# Software Requirements Specification

**for**

# Sign Language CV App

**Version 2.1**

**Nurman Mahammadov**

**AHS**

**11 November 2024**

# Table of Contents

# Revision History

| Name | Date | Reason For Changes | Version |
|---|---|---|---|
| Nurman Mahammadov | 11/11/2024 | Created the original version with limited capability | 1.0 |
| Nurman Mahammadov | 04/13/2025 | Improved the application with improved front to back-end API and with a revised sign prediction model | 1.1 |
| Nurman Mahammadov | 04/25/2025 | Redesigned the application with a new clip capture and prediction process | 2.0 |
| Nurman Mahammadov | 04/26/2025 | Created a new front-end design with more features like the sign prediction | 2.1 |

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to give insight into our sign language app and to explain some of the background and the purpose of some of the components. It is intended to be read by both people with technical knowledge to get understanding of the application's backend in addition to regular consumers for learning about features of the application.

## 1.2 Document Conventions

In this document we use the word "application" to simultaneously refer to the downloaded application and the web application versions of our product unless it is specified when describing their differences.

## 1.3 Intended Audience and Reading Suggestions

The intended audience of this document is developers and application users without technical knowledge for getting insight into the application's features and its background logic for more advanced/ knowledgeable document readers. We suggest that the readers carefully look through the documentation and take notes on the application's structure in assisting the understanding of the application.

## 1.4 Project Scope

Our application is intended to convert a signed language into English text. It is also intended to teach sign language overall assisting communication between the intended audience of the

application. Our goal is to create an application that is easy to use for anyone without specific technical knowledge, in addition to being cheap to run by the user/ maintain by our team in terms of web hosting.

### 1.5 References

None as of right now.

# 2. Overall Description

### 2.1 Product Perspective

The project originated from the struggles of ASL users in communicating effectively with the non ASL using population which is drastically larger than the population that does know sign language.

### 2.2 Product Features

Our product will feature a live camera feed of signed language that is converted into written, on-screen English for the purpose of translating ASL hand gestures into words for the ease of communication between ASL users and people who don't know sign language. In addition, our application will include extra features such as English to other language translation, text to speech, and as-well a learning mode that is intended to teach users sign language based on pictures and practice trials of the user attempting the sign presented.

### 2.3 User Classes and Characteristics

The users for this application are people that suffer from being mute or deaf that wish to communicate with the population that is not familiar with ASL through sign language, as well as the other way around, users with no knowledge of ASL that wish to communicate with a person that is using ASL. The mute and deaf struggle to effectively communicate with just spoken English which forces them to communicate with sign language and hand gestures instead.

### 2.4 Operating Environment

Our application will be powered by python and Roboflow for the training image dataset. In addition we will utilize machine learning libraries like pandas, NumPy, matplotlib, TensorFlow, in addition to specific computer vision libraries such as OpenCV, and YOLO. For hardware we will employ a training computer with a i9-14900k and a GeForce RTX 4090. We will use REACT and Node JS for creating a user interface, and a hosting service that is to be determined.

### 2.5 Design and Implementation Constraints

Our application will utilize python, in application form our product will require a modern and semi-powerful computer for real-time calculations and model prediction. In web application form our product will only require a good internet connection and a device that can smoothly run a web browser such as google, bing, etc.

### 2.6 User Documentation

Our application will feature a tutorial on start-up that will run through the basic features of our application and how to access various sections/parts of the application. There will also be a help tab in our application that will redisplay the tutorial and include additional information for increased understanding.

### 2.7 Assumptions and Dependencies

It is assumed that the user of the application has access to the internet and a medium-high resolution camera for the purpose of real time sign language translation. As well it is assumed that the user has access to a mobile device or a computer with adequate specifications.

## 3. System Features

### 3.1 Functional Requirements

The software must be able to capture a series of images and then track the hand positions in order to predict an ASL sign that is associated with that movement, in short, the software must be able to capture and convert a clip into an ASL sign. Additionally, the software must be able to store the previous signs to be able to form a complete sentence/paragraph.

## 4. External Interface Requirements

### 4.1 User Interfaces

The user must have a display such as a monitor or even a mobile phone to display the video stream and to display the translations and they will be able to interact through inputs in the program.

### 4.2 Hardware Interfaces

The user must own/have a camera to be able to capture the ASL signs for the program to be able to convert it into English, and they must own a display like mentioned in the earlier point.

### 4.3 Software Interfaces

The program will interface through a webpage on any web browser.

### 4.4 Communications Interfaces

There will be communication through text on the display and with buttons on the interface that the user can interact with to capture a clip, to reset the text box, and other features of the program.

# 5. Nonfunctional Requirements

## 5.1 Performance Requirements

The expected performance would be roughly 15-30 frames per second for the clip capture and less than 500 milliseconds per clip prediction.

## 5.2 Safety Requirements

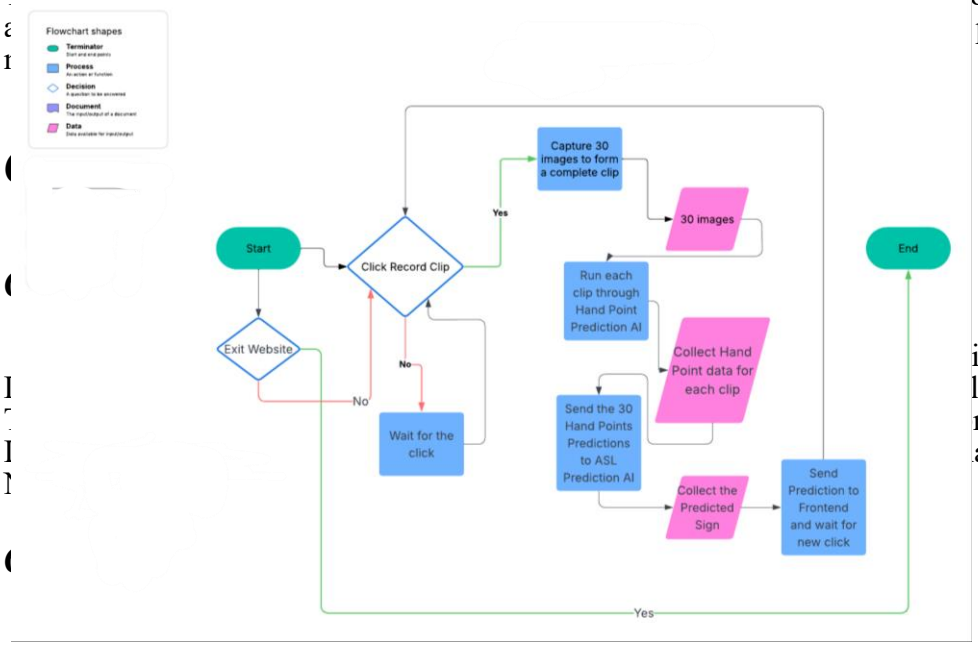
There aren't any safety requirements but there won't be any foul language or curse words in the possible sign predictions.

## 5.3 Security Requirements

There won't be any saving or collecting of video information / clip captures and they will only be temporarily processed then deleted, additionally there won't be any saving of other types of data like the predicted signs / predicted text.

## 5.4 Software Quality Attributes

The program should always be available for use and should be accessible from a variety platforms
prediction model for
ication, Pandas:
lotlib: graphing library,
rary, YOLO: You Only
aking web applications,




## 6.3 Process Flowcharts

My application collects a clip with 30 images upon the user clicking a “record clip” button, then that data is sent from my React JavaScript front end to a Python backend where the clip is processed with every photo being sent though a Hand Point detection AI made by google, then the data of the Hand Points of the 30 images is sent to my Custom Made AI that makes a prediction and sends it back to my front end for the user to see.

## 6.4 Other Supporting Information

N/A