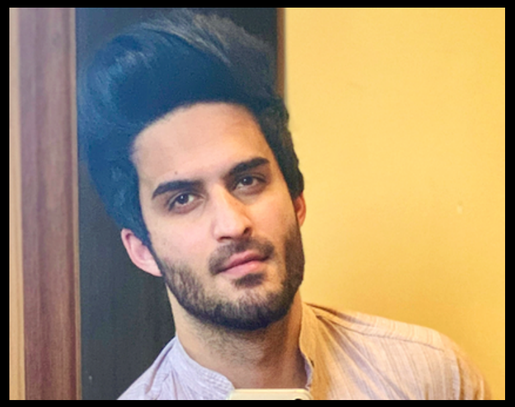
GESTURE AI

HELPING DEAF AND DUMB

A MACHINE LEARNING MODEL

Team Members:



MOHAMMAD ZARAK

KUNJ SHAH

GAAYATHRI CHALASANI

SAI PRASANTHI

Introduction

Sign Language Recognition Using Python

There have been several advancements in technology and a lot of research has been done to help the people who are deaf and dumb. Aiding the cause, Deep learning, and computer vision can be used to make an impact on this cause. This can be very helpful for the deaf and dumb people in communicating with others as knowing sign language is not something that is common to all, moreover, this can be extended to creating automatic editors, where the person can easily write by just their hand gestures.

Implementation Plan

Phase I

- Generating Dataset
- Pre-processing Data
- Cleaning Data
- Data Analysis

Phase II

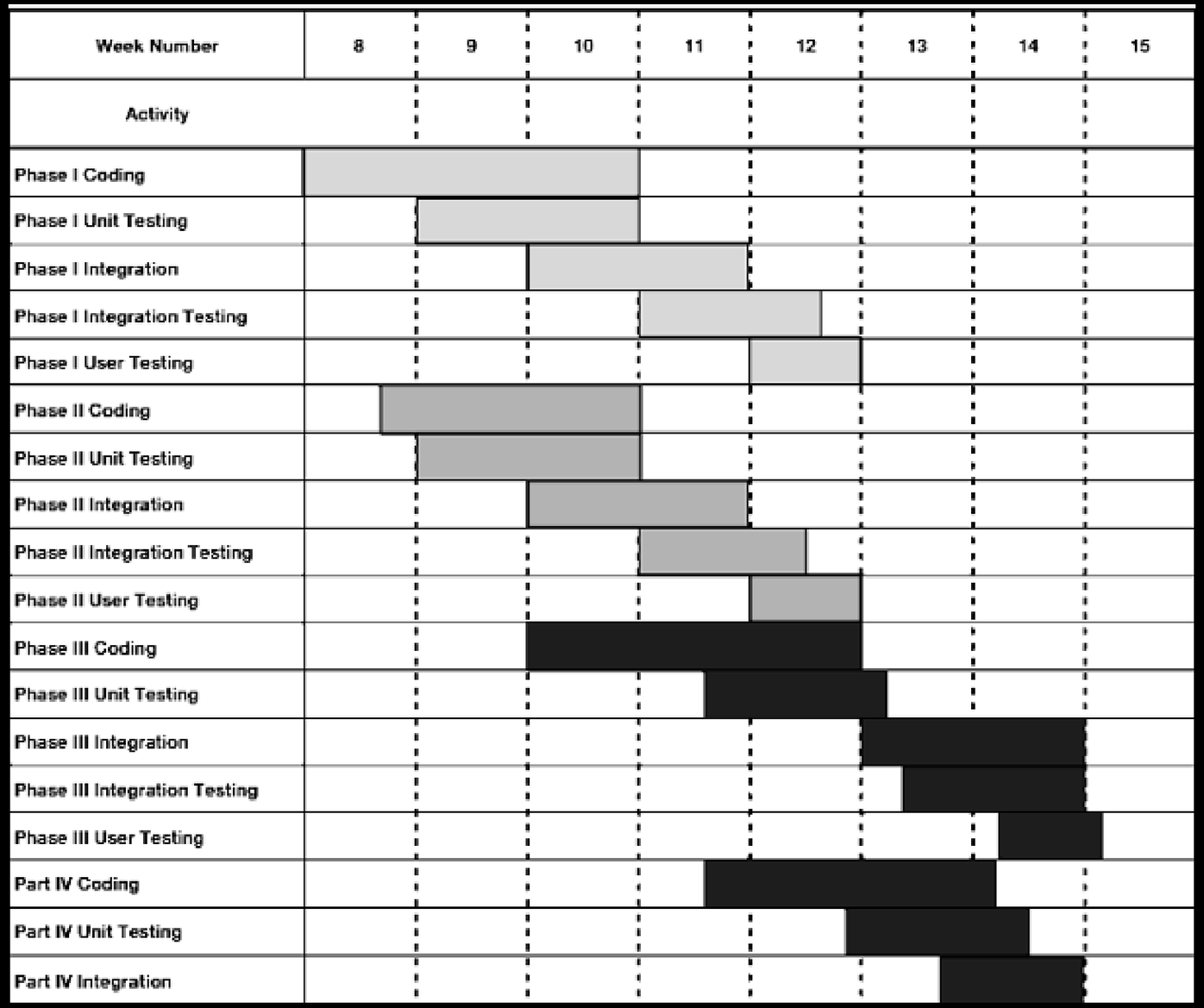
- Choosing Model
- Creating Model
- Training the model
- Validating the model
- Testing model on the dataset
- Checking accuracy of the model

Phase III

- Creating a local machine web application
- Intergrating the model and the web application
- Creating a basic front end
- Testing the web application

a little bit of body text

Implementation Breakdown



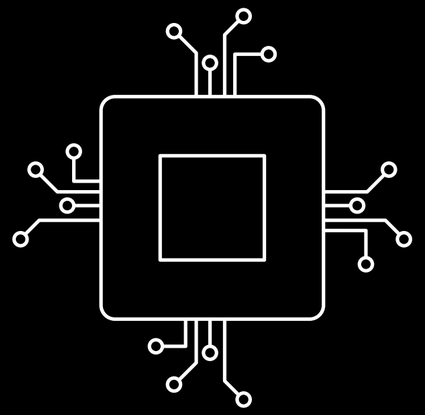
Implementation Overview

The implementation of the application and model will be approached in a “Top-Down” fashion, where the Phase I would be accomplished first and then moving down further accordingly as shown in the implementation plan. Individual stubs will have to be written to test some methods and the correct performance of the main classes and how they are interacting with the overall end-product. Our implementation will be broken up into 3 phases.
The first phase will consist of the mainly generating the dataset and then preprocessing it, creating a proper dataset will be the driving force behind the software.
Phase two will consist of the creation of the novel Machine learning model that will be trained and tested on the dataset that is being created in the phase I. This model will help us in predicting the gestures that a user is doing via the device camera and the predicted alphabets would be shown above on the device screen.
Phase three will contain the remaining portions of the use cases. Unit testing will be performed along the way a simple web user-interface would be created that would help us to use this model via a local web page coded in Flask.
All coding will be done through Jupyter Notebook and Vs code on the Windows platform and the coding will be done in Python and Flask framework. Testing will be done on a Windows Laptop to get real time results and to aid in visual and program debugging.

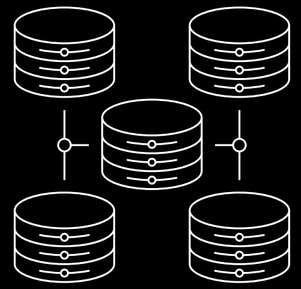
Design Goals

System	Design Goal	Description
User Interface System	• Effectiveness	• To increase the accuracy and completeness
	• Efficiency	• To reduce the resources expended
	• Satisfaction	• To ensure users' comfort and acceptability of use
Instructional System	• Clarity	• To make learning materials clear
	• Impact	• To increase users' attitude

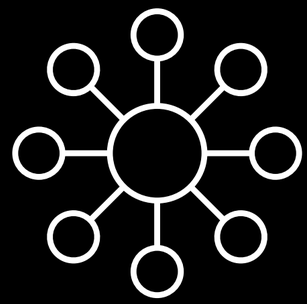
Objective:



Least hardware



Large scability



Variety of situation

Design changes overtime

- The first design was to take keypoints (bone structure) of face, arm and hands but with time we understood that to keep the dataset usable with achievable storage we removed the keypoints of face as it was generating really large files with the keypoints of the face.

- For the sake of getting greater accuracy the keypoints of arm were also removed so that the hand gesture could be predicted more accurately by the model.

- First we wanted to deploy the project online but as to deploy such a web-application the domain space needed to be purchased, therefore the project was run on localhost only.

System requirements

Identifier	Priority	Requirements
REQ-1	5	The program should be able to use the device's camera to take the key points from the data processed to it.
REQ-2	5	The program must have the microphone of the device accessible very time an audio option is selected for giving data to the system via audio source.
REQ-3	4	The program must have a user-interface, where the user can use the device camera for showing gestures so that the model takes that data and converts it into text.
REQ-4	1	The program's GUI should also have a textbox, where the user can type in the text which could be converted to gesture language.

REQ-5	**2**	**The program should be asking the user whether he wants to allow the software to use the device camera or not**
REQ-6	**4**	**The program should be predicting the gestures accurately.**
REQ-7	**1**	**The program should be able to recognize the hand of the user correctly**
REQ-8	**2**	**The program should be able to use the speaker of the device**
REQ-9	**1**	**The program should have a drop box for switching from Self-training option and Gesture option**
REQ-10	**2**	**The program should have an option for choosing where the model currently should be whether at collecting data, training data or testing data**
REQ-11	**2**	**The program should have a scale where the user can select how many samples would be taken for self-training.**
REQ-12	**2**	**The program should also have scales for test ratio, layers and neurons to be selected for the current self-training regime being done.**

REQ-13	2	The program should also have provision to generate audio from he sign language captured for people with low or no eyesight.

Non-functional requirements:

Identifier	Priority	Requirements
Req-1	4	The program's response time of the model should be fast
Req-2	3	The program's model should be scalable and be able to handle large amounts of data
Req-3	3	The program's code should be easily manageable and understood by other programmers
Req-4	4	The program should be able to provide security to data taken from users.
Req-5	3	The program should be able to read each gesture in 1 minute or less than that.

Req-6	5	The program should be easily running on low-end devices.
Req-7	3	The program should be able to optimize the memory usage of the system.

Enumerated User Interface:

Identifier	Priority	Requirements
Req-1	5	The program's user-interface must be simple to understand and use
Req-2	4	The program should have a consistent theme for all pages.
Req-3	3	The program's interface must be able to adjust according to different devices and different screen sizes.

Class list:

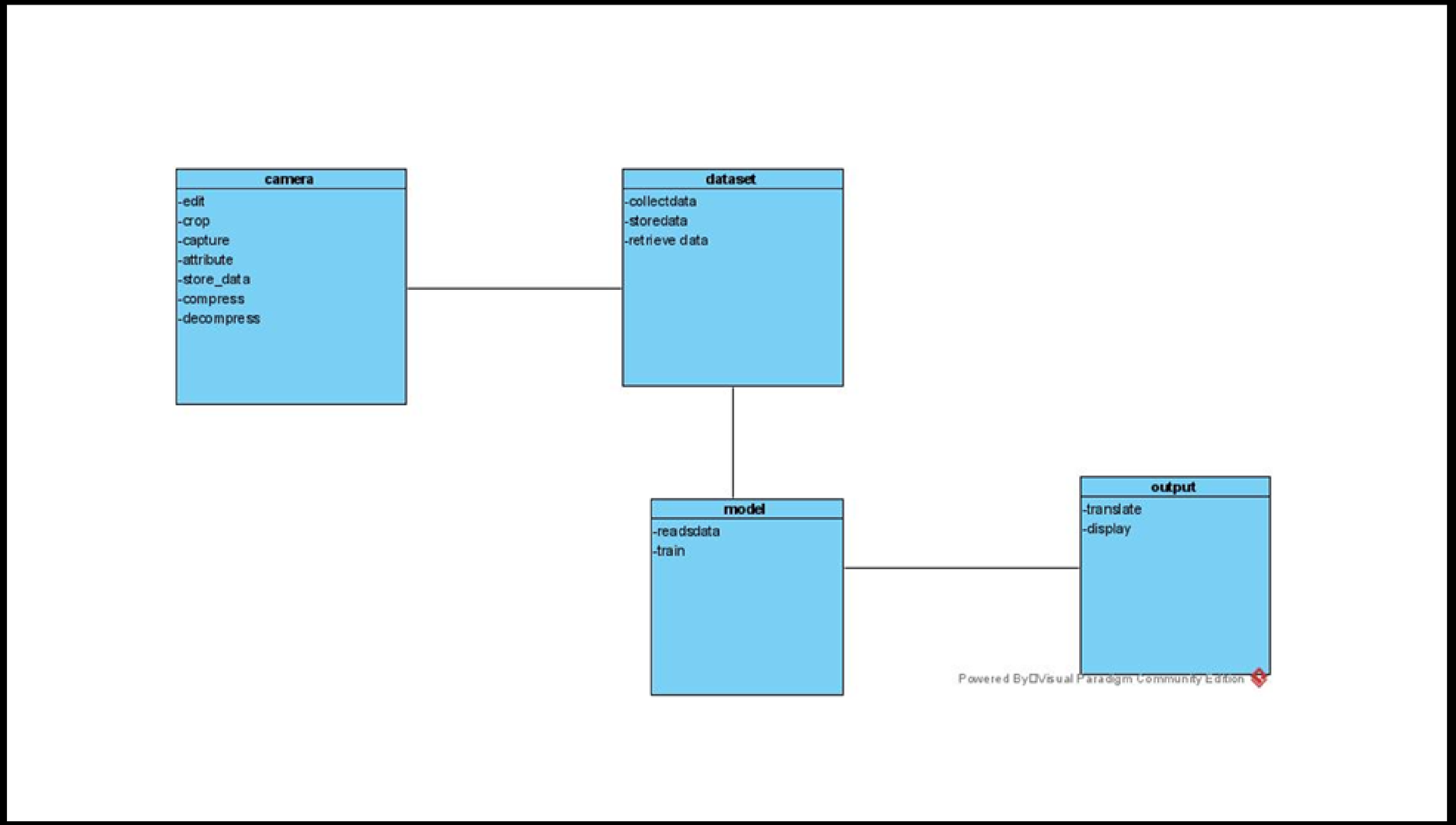
Camera: edit, crop, capture, attribute, store_data, compress, decompress

Dataset: collect data, store data, retrieve data

Model: reading data, train, test

Output: translate, display

Class diagram:

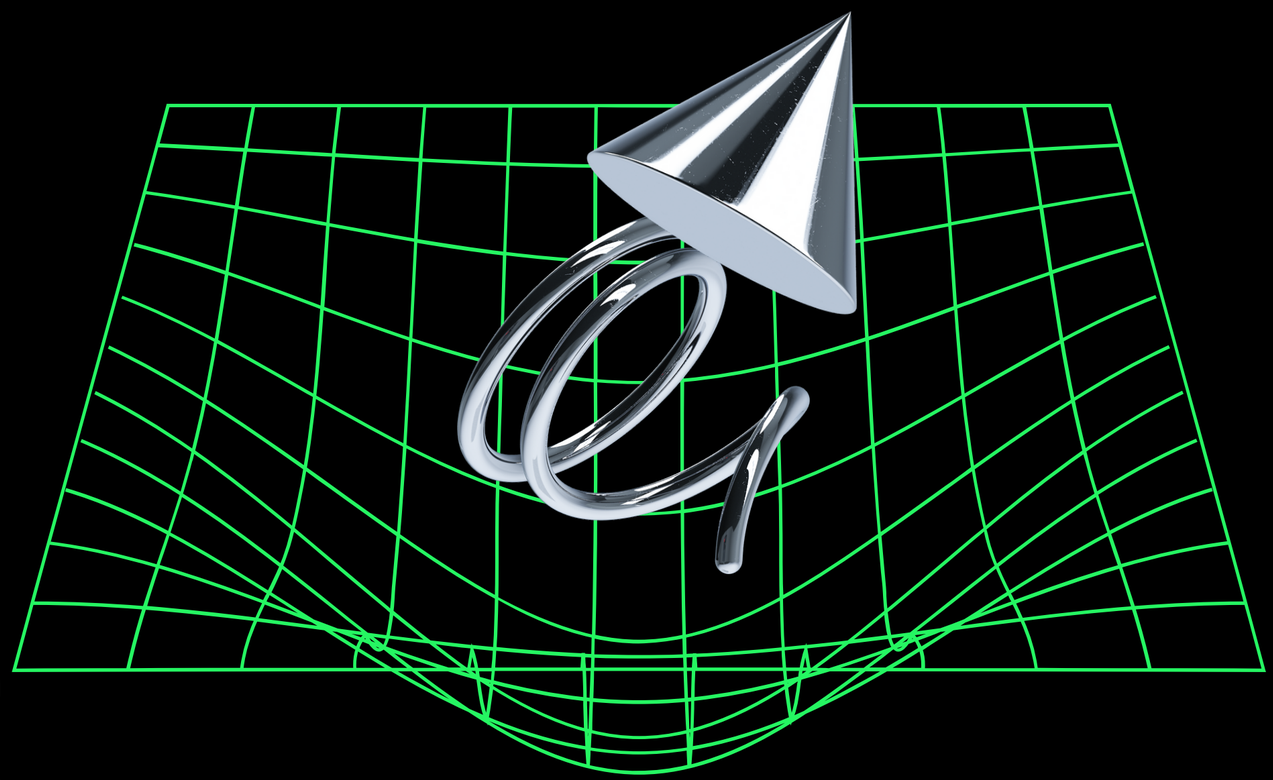


Use cases:

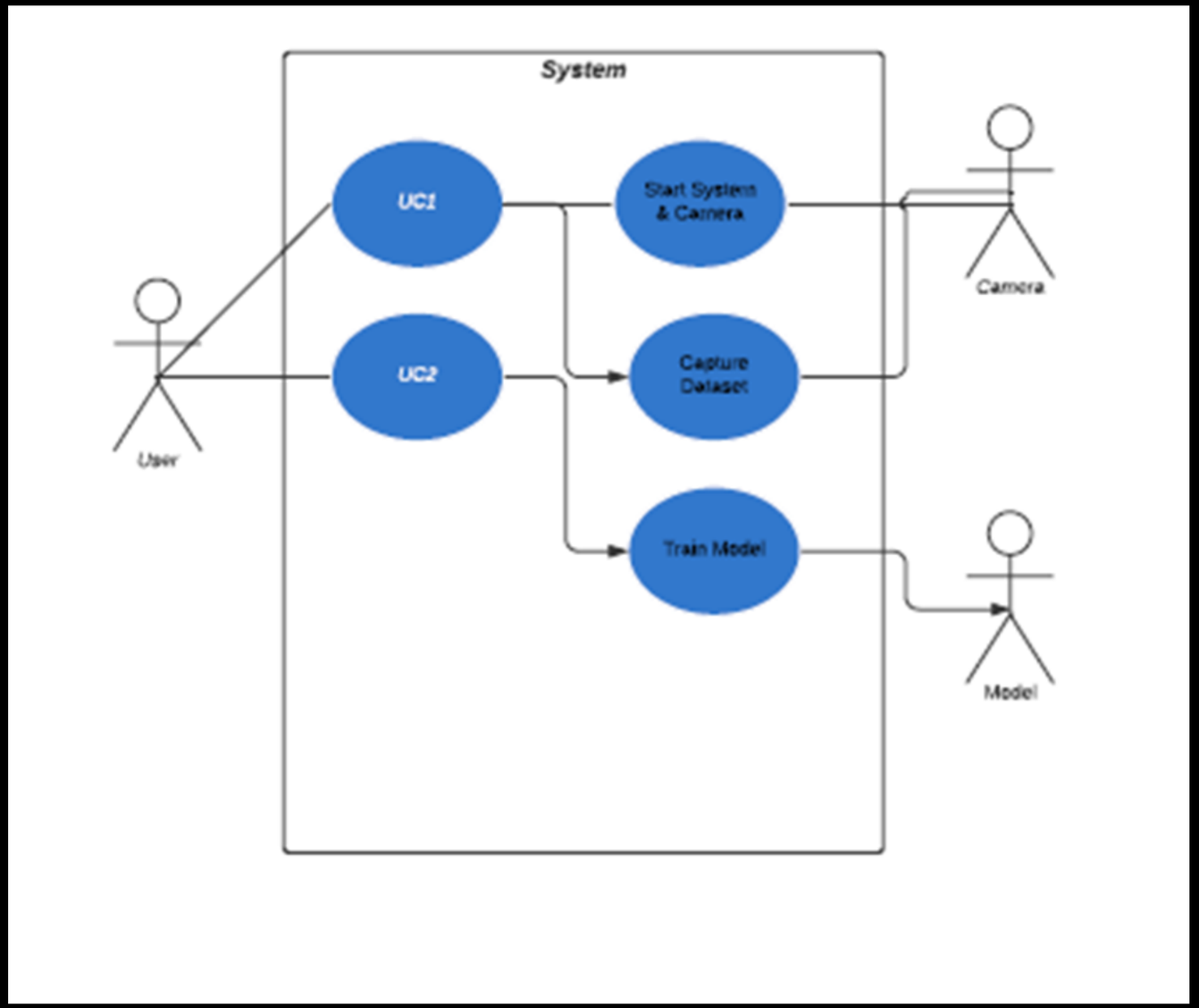
Actor	Actor's Goal (what the actor intends to accomplish)	Use case Name
User	To use the camera on which the program is running	Camera (UC-1)
User	To use the microphone of the device	Microphone (UC-2)
User	To type text in a textbox for converting it to gesture language.	Text (UC-3)
User/Camera	To be able to show the gestures to the camera so that it reads the user's hand correctly	Gesture (UC-4)
User	To use the speaker of the device via the program	Speaker (UC -5)
User	To navigate between Self-training mode and gesture reading option	Navigate (UC-6)
User	To select the size of samples for training	Size (UC-7)

User	To listen the audio generated from sign language	Audio (UC-8)
System	To be able to scale the program with high response time and security	Scale (UC-9)
Camera	To capture the dataset properly	UC-1, UC-6, UC-7

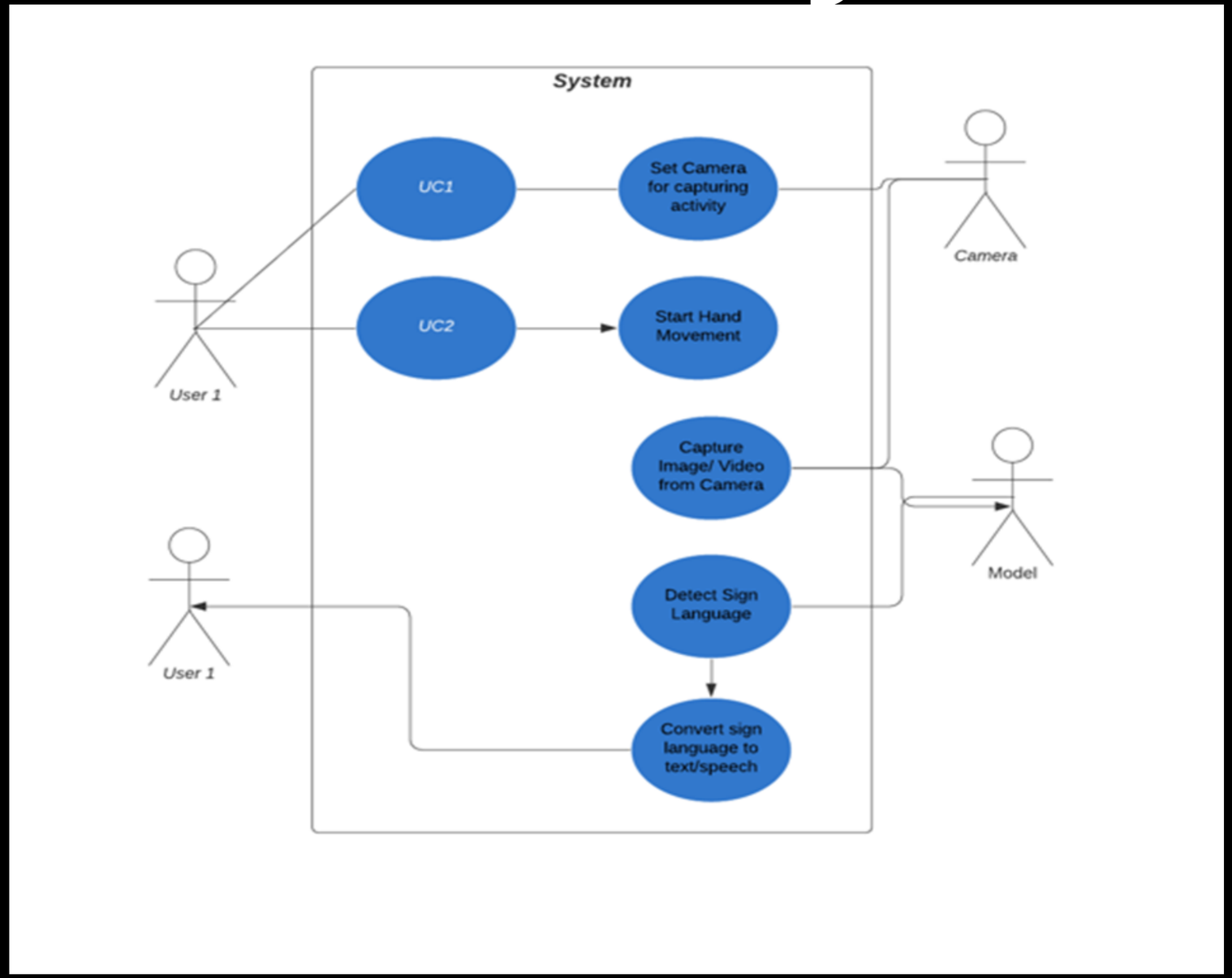
USE CASE DIAGRAMS



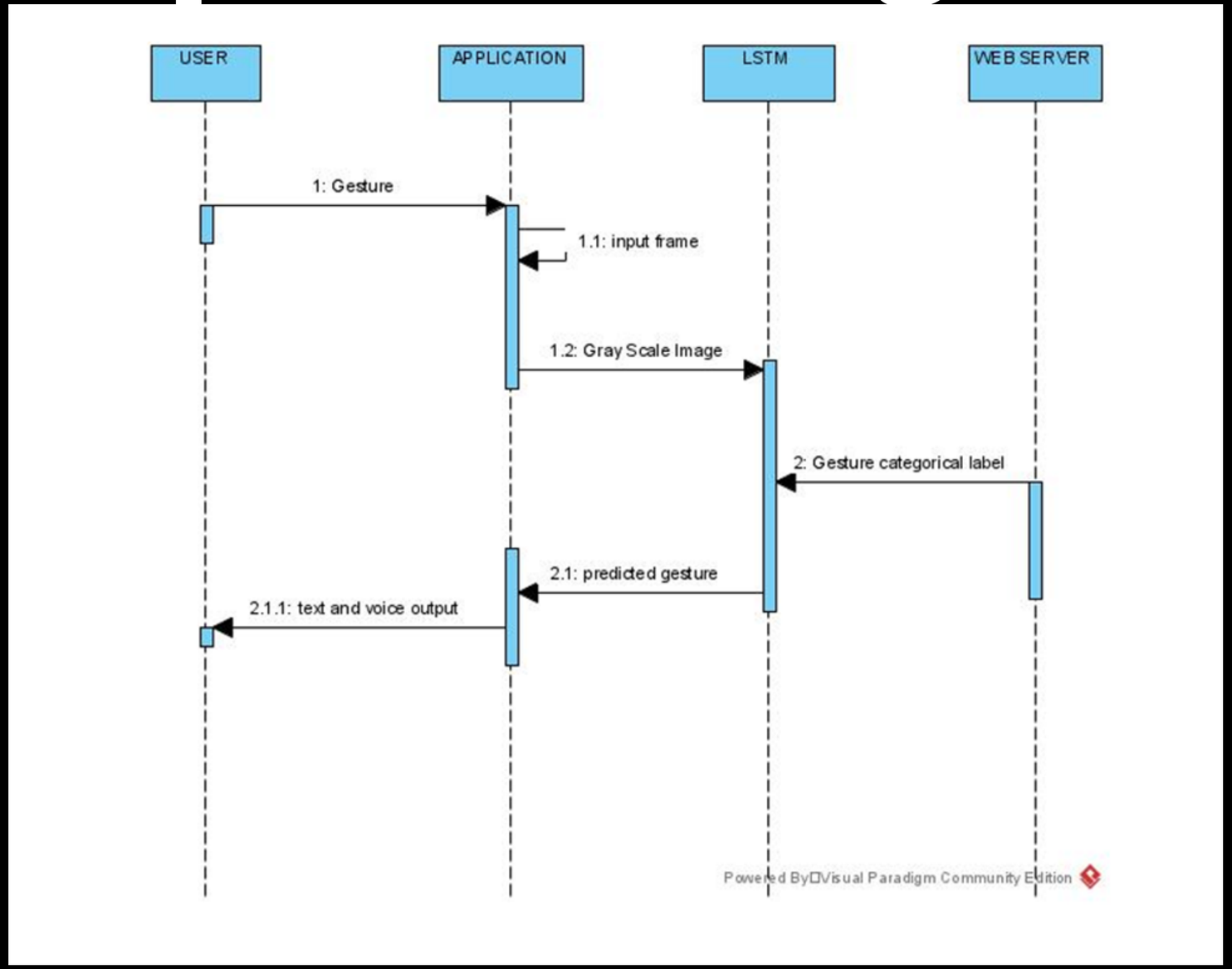
Data Modelling System



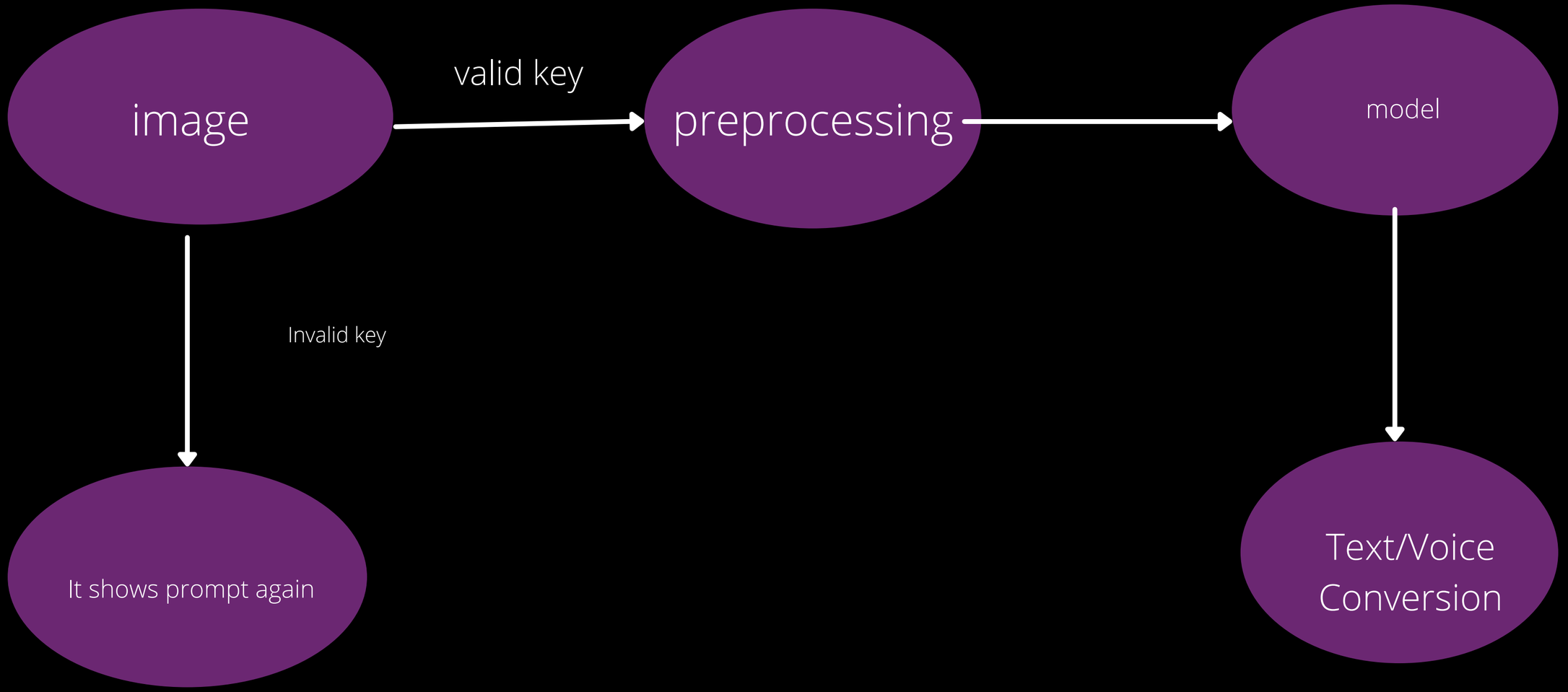
Gesture Subsystem



Sequence diagram



State diagram



USER INTERFACE:

1.We would start by using the users device camera to record the sign using hand gestures which would ultimately be the sign language.

2. We are creating a machine learning model that will read in the hand gestures and will try to match it with dataset of all the gestures we have trained the model with.

3. After the matching process the machine learning model predicts the gestures which the current user has used and with help of deep learning neural networks hidden layers it would detect the sign language and convert it into simple text which could be displayed on the screen for the user.

4. This will eventually help people who don't know sign language to understand what a deaf and a dumb person wants to communicate and on the other side it would also help the deaf and dumb person to communicate with people easily.

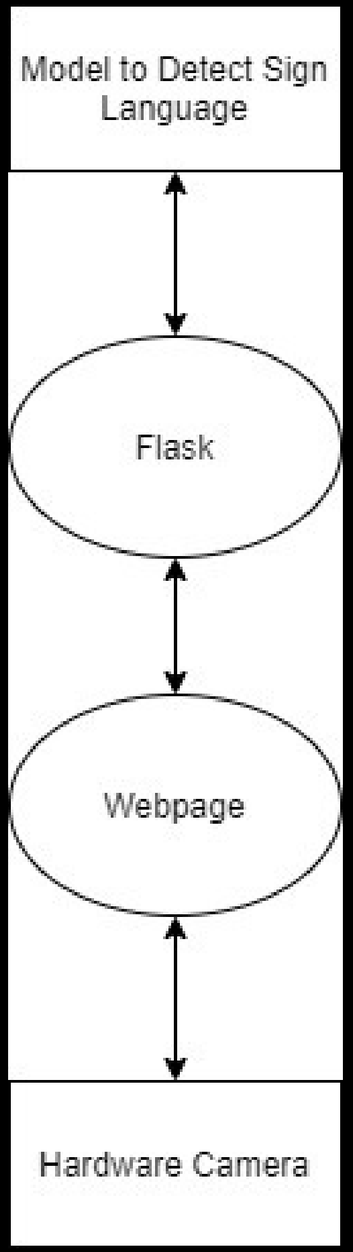
Integration

Machine learning model built in python is being used for detection of sign language

Flask Python is used for API's

HTML , CSS , Bootstrap is used for web user interface which would be helpful in:
Selecting between Hand Gesture to Speech
Selecting between Hand Gesture to Text

We would using device camera to capture video using web and pass it to flask to detect sign language



Demo of Application

Closer look at the concepts being used:

Machine Learning is very useful for a variety of real-life problems. It is commonly used for tasks such as classification, recognition, detection and predictions. Moreover, it is very efficient to automate processes that use data. The basic idea is to use data to produce a model capable of returning an output. This output may give a right answer with a new input or produce predictions towards the known data.

Deep Learning is part of a broader family of machine learning methods. It is based on the use of layers that process the input data, extracting features from them and producing a mathematical model

TECH STACK

Python.

FLASK

Scikit learn.

Pandas.

Tensor-Flow.

CV2.

Html -CSS.

Mediapipe

Pros and Cons of technology used

Python Pros:

- Extensive Libraries
- Simple and Easy
- Readable

Python Cons:

- Speed Limitations
- Weak in Mobile Computing and Browsers
- Design Restrictions

Mediapipe pros

- Fast
- Modular and Reusable
- Doesn’t Care about your Deployment Platform

Mediapipe Cons:

- Confusing

Changes with more knowledge

- We would have rather taken less frames per a gesture if we knew more about the model creation in depth at start, which we eventually did as it made our model more accurate and reliable
- We would really wanted to take the face keypoints but if the storing them was more easy and efficient.
- Learning more about ASL sign language would have been a helping hand in creating the dataset.

Summary:

Techniccal challenges:

- Problem of Computer vision (cv2) library with Mac OS
- Large file creation with more frames per gesture
- Higher accuracy of model with more training

Non-technical challenges:

- Less time for collaborative work

Overcoming the challenges

- We switched to Windows OS for using Computer vision library properly
- We tried hit and trail for finalizing the frames per gesture to be used
- We changes training data and testing data split to increase accuracy.
- We worked individually and kept team meeting short and merged all work on meetings

Lessons Learned

- Mediapipe can really change the future of how special people interact.
- Machine learning would be extensively effecting our lives in near future.
- Team work is more important than hardwork.

THANK YOU.