Final Phase

Group#5

#	Student Name	ID
1	Albatoul Alkahlan	443200696
2	Daniah Alkathiri	443200838
3	Ghala Alsugair	443200968
4	Omaima Rahhal	443203897
5	Suhad Aljuraysh	443200660

Instructor Name: L.Rawan Alabdulrahman

Table of contents

Table of Contents

Revision Control History

Version Number	Date	Description
1	7/02/2024	Identifying Problem Domin and gathering requirements
1	16/2/2024	Project adjustment
2	1/4/2024	Use case diagram review, component diagram review
3	12/5/2024	Diagrams review

Table 1: revision control history

Updates

Updated section	Description
Non-Functional Requirements	Usability under non-Functional Requirements is optimized to be measurable
Functional Requirements	Adding additional requirement for the user (requirement number 14)

System Glossary

Abbreviation	Description
Deaf	Deaf individuals have a significant degree of hearing loss and rely primarily on visual methods of communication, such as sign language.
Hearing	Hearing individuals can perceive sound and interpret it through the auditory system.
User	Hearing and Deaf induvial.
Avatar	A graphical representation or visual representation of a person or character in the digital realm.
Community	Community is a part of the system that enables users to post pictures accompanied by captions, allowing them to share experiences, insights, and updates with others. Additionally, users can interact with posts by liking and commenting, fostering meaningful discussions and connections.
NLP	Natural language processing (NLP) helps computers process, interpret, and analyze human language and its characteristics by using natural language data.[4]
AI	Artificial intelligence (AI) is a subset of computer science that deals with computer systems performing tasks with similar, equal, or superior intelligence to that of a Human.[4]
Computer vision	Computer vision focus on enabling computers to identify and understand objects and people in images and videos

Table 2: System Glossary

Introduction

The Monset system is a groundbreaking project that makes communication easier for people who are deaf or hard of hearing. Monset's solution revolves around its innovative approach to translating between signed and spoken languages, facilitating communication between individuals with and without hearing impairments. Through intuitive interfaces and seamless integration into everyday communication channels, Monset ensures accessibility and inclusivity for all users.

We chose to embark on this project because we were driven by a shared desire to make a meaningful impact in the lives of deaf individuals and contribute to making their daily experiences easier and more inclusive. Recognizing the challenges faced by the deaf community in communication. By focusing our efforts on developing innovative solutions to enhance communication accessibility, we aim to empower deaf individuals, foster greater understanding between different communities, and ultimately contribute to a more equitable society.

Problem Domin Analysis

Deaf individuals in Saudi Arabia face significant communication barriers when interacting with hearing individuals who are not fluent in Arabic sign language. These barriers lead to misunderstandings, limited social interactions, and challenges in expressing thoughts and emotions. Conversely, hearing individuals encounter difficulties in learning Arabic sign language and effectively communicating with the deaf. Furthermore, deaf individuals in Saudi Arabia struggle to find suitable training courses and learning materials to improve their proficiency in Arabic sign language. Monset aims to address these issues and provide a solution.

Solution Description

Monest promotes inclusion, helps bridge communication gaps, and gives people with hearing impairments more opportunities to engage in social, educational, and work settings.

Using advanced technology, Monset translates sign language into written words and vice versa in real-time, allowing deaf and hearing individuals to understand each other better hence communicate better. With features like turning text or audio into sign language graphical representation and providing communities for users to participate more.

Using technologies like Computer Vision, Monset utilizes computer vision algorithms to analyze and interpret gestures and movements in signed language, enabling accurate translation into text and graphical representations. Additionally, Natural Language Processing (NLP) is employed to process spoken and written language, facilitating translation into sign language

representations and text. Moreover, graphical rendering technologies are utilized to generate clear and comprehensible visual representations of sign language gestures, ensuring accurate communication of information to deaf users.

The System Context View

Scenario #1

Deaf individual meeting a hearing user
The scenario involves a deaf individual and a hearing user using the Monset app. The hearing user logs in using email and password, upon logging in, they are directed to the home page. From there, they click on the camera icon to activate the camera. Holding the device steady and pointing it towards the person using sign language. The app analyzes the gestures in real-time and translates them into text and voice output. The translated text is displayed on the screen with an option for voice output. The app clears the text after 2 seconds of inactivity, ensuring a clean interface. Monset enables effective communication between the deaf individual and the hearing user.

Scenario #2

Hearing person communicate with a deaf individual
When a hearing person wants to communicate with a deaf individual using the Monset app, they log in using their email and password. Upon logging in, they are directed to the home page. From there, they click on the avatar icon, which opens a page featuring an interactive avatar. On the avatar page, the hearing user is presented with options to communicate via text or voice recording. They can type a message in the text box or choose to record their voice message. The system then utilizes the avatar to translate the message into sign language. The avatar visually represents the sign language interpretation of the input, allowing the hearing user to effectively communicate with the deaf individual.

Functional Properties

- **System requirements:**
  1. The system shall recognize sign language gestures and movements in real-time from the video input stream.
  2. The system shall translate provided text into sign language gestures using avatar.
  3. The system shall translate provided audio into sign language gestures using avatar.

4. The system shall translate provided sign language gestures into audio.
5. The system shall translate provided sign language gestures into text.

- **User requirements:**

1. The user shall be able to create an account by providing his/her first name, last name, phone number, email address, password, age and type (hearing user or deaf user).
2. The user shall be able to log in to the system using his/her email address and password.
3. The user shall be able to view his/her account information.
4. The user shall be able to edit his/her account information.
5. The user shall be able to delete his/her account information.
6. The user shall be able to join the community.
7. The user shall be able to post pictures of his/her daily activities to the community.
8. The user shall be able to add caption to his/her posts in the community.
9. The user shall be able to like posts available in the community.
10. The user shall be able to comment on posts available in the community.
11. The user shall be able to provide video input containing sign language gestures and movements through camera source.
12. The user shall be able to provide input text messages for translation into sign language gestures.
13. The user shall be able to provide input audio messages for translation into sign language gestures.
14. The user shall be able to select a photo from the Photo Gallery.

- **Admin requirements:**

1. The admin shall be able to add course advertisements.
2. The admin shall be able to edit course advertisements.
3. The admin shall be able to delete course advertisements.
4. The admin shall be able to add volunteering advertisements.
5. The admin shall be able to edit volunteering advertisements.
6. The admin shall be able to delete volunteering advertisements.
7. The admin shall be able to create community for users
8. The admin shall be able to remove users from the community.

Non-Functional Properties

- **Performance:**

1. The system shall have a latency of less than 500 milliseconds for translating sign language gestures into text or graphical representations.
2. The system shall handle at least 100 concurrent translation requests per second.

- **Accuracy and Reliability:**

1. The translation algorithms shall be able to achieve a recognition accuracy of at least 95% for sign language gestures.
2. The system shall maintain an uptime of 99.9% over a one-month period, with no more than 30 minutes of downtime allowed for maintenance or unexpected issues.

- **Usability:**

1. The user shall be able to use the system within 1 days of training.

Challenges

The challenge we may face when developing the "Monset" app is how we can design an effective recommendation algorithm that seamlessly and accurately translates sign language into speech or text, and vice versa. We may also encounter difficulties in dealing with advanced video and image processing technology to capture and interpret the subtle nuances of sign language gestures with precision. Additionally, dealing with variations in sign language dialects and regional differences adds another layer of difficulty to the development process.

Projection

At the end of this semester, we aspire to develop a system that meets the needs of the deaf community in our society with high quality, by selecting the best architectural design for this system.

Our final product will be a mobile application created and worked on for 7 weeks (about 1 and a half months) by a team of 5 members collaborating on assigned tasks to achieve the desired expectations and gain skills and experience.

The Functions and tasks of the system that are expected to be accomplished:

- Utilizing artificial intelligence effectively to serve individuals with hearing disabilities.
- Creating a vibrant community for the deaf where they exchange their interests.
- Making communication between the deaf community and hearing individuals easy and fast.
- Assisting the deaf in engaging with the community through participation in various activities.

By the end of the semester, we aim to develop key skills and knowledge in software architecture:

- **Improved Teamwork and Communication:** Working better together as a team, communicating ideas effectively, and problem-solving collaboratively.
- **Applying Software Architecture Basics:** Using fundamental principles to solve real-world problems in software design and development.
- **Breaking Down Complex Systems:** Learning how to break large, complicated systems into smaller, manageable parts for easier understanding and development.
- **Choosing the Right Technology:** Knowing how to pick the best tools and methods based on what the project requires.
- **Documenting Software Architecture:** Getting better at writing down and explaining how software is built using the right tools and methods.
- **Using Best Design Practices:** Learning and applying industry-standard techniques to create well-designed software architectures.

By achieving these goals, we hope to become proficient software architects ready to tackle complex projects in the future.

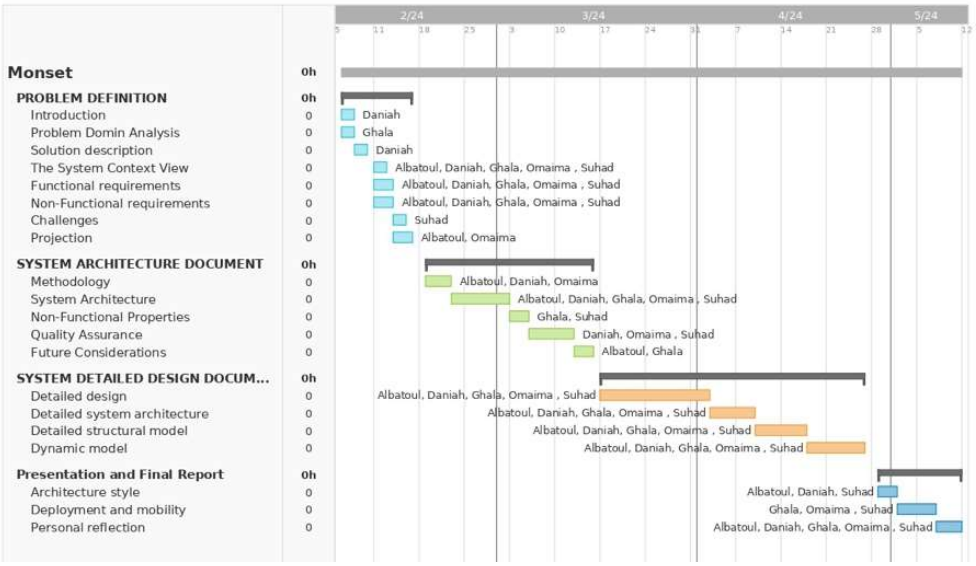


Figure 1: Gantt Cha

Methodology:

After thoroughly analyzing our system requirements, goals, and user demands, we have determined that Agile Development is the ideal methodology for our Monster app. By embracing Agile Development, we can achieve successful and efficient project execution while placing a strong emphasis on continuous improvement and collaboration.

When it comes to software development, Agile methodology provides us with valuable benefits such as flexibility, customer satisfaction, and enhanced collaboration. These advantages empower the Monset team to deliver incremental value with each sprint. The iterative nature of Agile allows for swift feedback loops, enabling us to adapt and make necessary refinements throughout the development journey.

Iteration number	Module/subsystem	duration
1	Registration, Community, Sign language video translator	30 days
2	Text/audio translator, Avatar	17 days

Table 3: Monset iteration plan

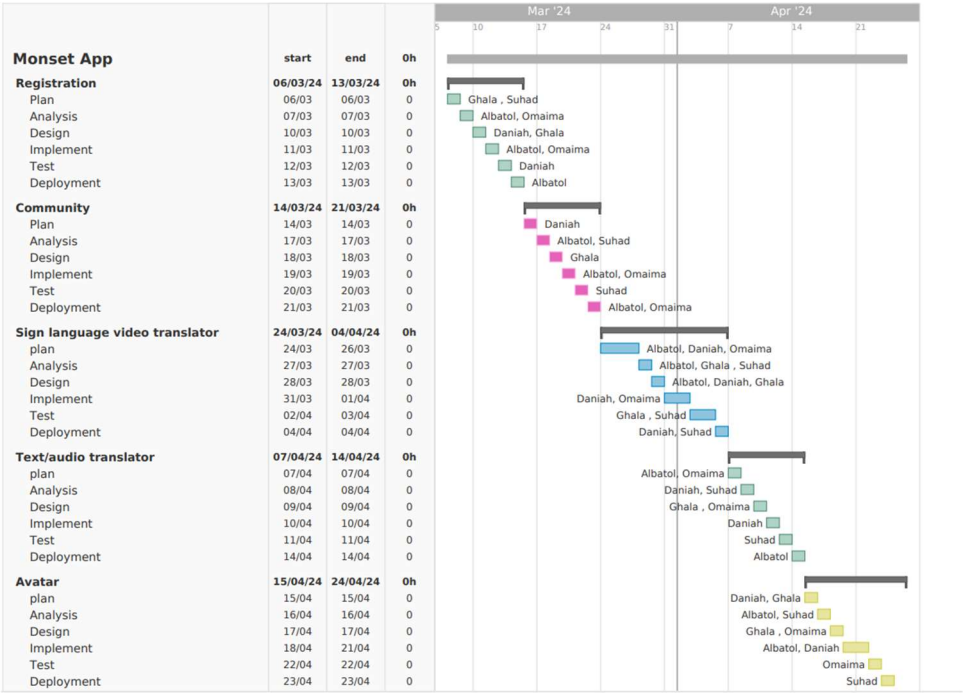


Figure 2: Gant chart applying Agile methodology plan for Monset app.

Ensuring precise accuracy in avatar sign language gestures and video detection is a critical aspect of our Monset system. We understand the risks involved in capturing

and interpreting these gestures accurately. Therefore, we prioritize this aspect and allocate significant resources to ensure utmost precision.

Our system integrates advanced technologies for accurately detecting and interpreting sign language from video input. This capability is vital for effective communication and meaningful translation. Despite the challenges, we are committed to continuously improving our system to provide the best accuracy and reliability in gesture interpretation.

We strive to enhance the user experience and meet the highest standards of performance and quality. By investing in research and development, we aim to mitigate risks and deliver a Monset system that facilitates inclusive communication for individuals using sign language.

System Architecture:

Design decisions:

How Monset will translate sign language into text or audio?

Monset utilizes Sign Language Recognition **(SLR)** technology to translate sign language into text or audio. By analyzing live video feeds of sign language gestures, with a focus on hand movements and body postures, Monset's **SLR** model can recognize and classify these gestures. It then maps them to corresponding words or phrases in Arabic. The system offers two translation options: written text and synthesized audio. For text translation, Monset employs Natural Language Processing (NLP) techniques to generate written representations. For audio translation, it uses Text-to-Speech (TTS) systems to convert the recognized sign language gestures into synthesized spoken language in Arabic.[8]

How Monset will translate text or audio input into sign language gestures?

Monset presents the translated sign language gestures using the **PlayerMe** avatar API. The animated avatar visually communicates the corresponding sign language gestures, allowing users to understand the input in a visual and accessible manner.[6]

How Monset will store user's data securely?

To ensure secure user data in the database, we'll utilize RSA encryption, which involves generating an RSA key pair, encrypting data with the public key, and securely storing the private key. Decryption requires the private key for accessing the original information.

How will Monset ensure clear and easy-to-understand output?

To ensure clear and easy-to-understand output in Monset, we implement several strategies. We use high-quality sign language representations, such as Avatars, to accurately convey translations. This emphasizes clarity and distinctiveness in gestures, ensuring that users can easily interpret the meaning of the signs. Additionally, we conduct usability testing with representative users, including those who are deaf or hard of hearing. This allows us to gather feedback and continuously refine the output, enhancing overall clarity and effectiveness in communication. By incorporating these strategies, we strive to make Monset's output as clear and understandable as possible for all users.

How will Monset's ensure the correctness of the output in sign language translation?

Monset will provide multiple mechanisms to ensure the correctness: Train and validate: We will use a large dataset of sign language gestures to train Monset's and evaluate its performance using a separate set of data. Regular updates and maintenance: We will regularly update and maintain the system, incorporating advancements in sign language recognition and language processing technologies to enhance the accuracy of the output.

Domain model:

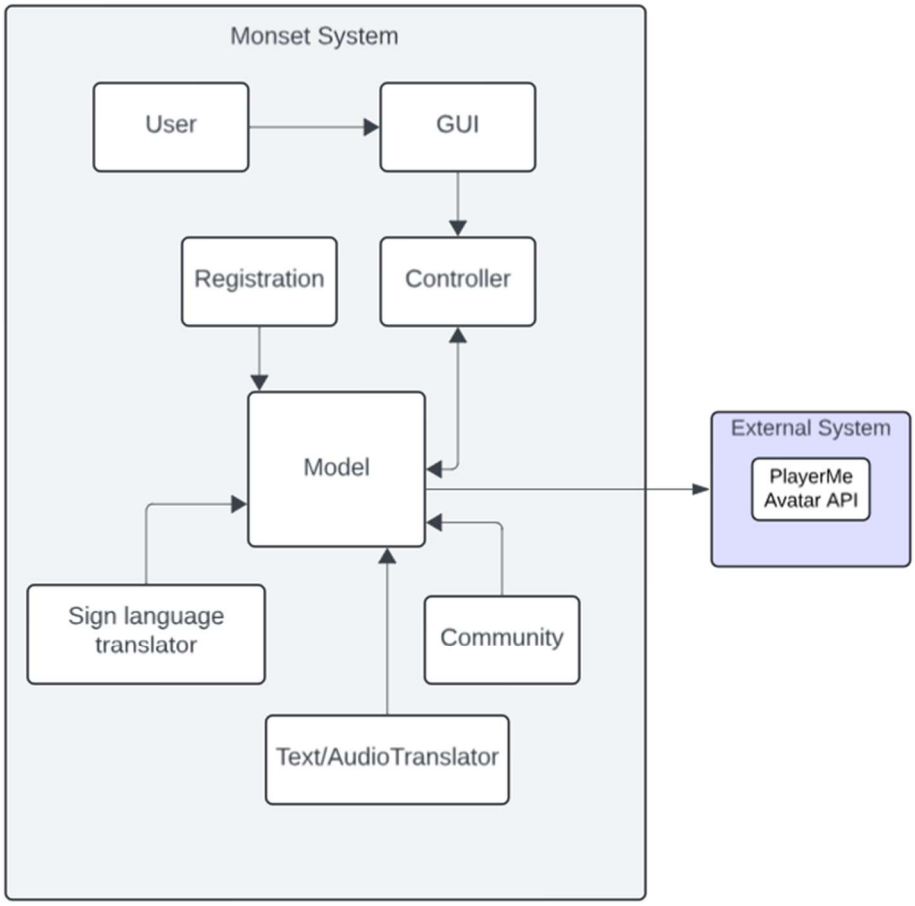


Figure 3: Domin Model diagram of Monset app [5]

Architectural Style:

Regarding the architectural approach, we've opted to implement the Model-View-Controller (MVC) style for the Monset system. MVC divides the application into three primary components: the model, the view, and the controller.

The model handles both data management and business logic. It's responsible for processing user input and performing database operations like data retrieval and addition. The view is tasked with presenting information to users in an appropriate

format. Finally, the controller serves as a bridge between the views and the model, facilitating communication and acting as an intermediary.

MVC is widely used architecture style in mobile development. A framework for creating extensible and scalable projects. The reason for choosing this model is that it offers several key advantages which we find important for Monset system. Firstly, it ensures a clear Separation of Concerns, dividing Monset system into distinct components - Model, View, and Controller - simplifying code organization and maintenance. Secondly, it enables scalability, allowing for the extension and modification of individual components to accommodate changing requirements without disrupting the entire system. Lastly, MVC enhances maintainability by encouraging a clear separation of concerns, leading to improved code quality and easier troubleshooting. Additionally, it provides flexibility, enabling the addition of new features or functionality without extensive refactoring.

Expanding the architectural approach to include the 3-tier distributed architecture alongside the MVC style for Monset introduces a distributed computing model that separates the application into three layers: presentation, application processing, and data management.

In this architecture, the Model and Controller components reside on the server-side, responsible for handling data management, business logic, and application processing. The View component is situated on the client-side, responsible for presenting information to users in a user-friendly format. Finally, the database is in the third tier, responsible for storing and managing data.

The integration of the 3-tier distributed architecture complements the MVC style by extending its benefits to a distributed environment. It ensures that the system can efficiently handle many users and scale horizontally as needed while maintaining a clear separation of concerns, facilitating code organization, maintenance, and flexibility for future enhancements.

This hybrid approach offers Monset system the advantages of both MVC and distributed architecture. It ensures that the system remains modular, scalable, and maintainable while efficiently utilizing resources and providing a seamless user experience across different client platforms. Additionally, it enhances the system's resilience to failures and allows for easier deployment and management in a distributed environment.

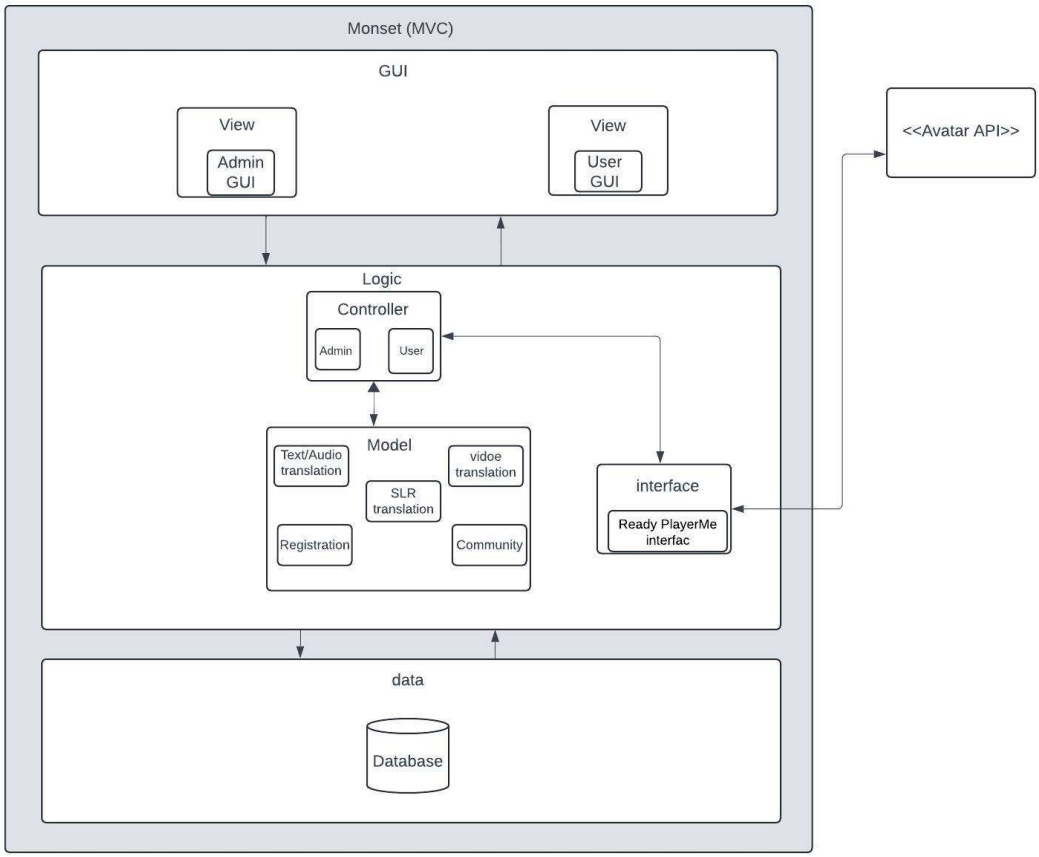


Figure 4: high level diagram of Monset app [5]

We considered several architecture styles but ultimately decided to avoid two specific ones. Firstly, we rejected the Data Center Architecture due to our requirement for real-time processing of gestures and immediate translation to text or audio. Utilizing a data center architecture could introduce latency as it would involve sending gesture data to servers for processing, which could hinder the responsiveness of our system.
Secondly, we opted against the Batch Sequential Architecture. While this architecture simplifies the division of the application into subsystems, it proved unsuitable for our needs. It lacks the capability to provide real-time responses necessary for features such as instant voice/text translation to gestures for deaf users. Additionally, it lacks the interactive interface essential for our application's usability and effectiveness.

Structural mode

Class diagram

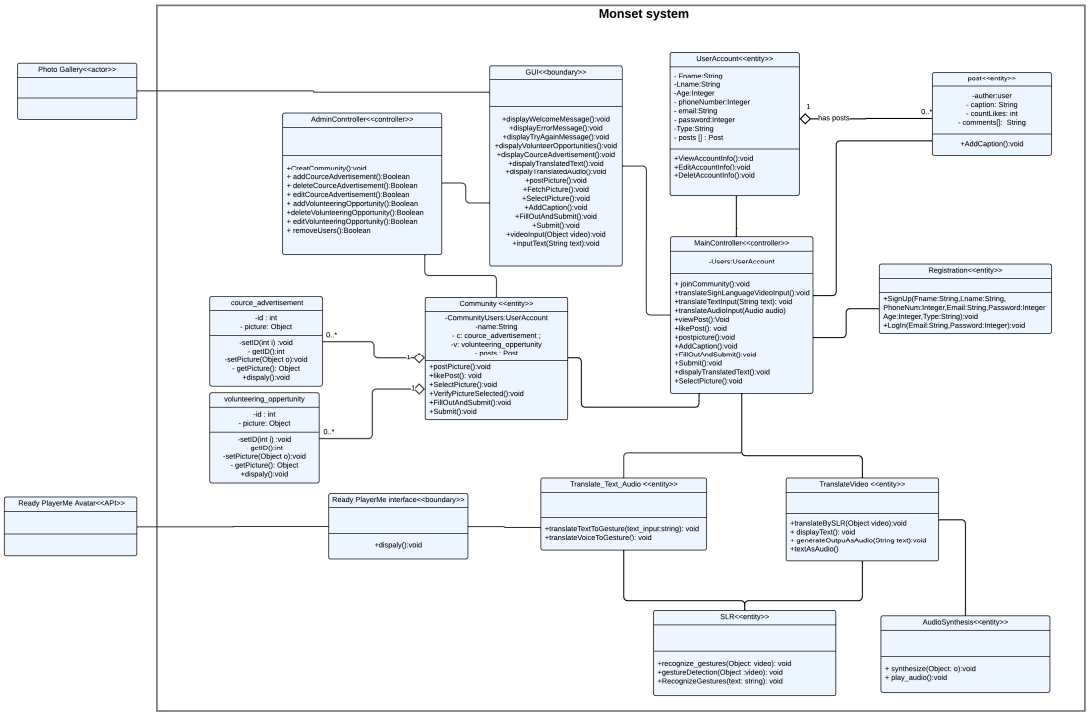


Figure 5: Class diagram of Monset app

Component diagram

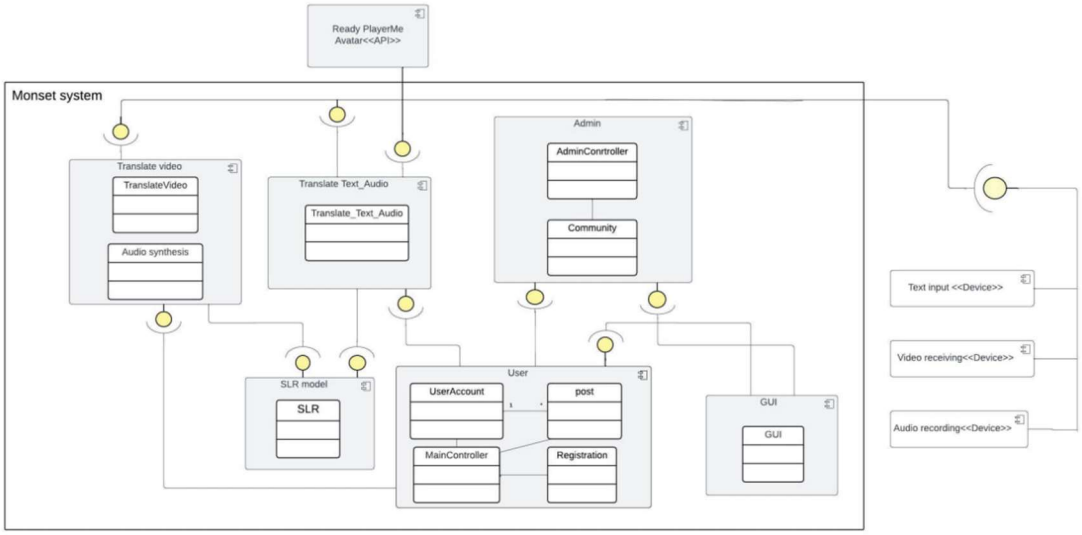


Figure 6: Component diagram of Monset app

Component's descriptions: -

Translate video: Component used to translate provided video stream, to a text representation, or optionally, to audio representation. It accepts a video stream as input, processes it, then utilizes the SLR class to detect and recognize sign language gestures. The SLR returns the result as a text representation. Depending on the user's preference, the output can either be the translated text or an audio representation. If the user requests audio output, the Audio Synthesis class is used to generate the corresponding audio.

Translate Text_Audio: Component used to translate provided text or audio, to a graphical representation of sign language gestures. It accepts either text or audio input and translates it. It utilizes the SLR model class to generate sign language gestures. The output, consisting of the graphical representation of gestures, is then displayed using an external system Avatar.

The SLR (Sign Language Recognition): SLR is a pre-trained model developed by researchers at the University of Surrey for sign language recognition. This model component integrated with our system. It analyzes visual information representing sign language gestures and translates them into text representations.

User: Component that provides main functionalities to the users, account and registration. Allow users to sign up, login, manage account information, and participate in the community.

Admin: Component that provides management functionalities to the admin. Manage community and users participating. It will provide services for the admin to add course advertisements and volunteering Opportunities. As well as removing users from the community.

Ready PlayerMe Avatar: Ready Player Me Avatar provides an API for creating customizable 3D avatars. Monset will utilize this API to generate avatars representing sign language gestures when translating text or audio input into graphical representations.

State machine diagram:

User state machine diagram

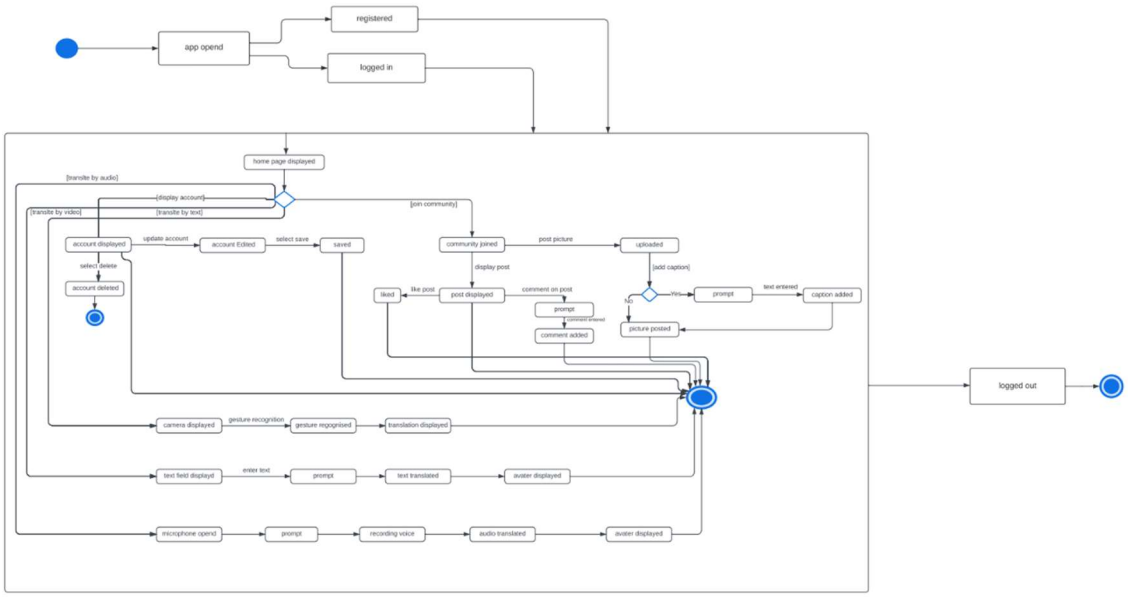


Figure 41: State machine diagram of Monset System

Admin state machine diagram

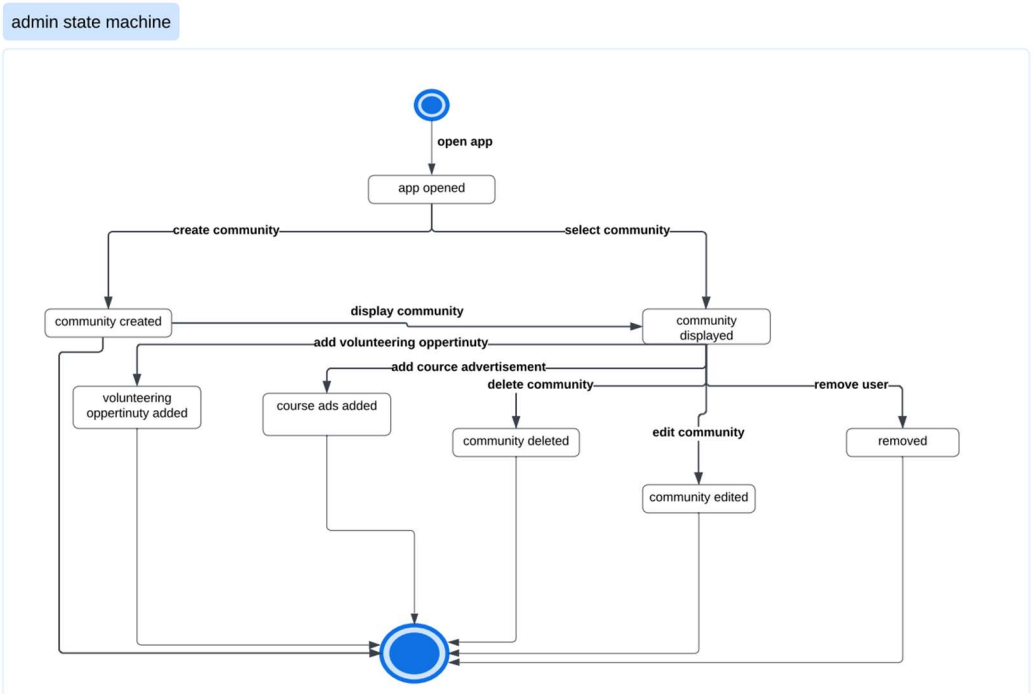


Figure 42: state machine diagram of Monset System

Deployment diagram:

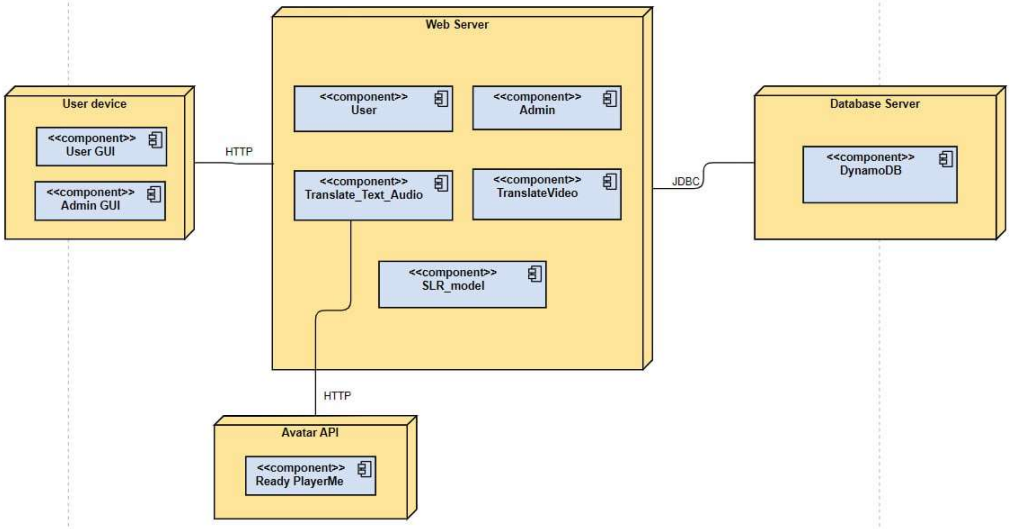


Figure 43: Deployment diagram of Monset System

Behavioral model:

Use Case diagram:

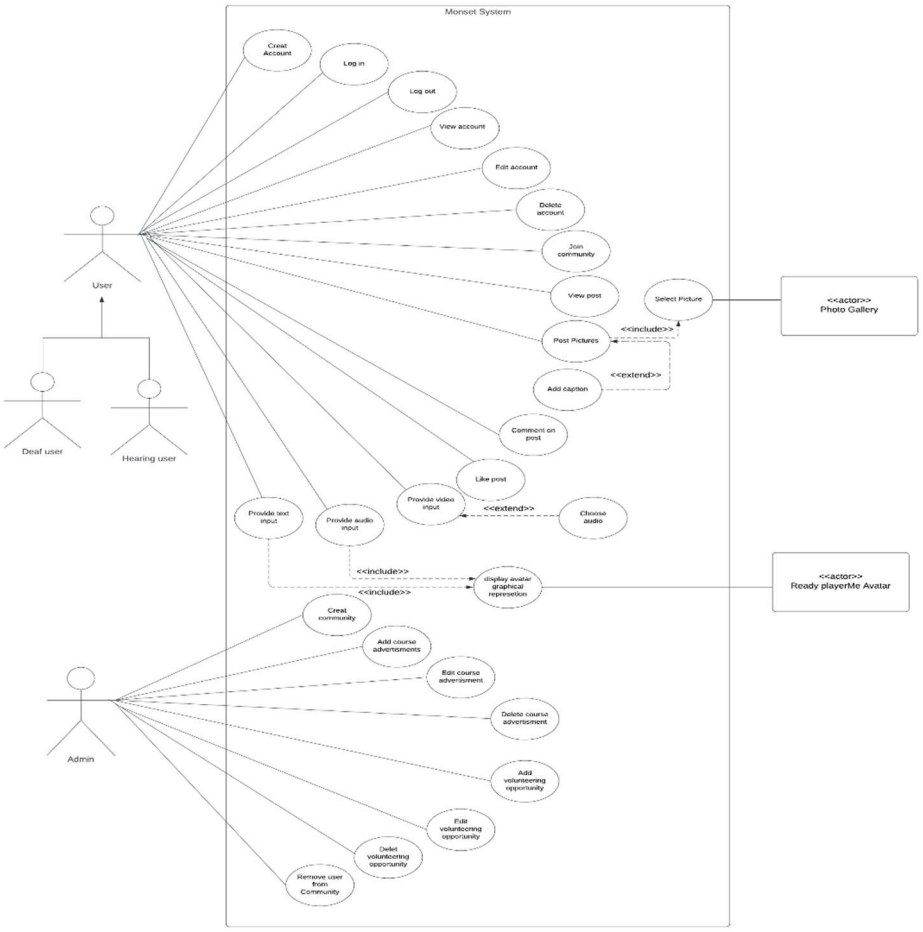


Figure 7: Use Case diagram of Monset app

Use Case Descriptions

- Post picture use case description.

Use Case Description	
System: Monset System	
Use Case name: Post picture	
Primary actor: User	**Secondary actor(s):** -
Description: This use case allows user to post a picture in the community	
Relationships: **Includes:** Fetch pictures **Extends:** Add caption	
Pre-conditions: The user must be logged in. The user must join the community.	

Basic flow:	
Primary Actor (user)	**System**
1.The user selects 'Post picture' option. 4. The user selects a specific picture. 6.The user selects 'Submit' option	2.'Fetch pictures 'use case is performed. 3.System display pictures 5.System, verify that the picture is selected. 7.System displays successful message.

Alternative and exceptional flows:
1.if the user selects 'Add caption' option on a certain picture before step 6:
1.a 'Add caption' use case is performed.
1.b step 6 is resumed.
2.No picture selected:
If in step 4 user did not selects any picture
2.a the system displays a message that no picture is selected.
2.b step 4 is resumed.

Post-conditions:

Successful condition: The picture is posted successfully.

Failure condition: No picture is posted in the community.

- Provide video input use case description.

Use Case Description	
System: Monset System	
Use Case name: Provide video input	
Primary actor: User	**Secondary actor(s):** -
Description: This use case allows the user to provide video as input to the camera source, the input video contains sign language gestures that will be translated into text.	
Relationships: **Includes: -** **Extends: choose audio**	
Pre-conditions: The user must be logged in.	

Basic flow:	
Primary Actor (user)	**System**
1.The user selects 'video input' option. 3.The user directs the camera to the deaf individual who uses sign language to communicate.	2.System displays video frame. 4.System starts processing the video in real time. 5.System translates the gestures to text. 6.System displays translation gestures on screen.
Alternative and exceptional flows: **1.After step 6 if the user selects 'audio' option after translation done:** 1.a 'choose audio' use case is performed. 1.b The system generates the audio for the user. **2. incorrect recognition:** In step 3 If the user direct camera for non-human object: 2.a The system displays an error message indicating that the recognition process doesn't perform. 2.b step 3 is resumed.	
Post-conditions: **Successful condition:** The recognition process is performed, and gestures are translated to text to display it for user successfully. **Failure condition:** No recognition for gestures is done.	

- Provide text input use case description.

Use Case Description	
System: Monset System	
Use Case name: Provide text input	
Primary actor: User	**Secondary actor(s):** -
Description: This use case allows the user to provide text as input to the Monset app, the text input will be translated to sign language visually displayed.	
Relationships: **Includes:** Display avatar **Extends: none**	
Pre-conditions: The user must be logged in.	

Basic flow:	
Primary Actor (user)	**System**
1. The user selects 'enter text'-option. 3. The user enters the text to be translated.	2. The system displays a text field. 4. The system translates the text to sign language gestures. 5. The system invokes 'Display avatar' use case 6. 'Display avatar' use case performed.

Alternative and exceptional flows:

1. Invalid Text Entry:

If in step 3 the user enters Invalid Text

1.a the system displays a message indicating that text is unacceptable.

1.b step 3 is resumed.

2. Avatar Display Failure:

If in step 5 the system fails to invoke the 'Display avatar' use case after translating the text

2.a The system displays an error message indicating that the avatar display could not be performed.

2.b exit.

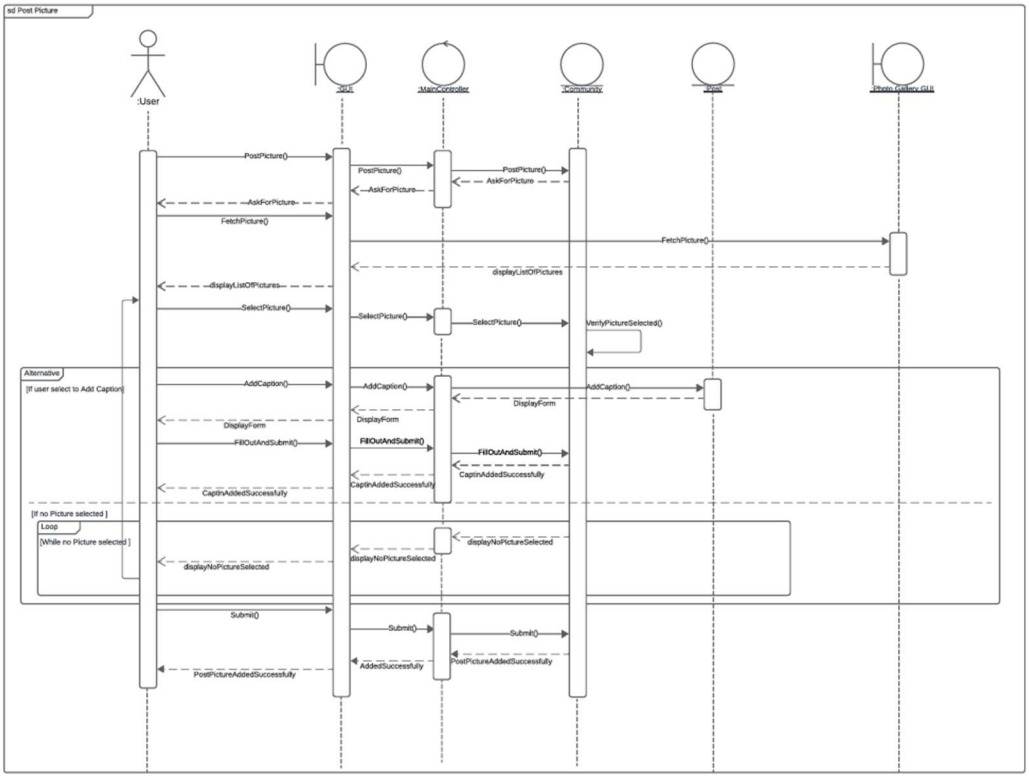
Post-conditions:

Successful condition: The translated text, represented as sign language gestures, is displayed to the user Successfully.

Failure condition: No avatar is displayed.

Sequence diagrams.

Post picture sequences diagram



.

Figure 8: Sequence diagram of Post picture use case in Monset app.

Provide video input sequence diagram

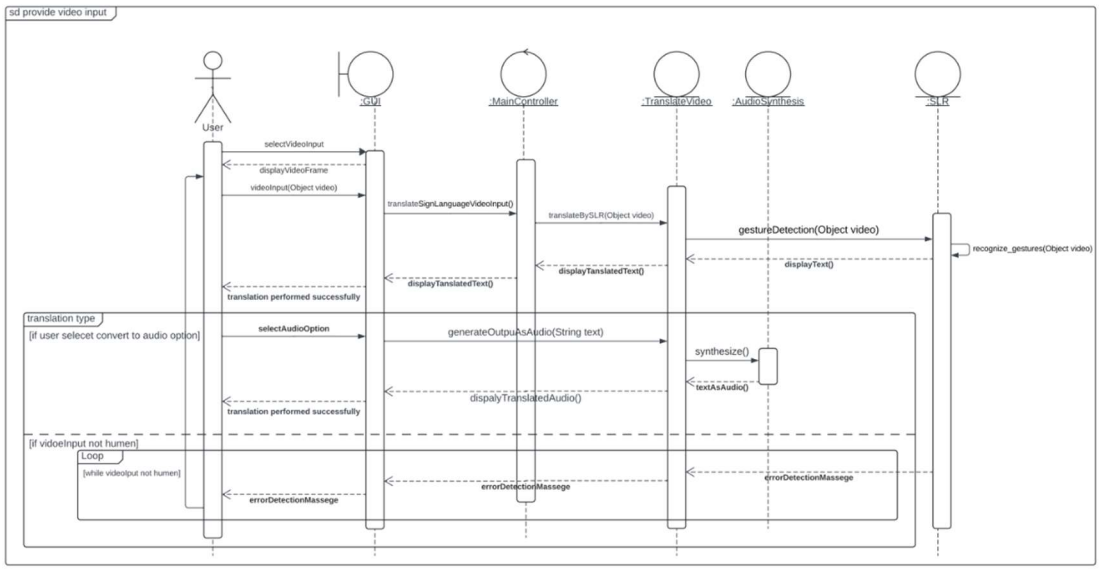


Figure 9: Sequence diagram of provide video input use case in Monset app

Provide text input sequence diagram

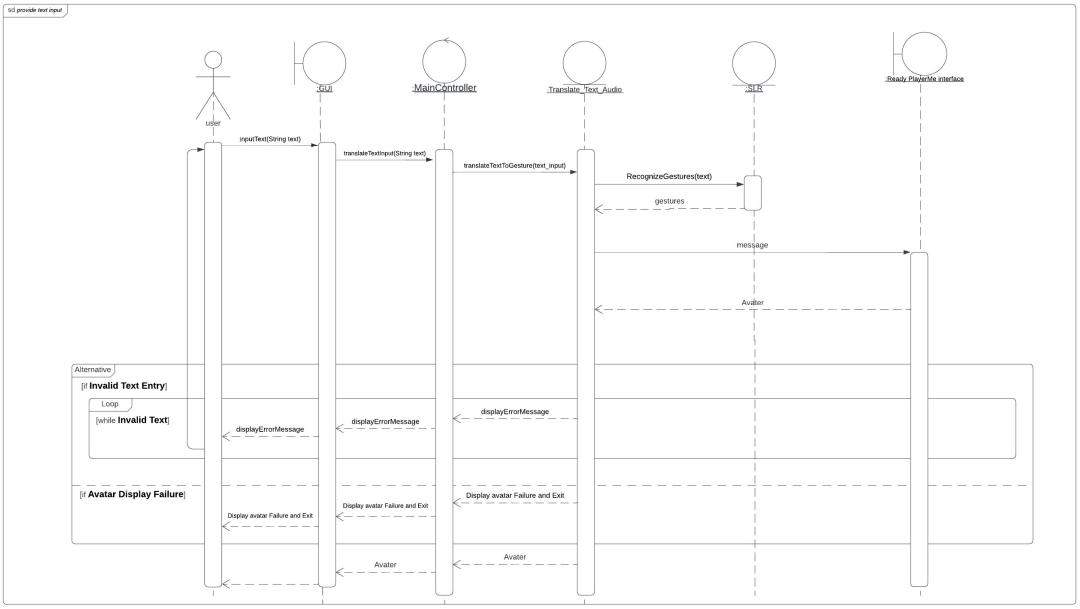


Figure 10: Sequence diagram of provide text input use case in Monset app

User interface:

USER INTERFACE

Figure 11: Welcome page.

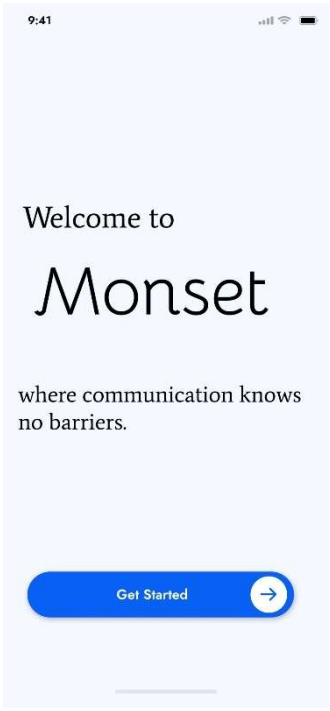


Figure 12: Welcome page.

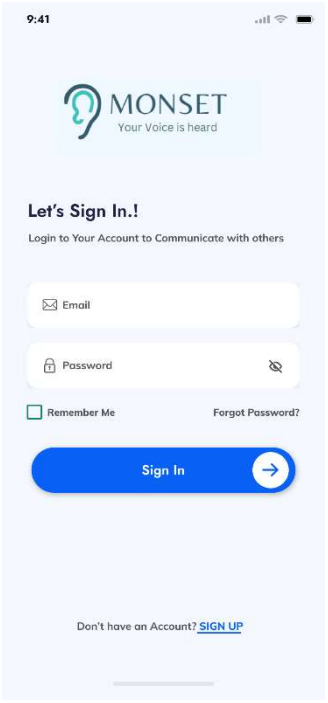


Figure 13: Sign In page.

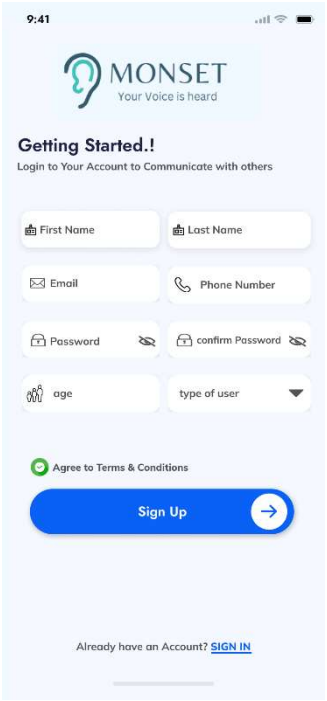


Figure 14: Sign Up page of.

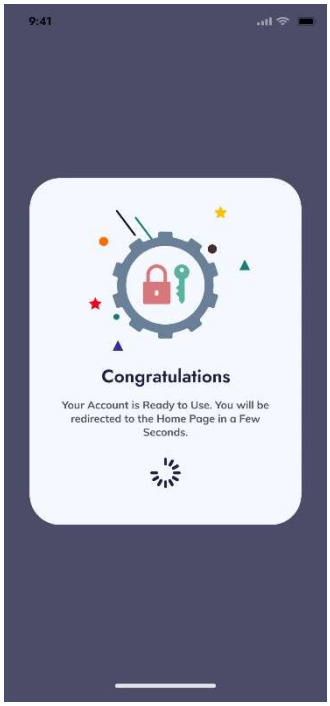


Figure 15: Congratulations message for creating account.

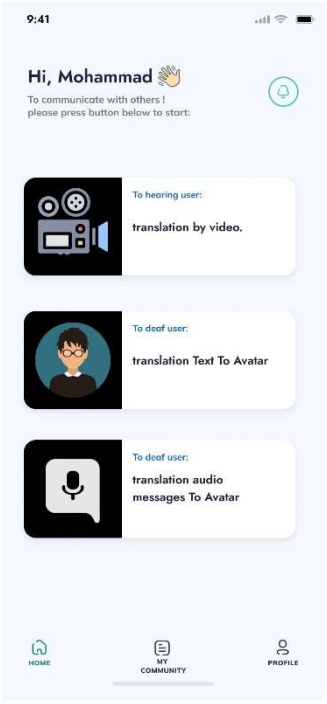


Figure 16: Home page of User.

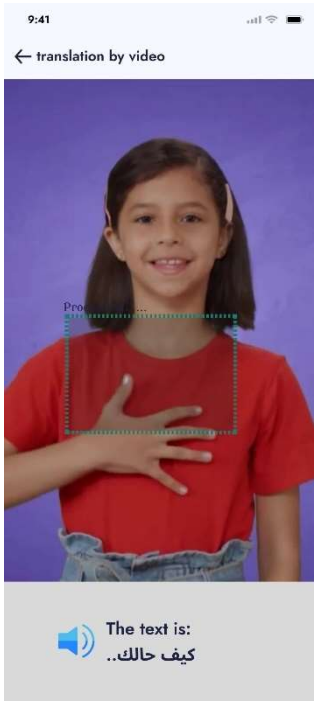


Figure 17: translation by video option that processing gestures of human and translates them into text.

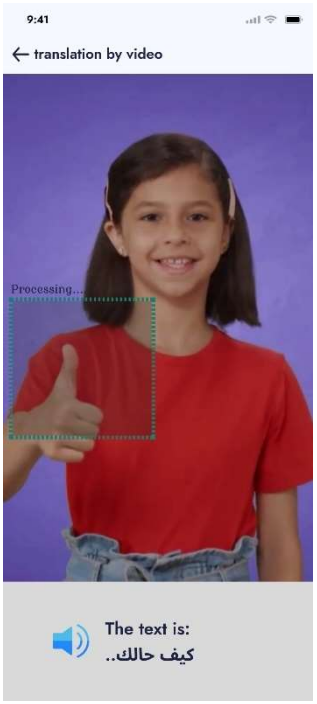


Figure 18: translation by video option that processing gestures of human and translates them into text.

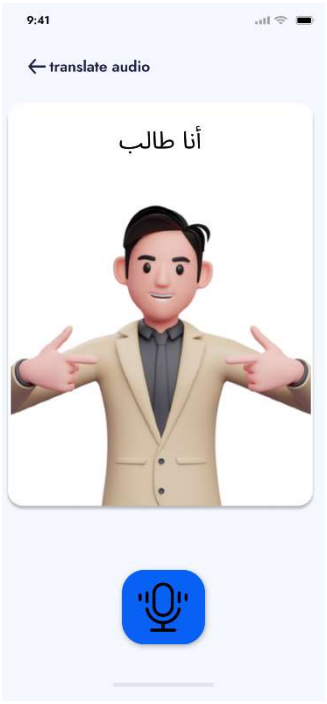


Figure 19: translation by audio option that processing audio of humans and translates them by the avatar.

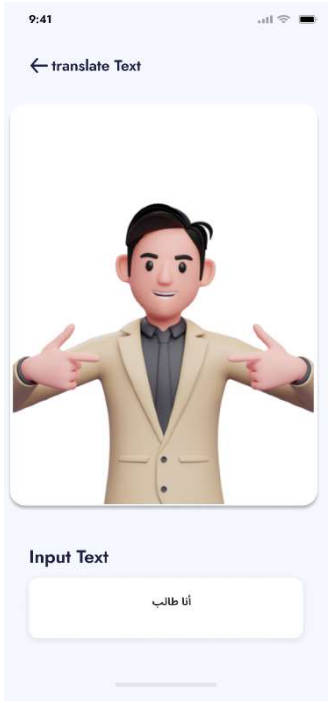


Figure 20: translation by text option that processing text input and translates them by the avatar.

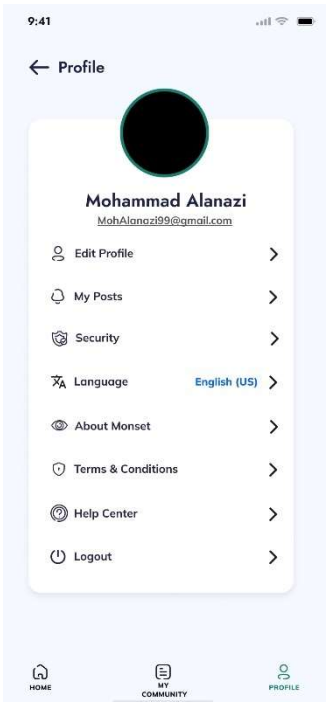


Figure 21: view profile page.

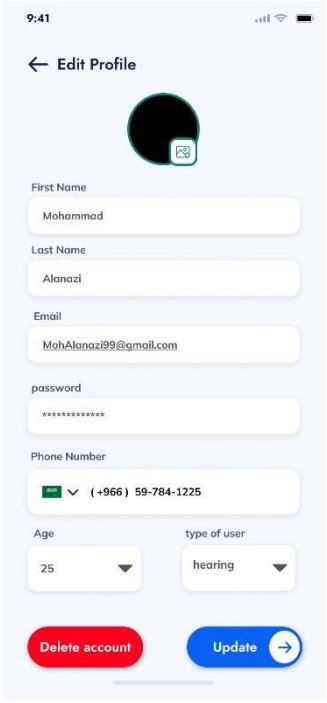


Figure 22: Edit profile page.

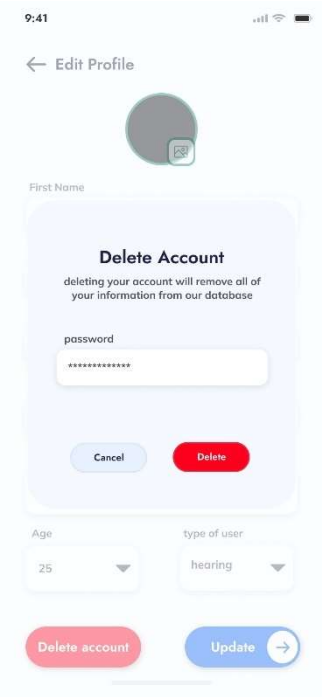


Figure 23: delete profile page.

Figure 24: my community page to show available communities.

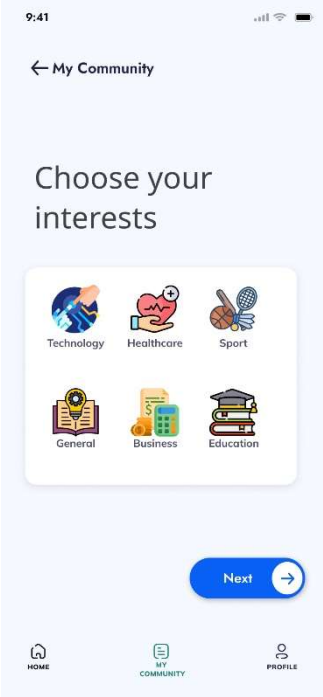


Figure 25: my community page to show available communities.

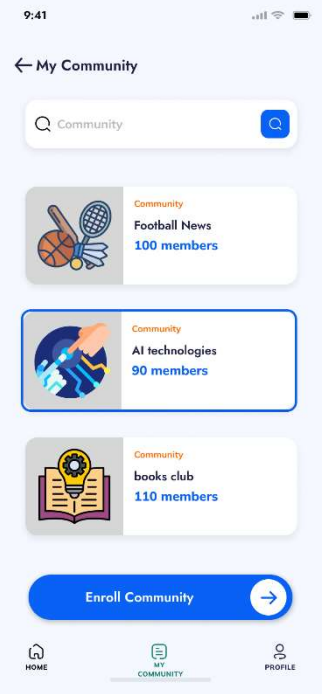


Figure 26: available communities.

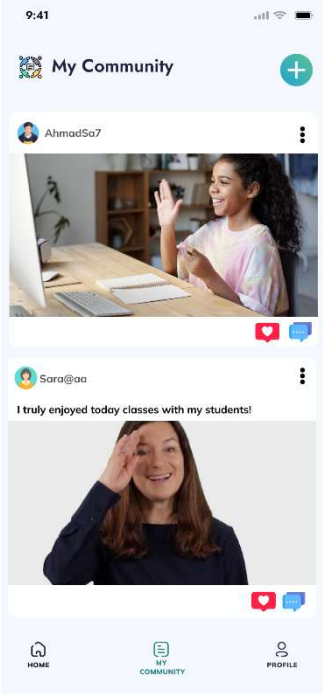


Figure 27: my community page to view posts.

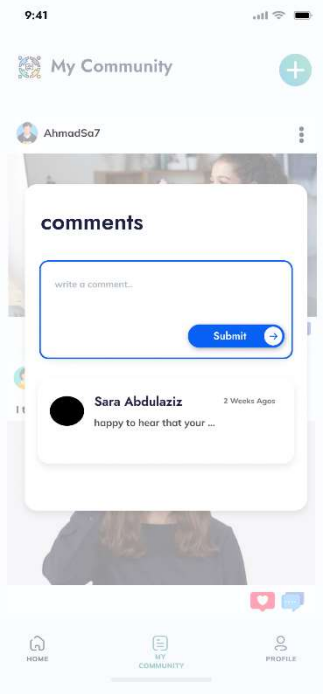


Figure 28: my community page to view posts and write comments.

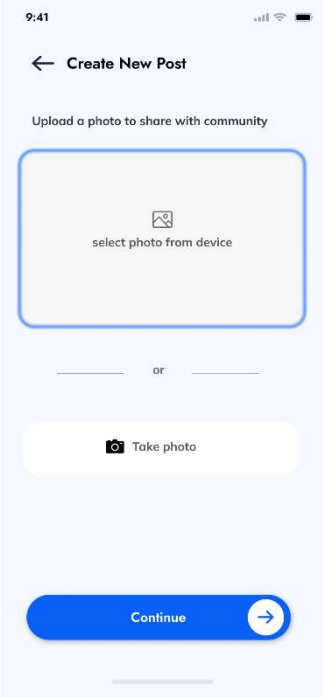


Figure 29: post picture option in community to add picture.

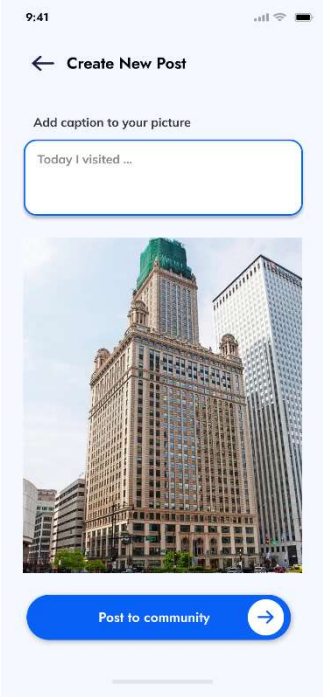


Figure 30: add caption on post picture.

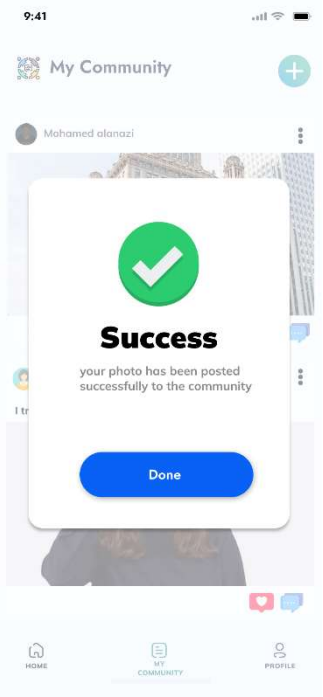


Figure 31: successful message of post picture in community.

ADMIN INTERFACE

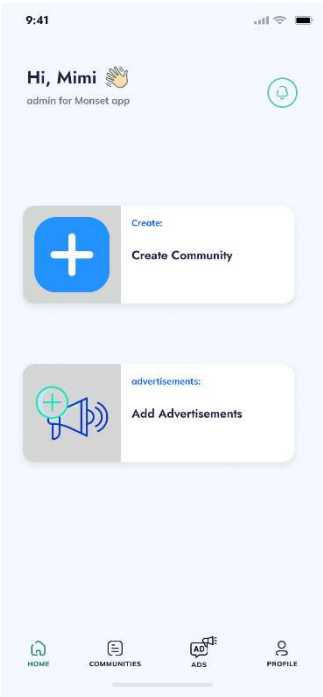


Figure 32: home page for admin.

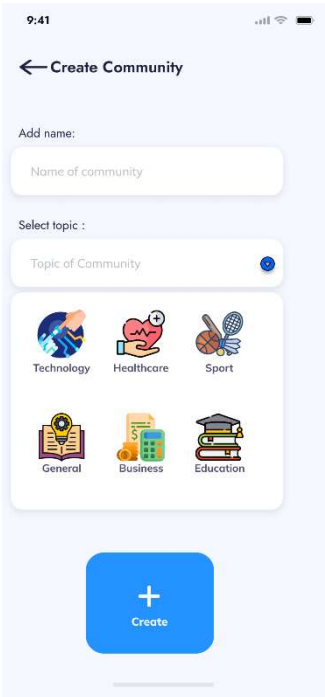


Figure 33: create community page for users.

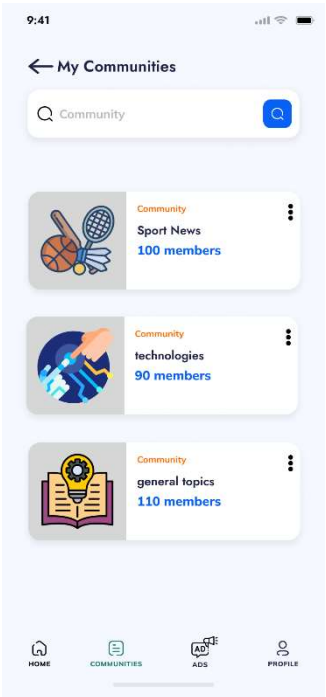


Figure 34: community page that created for users.

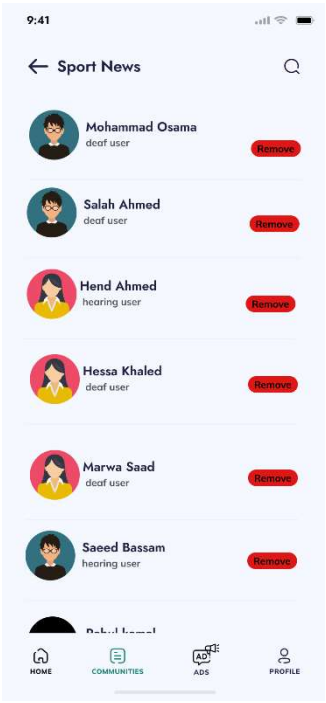


Figure 35: view users that join to community and ability to remove them.

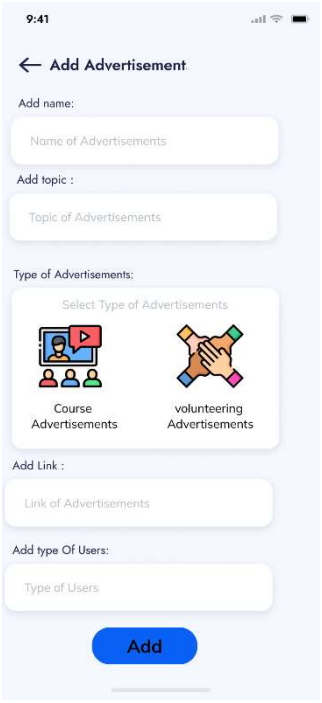


Figure 36: add advertisements page.

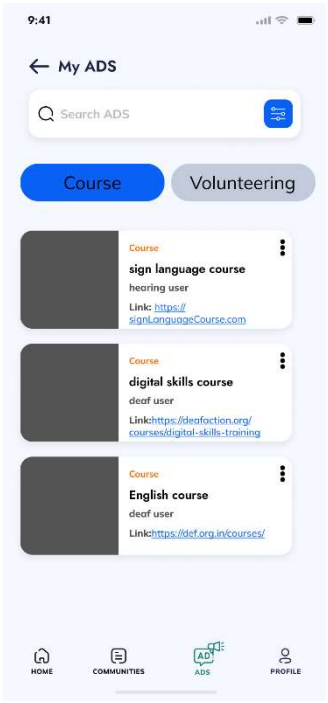


Figure 37: view advertisements page.

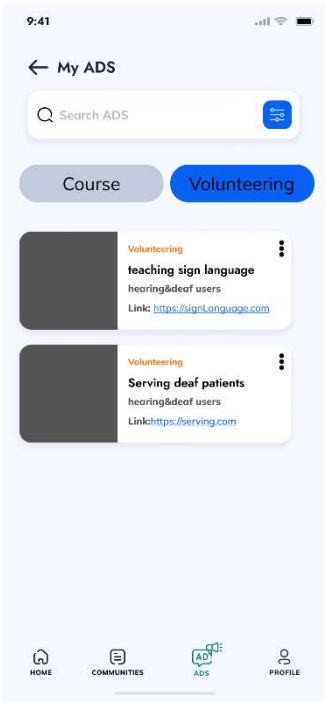


Figure 38: view advertisements page.

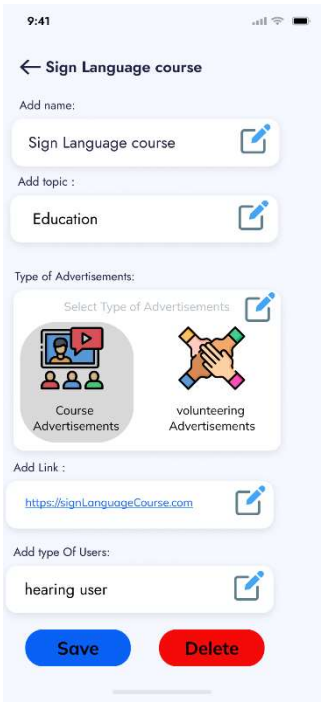


Figure 39: edit course advertisements page.

Figure 40: edit volunteering advertisements page.

Mapping class:

Class name	Component name
MainController	User
Translate_Text_Audio	Translate Text_Audio
TranslateVideo	Translate video
SLR	SLR model
AudioSynthesis	Translate video
Ready PlayerMe interface	Ready PlayerMe Avatar
Registration	User
Post	User
UserAccount	User
GUI	GUI

AdminConrtroller	**Admin**
Ready PlayerMe Avatar	**Ready PlayerMe Avatar**
Community	**Admin**
volunteering_oppertunity	**User**
cource_advertisement	**User**

Non-Functional Properties:

1. Reliability:
   It is significant to the system since
   We expect some users to rely on Monset for their work. Monset system maintains an uptime of 99.9% over a two-month period, with no more than 5 minutes failure time.

2. Usability:
   Monset system should have a friendly interface that makes it easy for users, especially those unfamiliar with technology, to quickly access and navigate. Users shall be able to start using the system confidently within just one day of training.

3. Correctness:
   Correctness is crucial in Monset system, which translates between sign language and text/audio formats. Achieving up to 98% accuracy in translating gestures to Arabic text or voice ensures the system's effectiveness. Any errors could compromise its core functionality.

4. Maintenance:
   The documents in the Monset system must achieve a clarity and precision rating of at least 95%, facilitating developers' quick and comprehensive understanding of the entire system.

Constraints

1. Monset can face constraints in terms of time. If the system is delayed more than 0.5 seconds in translating video input into text or audio, the meaning of the sign language syntax may be altered. Therefore, it is crucial for the translation to be timely and accurate to ensure the intended meaning is conveyed correctly.
2. Munset system has a hardware constraint that requires the user to have a device with a camera, as the system relies on capturing video input for translation.

Without a camera, the system cannot function effectively and the translation would not be meaningful. Therefore, the availability of a camera-equipped device is necessary for the Munset system to be used properly.

3. It's important to ensure that the Munset system has access to an avatar API. This API enables the system to generate avatars of sign language gestures, enhancing the user experience. Therefore, the availability and integration of an avatar API are essential for the Munset system to provide accurate and visually appealing translations for deaf individuals.

Quality Assurance:

- Reviews:

Our team at "Monset" will conduct walkthrough reviews throughout the software development lifecycle to ensure the quality and effectiveness of the system. These walkthrough reviews involve the participation of peers such as designers, programmers, and other stakeholders.

The initial type of walkthrough is about the requirements. During this review, the team will go deeply in the user needs and requirements, their goal is to understand the desired solution and git red of any potential ambiguities or gaps within the requirements, ultimately working towards a precise solution that aligns with the users' expectations.

Next, the team will conduct a Design walkthrough, where they collaboratively evaluate the database architecture, user interface, and system architecture. The purpose is to align the design with the users' expectations and improve the software's usability. Through this walkthrough, the team ensures that the design is both easy and attractive to use.

Third, during the Code walkthrough our team will make sure that coding guidelines are followed, finding errors, and creating a dependable system.

Last Documentation walkthrough that we will make sure that technical specifications, manuals for completeness, accuracy are checked and giving users clear instructions to empower them.

By conducting these walkthrough reviews at various stages of the development Life cycle, the team aims to enhance the quality, functionality, and usability of the software, so delivering solutions that meet the users' needs.

- Verification:

Based on the verification process, which ensures that the software product achieves its intended objectives and is free of faults, we utilized two methods (inspection & walkthrough) in the Monset application to ensure software quality assurance (QA).

During the inspection phase, a dedicated team was assigned to examine the sign language matching system with the corresponding Arabic letters and ensure that it was perfectly correct. This feature holds significance in the Monset application due to its precision and potential error sensitivity, which could diminish the application's effectiveness.

As for the rest of the system, we will apply the walkthrough method, which allows us to benefit from expert and user feedback by actively engaging with the application and expressing their opinions impartially.

In the Monset application, we prioritize high-quality specifications and meeting requirements. Therefore, employing these two methods during the verification phase will enhance the application's effectiveness and desirability in the market.

- Validation:

Validation testing is the process of ensuring that the product meets the true needs and expectations of stakeholders.

In the Monset application, we prioritize validation throughout the system, starting from individual components and extending to the entire system.

We begin the validation process with unit testing, which involves testing small pieces of the software application in isolation, without relying on external systems. Unit testing focuses on components that do not interact with external databases, files, or networks, and it helps identify issues early in the development phase.

Once unit testing is complete, we proceed to integration testing, which aims to validate that different software components, subsystems, or applications work together seamlessly to achieve the desired functionality and performance. Integration testing helps detect and resolve issues such as compatibility problems, performance issues, incorrect communication, or data corruption that may arise when components are combined.

Finally, after validating individual units and their integration, we conduct system testing, which verifies that the application performs tasks as intended. System testing is a form of black-box testing that evaluates the functionality of the application as a whole, without focusing on its internal workings (which is the concern of white-box testing).

Acceptance Criteria:

i. Procedures:

We will make sure that the installation process is user-friendly and easily accessible. Furthermore, the software will be distributed through an app store, allowing users to install it effortlessly using the platform's installation mechanism like app store.

ii. Testing:

We will use a various type of testing, that we will conduct to ensure the quality of the software:

- User Acceptance Testing (UAT): Monset well tested in a real-world scenario by the end-user to make sure it works as expected.

- Alpha testing: is an early phase of software testing where a limited group of users, typically the development team or internal stakeholders, test the software in a controlled environment to identify major issues and validate its functionality before release.

iii. Training:

There is a video that teaches the user how to use Monset to be familiar with the system functionalities. These videos are clear and simple, with audio and text.

iv. Documentation:

We will apply it by providing a user manual for our system called MonsetGuid. This manual will serve as a helpful guide, explaining the system's features and providing instructions on how to use it efficiently. It aims to empower users by giving them clear and straightforward information, enabling them to make the most out of Monset.

Future Considerations:

When planning for the future of Monset, several key areas are being considered to improve its functionality, usability, and overall impact. Firstly, Monset is focused on enhancing the accuracy of sign language translations through continuous refinement of its SLR models, expanding training datasets, and integrating error correction mechanisms. This dedication to accuracy aims to ensure that users receive dependable and precise translations, facilitating effective communication in sign language.

Secondly, Monset aims to broaden its language support beyond its current offerings to accommodate the diverse range of sign languages worldwide. Collaboration with experts and communities in sign language linguistics will be pivotal in expanding the system's linguistic coverage and promoting inclusivity for users from various regions and cultural backgrounds.

Additionally, Monset envisions the implementation of Real-Time Feedback mechanisms to offer users immediate guidance and assistance during sign language translation. Features like instant corrections, alternative sign suggestions, and

interactive learning tools will empower users to enhance their signing proficiency and improve their communication experiences in real-time.

Lastly, Monset is committed to fostering Community Engagement by actively involving users, educators, researchers, and advocates in its development and evolution. By encouraging collaboration and contributions from the community, Monset aims to enrich its functionality, language coverage, and educational resources, ensuring its continued relevance and impact in the realm of sign language communication.

References

[1] The easiest free Gantt chart maker online," TeamGantt, https://www.teamgantt.com/ [Accessed Feb. 17, 2024].

[2] "مؤسسة الأميرة العنود الخيرية," تمكين, http://alanood.org.sa/Ar/Pages/home.aspx [accessed Feb. 17, 2024].

[3] الترجمان في لغة الاشارة، [Application]. [Accessed February 2024]

[4] R. Reynoso, "A to Z Artificial Intelligence Terms in Technology Space," 17. Available: https://www.g2.com/articles/artificial-intelligence-terms. [Accessed February 2024].

[5] *Intelligent diagramming Lucidchart*. Available at: https://www.lucidchart.com/ (Accessed: 31 March 2024).

[6] Ready *Player Me API*. Available at: https://docs.readyplayer.me/ready-player-me/api-reference/rest-api (Accessed: 31 March 2024).

[7] *The Collaborative Interface Design Tool Figma*. Available at: https://www.figma.com/ (Accessed: 02 April 2024).

[8] Garimella, M. (2022) *Sign language recognition with Advanced Computer Vision*, *Medium*. Available at: https://towardsdatascience.com/sign-language-recognition-with-advanced-computer-vision-7b74f20f3442 (Accessed: 02 April 2024).