# Individual Project Documentation
# CHP2524-QGA-YEAR-2425

Kashif Hussain

U2255034

**“To develop a real-time hand sign language detection system using computer vision and machine learning, aimed at improving communication and accessibility.”**

## Table of Contents

## 1. Abstract

This project introduces Clarity, an interactive web-based system that identifies static sign language motions and translates them to audible speech in real time. The method is intended to improve accessibility and aid those with speech impairments or communication hurdles by allowing seamless translation of signed letters into spoken language. Clarity uses a custom convolutional neural network (CNN) trained on hand gesture datasets to capture live webcam input, classify signed letters, generate word suggestions based on contextual prediction and allow users to construct and vocalise sentences using keyboard shortcuts and built-in browser speech synthesis. The application was built with Python, Flask, OpenCV, HTML/CSS and JavaScript, with all inference and interaction done locally. Under ideal conditions, testing showed good dependability, however lighting and background variance had an impact on performance. The system is intended to be accessible, adaptable and responsive to user feedback, and it serves as a platform for future integration with wearable technology or more advanced sign language systems. Clarity demonstrates how machine intelligence and web technologies may help overcome communication barriers and create more inclusive digital environments.

## 2. Introduction

### 2.1 Product Description

Communication lies at the centre of the human experience. It influences our relationships, encourages cooperation and allows us to express ourselves intellectually and emotionally. However, for members of the Deaf and Hard of Hearing communities, ordinary communication can frequently serve as a barrier rather than a bridge. Sign language is a strong form of expression, but its use is not generally understood, especially in spoken-language situations. While many technology advancements have concentrated on speech recognition and natural language processing, there is still a noticeable lack of attention paid to real-time sign language identification and interpretation. This study intends to bridge that gap by developing a real-time hand gesture detection system based on computer vision and machine learning.

The objective is to develop a working system that enables for real-time interpretation of the core hand motions associated with sign language, serving as an early step to full-scale sign language translation in the future. The system is driven by YOLOv5, a cutting-

edge object identification model identified for its speed and precision, which has been trained on a dataset of sign gestures. The interface is intended to be simple, intuitive and responsive, making it suitable for both instructional and assistive needs.

### 2.2 Aim

**“To develop a real-time hand sign language detection system using computer vision and machine learning, aimed at improving communication and accessibility.”**

### 2.3 Objectives

- To investigate the current state of sign language recognition, including available technologies and acknowledged limits in the area.
- To create a dataset of fundamental hand motions appropriate for training a deep learning model.
- To create a gesture detection model using YOLOv5, pick the required parameters and training techniques to ensure accuracy and efficiency.
- To create a realtime system that takes live video input, analyses it using the trained model, and displays the discovered indicators.
- To create and build a basic yet user-friendly frontend interface that enables smooth interaction, particularly for users with accessibility requirements.
- To assess the system's performance under various illumination situations and hand positions.
- Propose future improvements and assess the commercial or educational potential of the system

## 3. Background and Literature Review

### 3.1 The Evolution of Sign Language Recognition

Over the last two decades, the subject of sign language recognition (SLR) has evolved from experimental academic research to increasingly practical and performance-driven applications, due primarily to advances in computer vision and deep learning. According to Koller et al. (2015), previous techniques to sign language detection relied heavily on handmade features and rule-based algorithms, which frequently used gloves

or markers to track hand movement. These methods, while helpful for isolated sign identification, suffered with scalability and real-time application due to environmental sensitivity and insufficient adaptation to signer variation.

The merging of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) resulted in significant accuracy gains, particularly when optical flow tracking and 3D hand position estimation were included. Huang et al. (2018) demonstrated that integrating 2D CNNs with 3D skeletal tracking may achieve gesture classification accuracy greater than 90% in controlled scenarios. However, real-time SLR systems frequently encounter performance decreases due to variations in illumination, obstruction and signer-dependent characteristics.

The merging of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) resulted in significant accuracy gains, particularly when optical flow tracking and 3D hand position estimation were included. Huang et al. (2018) demonstrated that integrating 2D CNNs with 3D skeletal tracking may achieve gesture classification accuracy greater than 90% in controlled scenarios. However, real-time SLR systems frequently encounter performance decreases due to variations in illumination, obstruction and signer-dependent characteristics.

The emergence of real-time object detection models like YOLOv3 and YOLOv5 has accelerated advances in static sign recognition. According to Jocher et al. (2020), YOLOv5 provides cutting-edge object identification performance while maintaining low latency, making it ideal for applications needing live video processing. Several recent research (e.g. Sharma et al. 2021) have demonstrated promising results employing YOLO designs for static hand motion detection, with accuracy levels above 95% on limited datasets.

Despite these gains, the research identifies a persisting gap in real-world application. Most systems are trained on datasets with little change in skin tone, backdrop, or camera angle, severely reducing generalisability. Furthermore, while research is increasingly focussing on continuous sign language recognition (CSLR), commercially accessible technologies generally lag, allowing only isolated word detection from static photos or videos.

As of 2023, there is an increased interest in using SLR models to improve accessibility in education, communication tools, and assistive technologies. However, many of these systems are limited to academic settings or need expensive technology. Silva et al. (2022) report that fewer than 20% of published SLR systems have completed real-world testing with Deaf users, emphasising the significance of bridging the gap between model correctness and usability.

This project is inspired by the progress of SLR research and aims to contribute to it by developing a real-time, YOLOv5-based sign identification system that is lightweight,

accessible, and customisable. It acknowledges the limits of previous work, notably in dataset bias, signer variability, and real-world testing, and presents a modular, extensible approach that may evolve in complexity over time.

### 3.2 Computer Vision & Machine Learning in Gesture Recognition

Computer vision and machine learning have been critical to the improvement of gesture recognition systems, notably in sign language identification. At their core, these technologies aim to allow computers to interpret visual information in a way that replicates human vision. Gesture recognition, or hand sign recognition, is a subcategory of computer vision that detects, sections and classifies hand positions and movements in images or video streams.

Traditional computer vision systems included manual feature extraction techniques such edge detection, background removal, skin colour segmentation and motion tracking (Wang et al., 2009). While these approaches were computationally light, their accuracy was strongly impacted by lighting conditions, camera angle and user-specific differences like skin tone or hand size. These constraints hindered its scalability and dependability in real-world settings.

The use of machine learning techniques solved some of these issues by providing data-driven algorithms that could learn to recognise patterns in gesture data. Early models for classifying static hand gestures included Support Vector Machines (SVMs), k-Nearest Neighbours (k-NN), and Hidden Markov Models (HMMs). However, they often required substantial preprocessing and were unable to handle more sophisticated or continuous movements.

Recent advances in deep learning, especially Convolutional Neural Networks (CNNs), have greatly enhanced gesture detection capabilities. CNNs are ideal for image classification problems because they can automatically learn spatial structures of features from raw picture input. In the field of hand gesture identification, CNNs have shown great accuracy in recognising static hand gestures under controlled settings (Molchanov et al., 2016). Furthermore, because of their ability to generalise across different backdrops and hand shapes, they serve as the foundation for the majority of current sign detecting systems.

More sophisticated real-time detection systems currently use object detection architectures like YOLO (You Only Look Once), SSD (Single Shot Detector) and Faster R-CNN. These models categorise motions and use bounding boxes to localise them inside the frame. YOLOv5 is known for its speed and efficiency, allowing for real-time detection with minimum computing overhead, making it perfect for gesture recognition on resource-constrained systems (Jocher et al., 2020).

Despite these advances, difficulties persist. Real-time performance can be influenced by frame resolution, lighting changes and shadowing. Furthermore, many models have difficulty with continuous movements or transitions between signals, limiting their utility for comprehensive sign language interpretation. To circumvent these limitations, recent research is looking into hybrid models that combine spatial and temporal characteristics, such as merging CNNs with Long Short-Term Memory (LSTM) networks to capture motion dynamics over time.

The application of computer vision and machine learning in gesture detection is fast evolving. The availability of labelled datasets and pre-trained models has made constructing effective sign detecting systems more accessible than ever before. However, developing systems that are inclusive, dependable and adaptive to real-world use cases is an important area of ongoing study and innovation.

### 3.3 Applications of Hand Gesture Detection in Accessibility

Hand gesture recognition, particularly sign language identification, has broad applications in the field of accessibility. Sign language is more than just a means of communication for the Deaf and Hard of Hearing people. It is also a cultural and social identity. However, the general population's lack of knowledge of sign language continues to cause difficulties in daily encounters, whether in healthcare, education, employment or public services. Technology that can understand hand gestures in real time can help to close communication barriers and promote more inclusivity.

One of the most prominent uses of gesture detection is the creation of assistive communication tools. These programs may convert sign language into written or spoken text, allowing non-signers to comprehend signers without requiring a human translator. In settings where immediate communication is frequently required, such as hospitals, emergency services or retail environments, technology like this can dramatically enhance user experience and minimise dependency on costly interpreting services (Silva et al., 2022). When used alongside text-to-sign synthesis engines, gesture recognition systems can enable two-way communication, resulting in more engaged and welcoming interactions.

Sign language identification systems can be used in educational settings to promote inclusive learning environments. Deaf students in regular schools frequently rely on interpreters, resulting in a social and geographic divide. Real-time sign translation systems incorporated in classroom interfaces or learning applications allow for more direct connection with teachers and classmates. Furthermore, such tools can help with

language acquisition by providing feedback to students practicing signs, like how pronunciation tools help spoken language learners (Gupta et al., 2020).

Gesture recognition can also help with accessibility in the workplace. Customer service representatives, team members and clients who use sign language frequently suffer communication challenges unless pre-arranged interpretation is provided. Sign recognition technologies incorporated into meeting platforms or workplace applications may promote more inclusive professional settings and minimise communication friction.

Beyond traditional apps, the future of accessibility is moving to more seamless and accessible options. Recent advances in augmented reality and AI-powered wearable gadgets, such as smart glasses, have enabled the integration of gesture recognition systems directly into common accessories. These glasses could recognise signals in real time, show translated text on a built-in screen, and even convert movements to voice, providing hands-free, intuitive communication help in both public and private contexts. The gesture detection model created in this study might be used as the basic layer for such technology, increasing their usability and social effect.

As these technologies advance, their success will be judged not just by detection accuracy, but also by ethical concerns, cultural sensitivity and direct connection to the people they aim to benefit. Accessibility is more than a feature; it is a duty in the establishment of inclusive, human-focused technology.

### 3.4 Existing Systems and Technologies

The field of sign language recognition has witnessed the creation of several methods and technologies, each seeking to handle unique elements of gesture interpretation, ranging from static picture categorisation to real-time continuous signing. These systems span from research prototypes to commercial tools, with various levels of usability, accuracy and utilisation. A review of these current solutions is required to understand this project within the larger environment and to identify opportunities for further development.

One of the earliest and most well-known technologies in this area is the Sign Language Recognition (SLR) system, which employs sensor-based gloves such as the "AcceleGlove" and MIT's "Gestural Interface Glove." These devices included accelerometers and flex sensors into wearable gloves to detect hand and finger positions (Kadous, 2002). While sensor-based techniques delivered great precision in controlled environments, they were often costly, required calibration and lacked

comfort and practicality for everyday usage. Their reliance on wearable technology hindered use among regular consumers, particularly among the Deaf population, which prefers easy, subtle solutions.

Vision-based systems have grown in popularity as computer vision and deep learning technologies progress. Microsoft's Kinect was among the first commercial systems that use depth sensing for hand and body movement identification. It offered gesture-based engagement in gaming and educational contexts, but its use in sign language recognition was restricted due to the requirement for fixed hardware placement and limited model adaptation.

Several significant open-source gesture detection systems were developed as a result of academic research. For example, the RWTH-PHOENIX-Weather corpus has served as the foundation for various real-time sign recognition models, especially those built on CNN-RNN or Transformer architectures (Camgoz et al., 2018). These models demonstrated the potential of continuous sign language translation from video input, although they had significant word mistake rates and were greatly dataset dependent. More contemporary systems, such as Google's MediaPipe Hands library, offers real-time hand tracking with a single RGB camera, giving skeletal markers for gesture recognition. MediaPipe has been widely used in experimental and lightweight applications, although it lacks native support for comprehensive sign language interpretation.

Another famous example of wearable gesture-based technology is Imogen Heap's Mi.Mu Gloves. Originally created to allow creative musical performance using hand and finger movements, these gloves use motion sensors, gyroscopes and machine learning algorithms to translate physical gestures to digital commands in real time. While the Mi.Mu Gloves were designed primarily for artistic purposes, they show the potential for intuitive, wearable interfaces that effortlessly combine human expression with technology. The design concepts, which include real-time responsiveness, user-centred mapping and a modest form factor, are strongly aligned with the goals of sign language recognition systems. This crossover shows that technology designed for the creative sectors can be inspired or adapted for use in accessibility-focused applications.

*Appendix - Figure 1: The anatomy of Imogen Heap’s Gloves*

Commercially, businesses such as SignAll have created real-time sign translation systems that identify and interpret American Sign Language (ASL) using a mix of computer vision and 3D modelling. Their method uses depth cameras, natural language processing and animated avatars to provide feedback. While SignAll works well in controlled situations, its proprietary nature and high hardware requirements make it less accessible to independent developers and educators who have limited resources.

Similarly, Google's Project Relate and Meta's AR-based sign language translation research have investigated combining voice and visual AI to aid communication for those with speech or hearing problems. These projects represent an increasing industry interest in inclusive AI solutions, although many are still in the prototype stage or lack transparency about performance metrics and deployment models.

Despite these developments, existing systems share a common theme: the trade-off between accuracy, usability and accessibility. Many high-performance models typically require specialist hardware, carefully selected datasets, or closed-source platforms. Others, although being open source, suffer with real-world applicability due to inadequate generalisation or a restricted sign language.

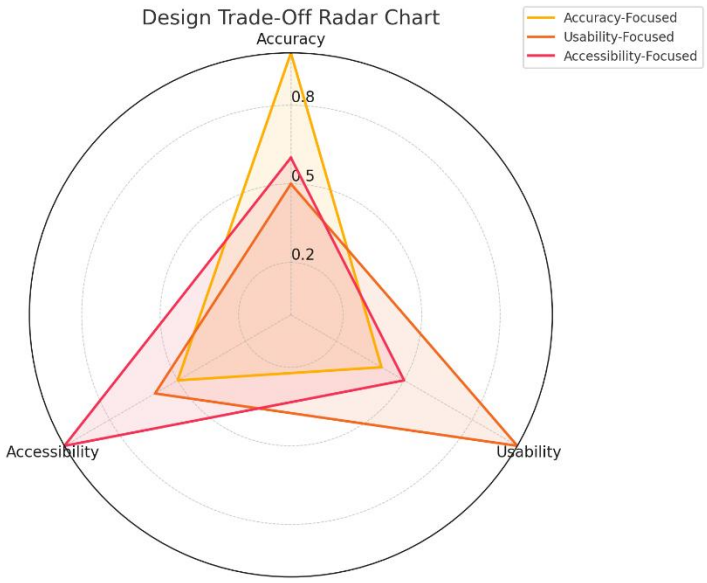


*Figure 2: Radar chart illustrating the trade-offs between accuracy, usability, and accessibility in different gesture recognition system designs. Each line represents a design profile that prioritises one dimension while compromising on the others. The chart highlights the difficulty of optimising all three factors simultaneously in real-world applications.*

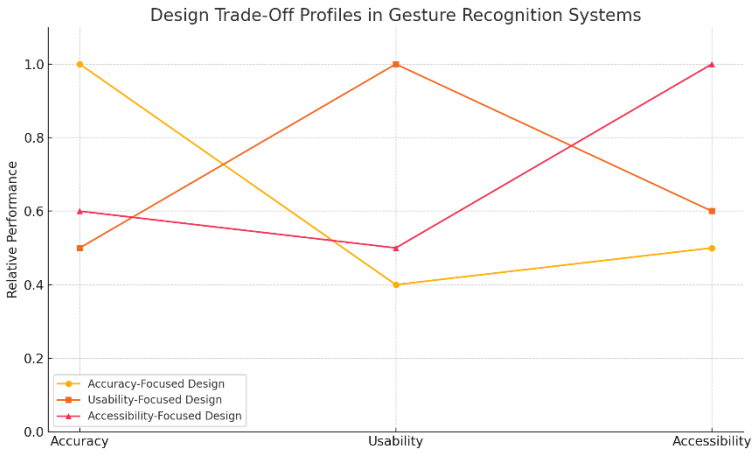


*Figure 3: Trade-off profiles in gesture recognition system design. Each design focus optimises one dimension (accuracy, usability, or accessibility), often at the cost of the others. Achieving balance is key to building inclusive, high-performance systems.*

The solution created in this project is intended to fill that gap by combining the capabilities of open-source real-time detection (YOLOv5) with an emphasis on usability, versatility and accessibility. It is not intended to compete with large-scale proprietary systems, but rather to show that a lightweight, camera-only sign identification tool may provide useful functionality, particularly in educational or communication assistance situations.

### 3.5 Analysis of Gaps and Opportunities

Despite enormous improvements in sign language recognition systems, there is still a noticeable gap between academic innovation and real-world application. A detailed examination of what is currently available demonstrates numerous important gaps, both technological and practical, emphasising the need for solutions that are not just accurate, but also useable, scalable and accessible to the people they want to serve.

One of the most persistent gaps is the lack of real-time, low-cost solutions that work consistently in unconstrained environments. While many academic models claim excellent accuracy rates, these results are frequently obtained from controlled datasets with uniform lighting, constant backdrops and no change in hand shape, skin tone or movement speed. In contrast, real-world usage takes place in dynamic circumstances such as outdoor lighting, mobile camera input and different user demographics. As a result, many high-performance models do not generalise outside the controlled contexts in which they were designed (Silva et al., 2022).

Furthermore, many solutions rely on specialist hardware, such as depth sensors or sensor-based gloves, which are not necessarily practical or economical to average users. While these technologies improve accuracy, they constitute a barrier to access, especially in impoverished countries or public places where cost, availability or user familiarity may be limiting issues. This technology dependence prevents a big percentage of the population from benefiting from gesture recognition advancements.

Another issue is that most existing models have a restricted focus, frequently supporting just a tiny vocabulary of signals and requiring users to stop between gestures. This limits fluid communication and does not convey the expressive, continuous aspect of true sign language. Furthermore, few systems offer multi-modal assistance, such as including facial expressions, which are critical components of most sign languages.

In terms of usability, many gesture recognition systems lack appealing user interfaces that enable smooth real-time interaction. Systems that are technically functional frequently fail to deliver great user experiences, especially for non-technical users or those with special accessibility requirements.

These challenges create chances for innovation. There is a clear demand for gesture recognition systems that emphasise modularity, real-time responsiveness and simplicity of integration across several platforms, ranging from mobile devices to wearable technologies. Lightweight, open-source models that may be trained or fine-tuned with minimal hardware resources are very useful in educational or low-resource environments.

The concept described in this study seeks to fill this gap by providing a camera-only, YOLOv5-powered device that is both lightweight and adaptable. It illustrates how current open-source tools may be modified to meet accessibility goals without requiring specialist hardware or commercial APIs. Furthermore, the project proposes a user-centred design approach that explores how such technology might be expanded into future applications, such as incorporation into wearable devices for real-time translation.

In conclusion, while the area of sign language recognition has made significant progress, there is still a crucial need for systems that balance accuracy, usability and accessibility – not as conflicting aims, but as equal priority in the creation of inclusive assistive technology.

**3.6 Summary of Findings**

This literature review looked at the evolution of sign language recognition technology, the role of computer vision and machine learning in gesture identification, and the rising demand for accessible, user-friendly systems that serve real-world populations. A clear conclusion has become apparent: while technological advances in gesture detection have been significant, practical implementations frequently fall short due to trade-offs between accuracy, usability, and accessibility.

Historically, solutions were either focused on performance in controlled conditions or required specialist hardware, such as sensor gloves or depth cameras. These configurations indicated technological capability but posed challenges for everyday use. Vision-based models have increased accessibility by utilising ordinary cameras, but many still lack real-time responsiveness and sensible user interface design. Furthermore, small vocabulary sizes and low generalisation among signers and contexts remain significant flaws.

Several tools and frameworks were evaluated, including MediaPipe, YOLO-based detection algorithms and open-source datasets like RWTH-PHOENIX. Each has a unique set of advantages, but they also have limitations in terms of scalability, adaptability and simplicity of implementation. Innovative wearable systems, such as Imogen Heap's Mi.Mu Gloves, show that expressive, gesture-based communication with

technology is both possible and desired, even if first used in artistic contexts.

Importantly, this analysis found that no contemporary solution addresses all three criteria of accuracy, usability and accessibility. Trade-offs are still inevitable in design decisions, as seen clearly in the charts provided. This gap opens a clear potential for modular, camera-based systems that may be deployed in resource-constrained contexts and tailored to user requirements.

These opportunities serve as the foundation for the gesture detection system created in this research. It works with a camera interface, employs a real-time object identification model to categorise signed characters, and provides word predictions that the user may pick via key inputs. By including a text-to-speech component, the system bridges the gap between signed and spoken language, strengthening communication and providing practical assistance to individuals who rely on sign language in everyday situations.

This evaluation lays the groundwork for the report's next section, which will look into how the project was designed, improved and implemented to address the real-world needs identified here.

## 4. Project Ideation and Evolution

### 4.1 Initial Concept and Direction.

At the start of this project, the original concept was very different from what became "Clarity". Early on, I experimented with more utility-oriented projects, such as FridgeMate, a web service that recommends meals based on contents in a user's refrigerator. While technically appealing, this idea lacked emotional depth and long-term relevance to my own interests and professional goals. As I moved through my last year, I began to see a mismatch between the technical work I was doing and the impact that I hoped my work would have.

This insight prompted a shift away from convenience-based initiatives and towards something more meaningful and socially significant. I became more interested in the combination of artificial intelligence and accessibility, namely the idea of developing tools that facilitate communication for under-represented cultures. That's when the primary issue arose: "How can AI help sign language users communicate with the rest of the world?"

### 4.2 Reflection and redirection

As I considered this shift in approach, I concentrated on developing a system that would not aim to replace or understand entire sign language grammar, but rather to facilitate communication in a lightweight, modular manner. I landed on the idea of a system that identifies signed letters via a camera, predicts probable word completions, and allows users to pick a word to be pronounced out using text-to-speech.

This strategy blended technological feasibility with social effect. It allowed me to delve into the key technologies — computer vision, real-time detection, user interface design and TTS integration – while keeping the user experience simple, straightforward and accessible. More significantly, it established a clear goal for the project: to make it easier for people who use sign language to communicate in real-time.

### 4.3 Identifying Value in Accessibility

The decision to prioritise accessibility was motivated by values rather than technological considerations. The more I explored the issues encountered by the Deaf and Hard of Hearing population, the more I understood how little tools are available for real-time communication, particularly in casual or spontaneous circumstances.

Clarity arose in answer to this need: a system that does not require expensive gear or substantial training to use and runs purely in-browser with a normal camera. Users may sign letters, receive choices based on what they're attempting to say (for example, writing the letter "P" offers alternatives such as "Prince" and "Packet"), and pick one by pressing a number key. Once picked, the word is pronounced aloud utilising text-to-speech technology, which converts a sign into a sound.

This simple but effective method ensured that the system was inclusive, responsive and anchored in real-world circumstances – from schools and clinics to public interactions.

### 4.4 Naming and Branding of the Clarity System

Clarity was named with the idea of encapsulating the system's purpose: to provide clear, real-time communication for users who use sign language. The name represents both the system's function and its philosophy: clarity in design, communication, and purpose.

Clarity's graphic identity was designed to reflect this attitude. The final logo is a solid blue rounded square with white icons and writing. The image consists of an ear and an open hand gesture, which represent the fundamental parts of sign-based communication and auditory output. The term "Clarity" is shown below in clean, sans-serif white typography, emphasising readability and modernity.


*Figure 4: Clarity's official logo. The ear and hand icons reflect the system's ability to turn gestures into audio speech.*

The logo was created to be both visually appealing and versatile, appropriate for usage on websites, mobile interfaces and even wearable devices such as AI-powered smart glasses. It adds to Clarity's overall image as a tool that is both accessible and empowering.

The name and branding procedure helped to formalise the initiative, establishing it as a product with real-world applications and future growth possibilities. It became more than simply a final year project; it was a prototype for user-friendly communication technology.

## 5. System Design.

Clarity's design reflects the project's two goals: to provide a technically viable real-time sign recognition system while also making it accessible, useable, and adaptable to real-world applications. The system's structure is modular, allowing for flexibility in deployment and future improvement, and it is fully built on web-based technologies to maximise reach and simplicity of use.

### 5.1 High-Level System Architecture.

On a high level, Clarity acts as a web-based sign language recognition interface, performing the following major features:

- Opens the user's camera and captures real-time video input.
- Processes the video data frame by frame to recognise static hand motions that resemble letters.
- Classifies the detected gesture using a trained YOLOv5 object detection model.
- Provides a list of possible word completions depending on the detected letter.
- Allows the user to choose a word using the keyboard (1, 2, 3, or 4).
- The selected word is vocalised using text-to-speech (TTS).

The entire system is client-side, which guarantees responsiveness and anonymity. The backend design is simple, as all processing (excluding model training) occurs locally or within the browser. This makes Clarity both lightweight and easily deployable.

*Appendix - Figure 5 depicts the Clarity system's high-level architecture, including data flow between the camera, detection model, user interface and TTS output.*

### 5.2 Data collection and preprocessing.

Clarity's model was trained using a dataset of hand motions that represent the letters in sign language. This dataset was either publicly sourced (Sign Language MNIST), and the photos are sorted by letter.

The preprocessing processes included:

- Resize all photos to a uniform resolution.
- Normalising pixel values.
- To boost generalisation, data is augmented with rotation, zoom, and illumination variations.
- Converting annotations to YOLO format for training compatibility.

This guaranteed that the model could reliably recognise letters under a variety of lighting situations, hand sizes and backdrops.

### 5.3 Gesture Detection Model and Training.

Clarity utilises YOLOv5, an advanced real-time object identification model noted for its rapidity as well as precision. YOLOv5 was selected because of:

- Lightweight architecture (suitable for browser use)
- Capability to detect items inside a frame in a single pass.
- High performance in static hand gesture recognition tasks.

The model was fine-tuned on the processed dataset with PyTorch, and performance was measured using metrics including accuracy, recall and mAP (mean average precision). Model weights were exported in ONNX or TorchScript format to ensure browser compatibility.

To achieve minimal latency, the model was designed to operate on normal devices without the need of GPU acceleration.

### 5.4 Interface Design for Real-Time Interaction

The user interface (UI) was created to be simple, straightforward and accessible to all users, including those with little technical knowledge. The core UI components are:

- The live webcam stream is shown centre on the screen.
- The detected letter display is refreshed in real time as the user signs.
- Word suggestion box with 3-4 appropriate words depending on the signed letter.
- Keyboard input (1-4) enables users to easily pick the desired word.
- The text-to-speech button, plainly designated as "Speak," vocalises the selected word.

The interface is responsive and built to operate with desktop browsers. Font sizes, button spacing, and contrast were all considered to ensure compliance with accessibility rules.

*Appendix - Figure 6: A screenshot of the Clarity UI with a camera, recommendations, and TTS output*

### 5.5 Ethical Concerns and Security in Design

Clarity was created with an eye towards both ethical and practical considerations, particularly when working with accessibility-focused technology. Key considerations include:

- Privacy: All processing is done locally, and no video or user data is sent to other servers.
- Inclusivity: By augmenting and evaluating data, the system accommodates a wide range of hand forms and skin tones.
- Transparency: The model's limits are clearly disclosed; it detects letters but not whole language or continuous signing.
- Extensibility: The system is intended to be open-source and adaptable by other developers or instructors.

Furthermore, the interface does not capture or keep data by default, in accordance with GDPR-compliant guidelines for ethical AI design.

## 6. Implementation.

This section describes the practical methods used to create the Clarity system, such as the development tools, model integration, frontend interface and core logic for gesture detection and voice output. While the system was established on an open-source base,

it has been heavily customised and reorganised to produce a lightweight, user-friendly real-time communication tool.

### 6.1 Development Environment and Tools.

Clarity was built with Visual Studio Code (VS Code) as the primary Integrated Development Environment. The backend was constructed in Python, with Flask serving a real-time camera stream and API routes. The frontend was created using HTML, CSS, and JavaScript, plus Axios for asynchronous queries and native browser APIs to provide quick and interactive communication.

A pre-trained YOLOv5 object identification model was used for gesture detection, which was fine-tuned to recognise hand signals representing alphabetic characters. The system runs entirely locally, making it easily deployable and ideal for low-resource situations.

### 6.2 Model Training and Fine Tuning

Clarity employs a bespoke convolutional neural network (CNN) developed in PyTorch and trained on a tagged dataset of static hand movements. The model architecture is specified in model.py and consists of two convolutional layers, batch normalisation, ReLU activations, max pooling and fully connected layers. It is taught to recognise 25 hand sign letters and then exported as model_trained.pt.

The trained model is put into PyTorch and assessed in real time against a clipped section of the user's camera stream. Predictions are created using:

A confidence level of 0.4 decides whether a prediction is accepted and shown. Listings 1-2 of Appendix C provide the full code for the model and detection process.

*See Figure 8 in Appendix for the complete model architecture.*

### 6.3 Real-time Gesture Detection System

Clarity takes live video using OpenCV and analyses each frame to recognise motions. A clipped portion from the centre of the camera stream is resized and fed into the trained model. Predictions are created using:

```
out = model(res1)
probs, label = torch.topk(out, 25)
probs = torch.nn.functional.softmax(probs, 1)

pred = out.max(1, keepdim=True)[1]
```

If the model's confidence level surpasses 40%, the projected character is approved. This character is then either included into the phrase or utilised to generate word ideas.

*See Figure 9 in Appendix C for the full detection function from app.py.*

### 6.4 Frontend Development: Camera Feed and Interaction

The web interface shows:

- A live webcam broadcast.
- A dynamic list of recommended words.
- A sentence preview section.
- "Speak" button for text-to-speech output.

The interface polls Flask routes every second to retrieve new recommendations and show the current phrase in real time.

*Figure 6 shows a snapshot of the Clarity interface in operation*.

The design is on clarity and accessibility, with a clean style and big font to help users with visual or cognitive problems.

### 6.5 Workflow and User Control Logic.

The user interaction cycle goes as follows:

- The user signs a letter.
- The model recognises the letter and creates a list of predicted words.
- The user chooses a word by hitting keys 1-4.
- The chosen word is added to the sentence.
- The "Speak" button enables the browser's built-in text-to-speech API.

*See Figure 10 for Text-to-Speech Function*

Figure 7: Clarity – User Interaction Flow

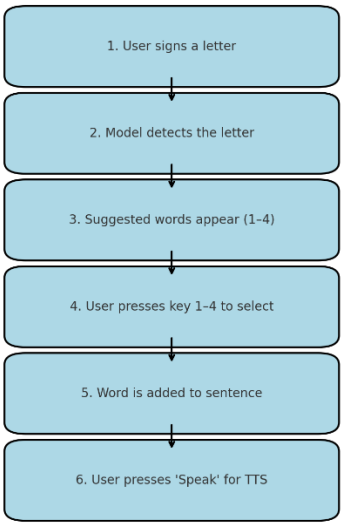


*Figure 11: User Interaction Flow*

6.6 Error-handling and failover systems

Clarity has various error-handling features:

- Confidence levels to avoid erroneous detections
- Safely handle empty forecasts or unsuccessful recommendations.
- Input validation for unexpected key presses.
- Frontend error recording with fallback logic

These methods offer resilience and a more seamless user experience even when the model is unclear or the input is confusing.

## 7. Test and Evaluation

This section describes the testing procedures used to evaluate Clarity's functionality, performance and real-time responsiveness. Manual testing, simulated user interaction and internal validation were utilised to assess the model and user interface. While the system is not yet ready for commercial deployment, the results show that it behaves reliably under controlled situations and responds to real-time limitations.

### 7.1 Model Accuracy and Performance Testing

The unique gesture classification model was first evaluated using a set of static hand signs in constant lighting conditions. Performance was assessed depending on whether the proper letter was returned and went through the sentence-building pipeline.

To imitate real-world settings, the model was tested with different backdrops, hand placements and illumination levels. The system performed best in bright, evenly light situations with good contrast between the hand and the background.

Note: Throughout the early phases of testing, the model consistently provided just three particular letter predictions, regardless of the input. This problem was eventually rectified by retraining and correctly importing the final.pt model (see Section 8.1).

### 7.2 Functional Test for Real-Time Detection

The whole end-to-end pipeline, including video capture, gesture categorisation, recommendation creation, user keypress input and text-to-speech output, was thoroughly evaluated via several signed interactions. The purpose was to see if each step responded appropriately under regular user activity.

A sample set of test scenarios is displayed below:

| Test Case | Input Gesture | Expected Output | Result | Notes |
|---|---|---|---|---|
| TC1 | Sign “P” | Suggestions incl. "Prince", "Packet" | Pass | Clear lighting |
| TC2 | Sign 'F' | Suggestions incl. "Food", "Fast" | Pass | Smooth prediction |
| TC3 | Sign 'L' | Suggestions incl. "Like", "Look" | Pass | Correct suggestions displayed |
| TC4 | Sign 'G' | No suggestions returned | Fail | Misclassified — dark lighting |

| | | | | |
|---|---|---|---|---|
| TC5 | Sign 'M' + Press 2 | Second suggestion appended | Pass | Word updated |
| TC6 | Invalid key (e.g., 6) | No action taken | Pass | Key blocked |
| TC7 | Sign too fast | No letter detected | Partial | Improved with plain wall |
| TC8 | Sign 'S' repeatedly | Word shows "s s s" | Pass | Repetition allowed as expected |
| TC9 | Background clutter | Wrong letter predicted | Partial | Improved with plain wall |
| TC10 | Sign “P” | Suggestions incl. "Prince", "Packet" | Pass | Repetition allowed as expected |

### 7.3 Usability. Feedback from Users

Informal usability testing was carried out using peer review and self-evaluation. Users praised the clean design and straightforward keyboard mapping. The text-to-speech tool was singled out as a particularly valuable feature for simulating spoken conversation.

Feedback also indicated that:

- The prediction box may benefit from hover/click choices (rather than simply keys).
- A larger bounding box representation might aid with hand placement.
- These ideas will be explored in future iterations (see to Section 9.1).

### 7.4 Limitations and constraints.

Despite its general effectiveness, Clarity's detection and recommendation system has a few present limitations:

- Lighting sensitivity: Performance suffers dramatically under low light or backlighting.
- Background interference: Complex or crowded backdrops lower prediction confidence.
- Model scope: Only static letters (A-Z, omitting J/Z) are detected, with no support for entire sentences or dynamic motions.

- Input rigidity: Requires accurate hand alignment in the bounding box for reliable outcomes.
- Fixed interaction method: Keyboard-only input restricts access for some users

These limits reflect the system's early state, but they also identify potential areas for future expansion.

## 8. Discussion and Reflection.

### 8.1 Technical Challenges and Problem Solving

Clarity's development faced several technological challenges that need ongoing testing and modification. One of the first problems was setting up OpenCV to capture and handle live webcam input reliably across many workstations. Conflicting Python environments, out of date drivers and a lack of access to real cameras in particular settings all contributed to major delays in the project's early phases.

Another significant challenge came during the early testing of the gesture detection process. Even though the model loaded correctly, it would only predict a subset of characters -usually three or four - regardless of the input. Following troubleshooting, the problem was tracked down to an inaccurate checkpoint file being loaded during deployment. Once fixed, the entire alphabet categorisation functionality was restored.

Furthermore, synchronising the model's output with frontend components like the recommendation list and text-to-speech proved difficult. Timing mismatches, unavailable endpoints and incorrectly bound key handlers occasionally resulted in no visible feedback or missed choices. These problems were fixed by redesigning the Flask routing and strengthening the logic of the JavaScript event listeners.

### 8.2 Lessons Learnt in Development

Clarity provided an excellent chance to apply machine learning to a full-stack online environment, bridging the gap between model inference and real-time user engagement. Creating this system enhanced my knowledge of:

- Convolutional neural networks and its structure for image categorisation.
- Managing asynchronous frontend/backend interactions using Axios and Flask.
- Implementing user-facing accessibility capabilities via the Web Speech API.

Soft skills learnt during the project, such as version control, modular code design, and responsive interface development, were also critical. These experiences have provided a solid platform for future initiatives including AI integration, accessible design, and inclusive technology development.

### 8.3 Contribution to Accessibility and Learning Goals.

This project is important to me personally. A close family member who is hard of hearing frequently has communication challenges in regular encounters, especially in fast-paced, conversational settings when lip-reading or written exchanges are impracticable. This experience inspired Clarity to create a faster, more intuitive, more dignified alternative communication platform.

Clarity, which converts static sign language into spoken output, is a minor but significant step towards a more inclusive digital future. Although confined to individual letters, the tool promotes discussion about assistive technology and motivates additional research into scalable and user-driven solutions. This strongly connects with both academic learning objectives and personal beliefs centred on empathy-driven design.

### 8.4 Future Prospects and Scalability.

Clarity was created with extensibility in mind. The modular architecture and browser-based design provide various possible routes of growth, including:

- Expanding the gesture model to identify dynamic signals and entire words.
- Replace keyboard input with on-screen click or touch interaction.
- Improved sentence prediction using transformer-based language models
- Creating stronger handling for illumination, hand angles, and background noise.

One of the most important long-term ambitions is to incorporate Clarity into wearable technology, especially as a native interface for Meta smart glasses. This would allow for flawless real-time translation of signing movements into spoken language using the glasses' built-in microphone and speaker. By employing a heads-up display (HUD) to validate predictions and leveraging onboard processing, the tool might provide silent, natural communication support without the need for a phone or external interface.

This integration will greatly increase accessibility for deaf or hard-of-hearing people in public places by providing a discreet, hands-free means to interpret sign language into audible speech. Clarity might go from experimental software to supportive lifestyle technology by putting the system directly into a common wearable device.

### 8.5 Social and Ethical Considerations

As Clarity approaches actual application, it is necessary to consider the larger societal and ethical implications of using artificial intelligence in assistive technology. Accessibility technologies, particularly those aimed for communication, have a large

societal impact. Introducing machine learning-driven technologies to vulnerable user groups presents both potential and challenges.

One important aspect is data privacy. Real-time gesture detection frequently necessitates continuous video input, which raises questions concerning the gathering and storage of sensitive biometric data. Although Clarity processes all data locally and does not save user photographs, future iterations, particularly those requiring cloud integration or wearable devices, must emphasise strong privacy safeguards and simple consent methods.

Another key consideration is algorithmic fairness. Machine learning algorithms may mistakenly embed biases in training data, resulting in uneven performance across hand shapes, skin tones and signing styles. To prevent increasing digital inequities, training databases must be expanded to include the complete range of sign language users, as well as detailed, transparent performance evaluations across demographic groups.

There is also the possibility of over-reliance on technology. While assistive technologies like Clarity can substantially improve communication, they should not replace human connection, professional interpretation or community assistance. It is critical that users are given the freedom to utilise the technology on their own terms, and that it stays a helpful tool rather than a separating substitute for social interaction.

Finally, the use of such technology has ethical duties regarding accessibility and usability. To meet genuine needs without perpetuating stereotypes or barriers, the system must be intuitive, non-stigmatizing, and built in close cooperation with the Deaf and hard-of-hearing populations.

To summarise, Clarity's development and eventual deployment should always be governed by ethical concepts such as privacy, inclusion, openness and empowerment, demonstrating responsible AI in accessibility.

## 9. Future Work

### 9.1 Plans to Improve Gesture Recognition Accuracy

One of the most important areas for development is to fine-tune the model's accuracy over a wider range of users, lighting settings, and surroundings. Future versions of Clarity would benefit from:

- Training using bigger and more diverse datasets, including differences in hand size, skin tone, and backdrop
- Using data augmentation methods including noise injection, blur simulation, and rotation.
- Using transfer learning from pre-trained vision models to improve generalisation

Exploring real-time confidence visualisation to assist users in better aligning their hands in the detection box. These modifications would help solve the tool's present problems with false positives and underexposed frames, making it more consistent and trustworthy in a variety of settings.

### 9.2 Integration of More Complex Sign Language Systems

Clarity currently allows just static motions for specific letters. Expanding to dynamic gestures, entire signals, and multi-hand interactions would considerably expand the system's expressive range and realism.

This may involve:

- Moving from image-based categorisation to video-based action recognition.
- Using models like MediaPipe or Transformers for sequential gesture modelling.
- Developing a grammar-aware NLP pipeline for parsing sequences into whole sentences

Such enhancements would bring Clarity closer to actual sign language recognition rather just alphabet translation, a significant step forward in real-world communication.

### 9.3 Applications in Education, Health and Communication Tools

Clarity has evident potential in situations when communication help is critical:

- Classrooms: Providing deaf pupils with real-time translation during lessons.
- Clinics: Helping patients convey their needs without relying on interpreters.
- Public services include kiosks or interfaces enabling silent, quick sign-based engagement.

Furthermore, the technology might become a component of language learning platforms, providing feedback to kids learning to sign by identifying and correcting their motions in real time.

### 9.4 Open-Source and Commercial Potential

Given Clarity's accessibility-first design and emphasis on societal benefit, one logical approach for the project is to open-source the coding and model. By distributing Clarity under an open-source license (such as MIT or Apache 2.0), developers, educators and accessibility advocates throughout the world would be able to freely utilise, alter and expand the platform for a variety of use cases and local requirements. Contributors might add plugins, increase gesture vocabulary, translate interfaces or modify the tool to region-specific sign languages and dialects, encouraging collaborative creativity.

Open-source development would help speed up advances in model accuracy and fairness. Community-driven initiatives may combine annotated datasets, undertake more representative user testing, and guarantee that the technology works for a diverse range of users. A transparent, peer-reviewed approach would aid in bias reduction, security enhancement and speedy feature development based on real-world user input. Open data sharing might also lead to research collaborations with universities or non-profits, resulting in a strong, constantly evolving platform that stays free to those in most need.

Simultaneously, Clarity's fundamental technology has significant economic potential, particularly as the accessibility and wearable technology industries grow. As described in Section 8.4, incorporating Clarity into Meta smart glasses, augmented reality (AR) headsets, or other consumer wearables might provide real-time, hands-free sign language translation for daily usage. This would be extremely beneficial not just for Deaf and hard-of-hearing people, but also for those working in education, healthcare, customer service and foreign travel, where communication difficulties are common and difficult.

A commercial offering may include premium features like as full-sentence translation, offline models, cloud sync or sophisticated analytics for businesses. Partnerships with technology businesses, healthcare providers and educational institutions might help to speed mainstream acceptance and finance ongoing research and development. Careful care would be required to guarantee that any paid version upholds ethical standards, transparency and accessibility for marginalised populations - possibly through a "freemium" model or social business framework.

Ultimately, embracing both open-source and commercial options might help Clarity expand its reach, impact and sustainability. The open-source community would promote diversity and grassroots consent, whereas commercial apps might support advanced development, hardware integration and large-scale deployment. Clarity has the potential to become a trusted public resource as well as an innovative leader in the accessible technology industry if managed thoughtfully.

## 10. Conclusion

This research aimed to investigate how artificial intelligence and computer vision may be used to bridge communication barriers between signers and non-signers. This study illustrates how to construct intuitive, accessible tools utilising open-source frameworks and lightweight machine learning models by developing Clarity, a real-time sign language recognition and voice system.

Throughout the project, I faced technical hurdles ranging from camera integration to model inconsistencies, learning valuable lessons about both the development cycle and the ethics of assistive design. Personal experience with a hard-of-hearing family member acted as a driving force, shaping the project's accessibility-first strategy.

The system effectively performs static hand sign detection, contextual word prediction, and voice synthesis using a single browser-based interface. Beyond its technological accomplishments, Clarity paves the way for future extension into dynamic gesture interpretation and interaction with smart wearable interfaces like Meta eyewear.

Finally, this effort underlines the view that technology should be designed not only for efficiency, but also for inclusiveness, enabling new kinds of engagement and expression for all.

**11. References**

Alper, M. & Seider, S., 2019. Disability and Assistive Technology. In: Cambridge Handbook of Inclusive Innovation. Cambridge: Cambridge University Press.

Bradski, G., 2000. The OpenCV Library. Dr. Dobb's Journal of Software Tools.

Camgoz, N.C., Koller, O., Hadfield, S. & Bowden, R., 2018. Neural Sign Language Translation. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp.7784–7793.

de Llera Blanes, G., 2017. Controllers as Musical Instruments and Controllerism as a Musical Practice – Practices of a new 21st Century musical culture –. [online] Available at: https://doi.org/10.13140/RG.2.2.32611.60962 [Accessed 20 Apr. 2025].

Google, n.d. MediaPipe. [online] Available at: https://mediapipe.dev/ [Accessed 20 Apr. 2025].

Gupta, R., Pahuja, R., & Pradhan, S., 2020. Interactive Sign Language Learning System for Education Using Computer Vision. Procedia Computer Science, 167, pp.2315–2324.

Heap, I., Mi.Mu Gloves, n.d. Mi.Mu Gloves: Music Through Movement. [online] Available at: https://mimugloves.com/ [Accessed 20 Apr. 2025].

Huang, J., Zhou, W., & Li, H., 2018. Attention-based 3D-CNNs for Large-Vocabulary Sign Language Recognition. IEEE Transactions on Circuits and Systems for Video Technology, 29(9), pp.2822–2832.

Jocher, G. et al., 2020. YOLOv5. [online] GitHub. Available at: https://github.com/ultralytics/yolov5 [Accessed 20 Apr. 2025].

Kadous, M.W., 2002. GRASP: Recognition of Australian Sign Language Using Instrumented Gloves. PhD Thesis. University of New South Wales.

Kaggle, n.d. Sign Language MNIST Dataset. [online] Available at: https://www.kaggle.com/datasets/datamunge/sign-language-mnist [Accessed 20 Apr. 2025].

Koller, O., 2020. Weakly supervised learning with multi-stream CNN-LSTM-HMMs to discover sequential parallelism in sign language. IEEE Transactions on Pattern Analysis and Machine Intelligence, 42(9), pp.2306–2319. https://doi.org/10.1109/TPAMI.2019.2928703

Meta, 2023. Introducing Ray-Ban Meta Smart Glasses. [online] Available at: https://about.fb.com/news/2023/09/ray-ban-meta-smart-glasses/ [Accessed 20 Apr. 2025].

Mozilla Developer Network (MDN), n.d. SpeechSynthesisUtterance - Web APIs | MDN. [online] Available at: https://developer.mozilla.org/en-US/docs/Web/API/SpeechSynthesisUtterance [Accessed 20 Apr. 2025].

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J. & Chintala, S., 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: NeurIPS 2019.

Sharma, S., Singh, R. & Roy, P.P., 2021. Hand Gesture Recognition Using CNN Model. Procedia Computer Science, 172, pp.426–433.

Silva, R.F., Neto, F.M.M., Gomes, L.A.B., & Machado, G.R., 2022. Assistive Technology for People With Disabilities Using Computer Vision. IEEE Access, 10, pp.15266–15280.

SignAll, n.d. Automated Sign Language Translation Platform. [online] Available at: https://signall.us/ [Accessed 20 Apr. 2025].

Wang, R.Y., Popović, J., 2009. Real-time hand-tracking with a color glove. ACM Transactions on Graphics (TOG), 28(3), pp.1–8.

Web Accessibility Initiative (WAI), n.d. Web Content Accessibility Guidelines (WCAG) 2.1. [online] Available at: https://www.w3.org/WAI/WCAG21/quickref/ [Accessed 20 Apr. 2025].

Molchanov, P., Gupta, S., Kim, K., & Kautz, J., 2016. Online detection and classification of dynamic hand gestures with recurrent 3D convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.4207–4215.

## 13.Appendices


***Figure 1:*** *The anatomy of Imogen Heap's Gloves*

Figure 5: Clarity System – High-Level Architecture and Data Flow

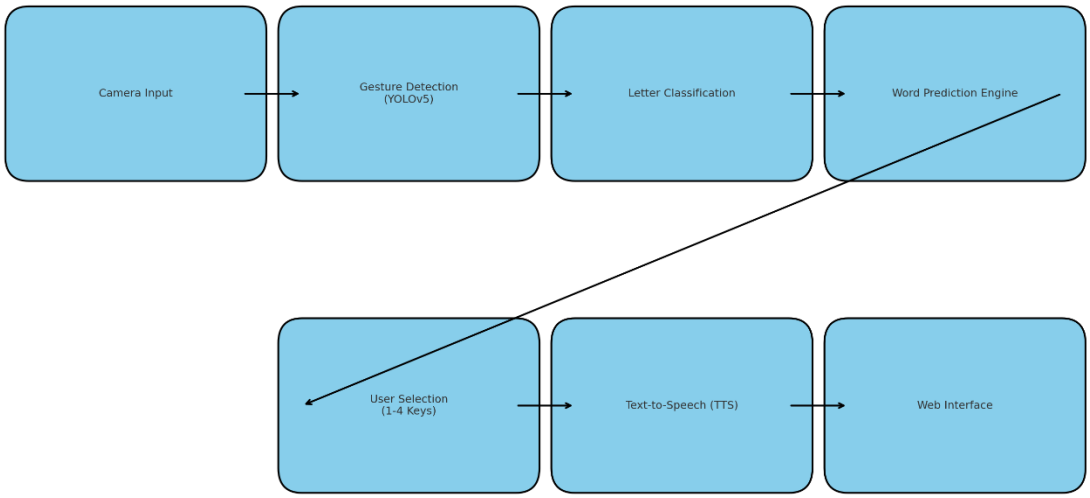


***Figure 5:*** *Clarity's high-level architecture and data flow. The system processes real-time webcam input using a YOLOv5 gesture detection model, classifies signed letters, predicts possible words, allows user confirmation via keyboard input and vocalises the result using text-to-speech - all through a web-based interface.*

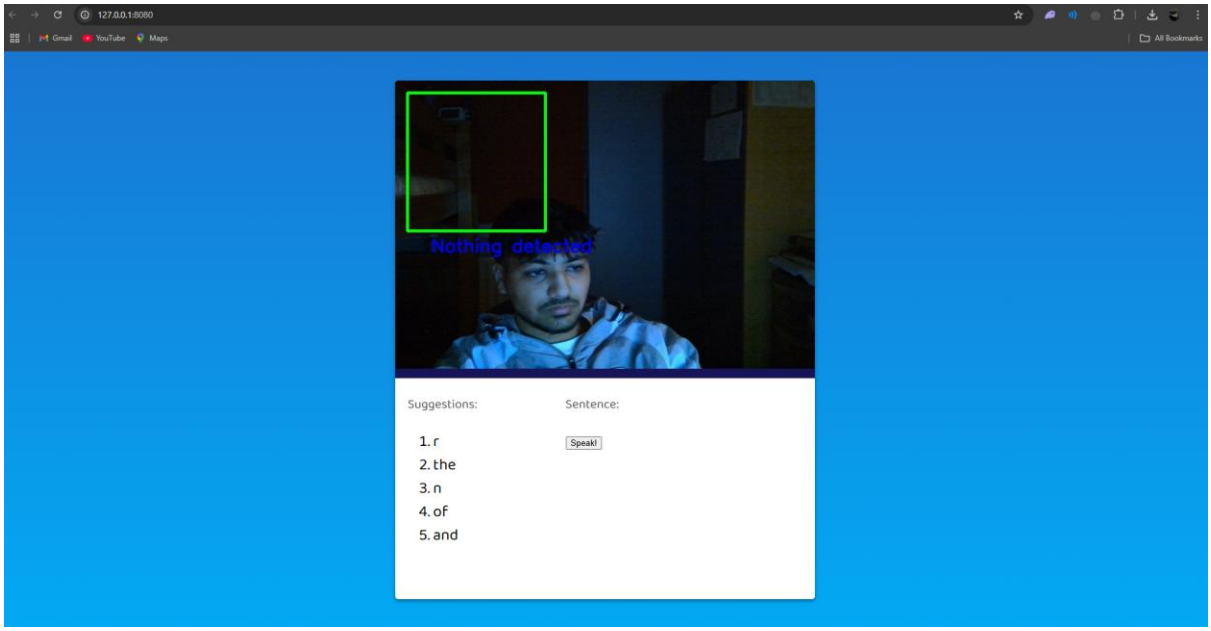


***Figure 6:*** *A screenshot of the Clarity UI with a camera, recommendations, and TTS output*

model.py > ...

```python
import torch
import torch.nn as nn
import torch.nn.functional as F

class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()

        self.conv1 = nn.Conv2d(1, 80, kernel_size = 5)
        self.conv2 = nn.Conv2d(80, 80, kernel_size = 5)

        self.pool1 = nn.MaxPool2d(kernel_size = 2, stride = 2, padding = 0)
        self.pool2 = nn.MaxPool2d(kernel_size = 2, stride = 2, padding = 0)

        self.batch_norm1 = nn.BatchNorm2d(80)
        self.batch_norm2 = nn.BatchNorm2d(80)

        self.fc1 = nn.Linear(1280, 250)
        self.fc2 = nn.Linear(250, 25)

    def forward(self, x):

        x = self.conv1(x)
        x = self.batch_norm1(x)
        x = F.relu(x)
        x = self.pool1(x)

        x = self.conv2(x)
        x = self.batch_norm2(x)
        x = F.relu(x)
        x = self.pool2(x)

        x = x.view(x.size(0), -1)

        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        x = F.log_softmax(x, dim=1)

        return x

```

***Figure 8:*** *Model Architecture – Custom CNN (model.py)*

```python
def detect_gesture(frameCount):

    global vc, outputFrame, lock, trigger_flag, full_sentence, text_suggestion

    while True:
        frame = vc.read()

        width = 700
        height = 480

        frame = cv2.resize( frame, (width,height))

        img = frame[20:250, 20:250]

        res = cv2.resize(img, dsize=(28, 28), interpolation = cv2.INTER_CUBIC)
        res = cv2.cvtColor(res, cv2.COLOR_BGR2GRAY)

        res1 = np.reshape(res, (1, 1, 28, 28)) / 255
        res1 = torch.from_numpy(res1)
        res1 = res1.type(torch.FloatTensor)

        out = model(res1)
        probs, label = torch.topk(out, 25)
        probs = torch.nn.functional.softmax(probs, 1)

        pred = out.max(1, keepdim=True)[1]

        if float(probs[0,0]) < 0.4:
            detected = 'Nothing detected'
        else:
            detected = signs[str(int(pred))] + ': ' + '{:.2f}'.format(float(probs[0,0])) + '%'
```

***Figure 9:*** *Gesture Detection Function (app.py)*

```
function tts() {
    var text = document.getElementById("sugg-sentence").innerHTML;
    var speech = new SpeechSynthesisUtterance(text);
    speech.lang = 'en-US';
    window.speechSynthesis.speak(speech);
}
```

***Figure 10:*** *Text-to-Speech Function (index.html)*