TED UNIVERSITY

CMPE 491

Senior Project

Gesture Guide: Virtual Assistant for the Hearing-Impaired

High-Level Design Report

Submission Date: 07.01.2024

Team Members:

- Berke Lahna
- Canberk Aydemir
- Ceyhun Toker
- Mehmet Fatih Ülker

Advisor: Venera Adanova

Jurors: Gökçe Nur Yılmaz and Eren Ulu

Web Site: https://berkelahna.github.io/Pages

TABLE OF CONTENTS

1. Introduction

1.1 Purpose of the System

Purpose of the gesture guide system is to provide an interface that will equalize the conditions of communication of the hearing impaired. Considering the problems hearing impaired comes across in their daily lives, the hearing impaired can be aided in a way that solves all of their problems in an efficient way. For this purpose, we have proposed the gesture guide application which would provide a specialized virtual assistant and a real time sign language interpreter. Considering the idea behind gesture guide we can state that the purpose of the system is to provide an interface that is specialized for hearing impaired, by providing a sign language interpreter and a virtual assistant specifically designed for their use.

1.2 Design Goals

Design goals for the gesture guide application is to create a natural environment for the hearing impaired to communicate with people. By providing both a virtual assistant and a sign language interpreter the gesture guide aims to create a robust system that can aid the target users. For better user experience we aim to have a both simple and efficient application that can help anybody in the hearing-impaired community. From the technical aspect the design goal of the gesture guide is to create an application that uses the force of machine learning based models to deliver solutions to object detection problem. Overall, our design goals are to both create a technically robust system and a system that will help users in their daily lives.

1.3 Definitions, Acronyms, and Abbreviations

Abbreviations

- YOLO: You Only Look Once
- UI: User Interface
- CNN: Convolutional Neural Networks
- ANN: Artificial Neural Networks
- GPU: Graphics Processing Unit

- CPU: Central Processing Unit

For detailed definitions please refer to the glossary section located at the end of this document.

1.4 Overview

This document as a whole is dedicated to completely the design of the gesture guide application. The document goes through robust details of the proposed system ranging from subsystem decomposition to boundary conditions. Throughout the document we will be demonstrating all aspects of the system to obtain a complete understanding of the implementation phase of the project. With a focus on the whole system, we will be decomposing this system into smaller pieces and investigating all of these parts in detail. All the questions regarding the design of the gesture guide will be examined subsystem-by-subsystem. Nonetheless this document will be the blueprint for all the later phases of this project from low level design to testing.

2. Current System

In our previous project documentation, we indicated that the process of detecting sign language through video, which is one of the key functions we aim to implement in the final version of our project, did not meet the desired performance level. We expressed our desire to make this process faster and more stable. Additionally, another important aspect we focused on was the creation of an interface that the user will use and seek assistance from. Another feature we wanted to add to the development was the login system, which we considered important for privacy rights. Between the last documentation and the current status, efforts have been made and additions have been implemented to these features. To elaborate on these:

Implementation of Model Selection Menu

We added a model selection menu to the opening page of our application, enabling the transition between two important functionalities. With this menu, the user can choose between the options "Model to Text" and "Model to Activity." By selecting "Model to Text," the application transforms hand gestures into text, while choosing "Model to Activity" allows fulfilling requests such as SMS, CALL, and CALENDAR actions. The Confidence Level set for the model is "80," and when a model with a confidence level below this threshold is selected, we added a warning message displaying "unsure." Additionally, beneath this warning message, we included a "Delete Last Letter" button to facilitate the user in removing the last letter.

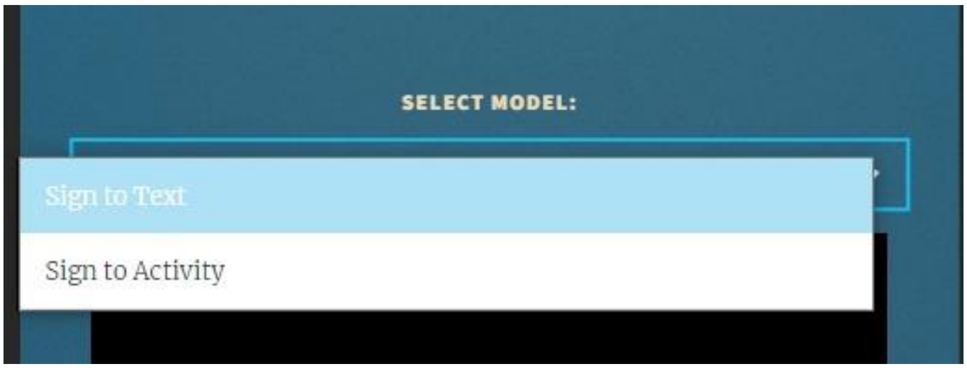

Figure 1: Image of Model Selection Menu

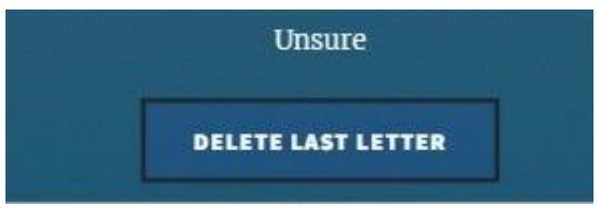

Figure 2: Image of Delete Button

Icon update

In the current process of our project, we made some visual enhancements, thinking that adding them would make our application more appealing to users. Some of these additions include adding icons. Alongside CSS, we applied different styles to the icons based on screen width using media queries. In this way, we worked on achieving a responsive design method, ensuring that the icons appear properly in different sizes.

Figure 3.1: Icon of our mobile app **Figure 3.2:** Second Icon Designed

Implementation of Login Page

As specified, having a login page in our application is necessary and important. Creating this function in its basic form, for future development, was important for us. In this regard, we created a simple login page with basic authentication functionality. While making this improvement, we also needed to make changes and enhancements to the page design. We used stylesheets for making changes to the visual appearance. To create the login section, we used JavaScript functions and components like "button." For the automatic resetting of the login section when the page is refreshed, we implemented automatic redirection through the HTML page.

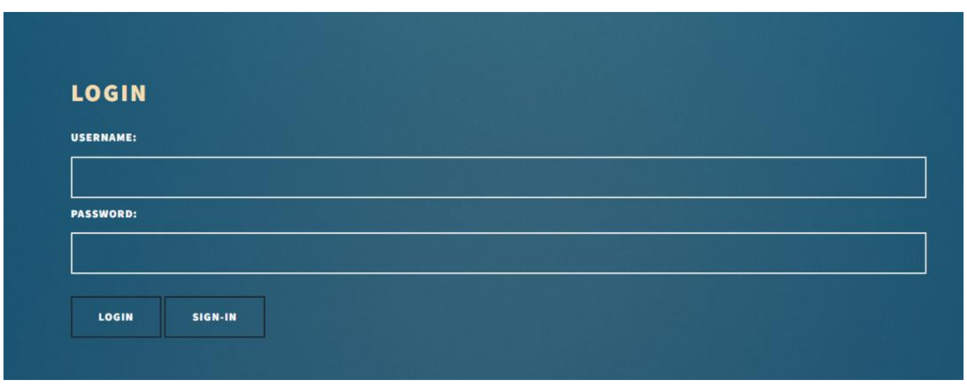

Fig 4: Image of Login Part

Virtual Assistant

We mentioned that the primary goal of our project is to facilitate the lives of the hearing-impaired. Currently, we have taken the initial important steps to achieve this. Through the Model Selection Menu, users can now select the "Model to Activity" option, allowing them to send SMS, make calls, and access the calendar. We plan to increase the functionality of these critical functions in the future by making new additions.

In conclusion, significant steps have been taken towards realizing the important functions and developments promised in our project in the current status. In the upcoming period, we aim to enhance these developments and place additional emphasis on the design area. As mentioned above, it is necessary to add new options to the Model Selection Menu and increase the transition speed and functionality among these options. Producing new designs and visuals in addition to the added icons and making visual adjustments functional by giving importance to the CSS part are also among our goals. We are aware that the Authentication and Login section, which is currently in a basic and prototype state, can be made more serious by integrating it with the backend server and database. We can say that the current state of the application is much better than the one mentioned in the previous documentation.

3. Proposed System

3.1 Overview

Gesture Guide is a project that heavily depends on machine learning algorithms and computer vision methodologies, but at the end of the day it is nonetheless a software system. In this system while there are machine learning models and preprocessing tools there are also systems like a user system to handle users and their interactions with other sub-systems. So, we can say that the Gesture Guide project as a software system is somewhat of a melting pot featuring distinct systems that handle unique aspects of the overall system. This section is dedicated to demonstrating these systems, establishing the hardware software mapping, constructing the database structures and demonstrating other aspects of the overall system.

3.2 Subsystem Decomposition

Being compliant with the functional requirements as well as the software models mentioned in the project analysis report, there can be 5 sub-systems that interact with each other to create the bigger system of Gesture Guide. These sub-systems are the sign language interpretation system, virtual assistant system, gesture recognition system, preprocessing system and user system. This section focuses on demonstrating these systems on the surface level and establishing the relationships of sub-systems. Figure 5 bellow also displays the sub-systems and their interactions with each other.

- **Sign language Interpretation System:** System that is responsible for predicting sign language given a picture or a frame. This system is a machine learning model that has been trained specifically in sign language to make accurate predictions. Sign language interpretation system works with the preprocessing system which is concerned with obtaining a frame and applying necessary preprocessing like augmentation to make better predictions. At the same time users interact with this system so it also depends on the user system.
- **Virtual Assistant System**: Virtual Assistant System is the system that handles the basic virtual assistant tasks like opening a specific app or directing users to a webpage. This system works heavily dependent to the gesture recognition system to create a response to a gesture displayed by the users. While the gesture recognition system makes prediction on a gesture, virtual assistant system executes the mapped function. Also, users interact with the virtual assistant, so user system also interacts with this system.
- **Gesture Recognition System:** This system is the second system that is a machine learning model. This system is responsible for making predictions on gestures to create the functionality of the virtual assistant. This system also works coordinately with the preprocessing system to make accurate predictions. This system works as a part of the virtual assistant system meaning that users do not interact directly with this system. Therefore, this system has no dependencies towards the user system.

- **Preprocessing System:** Model accuracy is a great concern for the Gesture Guide application considering the target users as well as emergency situations. To get proper input from users and preprocess the input using augmentation or gray scaling, a preprocessing system is required. This system is a crucial part of making predictions for both of our models. Therefore, this system works with both sign language interpretation system and the gesture recognition system to make better predictions and obtain better recall and precision scores.
- **User System:** In order to manage users and their interactions with other subsystems there should also be a system that is specifically targeted at users of the application. This system is the only sub-system that is directly linked to a database. Users should be represented as entities in the database in order to appropriately manage user data. Operations like authentication, fetching user details, changing user preferences, enabling users to edit their profiles are some of the functions handled by this system. Users interact with all the aspects of the overall application therefore this system depends on almost all of the other sub-systems.

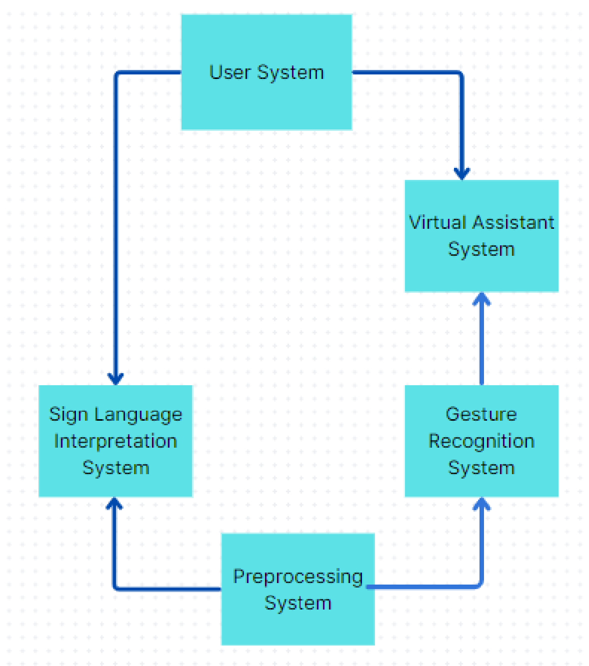

Figure 5: The system decomposition for Gesture Guide system.

3.3 Hardware/Software Mapping

Considering the device compatibility issues, hardware software mapping requires deeper analysis for Gesture Guide application. For this cause we need to investigate hardware software mapping for model training, web application, android application and the back-end server.

Model Training: For model training we will be using CNNs and YOLO, for this purpose a good amount of computing force is required. Since the GPU crucially improves upon the time spent on training, we will be using GPU for training rather than CPU. What takes hours for a CPU to compute can take only minutes when training with GPU. That makes the GPU an important hardware component for model training considering we have large datasets and have to train for multiple hundreds of epochs.

Web Application: The web application depends on hardware at a surface level. Multiple sub-systems require the usage of device camera. For this purpose, we have chosen to work with the OpenCV eco-system. Using OpenCV we can obtain feed from the webcam. Then this feed can be redirected to preprocessing system that is concerned with getting input ready for making predictions. Nonetheless the main hardware the web application is dependent on is going to be the webcam. Webcam will work as the source of gathering required input. Also, the application will run on the browser and will use the browser engine just like every other JavaScript application.

Android Application: For android devices we will be utilizing the device camera just like in the web application. The device camera will create frames from the video input that will be fed to the preprocessing system so that predictions can be made properly. Application will be using the computing force of the device at all the times. So, device CPU takes the crown as the most important hardware component that is responsible for the application.

Backend Server: In order to encapsulate the users and their interactions with the sub-systems, we require a user system. The most important aspect of the user system is a backend server and the database server the backend server interacts with. The backend server can be a computer, but it can also be leveraged to cloud platforms for universal access. For the scope of our project at least initially the backend server will be a computer running a process but if we can find the opportunity, we can leverage the back-end server to a cloud platform. The cloud-based backend server would be a better choice for completely delivering some of the non-functional requirements.

Now all the hardware software mappings are constructed we can better visualize this mapping by the help of a UML deployment diagram. Figure 6 bellow shows the hardware software interactions of the system displaying the components mentioned in this section with the addition of a database server (For database server refer to persistent data management section).

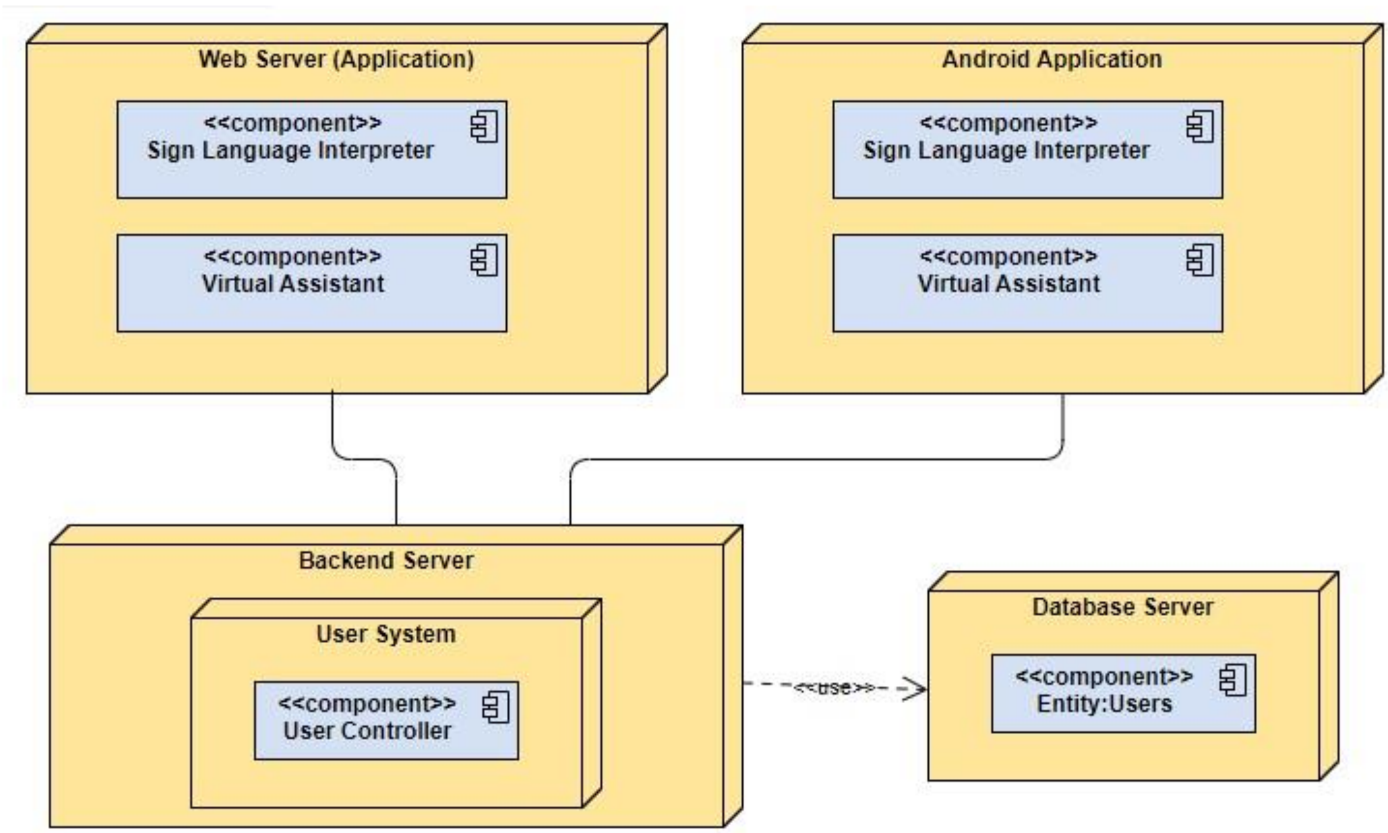

Figure 6: The proposed UML deployment diagram.

3.4 Persistent Data Management

Even though the Gesture Guide project does not require complex data storage there is still a need for a persistent data management especially for storing user information. The sub-systems such as preprocessing system or machine learning based systems do not require a persistent data storage due to their dynamic structure, but the user system is a system that deeply requires data storage.

For user sub-system to carry out user operations, it needs a dedicated user database so that the user information is stored and manipulated depending on the needs of the users.

Database Design: The persistent data in our application is the information about users ranging from date of birth to passwords as we have mentioned before. This persistent data should be stored in a DBMS. The database system can be relational as in SQL based DBMS languages, but it can also by non-relational or NoSQL. For our purposes we have chosen a relational database management system, which is PostgreSQL. What makes PostgreSQL a great choice is the fact that it is fairly easy to use as well as open source.

If we need to specify the user as an entity in the database, it can be modelled with fields user_id as a primary key, first_name, last_name, date_of_birth, email, password(encrypted) and role. Whenever a user registers, their information are stored as record in the database and whenever this data needs be fetched it gets fetched directly from the database. This way the whole data lifecycle for users are established. Figure 7 bellow displays the proposed user entity.

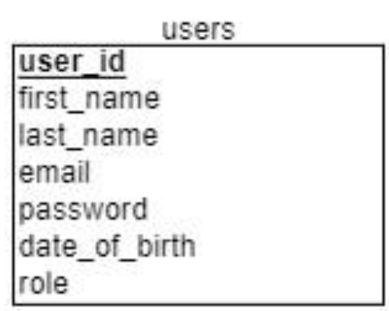

Figure 7: The user entity displayed as a part of relational schema.

3.5 Access Control and Security

Security is another crucial component for the proposed system to ensure the protection of user data as well as to handle proper access to the system through authentication and authorization. For this section let us go through authentication, authorization and possible security schemes for the gesture guide application.

Authentication: Whenever users are trying to login to the system, they need to enter their credentials and their credentials should be validated. This validation is the process where user credentials are searched in the database by a specific field like email and then compared with the entity who has matching emails. If the validation fails users can not gain access to other features of the system and are required to enter their credentials again or create an account. The access control scheme here only allows the login page for the unauthenticated users.

Role Based Authorization: For gesture Guide we have not declared specific roles for users but for this context we can declare two roles which are users and admins. Users are the casual users of the system who have authorization to use the features of the application but do not have the authority to change anything about the system state. Where admins are users who possess all kinds of privileges like editing specific user's information or manipulating the machine learning models used by the sub-systems. This discrimination between roles is the blueprint for constructing the access control based on roles. Whenever users are logged in, they get granted authority based on their role. And based on their authorities they can interact with different aspects of the system.

Authentication and authorization can be constructed simply by a basic name-password mechanism, or it can also be constructed in a token-based manner. Since token-based authentication is way more secure compared to casual name-password authentication we have chosen a token-based approach which is Oauth2. Oauth2 based login would be the best choice for this mechanism thanks to its high security standards. This way we can add authentication through Google or GitHub accounts while also leveraging the security of Oauth2.

3.6 Global Software Control

For global software concerns we will be heavily focusing on the subsystem interactions while also addressing other aspects of global software control.

- **Subsystem Interactions**: For proper global software control, establishing the interactions between sub-systems is a great concern. When having multiple subsystems, the systems should have no problems running concurrently and sequentially. For example, preprocessing system works sequentially with both the sign language interpretation system and the gesture recognition system, therefore when implementing the preprocessing system we should make sure that it can work sequentially with the other subsystems. To better demonstrate the subsystem interactions, we have created three figures which are Figures 8, 9, 10. Figure 8 shows the interactions related to preprocessing system, Figure 9 shows the interactions related to gesture recognition and Figure 10 shows the workflow for sign language interpretation.

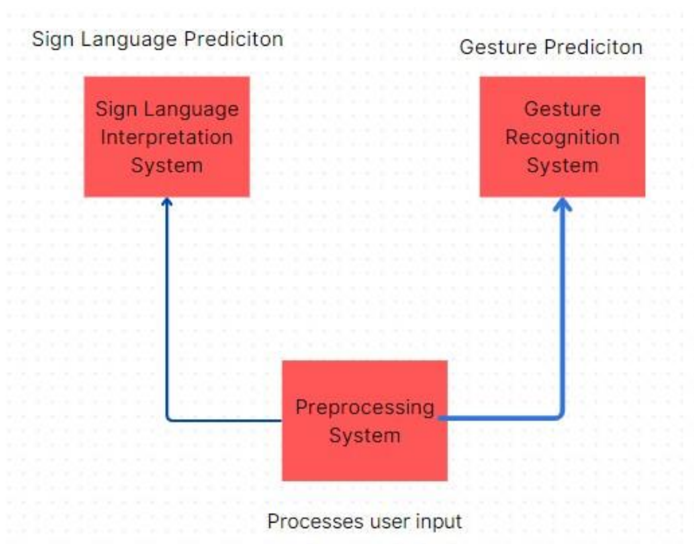

Figure 8: The preprocessing system's influence on machine learning based systems.

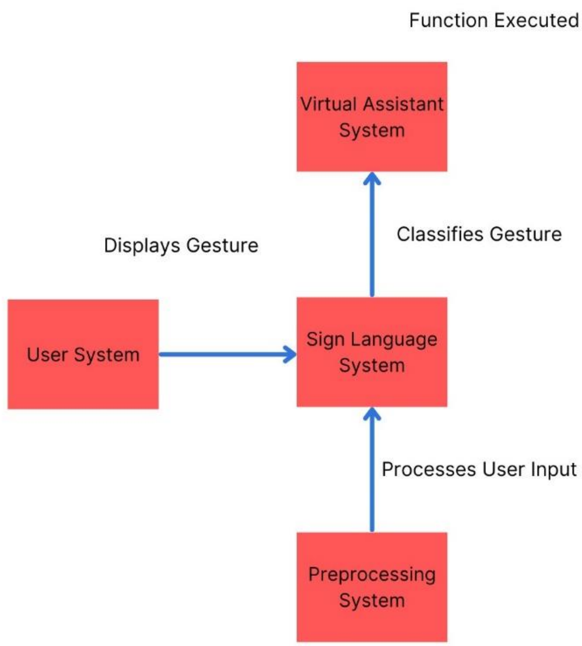

Figure 9: The interactions between systems for the virtual assistant lifecycle.

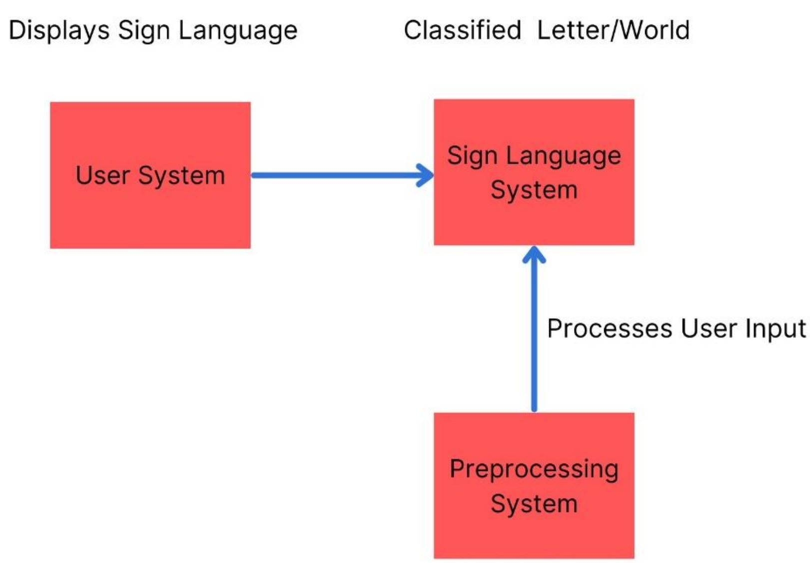

Figure 10: The sign language interpretation workflow.

- **Other Concerns**: Security is another concern which should be handled at the global scope of the application so that the whole application is secure. The security mechanisms we propose earlier complement global software control. Finally, we can also consider leveraging our services to a cloud provider so that we can better control the global software. Since cloud brings a lot of flexibility applying global software control would become easier.

3.7 Boundary Conditions

Boundary conditions encapsulate the unexpected conditions that can arise upon interactions with the system. To better understand the boundary conditions the expected flow of the system should also be considered to a greater extent. Upon investigating the casual flow of the system, we have come up with 3 possible boundary conditions which can occur when logging in, when trying to predict a sign or a gesture, and when obtaining input through device camera.

- **Boundary Conditions When Logging in**: Users can come across various problems while trying to log in to the system. The password cannot be validated due to a problem occurring on the database side or the backend server which leads to users not accessing the sub-systems of the applications. Another scenario is when users enter their credentials in the wrong way and cannot gain access, this scenario is not specifically a boundary condition but still leads to users not accessing the system. Overall, there are a lot of boundary conditions that can occur when logging in whether it is caused by the system or the users themselves.

- **Boundary Conditions When Making Predictions**: Making predictions is the most important aspect of the Gesture guide application. In the first case, unexpected objects in the frame can be predicted to be a hand, therefore a completely false prediction can occur. Another case is when a sign or a gesture is displayed and is falsely classified to be a part of another class which is a classification error. For overcoming these problems machine learning models should be hyper parameter tuned based on metrics such as F1 score, precision, recall and MAP-50.

- **Boundary Conditions When Using Device Camera:** Device camera is a very important hardware component for the gesture guide application. When using device camera whether on the web or the mobile, there can be boundary conditions happening due to the hardware components and the device camera. Device camera might be corrupted and cannot be getting input accurately, leading to mispredictions. As a hardware component device camera can obviously cause boundary conditions.

4. Subsystem Services

This section is concerned with investigating every sub-system and demonstrating their services. The sub-systems that will be explained are the same as the ones we have declared before which are preprocessing system, user system, virtual assistant system, sign language interpreter system and the gesture recognition system.

- **Preprocessing System**
  - **Gathering Input:** The preprocessing system is the system that interacts with the device camera in order to both obtain input and then apply necessary preprocessing to enhance model predictions. For these purposes the most powerful computer vision ecosystem, which is OpenCV, can be used for processing a video or a webcam feed or a picture and applying necessary computer vision methods. By obtaining the input though device camera the preprocessing system feeds the machine learning models.
  - **Image Rescaling:** Preprocessing system may need to rescale images so that images have the same dimension as the images of the training set. This way the preprocessing system can improve the prediction accuracy.
  - **Gray scaling:** The images can be turned into gray scale domain by the preprocessing system in order to improve upon prediction accuracy. Applying gray scaling to images have shown to improve the prediction metrics especially if the model is trained on an augmented dataset.

- **User System**
    - **Gathering User Information:** The user system is the only sub-system that requires a dedicated database where information about users is stored. The user system requires fetching user records for validating users, updating user information and obtaining virtual assistant history. For these purposes the user system runs the required query on the database in order to obtain the required user information.
    - **Validating Users (Authentication/Authorization):** When users are logging in, credentials they enter should be validated by the user system. Initially a query is made to the database to see if a record matches this information, if the records does not match the user is not permitted to access the system. If the record exists, the user should be granted authority based on the role compliant to the role-based authorization. The whole authentication and authorization scheme is handled by the user system with the help of the database.
    - **Updating User Information:** The user system should also be able to manipulate user information on the system. For this purpose, the user system makes multiple queries to both find the user and then update the user entity. By this process the user system can safely edit user information and make sure the user entity on the database is updated as well.
- **Gesture Recognition System**
    - **Classifying Gestures:** The gestures refer to a motion displayed by hand movements. As a part of our virtual assistant, we have proposed a machine learning model that would classify specific gestures to execute a mapped function. The proposed model would be trained in multiple gestures to classify them. This model can be considered as a system due to the importance of gesture classification on the virtual assistant. The system’s only task is getting a gesture input through the preprocessing system and predicting this gesture correctly. Due to the nature of classification problems, model accuracy is the greatest concern of this system. For better predictions the models should be fine-tuned and should demonstrate low class loss metrics.
- **Sign Language Interpreter System**
    - **Predicting Letter Level Sign Language:** Letter level sign language encapsulates the specific letters which can be represented as images in our context. One goal of this system is to make predictions on images obtained through the preprocessing system in order to classify the letter displayed in sign language. At the end of the day this system is a machine learning model which makes classification metrics very important. For the purpose of letter predictions, we have trained models using CNNs through YOLO.

  - **Predicting Word Level Sign Language:** Word level sign language is the subset of sign language that is concerned with words. Words are represented as motions in sign language. This means that in the context of this system a word is a whole video that needs to be classified. This service is concerned with making predictions on video to predict words of the sign language. The video classification of this system can be handled using sequential models or a complex frame-by-frame analysis.
- **Virtual Assistant System**
  - **Executing Mapped Function:** Virtual Assistant system works cooperatively with the gesture recognition system to create the functionality for the specialized virtual assistant. After gesture recognition system makes a prediction, the virtual assistant system is responsible for executing a mapped function. For example, whenever a wave gesture is displayed the virtual assistant should open the calendar app. So, the system basically works as the motor neurons if the gesture recognition system is the sensory neurons.
  - **Getting Virtual Assistant History:** Another function of the virtual assistant system is to obtain users previous usage history. For this purpose, the virtual assistant system interacts with the user system to obtain the user information from the database and correctly display previous virtual assistant commands. Figure 11 displays the system services in the form of a hierarchy diagram.

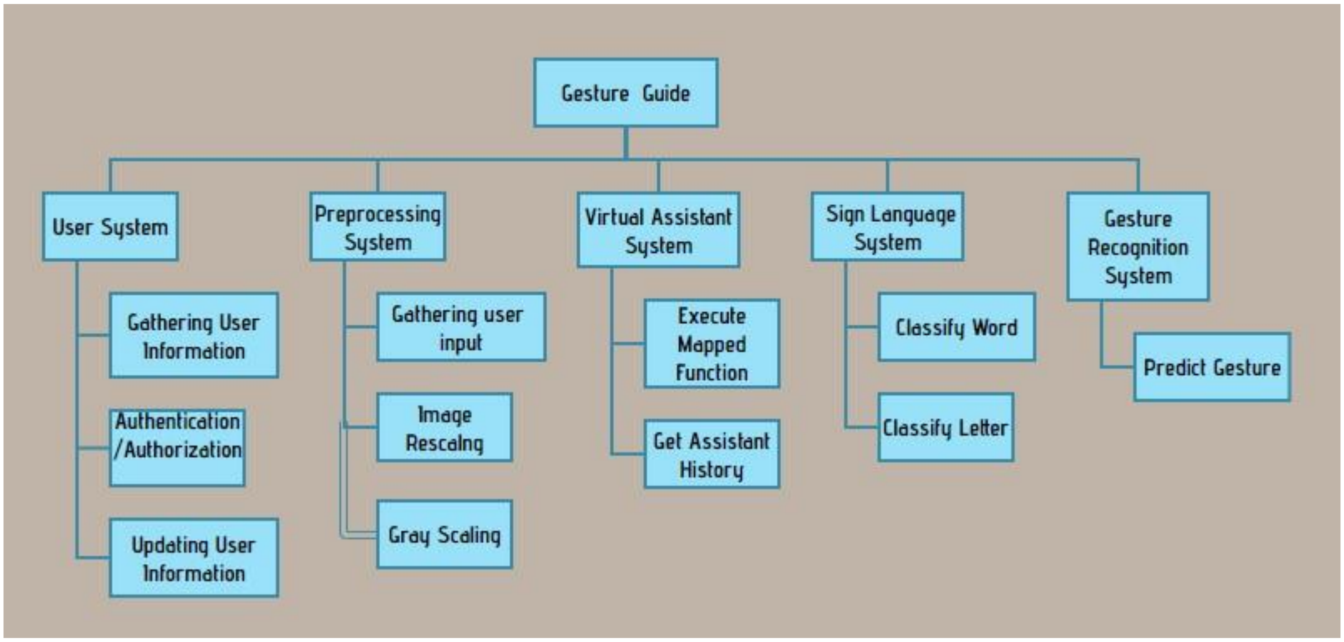

Figure 11: System decomposition and subsystem services.

5. Glossary

- **Authentication**: In a software system, validating if a user exists in the system.
- **Authorization**: In a software system, granting authority to users based on their roles.
- **Classification**: In machine learning, the process of determining which class a sample belongs to.
- **Credentials**: Personal information that users use to register/log in to a system. Credentials are a crucial part of the authentication process.
- **Convolutional Neural Networks**: Neural Networks that use the benefits of digital signal processing methods to make predictions on images.
- **F1 Score**: A weighted sum of recall and precision metrics that are used for classification problems.
- **Model (Machine Learning)**: An algorithm that is used to predict on regression or classification tasks.
- **Artificial Neural Networks**: Machine learning models that is inspired by the central nervous system which consists of nodes called neurons.
- **Recall**: A model classification metric that focuses on number of false negative predictions (Type 2 error also knowns as Beta).
- **Precision**: A classification metric that focuses on false positive prediction (Type 1 error also known as Alpha).
- **Yolo**: Short for You Only Look Once, a single shot object detection model that does not use anchoring method like the RCNN models.
- **MAP-**50: A metric that is heavily used for the YOLO models, mean precision at 50 percent intersection over union.

6. References

- Sommerville, I. (2016). Software Engineering, 10th edition. Pearson Education Limited. ISBN 10: 1-292-09613-6
- Martin, R.C . (2002). UML for Java Programmers. 1'st edition. Prentice Hall
- Bruegge, B et al. (2004). Object-Oriented Software Engineering, Using UML, Patterns, and Java, 2nd Edition. Pearson