and contributed to the acceleration of change along with creating more opportunities for innovation in different fields.

3.1.1 Architecture Diagram

The architecture consists of the following components:

**Frontend:**  
The frontend of the system includes several important parts that effectively deliver an application with the friendly interface for the students and administrators. These include Smartphone App, Student / Administrator Console, and Website. The Smartphone App is more flexible because it provides the users with an opportunity to carry out desired transactions using their smart phones anytime, anywhere. Console for Student/Administrator is a more extended version of the panel with more options and settings accessible to the admin side of the site only. The main use of Website is in that both students as well as the administrator can log in order to perform their tasks, seek for information, materials and to exploit different facilities of the system. Such components guarantee cohesive and smooth blended experience to enable users to appreciate the product at various interfaces.   
  
**Backend:**  
Backend is like the engine of the system as it processes all of the server-side system requirements and mediates the interaction between the front-end, API, and the database. The backend is completely different than the frontend as the backend is built with heavy server-side coding whereas frontend only put common HTML and CSS together with javascript but does not use Flask at all. It controls and coordinates the use of business logic which the system uses to perform general and specific tasks and operations. The backend focuses on the interaction between the user interfaces and the databases of the application and also manages the interaction with the diverse APIs, thus enabling all the aspects of the system to work simultaneously and cohesively.   
  
**Database:**  
The database is one of the key functionalities of the system; it is devoted to managing and holding all student-related data. These components include, but is not limited to, personal data, schedules, calendars, and participation and other interactions. Thus, the database is supported and controlled by XAMPP that allows creating a convenient system for database management. Using XAMPP, multiple tasks can be started and controlled by the administrators with a high level of access rights to the database where all the data of the students is stored.

**Indoor Navigation:**

Campus navigation helper works through ARCore’s augmented reality (AR) technology and Unity game engine. The system provides indoor navigation assistance that presents visual cues in real-time, simplifying the process of finding respective destinations for students and guests. In addition, this feature delivers an interactive and immersive user experience whereby digital information is superimposed on physical environment.

**Notification System:**

There is a notification system that takes care of dispatching instant notifications to users. Some of the reminders sent include class schedules, upcoming events among others. This means that the system does not fail to inform the users in good time leading to organization. The features are push notifications made by Smartphone App or alerts via Website which enable users to stay connected and informed about important aspects.

**Geolocation Services:**

System integration has geolocation services that monitor where students are within campus premises. With coordinates set, several location-based applications like locating nearby resources or ensuring student safety can be achieved by the system. By incorporating real-time geolocation into its overall functionality, the services enhance it tremendously.

By incorporating these components, the system provides a comprehensive and sophisticated platform that meets the diverse needs of students and administrators in an educational setting. Each component plays a vital role in ensuring the system's overall efficiency, usability, and effectiveness.

3.2 Timetable Integration

A major aspect of this platform is integration with one’s timetable: it allows students to view and organize their class schedules more effectively. This part discusses the design and implementation of the timetable integration module, focusing on how it improves user interaction and simplifies access to academic calendars.

3.2.1 Data Ingestion

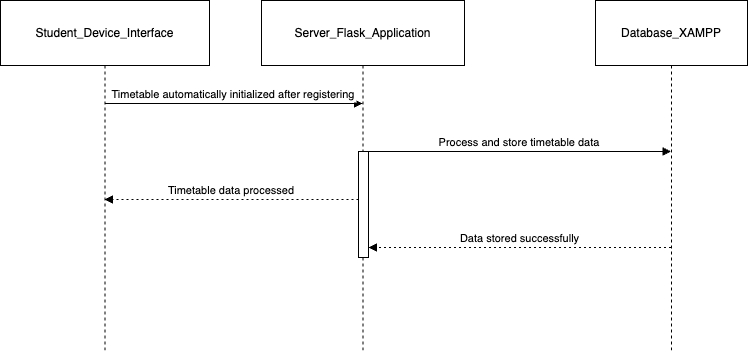
The time table data ingested into the system comes from the administration of schools. It includes class schedules, subjects, classroom locations, times which represent a good cross-section of all key details found in any conventional schedule. The process entails parsing timetables that are normally available as CSV or other structured formats like Excel or XML among others. After validation for accuracy and consistency, such data is stored within databases. To address regular updates from school authorities on adjustments in timetables automated ingestion process has been adopted. Such robust mechanism ensures that learners can always obtain the latest version of their class schedule information.

3.2.2 Schedule Management

The students can access their timetable via the Android and iOS application, as well as the web platform. The stated frontend communicates with the backend API to obtain the student timetable information stored in the database. When retrieved, the schedule is shown in a structure that is easy to follow and comprehend in a matter of time. This format often comprises a weekly or daily schedule with reference to classes that are in the days to come, class locations as well as the class times. They can include color-coded subjects, clickable links, which when clicked show the classroom maps, or a filter where one can view lessons for a particular day or subject. The interface also provides the provision of customizing the home page whereby the student can be able to drop a note or reminder concerning a particular class. This timetable management system makes sure that the student is well scheduled to meet all his or her academic activities in every faculty and is able to meet all his or her commitments within the stipulated time.

3.2.3 Real-Time Notifications

Since the notification system completes its objectives of informing the students of their schedules real-time, it is quite vital. The system also has a time comparison cycle where it compares the current time with the time a student has planned for a course. Instead, when a particular class is due, a notice is given to the student’s device consisting of class details including the subject area of the class, time for the class, and even the room in which the class will take place. Such notifications can be made to be personalized depending on each customers’ requirements; the student for instance can set the number of weeks or days they want the system to notify them of an upcoming exam. Furthermore, the notification system helps users to receive the notification regarding the schedules and any changes, either in cancellations or special announcements made by the instructors. This feature of real-time notification is intended to help students avoid being absent in classes and also help them to be conversant with other schedules that they are to adhere to on daily basis. It also makes students punctual and prepared for classes hence improve their attendance.   
  
In summary, the timetable integration module remains one of the central parts of AI Virtual Assistant’s system since it is aimed at facilitating the management of schedules within the academic environment for students. Via data ingestion, one can easily and quickly keep track of the activities to be done in the classroom or related to the class, schedule management - a user-friendly tool thus enabling the students meet these deadlines and commitments and last but not least the notifications that the system offers in real time.

  
Figure 3.1: Sequence Diagram of Timetable Integration

3.3 Indoor Navigation and Wayfinding

The indoor navigation system is one of the most important features of the app which helps students to navigate the school campus easily. This system facilitates students by leveraging technologies such as augmented reality (AR) and QR codes that enables the students to reach their destination faster. The navigation process is more fun with this system as it assists students in every way possible.

3.3.1 ARCore Integration

For a better augmented reality experience, the system uses ARCore, which is a platform from Google that helps in the creation of AR applications. AR functionality within this app is created using Unity engine which is a powerful tool that helps to create 3D interactive work for engineering, gaming etc. With the help of this integration, the AR application of navigation furnishes current location of the person referring to the real-world environment as seen through a smartphone's camera, thus overlaying information such as arrows, markers and pathways on the existing view. This helps the student’s reaching his/her destination. Using ARCore, data gets recorded through smartphone's camera in the form of images and videos and provides navigation in future with the help of a real-world environment. This will help to reach destination without any difficulty due to technological advances, as AR provides the look of virtual world to real world with overlay functions.

3.3.2 QR Code Scanning

The system also uses QR code, which assists location-based services of the app for reaching the destination faster. This can be done by using the ZXing library, which is a robust and most useful toolbar code scanning in any platform. QR codes will be fixed at different spots of the school such as entrances, corridors and near to important facilities so that students can scan those codes to get the needed information. Once a student scans QR codes with the help of a compatible smartphone's camera, the app will recognize that student's current location and provide step-by-step shortest route to reach next class or a particular facility.

3.3.3 Navigation Aids

Students are shown to go to their classrooms as well as any other places within the school by using visual cues and guidance provided by the program. This feature is most applicable if new students who do not know where various parts of the school are located. To make movement easier, arrows, path lines and distance indicators are used as indicators in the maps.

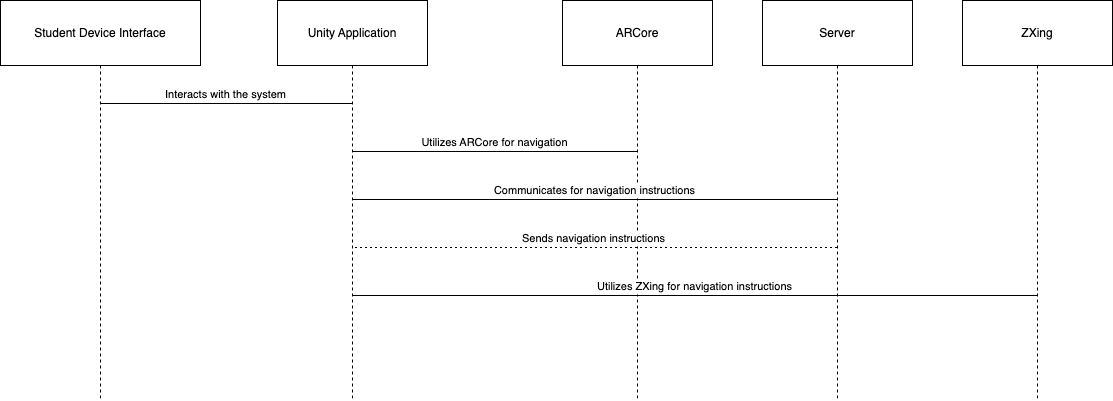


Figure 3.2: Sequence Diagram for Indoor Navigation and Wayfinding

3.4 Automated Absence Tracking

To keep accurate records of who is present and who is not, is extremely important. By combining the facial recognition technology and geolocation services in this module, the presence of students within a class room is being confirmed making sure that an attendance system that is stable and effective is maintained.

3.4.1 Ensuring Position in the Classroom

The process of classroom position verification is aimed at ascertaining that students are physically in the class when attendance is being taken. This means that advanced geolocation services are used to monitor and verify student locations in real time.

3.4.1.1 Geolocation Services

The system tracks the student's location inside the campus using GPS technology. The detection of the student happens when the student comes under the classroom area. The system sends that information to the backend for further processing. This information of GPS is then cross-referenced with predefined classroom coordination’s, that are already marked inside the system. This Cross-referencing will make that detected location match the exact boundaries of the particular classroom. The system can accommodate several classroom coordinates for giving precise tracking in case of different rooms, buildings, and so on within the campus.

3.4.1.2 Position Verification Process

A backend logic is set up for this purpose: comparing geolocation data tagged to a student with the predefined classroom coordinates. The algorithm involves sophisticated logic of checking whether the student is well within the defined classroom boundaries. The logic also takes into account the error that may occur via GPS and takes in a margin of error and some other checks; if the distance of the student happens to be in the acceptable spectrum, it confirms his presence and marks him as 'present' in the database. This is the most crucial stage in terms of making the attendance list really accurate thus leaving behind those students who are not present in class physically.

3.4.2 Face Recognition

Mobile face recognition also offers an extra layer of check. The system ensures that the student who is physically in class is the one registered, leaving no room for fraud in attendance.

3.4.2.1 Face Capture

Once the classroom is reached, then the mobile application will prompt the pupil to click a front camera selfie. The entire process is visualized to be quick and very easily accessible by taking less than five seconds. This gives the facial recognition software with primary data in the form of a picture conferment regarding who the student actually is. The timer is set such that the entire prompt occurs at the very start without wasting any class time.

3.4.2.2 Image Processing

With OpenCV as an open source for computer vision, it gives a robust way to do the processing on the captured image. The system compares the captured image with the stored facial features of the student using structural Similarity Index. SSIM is good in handling image similarities, which present a great avenue for verification of images. The facial recognition model is trained for various lighting conditions, angles, and facial expressions for high degrees of accuracy. In the event of a successful match with the comparison, a student is marked as present in the respective class. The dual-layer process - geolocation and facial recognition - brings high reliability to the attendance system in place.

3.4.3 Campus Domain Connectivity

As a way to make the attendance tracking system even more secure and reliable, it will be mandated that all students are connected to the campus domain network whenever they check in for attendance. Making it a requirement ensures that only students who are physically present on campus can contribute to being counted in the attendance process and that there's no possible way for remote and fraudulent check-ins.

The system confirms if a student is actually on campus by checking his or her connection to the campus domain network in the following ways:

Network Authentication: When trying to check in, the mobile application checks whether it is connected to the campus Wi-Fi network. This is made possible by network authentication protocols which confirm whether one is connected to a network.

Domain Verification: It looks for campus-specific network credentials. It ensures that the device is a part of the campus network. It checks the SSID of the network and cross-checks the same with the pre-defined campus network identifiers.

Backend Verification: The authentication of the network routing is sent to the backend system where it is related to the campus domain.  Only if the device's connectivity to the authorized campus network is verified, is the check-in request for attendance approved.

The above-mentioned connectivity requirement shall not only make the attendance tracking system more secure but also ensure that every attendance record matches the physical presence of students on campus. In this respect, a geolocation component, combined with facial recognition and network authentication, offers a seamless and secure method of automatic attendance tracking.

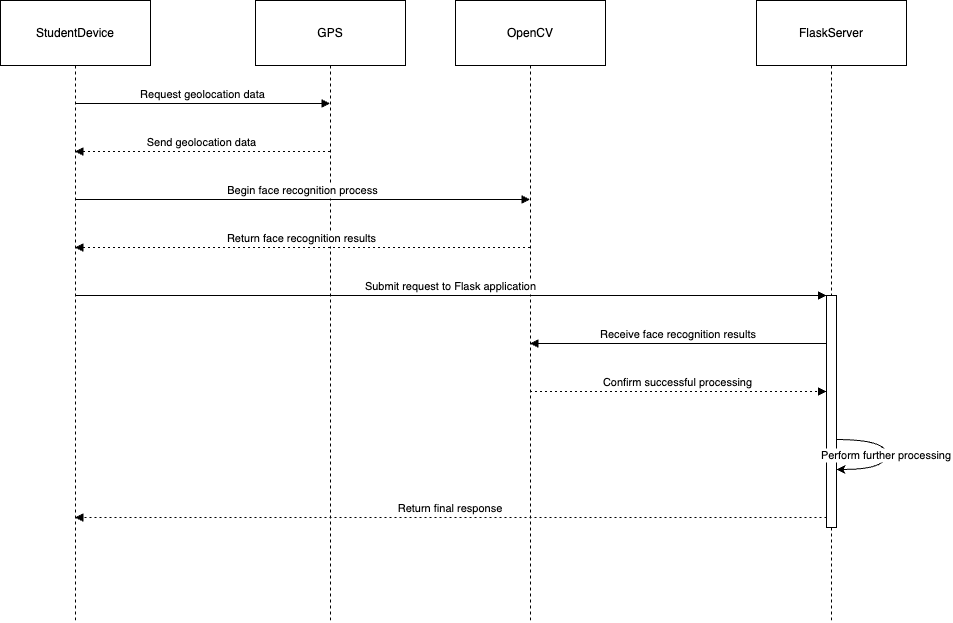


Figure 3.3: Sequence Diagram of Absence Tracking

3.5 Chatbot Interface

This is a vast area of functionality in the AI Virtual Assistant and hence a general chat interface would constitute the backbone of any conversational interaction on the part of students with the system. The students would be enabled to seek information actively and perform tasks within the system using natural language queries that would encapsulate an enhanced user experience in terms of intuition and efficiency.

3.5.1 Development of Chatbot

The framework chosen for the development of the chatbot is PyTorch-a very powerful and flexible deep learning framework. As it is powered by PyTorch, a highly sophisticated deep learning mode allows the chatbot to analyse and respond to any student queries with extreme accuracy. Once trained on a large data set of queries and responses of students, this chatbot can respond to almost all kinds of requests starting from timetable-related queries to navigating around the campus and even administrative requests. The natural language processing embedded in the chatbot can differentiate the various contexts of the user inputs and capture the intents. Thus, such schemes as tokenization, entity recognition, sentiment analysis enable the chatbot to get accurate differentiation and response capability for diverse and complex innuendos.

3.5.2 Integration with Back-End Services

The chatbot interacts seamlessly with the back-end services in order to fetch the data or execute any action in real time. For instance, when the student asks the chatbot about his next class, it will look for the timetable module to get the information to make the appropriate response. In another case, a student may be lost and asks for his way, the chatbot would revert to the indoor navigation module to give directions to the lost student. All these cook the chatbot to give almost timely and accurate responses since it fetches real-time data stored on the system's database. It provides the possibility of real-time access to updated information and smooth efficient data flow by setting up different APIs that support the chatbot in communicating with each of these backend modules. This makes the utility and reliability of the chatbot ++, rendering it as a proper utility tool for the students.

3.5.3 User interaction

The students shall be using the software through both the mobile application and the web interface as well, which would give it flexibility, and convenience. It allows a student to obtain the desired information in a much easier way by letting the student either type or even speak out the question to be answered, rather than going through multiple menus or inter-faces. It makes those interactions more usable, natural, and intuitive. The proposed chatbot accepts both types of input, including text and voice. The tool also comprises voice recognition technology as incorporated in the chatbot. This would principally assist students who may have some problems in typing or want to experience hands-free interactions. In addition, the interface of the chatbot is to be contextual, responsible and adaptable for consistency and seamless usability in various devices with different screen sizes.

Taken together, a chatbot interface is a constituent part of an AI Virtual Assistant to provide such a user-friendly and seamless, as well as accessible method of students' communication with the system as possible. It has been developed using the latest achievements in chatbot creation, convenient integration with the backend services, and multiple ways of user interaction. This is helpful to improve the student experience with fast and easy access to all the necessary information and services.

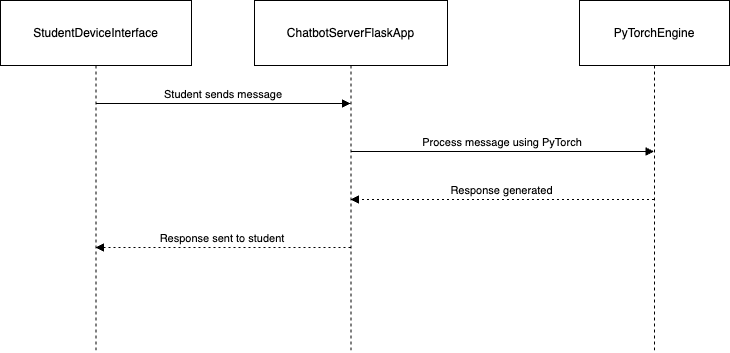


Figure 3.4: Sequence Diagram of Chatbot Interface

**Chapter 4: Implementation of the Program**

4.1 Development Tools and Environment

In this project of AI Virtual Assistant, varieties of tools and different environments were used for effective implementation. These are the major tools and technologies used:

* Flask: A Python-based lightweight web framework is used at the back end of the system.
* Unity: This is a hybrid cross-platform game engine. As ARCore is integrated with the indoor navigation and way-finding system, this hybrid masterpiece is created.
* ARCore: Augmented reality applications can be built with an SDK developed by Google.
* ZXing: Open-source, multi-format 1D/2D barcode image processing library in Java. It is used in Unity for QR code scanning.
* OpenCV: It is like a library of programming functions mainly applied to real-time computer vision. Here, it is applied to face recognition.
* SSIM: Structural Similarity Index is a measure of similarity between two images. Used with OpenCV to perform face recognition.
* PyTorch: It is an open-source machine learning library that is getting based from the Torch library, used for the chatbot development.
* XAMPP: It is a free and open-source cross-platform web server solution stack package, which has been developed by the Apache Friends, and it is being used for Database Management in this study.
* GPS: It stands for Global Positioning System. This has been used to provide some geolocation services that will pinpoint the way students are present in class.
* Voiceflow: It is a web-based platform which allows the development of chatbots without coding. This is used in this project to show an innovative way of developing chatbots.

4.2 Implementing Timetable Integration

4.2.1 Collecting and Formatting Timetable Data

To start integrating student timetables into the AI Virtual Assistant system, some actions need to be performed before the integration procedure: Gathering timetable data; Organizing collected timetable information; Analyzing the results that the system shows based on working with the timetable data. This process is important to avoid incidence of having dirty data which may be inconsistent and time-consuming when used in the current system.

The timetable data is typically provided by the school administration in a structured format, such as CSV or Excel files. These files contain detailed information about the class schedules, including the class name, interval of time, course time, class days, and any additional notes based on the way it can be structured. Here is the csv file that we have used as a timetable for this project which is called timetable.csv:

*Day,Class,09:00 - 11:00,11:00 - 13:00,13:00 - 15:00  
Monday,a1,Calculus I,Academic Writing,  
Tuesday,b2,Chemistry,Computer Application,  
Wednesday,c3,Computer Science,Operating Systems,  
Thursday,a1,Calculus I,Chemistry,  
Friday,b2,Computer Science,Computer Application,*

After having ready the timetable, we need to ingest the data or simply read the csv file. This is done by using the pandas library, which is a library that deals with data sets. It has functions for reading, manipulating and cleaning the data. Here is the code for reading the csv file of the timetable:

# Load timetable from CSV file  
timetable = pd.read\_csv("timetable.csv")

4.2.2 Processing Timetable Data

Now that the timetable is initialized, we need to extract the data needed from it. While taking the attendance, we need the course name in the exact day and time. This is needed because then we are going to count the absences of each course and store them in the database. Here is the code to implement for this function:

def get\_current\_course():  
 now = datetime.now()  
 current\_time = now.strftime("%H:%M")  
 current\_day = now.strftime("%A")  
  
 logging.debug(f"Looking for current course at time {current\_time} on {current\_day}")  
 for index, row in timetable.iterrows():  
 if row['Day'] == current\_day:  
 for time\_slot in ['09:00 - 11:00', '11:00 - 13:00', '13:00 - 15:00']:  
 try:  
 start\_time, end\_time = time\_slot.split(' - ')  
 start\_time = start\_time.strip()  
 end\_time = end\_time.strip()  
 if start\_time <= current\_time <= end\_time:  
 course = row[time\_slot]  
 if pd.notna(course):  
 logging.debug(f"Found current course: {course}")  
 return course, start\_time  
 except ValueError as e:  
 logging.error(f"Error parsing time slot '{time\_slot}': {e}")  
 logging.debug(f"No course found for current time {current\_time} on {current\_day}")  
 return None, None

The ‘get\_current\_course()’ function is expected to take the current time and the current day as the input and enable a course to be determined for a student to attend. It first calls datetime to get the current date and time at which it is being executed. They convert now() and formats these into a 24-hour time format and the full name of the day. The function logs this information just for the sake of debugging. It then loops through all the rows of the ‘timetable’ data frame to locate the correct row that is reserved for the current day of the week. For each row corresponding to the current day, it examines predefined time slots (‘09 - 11:00’, ‘11:00 – 13:00’ , ‘13:00 - 15:00’). Instead of using these time segments as the interval type, splitting these time slots into start and end periods of time, the function is able to determine if the current time belongs to any of the said time segments. If the current time means in time slot, it fetches the course put down for the given period in the DataFrame. If valid course is discovered that is, course entry is not NaN then it prints the course and the start time and returns the course name.  
  
What happens if the function encounters a time slot parsing error? It simply logs the error, rather than stalling the whole thing. It then checks its timetable for the current time and day; if this time is not present in any course in the timetable after going through all courses, it transcribes a message and returns None, None. This systematic approach of checking the timetable ensures that the function accurately identifies the current timetable and thus people can be able to manage their schedule immediately it is produced. As a matter of fact, the design will be highly suitable for educational uses as well as any organization where quick and efficient access to the schedules is important to both the students and other officials.

4.2.3 Generating and Sending Notifications

Notifications are generated and sent based on the timetable. Only 10 minutes before the class starts, the student will be notified. Furthermore, the student can be advised to study for the courses it has the day after. Here is the code that will deal with these functionalities:

import mysql.connector  
from datetime import datetime, timedelta  
  
# Function to fetch today's timetable  
def fetch\_todays\_classes(user\_id):  
 today = datetime.now().date()  
 query = "SELECT \* FROM Timetable WHERE user\_id = %s AND DATE(class\_time) = %s"  
 cursor.execute(query, (user\_id, today))  
 return cursor.fetchall()  
  
# Function to fetch tomorrow's timetable  
def fetch\_tomorrows\_classes(user\_id):  
 tomorrow = datetime.now().date() + timedelta(days=1)  
 query = "SELECT \* FROM Timetable WHERE user\_id = %s AND DATE(class\_time) = %s"  
 cursor.execute(query, (user\_id, tomorrow))  
 return cursor.fetchall()

The code given in the solution aims to retrieve class schedules of a particular user from a database where the emphasis is given to today and tomorrow schedules. The code begins by importing necessary modules: mysql, importing the sqlalchemy library for creating connections to the database and importing the datetime and timedelta classes from the datetime module for handling date and time calculations. The function fetch\_todays\_classes is responsible for collecting the list of classes for a particular user for the current day. It first asks for the present day by using the function called datetime. now(). date(), to which the program then forms a SQL query which then pick all records in the Timetable table that belongs to the given user ID and where the class\_time is equal to today’s date. The query is executed using cursor to get the selected records. Commit(), and the result obtained is returned with the cursor execute(). fetchall().  
  
Likewise, the fetch\_tomorrows\_classes function is aimed at providing the calendar with the classes to be held the next day. It can generate tomorrow’s date by using timedelta and adding one to today’s date through timedelta(days=1). Functionally similar to the previous one, it builds an SQL query for finding records in the timetable with the given user\_id and the class\_time coinciding with the date of the following day. Cursor does the similar function of query executing and retrieval of result with execute(), cursor and fetchall(). This way, it is possible for users to view their calendar for not only the current day but also the next day – allowing them to make much better scheduling decisions.

4.3 Building Indoor Navigation and Wayfinding

4.3.1 Setting Up Unity and ARCore

Indoor Navigation System Set up begins with the installation of the framework, Unity and ARCore for the application. To build the described application, there is the Unity platform that allows integrating AR by using ARCore.  
  
Steps to Set Up Unity and ARCore:Steps to Set Up Unity and ARCore:  
  
1. It’s preferable to install Unity as the primary game development software and then make a new project.  
2. Download the latest version of ARCore SDK for Unity from the Google’s official website.  
3. ARCore is set up as a plugin for the Unity project.

4. Install the ZXing library.  
5. Prepare the ARScene for ARCore usage by installing ARCore components in the scene.

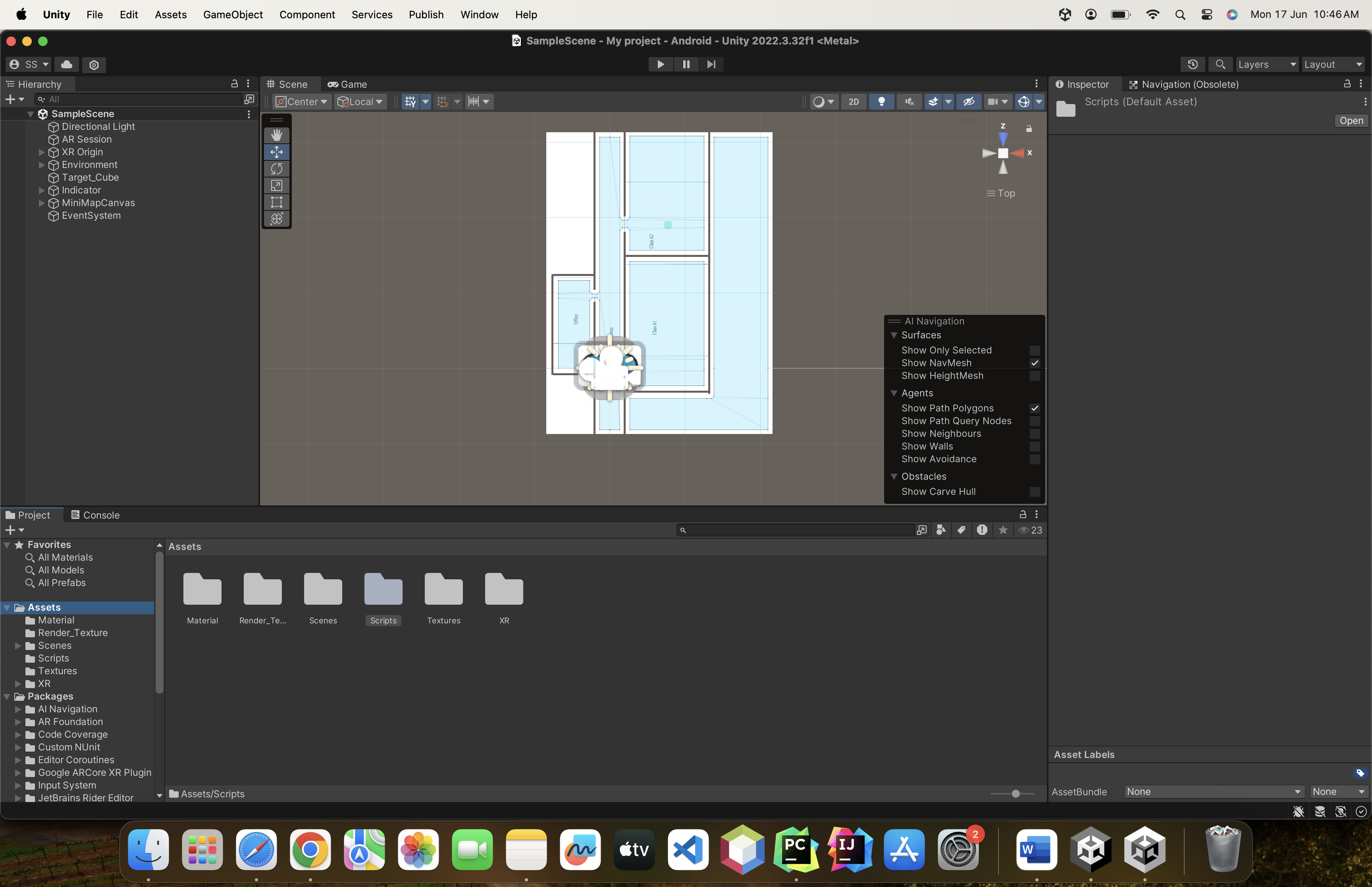


Figure 4.1: Development of Indoor Navigation in Unity

4.3.2 Generating Navigation Instructions

Navigation Instructions are very important when it comes to navigating. A script is created in order to set navigation targets:

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class SetNavTarget : MonoBehaviour

{

[SerializeField] private Camera TopDownCamera;

[SerializeField] private GameObject navTargetObject;

private UnityEngine.AI.NavMeshPath path;

private LineRenderer line;

private bool lineToggle = false;

// Start is called before the first frame update

private void Start()

{

path= new UnityEngine.AI.NavMeshPath();

line=transform.GetComponent<LineRenderer>();

}

// Update is called once per frame

private void Update()

{

if ((Input.touchCount > 0)&&(Input.GetTouch(0).phase == TouchPhase.Began)){

lineToggle = !lineToggle;

}

if(lineToggle){

UnityEngine.AI.NavMesh.CalculatePath(transform.position, navTargetObject.transform.position, UnityEngine.AI.NavMesh.AllAreas, path);

line.positionCount = path.corners.Length;

line.SetPositions(path.corners);

line.enabled = true;

}

}

}

4.3.3 Integrating QR Code Scanning

The functionality of QR Code Scanning and then the navigation to the respective class is done by the help of the ZXing library which is mentioned before. Here is the script used for this purpose:

using Unity.Collections;

using Unity.XR.CoreUtils;

using UnityEngine;

using UnityEngine.XR.ARFoundation;

using UnityEngine.XR.ARSubsystems;

using ZXing;

public class QrCodeRecenter : MonoBehaviour {

[SerializeField]

private ARSession session;

[SerializeField]

private XROrigin sessionOrigin;

[SerializeField]

private ARCameraManager cameraManager;

[SerializeField]

private TargetHandler targetHandler;

[SerializeField]

private GameObject qrCodeScanningPanel;

private Texture2D cameraImageTexture;

private IBarcodeReader reader = new BarcodeReader();

private bool scanningEnabled = false;

private void OnEnable() {

cameraManager.frameReceived += OnCameraFrameReceived;

}

private void OnDisable() {

cameraManager.frameReceived -= OnCameraFrameReceived;

}

private void OnCameraFrameReceived(ARCameraFrameEventArgs eventArgs) {

if (!scanningEnabled) {

return;

}

if (!cameraManager.TryAcquireLatestCpuImage(out XRCpuImage image)) {

return;

}

var conversionParams = new XRCpuImage.ConversionParams {

// Get the entire image.

inputRect = new RectInt(0, 0, image.width, image.height),

// Downsample by 2.

outputDimensions = new Vector2Int(image.width / 2, image.height / 2),

// Choose RGBA format.

outputFormat = TextureFormat.RGBA32,

// Flip across the vertical axis (mirror image).

transformation = XRCpuImage.Transformation.MirrorY

};

// See how many bytes you need to store the final image.

int size = image.GetConvertedDataSize(conversionParams);

// Allocate a buffer to store the image.

var buffer = new NativeArray<byte>(size, Allocator.Temp);

// Extract the image data

image.Convert(conversionParams, buffer);

image.Dispose();

cameraImageTexture = new Texture2D(

conversionParams.outputDimensions.x,

conversionParams.outputDimensions.y,

conversionParams.outputFormat,

false);

cameraImageTexture.LoadRawTextureData(buffer);

cameraImageTexture.Apply();

// Done with your temporary data, so you can dispose it.

buffer.Dispose();

// Detect and decode the barcode inside the bitmap

var result = reader.Decode(cameraImageTexture.GetPixels32(), cameraImageTexture.width, cameraImageTexture.height);

// Do something with the result

if (result != null) {

SetQrCodeRecenterTarget(result.Text);

ToggleScanning();

}

}

private void SetQrCodeRecenterTarget(string targetText) {

TargetFacade currentTarget = targetHandler.GetCurrentTargetByTargetText(targetText);

if (currentTarget != null) {

// Reset position and rotation of ARSession

session.Reset();

// Add offset for recentering

sessionOrigin.transform.position = currentTarget.transform.position;

sessionOrigin.transform.rotation = currentTarget.transform.rotation;

}

}

4.4 Automating Absence Tracking

4.4.1 Utilizing Location Tracking

Location Tracking is done in the moment that the user starts to take the attendance. The user position must be verified that belongs to the class where he/she has lesson. Here is the code for getting the exact user location:

function getLocation() {  
 if (***navigator***.geolocation) {  
 ***navigator***.geolocation.getCurrentPosition(sendLocationToServer);  
 } else {  
 alert("Geolocation is not supported by this browser.");  
 }  
}  
  
function sendLocationToServer(position) {  
 const data = {  
 latitude: position.coords.latitude,  
 longitude: position.coords.longitude  
 };

The given code includes a function that is named getLocation and is used to locate the geographic location of the user by tapping into the ability of the browser. When the getLocation method is invoked, it implies to check the support of Browser Geolocation through the navigator. geolocation object. If supported, it invokes the call-navigator function. geolocation. Sending the current position to the server by using the function getCurrentPosition with a function called sendLocationToServer to handle the location data. If geolocation is not supported, it informs the user so that he/she can enable it or use a different way. The sendLocationToServer function is then defined which takes a position object from a geolocation API call, then extracts position.coords.latitude and position.coords.longitude. Here, the result is a collection of reg\_coord structs, or coordinates, that need to be formatted and organized into a data object. This data object includes the user’s geographic coordinates, which can be directly sent to a server for the next handling step.

4.4.2 Implementing Face Recognition

Face recognition is implemented using OpenCV and SSIM. Below is written the code that implements both algorithms:

def compare\_images(img1, img2):  
 try:  
 s = ssim(img1, img2)  
 logging.debug(f"SSIM: {s}")  
 return s > 0.4 # Adjust threshold as needed  
 except Exception as e:  
 logging.error(f"Error during image comparison: {e}")  
 return False

for (x, y, w, h) in faces:  
 cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)  
 roi\_gray = gray[y:y + h, x:x + w]  
 try:  
 logging.debug(f"Predicting face at position {(x, y, w, h)} with ROI shape: {roi\_gray.shape}")  
 id\_, confidence = recognizer.predict(roi\_gray)  
 logging.debug(f"Detected face with id {id\_} and confidence {confidence}")  
 except cv2.error as e:  
 logging.error(f"OpenCV error during face prediction: {e}")  
 continue  
 except Exception as e:  
 logging.error(f"General error during face prediction: {e}")  
 continue  
  
 if confidence < 130:  
 user\_id = id\_  
 photo = get\_user\_photo(user\_id)  
 if photo is not None:  
 try:  
 user\_img = cv2.imdecode(photo, cv2.IMREAD\_COLOR)  
 if user\_img is None:  
 logging.error("Failed to decode user image")  
 continue  
  
 user\_img\_gray = cv2.cvtColor(user\_img, cv2.COLOR\_BGR2GRAY)  
 user\_img\_gray\_resized = cv2.resize(user\_img\_gray, (w, h))  
 logging.debug(f"Detected ROI shape: {roi\_gray.shape}, User image shape: {user\_img\_gray.shape}, Resized user image shape: {user\_img\_gray\_resized.shape}")  
 logging.debug(f"Detected ROI mean: {roi\_gray.mean()}, Resized user image mean: {user\_img\_gray\_resized.mean()}")  
 logging.debug(f"Normalized ROI mean: {roi\_gray\_normalized.mean()}, Normalized resized user image mean: {user\_img\_gray\_resized\_normalized.mean()}")  
  
 # Compare images using SSIM  
 if compare\_images(roi\_gray\_normalized, user\_img\_gray\_resized\_normalized):  
 user\_coords = (latitude, longitude)  
 if is\_user\_within\_rectangle(user\_coords, rectangle\_coords):  
 logging.debug("User is within the attendance area")  
 course = current\_class  
 record\_attendance(user\_id, course)  
 cap.release()  
 cv2.destroyAllWindows()

This code is aimed to detect the face in the frame, recognize the face, and, if some specific aspects are met, take attendance for the distinguished individual. First it changes the frame from color to gray as it is easier to detect faces in grayscale image using OpenCV cvtColor. It then utilizes a trained Haar cascade classifier (face\_cascade.detectMultiScale) to analyze the grayscale image for the presence of faces. If no faces are detected it will do so and move to the next frame where it will look again for faces. For each of the faces found and its bounding box coordinates (x, y, w, h), it overlays a rectangle on the given frame and operations on the grayscale image perform an operation called region of interest (ROI) extraction. The recognizer then tries to estimate the ID and the degree of faces’ confidence within the ROI. If the confidence to make a prediction is still low, it goes back to a database and pulls out the user’s photo, decodes and then processes it. The code then converts both the grayscale images of the detected face and the user photo to float to facilitate matrix normalization and proceeds to perform the Structural Similarity Index (SSIM) comparison. If the faces found correspond to the existing user images and the second set of coordinates corresponds to the defined area, the user’s attendance for the current class is considered enrolled. There are several logging statements incorporated in this process to track every possible step and also to look for any error that might occur.

4.4.3 Recording Attendance in Database  
After taking the attendance successfully, the information has to be submitted somewhere. For this reason, tables in the database are created and here is the code that we have used:

def record\_attendance(user\_id, course):  
 now = datetime.now()  
 time = now.strftime('%Y-%m-%d %H:%M:%S')  
  
 try:  
 cursor.execute(“INSERT INTO Attendance (user\_id, time, course) VALUES (%s, %s, %s)",  
 (user\_id, time, course))  
 db\_connection.commit()  
 logging.debug(f"Recorded attendance for user\_id {user\_id} in course {course} at {time}")  
 except mysql.connector.Error as err:  
 logging.error(f"Error recording attendance: {err}")

Simply after the user has passed the verification process of face recognition and position verification, the records will be inserted in the database.

On the other hand, if the user does not check in, he will get an absence. This is done by a Cron job which checks for the existing users that have checked in and triggers an event so that the others will get an absence. Here is the code:

def trigger\_absence\_recording():  
 try:  
 attendance\_taker.record\_absences()  
 return jsonify({"message": "Absences recorded successfully"}), 200  
 except Exception as e:  
 logging.error(f"Error recording absences: {e}")  
 return jsonify({"error": str(e)}), 500  
  
def schedule\_jobs():  
 scheduler = BackgroundScheduler()  
 scheduler.add\_job(trigger\_absence\_recording, 'cron', hour='11,13,15')  
 scheduler.start()

4.5 Creating the Chatbot Interface

4.5.1 Setting Up Chatbot Front-End

Indeed, front-end is one of the most vital components in AI system since that is where people converse with the system in question. It has the potential to give the user a friendly interface that feels like an everyday conversation to even boost their experience and satisfaction. The use of the natural language check-in procedure offered to the students through the chatbot front end removes unnecessary barriers and makes the system more easily navigable for students. It is also versatile for handling numerous queries simultaneously and effectively, lessening the dependence for immediate client interfacing and consequently cuts down time and costs for students. Here is the front-end code for the chatbot:

<div class="container">  
 <div class="chatbox">  
 <div class="chatbox\_\_support">  
 <div class="chatbox\_\_header">  
 <div class="chatbox\_\_image--header">  
 <img src="https://img.icons8.com/color/48/000000/circled-user-female-skin-type-5--v1.png" alt="image">  
 </div>  
 <div class="chatbox\_\_content--header">  
 <h4 class="chatbox\_\_heading--header">Chat support</h4>  
 <p class="chatbox\_\_description--header">Hi. My name is Anwel. How can I help you?</p>  
 </div>  
 </div>  
 <div class="chatbox\_\_messages">  
 <div></div>  
 </div>  
 <div class="chatbox\_\_footer">  
 <input type="text" placeholder="Write a message...">  
 <button class="chatbox\_\_send--footer send\_\_button">Send</button>  
 </div>  
 </div>  
 <div class="chatbox\_\_button">  
 <button><img src="{{ url\_for('static', filename='images/memo\_7.png') }}" alt=""/></button>  
 </div>  
 </div>  
</div>  
  
<script>  
 $SCRIPT\_ROOT = {{ request.script\_root|tojson }};  
  
 function sendLocationToServer(latitude, longitude) {  
 fetch(`${$SCRIPT\_ROOT}/submit\_location`, {  
 method: 'POST',  
 headers: {  
 'Content-Type': 'application/json',  
 },  
 body: ***JSON***.stringify({ latitude: latitude, longitude: longitude })  
 })  
 .then(response => response.json())  
 .then(data => {  
 ***console***.log('Success:', data);  
 })  
 .catch((error) => {  
 ***console***.error('Error:', error);  
 });  
 }

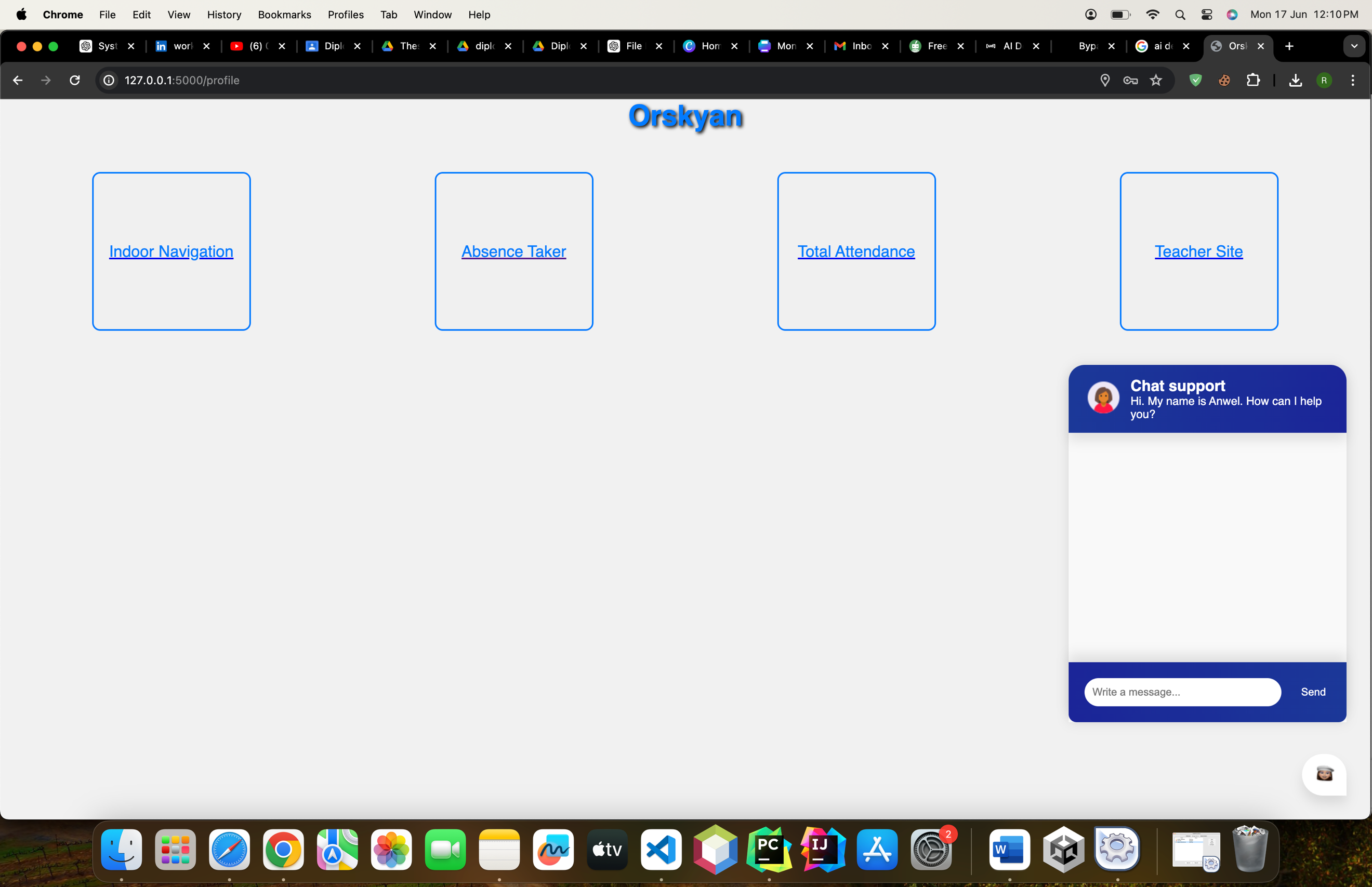


Figure 4.2: Chatbot interface representation

4.5.2 Chatbot Deployment

The first thing to do when deploying the chatbot is to create the model. In this project we have used PyTorch to do that. Here is the code representation:  
  
  
class NeuralNet(nn.Module):  
 def \_\_init\_\_(self, input\_size, hidden\_size, num\_classes):  
 super(NeuralNet, self).\_\_init\_\_()  
 self.l1 = nn.Linear(input\_size, hidden\_size)   
 self.l2 = nn.Linear(hidden\_size, hidden\_size)   
 self.l3 = nn.Linear(hidden\_size, num\_classes)  
 self.relu = nn.ReLU()  
   
 def forward(self, x):  
 out = self.l1(x)  
 out = self.relu(out)  
 out = self.l2(out)  
 out = self.relu(out)  
 out = self.l3(out)  
 # no activation and no softmax at the end  
 return out

The following code presents a simple feedforward-neural-network architecture implemented in the context of PyTorch programming. The NeuralNet class inherits from nn, which is a convolutional neural network that has been utilized in the paper. Module it consists of three densely connected layers which are l1 l2, and l3. The ReLU activation function is used after working first and second layers. The forward method corresponds to the forward pass of the model, performing feedforward on the input x through the layers’ transformation and activation functions, but without applying an additional activation or feeding the resulting vector into the softmax function at the end of the forward pass.

Now that the model is created it needs to be trained. For that, an intents.json file is created which contains possible questions and answers. Below you may see the code:  
  
with open('intents.json', 'r') as f:  
 intents = json.load(f)  
  
all\_words = []  
tags = []  
xy = []  
# loop through each sentence in our intents patterns  
for intent in intents['intents']:  
 tag = intent['tag']  
 # add to tag list  
 tags.append(tag)  
 for pattern in intent['patterns']:  
 # tokenize each word in the sentence  
 w = tokenize(pattern)  
 # add to our words list  
 all\_words.extend(w)  
 # add to xy pair  
 xy.append((w, tag))  
  
# stem and lower each word  
ignore\_words = ['?', '.', '!']  
all\_words = [stem(w) for w in all\_words if w not in ignore\_words]  
# remove duplicates and sort  
all\_words = sorted(set(all\_words))  
tags = sorted(set(tags))  
  
print(len(xy), "patterns")  
print(len(tags), "tags:", tags)  
print(len(all\_words), "unique stemmed words:", all\_words)  
  
# create training data  
X\_train = []  
y\_train = []  
for (pattern\_sentence, tag) in xy:  
 bag = bag\_of\_words(pattern\_sentence, all\_words)  
 X\_train.append(bag)  
 label = tags.index(tag)  
 y\_train.append(label)  
  
X\_train = np.array(X\_train)  
y\_train = np.array(y\_train)  
  
# Hyper-parameters   
num\_epochs = 1000  
batch\_size = 8  
learning\_rate = 0.001  
input\_size = len(X\_train[0])  
hidden\_size = 8  
output\_size = len(tags)  
print(input\_size, output\_size)  
  
class ChatDataset(Dataset):  
  
 def \_\_init\_\_(self):  
 self.n\_samples = len(X\_train)  
 self.x\_data = X\_train  
 self.y\_data = y\_train  
  
 # support indexing such that dataset[i] can be used to get i-th sample  
 def \_\_getitem\_\_(self, index):  
 return self.x\_data[index], self.y\_data[index]  
  
 # we can call len(dataset) to return the size  
 def \_\_len\_\_(self):  
 return self.n\_samples  
  
dataset = ChatDataset()  
train\_loader = DataLoader(dataset=dataset,  
 batch\_size=batch\_size,  
 shuffle=True,  
 num\_workers=0)  
  
device = torch.device('cuda' if torch.cuda.is\_available() else 'cpu')  
  
model = NeuralNet(input\_size, hidden\_size, output\_size).to(device)  
# Train the model  
for epoch in range(num\_epochs):  
 for (words, labels) in train\_loader:  
 words = words.to(device)  
 labels = labels.to(dtype=torch.long).to(device)  
   
  
 outputs = model(words)  
  
 loss = criterion(outputs, labels)  
   
 data = {  
"model\_state": model.state\_dict(),  
"input\_size": input\_size,  
"hidden\_size": hidden\_size,  
"output\_size": output\_size,  
"all\_words": all\_words,  
"tags": tags  
}

This code stream describes how to use a neural network to implement a learning model that can identify the intents of the user for a chatbot. It begins by loading an intent and pattern in a JSON file that is required for the web application to be developed. The code of this process comes up with the result of words and tags creation where the words are tokenized, stemmed, and organized into the list of stemmed words (converted to acceptable form) and the list of tags. This data is then passed through the next stage which involves producing training data where the pattern sentences will be transformed into bag of words. These are the arrays for the training data (or features) and the targets as NumPy arrays to be used when training the neural network.  
  
This follows the definition of the ChatDataset class, which aims at creating a reformulated dataset for the training process to embrace vectorization Return the index of the input and the number of classes in the custom training set. A DataLoader is then utilized to accommodate a large number of samples during the training phase. The model is created using the NeuralNet class while incorporating input size, hidden size, and output size we specified; the model is then sent to the right device, which is either CPU or GPU depending on what we specified. Specifically, the training loop iterates a fixed number of epochs as mentioned above and for each epoch the model loss versus the actual labels is quantified using the cross-entropy loss function. The optimizer (Adam) alters the model weights in a manner that lessens the optimality of the loss. This is a common approach for saving the state of a model and the appropriate parameters for later use – often interpreted as the end of the training process.

4.5.3 Voiceflow Chatbot Implementation

Creating a chatbot with Voiceflow is a revolutionary way, since it does not require any code. What we have to do is just to provide files in order to train it.

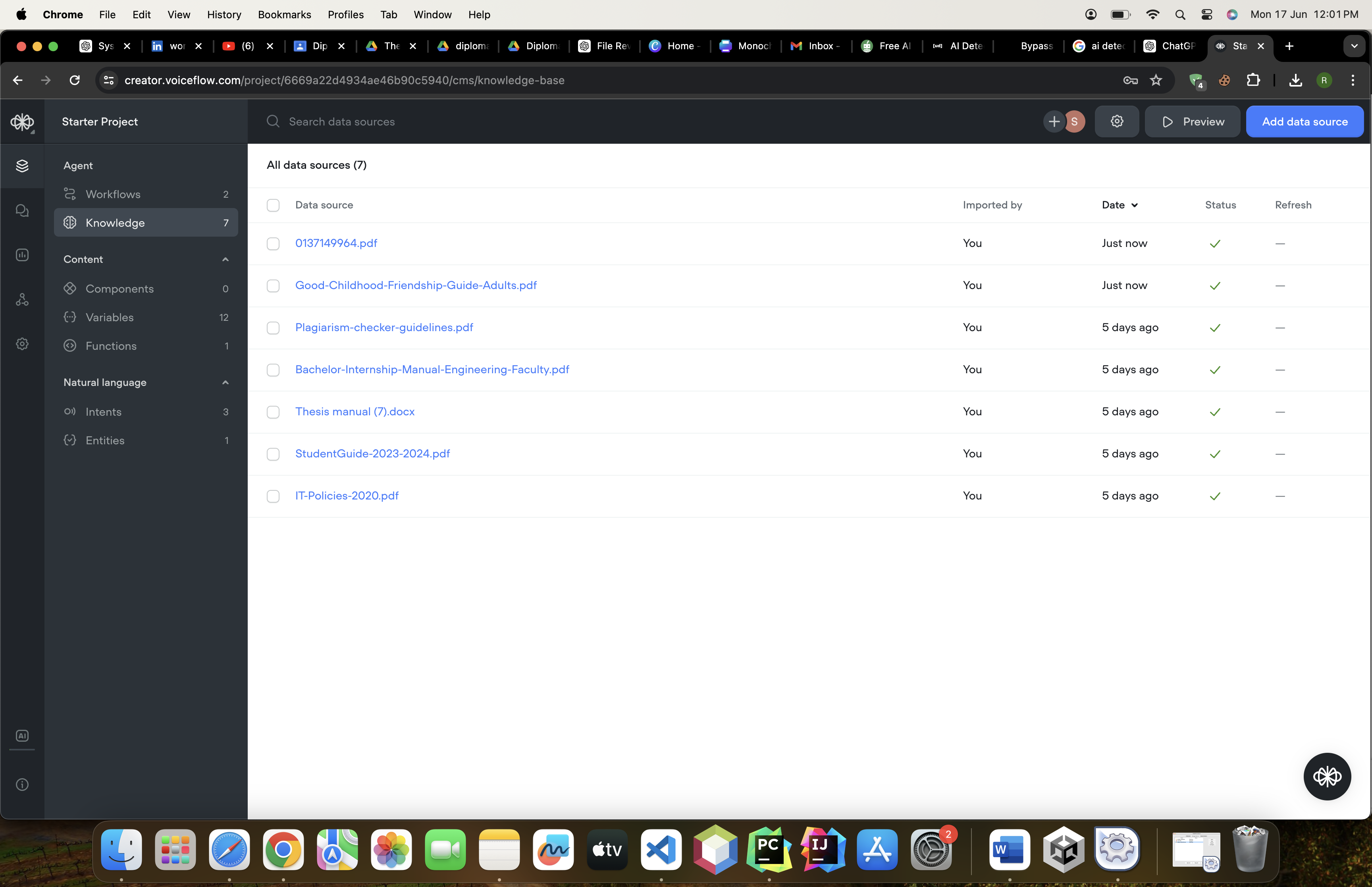


Figure 4.3: Data to train the Voiceflow Chatbot

After creating and training this model, it can be embedded in any website with just a few lines of javascript code:

<script type="text/javascript">  
 (function(d, t) {  
 var v = d.createElement(t), s = d.getElementsByTagName(t)[0];  
 v.onload = function() {  
 ***window***.voiceflow.chat.load({  
 verify: { projectID: '6669a22d4934ae46b90c593f' },  
 url: 'https://general-runtime.voiceflow.com',  
 versionID: 'production'  
 });  
 }  
 v.src = "https://cdn.voiceflow.com/widget/bundle.mjs"; v.type = "text/javascript"; s.parentNode.insertBefore(v, s);  
 })(***document***, 'script');  
</script>

It is not just so easy to implement but it also comes with a nice interface and an outstanding performance. With the technological advancements, this has the potential to grow more in a short period of time. It has reduced the extensive work with no loss of performance at all.

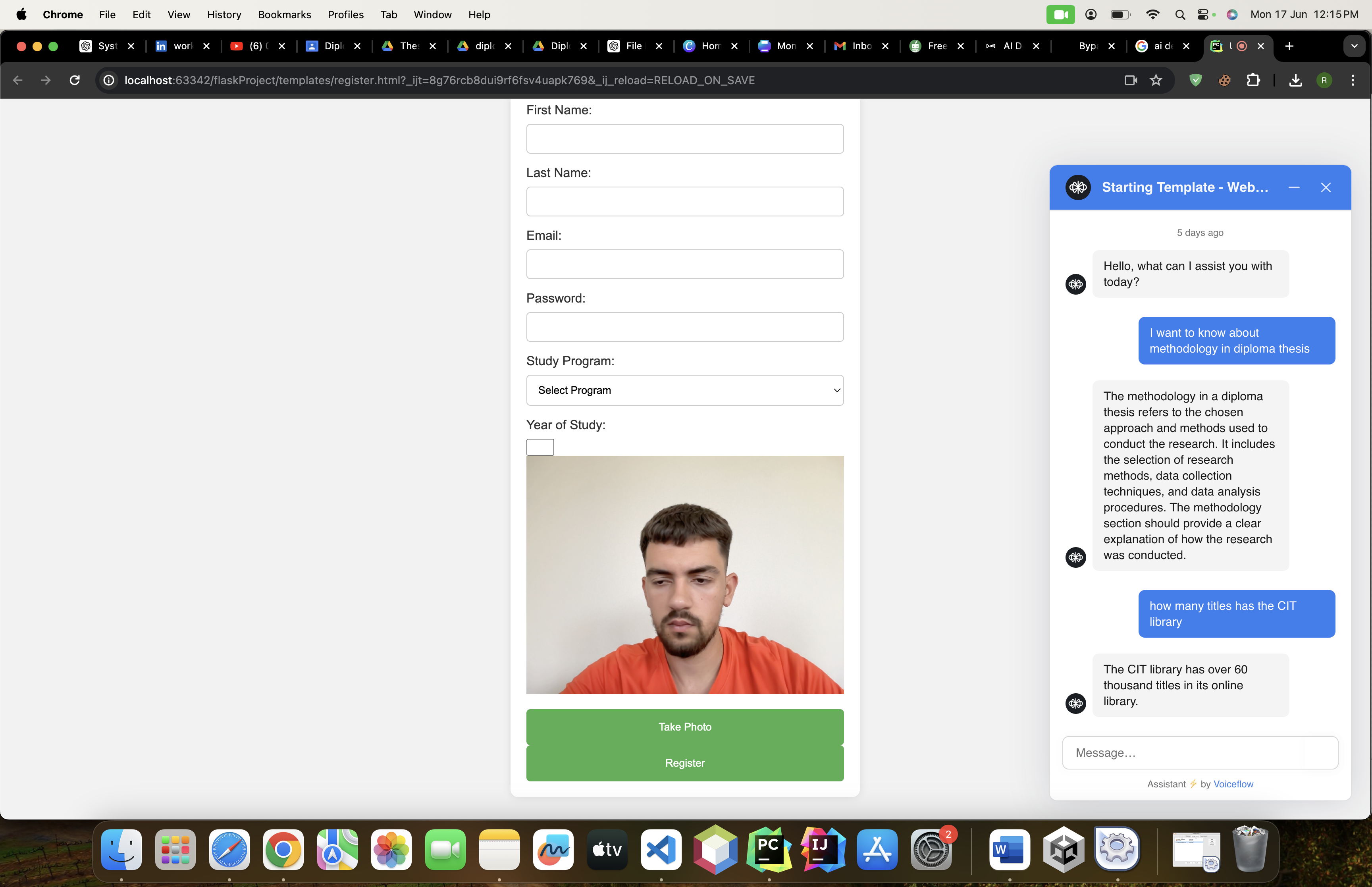


Figure 4.4: Voiceflow Chatbot Interface

**Chapter 5: Challenges and Future Work**

5.1 Development Challenges

This section outlines all the challenges faced during the development of the program.

5.1.1 Integration Challenges

Implementing the various components including the timetable Systems, indoor Navigation, Class Reminders, absence tracker, and the Chatbot raised a number of major concerns. I began this project expecting that all the modules could be created using the same suite of technologies but quickly realized the need for a detailed sequenced plan of how they were going to interact with each other. For instance, synchronizing the Flask backend with the Unity-based indoor navigation system required creating the same Named Parameter Puzzles API and making proper handling of asynchronous request/response data. This meant having in-depth knowledge of the various related technologies and how these could interoperate.   
  
Moreover, the maintenance of data Integration/Inconsistencies of the data in all the designed Modules was an extra challenge. It was necessary to incorporate elaborate error-check and validation mechanisms to prevent loss or distortion of data that was being passed from one module to another. This was important both on the component level and the system level, because stability of some of the parts is vital for the stability of the entire system. Integration was a tough task since it required proper testing and debugging in some circumstances to find out the glitches that may develop during interaction between the independent modules and produce a comprehensive program that could competently deal with actual data and user connections.

5.1.2 Performance Optimization

An important factor to consider in the creation of the AI Virtual Assistant was the need to run at real-time, this made performance optimization to be a key consideration. To make timely notifications, the assistant had to process data concerning the personnel and manage their absence, and this had to be done as efficiently and as fast as possible. The important parameter that needed much optimization was the face recognition module which was implemented using OpenCV and the SSIM constituent. Further adjusting these algorithms was required to optimize the accuracy against the time of their calculations to avoid significant non-rendered time, while the system would identify a student reliably.   
  
Further, stress and load testing’s were performed to demonstrate the feasibility of applying the service under multiple clients loads with no considerable delay or slowness. This was achieved in a way that consisted of recreating peak conditions in order to analyze the applicability of existing knowledge and focus on what needed correction. To improve the performance of the site more optimizations like caching the most requested data the site used to look up in the database was applied, this eliminated the need for repetitive DB calls and improve the response time. Third, an optimal design of the queries was necessary; innovations in query optimizers and better index schemes were useful to reduce the time taken in data access and retrieval. These cumulative activities of performance optimization in AI system guaranteed that the AI Virtual Assistant could run with ease to meet the functionalities of its real-time applications, and effectively offered the users with encounter services.

5.1.3 Security Concerns

Student data during the project had to be protected with high concern with an aim of protecting privacy of students. Stringent security procedures were carefully adhered to in hopes of preserving the identity of all information, including facial recognition images as well as location information. These measures included additional layers of good encryption methods in the transfer and storage of data to avoid tampering and possible leakage of sensitive information.   
  
Some of the key practices that have guided this area of the study include: Compliance with strict data protection laws has been another practice advocated in this area of the study. As a matter of fact, the system was highly developed to adhere to the current laws and regulations that govern the security of such information and permission was granted only to the individuals who have the full rights for that information. It also ensured that student data was protected while ensuring stakeholders were assured of their privacy rights under the given framework.   
  
Security patrols that were conducted at various intervals also included security scrutinization and exhaustive risk appraisal. It was important that such preventive steps were taken, as they would allow the system to recognize risk factors that may lead to vulnerabilities, and eliminate them early to guard against new threats that may emerge over time.   
  
Furthermore, notable resources were channeled toward enhancing the security measures of user credentialing. In an effort to improve on this area, we worked on making the authentication process so strict that only users with authenticated and verified accounts could use the system. It also played a proactive role in minimizing the chances of gaining unauthorized access by other people whereas improving the general integrity of the system’s security.   
  
Thus, let me sum up the security plan as one of the crucial pillars, which were based on the encryption, legislation compliance, anti-virus protection adoption, and resilient user identification. Ensuring that these measures are fully incorporated into the system, we were duty bound to maintaining the highest standards of security and privacy of student data that the system handles at all stages of its cycle.

5.1.4 User Interface

One of the main difficulties when creating the AI Virtual Assistant was ensuring the developed interface design was highly interactive and easily understandable for the end–user, despite the fact that the target market of student users could differ greatly in terms of their level of IT literacy. The initial objective was to maintain tight and friendly interface design to grant all students easy device access regardless of a higher technology literacy. To do this, and incorporate user feedback effectively and iteratively, there was user testing conducted at every step of the process. These tests provided a good amount of feedback within a number of categories with in relation to the parameters of the interface, to determine where the users had problems or became confused.   
  
This feedback was useful in making changes that were applicable in later designs. For example, the navigation was made clear in order for people to locate important sections instead of being overwhelmed by an excess of icons. Another important thing to be taken into consideration was the unity of design across device types, as the user was to interact with the website both using a computer, tablet, and a phone. The design of I-See at the user interface level made effective use of images and textual prompts to ensure that users do not need much time to familiarize with it since instructions were informative enough. Extra emphasis was placed on the mobile web version because most of the interaction by the students with the assistant would likely occur on the small screen of their phones. There are some guidelines that were followed; when developing the application, it was ensured that the interface was adjusted to the screen size and that all key functions were equally easy to reach from a small screen, granting a comfortable and efficient usage of the application from any device.

5.2 Lessons Learned

5.2.1 User Feedback

The most crucial step in the process of developing the final application was feedback from real users, as this form’s reliance on the users’ point of view supports a user-centered approach. Students and educators were included and involved in the development stage of the system, and thus, their requirements and choices were gaining attention. Users’ and experts’ feedback interviews and usability tests had biweekly sessions to obtain their comprehensive opinions and usage experience in pinpointing and discussing exact pain points and areas that need enhancement. For example, users stated that they would like to receive notifications that are delivered based on when they would be free to attend, a more specialized system which is tailored to fit each user’s timetable and priorities and navigational improvements which would allow the users to feel comfortable in the application, and would not give them much trouble in searching for the required features.   
  
This feedback was extremely helpful in formulating the system, and the incorporation of this feedback was effective in helping to improve the system as well. To address concerns that the team received from the end users, changes were made by the development team like improving the avatar customizing and mitigating the complexity of the navigation by reducing the amount of icons used. This backward and forward fashion of collecting data from the users and modifying the system to meet those demands was very effective in ensuring that the final product development was done to the expectations of the clients and users of the system. As a result, the AI Virtual Assistant was fine-tuned and received better acceptance throughout its target consumers since the problems they raised were comprehensively solved and the consumers’ experience was escalated. It also helped instill a sense of ownership over the system among users, meaning that when the system was implemented, most of them accepted it well.

5.2.2 Balancing Features and Usability

One of the key lessons learned was to realize that even with this was to work harder for a balance to be reached between the immense number of features and ease of use. At the beginning critical decisions were leaning heavily towards implementing multitude of features that could extend usability with the innovative cutting-edge technologies. However, with time, the reality struck on the fact that simplicity and user-friendliness were important ingredients to maximize on user satisfaction.   
  
This specific decision of not going for numerous features, which are likely to be non-performance-critical, while concentrating on what most would consider being core functions, known to impact the most the overall use of the product by the client, was the better strategy in comparison to having a long list of preferences. Therefore, by making these core features immediately recognizable and efficiently clickable together with ensuring their stability, we not only optimized the interface but also increased its functionality.   
  
While this feature simplified user interactions, it made the overall system design and it’s malfunctioning more comprehensible and easier to address. In addition, by streamlining the workspace, the team could focus efforts on enhancement of specific functionalities or immediate response to user complaints instead of having to deal with a bloated clutter of features.   
  
In the same spirit, simplicity favoring over usability helped in delivering the product in a more streamlined manner. They made it possible for quicker reviews and improvements on the areas that people felt were inadequate hence placing it in a position of serving as a responsive and continuous improvement system owing to the volatile or changing needs from the users.   
  
Ultimately, the journey emphasized the need to have a good and balanced approach while coming up with new features for having effective usability and user experience while not neglecting core features. This approach not only improved users’ satisfaction but also improved the effectiveness and the robustness of the deployed system.

Conclusion

6.1 Summary of Findings

The creation of the AI Virtual Assistant regarding educational aims and goals has provided a clearer look at how technology can be advanced for the students’ benefit. In general, it has been possible to determine how this project has served to showcase that it is possible to integrate different technological elements to produce a conceptual solution that can support essential areas that are relevant to the academic setting. The following points summarize the main findings: The following points summarize the main findings:   
  
Timetable Integration: An assistant is capable of import student timetables and provide real-time updates and access to individual schedules and plans. This feature has helped the students address the issue of multiclass management, and increase attend KING-Full Article-2013-Selim punctuality. With such an invaluable tool, students can easily manage their time well and also improve on their performance by always having the updated timetable of their classes.   
  
Indoor Navigation and Wayfinding: Applying Unity environment and ARCore, the assistant ensures that students are able to navigate properly within the buildings – get to classes and other places within the campus. So, the given functionality has been really helpful for newcomers and guests to avoid distractions and walk around in search of certain places. This augmented reality guidance system improves the user experience since clients can easily navigate through a store without feeling lost.   
  
Class Reminders: With the send only the relevant and time sensitive reminders along with the location-based reminders, the problem of student absenteeism and lateness have significantly reduced. Custom alerts provide asynchronous notification for students to remind them of their classes in manners that creates an improved readiness and interaction from the learners. This started proactive perspective of reminding students enhances orderliness in the classes, such that reminder enhances efficiency and less chances of missing classes.   
  
Automated Absence Tracking: Besides geolocation services and the mobile face recognition, the assistant can provide a valid solution for tracking a student attendance. This automation aside from cutting down on administrative work also makes employees’ attendance incredibly reliable and non-prejudiced. Modern technologies apply are rather effective to provide accurate attendance figures and eliminate the problem of cheating or manipulating with the data.

Chatbot Interface: Chatbot using PyTorch has been used to establish an interface to access various information and seek help from the reserved resources. The combined features have made it possible for it to meet students’ requirements and offer very vast support through the queries handled. This is because with the help of natural language processing the chatbot provides students with the opportunity to engage in natural discourse with the system which in turn increases user satisfaction and the rate at which students can retrieve information.   
  
In general, the processes involved in the implementation of the AI Virtual Assistant project have demonstrated how the incorporation of existing technologies could improve the general experiences of the students in many ways from managing their time and schedules to physical movement and taking records of attendance. The results of the present research suggest that the type of the systems studied should be extended and integrated into educational environments.

6.2 Future Work

Sophisticated developments in technologies applied to learning remains a core feature of progressive future paradigms that offer optimization possibilities to widespread application spheres. They revealed one key domain of improvement as regards the integration of timetables; the future plans of this system should establish methods for dynamic updating and should be designed to handle more intricate scheduling functions. This encompasses being agile in order to customize for the changes that may occur at the eleventh hour and compatibility with other timetables that are employed in learning institutions. Furthermore, the change in this module could mean giving more specific prompts such as, ‘Hey there’s an assignment due tomorrow’ or ‘There’s an exam next week’, which serves to engage the students as well as help them manage their time more effectively. There are plans to develop indexes of studies with the help of predictive algorithms based on using machine learning which will help to determine efficient schedules of work for every student studying in groups and adapt it to his individual timetable.   
  
Another area where technological development can be witnessed to some extent is the indoor navigation or within educational campuses. The use of AR and development of a smart algorithm by the buddy of machine learning supports the provision of the navigation services. In future, the real time systems can utilize ARCore technology to offer the directions that in turn facilitate the users’ spatial orientation in indoor environments. The mobile environments could be better tracked by machine learning models and on the basis of the user’s preferences and navigation paths, further optimization paths for transportation could be performed to enhance both efficiency and user satisfaction. One would propose that creating an intro of essential campus amenities such as libraries and cafeterias would be a complete strategy that provides for navigation throughout the campus.

Another area which concerns absence tracking can also be viewed as the one where new technological solution can be utilized for improvement. Promises in the innovation in face recognition and geolocation technologies promise to improve the dependability of absence tracking systems. Other possible refinement approaches include, for example, seeking for other ways for verification of the data, like biometric identification or RFID systems as to improve the results and decrease the amount of paperwork. Different capabilities of machine learning algorithms are useful for the additional features like deviations for example the variation in student’s attendance in different classroom at the same time It makes sure it gets all and accurate attendances of students.

This kind of advanced functions of chatbots are a significant step to increase the services offered to students and engagement in learning environments. Chatbots can not only comprehend and address diverse questions from students through applying NLP, but also, they can adapt themselves to respond to those questions efficiently. Future developments could work on providing more meaningful and personalized responses to students with chats such as advising students on study materials that will best suit their performance and passion. These yes/no chatbot applications can be integrated into current and future educational solutions and services, which in turn would enhance the support quality and extend the support process across the educational process.   
  
Surprisingly, these improvements point towards a revolutionary push for creating a better learning space by leveraging technology. As far as developments that prioritize usability, personalization, and integration of the various functionalities in it, these hold the potential of creating meaningful and efficient complementary mechanisms of organization, improved usability, and increased user engagement and purpose, all towards serving the needs of the student’s quest for academic success.

6.3 Final Thoughts

This paper’s AI Virtual assistant is a landmark in the application of incorporating technology in it enhancing the learning process. The use of the system also has the implication of solving some of the main issue encountered by students and teachers; the implications are; enhanced performance, decrease in workload, and an increase in attractiveness of learning environment.   
  
Another of the most significant effects of the implementation of the AI Virtual Assistant is the prospect of saving significant amounts of time spent on various avocations, which in their turn will let the students and the educators concentrate on the basic lines of education. Features like managing timetables, sending alerts in real-time, and using virtual tools for self-assessment helps in minimizing the workload and enhances their efficacy. The indoor navigation and chatbot interface are also effective in enhancing the flow for a fluid user experience where students can easily understand the layout of the campus facilities or gain other necessary information.   
  
As technologies advance, the use of artificial intelligence and machine learning in education will undeniably reign as one of the prominent prospects for the advancement of education. The achievements of this project, in turn, prove the feasibility of such innovations to develop the individual approach to learning and to apply the improvements that contribute to achieving the increase in the students’ inclusiveness. Thus, it is possible to forecast the emergence of better and better AI technologies and their possibilities that will broader the opportunities in the sphere of education even more. In conclusion, this AI Virtual Assistant does not only show that is possible to develop new ways of learning and teaching to apply in the new school setting, but also it provides a basis for fulfilling the potential of new technologies for education.

**References**

Mugavin, B. (2024, January 31). *What is Traditional and Non-Traditional Time Management?* Flashpointleadership.com; FlashPoint Leadership. https://www.flashpointleadership.com/blog/what-is-traditional-and-non-traditional-time-management

‌

‌

*Timetable Management System | School ERP - Fedena*. (2024). Fedena.com. <https://fedena.com/feature-tour/timetable-management-system>

admin. (2021, April 7). *School Timetable Management System for Better Handling of Curriculum*. Edecofy Blog. https://www.edecofy.com/blog/handle-your-school-curriculum-effectively-by-automating-your-school-timetable-management/

‌

*Indoor Navigation for Universities | Enhance the Visitor Experience*. (2021, October 11). NEARMOTION. https://nearmotion.com/news/indoor-navigation-for-universities/

‌

J. Nithya, Vignesh, S. R., A. Devadarsan, & Venkatesh, S. (2022). AI based contactless attendance monitoring and management system. *International Journal of Health Sciences (IJHS) (En Línea)*, 10600–10614. https://doi.org/10.53730/ijhs.v6ns3.8365

‌

Guardian staff reporter. (2018, April 13). *Why are students faking attendance? They feel cheated by the system*. The Guardian; The Guardian. https://www.theguardian.com/higher-education-network/2018/apr/13/why-are-students-faking-attendance-they-feel-cheated-by-the-system

‌

Anyforsoft. (2024, March 8). *Use Cases of AI Chatbots in Universities - AnyforSoft*. AnyforSoft. <https://anyforsoft.com/blog/how-universities-use-ai-chatbots/>

GeeksforGeeks. (2018, October 16). *Introduction to Web development using Flask*. GeeksforGeeks; GeeksforGeeks. https://www.geeksforgeeks.org/python-introduction-to-web-development-using-flask/

‌

*SSIM: Structural Similarity Index | Imatest*. (2020). Imatest.com. https://www.imatest.com/docs/ssim/#:~:text=Introduction%20%E2%80%94%20The%20Structural%20Similarity%20Index,by%20losses%20in%20data%20transmission.

‌

Amruta Kadlaskar. (2021, June 28). *Learn How to Implement Face Recognition Using OpenCV with Python!* Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/06/learn-how-to-implement-face-recognition-using-opencv-with-python/

‌

Wikipedia Contributors. (2024, June 6). *XAMPP*. Wikipedia; Wikimedia Foundation. https://en.wikipedia.org/wiki/XAMPP

‌

*Unity Real-Time Development Platform | 3D, 2D, VR & AR Engine*. (2024). Unity. https://unity.com/

‌

*Build new augmented reality experiences that seamlessly blend the digital and physical worlds  |  ARCore  |  Google for Developers*. (2024). Google for Developers. https://developers.google.com/ar

‌

GeeksforGeeks. (2020, March 24). *How to generate and read QR code with Java using ZXing Library*. GeeksforGeeks; GeeksforGeeks. https://www.geeksforgeeks.org/how-to-generate-and-read-qr-code-with-java-using-zxing-library/

‌

*What is PyTorch?* (2017). NVIDIA Data Science Glossary. https://www.nvidia.com/en-us/glossary/pytorch/

‌

*The collaborative platform to build AI agents*. (2024). Voiceflow.com. <https://www.voiceflow.com/>

*W3Schools.com*. (2022). W3schools.com. https://www.w3schools.com/python/pandas/pandas\_intro.asp#:~:text=Pandas%20is%20a%20Python%20library,by%20Wes%20McKinney%20in%202008.

**Appendixes:**

**Appendix A:** Entire Code for handling the routes in the web application with Flask

os.environ["OPENCV\_AVFOUNDATION\_SKIP\_AUTH"] = "1"  
logging.basicConfig(level=logging.DEBUG)  
  
app = Flask(\_\_name\_\_)  
CORS(app)  
app.config['DEBUG'] = True  
  
# Generate a random secret key  
secret\_key = secrets.token\_hex(24)  
app.secret\_key = secret\_key  
bcrypt = Bcrypt(app)  
  
# Establish MySQL database connection  
database\_connection = mysql.connector.connect(  
 host="localhost",  
 port=3306,  
 user="root",  
 password="",  
 database="Thesis"  
)  
  
# Create a cursor object to execute SQL queries  
cursor = database\_connection.cursor()  
  
# User registration route  
@app.route('/register', methods=['GET', 'POST'])  
def register():  
 if request.method == 'POST':  
 first\_name = request.form['first\_name']  
 last\_name = request.form['last\_name']  
 email = request.form['email']  
 password = bcrypt.generate\_password\_hash(request.form['password']).decode('utf-8')  
 study\_program = request.form['study\_program']  
 year\_of\_study = request.form['year\_of\_study']  
 picture = request.form['photoData']  
  
 # Check if user with the same email already exists  
 cursor.execute("SELECT \* FROM Log WHERE Email = %s", (email,))  
 user = cursor.fetchone()  
 if user:  
 flash("Email already exists. Please choose a different email.", "error")  
 return redirect(url\_for('register'))  
 picture = base64.b64decode(picture.split(',')[1])  
  
 # Insert user into database  
 cursor.execute("INSERT INTO Log (First\_Name, Last\_Name, Program, Study\_Year, Picture, Email, Password) VALUES (%s, %s, %s, %s, %s, %s, %s)",  
 (first\_name, last\_name, study\_program, year\_of\_study, picture, email, password))  
 database\_connection.commit()  
  
  
 # Retrieve the ID of the newly registered user  
 cursor.execute("SELECT user\_id FROM Log WHERE Email = %s", (email,))  
 user\_id = cursor.fetchone()[0]  
  
 # Store user ID and email in session  
 session['user\_id'] = user\_id  
 session['email'] = email  
  
 flash("Registration successful. Please log in.", "success")  
 return redirect(url\_for('login'))  
  
 return render\_template('register.html')  
  
@app.post("/predict")  
def predict():  
 text = request.get\_json().get("message")  
 response = get\_response(text)  
 message = {"answer": response}  
 return jsonify(message)  
  
# User login route  
@app.route('/login', methods=['GET', 'POST'])  
def login():  
 if request.method == 'POST':  
 email = request.form['email']  
 password = request.form['password']  
  
 # Query to fetch user data by email from the Log table  
 cursor.execute("SELECT user\_id, First\_Name, Last\_Name, Email, Password FROM Log WHERE Email = %s", (email,))  
 user = cursor.fetchone()  
  
 if user:  
 user\_id, first\_name, last\_name, email, hashed\_password = user  
  
 # Check the password hash  
 if bcrypt.check\_password\_hash(hashed\_password, password):  
 session['user\_id'] = user\_id  
 session['first\_name'] = first\_name  
 session['last\_name'] = last\_name  
 return redirect(url\_for('profile'))  
 else:  
 return "Invalid credentials"  
 else:  
 return "Invalid credentials"  
 return render\_template('login.html')  
  
@app.route('/profile')  
def profile():  
 # Check if user is logged in  
 if 'user\_id' in session:  
 user\_id = session['user\_id']  
 # Fetch user data from database using the user ID stored in the session  
 cursor.execute("SELECT \* FROM Log WHERE user\_id = %s", (user\_id,))  
 user = cursor.fetchone()  
 if user:  
 # Pass user data to the profile template  
 return render\_template('profile.html', user=user)  
 else:  
 flash("User not found", "error")  
 return redirect(url\_for('login'))  
 else:  
 flash("Please log in to view this page", "error")  
 return redirect(url\_for('login')) # Redirect to login page if user is not logged in  
  
@app.route('/indoor\_navigation.html')  
def indoor\_navigation():  
 return render\_template('indoor\_navigation.html')  
  
@app.route('/absence\_taker')  
def absence\_taker():  
 # Check if user is logged in  
 if 'user\_id' in session:  
 user\_id = session['user\_id']  
 return render\_template('absence\_taker.html', user\_id=user\_id) # Pass user\_id to absence\_taker.html  
 else:  
 flash("Please log in to view this page", "error")  
 return redirect(url\_for('login')) # Redirect to login page if user is not logged in  
  
# Route to handle attendance taking  
@app.route('/take\_attendance', methods=['POST'])  
def handle\_attendance():  
 if 'user\_id' in session:  
 user\_id = session['user\_id']  
 latitude = session.get('latitude')  
 longitude = session.get('longitude')  
 if latitude is None or longitude is None:  
 return jsonify({"error": "Missing location data"}), 400  
  
 # Call the take\_attendance function  
 result = attendance\_taker.take\_attendance(latitude, longitude, user\_id)  
 # Return the result as JSON response  
 return jsonify(result)  
 else:  
 return jsonify({"error": "User not logged in"}), 401  
  
@app.route('/submit\_location', methods=['POST'])  
def submit\_location():  
 try:  
 data = request.get\_json()  
 latitude = data.get('latitude')  
 longitude = data.get('longitude')  
  
 if latitude is None or longitude is None:  
 return jsonify({"error": "Missing latitude or longitude"}), 400  
  
 # Save location in session or handle as needed  
 session['latitude'] = latitude  
 session['longitude'] = longitude  
 return jsonify({"success": True}), 200  
 except Exception as e:  
 return jsonify({"error": str(e)}), 500  
  
@app.route('/start\_attendance', methods=['POST'])  
def start\_attendance():  
 try:  
 if 'user\_id' not in session:  
 return jsonify({"error": "User not logged in"}), 401  
  
 user\_id = session['user\_id']  
 latitude = session.get('latitude')  
 longitude = session.get('longitude')  
  
 if latitude is None or longitude is None:  
 return jsonify({"error": "Missing location data"}), 400  
  
 print(f"Starting attendance with latitude: {latitude}, longitude: {longitude}, user\_id: {user\_id}")  
 result = attendance\_taker.take\_attendance(latitude, longitude, user\_id)  
 return jsonify({"message": "Attendance process completed", "result": result}), 200  
 except Exception as e:  
 print(f"Error in /start\_attendance: {e}") # Debugging line  
 return jsonify({"error": str(e)}), 500  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app.run(  
 debug=True, passthrough\_errors=True,  
 use\_debugger=False, use\_reloader=False  
 )

**Appendix B:** Entire Code for absence tracking and reading the timetable

import cv2  
import numpy as np  
import mysql.connector  
from datetime import datetime, timedelta  
from shapely.geometry import Point, Polygon  
import pandas as pd  
import logging  
import os  
from skimage.metrics import structural\_similarity as ssim  
from cv2 import data  
  
# Set environment variable to skip OpenCV AVFoundation authorization  
os.environ["OPENCV\_AVFOUNDATION\_SKIP\_AUTH"] = "1"  
  
# Configure logging  
logging.basicConfig(level=logging.DEBUG)  
  
# Load the pre-trained face detection model  
face\_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade\_frontalface\_default.xml')  
  
  
recognizer = cv2.face.LBPHFaceRecognizer\_create()  
model\_path = 'models/face\_recognizer.yml'  
if os.path.exists(model\_path):  
 recognizer.read(model\_path)  
else:  
 logging.error("Face recognition model not found. Please train the model first.")  
 raise FileNotFoundError("Face recognition model not found. Please train the model first.")  
  
# Initialize MySQL database connection  
db\_connection = mysql.connector.connect(  
 host="localhost",  
 user="root",  
 password="",  
 database="Thesis"  
)  
cursor = db\_connection.cursor()  
  
# Load timetable from CSV file  
timetable = pd.read\_csv("timetable.csv")  
  
def get\_current\_course():  
 now = datetime.now()  
 current\_time = now.strftime("%H:%M")  
 current\_day = now.strftime("%A")  
  
 logging.debug(f"Looking for current course at time {current\_time} on {current\_day}")  
 for index, row in timetable.iterrows():  
 if row['Day'] == current\_day:  
 for time\_slot in ['09:00 - 11:00', '11:00 - 13:00', '13:00 - 15:00']:  
 try:  
 start\_time, end\_time = time\_slot.split(' - ')  
 start\_time = start\_time.strip()  
 end\_time = end\_time.strip()  
 if start\_time <= current\_time <= end\_time:  
 course = row[time\_slot]  
 if pd.notna(course):  
 logging.debug(f"Found current course: {course}")  
 return course, start\_time  
 except ValueError as e:  
 logging.error(f"Error parsing time slot '{time\_slot}': {e}")  
 logging.debug(f"No course found for current time {current\_time} on {current\_day}")  
 return None, None  
  
def record\_attendance(user\_id, course):  
 now = datetime.now()  
 time = now.strftime('%Y-%m-%d %H:%M:%S')  
  
 try:  
 cursor.execute("INSERT INTO Attendance (user\_id, time, course) VALUES (%s, %s, %s)",  
 (user\_id, time, course))  
 db\_connection.commit()  
 logging.debug(f"Recorded attendance for user\_id {user\_id} in course {course} at {time}")  
 except mysql.connector.Error as err:  
 logging.error(f"Error recording attendance: {err}")  
  
def get\_user\_photo(user\_id):  
 try:  
 cursor.execute("SELECT Picture FROM Log WHERE user\_id = %s", (user\_id,))  
 result = cursor.fetchone()  
 if result:  
 photo\_data = result[0]  
 return np.frombuffer(photo\_data, np.uint8)  
 else:  
 logging.debug(f"No photo found for user\_id {user\_id}")  
 return None  
 except mysql.connector.Error as err:  
 logging.error(f"Error retrieving user photo: {err}")  
 return None  
  
def is\_user\_within\_rectangle(user\_coords, rectangle\_coords):  
 polygon = Polygon(rectangle\_coords)  
 point = Point(user\_coords)  
 is\_within = polygon.contains(point)  
 logging.debug(f"User coordinates {user\_coords} within rectangle: {is\_within}")  
 return is\_within  
  
rectangle\_coords = [  
 (41.35, 19.78),  
 (41.35, 19.8842004),  
 (41.30, 19.8842004),  
 (41.30, 19.78),  
 (41.35, 19.78)  
]  
  
  
def compare\_images(img1, img2):  
 try:  
 s = ssim(img1, img2)  
 logging.debug(f"SSIM: {s}")  
 return s > 0.4 # Adjust threshold as needed  
 except Exception as e:  
 logging.error(f"Error during image comparison: {e}")  
 return False  
  
def take\_attendance(latitude, longitude, user\_id):  
 user\_id = user\_id  
 current\_class, start\_time\_str = get\_current\_course()  
 if current\_class and start\_time\_str:  
 start\_time = datetime.strptime(start\_time\_str, "%H:%M").replace(year=datetime.now().year, month=datetime.now().month, day=datetime.now().day)  
 attendance\_end\_time = start\_time + timedelta(minutes=110)  
 now = datetime.now()  
  
 logging.debug(f"Class start time: {start\_time\_str}, Attendance window end time: {attendance\_end\_time.strftime('%H:%M')}")  
 if start\_time <= now <= attendance\_end\_time:  
 cap = None  
 try:  
 cap = cv2.VideoCapture(0)  
 if not cap.isOpened():  
 logging.error("Failed to initialize the webcam. Exiting...")  
 return {"error": "Failed to initialize the webcam"}  
  
 while True:  
 ret, frame = cap.read()  
 if not ret:  
 logging.error("Failed to capture frame from webcam. Exiting...")  
 break  
  
 gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)  
 faces = face\_cascade.detectMultiScale(gray, scaleFactor=1.2, minNeighbors=5, minSize=(30, 30))  
 logging.debug(f"Faces detected: {faces}")  
  
 if len(faces) == 0:  
 logging.debug("No faces detected.")  
 continue  
  
 for (x, y, w, h) in faces:  
 cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)  
 roi\_gray = gray[y:y + h, x:x + w]  
 try:  
 logging.debug(f"Predicting face at position {(x, y, w, h)} with ROI shape: {roi\_gray.shape}")  
 id\_, confidence = recognizer.predict(roi\_gray)  
 logging.debug(f"Detected face with id {id\_} and confidence {confidence}")  
 except cv2.error as e:  
 logging.error(f"OpenCV error during face prediction: {e}")  
 continue  
 except Exception as e:  
 logging.error(f"General error during face prediction: {e}")  
 continue  
  
 if confidence < 130:  
 user\_id = id\_  
 photo = get\_user\_photo(user\_id)  
 if photo is not None:  
 try:  
 user\_img = cv2.imdecode(photo, cv2.IMREAD\_COLOR)  
 if user\_img is None:  
 logging.error("Failed to decode user image")  
 continue  
  
 user\_img\_gray = cv2.cvtColor(user\_img, cv2.COLOR\_BGR2GRAY)  
 user\_img\_gray\_resized = cv2.resize(user\_img\_gray, (w, h))  
  
 # Save images for visual inspection  
 cv2.imwrite('detected\_roi.png', roi\_gray)  
 cv2.imwrite('user\_img\_gray.png', user\_img\_gray)  
 cv2.imwrite('user\_img\_gray\_resized.png', user\_img\_gray\_resized)  
  
 # Normalize images  
 roi\_gray\_normalized = cv2.equalizeHist(roi\_gray)  
 user\_img\_gray\_resized\_normalized = cv2.equalizeHist(user\_img\_gray\_resized)  
  
 # Log shapes and comparison  
 logging.debug(f"Detected ROI shape: {roi\_gray.shape}, User image shape: {user\_img\_gray.shape}, Resized user image shape: {user\_img\_gray\_resized.shape}")  
 logging.debug(f"Detected ROI mean: {roi\_gray.mean()}, Resized user image mean: {user\_img\_gray\_resized.mean()}")  
 logging.debug(f"Normalized ROI mean: {roi\_gray\_normalized.mean()}, Normalized resized user image mean: {user\_img\_gray\_resized\_normalized.mean()}")  
  
 # Compare images using SSIM  
 if compare\_images(roi\_gray\_normalized, user\_img\_gray\_resized\_normalized):  
 user\_coords = (latitude, longitude)  
 if is\_user\_within\_rectangle(user\_coords, rectangle\_coords):  
 logging.debug("User is within the attendance area")  
 course = current\_class  
 record\_attendance(user\_id, course)  
 cap.release()  
 cv2.destroyAllWindows()  
 return {"success": "Attendance taken successfully"}  
 else:  
 logging.debug("User is not within the attendance area")  
 else:  
 logging.debug("Face does not match with the user's photo")  
 except cv2.error as e:  
 logging.error(f"OpenCV error during photo processing: {e}")  
 except Exception as e:  
 logging.error(f"General error during photo processing: {e}")  
 else:  
 logging.debug("User's photo not found")  
 else:  
 logging.debug("Face not recognized")  
  
 cv2.imshow('Frame', frame)  
 if cv2.waitKey(1) & 0xFF == ord('q'):  
 break  
  
 except Exception as e:  
 logging.error(f"Error in take\_attendance: {e}")  
 return {"error": f"Error in take\_attendance: {e}"}  
 finally:  
 if cap is not None and cap.isOpened():  
 cap.release()  
 cv2.destroyAllWindows()  
 else:  
 logging.debug("Current time is outside the attendance window.")  
 return {"error": "Current time is outside the attendance window."}  
 else:  
 logging.debug("No current class found for attendance.")  
 return {"error": "No current class found for attendance."}

**Appendix C:** Picture from the production of the app

