

UNIVERSITY PARTNER



UNIVERSITY OF
WOLVERHAMPTON



HERALD
COLLEGE
KATHMANDU

Project and Professionalism (6CS007)

A2: FYP Final Report Template **American Sign Language Recognition (HandSpell)**

Student Id : 2049873
Student Name : Saman Tamrakar
Group : L6CG3
Supervisor : Mr. Sarjil Napit
Reader : Mr. Yogesh Bikram Shah
Cohort : 5

Submitted on : 27-04-2022

Declaration Sheet

Award Title: *BSc(Hons) Computer Science*

Declaration Sheet


(Presented in partial fulfillment of the assessment requirements for the above award.)

This work or any part thereof has not previously been presented in any form to the University or to any other institutional body whether for assessment or for other purposes. Save for any express acknowledgements, references and/or bibliographies cited in the work. I confirm that the intellectual content of the work is the result of my own efforts and of no other person.

It is acknowledged that the author of any project work shall own the copyright. However, by submitting such copyright work for assessment, the author grants to the University a perpetual royalty-free license to do all or any of those things referred to in section 16(I) of the Copyright Designs and Patents Act 1988. (viz. to copy work; to issue copies to the public; to perform or show or play the work in public; to broadcast the work or to make an adaptation of the work).

Student Name: Saman Tamrakar

Student ID Number: 2049873

Signature:  Date: ...2022-04-27.....

(Must include the unedited statement above. Sign and date)

Please use an electronic signature (scan and insert)

Abstract

Sign language is the primary means of communication for the deaf and hard-of-hearing people. The hearing-impaired community face different challenges while communicating with the hearing community. As a result, there is a communication gap between the two communities. This sign language recognition project helps to facilitate an easy means of communication using **American Sign Language**. The gestures are recorded through the system's webcam and the corresponding signs are predicted in real time. The objective of this project is to create a system which helps to bridge the communication gap between the hearing and hearing-impaired community. **Deep learning** and **computer vision-based** approach is used to predict the hand signs. In this project, a sign detector is created which detects 26 alphabets and predicts the respective American Sign Language manual alphabets.

The methodologies are broken down into stages: data collecting, pre-processing, segmentation, feature extraction, and classification, with the various algorithms for each stage detailed and their merits compared. We also go over the general issues and limitations of gesture recognition research, as well as those specific to sign language recognition.

Table of Contents

Abstract	3
1 Introduction	1
1.1 Project Briefing.....	1
1.1.1 Project Introduction.....	1
1.1.2 Implementation of AI.....	1
1.2 Aims	5
1.3 Objectives	5
1.4 Artefact.....	6
1.5 Academic Question	8
1.6 Scope and Limitation of the project	9
1.6.1 Scope	9
1.6.2 Limitation	9
2 Literature Review	10
2.1 Hand Gesture Recognition towards Enhancing Accessibility	10
2.2 Computer vision-based approaches for hand gesture recognition	11
2.2.1 Static hand gesture recognition	11
2.2.2 Dynamic hand gesture recognition	12
2.3 Technical approaches for data preprocessing of static image dataset.....	14
2.3.1 Image segmentation using edge segmentation	14
2.3.2 Skin Color Segmentation using Gaussian model and YIQ/YUV color space	15
2.3.3 Skin color segmentation using HSV color space and eliminating background	18
2.4 Feature extraction and gesture classification using Convolutional Neural Network	19

3	Project Methodology	22
3.1	Why Personal eXtreme Programming (PXP) was chosen	22
3.1.1	How PXP helped in the development of the project.....	23
3.1.2	Steps involved for the implementation of the methodology in this project .	23
4	Different Technology and Tools used for the project.....	26
4.1	Design Tools:	26
4.2	IDE:	26
4.3	GIT client:.....	26
4.4	Frontend framework:	27
4.5	Backend Framework	27
5	Artefact Designs.....	28
5.1	Artefact 1: User Management Sub-System (UMS).....	28
5.1.1	Software Requirements Specification (SRS)	28
5.1.2	Use Case Diagram	29
5.1.3	Activity Diagram.....	30
5.1.4	Class Diagram	30
5.1.5	Sequence Diagram.....	31
5.1.6	Testing.....	32
5.2	Artefact 2: Image Collection and Preprocessing (ICP)	34
5.2.1	Software Requirements Specification (SRS)	34
5.2.2	Use Case Diagram	35
5.2.3	Activity Diagram.....	36
5.3	Artefact 3: Sign Language Conversion (SLC)	37
5.3.1	Software Requirements Specification (SRS)	37
5.3.2	Use Case Diagram	39

5.3.3	Activity Diagram.....	40
5.3.4	Sequence Diagram	41
5.4	AI Artefact: Sign Classification Model (SCM)	42
5.4.1	Software Requirements Specification (SRS)	42
5.4.2	Use Case Diagram	43
5.4.3	Activity Diagram.....	44
5.4.4	Sequence Diagram.....	45
5.5	Data Collection.....	45
5.5.1	Development of Model.....	46
5.5.2	Optimization Evaluation	47
5.5.3	Algorithm performance and test data.....	49
6	Conclusion	50
7	Critical Evaluation of the Project	51
7.1	Findings and process	51
7.2	Planning, Management, and Quality of Sources	51
7.3	Self-reflection	52
8	Evidence of Project Management	53
8.1	Log Sheet.....	53
8.2	Gantt Chart.....	69
9	References.....	71

Table of Figures

Figure 1 Structure of Convolutional Neural Network (Bhavana, 2021)	3
Figure 2 Background subtraction in recorded dataset	4
Figure 3 Functional Decomposition Diagram	7
Figure 4 Dynamic hand gesture recognition flowchart (Zengeler, 2018)	13
Figure 5 Edge detection output on the left image and edge detection mask applied on the right image	15
Figure 6 Color distribution histogram for skin patches.....	16
Figure 7 Gaussian model fit	16
Figure 8 Skin color segmentation using Gaussian model fit on the left and YIQ and YUV color spaces on the right (Chen, 2017)	18
Figure 9 Hand region segmentation using HSV color space (Sofiane, 2018).....	19
Figure 10 Different layers of the CNN model for hand gesture recognition	21
Figure 11 Stages of PXP methodology	22
Figure 12 Design flow for user registration.....	24
Figure 13 Capturing of dataset.....	46
Figure 14 Labeled dataset images	46
Figure 15 Building CNN model.....	47
Figure 16 Optimizing the model	47
Figure 17 Model evaluation	48
Figure 18 Loss and accuracy of model.....	49
Figure 19 Predicted signs vs actual signs	49
Figure 20 Task list for Gantt Chart	69
Figure 21 Gantt Chart.....	70

1 Introduction

1.1 Project Briefing

1.1.1 Project Introduction

Hand signs and gestures are used by deaf and hard of hearing people in order to communicate and connect with the society. Ordinary people have trouble comprehending sign language. As a result, a system that identifies various signs and gestures and relays information to ordinary people is required. It connects people who are physically handicapped with others who are not. **The goal of this literature review is concerned with recognizing which alphabet of the American Sign Language is being signed from the video feed so that it can be converted into text.** We can recognize the indications and provide the appropriate text output using computer vision and neural networks. Many deaf and mute people would benefit enormously from such a translator since it would make it much easier for them to converse with others in everyday situations.

1.1.2 Implementation of AI

1.1.2.1 Addressing AI aspect

This is a **Computer Vision (CV)** based project which uses **Convolutional Neural Network (CNN)** to classify and then recognize hand gestures. The approach is based on vision, which is again divided into static hand gesture recognition and dynamic hand gesture recognition. Static is concerned with portrayal of motions in two dimensions, whereas dynamic is concerned with the capture of movements in real time. Gloves are difficult to wear and not practical for everyone despite their accuracy of over 90% (Hurroo & Walizad, 2020). This leaves us with vision-based approach from which static recognition is the preferred to obtain the highest amount of accuracy.

There are however certain critical issues that need to be resolved. To begin with, the vision-based hand sign recognition is primarily reliant on the sensitivity of the camera sensor and can be hampered by low picture quality. Secondly, without any human interactions, image processing algorithms are not as reliable and precise. Therefore, image segmentation is the most crucial part for hand sign recognition. (Chen & Li, 2017) proposed different image segmentation methods such as graph cut method, viewing image as electric circuit, and using new shape constraints for interactive image segmentation.

1.1.2.2 Learning approach

CNN is a supervised Deep Learning algorithm that is most employed in image recognition and computer vision. This project employs **supervised learning** as the hand images are classified. For the classification, a dense (or fully connected) layer is added which makes it a supervised approach.

1.1.2.3 Description of AI model

This project is materialized due to **Convolutional Neural Networks (CNN)**. Deep neural networks are one of the classes of deep neural networks, and CNN is one of them. CNN is used in a variety of fields, the majority of which include visual images. A CNN is made up of many neurons whose values, or weights, can be changed to achieve the desired result. These biases and weights can be learned. Non-linearity is caused by the dot products made between the neurons. The key distinction between regular neural networks and convolutional neural networks is CNN's assumption that all the inputs are explicit pictures. As a result, given our project relies around photographs of hand images, this will be the optimal method for training the model. As a result, CNN will be able to benefit from the architecture being bound in a meaningful way with images as input explicitly; a neuron layout is done in three dimensions: width, depth, and height. The volume required for activation is referred to as depth.

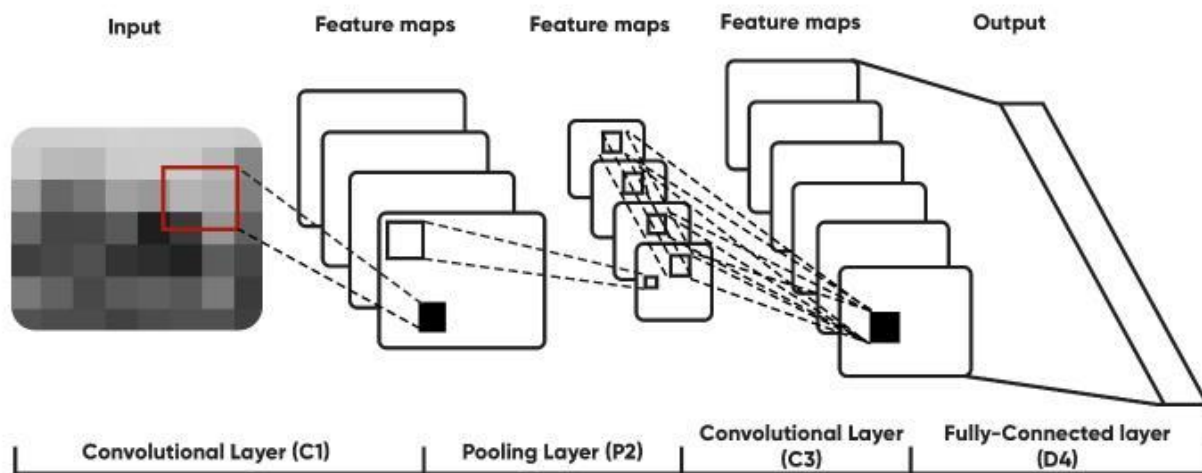


Figure 1 Structure of Convolutional Neural Network (Bhavana, 2021)

A convolutional network, as shown in the figure, consists of a number of layers, and the volume that exists is translated to another form using a function (differentiable). The pooling layer, convolutional layer, and fully connected layer are the layers that make up CNN architecture. A convolutional network design is created by stacking these in the correct order. Now that these layers have extracted features from the image that are unique to a property, the loss function must be minimized.

Background subtraction is the process of removing the background from a photograph. The foreground and background parts are separated using this technique. This is accomplished through the use of a mask that is produced according to the user's preferences. This method is used to detect through static cameras. Background subtraction is critical for object tracking, and it can be accomplished in a variety of methods.

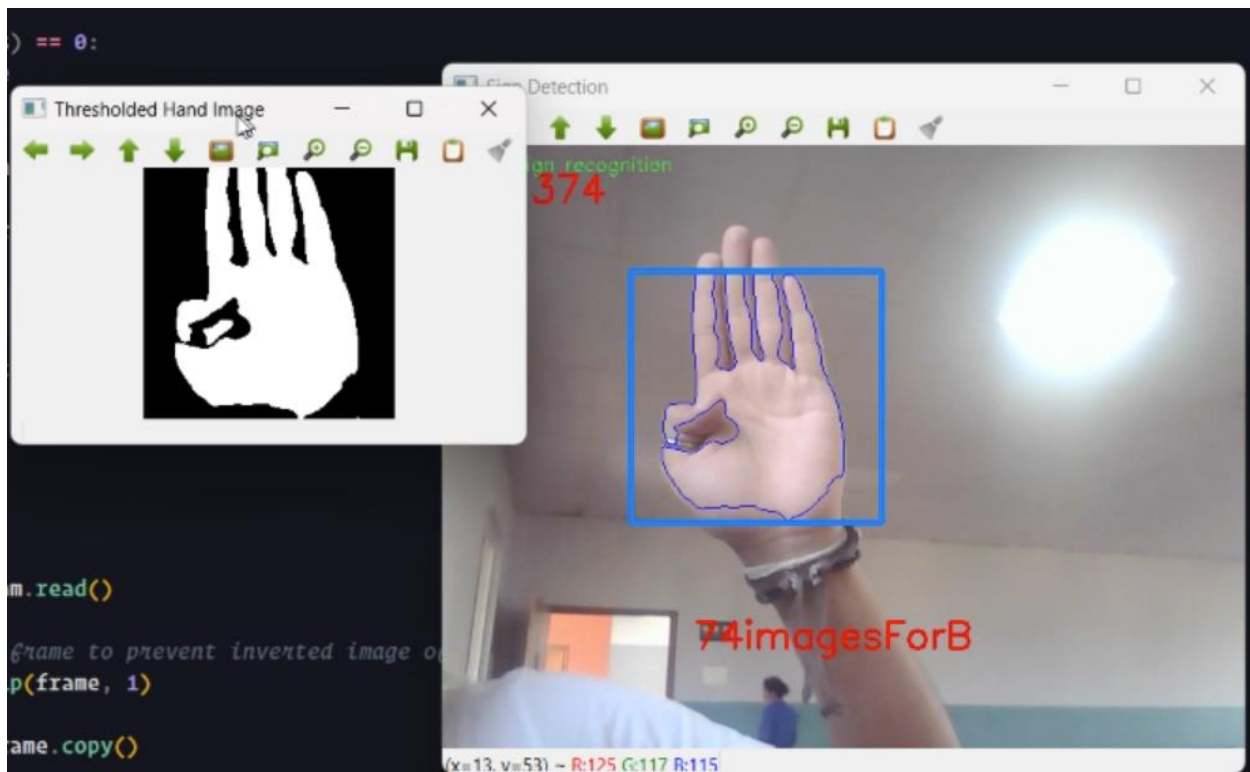


Figure 2 Background subtraction in recorded dataset

Proper hardware is necessary to train a deep neural network such as adequately fast processor, webcam, and storage. The dataset collection contains several photos of the American Sign Language alphabet, with a pixel size of 200x200 for each image.

To recognize the alphabets, we employ CNN (Convolutional Neural Network). Keras is used to describe what we do. The model, like any other CNN, starts with a few layers, such as **Conv2D** and **MaxPooling**, and then moves on to completely connected layers. The initial Conv2D layer captures the shape of the input image, while the last completely linked (dense) layer outputs 26 alphabets. To keep the training consistent, a Dropout is used after 2nd Conv2D layer. In the final layer, **Soft Max** is employed as the activation function, and the output is the probability for each letter.

1.2 Aims

- i. Provide the deaf and hard-of-hearing community access to communication by using sign languages instead of typing with the help of AI.
- ii. Enable people without disabilities to learn manual alphabets, consequently allowing them to communicate through sign language.
- iii. Provide access to communication with people who don't understand sign language.

1.3 Objectives

- i. Datasets for manual alphabets will be formulated which will be split into training data and testing data for the AI model.
- ii. Necessary image pre-processing will be applied to segment the hand from the background and only obtain the hand region.
- iii. A convolutional neural network (CNN) will be built on the created dataset to train the model.
- iv. Hand gestures will be captured in real time and the corresponding alphabets will be predicted.

1.4 Artefact

I. Webcam module for image acquisition

A laptop with an internal or external webcam will be required to capture the image of the hand in this artefact. The dataset will be created using the live cam feed using **OpenCV**. A **region of interest (ROI)**, which is a frame within we want to detect the hand for gestures is also created. The gestures trained in the dataset will include 29 gestures. Out of 29 gestures, 26 will be English alphabets and the rest three will be space, backspace, and nothing. These images are then resized, cropped, and labelled for training the dataset. The webcam will then be used in capturing the hand gestures of the user and enclose it within a bounding box.

II. Hand segmentation and gesture classification model

The image obtained from the webcam are converted into grayscale to make it easier to separate the hand region from the background. **Background subtraction** method is used in which a mask is created which removes everything except the hand image. Then, **binary thresholding** is used to make the hand gesture completely white while making the background black. This allows easier hand gesture detection for all types of skin colors and backgrounds.

The ability to identify and extract relevant elements from a picture is one of the most significant aspects of image processing. When images are recorded and saved as a dataset, they often take up a lot of space since they include a lot of data. Feature extraction assists us in solving this challenge by automatically decreasing the data after the key characteristics have been extracted. It also helps to preserve the classifier's accuracy while simultaneously reducing its complexity. Therefore, binary pixels of the images will be essential in this scenario. Gestures are detected from the acquired binary picture by comparing it to the numerous photos in the training dataset and returning the desired response.

III. Feature to learn manual alphabets and hand signs

In this sub-system, translation of text to American Sign Language is provided. In addition to deaf and dumb people being able to communicate through *HandSpell*, the hearing majority will also be able to communicate using this system. There will be two features in this artefact. One feature is simply to display all the sign language alphabets which can be used by the users as a reference for learning sign language.

The other feature is to provide text to sign language translation. Users can type words or sentences in the input field. The corresponding sign language will be displayed to the users right below the input field. This allows users who do not understand sign languages to be able to communicate to deaf and dumb people.

A **two-way method of communication** is possible through these features.

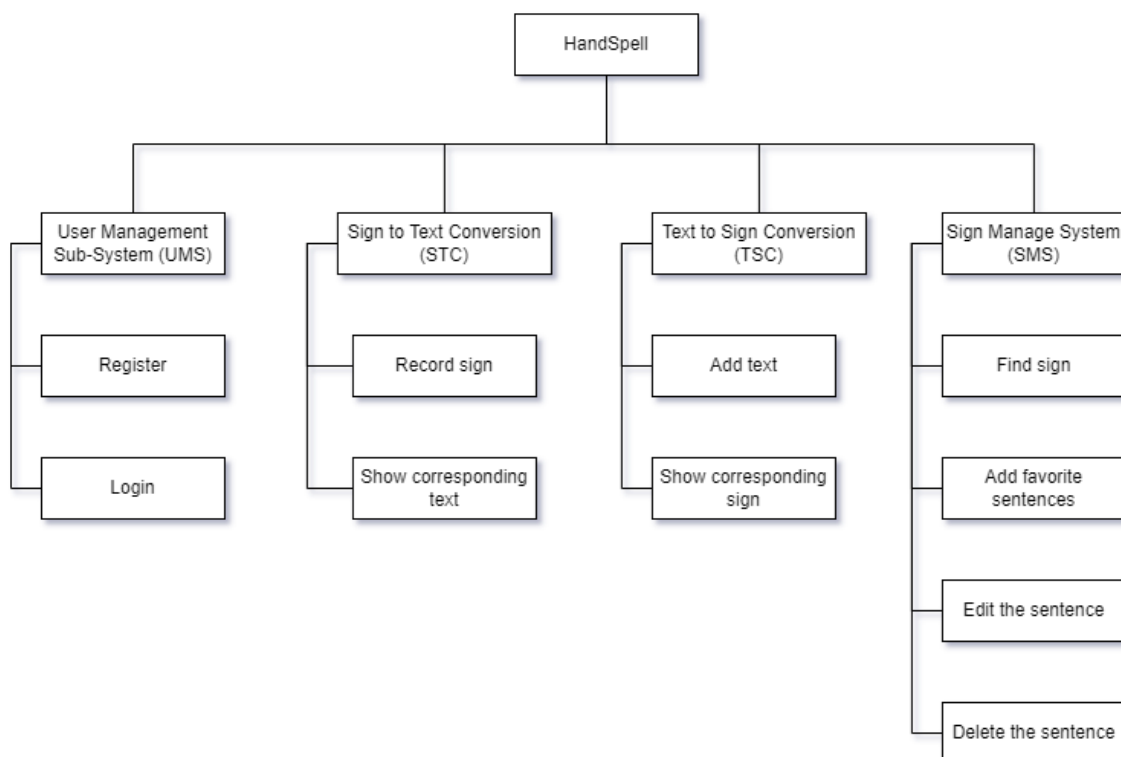


Figure 3 Functional Decomposition Diagram

1.5 Academic Question

How can artificial intelligence (AI) be used so that deaf and hard-of-hearing people can communicate using sign language rather than typing?

Sign language is the most common way of communication for deaf and hard-of-hearing people. However, the concept of sign language and manual alphabets (alphabets used by deaf represented by finger positions) is not common to hearing community. Deaf and hard-of-hearing people find it difficult to communicate with normally hearing people. This causes a disparity in the communication between the two communities. Deaf communities will thus be tied to only themselves and become reluctant to socialize with hearing people.

To mitigate such communication disparity, an application is required which not only allows deaf people to communicate through sign language but also encourage hearing people to learn the language through the application. The application should have a welcoming interface for both hearing and deaf people. It should provide all the necessary functionality for recognizing sign languages in real time as well as feature for learning the manual alphabets. All these features should be kept as simple as possible and easy for the user to navigate through.

Artificial Intelligence (AI) is a rapidly growing field, and the benefits of AI can be used for automating the recognition of hand gestures in real time. Convolutional Neural Networks (CNN) is the leading architecture for image recognition and classification task. The power of Neural Networks can be used for integrating in the proposed project.

1.6 Scope and Limitation of the project

1.6.1 Scope

HandSpell is a computer vision-based approach to detect hand gestures and predict the corresponding American Sign Language alphabets. The detection ranges from the 26 alphabets in the English language.

1.6.2 Limitation

The main limitation of *HandSpell* is that the detection is limited to only the manual alphabets which are still gestures. It cannot identify any gestures which incorporate any kind of movement or facial expressions.

2 Literature Review

2.1 Hand Gesture Recognition towards Enhancing Accessibility

Human Computer Interaction (HCI) has been a critical issue in the accessibility field, particularly in the context of a smart city setting, in which people must interact with artifacts scattered across a city. Such interactions require much more attention and ease of use for differently abled people like deaf and hard-of-hearing people. (Cardoso, 2016) states that, in the previous decade, HCI has seen substantial progress toward the ideal entirely natural user interface, such as the capacity to identify a variety of hand movements in real-time. A new concept has emerged known as Natural User Interface (NUI). These interfaces are seemingly invisible to the user and interact with them in such a way that feels natural to their behavior. Such interface can be achieved through the means of touch, gesture recognition, voice, or facial expression. One of the most popular solutions to achieve such interface is the use of electro-mechanical data gloves. These are highly accurate on recognizing hand gestures but come with the cumbersome requirement of carrying a large hardware and also comes at a heavy expense. Another commercially available solution presented by (Cardoso, 2016) is the MYO arm band which utilized electric impulses from the arm to remotely control devices wirelessly.

Some recent hardware attempts have been made to implant sensors capable of obtaining as much data on human body motions as feasible. Microsoft's Kinect Sensor is one example of this attempt. These technologies give a solution for body movement; but they lack detail, namely the hands, due to their small size in compared to the rest of the body. The SDKs provided by the manufacturers of these devices typically lack the critical hand movement information required for a successful hand gesture detection implementation. A hand gesture may be separated into dynamic and static components, which relate to the hand movement and the hand form, respectively. The dynamic and static components of hand gesture recognition techniques are extracted differently for each one, as explained in the following papers.

2.2 Computer vision-based approaches for hand gesture recognition

2.2.1 Static hand gesture recognition

Static gestures are movements that involve only the processing of a single picture at the classifier's input. The low complexity and simplicity of this technique is a benefit of this approach. For categorization and identification, many features are required. The study by (Islam & Hossain, 2019) focuses on static hand gesture identification by developing a CNN model that can evaluate vast amounts of visual data and detect static hand movements. Traditional pattern recognition techniques are unable to analyze natural data in its raw state. As a result, extracting characteristics from raw data takes a lot of time and effort, and it is not automated. CNNs are a type of deep learning neural network that can extract features on the fly and classify data using fully connected layers. These phases are combined in CNN to minimize memory requirements and computational complexity, resulting in improved speed. It can also comprehend the image's complicated and non-linear connections. As a result, the problem will be solved using a CNN-based technique. Other goals from the paper by (Islam & Hossain, 2019) include investigating existing gesture detection systems and assessing the impact of data augmentation in deep learning.

Dynamic gestures require a more complex approach to image sequence processing and gesture recognition. In the (Islam & Hossain, 2019), we find different cognitive methods based on supervised and unsupervised learning. Convolutional Neural Network (CNN), Support Vector Machine (SVM), Nearest Neighbor method, graph, distributed local linear embedding and others.

(Islam & Hossain, 2019) uses two image-based 24 gestures, some segmentation techniques, and the use of a convolutional neural network (CNN) for classification. As a consequence, the article demonstrated that outstanding results may be obtained with static gesture categorization utilizing a basic architecture of convolutional neural networks employing the suggested technique. We compared the proposed architecture with other existing networks in the literature and other gesture recognition methods.

There were however other limitations on static hand gesture recognition mentioned below:

- Errors and poor accuracy while recognizing signs which require movement such as the letters 'J' and 'Z'.
- Since each letter must be spelled individually, fingerspelling long sentences and paragraphs is difficult.
- The system is limited only to manual alphabets, and the use of facial expressions or other body languages cannot be recognized.

2.2.2 Dynamic hand gesture recognition

In dynamic hand gesture recognition, video classification is used to classify the American Sign Language (ASL) signs in a video dataset. The acquired videos from the dataset are broken down into frames of images, which are sent to the system for further analysis and interpretation. As stated by (Wankhede & Nagpal, 2019), dynamic gestures are often accepted in three different stages of movement preparation, lifting, and pulling. Among hand and arm gesture recognition, and body gesture recognition, the case of the former is studied more in depth in the research paper. Hand and arm recognition employs visual-based approaches to recognize hand motions, which may be divided into the categories: 3D model-based methods and appearance-based methods. 3D model-based approach uses several hand parameters, such as the palm, articulation point from the retrieved image, to create a 2D projection from a hand model in 3D. The display-based method measures the current pointer state by adjusting the 3D hand model with the highlights of the observed image. The volume model manages the 3D visual appearance of the human hand. There is an issue with the model proposed by (Wankhede & Nagpal, 2019) since it manages several parameters of the hand which result in a huge dimensionality and size when collected. The skeleton model solves this problem with the volume hand parameter model by limiting the placement of the parameters when showing the shape of the hand created from the 3D structure.

In presentation-based approach, also known as the view-based approach, the hand image is recreated with highlights removed from the visual appearance of the information image. It shows a hand that uses the power of 2D images and contrasts which are displayed separately from the information camera or video feed.

The model proposed by (Wankhede & Nagpal, 2019) is a Hidden Markov Model (HMM)-based technique that worked well for signals that were disconnected, but not for checking constant movement. In it, they showed a quick and basic calculation for hand signal confirmation in robot control applications. From the perspective of the 2D skeletal representation of the hand and the introductory histogram, it shows a technique which can be applied in recognizing hand signs in a restricted motion.

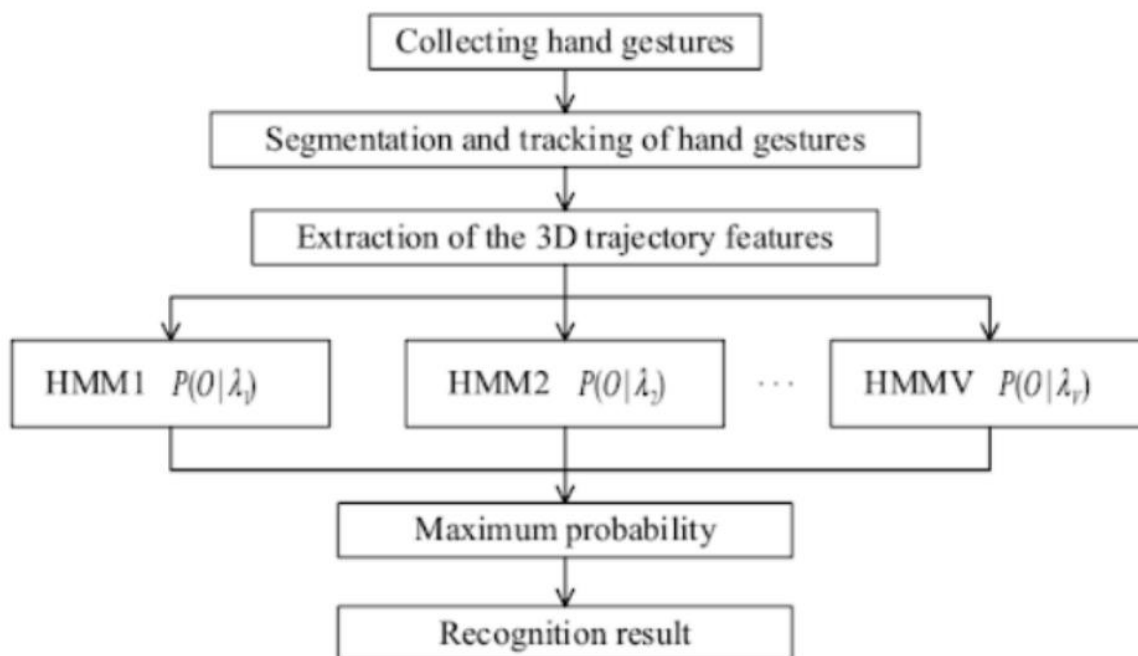


Figure 4 Dynamic hand gesture recognition flowchart (Zengeler, 2018)

According to (Zengeler, 2018), complicated backdrops, lighting, and variations in hand movement in space can all have a substantial influence on the accuracy of dynamic hand gesture detection. Based on a 3D time-of-flight camera (3DToF)., this research

offered an enhanced dynamic approach to hand gesture identification. For hand gesture segmentation, an adaptive segmentation technique that combines the frame difference approach and the depth threshold is utilized first. Next, use of depth data was used to develop dynamic hand gesture recognition technology based on 3D spatial-orbital functions and HMM. The inaccurate sample is then retrained to improve the identification of dynamic hand gestures. According to the model, the algorithm proposed in this work has a good identification rate for dynamic hand movements and is robust to a wide range of backdrops and lighting situations.

The limitations of dynamic hand gesture recognition are mentioned below:

- Training takes time and does not guarantee excellent outcomes.
- The hidden structure of HMMs, like that of multi-level neural networks, makes it impossible to examine their internal structure.
- Latency and speed problems arises due to hardware limitations such as camera quality and real time processing power.

2.3 Technical approaches for data preprocessing of static image dataset

2.3.1 Image segmentation using edge segmentation

One of the methods (Chen, 2017) has tried is the Canny Edge detector to find the relevant hand image in the field of the camera's view. Canny Edge detector is an edge detection algorithm which uses multi-stage algorithm to detect the edges in images. After the images were dilated, all remaining holes in the mask were filled to make a solid, continuous mask. After that, just the largest regions were taken to eliminate all of the backdrop clutter objects. This method simplifies the assumption that the largest items visible in segmentation are also the most interesting. This technique, however, did not always work since there were times when the largest object with identifiable edges was not the hand and it could have been a crowded background that hindered the result and so the image segmented badly. (Chen, 2017) chose to switch to a simpler, but maybe more accurate, color-based segmentation technique.

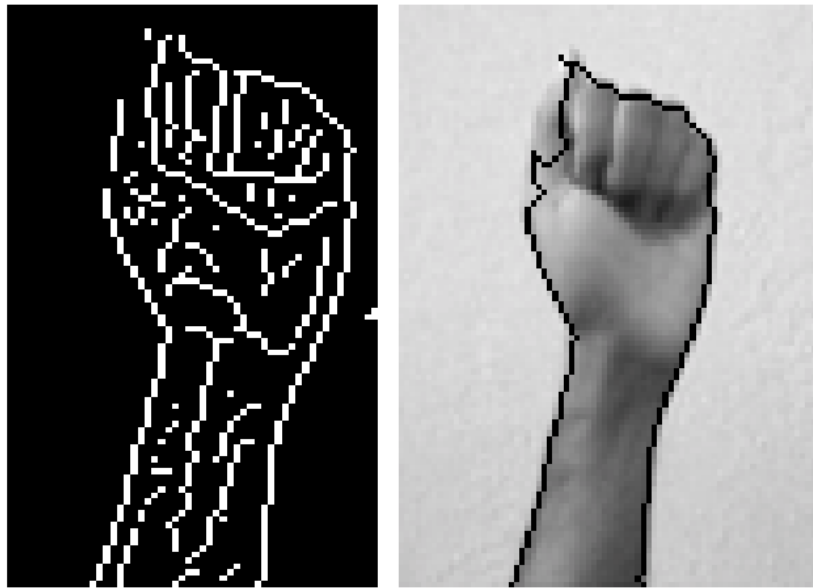


Figure 5 Edge detection output on the left image and edge detection mask applied on the right image

2.3.2 Skin Color Segmentation using Gaussian model and YIQ/YUV color space

In the case of skin color segmentation, images go through a series of steps in which the relevant image which is the hand is recognized using the RGB (Red Green Blue) color extraction technique. Using solely color information, (Chen, 2017) tested the two algorithms for skin segmentation. The initial method involves modeling the skin color using a 2D Gaussian curve and then estimating the chance of a particular color pixel being skin using this fitted Gaussian. First, (Chen, 2017) gathered skin patches from 40 random internet photographs. Each skin patch was a rectangular section of skin that was connected. Skin patches were obtained from people of various races so that our algorithm could properly forecast skin regions for a wide range of skin color. After that, the colors were adjusted as follows:

$$r = \frac{R}{R+G+B}, b = \frac{B}{R+G+B}$$

Here, only r and b values are considered as g is linearly dependent on the other two. Here, r and b are taken as the axes and the mean and covariance are calculated as shown below:

$$\text{Mean } m = E[x], \text{ where } x = [r, b]^T$$

$$\text{Covariance } C = E[(x - m)(x - m)^T]$$

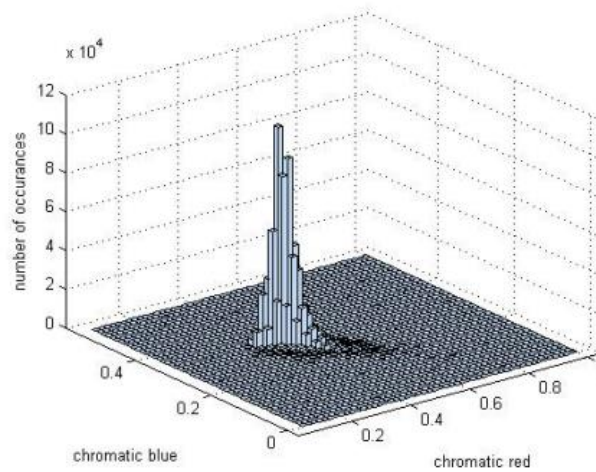


Figure 6 Color distribution histogram for skin patches

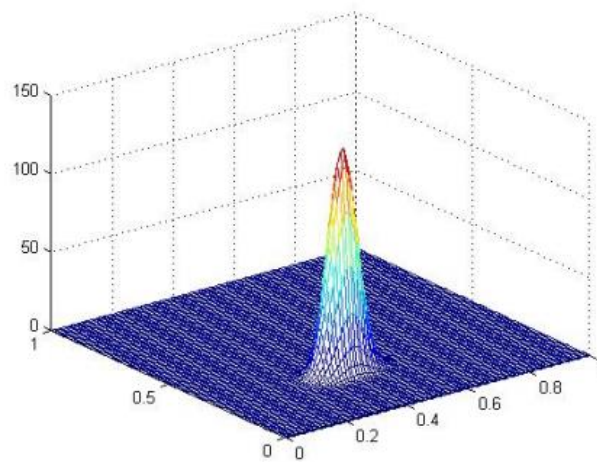


Figure 7 Gaussian model fit

By obtaining this skin color with Gaussian model fitted, the probability of skin for every pixel in each test image may be calculated. If the pixel has a chromatic pair value of (r, b), then the chance of it being a skin is given by:

$$Likelihood = e^{[-0.5(x-m)^T C^{-1}(x-m)]}, \text{ where} \\ x = [r, b]^T.$$

In the case of the second method proposed by (Chen, 2017), the image is converted from RGB to YIQ and YUV color spaces. YIQ is a colors pace used in Phase Alternating Line (PAL) while YUV is a color space used in National Television System Committee (NTSC). The theta parameter is then calculated as below:

$$\Theta = \tan^{-1}(V/U)$$

The U and V components represent yellow-blue and red-cyan differences in the YUV color space. Similarly, this is combined with the YIQ color space. If I is in the range of 30-100 and theta is in the range of 105-150, it is detected as skin.



Figure 8 Skin color segmentation using Gaussian model fit on the left and YIQ and YUV color spaces on the right (Chen, 2017)

The above figure clearly shows the Gaussian model performing poorly. The drawbacks of Gaussian model for skin color segmentation are:

- Inaccurate results in a dimly lighted setting
- Inconsistency when subjected to slightly noisy background

As a result, the YIQ and YUV color space approach was used as the Gaussian model could not be trained using enough variety of images. It is also a matter of fact that RGB cannot retain hue and saturation values as well as YIQ and YUV color spaces.

2.3.3 Skin color segmentation using HSV color space and eliminating background

The images go through a series of steps in which the backgrounds are recognized and removed using the HSV (Hue Saturation Value) color extraction technique. As stated above, RGB fails to contain the hue and saturation of an image. Therefore, (Hurroo & Walizad, 2020) states that the images need to be transformed into HSV color space. This model divides the colors of the image into three parts: hue, saturation, and brightness. HSV is a efficient way to improve image stability by separating brightness

and saturation. Hue elements are unaffected by any type of lighting, or shadow and can be used to remove the background. The trackbar with an H value of 0 - 179, an S value of 0 - 255, and a V value of 0 - 255 are used to recognize hand movements and convert the background to black. The hand gesture area is subjected to expansion and erosion techniques.

The first image is converted into grayscale. While this strategy may cause color loss in the skin gesture region, it will also make the system more resistant to changes in lighting or illumination. Non-black pixels in the modified image are binarized, while the rest remain intact, resulting in black. The hand gesture is divided into two parts: first, all of the image's associated components are removed, and then only the section that is relevant, in this example, the hand motion, is allowed. Blurring is applied, and the next stage is to apply morphological operations (dilation and erosion) to remove noise, or pixels that are not in use, in order to produce the best possible result.

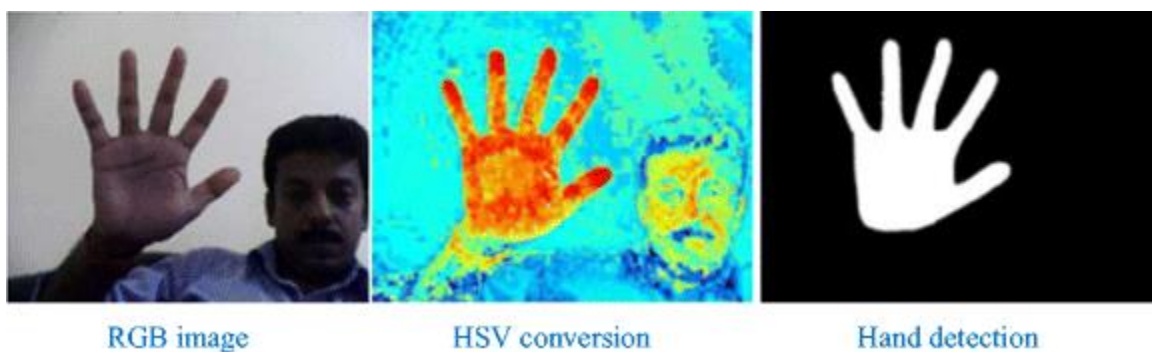


Figure 9 Hand region segmentation using HSV color space (Sofiane, 2018)

2.4 Feature extraction and gesture classification using Convolutional Neural Network

As stated by (Adithya & Rajesh, 2020), the model consists of an input layer, three convolution layers, a ReLu layer and a Maxpooling layer for feature extraction, a Softmax output layer, and a final fully connected output layer for gesture classification. In this study, the picture is resized to 100x100 pixels, and the dataset is divided into a training set and a test set before being fed into the CNN. RGB images of the hand

posture are sent to the input layer, which sends them to the feature extraction and classification layers. The convolutional layer is the true strength of CNN's deep learning. CNN uses the kernel's cascaded discrete convolutions and big picture, and an intermediate feature map to extract the most promising features for defining the shape of the hand. The following equation provides the convolution of the image. Here, f is the feature map and k is the kernel which is a square matrix.

$$f * k = \sum_{p,q=0}^{r-1} (f_{i+p,j+q})(k_{r-pr-q})$$

The proposed design by (Adithya & Rajesh, 2020) has three convolution layers. Eight filters on the first layer, 19x19 each, 16 filters on the second layer, 17x17 each, 32 filters on the third layer, 15x15 each. Padding is used on each layer of the convolution to keep the output the same size as the input. The convolutional output passes through a set of neurons using the ReLu activation function. The desaturation and non-linearity functions in the second equation below are used to replace the negative pooling layer value with zero. ReLu is a popular option as an activation function for deep neural networks due to its simple computation and lack of representation. The pooling layer that follows each ReLu layer reduces the size of the feature map without sacrificing important information. Maxpooling was utilized in this study because, due to its fast speed and enhanced convergence function, it outperforms the others in various pooling functions. After each convolution layer, the filter size (2,2) and step size (3,3) are used in the maxpooling operations to extract the maximum value from each local patch in the feature map.

$$y = \max(0, x)$$

The classification layer, which consists of a softmax layer and a fully connected layer/output layer, receives the most discriminative feature values retrieved by the multiple stacked structure made of convolution layers, ReLu layers, and pooling layers. The softmax layer has the same number of neurons as the output layer and uses a

multiclass sigmoid function to transform feature values from 0 to 1. In fact, the feature vectors derived from this layer accurately predict the likelihood of patterns appearing in the image. Based on the feature vector generated by the Softmax layer, the last fully connected layer will try to classify the input image into the correct gesture class. The number of neurons in this layer corresponds to the number of hand posture classes.

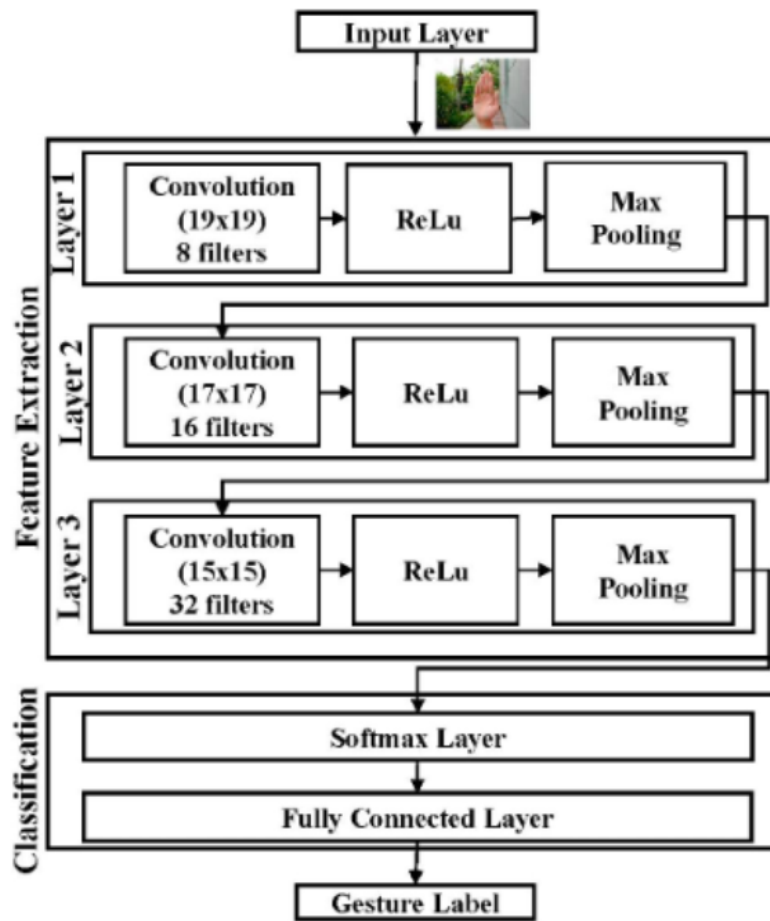


Figure 10 Different layers of the CNN model for hand gesture recognition

3 Project Methodology

3.1 Why Personal eXtreme Programming (PXP) was chosen

Personal eXtreme Programming (PXP), an agile methodology was used to develop this project. PXP was chosen as the project methodology as it is best suited for solo developers. SCRUM, Kanban, and Lean Development were also considered as the project methodologies for this project. These methodologies were discarded as they are more suited for a team and implementing this in a personal project means that these methodologies would not be implemented in its full manner. Waterfall model was also not selected as it is less flexible, takes a longer amount of time for project delivery, and the end product is less predictable.

PXP is a modified form of eXtreme Programming (XP) which focuses more for the project of a single person or a team with very small members. The values and principles are made more feasible in this project methodology. PXP methodology can be easily combined with MosCoW prioritization method which assists in the smooth development and completion of the project. The methodology begins with the design phase and ends with the retrospective stage, which is the end of the iteration stage.

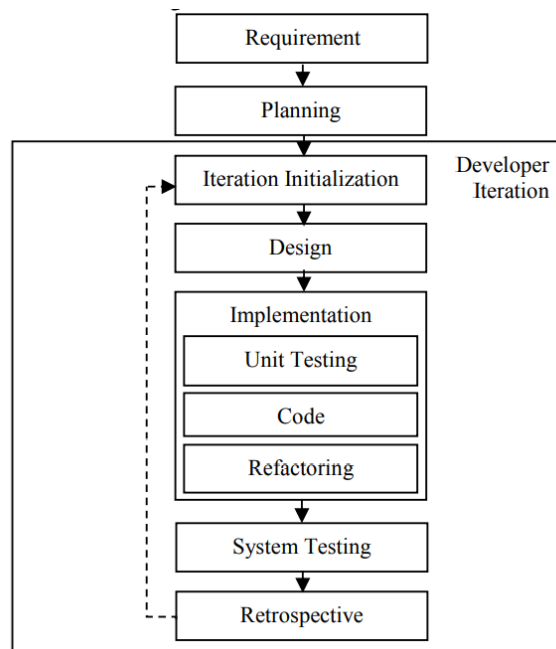


Figure 11 Stages of PXP methodology

3.1.1 How PXP helped in the development of the project

Personal extreme programming (PXP) can be used to quickly construct an application due to its ability to adapt to changing requirements. The MoSCoW technique was also helpful in planning and processing that focused on the most important requirements. Personal extreme programming was employed in this project to construct an application for the hand sign language detection in real time. PXP is able to meet the needs of clients with the development of ASL detection application to overcome the challenges encountered, based on the research that has been discussed.

3.1.2 Steps involved for the implementation of the methodology in this project

3.1.2.1 Requirements

In this stage, the requirements of the project were understood and the bare minimum artefacts that must be included in the project were pre-defined. The corresponding feedback of the client were also understood, and changes were made to the requirements.

3.1.2.2 Planning

The procedure on the user story derived from the requirements stage is described in the planning stage. I used Moscow to establish the projected time to complete the user narrative, the priority needs based on technical risk criteria, and business value, and to develop a release planning list.

Req. Code	Req. Desc	Use Case	MoSCoW Prioritization
ICP-F-1.0	The dataset is provided which contains adequate amount of hand images for training the model.	Hand image collection	Must Have
ICP-F-1.1	The images in the dataset should be rescaled to 200x200 pixels.	Image rescaling	Must Have

ICP-F-1.2	The images should be converted into grayscale to improve system's resiliency to changes in lighting.	Image grayscale	Must Have
-----------	--	-----------------	-----------

Further Content: [Artefact Designs including](#)

3.1.2.3 Iteration Initialization

The user requirements completed in the first iteration is to segment the hand from the background using background subtraction for data collection, according to the list of planned given in the above table. Such iteration steps were repeated for all the other artefacts.

3.1.2.4 Design

The following figure shows a design table for activity diagram and user stories that consists of a design flow for user registration system. In the next iteration, the work is to write the program code according to the designs that have been made.

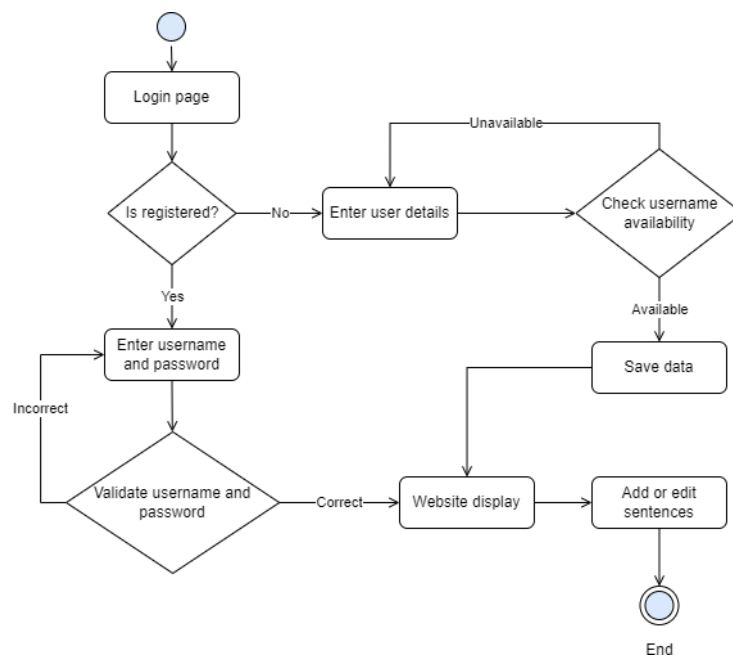


Figure 12 Design flow for user registration

3.1.2.5 Implementation

The program is implemented in three steps utilizing Test Driven Development (TDD): unit testing, code generation, and refactoring. All of the codes have passed unit testing. After each user story's program code passes unit testing, the process of completing the program code or code creation begins. In addition, the accuracy of the model and test findings are also included for each artefact.

3.1.2.6 System Test

This stage evaluates the outcomes of the features that have been implemented. The user acceptability test criteria are used in this test. It's based on a user's experience.

3.1.2.7 Retrospective

Based on the results of the first iteration implementation, verification is done at this stage to ascertain whether the realization time is equal to the initial estimation time.

Iteration 1				
User Stories	Priority	Story point	Estimation Time	Realization Time
Rescaling of hand images	Must Have	1	3	3

4 Different Technology and Tools used for the project

4.1 Design Tools:

Figma is a free UI designing and prototyping. This tool was used for wireframing and UI design as it is one of the biggest free UI/UX design tools with a great community support. It consists of thousands of plugins that can be used for UI design such as icon packs, ample variations of fonts, contrast checking tools, and so on. It is also a browser-based application so it can be easily accessed from any computer. It is easy to learn and provides easy tools for layouts and grids. Similarly, **Adobe illustrator** was used for designing the logo of *HandSpell*.

4.2 IDE:

Visual Studio Code (VS Code) is a lightweight but powerful source code editor which runs on almost any desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and has a rich ecosystem of extensions for other languages (such as C++, C#, Java, Python, PHP, Go) and runtimes. It provides a built-in terminal which makes migrations and running server convenient for a Django environment. It provides easy tools for changing the environment path to work separately on different projects. VS code is not technically an IDE, but it can be closely configured to function like one while retaining the responsiveness and fluidity. It also has a variety of themes and typefaces to work with which makes long coding sessions more bearable. It's downloadable Python and Django extensions makes it easily accessible for most of the developers.

4.3 GIT client:

GitHub was used as the version control system. GitHub's key feature is it allows users to sync their commit history. The code can be hosted on the Git client which is backed up and can be accessed from multiple computers. It provides the history of all the previous committed versions and the system can be easily rolled back to any of the previous versions if the code starts malfunctioning. It easily allows users to view the

changes that have been made after each commit and can figure out the bugs accordingly.

4.4 Frontend framework:

Django HTML templates were used for the frontend of the system. Django templates is provided by Django itself which is used for generating HTML dynamically. It contains both static and dynamic content of the required HTML. The Django template language is convenient to learn and implement as it ships with an API for using and rendering the templates. Normal HTML, CSS and JavaScript files can easily be used in Django templates. The main reason for using the simpler Django templates instead of ReactJS because this project is primarily a single page website. Most of the functionality is included in the homepage due to which reusing of models is not necessary and helps to minimize the complexity of the code.

4.5 Backend Framework

For the backend of the system, **Django web development framework** was used. It is a simple web development framework that simplifies the development process. This helped me in focusing more on the AI portion of the system rather than spending a lot of time on the backend development of the website. It runs on Python which is a high-level programming language and can be easily learned as well as highly flexible. Django also provides its own default administration as well as libraries to ease the implementation of user registration. It's ease of use and a rich community are also one of the major reasons for selecting Django.

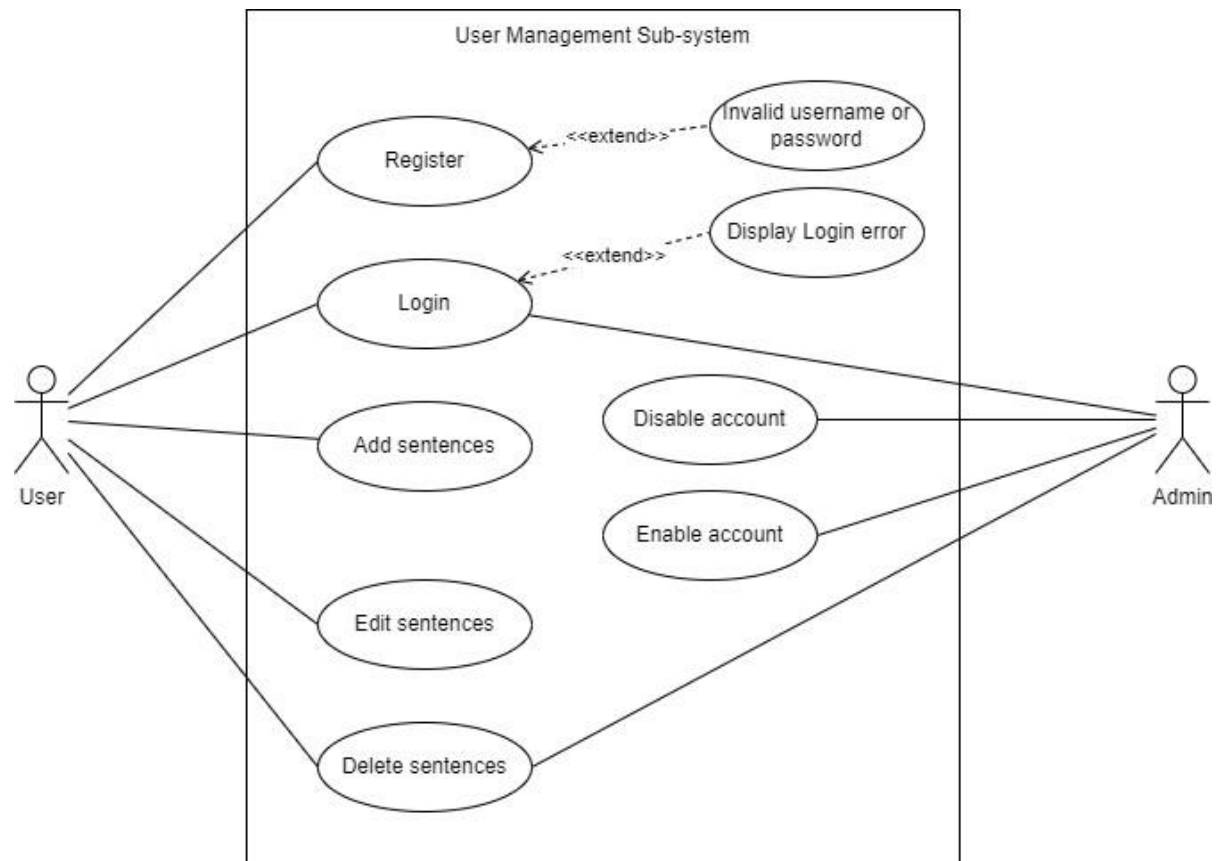
5 Artefact Designs

5.1 Artefact 1: User Management Sub-System (UMS)

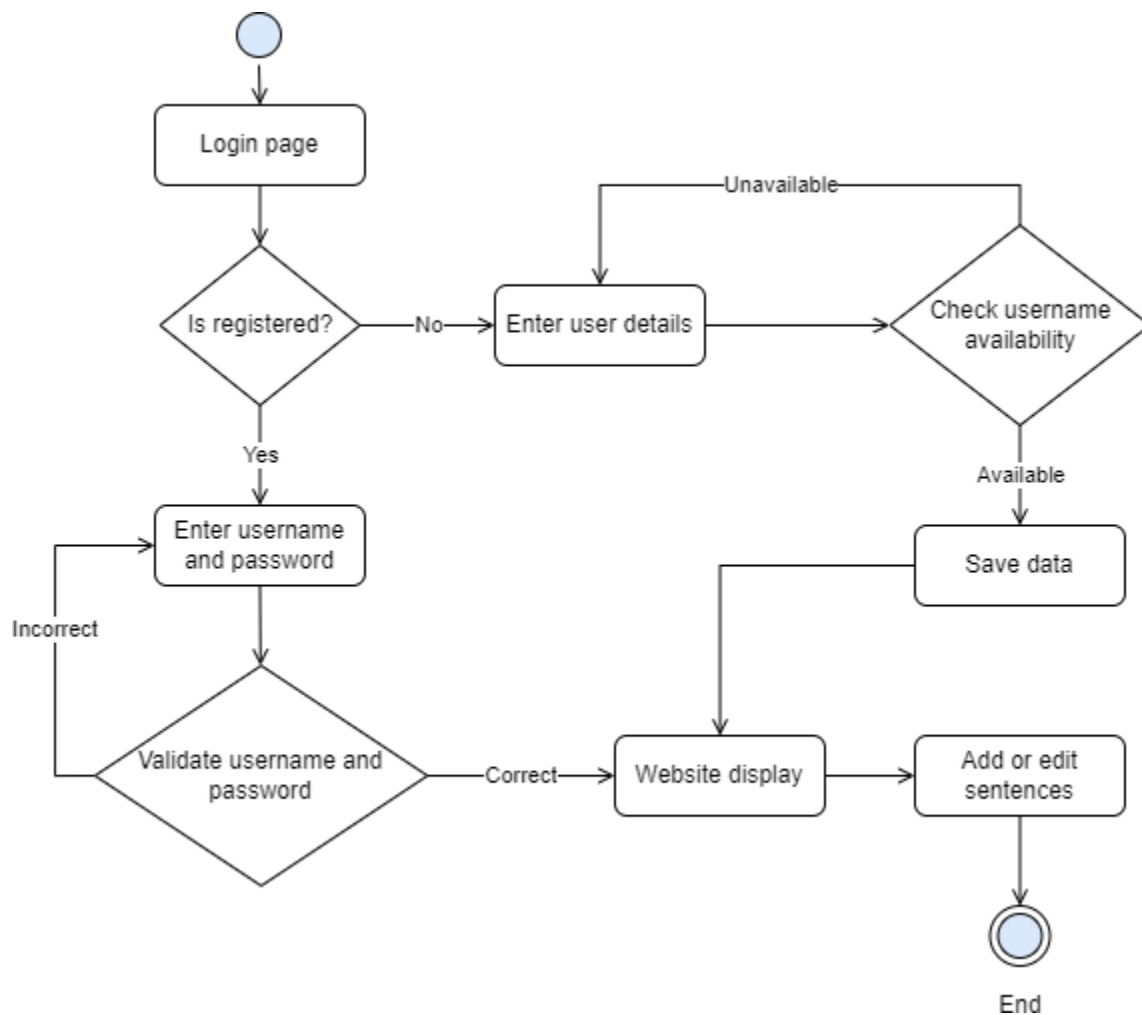
5.1.1 Software Requirements Specification (SRS)

Req. Code	Req. Desc	Use Case	MoSCoW Prioritization
UMS-F-1.0	This sub-system allows user to register and login.	Create User	Must Have
UMS-NF-1.1	The user's password must be a combination of letters, numbers, and special characters.	Password Criteria	Should Have
UMS-F-1.2	Users can register with only unique username or email address.	Manage distinct users	Must Have
UMS-NF-1.4	Users will also be able to register using Gmail, Facebook, or Twitter accounts.	Social media registration	Could Have
UMS-UR-1.5	There should be an option to show or hide password.	Password Protection	Should Have
UMS-F-2.0	The admin should be able to disable or re-enable a user's account.	Manage user account	Could Have
UMS-F-2.1	The admin should be able to delete the sentences saved by users if it is vulgar or inappropriate.	Lingual etiquette	Could Have
UMS-F-3.0	The users should be able to add their favorite sentences and save them.	Save sentences and words	Must Have
UMS-F-3.1	Users can also edit and delete the saved sentences or words.	Manage saved sentences	Could Have

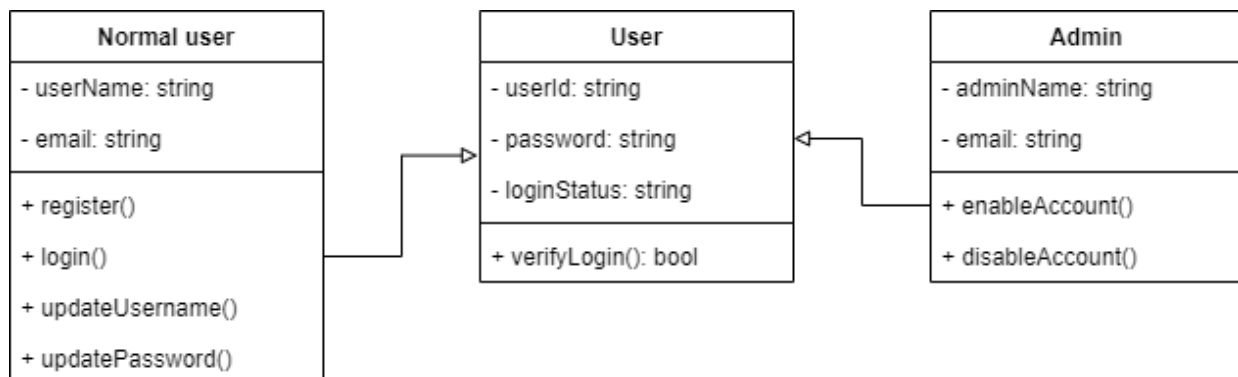
5.1.2 Use Case Diagram



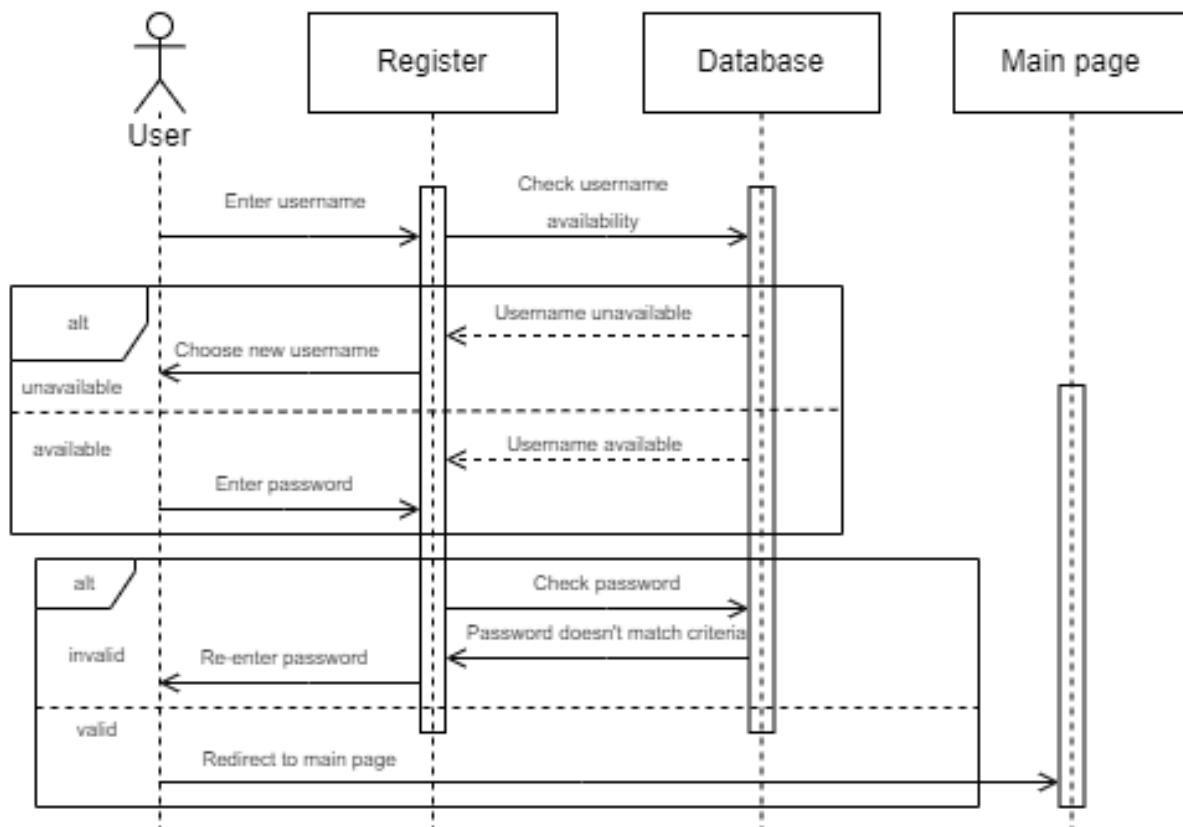
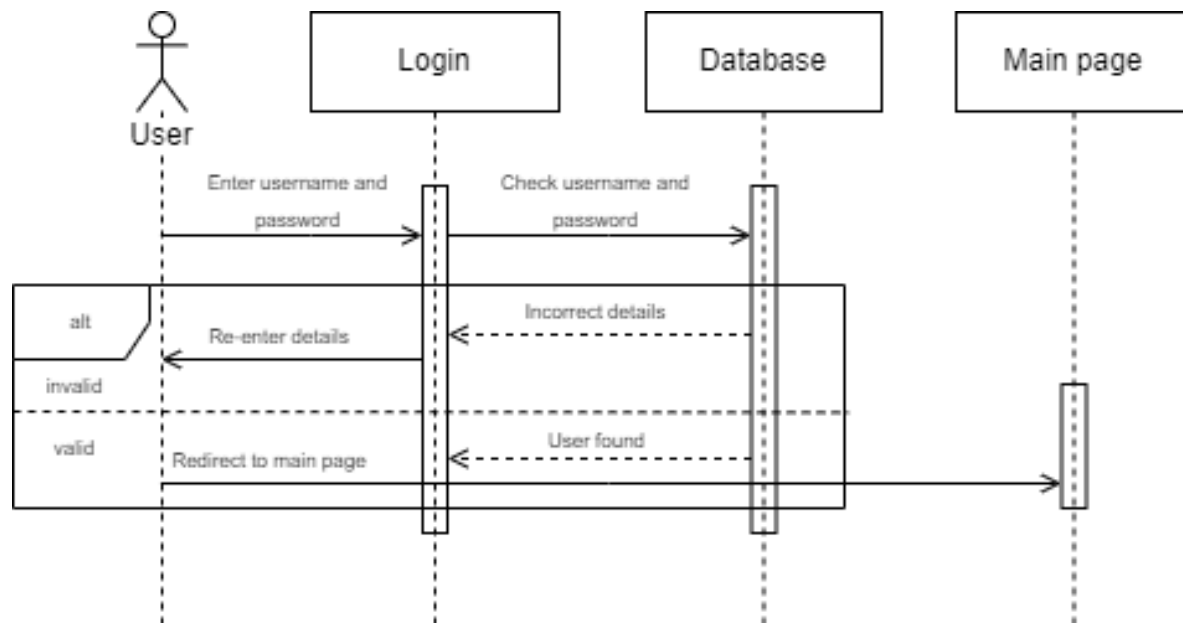
5.1.3 Activity Diagram



5.1.4 Class Diagram



5.1.5 Sequence Diagram



5.1.6 Testing

Signup

Serial No	Description	Detailed Steps	Expected Results	Actual Result
1	Enter signup page	Open http://localhost:3000/signup	Signup page is shown	Pass
2.	Incorrect email and valid other details	Enter invalid email credentials	Display error message that email is invalid	Pass
3.	Empty fields in signup	Keep one field empty	Display error message showing all fields is required	Pass
4.	Invalid password	Password must be at least 8 letters long	Show error message saying password is too short	Pass
5.	Different password and confirmation password	Enter different password and confirmation password	Show error message that the passwords do not match	Pass

6.	All details are valid	Enter correct details for all fields	Signup is successful	Pass
----	-----------------------	--------------------------------------	----------------------	------

Login

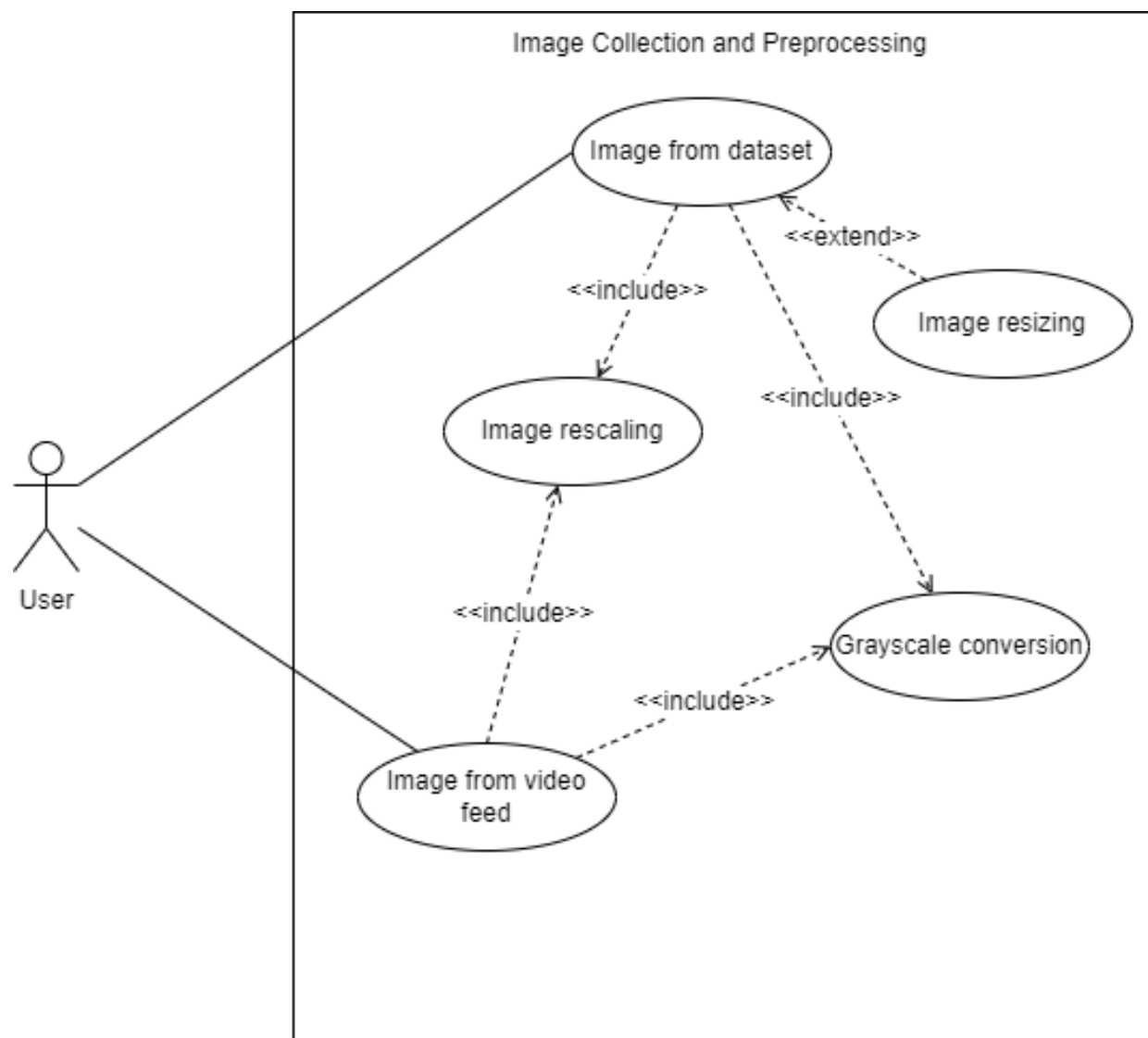
Serial No	Description	Detailed Steps	Expected Results	Actual Result
1.	Enter login page	Open http://localhost:3000/login	Login page is displayed	Pass
2.	Incorrect password	Enter wrong password	Show error message that password is incorrect	Pass
3.	Incorrect username	Enter invalid username	Show error message that user does not exist	Pass
5	Correct username and correct password	Enter correct username and correct password	Successful login	Pass

5.2 Artefact 2: Image Collection and Preprocessing (ICP)

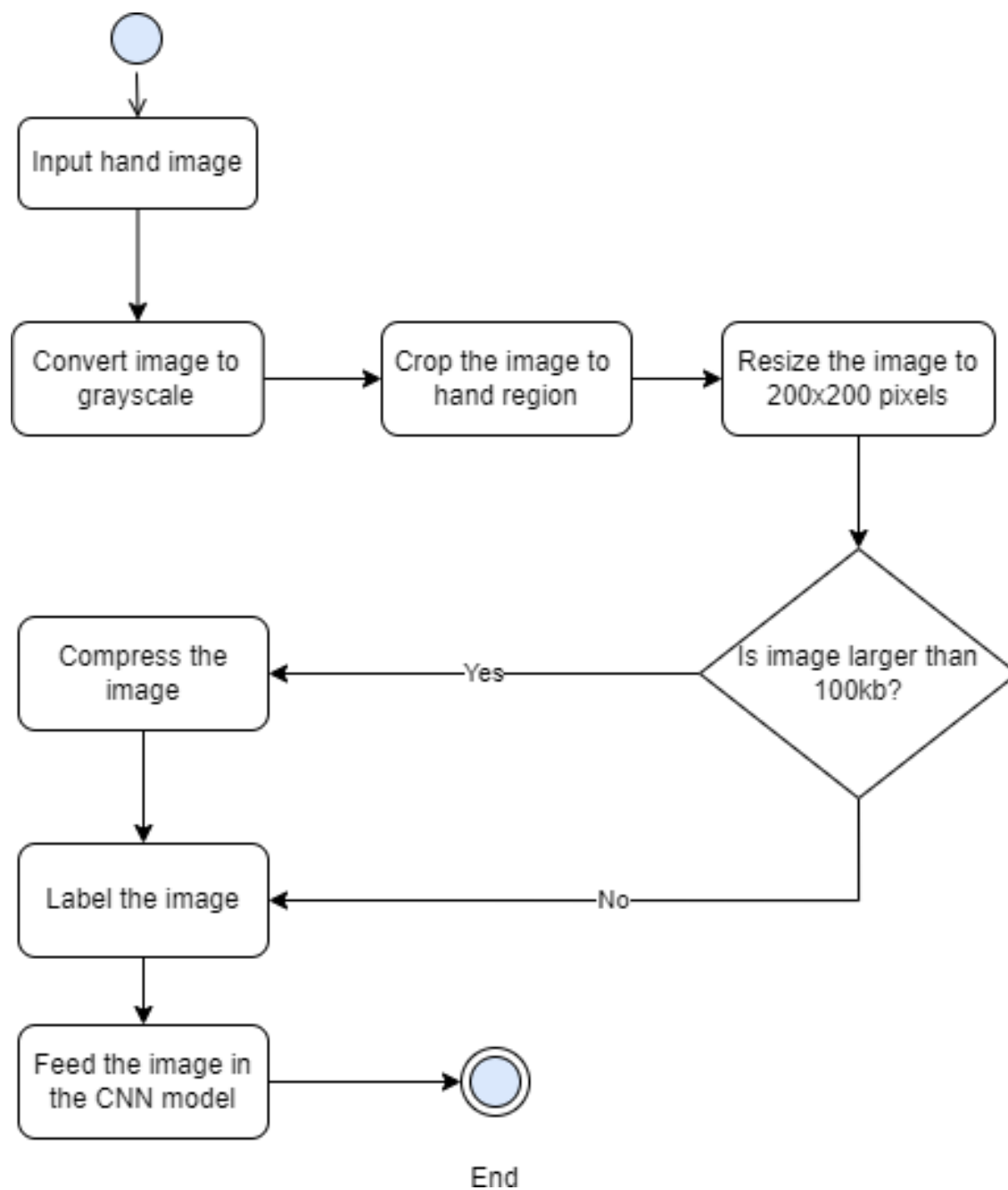
5.2.1 Software Requirements Specification (SRS)

Req. Code	Req. Desc	Use Case	MoSCoW Prioritization
ICP-F-1.0	The dataset is provided which contains adequate amount of hand images for training the model.	Hand image collection	Must Have
ICP-F-1.1	The images in the dataset should be rescaled to 200x200 pixels.	Image rescaling	Must Have
ICP-F-1.2	The images should be converted into grayscale to improve system's resiliency to changes in lighting.	Image grayscale	Must Have
ICP-NF-2.0	Images that are too large should be compressed to less than 100kb.	Image resizing	Should Have
ICP-F-3.0	The images collected should be saved in jpg format.	Image format	Should Have
ICP-NF-3.1	Images should be labelled along with their alphabet name and index (i.e., A1, A2, A7, C30, etc.)	Image labelling	Should Have
ICP-F-4.0	The image obtained from the live video feed should also be resized and rescaled.	Image scaling and resizing	Must Have

5.2.2 Use Case Diagram



5.2.3 Activity Diagram



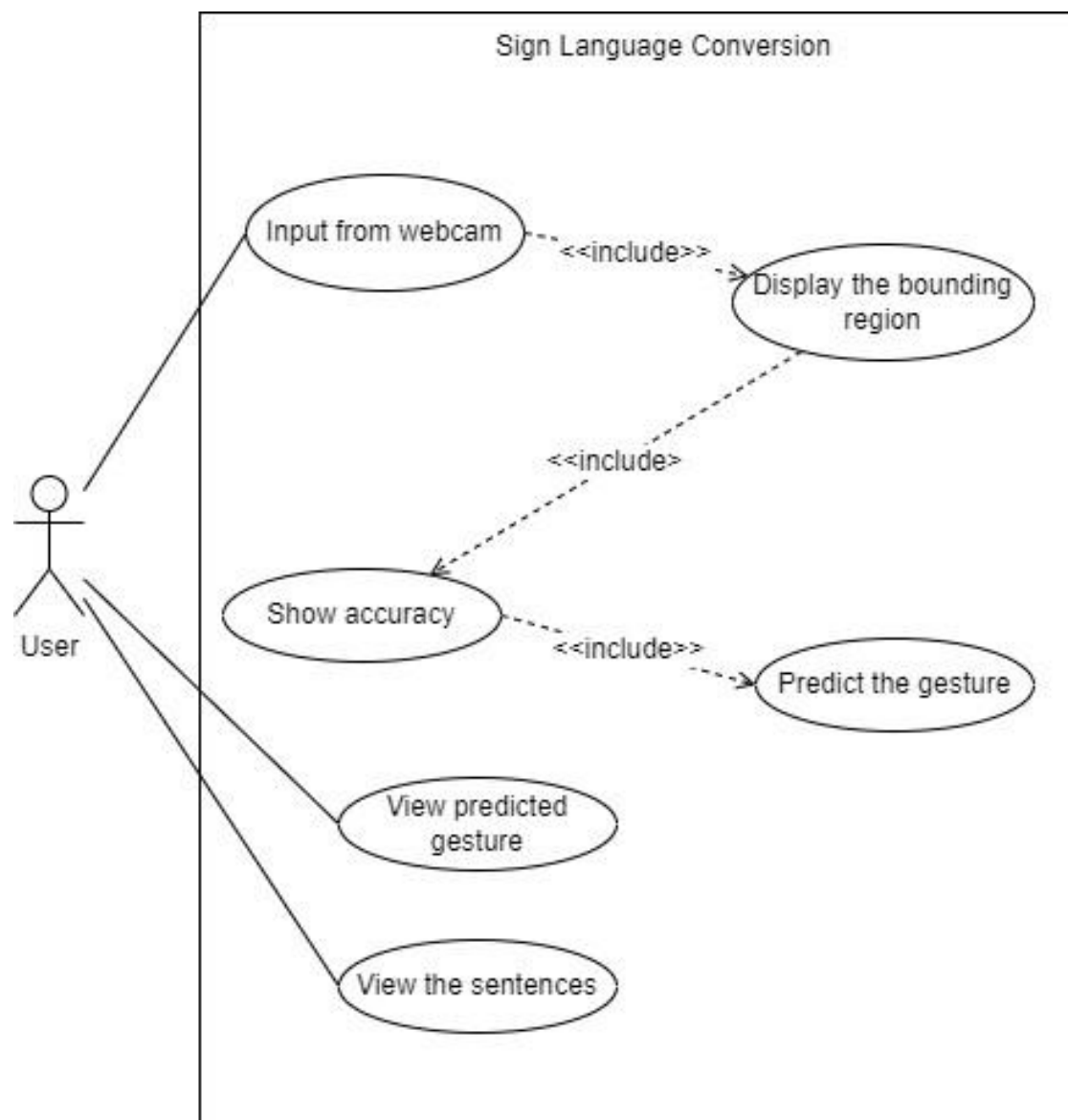
5.3 Artefact 3: Sign Language Conversion (SLC)

5.3.1 Software Requirements Specification (SRS)

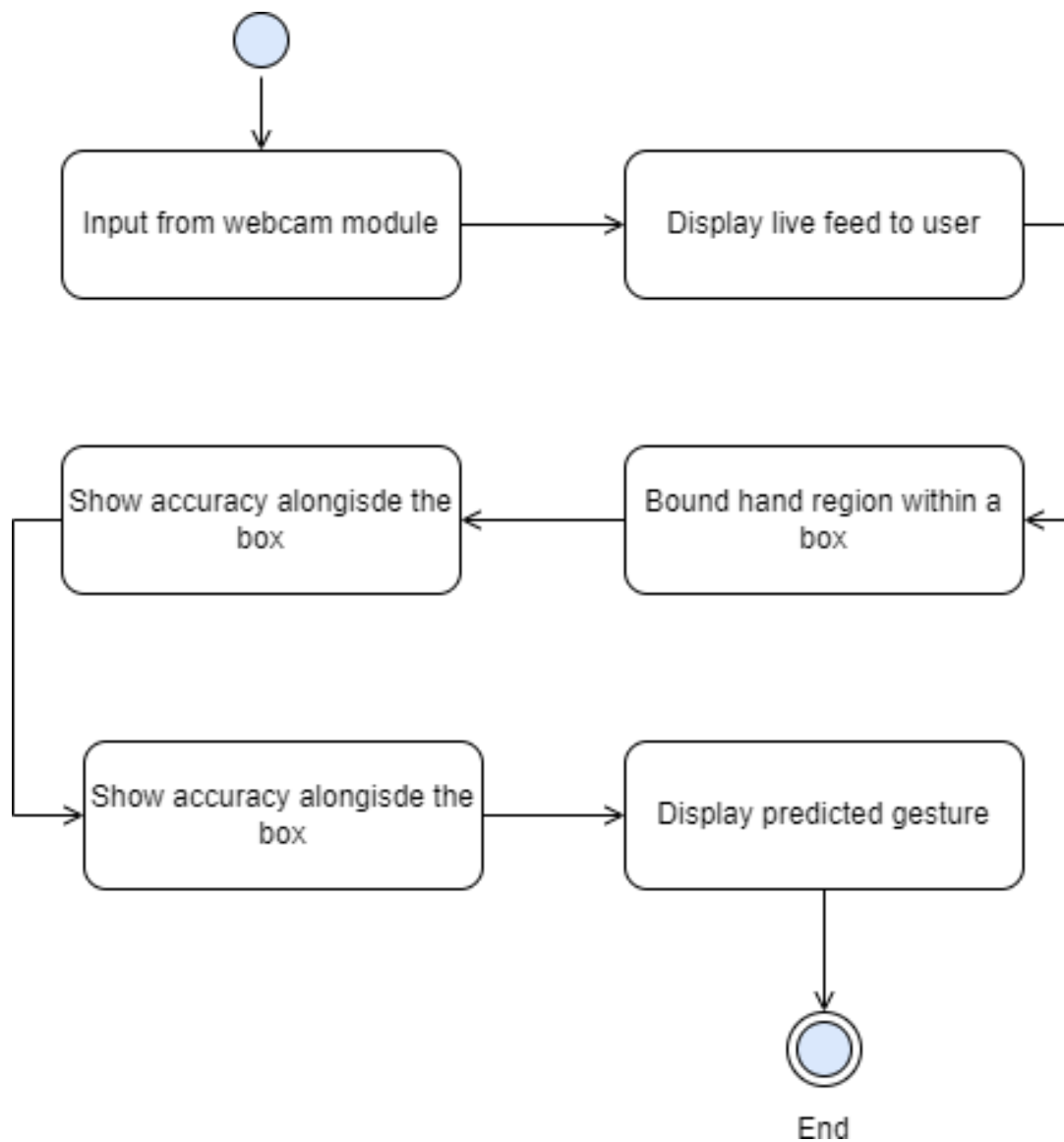
Req. Code	Req. Desc	Use Case	MoSCoW Prioritization
SLC-F-1.0	The sub-system should have a webcam module for input live video feed.	Input the live video.	Must Have
SLC-F-2.0	The sub-system should display the live video feed to the user.	Display the user's video.	Must Have
SLC-NF-2.1	The hand region (Region of interest) should be displayed distinctly inside a bounding box.	Display the hand region.	Must Have
SLC-NF-2.2	The video feed should also display the accuracy of the sign predicted alongside the bounding box.	Display the accuracy of gesture.	Could Have
SLC-UR-2.3	The user should be able to enable or disable the bounding box region.	Toggle the bounding box indicator.	Could Have
SLC-UR-2.4	The video should have an option to display a guideline where the placement of hand is most suitable.	Guideline for hand placement.	Could Have
SLC-F-3.0	The hand gesture is detected, and corresponding sign alphabets are displayed as words.	Display predicted sign alphabet	Must Have
SLC-NF-3.1	There should be a separate text box to display the words predicted as sentences.	Display the sentence	Should Have

Serial No	Description	Detailed Steps	Expected Results	Actual Result
1.	Accessing webcam	Enable the webcam module	The device's webcam should be turned on	Pass
2.	User's live video feed	Opening the webcam to view the user's video feed	Display user's live video feed	Pass
3.	Region of Interest detection	The region of interest should be correctly displayed on screen	ROI placement on live video feed	Pass
4.	Detecting hand sign	Showing hand gestures on live video feed for prediction	The corresponding sign letter should be displayed	Pass

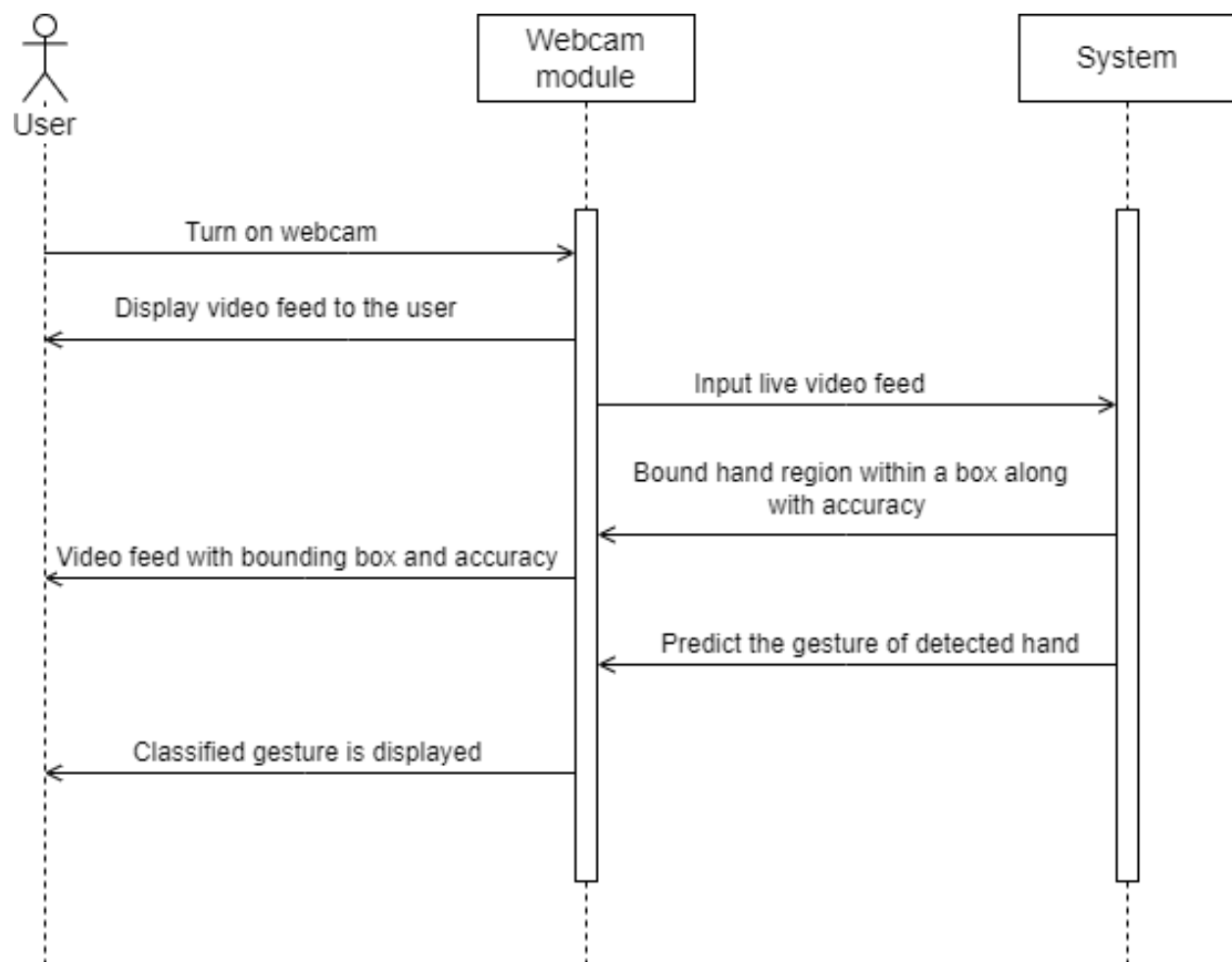
5.3.2 Use Case Diagram



5.3.3 Activity Diagram



5.3.4 Sequence Diagram

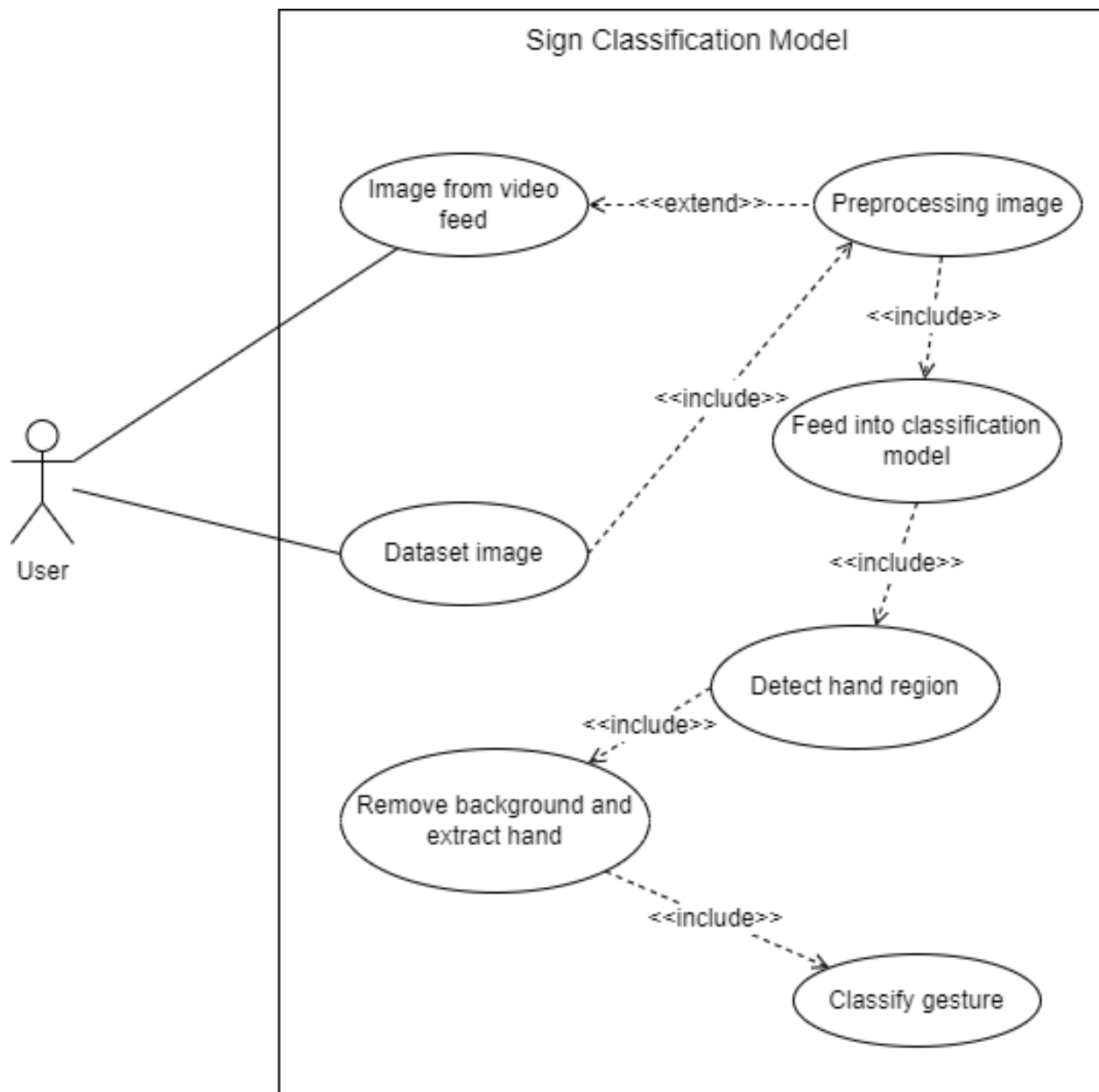


5.4 AI Artefact: Sign Classification Model (SCM)

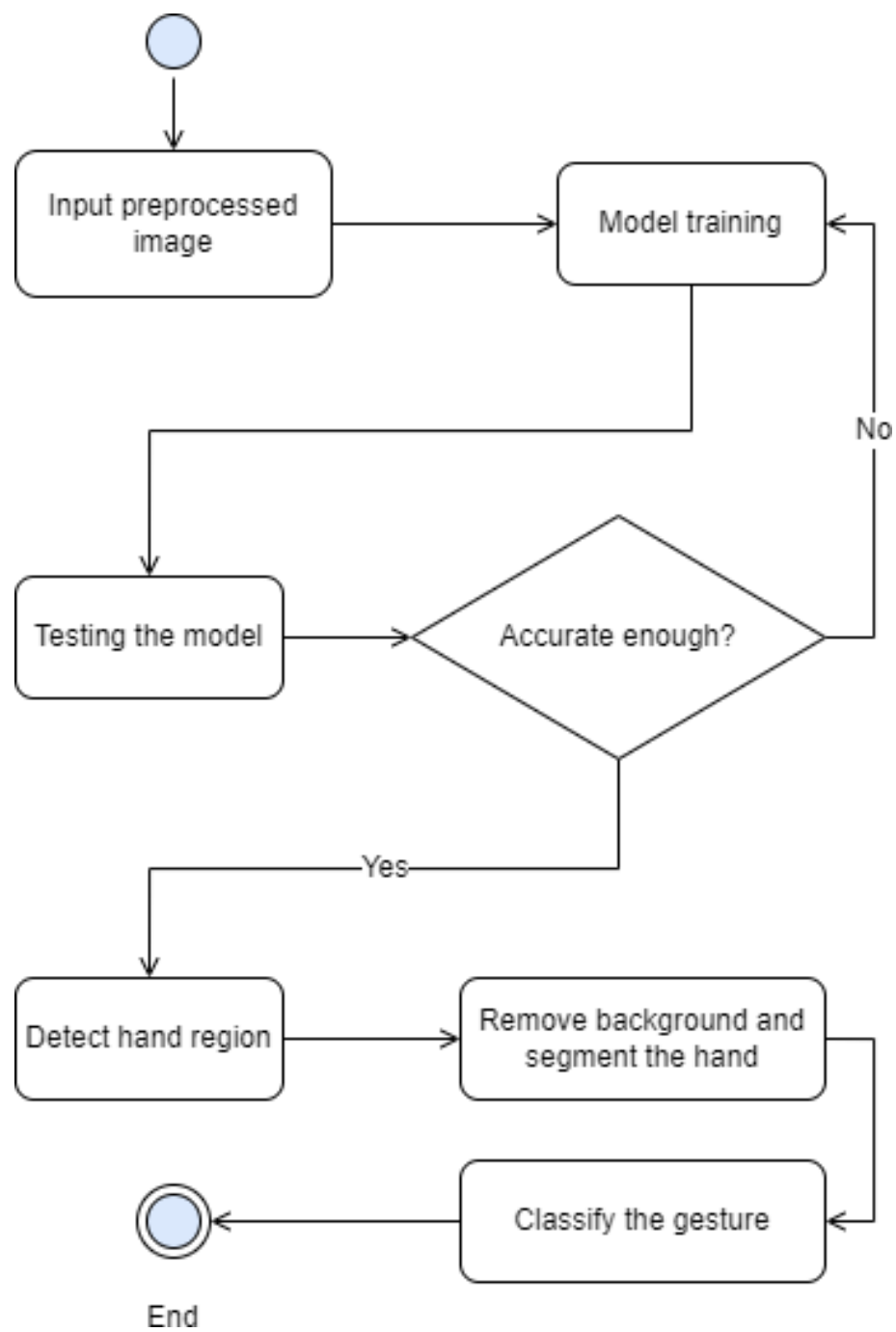
5.4.1 Software Requirements Specification (SRS)

Req. Code	Req. Desc	Use Case	MoSCoW Prioritization
SCM-F-1.0	The preprocessed dataset will be divided into training and testing data.	Training and testing	Must Have
SCM-F-2.0	The training dataset will be fed into a CNN network for several iterations to increase the accuracy of the model.	Training dataset	Must Have
SCM-F-3.0	The CNN network should then take the testing dataset as input for testing the model accuracy.	Testing dataset	Must Have
SCM-F-4.0	The model should take preprocessed input from the video feed and detect the hand region.	Hand region detection and segmentation	Must Have
SCM-F-4.1	The detected hand region should then be extracted from the background to identify gesture.	Feature extraction	Must Have
SCM-F-4.2	Gesture should be classified, and the corresponding text should be displayed.	Gesture classification	Must Have

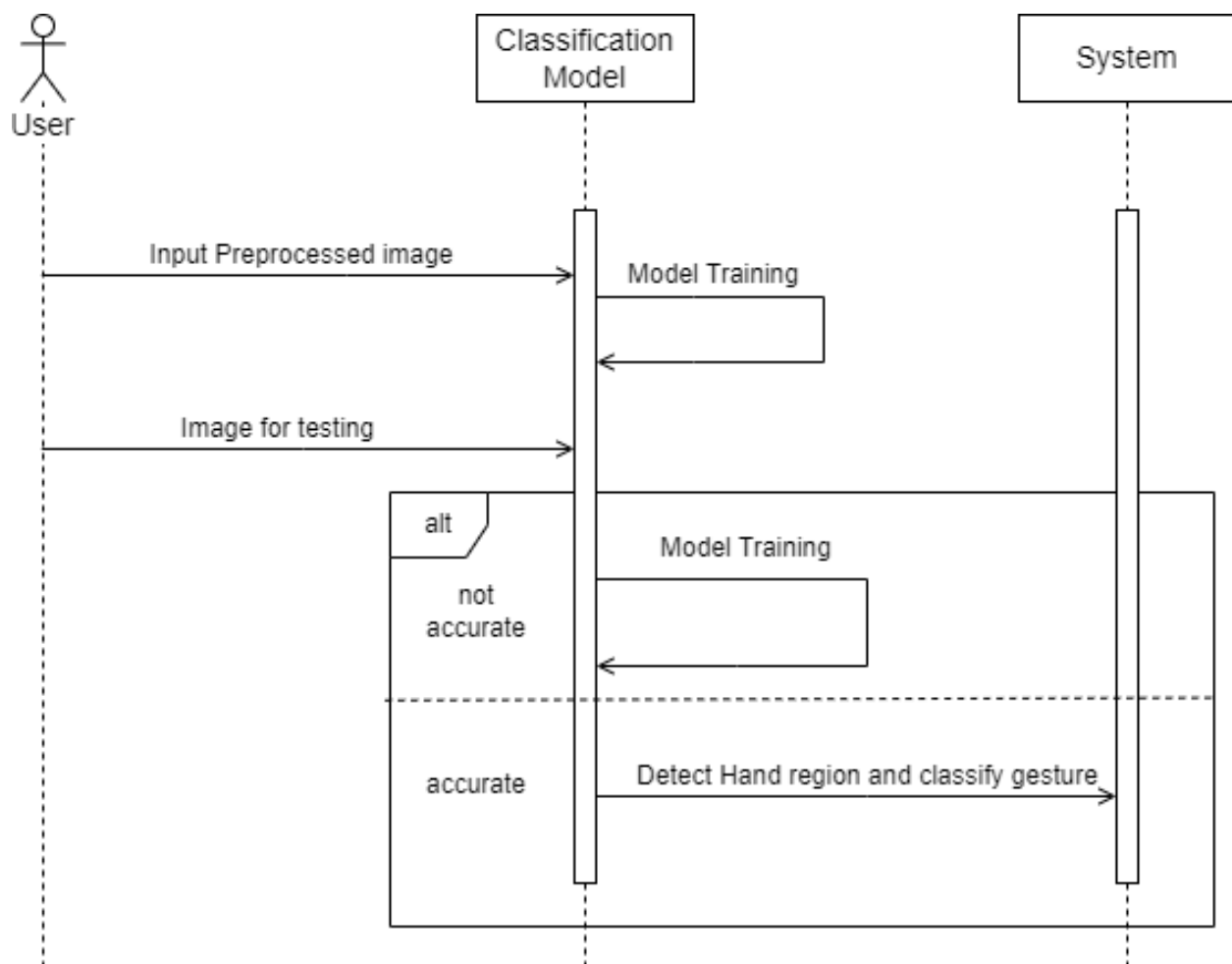
5.4.2 Use Case Diagram



5.4.3 Activity Diagram



5.4.4 Sequence Diagram



5.5 Data Collection

The dataset for each of the 26 letters were collected through background subtraction and hand segmentation. At first, the Region of Interest is defined with its dimensions. Then the accumulated weight of the empty background is calculated. Then the external contours are detected for the hand region. The system waits until it fetches the background. Then the hand is inserted inside the region of interest and captures a number of datasets as defined per the number of frames. In this case, the number of

frames where 300 so 300 images were captured which was split into training and testing.

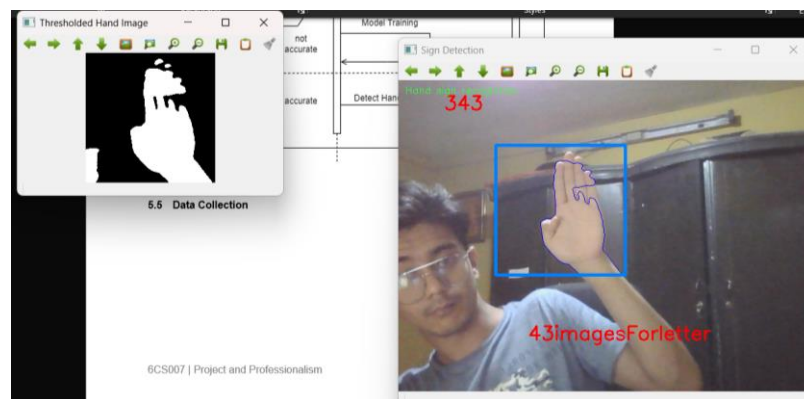


Figure 13 Capturing of dataset

5.5.1 Development of Model

Firstly, the train and test path were defined. Then a dictionary was defined where each letter in the English alphabet were labeled. The images along with their labels were observed first.

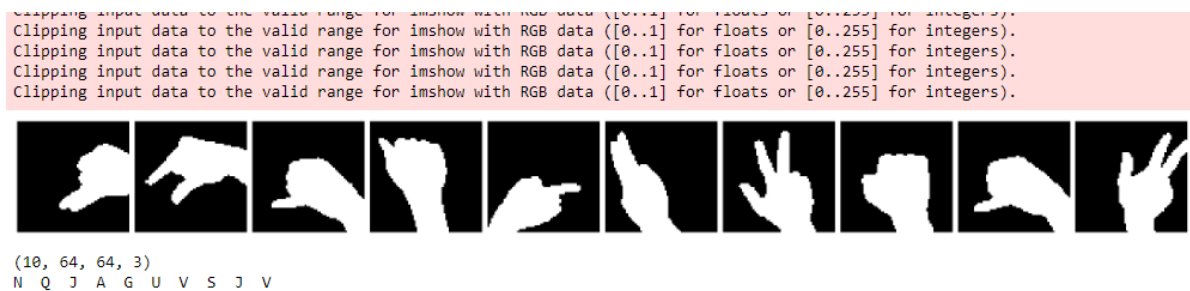


Figure 14 Labeled dataset images

A convolutional neural network was created following the image labelling. First the filter size of 32 was used, followed by a filter size of 64 and 128. The strides taken was 2 and the kernel size was 3,3. **Relu** was used as the activation function. In the first layer, the input shaped of the image was defined as 64,64. The model was then flattened. Then in the output layer again Relu activation was used with a size of 64, 128 and 128. Finally

in the final layer, 26 (the number of letters) size was given and **Softmax** was used for activation function.

```
model = Sequential()

model.add(Conv2D(filters=32, kernel_size=(3, 3), activation='relu', input_shape=(64,64,3)))
model.add(MaxPool2D(pool_size=(2, 2), strides=2))

model.add(Conv2D(filters=64, kernel_size=(3, 3), activation='relu', padding = 'same'))
model.add(MaxPool2D(pool_size=(2, 2), strides=2))

model.add(Conv2D(filters=128, kernel_size=(3, 3), activation='relu', padding = 'valid'))
model.add(MaxPool2D(pool_size=(2, 2), strides=2))

model.add(Flatten())

model.add(Dense(64,activation = "relu"))
model.add(Dense(128,activation = "relu"))
#model.add(Dropout(0.2))
model.add(Dense(128,activation = "relu"))
#model.add(Dropout(0.3))
model.add(Dense(26,activation = "softmax"))
```

Figure 15 Building CNN model

5.5.2 Optimization Evaluation

The model was trained to 10 epochs with loss reduction, and overfitting solving techniques such as early stopping.

```
history2 = model.fit(train_batches, epochs=10, callbacks=[reduce_lr, early_stop], validation_data = test_batches)#, checkpoint]
imgs, labels = next(train_batches) # For getting next batch of imgs...
```

```
Epoch 1/10
650/650 [=====] - 18s 27ms/step - loss: 2.4209 - accuracy: 0.3372 - val_loss: 1.6477 - val_accuracy:
0.5069 - lr: 0.0010
Epoch 2/10
650/650 [=====] - 17s 27ms/step - loss: 0.7128 - accuracy: 0.8028 - val_loss: 1.1395 - val_accuracy:
0.7238 - lr: 0.0010
Epoch 3/10
650/650 [=====] - 17s 27ms/step - loss: 0.2307 - accuracy: 0.9665 - val_loss: 1.0402 - val_accuracy:
0.7223 - lr: 0.0010
Epoch 4/10
650/650 [=====] - 17s 27ms/step - loss: 0.1068 - accuracy: 0.9866 - val_loss: 0.7406 - val_accuracy:
0.8538 - lr: 0.0010
Epoch 5/10
650/650 [=====] - 17s 25ms/step - loss: 0.0062 - accuracy: 0.9995 - val_loss: 0.7517 - val_accuracy:
0.8408 - lr: 0.0010
Epoch 6/10
650/650 [=====] - 17s 26ms/step - loss: 0.0000 - accuracy: 1.0000 - val_loss: 0.7517 - val_accuracy:
0.8408 - lr: 0.0010
```

Figure 16 Optimizing the model

```

model.summary()

Model: "sequential_3"

```

Layer (type)	Output Shape	Param #
conv2d_9 (Conv2D)	(None, 62, 62, 32)	896
max_pooling2d_9 (MaxPooling 2D)	(None, 31, 31, 32)	0
conv2d_10 (Conv2D)	(None, 31, 31, 64)	18496
max_pooling2d_10 (MaxPoolin g2D)	(None, 15, 15, 64)	0
conv2d_11 (Conv2D)	(None, 13, 13, 128)	73856
max_pooling2d_11 (MaxPoolin g2D)	(None, 6, 6, 128)	0
flatten_3 (Flatten)	(None, 4608)	0
dense_12 (Dense)	(None, 64)	294976
dense_13 (Dense)	(None, 128)	8320
dense_14 (Dense)	(None, 128)	16512
dense_15 (Dense)	(None, 26)	3354

```

=====
Total params: 416,410
Trainable params: 416,410
Non-trainable params: 0

```

Figure 17 Model evaluation

5.5.3 Algorithm performance and test data

```
In [81]: print(scores)
print(model.metrics_names)

[2.3675999641418457, 0.800000011920929]
['loss', 'accuracy']
```

Figure 18 Loss and accuracy of model

The results of the model run on test data shows an accuracy of ~80% and a loss of 2.36. The predictions on a small test data shows the correct and incorrect predictions of the model.

```
In [83]: predictions = model.predict(imgs, verbose=0)
print("predictions on a small set of test data--")
print("")
for ind, i in enumerate(predictions):
    print(word_dict[np.argmax(i)], end=' ')

predictions on a small set of test data--

K A O N Y C E U X N
```

```
In [84]: plotImages(imgs)
print('Actual labels')
for i in labels:
    print(word_dict[np.argmax(i)], end=' ')

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
```



Actual labels
K A W N Y C E D X N

Figure 19 Predicted signs vs actual signs

Testing

6 Conclusion

Gestures play an important role in human communication, particularly hand gestures, which has many potential uses in the field of human computer interaction. When compared to other approaches, gesture-based recognition techniques, particularly vision-based hand motions, offer several benefits. The use of additional hardware such as data gloves or armbands are cumbersome and not practical for everyone. Vision-based approach is much more sophisticated which provides near identical results without the extra cost. **From the findings, static hand gesture recognition benefits the most for the project '*HandSpell*' as it reduces the processing requirement of the machine as well as have an increased accuracy over dynamic hand gesture recognition.**

Data preprocessing is a major factor for classifying the hand gestures. It is necessary to choose the most efficient preprocessing methodology. Out of the three mentioned in the paper, data preprocessing and image segmentation using HSV color space is selected as it provides better results in most of the settings as compared to edge detection and segmentation by YIQ/YUV color space. The proposed CNN architecture has high success rates at comparatively low computational cost. This will be possible because of proper image processing where only the region of interest is segmented from the busy background.

Following a well pre-processed dataset, a CNN model having three layers was created. The hidden layer consisted of three more layers with an increasing **filter size of 32, 62 and 128**. For all of the hidden layer **Relu activation** function was used and for the final output layer, a **Softmax function** was employed.

However, current work on hand gesture identification is merely a minor step toward attaining significant achievements in the field of sign language recognition. Only the motions of alphabets from American Sign Language may be recognized in our paper (ASL) and it can also be expanded upon to be able to understand sentences.

7 Critical Evaluation of the Project

7.1 Findings and process

The project had gone through a number of evaluations and tweaks which made it slightly different than the initially anticipated system design. Even though few of the artefacts had been tweaked, the overall working of the system remains nearly the same.

It is a simple and easy to use system which is accessible for both deaf and hearing community for sign language communication.

American Sign Language (ASL) is the most used sign language around the world which is based on English language. There are many other sign languages, but ASL was chosen as it was the most commonly used sign language and has the most easily understandable signs. The use of ASL was comparatively easier for this project as there are only 26 letters for prediction and the hand signs for each letter are also concise and easy to practice. There were few signs which incorporated slight hand movement. This was solved by using the end frame of the hand motion to map it to the letter.

The data collection part went smoothly due to proper data preprocessing and background subtraction. There were slight hiccups while training the CNN model as the accuracy was lower than expected at first. To increase the accuracy, the train datasets were taken again for all the 26 letters under proper lighting and background. This along with choosing the appropriate number of epochs, early stopping, and tweaking the layers helped to attain a good accuracy score.

7.2 Planning, Management, and Quality of Sources

The planning of the whole project was done through **Personal eXtreme Programming (PXP)** methodology. The project was primarily processed according to the priority of each sub-system. The features with higher priority were completed first followed by features with lower priority. A realistic Gantt chart was also prepared which dated the initialization from **17th November** to the final completion on **24th April**.

For the research of Hand Sign recognition, a total of 20 research papers were accumulated beforehand. These papers and journal articles were all collected through different sources such as core, sematischolar, and connected papers. There was a vast knowledge on the use of Neural Networks on hand image classification and prediction. Among all of the different methods, the use of CNN had the best results which was used in this project.

7.3 Self-reflection


This project, *HandSpell* was a great learning experience for me. The fact that I was able to conduct in-depth research on **Neural Networks** and **Deep Learning** was the major plus point for me. There were research on different research papers which used supervised as well as unsupervised learning approach for sign prediction. Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Naïve Bayes algorithm, and Convolutional Neural Networks (CNN) were few of the methods used for image classification. I learned that CNN is the most effective neural network as it reduced the number of parameters without losing on the model's quality. I learned that CNN is the most efficient neural network for image processing and classification. This project was a great introduction to Deep Learning and Neural Networks. I also learned data pre-processing for 2D images, creating an AI (CNN) model and implementing it in a website, and finally testing the website and the model. The whole process of creating a ASL recognition system has overall helped me in my personal skills in both web development as well as AI field.

8 Evidence of Project Management

8.1 Log Sheet

- Log Sheet 1

Faculty of Science and Engineering
School of Mathematics and Computer Science



UNIVERSITY OF
WOLVERHAMPTON

PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. <u>Sanjit Napti</u>
Project Title: HandSpell	Month: December
What have you done since the last meeting	
Researched on 20 different articles and research papers and created notes from the findings from each of the research papers.	
What do you aim to complete before the next meeting	
Start the literature review and complete the introduction section. Research on the available datasets on hand sign language alphabets.	
Supervisor comments	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____


Date: 03-12-2021


Supervisor Signature: Sanjit Napti

Date: 03-12-2021

- Log Sheet 2

Faculty of Science and Engineering

 UNIVERSITY OF WOLVERHAMPTON

 School of Mathematics and Computer Science

PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sanjit Nadi
Project Title: HandSpell	Month: December
What have you done since the last meeting	
Introduction section of literature review has been completed. Researched on sign alphabets datasets.	
What do you aim to complete before the next meeting	
Continue the body section of literature review. Label the images from the dataset and divide them into training and testing datasets. Complete the wireframe design for the HandSpell web-application.	
Supervisor comments	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____


Date: 10-12-2021


Supervisor Signature: _____

Date: 10-12-2021

- Log Sheet 3

Faculty of Science and Engineering

 School of Mathematics and Computer Science

 UNIVERSITY OF
WOLVERHAMPTON

PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjit Napit
Project Title: HandSpell	Month: December
What have you done since the last meeting	
Studied research papers and completed four paragraphs from four different ideas. Designed the wireframe of the system.	
What do you aim to complete before the next meeting	
Complete the literature review and present the full prototype design of the system.	
Supervisor comments	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____

Date: 12-18-2021

Supervisor Signature: _____

Date: 12-18-2021

- Log Sheet 4

Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: January
What have you done since the last meeting	
Completed the literature review and prototyping of the system was done.	
What do you aim to complete before the next meeting	
Database for storing hand sign images for searching manual alphabets.	
Supervisor comments	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____

Date: 01-07-2022

Supervisor Signature: _____

Date: 01-07-2022

- Log Sheet 5

Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: January
What have you done since the last meeting	
<p>Researched on how to store images in django's database for hand sign storing.</p>	
What do you aim to complete before the next meeting	
<p>Complete artifact designs which included FDD, SRS document, class diagram, use case diagram, activity diagram, and sequence diagram.</p>	
Supervisor comments	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____

Date: 01-14-2022

Supervisor Signature: _____

Date: 01-14-2022

- Log Sheet 6

Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: January
What have you done since the last meeting	
Completed artifact designs which included FDD, SRS document, class diagram, use case diagram, activity diagram, and sequence diagram.	
What do you aim to complete before the next meeting	
Complete the feature for searching hand sign alphabets.	
Supervisor comments	
Execute tasks based on their priority. Also, try expanding the class diagram before the final report submission as it is very limiting.	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____

Date: 01-28-2022

Supervisor Signature: _____

Date: 01-28-2022

- Log Sheet 7

Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: February
What have you done since the last meeting	
Completed the feature for searching hand sign alphabets.	
What do you aim to complete before the next meeting	
Complete the frontend for the user panel and initiate backend work.	
Supervisor comments	
Focus on a single task rather than overwhelming yourself with multiple tasks.	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____

Date: 02-11-2022

Supervisor Signature: _____

Date: 02-11-2022

- Log Sheet 8

Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: February
What have you done since the last meeting	
Worked more on frontend and changed few design elements of the home page.	
What do you aim to complete before the next meeting	
Work on professionalism report and submit draft.	
Supervisor comments	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____

Date: 02-18-2022

Supervisor Signature: _____

Date: 02-18-2022

- Log Sheet 9

Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: February
What have you done since the last meeting	
Worked on professionalism report and completed up to ethical aspects.	
What do you aim to complete before the next meeting	
Complete the professionalism report for the final submission.	
Supervisor comments	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____

Date: 02-26-2022

Supervisor Signature: _____

Date: 02-26-2022

- Log Sheet 10

Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: March
What have you done since the last meeting	
Completed the professionalism report, researched on OpenCV.	
What do you aim to complete before the next meeting	
Capture the dataset of different hand signs using OpenCV.	
Supervisor comments	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____

Date: 03-04-2022

Supervisor Signature: _____

Date: 03-04-2022

- Log Sheet 11

Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: March
What have you done since the last meeting	
Data capture of different hand signs using OpenCV for training the model.	
What do you aim to complete before the next meeting	
Set up the requirements for model and use the captured dataset for training.	
Supervisor comments	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____

Date: 03-11-2022

Supervisor Signature: _____

Date: 03-11-2022

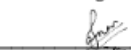
- Log Sheet 12

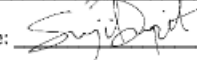
Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: March
What have you done since the last meeting	
Requirements for the CNN model like tensorflow, OpenCV, and object detection libraries were set up.	
What do you aim to complete before the next meeting	
Start draft report of final report and advanced version of artefacts.	
Supervisor comments	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature:  Date: 03-25-2022

Supervisor Signature:  Date: 03-25-2022


- Log Sheet 13

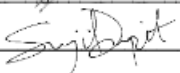
Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: April
What have you done since the last meeting	
Start draft report of final report and advanced version of artefacts.	
What do you aim to complete before the next meeting	
Train the model and detect hand signs in real time.	
Supervisor comments	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature:  Date: 04-01-2022

Supervisor Signature:  Date: 04-01-2022

- Log Sheet 14

Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: April
What have you done since the last meeting	
The model was trained on the training dataset and researched for errors on real time video detection.	
What do you aim to complete before the next meeting	
Detect video feed from the webcam in real time.	
Supervisor comments	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____

Date: 04-08-2022

Supervisor Signature: _____

Date: 04-08-2022

- Log Sheet 15

Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: April
What have you done since the last meeting	
Text to sign conversion artefact was done and further UI improvements.	
What do you aim to complete before the next meeting	
Work on completing sign to text conversion using live video feed.	
Supervisor comments	
Please focus on a single and more realistic task at a time.	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____

Date: 04-15-2022

Supervisor Signature: _____

Date: 04-15-2022

- Log Sheet 16

Faculty of Science and Engineering
School of Mathematics and Computer Science



PROJECT MANAGEMENT LOG	
First Name: Saman	Surname: Tamrakar
Student Number: 2049873	Supervisor: Mr. Sarjil Napit
Project Title: HandSpell	Month: April
What have you done since the last meeting	
Researched and worked on integrating the CNN model with the Django website.	
What do you aim to complete before the next meeting	
Complete model integration and testing of the website.	
Supervisor comments	
Please fix the important functionality of your system as discussed in the meeting.	

We confirm that the information given in this form is true, complete, and accurate.

Student Signature: _____

Date: 04-22-2022

Supervisor Signature: _____

Date: 04-22-2022

8.2 Gantt Chart

	ACTIVITIES	ASSIGNEE	EH	START	DUE
	Data Collection and Analysis:		-	17/Nov	28/Nov
1	✓ Capturing Image with web camera		-	17/Nov	25/Nov
2	✓ Dataset image labelling		-	25/Nov	28/Nov
	Image Processing and UI Design:		-	28/Nov	30/Dec
4	✓ Hand identification and segmentation		-	28/Nov	10/Dec
5	✓ Rescale, cropping, and labelling of image		-	10/Dec	22/Dec
6	✓ Wireframing and Prototype Design		-	22/Dec	30/Dec
	Feature Extraction and Development:		-	30/Dec	15/Feb
8	✓ Enclose hand region within bounding box		-	30/Dec	05/Jan
9	✓ Binary conversion of image and scaling		-	05/Jan	10/Jan
10	✓ Database designing		-	10/Jan	31/Jan
11	✓ Frontend Development		-	01/Feb	15/Feb
	Gesture Classification:		-	16/Feb	06/Apr
13	✓ Building CNN model		-	16/Feb	20/Mar
14	✓ Backend Development		-	16/Feb	06/Apr
15	✓ Model Evaluation		-	28/Mar	05/Apr
	Text to Sign Language Translation:		-	06/Apr	24/Apr
17	✓ Collection and labelling of sign alphabets		-	06/Apr	08/Apr
18	✓ API for fetching the images		-	08/Apr	15/Apr
19	✓ Black box and White box testing		-	15/Apr	20/Apr
20	✓ Deployment		-	20/Apr	24/Apr

Figure 20 Task list for Gantt Chart

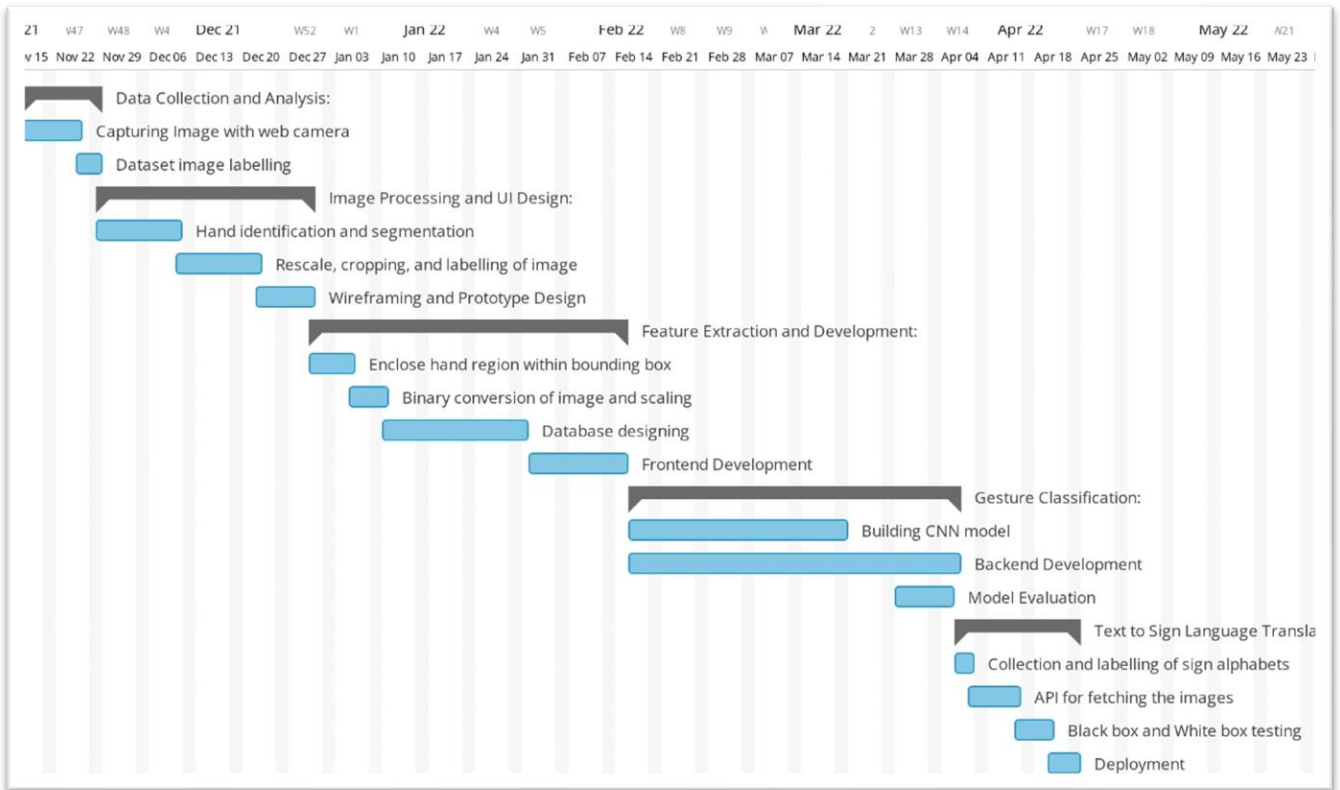


Figure 21 Gantt Chart

9 References

- Adithya & Rajesh, 2020. Third International Conference on Computing and Network Communications. *A Deep Convolutional Neural Network Approach for Static HandGesture Recognition*, p. 9.
- Bhavana, D., 2021. International Journal of Performance Analysis in Sport. *Hand Sign Recognition using CNN*.
- Cardoso, T., 2016. 6th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Infoexclusion. *Hand Gesture Recognition towards Enhancing Accessibility* , p. 11.
- Chen, D. & Li, G., 2017. Physical Sensors. *An Interactive Image Segmentation Method in Hand Gesture Recognition*.
- Chen, J., 2017. Stanford University CS231A. *Sign Language Recognition with Unsupervised Feature Learning*, p. 8.
- Hurroo, M. & Walizad, M. E., 2020. International Journal of Engineering Research & Technology (IJERT). *Sign Language Recognition System using Convolutional Neural Network and Computer Vision*.
- Hurroo, M. & Walizad, M. E., 2020. International Journal of Engineering Research & Technology (IJERT). *Sign Language Recognition System using Convolutional Neural Network and Computer Vision*, p. 6.
- Islam, M. Z. & Hossain, M. S., 2019. 2019 Joint 8th International Conference on Informatics. *Static Hand Gesture Recognition using Convolutional Neural Network with Data Augmentation*.
- Sofiane, M., 2018. Multimedia Tools and Applications. *Automatic Hand Detection in Color Images based on skin region verification*.

- Visual Studio, 2022. *Getting Started*. [Online]
Available at: <https://code.visualstudio.com/docs>
[Accessed 24 March 2022].
- Wankhede, P. & Nagpal, P. N., 2019. International Journal for Research in Applied Science & Engineering Technology (IJRASET). *Dynamic Hand Gesture Recognition*, 7(V), p. 6.
- Zengeler, N., 2018. Physical Sensors. *Hand Gesture Recognition in Automotive Human–Machine Interaction Using Depth Cameras*, p. 28.