UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA
DIPARTIMENTO DI ECCELLENZA

Corso Enterprise Mobile Application Development

Progetto: UniVerse
Requirement Analysis Document
Versione 1.01

Data: 21/11/2023

Progetto: UniVerse	Versione: **1.1**
Documento: Requirement Analysis Document	Data: **21/11/2023**

Coordinatore del progetto:

Nome	
Luca Greci	
Alessio Celli	
Alessandro Vergari	

Partecipanti:

Nome	**Matricola**
Ferdinando Boccia	0522501750
Giovanni Cascone	0522501742
Said Trujillo	ERASMSIN03798

Scritto da:	Ferdinando Boccia - Giovanni Cascone – Said Trujillo

Revision History

Data	**Versione**	**Descrizione**	**Autore**

INDEX

INTRODUCTION 4
1.1. PROBLEM DOMAIN 4
1.2. SYSTEM GOALS 4
1.3. SUCCESS CRITERIA 4
PROPOSED SYSTEM 4
2.1. FUNCTIONAL REQUIREMENTS 5
2.2. NON-FUNCTIONAL REQUIREMENTS 6
SYSTEM MODEL 6
3.1. SCENARIOS 6
3.2. USE CASE MODEL 7
3.3. USE CASE 7
OBJECT MODEL 9
4.1. CLASS DIAGRAM 9
4.2. ENTITY CLASS DIAGRAM 9
PROTOTYPE 10
5.1. NAVIGATIONAL DIAGRAM 10
5.2. MOCKS UP 12
5.2.1. WEB APPLICATION INTERFACE 12
5.2.2. VR APPLICATION INTERFACE 13
BUSINESS CANVASS MODEL 14

INTRODUCTION

1.1. PROBLEM DOMAIN

The problem objective is to implement a ChatBot intended for university students. It must be able to handle requests regarding class schedules, classroom occupancy, and more, responding accurately by providing the correct information.

The ChatBot can be used through two different modes, namely Web Interface and Virtual Reality (VR) systems. The service offered by the two different platforms will be the same, differing in the mode of display and interaction.

Different technologies and methodologies aimed at the implementation of ChatBot will be studied appropriately, evaluating which one is the most suitable for the system under development.

1.2. SYSTEM GOALS

- **Improve User Experience**: Simplify the access process to crucial information for students and professors.
- **Provide Instant Support**: Offer instant support through the chatbot for frequently asked questions, reducing the need for human interactions for routine issues.
- **Promote the Use of Advanced Technologies**: Leverage technologies such as VR to provide an innovative and engaging experience for users.
- **Use Text-To-Speech / Speech-To-Text engines**: Ensure communication with the ChatBot through the use of voice.
- **Implement Lip-Sync mechanisms**: Ensure greater realism to the communication system with the 3D virtual assistant, in the case of use through VR platform.

1.3. SUCCESS CRITERIA

- The application must have a **low response time**.
- The application must be **user friendly**, **easy to use**, and **intuitive**.
- **Continuous Updates and Improvements**: Monitor hao often the chatbot is updated and improved based on user needs.

PROPOSED SYSTEM

The multi-channel app with ChatBot for university support is an innovative system designed to simplify the experience of students, professors, and university staff by providing quick and easy access to crucial information through different platforms, such as websites and Virtual Reality (VR) devices.

The ChatBot will be presented in the form of an Avatar, with which the user will be able to interact through text (only in the case of use via Web App) and voice.** Differentiation between Stardard and Logged-in User has LOW priority (will not necessarily be developed).

The proposed system will be usable by different types of users: Standard User and Logged-in User. The **Standard User** is an anonymous visitor to the multichannel app with chatbot for university support. This type of user can access a wide range of features without logging in or creating an account in the system. The Standard User can:

- Chatbot Assistance: Interact with the chatbot to get quick answers and instant information on various university topics: class schedules, classroom occupancy, cafeteria meals, etc.
- Limited Access to Notifications: Receive general notifications or public alerts related to the university without customizations based on a specific profile. ??

?? The **Logged-in User**, on the other hand, is a user who has created an account in the system, providing personal data and accessing the app through his or her own credentials. The main differences for the logged-in user include:

- **Personal Profile Management**: The logged-in user can create and manage his or her own personal profile. This includes updating personal information such as name, email, course of study, and notification preferences.
- **Custom Notifications**: Can customize notification preferences, choosing to receive specific alerts such as events, changes to class schedule, and other important communications. ??

In summary, the standard user can enjoy generic services and basic information without the need for an account, while the logged-in user enjoys a more comprehensive personalized experience, with access to advanced features and specific notifications based on their profile and preferences.

2.1. FUNCTIONAL REQUIREMENTS

Legend:

- HIGH PRIORITY: IMPLEMENTATION REQUIRED
- LOW PRIORITY: IMPLEMENTATION OPTIONAL
- ?? = NEED TO EVALUATE WHETHER IT MAY BE USEFUL TO IMPLEMENT IT

User Profile Management: ??

RF_USER_01: Users should be able to create and manage their own user profiles, including information such as name, email address, course of study, and notification preferences. (**LOW PRIORITY**)

RF_USER_02: The system must ensure the security of user data and comply with privacy regulations. (**LOW PRIORITY**)

Chatbot Support:

RF_HELP_01: The chatbot must answer user questions clearly and accurately. Questions can be on a variety of topics, such as searching for class schedules, free classrooms, or today's meals in the cafeteria.(**HIGH PRIORITY**)

RF_HELP_02: The chatbot must be able to handle frequently asked questions on topics such as enrollment, schedules, events, and other university information. (**HIGH PRIORITY**)

Notifications and Alerts: ??

RF_ALERT_01: The app must send notifications to users about changes in class schedule, important alerts, and other relevant information. (**LOW PRIORITY**)

RF_ALERT_02: Users must be able to manage notification preferences through the app. (**LOW PRIORITY**)

Multi-Channel Support:

RF_MCS_01: The app must be accessible via Web and VR devices, ensuring a consistent experience across all platforms. (**HIGH PRIORITY**)

2.2. NON-FUNCTIONAL REQUIREMENTS

- **Usability**: The app must have an intuitive user interface that is easy to navigate to ensure an effortless experience for the standard user.
- **Performance**: The system must ensure fast response times for standard user requests, both for information retrieval and chatbot responses.
- **Reliability**: The system should always be online with low risk of service crashes.

SYSTEM MODEL

3.1. SCENARIOS

Scenario 1: Class Schedule Search
Participants: User (standard or logged in), System
Description: The user asks the chatbot to provide the class schedule for a specific course. The system processes the request, retrieves the timetable from the database, and returns the class schedule for the selected course to the user.
Event flow:

1. User:
   - Logs into the app and asks the chatbot for the class schedule using text or voice as inputs.
   - Asks for the course name or course code for which they want to know the class schedule in the next semester.
2. System:
   - Performs a search in the system using the course name or code provided by the user.
   - Retrieves the class schedule for the specified course in the next semester.
   - The system will respond to the user through the use of a text-to-speech engine.
3. User:
   - View/Listen the class schedule for the searched course.
   - Explore schedule information, including days of the week, class start and end times, and location of classrooms.
4. Additional Options:
   - If the user wants more details, he/she can request information regarding a specific class to view the professor's name, exact classroom, and other related information.

Scenario 2: Conversation with chatbot via Virtual Reality Visor
Participants: User (standard or logged in), System
Description: The user wants to have a conversation with the chatbot using a virtual reality viewer for a more immersive experience.
Event Flow:

1. User:
   - Wears the virtual reality viewer and accesses the dedicated application inside the viewer.
   - User enters a virtual environment viewing the avatar with which he/she can interact.
   - The user converses with the avatar(chatbot) by asking questions either through text or

using voice.

2. System:
    - The system interacts through the virtual assistant avatar by answering questions using a text-to-speech engine.
3. User:
    - View/listens to the assistant's response and decides whether to stop or continue with more questions.

3.2. USE CASE MODEL

WebUser refers to users who connect to the web interface. **VRUser**, on the other hand, applies to those users who participate in the virtual reality environment. Both users implement ***User***, allowing them to perform written and audio inquiries. In addition, they have the ability to enable or disable the playback of audio responses, configure language settings and select frequently asked questions.

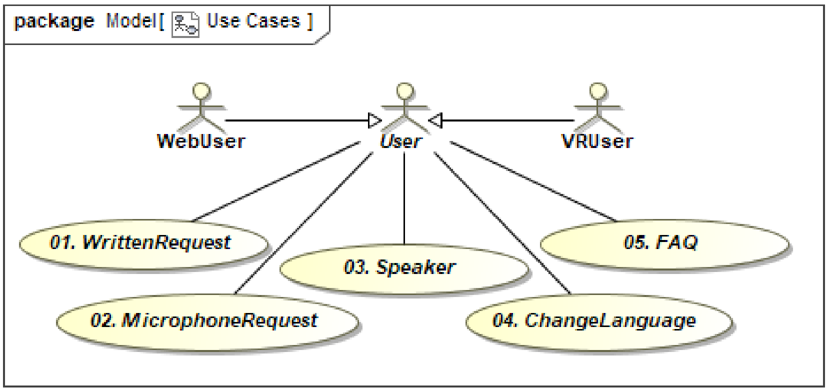


Figure 1: Use case diagram in which the User actor is an abstract class which will be implemented in both WebUser and VRUser.

3.3. USE CASE

Name:	**WrittenRequest**
Description:	**User types a request via keyboard.**
ID:	**UC01**
Actors:	***User***
Event Flow:	A. The *System* displays a welcome message. B. The *User* enters a request in the inbox. C. The *System* sends the request to the *Server*. D. The *Server* returns a response. E. The *System* converts the text to audio. F. The *System* displays the response and plays the audio. G. The *System* returns to step B.
Entry Condition:	No condition.
Output Condition:	No condition.
Extension:	The audio playback option is disabled: Variant E. 1 • The *System* displays the response. • The *System* returns to step B.

Name:	**MicrophoneRequest**
Description:	**User makes a request to the bot via microphone.**
ID:	**UC02**
Actors:	***User***
Event Flow:	A. The *User* clicks the microphone button. B. The *System* records the audio request. C. The *System* converts the audio to text. D. The *System* sends the request to the *Server*. E. The *Server* returns a response. F. The *System* converts the text to audio. G. The *System* displays the response and plays the audio. H. The *System* returns to step A.
Entry Condition:	No condition.
Output Condition:	No condition.
Extension:	The audio playback option is disabled: Variant F. 1 • The *System* displays the response. • The *System* returns to step A.

Name:	**Speaker**
Description:	**Enables or disables the playback of speaker feedback.**
ID:	**UC03**
Actors:	***User***
Event Flow:	A. The *User* clicks the speaker button. B. The *System* activates audio playback. C. The *System* ends the use case.
Entry Condition:	No condition.
Output Condition:	Audio playback status has changed.
Extension:	Audio playback was already activated: Variant B. 1 • The *System* deactivates audio playback. • The *System* goes to step C.

Name:	**ChangeLanguage**
Description:	**Modifies the language of the conversation.**
ID:	**UC04**
Actors:	***User***
Event Flow:	A. The *User* clicks on the language button. B. The *System* displays a list of languages to choose from. C. The *User* selects the language of interest. D. The *System* saves the selection.
Entry Condition:	No condition.
Output Condition:	The language selection selected by the user is saved.
Extension:	No extension.

Name:	**FAQ**
Description:	**The system displays a set of frequently asked questions to the user.**
ID:	**UC05**
Actors:	***User***
Event Flow:	A. The *System* displays a set of frequently asked questions. B. The *User* selects one of these questions.

	C. The *System* sends the request to the *Server*. D. The *Server* returns a response. E. The *System* displays the response to the *User*. F. The *System* returns to step B.
Entry Condition:	No condition.
Output Condition:	No condition.
Extension:	No extension.

OBJECT MODEL

4.1. CLASS DIAGRAM

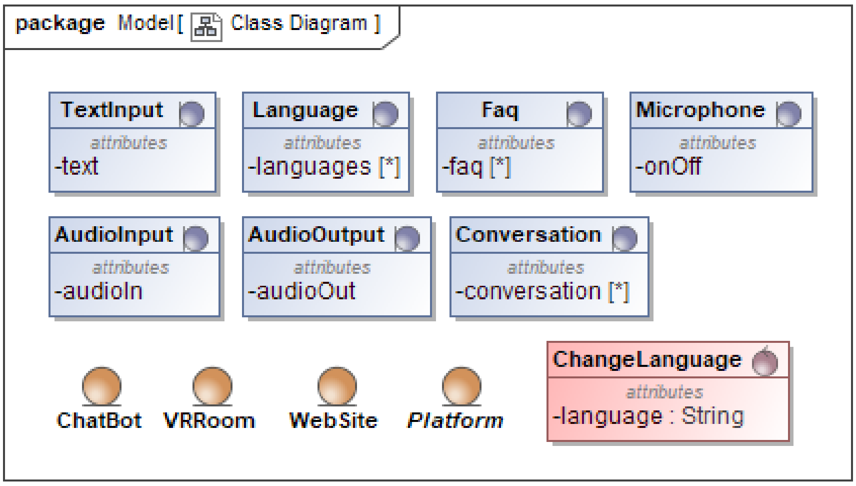


Figure 2. Class diagram

Since the application lacks databases dedicated to interaction with users, the number of control classes is limited to the one designed to modify the language of the conversation. On the other hand, a large set of boundary classes designed for direct interaction with users is identified. These classes facilitate the input of information by the user and the subsequent response of the application. Finally, there are the entity classes, which will be discussed in detail in later sections.

4.2. ENTITY CLASS DIAGRAM

The initial entity class structure of the system consists of four main classes:

- ***Platform***: this class is abstract and will be implemented by the WebSite and VRRoom classes. It contains attributes such as maxChar (to limit the number of characters that can be sent to the chatbot simultaneously), history (stores the strings present in the conversation), language and faq (an enumeration of frequently asked questions). In addition, it includes a set of methods that must be implemented by the classes that inherit from it.
- **WebSite**: In charge of the user's interaction with the web page. It contains methods and attributes to initialize the web interface, as well as those that must be implemented by the *Platform* class.
- **VRRoom**: Similar to the WebSite class, but responsible for virtual reality. It has methods and attributes to initialize the graphic interface. In addition, it includes a messageCounter variable

that identifies the current message, allowing to keep a history of the conversation without occupying the whole interface.

- **ChatBot**: It refers to the response generator. It has a set of references to *Platform* that allows it to know the source of a query and to which one it should return a response. As shown in Figure 3, the system will have a single instance of ChatBot and several instances of *Platform*.

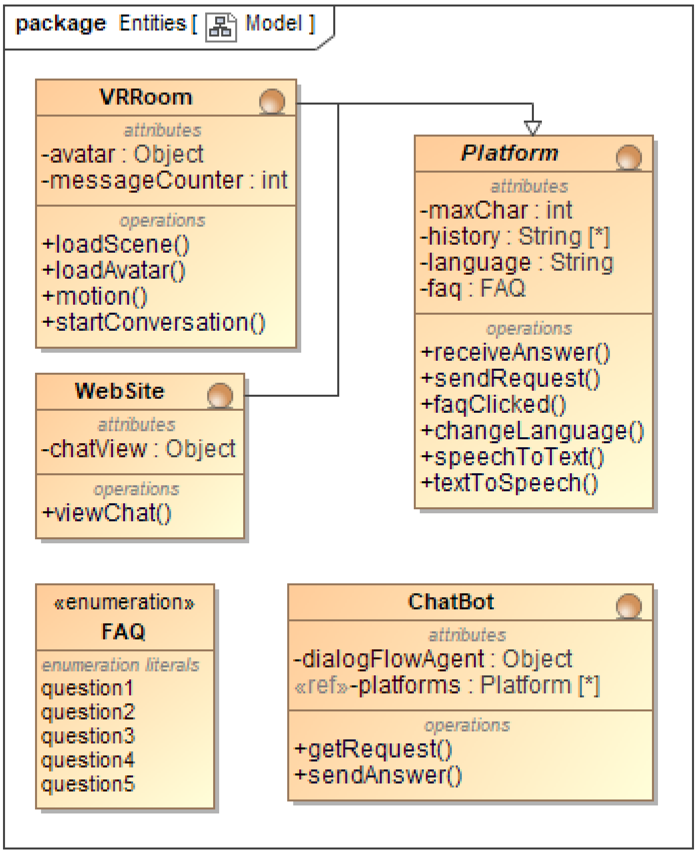


Figure 3. Class diagram

PROTOTYPE

5.1. *NAVIGATIONAL DIAGRAM*

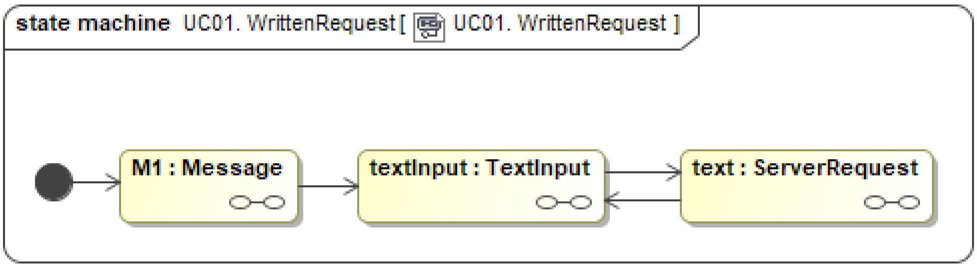


Figure 4. Navigation diagram corresponding to text inquiries

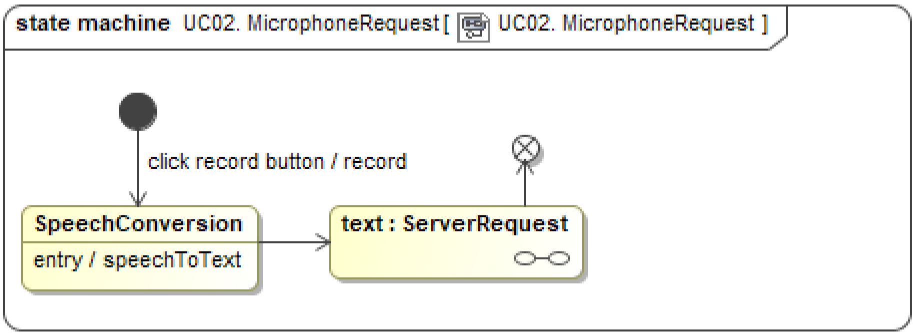

state machine UC02. MicrophoneRequest [UC02. MicrophoneRequest]
click record button / record
SpeechConversion
entry / speechToText
text : ServerRequest

Figure 5. Navigation diagram corresponding to audio inquiries

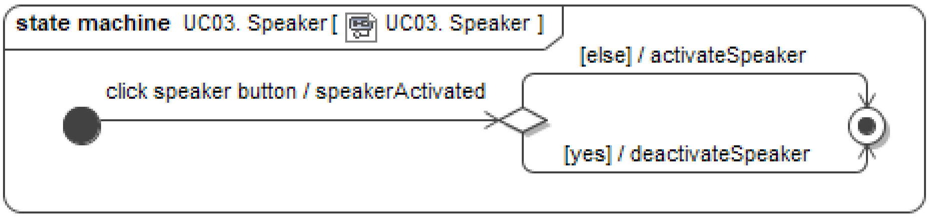

state machine UC03. Speaker [UC03. Speaker]
click speaker button / speakerActivated
[else] / activateSpeaker
[yes] / deactivateSpeaker

Figure 6. Navigation diagram corresponding to the enable/disable audio playback action

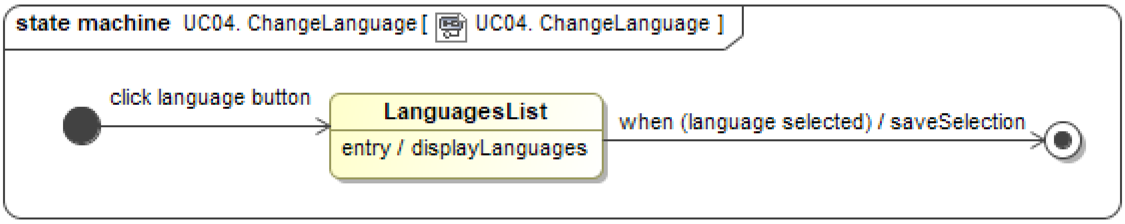

state machine UC04. ChangeLanguage [UC04. ChangeLanguage]
click language button
LanguagesList
entry / displayLanguages
when (language selected) / saveSelection

Figure 7. Navigation diagram corresponding to the language change action

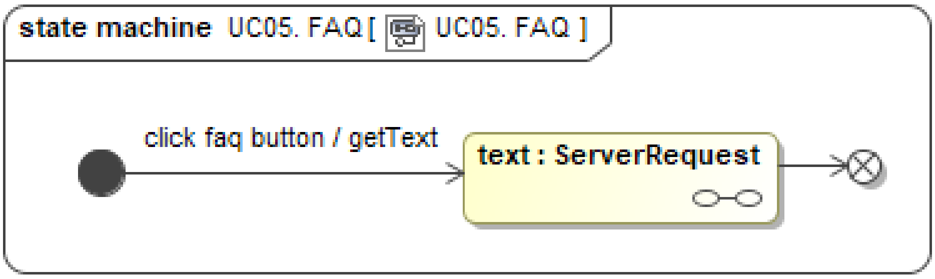

state machine UC05. FAQ [UC05. FAQ]
click faq button / getText
text : ServerRequest

Figure 8. Navigation diagram corresponding to the action of detecting the faq selection

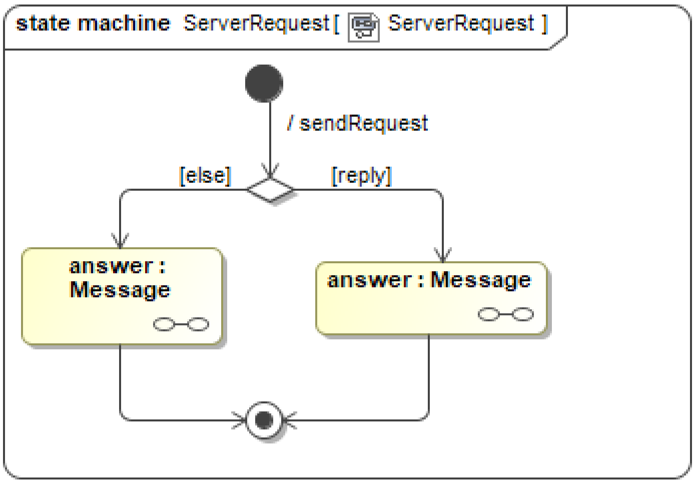


Figure 9. Navigation diagram corresponding to the text inquiries sent to the server

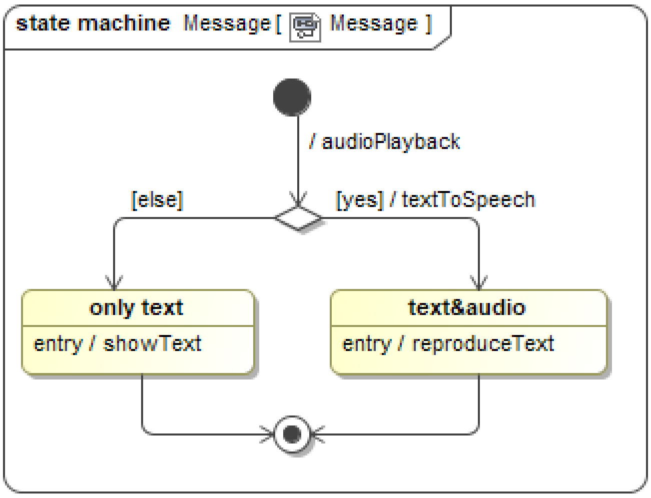
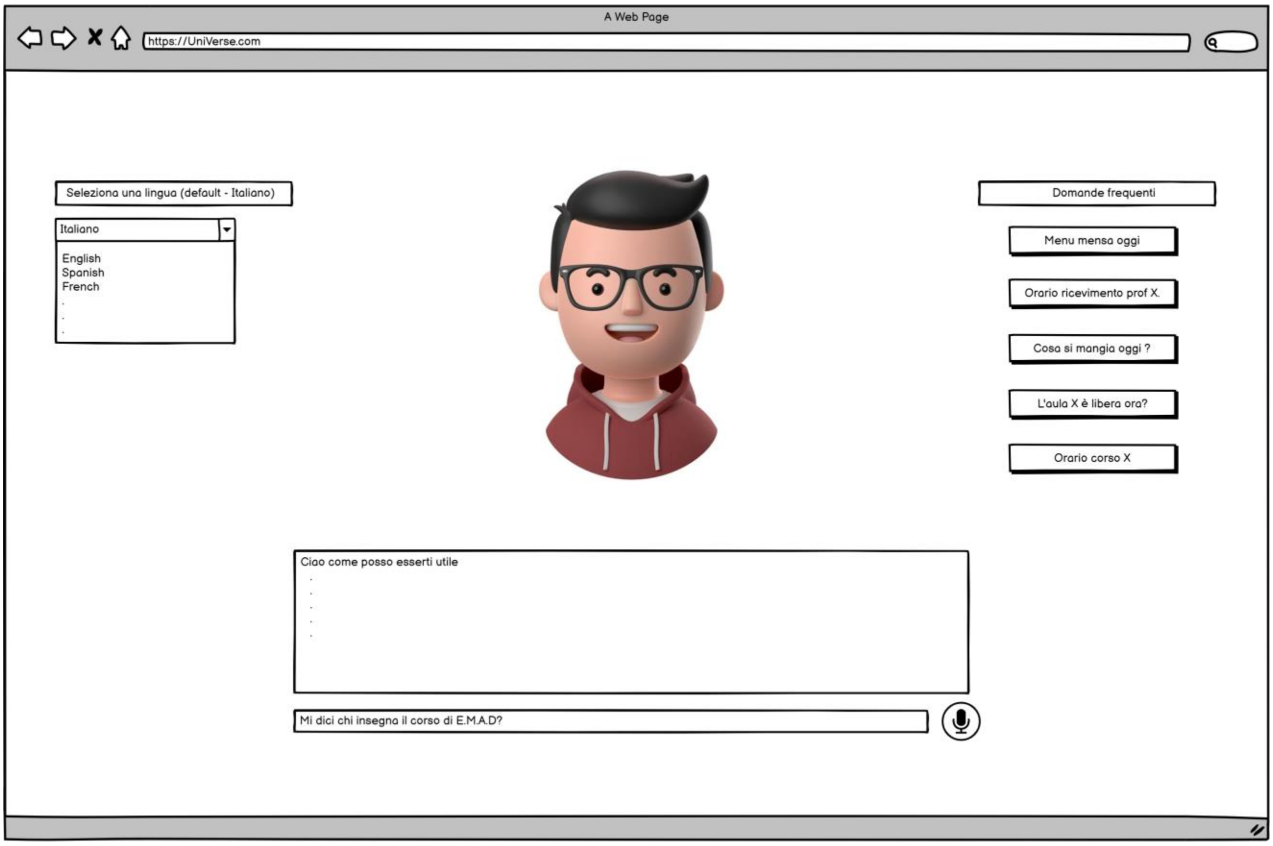


Figure 10. Navigation diagram corresponding to the action of displaying messages to the user

5.2. *MOCKS UP*

5.2.1. WEB APPLICATION INTERFACE

In the web interface, the avatar is presented in the center. Just below, there is a box that houses the conversation, displaying both user and system messages. An additional box, located below, is designated for text input by the user. To the right of the latter, a microphone icon is represented, suggesting the option of audio queries when clicked.
In the left section of the interface, there are two fields: one indicates the current language of the queries, and a drop-down menu allows to select a different language to change it.
Finally, in the right section, a set of buttons showing the most frequently asked questions is presented. This allows the user to perform queries of this type by simply clicking on the one of interest.



Ciao come posso esserti utile

Mi dici chi insegna il corso di E.M.A.D?

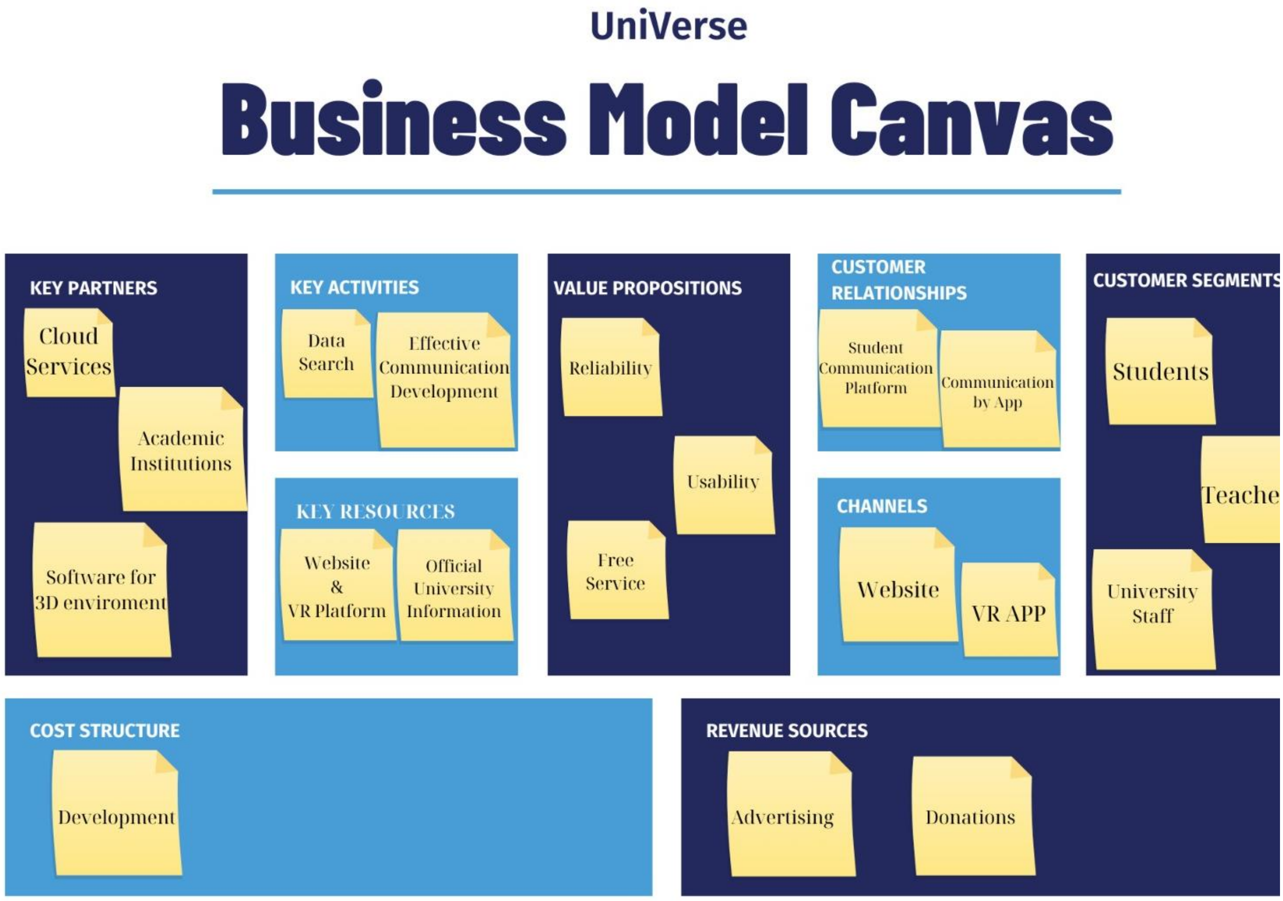
Figure 11. Web application interface

5.2.2. VR APPLICATION INTERFACE

Since the prototyping of the VR interface is not so straightforward, we will choose to describe its appearance. We are looking to reproduce an office-like environment, incorporating elements such as desks, cabinets, and computers to immerse the user in the experience. The avatar, of human appearance, will be centered in the scene and will be designed in a changing way to show diversity and dynamism (it will not always have the same characteristics).

The office will be visualized as a dynamic environment with other avatars pretending to be colleagues. Although the final layout of buttons, such as those for language switching or microphone activation/deactivation, is not yet determined, it is contemplated to keep them fixed on the map. Dialogs will be displayed statically at the bottom of the screen, allowing the reading of previous messages with a scroll, which means that only one message can be read at a time. It is important to note that these details may change as the development of the interface progresses.

BUSINESS CANVASS MODEL



UniVerse

Business Model Canvas

KEY PARTNERS
- Cloud Services
- Academic Institutions
- Software for 3D enviroment

KEY ACTIVITIES
- Data Search
- Effective Communication Development

KEY RESOURCES
- Website & VR Platform
- Official University Information

VALUE PROPOSITIONS
- Reliability
- Usability
- Free Service

CUSTOMER RELATIONSHIPS
- Student Communication Platform
- Communication by App

CHANNELS
- Website
- VR APP

CUSTOMER SEGMENTS
- Students
- Teache
- University Staff

COST STRUCTURE
- Development

REVENUE SOURCES
- Advertising
- Donations

Figure 12. Business Model