Sign Vision: Sign Language and Symbol Interpretation

Submitted by	**Waseem Akram**
	M. Adil Bari
	M. Abdul Sami Jamil
Roll. No	**0077-BSCS-20**
	0092-BSCS-20
	0098-BSCS-20
Session	**2020 - 2024**
Supervised by	**Dr. Ayesha Atta**
	Assistant Professor

BS(HONS)
IN
COMPUTER SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

GC UNIVERSITY LAHORE

Sign Vision: Sign Language and Symbol Interpretation

Submitted to GC University Lahore in partial fulfillment
of the requirements for the award of degree of

BS(HONS)
IN
COMPUTER SCIENCE

Submitted by	Waseem Akram
	M. Adil Bari
	M. Abdul Sami Jamil
Roll. No	0077-BSCS-20
	0092-BSCS-20
	0098-BSCS-20
Session	2020 - 2024
Supervised by	Dr. Ayesha Atta
	Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE

GC UNIVERSITY LAHORE

Declaration

We, **Waseem Akram (0077-BSCS-20)**, **M. Adil Bari (0092-BSCS-20)**, and **M. Abdul Sami Jamil (0098-BSCS-20)** students of **BS(Hons)** in the subject of **Computer Science** session **2020-2024**, hereby declare that the matter printed in this thesis titled, **Sign Vision: Sign Language and Symbol Interpretation** is our own work and has not been printed, published and submitted as research work, thesis or publication in any form in any University, Research Institution etc in Pakistan or abroad.

Date: July 10, 2024

Signatures of Deponent

Research Completion Certificate

It is certified that the research work contained in this thesis titled **Sign Vision: Sign Language and Symbol Interpretation** has been carried out by **Waseem Akram, Muhammad Adil Bari and Muhammad Abdul Sami Jamil** Roll. No **0077-BSCS-20, 0092-BSCS-20 and 0098-BSCS-20** under my supervision.

Dr. Ayesha Atta
Assistant Professor

Date: July 10, 2024

Submitted Through

Prof. Dr. Muhammad Waqas Anwar
Chairperson
Department of Computer Science
GC University Lahore

Controller of Examination

GC University Lahore

Acknowledgements

I am grateful to the Almighty **Allah**, Who blessed me with health, wisdom, knowledge, thoughts and opportunity to make some contribution in the form of present effort. I offer my humblest thanks from the deepest core of my heart to the **Holy Prophet Muhammad (Peace be upon him)**, the most perfect and excelled among and ever born on the surface of earth.

The research work embodied in this dissertation was accomplished under the able guidance and affectionate supervision of **Dr. Ayesha Atta**, Assistant Professor, GC University, Lahore. I will always remember his moral encouragement, skillful guidance, positive criticism and valuable advice throughout the course of my study.

I also express my feelings of love and respect for my beloved parents that are the real asset of my life and gave me confidence and my friends also deserve my thanks for their loving encouragement and prayers for my success.

Dedication

This project is for the deaf and hard of hearing people who have motivated me to close the communication gab. This project is dedicated to their spirit and tenacity. I also dedicate this work to my family and my honorable teacher, Dr. Ayesha Atta, who guided me to make this vision real. My hope is that through SignVision the society will be a better place for everyone to live in.

Abstract

SignVision is a mobile application designed to bridge the communication gap for individuals who rely on sign language. Developed using Flutter and advanced machine learning techniques, SignVision provides level-wise lessons to teach users how to perform specific signs, followed by a real-time sign detection feature for practice. This self-guided learning approach empowers users to learn sign language independently, without external assistance. Additionally, the app includes a module for learning general symbols, offering users the ability to explore and understand various symbols and their meanings. By facilitating effective sign language learning and symbol recognition, SignVision addresses the critical need for accessible communication tools, promoting inclusivity and enhancing the quality of life for sign language users.

Contents

Declaration i

Research Completion Certificate ii

Acknowledgements iii

Dedication iv

Abstract v

Contents vi

List of Figures x

1 Introduction 1
1.1 Introduction 1
1.1.1 Motivation for Developing SignVision 2
1.1.2 Project Objectives 2
1.1.2.1 Empowerment through Education: 2
1.1.2.2 Real-time Feedback and Practice: 2
1.1.2.3 Promoting Safety through Symbol Recognition: . . 2
1.1.3 Addressing a Critical Need 3
1.1.4 Brief 3
1.1.5 Objectives and Importance 3
1.1.5.1 Overcoming Learning Barriers: 3
1.1.5.2 Enhancing Communication: 4
1.1.5.3 Promoting Employment Opportunities: 4
1.1.5.4 Safety Awareness through Traffic Symbol Education: 4
1.1.6 What might a mobile application with a machine learning model accomplish? 4
1.1.6.1 Benefits to Individuals 5
1.1.7 Problem Definition 6

1.1.8 Scope . . . 7
1.1.9 Project Features . . . 8
1.2 Problem Statement . . . 9
1.2.1 Module 1: Admin Server . . . 10
1.2.2 Module 2: User Module . . . 10
1.3 Literature Review . . . 10
1.3.1 Analysis from Literature Review . . . 12
1.3.2 Limitations of Existing Work: . . . 12
1.3.3 Overcoming Research Gaps in Our Project: . . . 13
1.4 Methodology and Software Life-Cycle . . . 14
1.4.1 Rationale Behind Selected Methodology . . . 15

2 Requirement Specification 17
2.1 Functional Requirements . . . 17
2.1.1 Sign Up: . . . 17
2.1.2 Sign In: . . . 17
2.1.3 Forgot Password: . . . 18
2.1.4 Real-Time Sign Language Detection & Learning: . . . 18
2.1.5 Learning General Symbols: . . . 18
2.1.6 Preprocessing: . . . 18
2.1.7 User Data Management: . . . 19
2.2 Non-functional Requirements . . . 19
2.2.1 Reliability: . . . 19
2.2.2 Scalability: . . . 20
2.2.3 Performance: . . . 20
2.2.4 Cost: . . . 20
2.2.5 Security: . . . 20
2.2.6 Compatibility: . . . 20

3 Project Design 21
3.1 Use Case Diagram . . . 21
3.1.1 Use Cases . . . 23
3.1.1.1 Use Case For SignUp . . . 23
3.1.1.2 Use Case For Send OTP . . . 24
3.1.1.3 Use Case For Verification of OTP . . . 25
3.1.1.4 Use Case For SignIn . . . 26
3.1.1.5 Use Case For Authenticate User . . . 27
3.1.1.6 Use Case For Forgot Password . . . 28
3.1.1.7 Use Case For Reset Password . . . 29
3.1.1.8 Use Case For Detect Gesture . . . 30
3.1.1.9 Use Case For Capture Video . . . 31
3.1.1.10 Use Case For Display Meaning . . . 32
3.1.1.11 Use Case For Learn Sign Language . . . 33
3.1.1.12 Use Case For Get Lesson . . . 34

3.1.1.13 Use Case For Learn General Symbols 35
3.1.1.14 Use Case For Search Symbols 36
3.2 Class Diagram 37
3.3 Sequence Diagrams 38
3.3.1 SignUp 38
3.3.2 SignIn 39
3.3.3 Forgot Password 40
3.3.4 Gesture Detection 41
3.4 Data Flow Diagrams 42
3.4.1 Level 0 Data Flow Diagram 42
3.4.2 Level 1 Data Flow Diagram 42
3.5 Activity Diagram 43

4 Implementation and Evaluation **44**
4.1 Development Stages 44
4.1.1 Definition: 44
4.1.2 Initiation: 44
4.1.3 Overview: 45
4.1.4 Technical Approach of the Project: 45
4.1.5 Toots and Technologies: 46
4.1.6 Execution: 46
4.1.7 Monitoring and Control: 46
4.1.8 Closure: 47
4.2 System Integration 47
4.2.1 User Interface 47
4.2.2 Admin Interface: 48
4.2.3 HardWare Interface: 48
4.3 User Interface 49
4.3.1 Welcome Page 49
4.3.2 SignUp Page 50
4.3.3 Verify The Account 51
4.3.4 Verification Complete 52
4.3.5 SignIn Page 53
4.3.6 Home Screen 54
4.3.7 Level-Wise Sign Language Detection and Learning 55
4.3.8 Learning and Detection 56
4.3.9 General Important Symbols Learning 58
4.3.10 About Us 59
4.4 Unit Testing 60
4.4.1 Test Case For SignUp: 60
4.4.2 Test Case For SignIn: 61
4.4.3 Test Case For Forgot Password Functionality: 62
4.4.4 Test Case For Real-Time Hand Gesture Recognition: 63
4.4.5 Test Case For Invalid Hand Gesture Detection: 64

4.4.6 Test Case For Accurate Result Display Verification: 65
4.4.7 Test Case For Live Video Hand Gesture Detection Performance: 66
4.4.8 Test Case For Efficient Symbol Learning: 67
4.5 BlackBox Testing 68
4.6 Path Testing 68
4.7 Compatibility Testing 68
4.8 GUI Testing 69
4.9 Integration Testing 69
4.10 System Testing 69

5 Conclusion & Future Work **70**
5.1 Conclusion 70
5.2 Future Work 71

Bibliography **72**

List of Figures

1.1 Agile Development Cycle 16

3.1 Use Case Diagram 22
3.2 Class Diagram 37
3.3 SignUp Sequence Diagram 38
3.4 SignIn Sequence Diagram 39
3.5 Forgot Password Sequence Diagram 40
3.6 Sequence Diagram of App 41
3.7 Context Level Data Flow Diagram 42
3.8 Level 1 Data Flow Diagram 42
3.9 Activity Diagram of App 43

4.1 Welcome Page 49
4.2 SignUp Page 50
4.3 Verification Page 51
4.4 Verification Complete 52
4.5 SignIn Page 53
4.6 Home Screen 54
4.7 Level Page 55
4.8 Learning 56
4.9 Detection 57
4.10 General Symbols 58
4.11 About Us Page 59

Chapter 1

Introduction

1.1 Introduction

The creation of a sign language detecting system is crucial because it helps users to avoid the challenges that are characteristic of sign language learning, namely the problems of accuracy and time consumption. This technology, in the way it facilitates efficient communication, improves and eliminates barriers to social interaction for those who are deaf. In addition, it enhances their ability to be absorbed into the workforce and employment opportunities for people with disabilities are increased significantly. To the beneficiaries who are deaf, this method provides an effective and convenient means of learning and practicing sign language thus improving their lives and self-sufficiency. Globally, hearing loss and deafness are recognized as major health problems affecting millions of individuals. By 2050, nearly 2.5 billion people will have hearing problems and at least 700 million will need hearing aids. (World, 2019) These figures show the necessity of efficient tools and interventions in the field of communication for people with such disorders.

1.1.1 Motivation for Developing SignVision

The primary motivation for the creation of SignVision is the increasing demand for the deaf and hard-of-hearing community to have effective and accessible communication tools. More than 1 billion young adults run the danger of permanent, preventable hearing loss as a result of unsafe listening practices, therefore developing solutions that raise awareness and offer educational materials is crucial. (World, 2019)

1.1.2 Project Objectives

The main objectives that we want to achieve through SignVision are:

1.1.2.1 Empowerment through Education:

The program enables the users to learn the sign symbols and the meaning of the symbols through the use of the ASL, an interactive platform. This educational component is crucial in the sense that diversity and communication skills must be enhanced.

1.1.2.2 Real-time Feedback and Practice:

SignVision is a tool that enables users to practice signing since it provides them with the sign symbol detection and feedback in real time. Feedback also enhances the practice since it provides instant correction of mistakes and makes the practice more engaging.

1.1.2.3 Promoting Safety through Symbol Recognition:

The second feature made available by SignVision is the traffic and street symbols' awareness. These symbols must also be understood in order to promote safety and adherence to traffic rules as this significantly reduces the occurrence of accidents and enhances the safety of the public.

1.1.3 Addressing a Critical Need

Most of the patients with hearing impairment reside in developing countries where there are few, if any, resources for learning and recovery. SignVision meets this need by providing a widely available, affordable option that is compatible with any Android device, anywhere. SignVision is an elaborate, but easily understandable, application for learning traffic signs and ASL with the help of advanced computer vision and machine learning solutions.

1.1.4 Brief

The SignVision project aims at developing an enhanced Android application with a machine learning model that acts as a comprehensive system for sign language identification and learning as well as learning street traffic symbols. This revolutionary application is designed to teach general traffic symbols to the users and help to reduce the communication barriers for the deaf and hard-of-hearing people.

1.1.5 Objectives and Importance

1.1.5.1 Overcoming Learning Barriers:

SignVision tackles the difficulties involved in learning American Sign Language (ASL). Many people find the learning process of ASL challenging because traditional approaches can be time-consuming and may not provide instant feedback. With real-time identification and feedback, our application offers an interactive and user-friendly platform for learning and practicing sign language, guaranteeing precise and efficient learning.

1.1.5.2 Enhancing Communication:

Deaf and hard of hearing people rely on communication. SignVision employs the machine learning technique to identify and translate the ASL signs effectively to enable proper communication. This feature enhances social inclusion and diversity since the users can easily and clearly communicate with other people.

1.1.5.3 Promoting Employment Opportunities:

Thus, one can state that the employment opportunities for deaf people can be greatly improved by focusing on the communication aspect. The study shows that with the help of SignVision, the users can master the signs of ASL that might be useful in the job. The application helps to broaden the set of opportunities for deaf people as it improves their communicative skills and therefore they can receive more vacancies and become more successful workers.

1.1.5.4 Safety Awareness through Traffic Symbol Education:

The second component of SignVision aims at providing the users with information on traffic and street signs. It is important to comprehend these indicators to be safe and avoid violating the rules of the road. To ensure that people conduct themselves safely in public places, the program contains features and information that help users learn and understand various signals.

1.1.6 What might a mobile application with a machine learning model accomplish?

Significant improvements in accessibility, education, and user empowerment can be made possible using a mobile application that integrates with a machine learning model. The machine learning-powered mobile application is advantageous to individuals and stands out as an exceptional offering:

1.1.6.1 Benefits to Individuals

- **Better Learning Experience:**
  - **Interactive and Real-time Feedback:** Learners can engage in the learning process and perfect the use of American Sign Language (ASL) and get feedback almost immediately thus correcting mistakes and improving on the accuracy.
- **Improved Communication:**
  - **Effective ASL Interpretation:** The application assists the deaf to communicate and be more effective in their social and working lives by accurately identifying and translating sign language.
  - **Inclusivity:** The app enhances the social relations between the deaf and the hearing persons since the latter gets a chance to learn sign language.
- **Dual-Focused Learning:**
  - **Comprehensive Training:** The model integrates sign language detection with traffic symbol education to ensure the learning platform addresses the requirements of safety and communication.
  - **Versatile Application:** Due to this two-pronged approach, a wider variety of users can get value from the app, starting with those who need to learn street and traffic signs for safety purposes and ending with those who are interested in ASL.
- **Advanced Machine Learning Integration:**
  - **High Accuracy:** The program is equipped with computer vision and highly effective neural networks for the identification and translation of sign language with a very high level of accuracy.
  - **Continuous Improvement:** The user's actions and feedback can be utilized to modify and enhance the machine learning algorithms, thus maintaining the app up to date and relevant.

- **Accessibility and Convenience:**
  -
  - **Mobile Platform:** SignVision is an application that can be used at any time and at any place, thus allowing the users the flexibility to study and practice while on the move.
  - **User-Friendly Interface:** All the users of the app, irrespective of their age or technical know-how will not have any issues in using the app because of its simplicity.

1.1.7 Problem Definition

Globally, hearing loss and deafness are well-known health conditions that affect millions of people across the globe. The World Health Organization (WHO) has estimated that by the year 2050, about 2.5 billion people will have hearing loss with over 700 million needing a hearing aid. (World, 2019) However, with this growing trend evident in the society the deaf and hard of hearing people encounter significant barriers to social inclusion and effective communication.

Traditional methods of teaching ASL are often proved to be unproductive, time-consuming and lacking of the real-time feedback that is necessary for the student to become effective. The learning process is complex and less effective because of this inefficiency. Some of the major challenges that are common among deaf persons include; They sometimes have a hard time communicating especially with non-sign language users. These difficulties may lead to feelings of exclusion and may end up with social isolation and no interaction in personal and professional life.

Moreover, for safety concerns, people should understand street and traffic signs; however, there are few books available that are designed especially for the hard of hearing. This safety education deficit poses the following risks: Lack of communication skills also results in higher unemployment and fewer educational opportunities for the deaf, thus limiting their chances to become economically productive members of society. These challenges are complex hence the need to come up with an all-inclusive solution that addresses the problems that the deaf and hard of hearing have in learning, communication, safety, and employment.

To solve these issues, the SignVision project aims to create a full Android application that combines advanced machine learning with engaging sign language learning modules. The application aims at enhancing safety consciousness, breaking communication barriers, and providing the deaf and hard of hearing people with employment and education opportunities.

1.1.8 Scope

The SignVision project is a large-scale project that aims to deliver a complete learning system that will include several sign languages and encompass all aspects of learning and communication. As for the project's objectives, the immediate deliverables of this project are as follows: However, the future expansions of this project are as follows in order to provide users with a "world of signs. "

The software will teach sign language to the users in detail in the shortest time possible with basic lessons such as alphabet and numeracy, basic signs, and difficult signs. Real-time sign language detection will make the learning process more interactive due to the opportunity to get feedback and present the material immediately. The software will also be designed to be easy to use with progress tracking and navigation being simple.

To cater for international clients, SignVision plans to integrate real-time traffic sign recognition with AI computer vision, introduce an AI chatbot for sign language practice, regularly update the content database and offer support in multiple languages. This comprehensive approach ensures that the users can be in a position to learn sign language and at the same time learn traffic laws and be in a position to contribute to society and reduce cases of accidents.

SignVision seeks to be a flexible solution that meets a range of user needs by combining all these characteristics into a single, all-inclusive application. In addition to the ease of learning and practicing sign language from home, users will gain from an organized instructional approach to traffic symbols and real-time feedback. Through traffic symbol education, this creative effort not only meets the immediate requirements of the deaf and hard-of-hearing community but also promotes safer public conduct, opening the door for a society that is more informed and inclusive.

1.1.9 Project Features

Our SignVision mobile app gateway is the easy entry point to our vast Internal Learning section, which is home to our sign language identification and training model. When a user starts using the application, they have to go through the registration process and create a new account; after that, the login process is rather straightforward. Users see two main options on the home screen after logging in: In the first category, we have "General Symbols Learning" and in the second category, we have "Sign Language Detection and Learning. "
The Sign Language Detection and Learning module enables the users to gradually familiarize themselves with the sign symbols. The software then allows the users to practice after illustrating to them how the particular sign is done. To practice with our machine learning model for sign language detection, the user can click the "Practice" button. This model provides feedback as learning is taking place hence enhancing the learning experience.
Traffic and street signs and signals are included in the General Symbols Learning module. In this way, the users will enhance their understanding of the indicators and their response to them as well as the traffic laws. This structured method makes it possible for the users to easily obtain and apply both the generic symbols and the sign language; thus, enabling people to engage in safer and more informed interactions in their everyday lives.

1.2 Problem Statement

For the deaf and hard of hearing community, sign language is an essential means of communication. It promotes inclusive environments, improves cultural understanding, and advances cognitive development. Both hearing and non-hearing people can benefit from learning sign language since it promotes social contact and wider communication, bridging the gap between various cultures. Anywhere from 500,000 to two million speak American Sign Language (ASL) in the United States alone. It's the fifth most-used language in the United States behind Spanish, Italian, German and French. (Clason, 2016)

Conventional approaches to teaching American Sign Language (ASL) are frequently tedious, slow, and lacking immediate feedback. It is challenging for students to become proficient because of this. Effective learning is hindered by the general lack of real-time detection capabilities and small datasets of existing models. Furthermore, a fragmented learning experience results from several platforms requiring users to visit different resources for distinct indicators.

These problems are addressed by our initiative known as SignVision which provides a holistic and coordinated learning solution. SignVision stands out from other models, and it has a broader and diverse database, as well as accurate real-time recognition of sign languages, which ensures that users get accurate and immediate responses. However, SignVision eliminates the necessity of users to switch between applications since it compiles a vast array of signs. This makes the process uniform and ensures that the content is of high quality and easily accessible to the learners.

These are the essential features that set SignVision apart from similar platforms like SignSchool (SignSchool, 2020). Although SignSchool provides valuable teaching resources, it lacks real-time identification and extensive data sets. For this reason, our program is not only an effective instructional aid but also a practical means of eradicating barriers to communication and increasing people's awareness of safety by familiarizing them with traffic and street signs.

The modules in this system will be as follows:

1.2.1 Module 1: Admin Server

User information like email IDs, passwords, and user IDs will be handled and stored in the admin server. This module is used for the storage of user credentials safely and for the efficient handling of the user login and access control.

1.2.2 Module 2: User Module

The User module will manage several tasks connected to users, including:

- Sign-in: Enabling users to access their accounts.
- Sign-up: Facilitating the creation of accounts for the new users.
- Recover Password: Allowing users to choose to have a password reset option in case they forget their password.
- Develop Sign Language Proficiency: Facilitating an environment through which users can get and enhance sign languages.

1.3 Literature Review

Paper Title: Real-time Sign Language Recognition using CNN, (PATEL, 2023)

Authors: 20it105 PRATHAMKUMAR PATEL, Medium Article

Summary: This article explains how CNN was employed to develop a sign language recognition system that is real-time. The method is collecting a set of movements that are used in sign language, processing the photos of the movements, and finally, classifying the outcomes by CNN. It is able to identify and categorize sign language motions in real-time video feeds using Python frameworks such as TensorFlow and OpenCV. The model showed its possibility of being useful in helping the deaf community's communication by attaining high accuracy in real-time recognition. The next phase of development will be to improve the ability of the system to adapt to changes in background and lighting as well as expand the gesture vocabulary.

Paper Title: Dynamic Gesture Recognition Using 3D-CNN and LSTM Networks, (Ur Rehman et al., 2022)
Authors: Muneeb Ur Rehman, Fawad Ahmed, Muhammad Attique Khan, Usman Tariq, Faisal Abdulaziz Alfouzan, Nouf M. Alzahrani and Jawad Ahmad

Summary: The focus of this paper is to assess the LSTM networks combined with 3D Convolutional Neural Networks (3D-CNN) for dynamic gesture recognition. Compared to the conventional methods, these two models are more accurate since they include temporal and spatial feature learning. The findings suggest that this combination method improves dynamic gesture recognition by a large margin and, therefore, can be useful for real-time sign language recognition. To avoid overfitting and to enhance the model's ability to generalize, several optimizers and data augmentation were used and the results showed that they had a positive effect on the model.

Paper Title: Sign Language Recognition with Unsupervised Feature Learning, (Chen, 2011)
Authors: Justin Chen

Summary: This paper employs unsupervised feature learning to analyze sign language recognition. The model focuses on 10 ASL letters using a self-created dataset of 12,000 RGB pictures and related depth data. Important approaches include the use of a sparse automatic encoder for feature extraction and a variety of manual segmentation techniques. On test data, the final model, which included a softmax classifier, had a 98% accuracy rate. The project is noteworthy for having a real-time demo that processes frames at a rate of about two seconds per. Future enhancements could include streamlining the implementation for quicker real-time processing and expanding recognition to include the complete ASL alphabet.

1.3.1 Analysis from Literature Review

Studies on the recognition of sign language emphasize several cutting-edge techniques to improve the precision and effectiveness of recognition systems. "Sign Language Recognition with Unsupervised Feature Learning" (Chen, 2011) investigates the use of deep learning techniques to extract relevant features from unlabeled data without the need for large labeled datasets. Convolutional Neural Networks are used in "Real-time Sign Language Recognition using CNN" (PATEL, 2023) to process video data and quickly recognize particular signs; this makes it appropriate for applications that need instant response. "Dynamic Gesture Recognition Using 3D-CNN and LSTM Networks" (Ur Rehman et al., 2022) efficiently recognizes complicated motions over time by utilizing a combination of 3D Convolutional Neural Networks for spatial feature extraction and Long Short-Term Memory networks for temporal dependency capture.

1.3.2 Limitations of Existing Work:

- **Limited labeled data:** Despite this, the approach tries to reduce the dependence on labeled data, it is still possible that large quantities of data will be required for feature extraction during the first stage of the unsupervised learning process.
- **Lack of real-time capability:** Since feature learning is more important than the speed of the computation, the method might not be good for real-time applications.

- **Constraints on the dataset:** Many of the current systems were trained using small datasets that may not contain all the sign language motions and this reduced their effectiveness and robustness.
- **Generalization problems:** The generalization of the model in different real-life situations may be affected by the model's inability to generalize across different signers and different signing styles.
- **Limited dataset and computational complexity:** The 3D-CNN and LSTM network combinations have higher computational complexity, which may not allow their use in devices with limited computational power such as mobile phones. Another disadvantage of this model is that the number of dataset symbols in this model is very small.
- **Real-time performance:** Although the model is designed for detecting dynamic gestures, it is challenging to obtain real-time performance with high accuracy.

1.3.3 Overcoming Research Gaps in Our Project:

SignVision incorporates several significant changes in order to address these limitations:

- **Comprehensive and Diverse Dataset:** To guarantee that our model can correctly identify a range of signals, we make use of a vast and diverse dataset that includes a wide range of sign language motions. This enhances generalization across many signers and signing styles and lowers the possibility of dataset restrictions.
- **Real-time Sign Language Detection:** Real-time sign language recognition is employed in our system and users will be able to receive an immediate response. This, along with other considerations, is done to achieve this and make the system suitable for mobile applications The best neural network

architectures that balance between computational complexity and performance are used.

- **Integrated Learning Modules:** SignVision has an interactive training module that enables the users to learn sign language at their own convenience. This module enhances learning since it provides on screen display and real time detection to enable the students to be corrected immediately.
- **A Holistic Approach to Education:** Our program also identifies signs in sign language and also educates the users on traffic and street signs hence enhancing their knowledge and reducing on cases of accidents. This dual focus enhances the communication skills while at the same time provides relevant information for appropriate interaction with the public spaces.

In contrast to other models and based on the identified research gaps, SignVision has advantages for users and is a convenient, fast, and efficient tool for learning and recognizing sign language. Another advantage that can be distinguished in our project is the large symbol set that we use in our work. We employ 56 symbols in total, which makes for 168,000 frames in total. Moreover, the collected data is in frames and this format is more suitable for identification and learning with higher accuracy and context.

1.4 Methodology and Software Life-Cycle

Using a variety of cutting-edge tools and computer languages, including Python, Keras, OpenCV, MediaPipe, TensorFlow, and Numpy, we created a machine learning model for sign language in our project. Flutter was utilized in the development of the application. The project builds an efficient and successful system for learning and detecting sign language using machine learning and recognition pattern concepts. For our project, we used a 3D CNN model, which improves output accuracy and yields better results in real-time symbol and sign language recognition. In order to identify hand motions, researchers employed 3D-CNNs with attention mechanisms, which outperformed 2D-CNNs in the learning of spatial-temporal features. On the UCF-101 dataset, their 3D-CNN model with Resnet101 architecture obtained 95.5% accuracy, demonstrating better gesture recognition capability. (Ur Rehman et al., 2022)

Several important programming paradigms and techniques were integrated into the development process:

- **Integrated Software Development:** To guarantee flawless functioning and performance, the project smoothly integrates several software components.
- **Object-Oriented Programming (OOP):** Modular, reusable, and maintainable code was produced by applying OOP concepts.
- **Algorithm Design and Analysis:** To improve the functionality of the sign language recognition system, effective algorithms were created and examined.
- **Data Structures:** The sign language data was efficiently managed and processed by using the right data structures.

Due to the fact that Agile approach is flexible, can accommodate changes in the requirements and is developed in iterations, it was chosen for the software life-cycle of this project. Integration and testing are facilitated by the Agile paradigm, which ensures that the application is constantly updated according to the users' feedback and new specifications.

1.4.1 Rationale Behind Selected Methodology

Due to the fact that Agile methodology is flexible, cyclical, and focuses on the users, which are all important when developing a real-time sign language recognition system, the Agile methodology was chosen for the development of the SignVision application. It ensures that the final product meets the user's needs and requirements since Agile allows changes to be made and encourages the improvement

FIGURE 1.1: Agile Development Cycle

of the process and the users' participation in the development process. (Layman et al., 2006) As a technique that supports frequent testing, modular development, and improved teamwork, the technique is ideal for integrating new technologies such as Python, Keras, OpenCV, MediaPipe, TensorFlow, and Numpy. Agile also enables speed improvement and quick prototyping which are crucial for scalability and real-time processing.

It becomes possible to build a complete learning environment with sign language and street/traffic symbols since Agile allows additions to be prioritized based on users' feedback and offers functional increments. Moreover, Agile enhances the application of strong data structures, OOP, and algorithmic design, which ensure accurate data handling, code reusability, and efficient algorithms. Therefore, Agile is a flexible, cyclic, and efficient development paradigm to design an excellent, friendly, and extendable sign language recognition system that eliminates current challenges and sets itself apart from similar solutions. (Deza & Hasan, 2018)

Chapter 2

Requirement Specification

2.1 Functional Requirements

The main functions of the system are explained by the functional requirements. They specify what the system should do—or not do—in reaction to particular inputs. Functional requirements usually include of data entry, calculations, conditional behaviors, and business processes.

2.1.1 Sign Up:

New users can create an account in SignVision by entering their email address and password. A user's information is safely saved in the database upon registration. This allows for progress tracking, customized experiences, and data recovery in the event that a password is lost.

2.1.2 Sign In:

By doing this, current users can access the application by entering their email address and password. Users can access their learning progress, preferences, and tailored content after completing the authentication process successfully. This guarantees that every session is customized based on the user's past interactions and educational background.

2.1.3 Forgot Password:

SignVision users can reset their password using this option if they can't remember it. Users can email requests for password reset links. This function keeps the user's learning process going by guaranteeing that they can access the application continuously even in the event that they forget their login information.

2.1.4 Real-Time Sign Language Detection & Learning:

This function uses the camera on the device to capture and immediately recognize sign language signs. People can perform the movements in front of the camera to be able to learn sign language. The technology assists the users to become more efficient and effective by providing prompt feedback and modifications. This interactive method enhances learning since the users can instantly feel the results of the efforts they have made.

2.1.5 Learning General Symbols:

It offers a guide on traffic and street signs and symbols in a systematic way. In this way, this module improves users' understanding and adherence to public safety laws by explaining important symbols they encounter in everyday life. For learning, it provides lessons and assessments in the form of quizzes and other interactive content

2.1.6 Preprocessing:

We are developing a 3D CNN algorithm model that improves the quality of video input to increase detection accuracy. The program preprocesses video inputs by executing operations like normalization, background subtraction, and noise reduction. This preprocessing makes sure that even under different circumstances, the sign identification system can reliably identify gestures. Cleaning and preparing the dataset for efficient machine learning model training is the preprocessing stage. This entails cleansing the data, transforming it into a format that may be used, and eliminating any unnecessary information. (Chitampalli et al., 2023)

2.1.7 User Data Management:

Email addresses, passwords, and learning progress are all securely stored and managed in the database. All user data is securely maintained by the admin server, guaranteeing data integrity and privacy. Because of this, the application may provide a simplified user interface, tailored content, and dependable data recovery solutions.

2.2 Non-functional Requirements

Non-functional requirements, as opposed to functional requirements, describe the standards by which a system's overall quality and performance are assessed. The system's non-functional requirements are as follows:

2.2.1 Reliability:

The system must have a high degree of dependability for the application to constantly be available and operate as intended without frequent failures or errors. To ensure uptime and data integrity, the backend server—which manages user data and authentication—needs redundancy and failover options.

2.2.2 Scalability:

SignVision needs to be scalable to accommodate an increase in both the number of users and the amount of data. Therefore, the architecture should allow for the horizontal scaling of both the application server and the database.

2.2.3 Performance:

The application should have the ability to detect signs in real time and respond to them immediately; the application should perform very well. Video inputs should be processed accurately and at a faster rate.

2.2.4 Cost:

To reduce the cost to the barest minimum, the application should incorporate free and open-source software libraries (Python, Keras, OpenCV, MediaPipe, TensorFlow, and Numpy). This ensures that consumers can still be able to use and pay for the application in the future.

2.2.5 Security:

The application should employ strict permission and authentication measures to ensure that the user's information is secure. Any sensitive information needs to be encrypted. Security risks have to be addressed through security audits and improvements.

2.2.6 Compatibility:

The software should be compatible with any Android phone any screen size and any OS version of the phone. This ensures that there will be no problems of compatibility whenever a number of consumers install and use the program.

Chapter 3

Project Design

3.1 Use Case Diagram

The use case diagram of SignVision shows the functions and features of the application from the user's point of view. It provides a picture of how actors (users) and the system work in relation to each other, helps to define the requirements of the system, sets user objectives, and explains how users accomplish those objectives with the help of the system. Thus, the use of this diagram helps in the requirement analysis, system design, and development processes, as well as in the communication with the stakeholders, highlighting the many use cases and their relationships. Finally, it is a valuable asset in guaranteeing that the system will be capable of meeting the users' expectations and needs and at the same time align with the project goals and objectives.

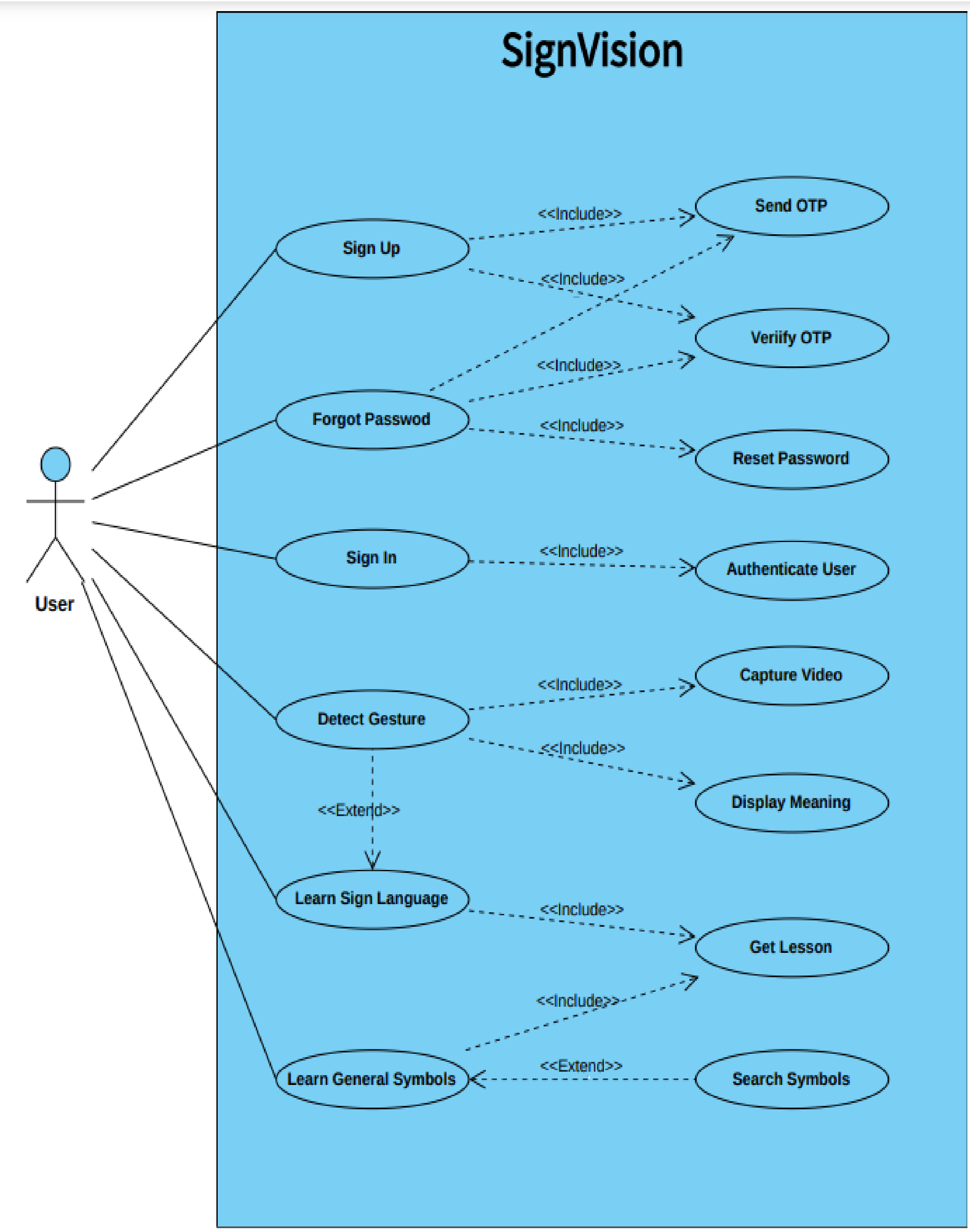

Figure 3.1: Use Case Diagram

3.1.1 Use Cases

3.1.1.1 Use Case For SignUp

Use Case ID	UC1 SignUp
Scope of Use Case	System level
Level of Use Case	User Goal
Primary Actor(s)	User
Secondary Actor(s)	Email Service (for sending OTP) Database (for storing user details)
Pre-conditions	The user has the SignVision application installed. The user is not already registered.
Post-conditions	The user has successfully created an account and can log into the application.
Main Success Scenario	1. User opens the SignVision app and navigates to the Sign-Up page. 2. User enters email and password, and the system sends OTP. 3. User receives and enters OTP, which is verified successfully. 4. A User account is created in the database. 5. User is notified of successful account creation
Alternative Scenario	The application handles invalid user inputs gracefully. If a user enters an invalid email format, the system displays an error message. If the OTP delivery fails, the user is prompted to resend the OTP. Additionally, if the user enters an incorrect OTP, they are shown an error message and asked to try again.
Frequency of Occurrence	Occasionally

3.1.1.2 Use Case For Send OTP

Use Case ID	UC2 Send OTP
Scope of Use Case	System level
Level of Use Case	Sub-Function
Primary Actor(s)	User
Secondary Actor(s)	Email Service
Pre-conditions	User has entered their email address during the sign-up or forgot password process.
Post-conditions	An OTP has been sent to the user's registered email address.
Main Success Scenario	1. The User initiates the "Send OTP" process by providing their email. 2. The System generates a unique OTP. 3. System sends the OTP to the provided email address via the Email Service 4. System confirms that the OTP.
Alternative Scenario	System fails to send the OTP due to email service issues. The system prompts the user to try again later. OR The user provides an invalid email address. The system displays an error message and prompts the user to re-enter their email.
Frequency of Occurrence	Frequently

3.1.1.3 Use Case For Verification of OTP

Use Case ID	UC3 Verify OTP
Scope of Use Case	System level
Level of Use Case	Sub-Function
Primary Actor(s)	User
Secondary Actor(s)	Database (for storing user details)
Pre-conditions	User has received an OTP via email.
Post-conditions	The user's OTP has been verified, allowing them to proceed with account creation or password reset.
Main Success Scenario	1. User initiates the "Verify OTP" process by entering the received OTP. 2. The system checks the entered OTP against the stored OTP in the database. 3. System confirms that the OTP is correct, and the user is allowed to proceed with the next step.
Alternative Scenario	User enters an incorrect OTP. The system displays an error message and prompts for re-entry. OR The OTP has expired. The system notifies the user and prompts them to resend a new OTP.
Frequency of Occurrence	Frequently

3.1.1.4 Use Case For SignIn

Use Case ID	UC4 SignIn
Scope of Use Case	System level
Level of Use Case	User Goal
Primary Actor(s)	User
Secondary Actor(s)	Authentication Service Database
Pre-conditions	The user has an existing account.
Post-conditions	The user is logged into the application.
Main Success Scenario	1. User opens the SignVision application and navigates to the Sign In page. 2. User enters their username and password. 3. System invokes the "Authenticate User" use case. 4. Authentication Service verifies the credentials. 5. System logs the user into the application. The user gains access to their account.
Alternative Scenario	User inputs an incorrect username or password. The system displays an error message. OR The account of the user is blocked after a number of incorrect tries. The system shows a locked account message and requires the user to reset the password.
Frequency of Occurrence	Frequently

3.1.1.5 Use Case For Authenticate User

Use Case ID	UC5 Authenticate User
Scope of Use Case	System level
Level of Use Case	Sub-Function
Primary Actor(s)	User
Secondary Actor(s)	Authentication Service
Pre-conditions	User has entered their username and password on the Sign In page.
Post-conditions	User's credentials are verified.
Main Success Scenario	1. System receives the username and password from the Sign In use case. 2. System sends the credentials to the Authentication Service. 3. Authentication Service verifies the credentials against the database. 4. Authentication Service confirms the credentials are correct. 5. Authentication Service verifies the credentials. The system grants access to the user.
Alternative Scenario	Authentication Service identifies the credentials as invalid. System displays an error message to the user. OR Authentication Service is unavailable. System displays an error message and prompts the user to try again later.
Frequency of Occurrence	Frequently

3.1.1.6 Use Case For Forgot Password

Use Case ID	UC6 Forgot Password
Scope of Use Case	System level
Level of Use Case	User Goal
Primary Actor(s)	User
Secondary Actor(s)	Email Service, Database
Pre-conditions	The user has an existing account and the user has forgotten their password.
Post-conditions	The user has reset their password and can log into the application.
Main Success Scenario	1. User navigates to the Forgot Password page. The user enters their registered email address. 2. System invokes the "Send OTP" use case. The user enters the OTP. System invokes the "Verify OTP" use case. 3. System invokes the "Reset Password" use case. User sets a new password. System updates the password in the database.
Alternative Scenario	User enters an email that is not in the database of the system. OR The user inputs the wrong OTP. The system gives an error message and requires the user to re-enter.
Frequency of Occurrence	Occasionally

3.1.1.7 Use Case For Reset Password

Use Case ID	UC7 Reset Password
Scope of Use Case	System level
Level of Use Case	Sub-Function
Primary Actor(s)	User
Secondary Actor(s)	Database
Pre-conditions	User has successfully verified their OTP.
Post-conditions	User's password has been updated in the database.
Main Success Scenario	1. User initiates the "Reset Password" process by entering a new password. System validates the new password (e.g., checking password strength). 2. System updates the password in the database. System confirms the password has been reset successfully, and the User is notified of the successful password reset.
Alternative Scenario	User enters a weak password. The system displays an error message and prompts for a stronger password. OR System fails to update the password in the database. The system displays an error message and prompts the user to try again later.
Frequency of Occurrence	Occasionally

3.1.1.8 Use Case For Detect Gesture

Use Case ID	UC8 Detect Gesture
Scope of Use Case	System level
Level of Use Case	User Goal
Primary Actor(s)	User
Secondary Actor(s)	Camera, Gesture Recognition System
Pre-conditions	The user is logged into the application. The camera is accessible and functional.
Post-conditions	The application successfully detects the user's gesture and displays the corresponding meaning.
Main Success Scenario	1. User initiates the "Detect Gesture" feature. The system activates the camera and invokes the "Capture Video" use case. 2. System processes the captured video to detect gestures using the Gesture Recognition System/ Deep Learning Model. The user sees the meaning of the detected gesture displayed on the screen.
Alternative Scenario	The system is not able to detect the camera. The system shows an error message and asks the user to grant the camera permission. OR The system does not identify the gesture. The system shows a message that the gesture was not understood and advises you to try again.
Frequency of Occurrence	Frequently

3.1.1.9 Use Case For Capture Video

Use Case ID	UC9 Capture Video
Scope of Use Case	System level
Level of Use Case	Sub-Function
Primary Actor(s)	User
Secondary Actor(s)	Camera
Pre-conditions	The user has initiated the gesture detection feature. The camera is accessible and functional.
Post-conditions	The system has captured a video for gesture recognition.
Main Success Scenario	1. System activates the camera. The system captures the video feed from the camera. 2. System provides the captured video to the gesture recognition system for processing.
Alternative Scenario	The system cannot see the camera. The system outputs an error message and then asks the user to grant permission for the camera. OR It does not record video through the camera. The system then gives an error message to the user and asks the user to try again.
Frequency of Occurrence	Frequently

3.1.1.10 Use Case For Display Meaning

Use Case ID	UC10 Display Meaning
Scope of Use Case	System level
Level of Use Case	Sub-Function
Primary Actor(s)	User
Secondary Actor(s)	None
Pre-conditions	The system has successfully detected a gesture.
Post-conditions	The meaning of the detected gesture is displayed to the user.
Main Success Scenario	1. System receives the detected gesture's information. 2. System retrieves the meaning of the detected gesture. 3. System displays the meaning on the screen to the user.
Alternative Scenario	The system is unable to find the meaning of the detected gesture. The system displays a message indicating that the meaning was not found and suggests trying again.
Frequency of Occurrence	Frequently

3.1.1.11 Use Case For Learn Sign Language

Use Case ID	UC11 Learn Sign Language
Scope of Use Case	System level
Level of Use Case	User Goal
Primary Actor(s)	User
Secondary Actor(s)	Database, Deep Learning Model
Pre-conditions	The user is logged into the application.
Post-conditions	The user has learned a new sign language gesture or phrase.
Main Success Scenario	1. User navigates to the "Learn Sign Language" section of the application. System invokes the "Get Lesson" use case to retrieve the lesson content. The user follows the lesson content to learn a specific sign. 2. If the user wants to practice, the "Detect Gesture" use case can be extended. 3. User successfully learns the sign language gesture or phrase.
Alternative Scenario	The requested lesson is not available. The system displays an error message. OR The user decides not to use the gesture detection feature.
Frequency of Occurrence	Frequently

3.1.1.12 Use Case For Get Lesson

Use Case ID	UC12 Get Lesson
Scope of Use Case	System level
Level of Use Case	Sub-Function
Primary Actor(s)	User
Secondary Actor(s)	Database
Pre-conditions	The user is in the "Learn General Symbols" section or the "Learn Sign Language" section.
Post-conditions	The system has retrieved and displayed the lesson content to the user.
Main Success Scenario	1. User requests a lesson. 2. System retrieves the lesson content from the database. 3. System displays the lesson content to the user.
Alternative Scenario	The system is unable to find the requested lesson. The system displays an error message.
Frequency of Occurrence	Frequently

3.1.1.13 Use Case For Learn General Symbols

Use Case ID	UC13 Learn General Symbols
Scope of Use Case	System level
Level of Use Case	User Goal
Primary Actor(s)	User
Secondary Actor(s)	Database
Pre-conditions	The user is logged into the application.
Post-conditions	The user has explored and learned about various general symbols.
Main Success Scenario	1. User navigates to the "Learn General Symbols" section of the application. The system invokes the "Get Lesson" use case to retrieve the lesson content for general symbols. 2. User reviews the lesson content to learn about specific general symbols. If the user wants to find a specific symbol, the "Search Symbols" use case can be invoked. 3. User successfully learns about the general symbols and their meanings.
Alternative Scenario	System notifies the user that no symbols are currently available. OR The system is unable to find the requested symbol during a search.
Frequency of Occurrence	Frequently

3.1.1.14 Use Case For Search Symbols

Use Case ID	UC14 Search Symbols
Scope of Use Case	System level
Level of Use Case	Sub-Function
Primary Actor(s)	User
Secondary Actor(s)	Database
Pre-conditions	The user is in the ”Learn General Symbols” section.
Post-conditions	The user has found and learned about a specific symbol.
Main Success Scenario	1. User initiates a search for a specific symbol. 2. System retrieves the symbol information from the database. 3. System displays the symbol information to the user.
Alternative Scenario	The system is unable to find the requested symbol. The system displays an error message and suggests alternative search terms.
Frequency of Occurrence	Frequently

3.2 Class Diagram

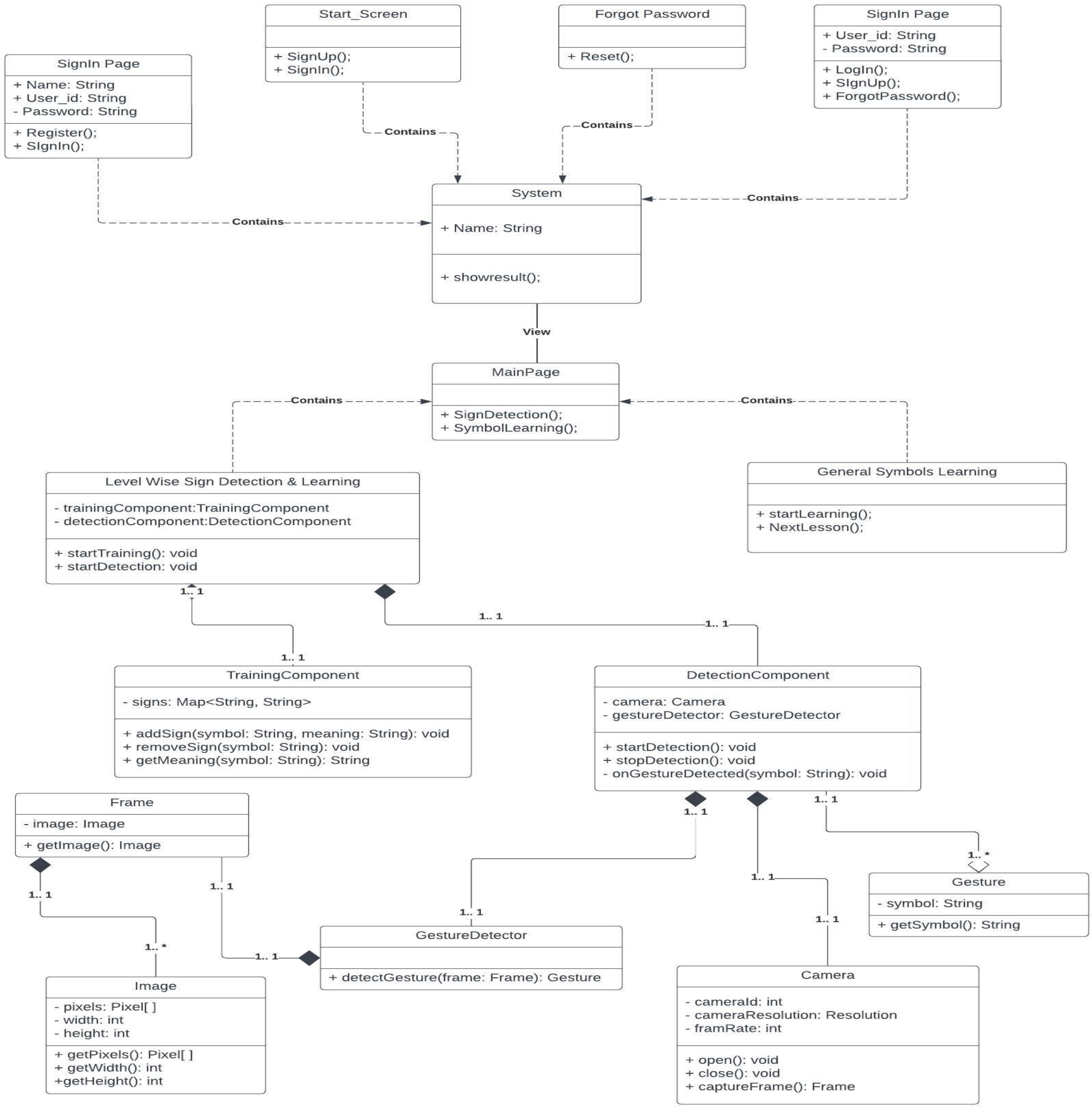

FIGURE 3.2: Class Diagram

3.3 Sequence Diagrams

3.3.1 SignUp

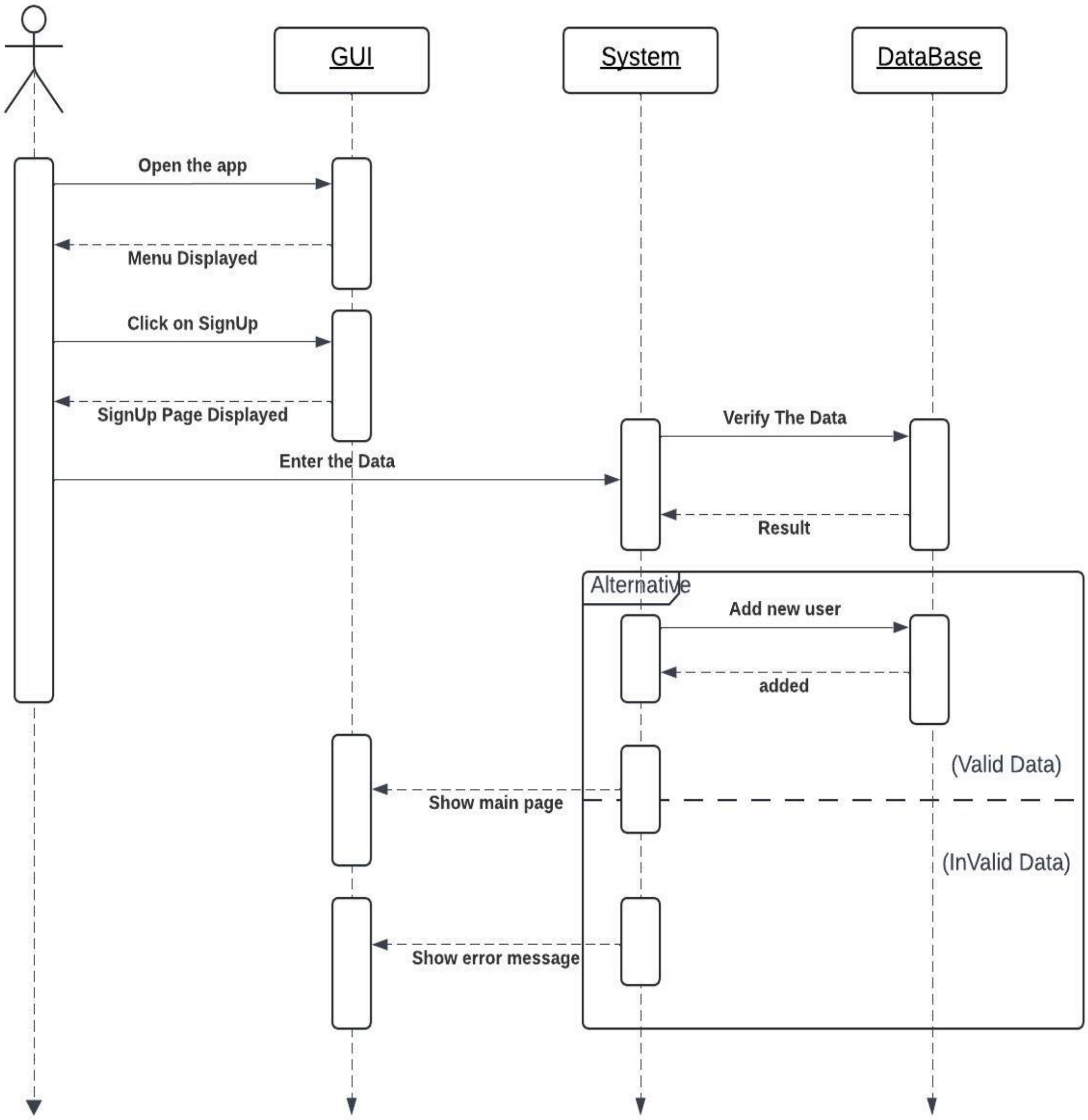

Figure 3.3: SignUp Sequence Diagram

3.3.2 SignIn

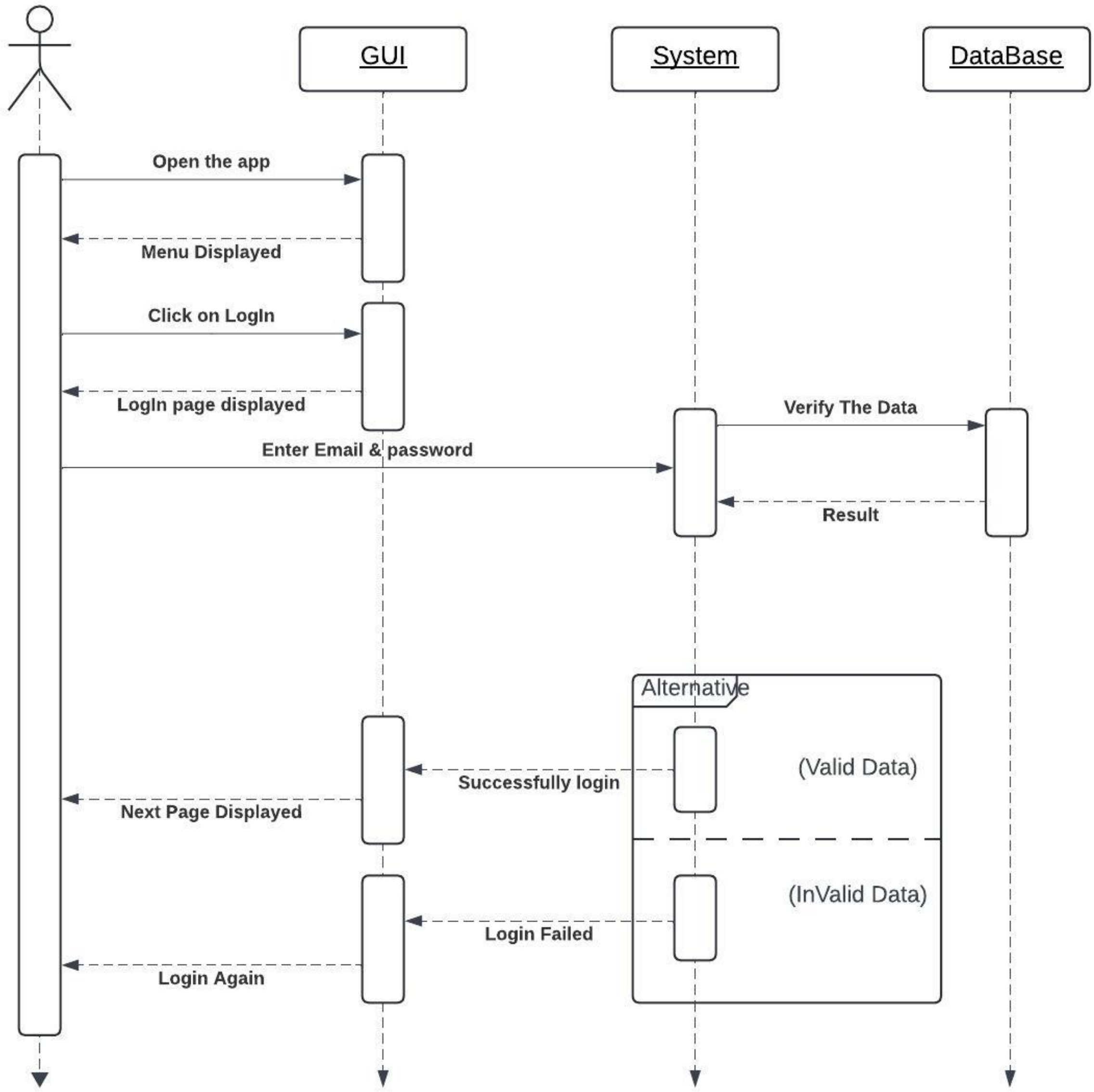

FIGURE 3.4: SignIn Sequence Diagram

3.3.3 Forgot Password

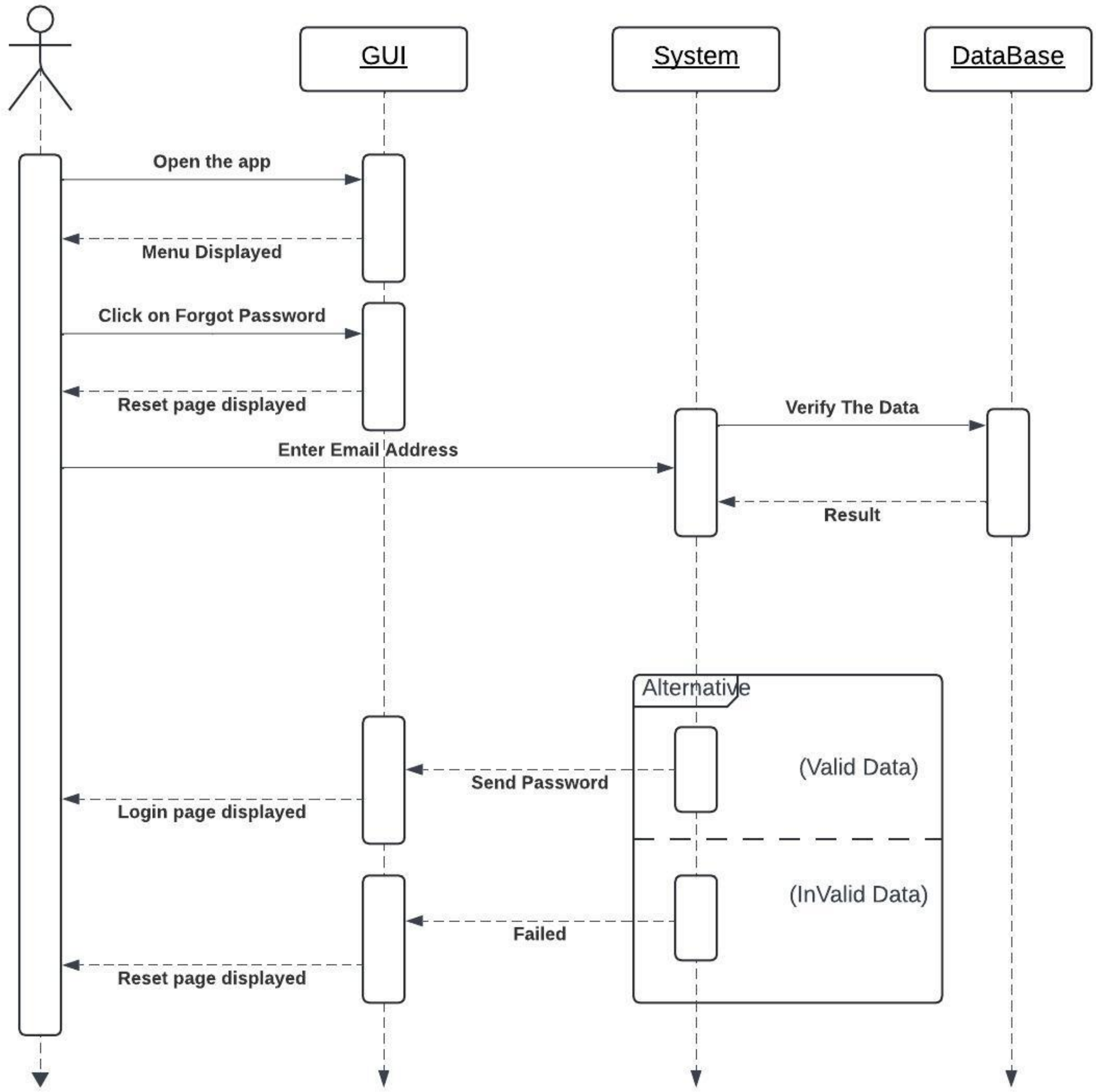

FIGURE 3.5: Forgot Password Sequence Diagram

3.3.4 Gesture Detection

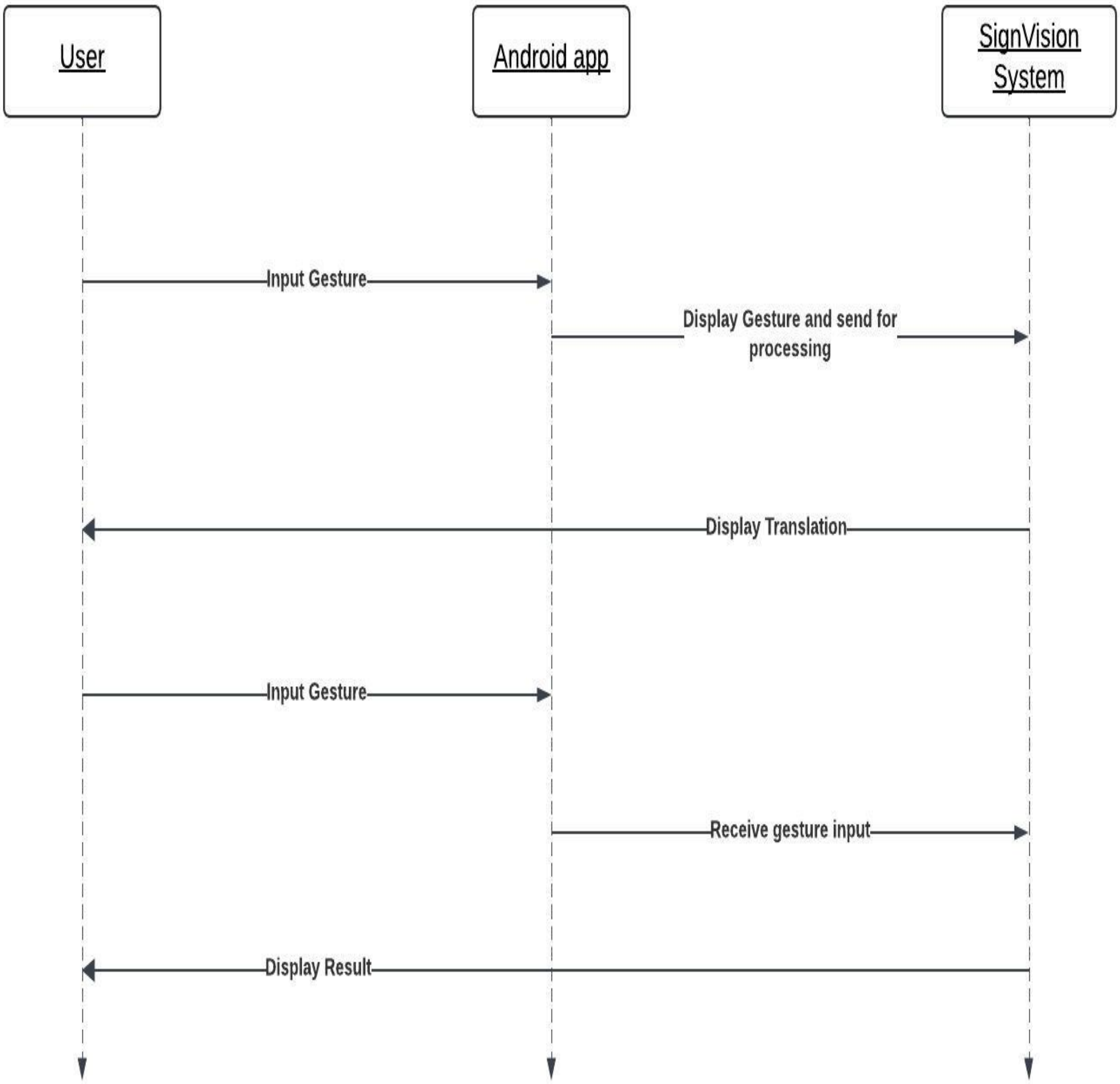

FIGURE 3.6: Sequence Diagram of App

3.4 Data Flow Diagrams

3.4.1 Level 0 Data Flow Diagram

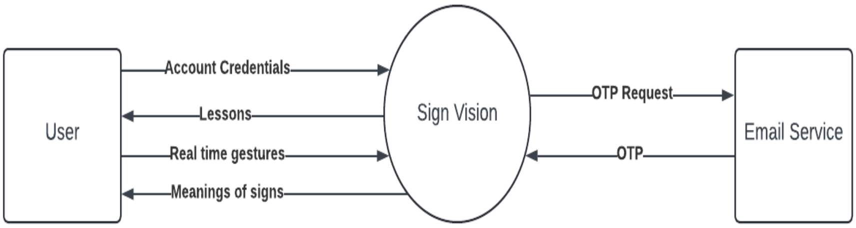

FIGURE 3.7: Context Level Data Flow Diagram

3.4.2 Level 1 Data Flow Diagram

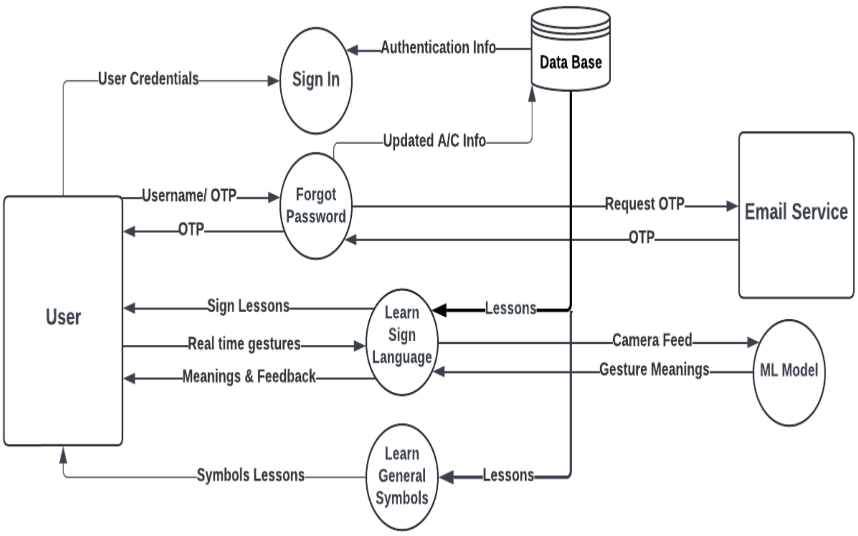

FIGURE 3.8: Level 1 Data Flow Diagram

3.5 Activity Diagram

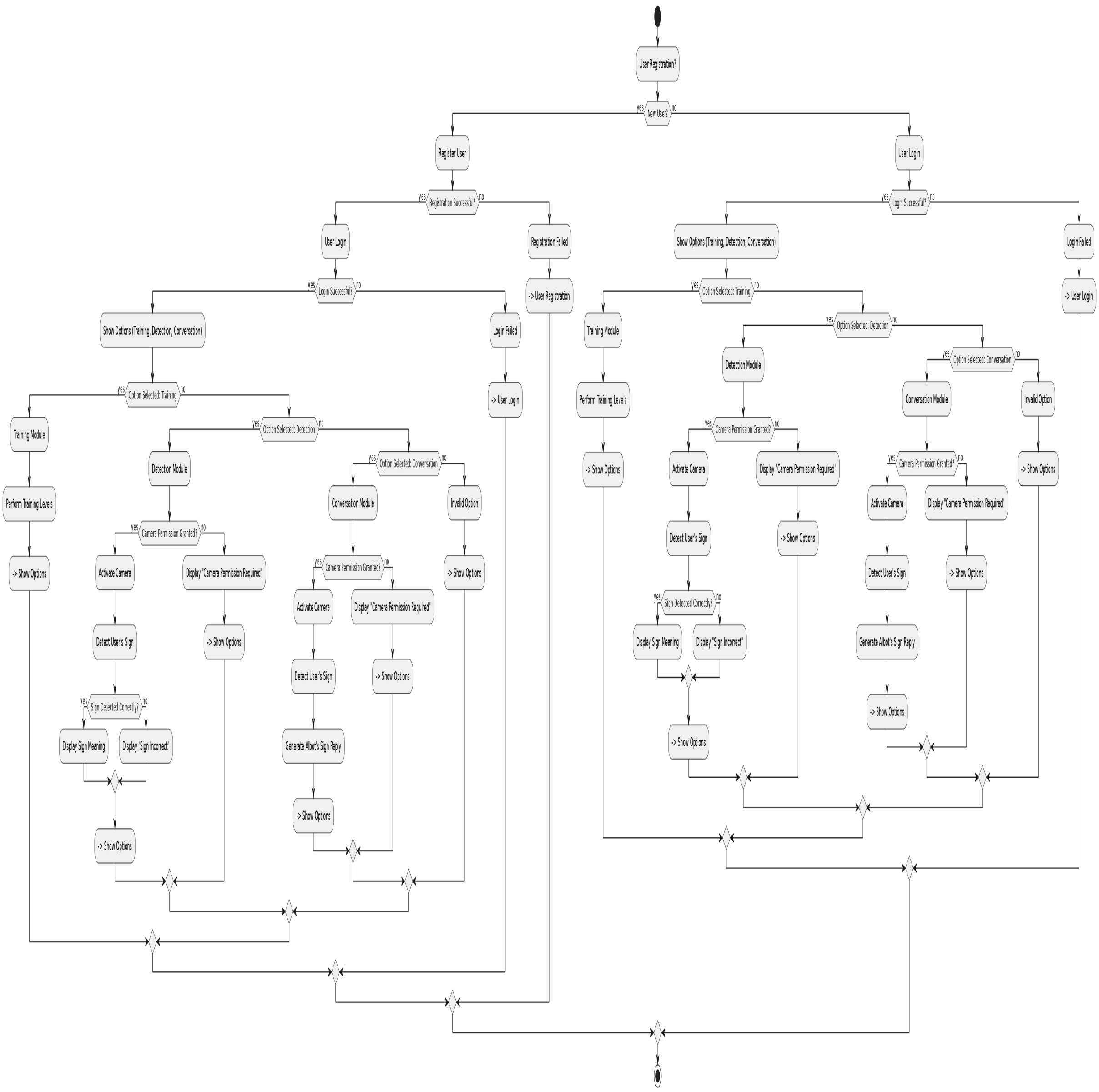

FIGURE 3.9: Activity Diagram of App

Chapter 4

Implementation and Evaluation

4.1 Development Stages

4.1.1 Definition:

SignVision project employs advanced machine learning and mobile development to build an application for real-time sign language recognition and learning. This application makes it possible for people to learn and use sign language effectively and at the same time solve the communication barriers that are experienced by deaf persons. Thus, SignVision significantly enhances the quality of life of deaf people and their social participation by providing real-time communication and a rich learning environment for improving their employment opportunities. Therefore, this project will enhance the quality of life of the non-listeners by enhancing the communication gaps and enabling equal opportunities. (Deza & Hasan, 2018)

4.1.2 Initiation:

Besides, there are many organizations and websites that focus on sign language and there are apps such as SignSchool that can be used in learning the sign language. (SignSchool, 2020) These systems however are not very effective for the need for urgent communication since most of them do not incorporate real-time detection systems. SignVision aims to fill this gap by creating an application that is simpler to use, costs less and is more accessible. The first step was to decide if a full instrument for learning and detecting sign language is needed. To understand SignVision's market and the limitations of the existing solutions, we carried out market analysis. When it comes to needs identification and defining the project's specific goals and objectives, key stakeholders were engaged.

4.1.3 Overview:

The planning stage was to identify the resources, dates, and several milestones of the project roadmap. To ensure that all the activities and the dependencies were captured, a project plan was created to act as a master document of the development process. Features like:

- The users can sign up and make an account.
- It also provides detailed guidelines on how one should sign properly through the use of the app.
- It helps the users to practice sign language and get familiar with general and important symbols to help in overall sign language learning.
- People can learn sign language and enhance their skills in using it with the help of feedback.

4.1.4 Technical Approach of the Project:

The technical solution involves the construction of an efficient sign language detection system through the use of recognition patterns and machine learning. By so doing, we ensure that the system is friendly to the users and that the users can easily get to and use the application without any assistance. This is made possible by the real-time recognition and translation of sign language which helps the user to learn and practice the sign language in a very efficient manner. To ensure that the system is constantly evolving and improving, the Agile methodology was selected for implementation, ensuring that the system adapts to the user's needs and new requirements. (Layman et al., 2006) This strategy ensures that the software is up-to-date and relevant in addressing the communication challenges experienced by the deaf populace as well as enhancing the user experience.

4.1.5 Toots and Technologies:

- **Programming Languages:** Python, Dart
- **Libraries and Frameworks:** Keras, OpenCV, MediaPipe, TensorFlow, Conv3D, MaxPooling3D, LSTM, Numpy, Flutter
- **Development Tools:** Visual Studio Code, Android Studio
- **Version Control:** GitHub

4.1.6 Execution:

During the execution phase, the project team focused on developing the machine learning models and integrating them into the mobile application that was developed using Flutter. This way the team ensured that every component was working in unison and that the application met all the requirements and performance benchmarks as was desired. The project development was done in one and half years, out of which one month was dedicated to development and four months to planning. This methodical approach also prevented the project from going astray and delivered an excellent, user-friendly application.

4.1.7 Monitoring and Control:

To keep tabs on project developments and handle hazards, ongoing monitoring and control systems were put in place. To assess the overall user experience as well as the efficacy of the machine learning models, performance measures were developed. Following the system's development process, each module is thoroughly tested both before and after integration.

4.1.8 Closure:

To make sure all project deliverables were fulfilled, the closing phase comprised last-minute testing and validation. For maintenance and support in the future, documentation was created. After that, the application was launched and given to our supervisor.

4.2 System Integration

SignVision is an app that is meant to work on any kind of smartphone thanks to its universal compatibility design.

4.2.1 User Interface

SignVision is a user-friendly app that's made to be easily accessible. On mobile phones, users can download the application, and the main screen offers options for signing up and logging in. Users must complete a form with their name, password, and email address to register. After registering, users can select between general symbols learning and level-wise sign language detection & learning by logging into their accounts. Users can practice and learn symbols in the sign language detection module. Real-time detection results are shown on the screen. As an alternative, users can study the general symbols learning module to learn about different traffic and street symbols. (WATANABE et al., 2002)

4.2.2 Admin Interface:

Through the admin interface, users can manage their profile, email, and password information. Administrators can control content, keep an eye on user behavior, and update course materials. The application is kept current and sensitive to user needs thanks to this interface.

4.2.3 HardWare Interface:

For real-time sign language detection, SignVision needs a camera-equipped device to record motions. The program is compatible with a variety of smartphones and tablets because it is optimized for mobile platforms. Hardware requirements include:

- Minimum 1 GHz processor
- Availability of a wireless adapter
- Minimum hard drive capacity of 32 GB
- Minimum 1 GB memory (RAM)
- Camera

4.3 User Interface

4.3.1 Welcome Page

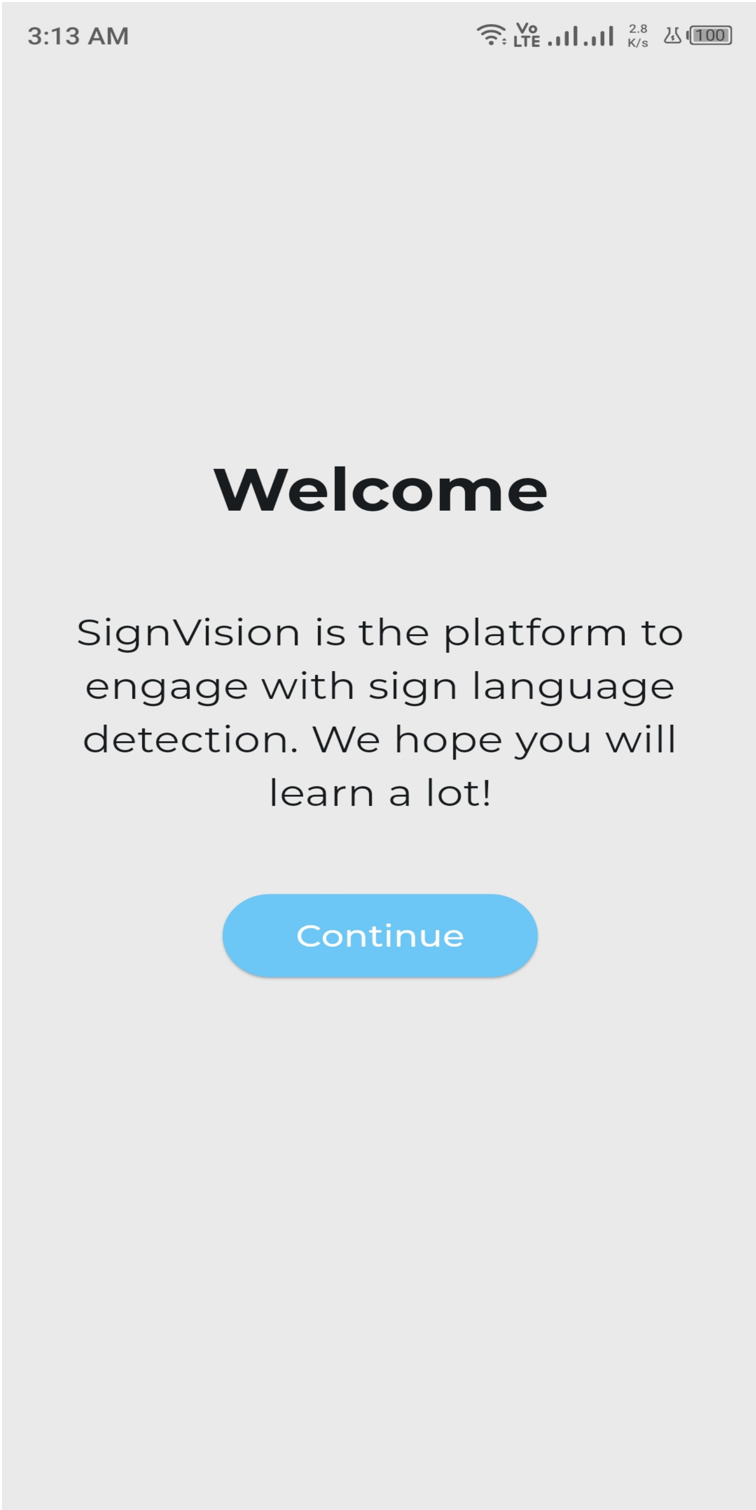

FIGURE 4.1: Welcome Page

4.3.2 SignUp Page

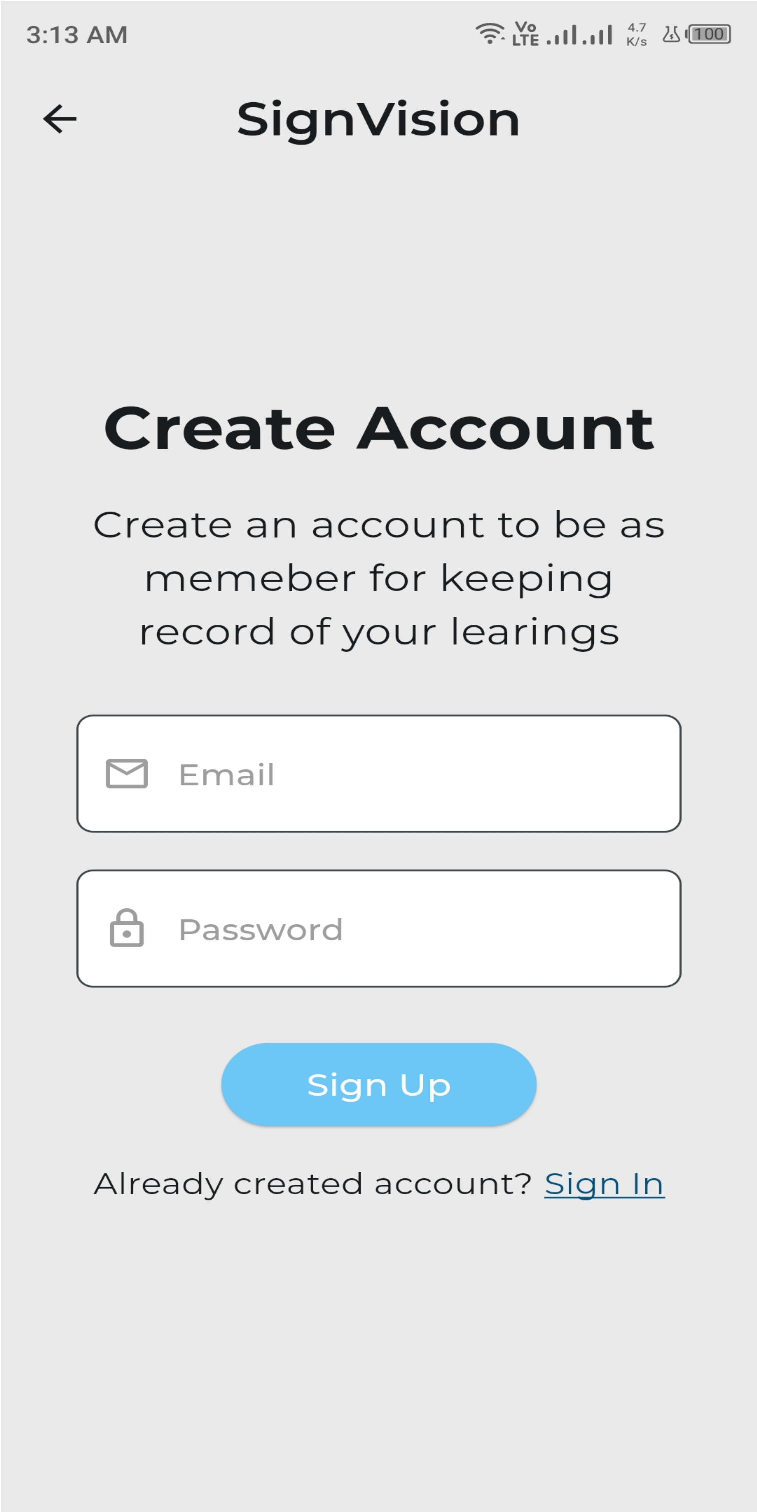

FIGURE 4.2: SignUp Page

4.3.3 Verify The Account

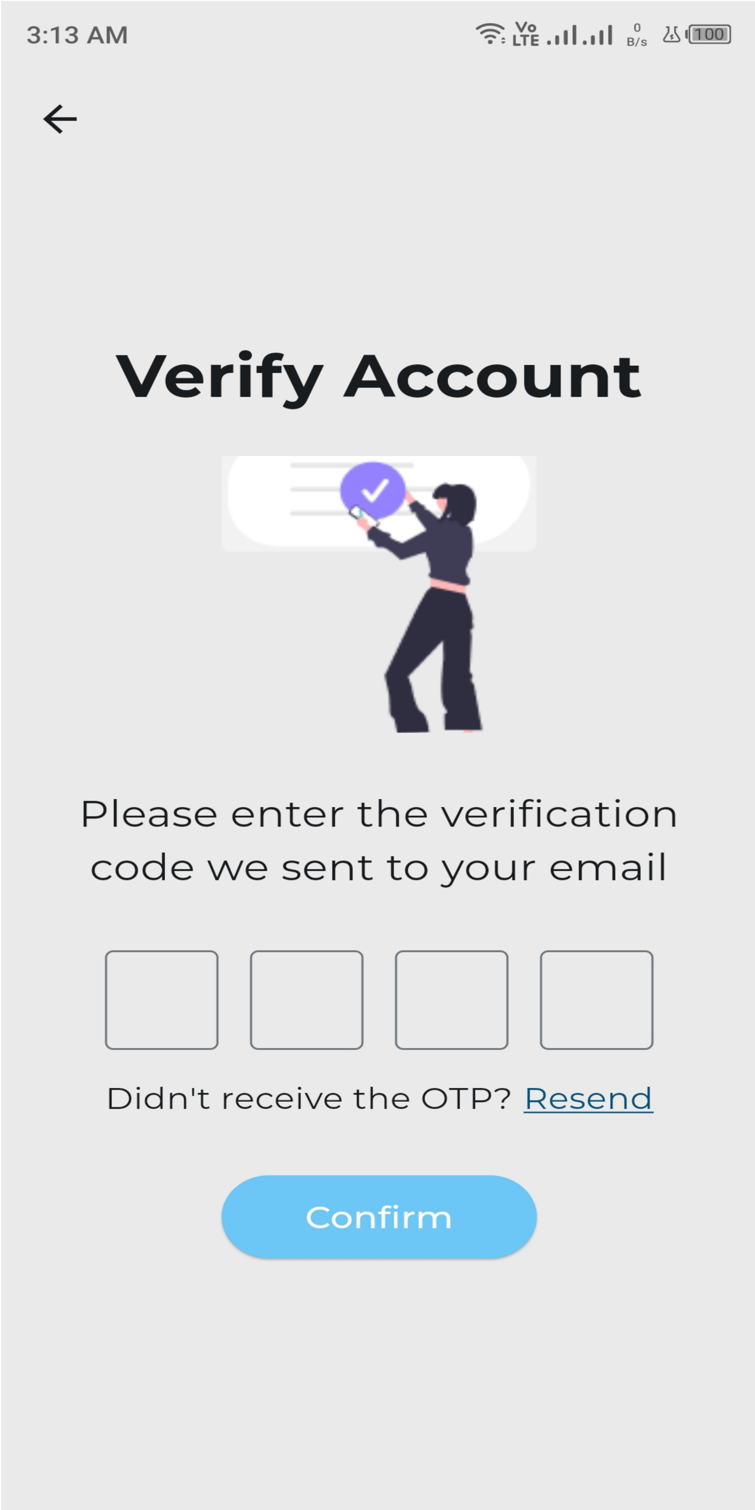

FIGURE 4.3: Verification Page

4.3.4 Verification Complete

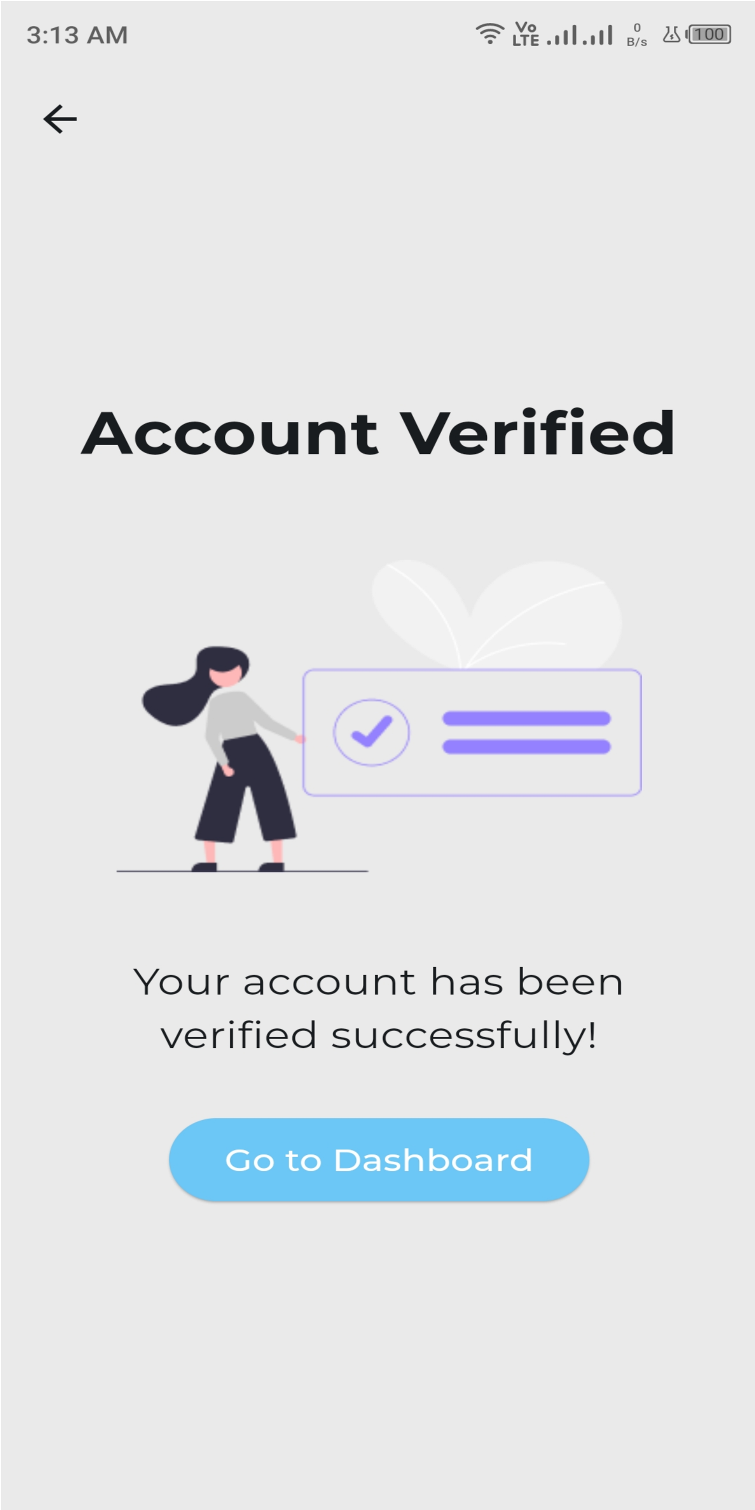

FIGURE 4.4: Verification Complete

4.3.5 SignIn Page

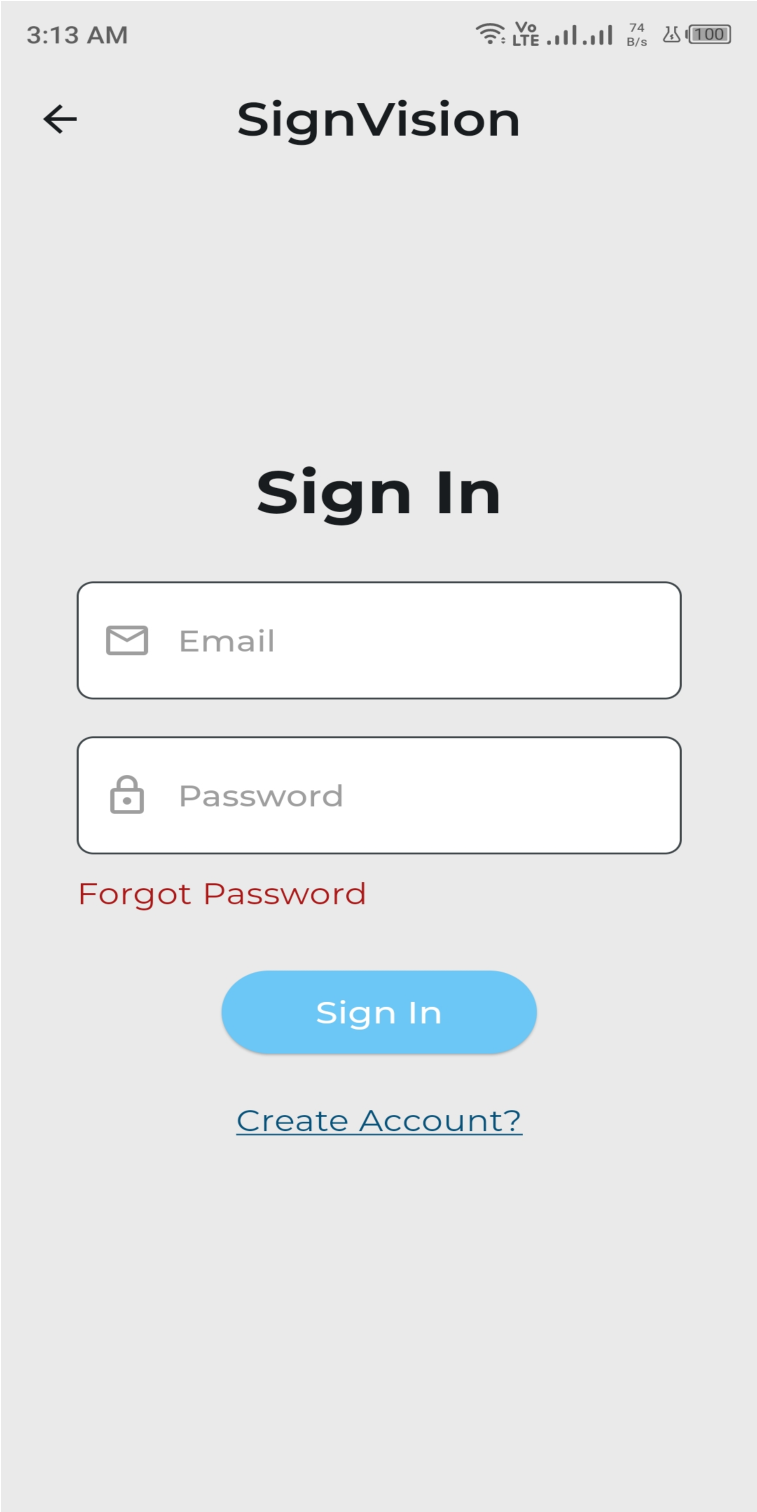

FIGURE 4.5: SignIn Page

4.3.6 Home Screen

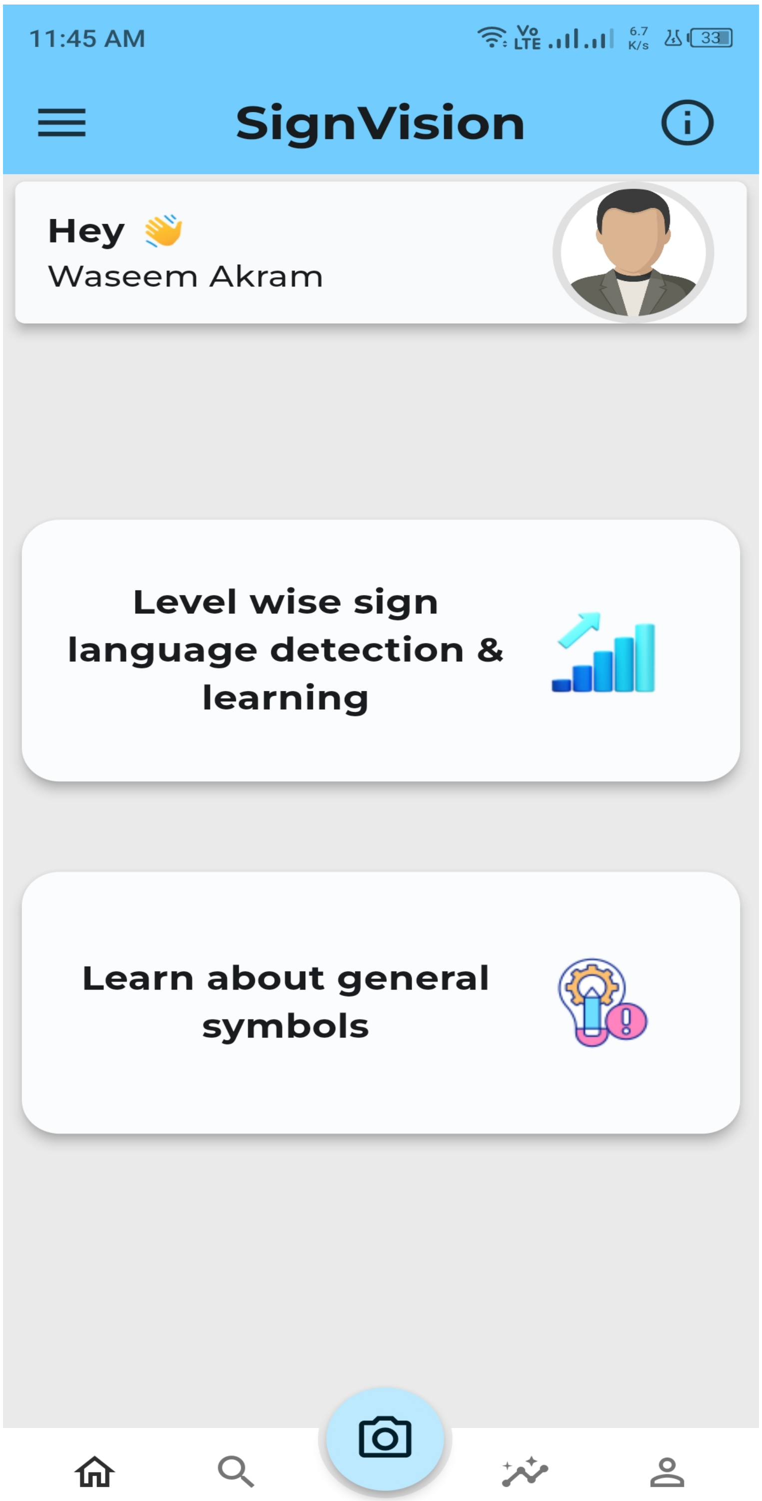

FIGURE 4.6: Home Screen

4.3.7 Level-Wise Sign Language Detection and Learning

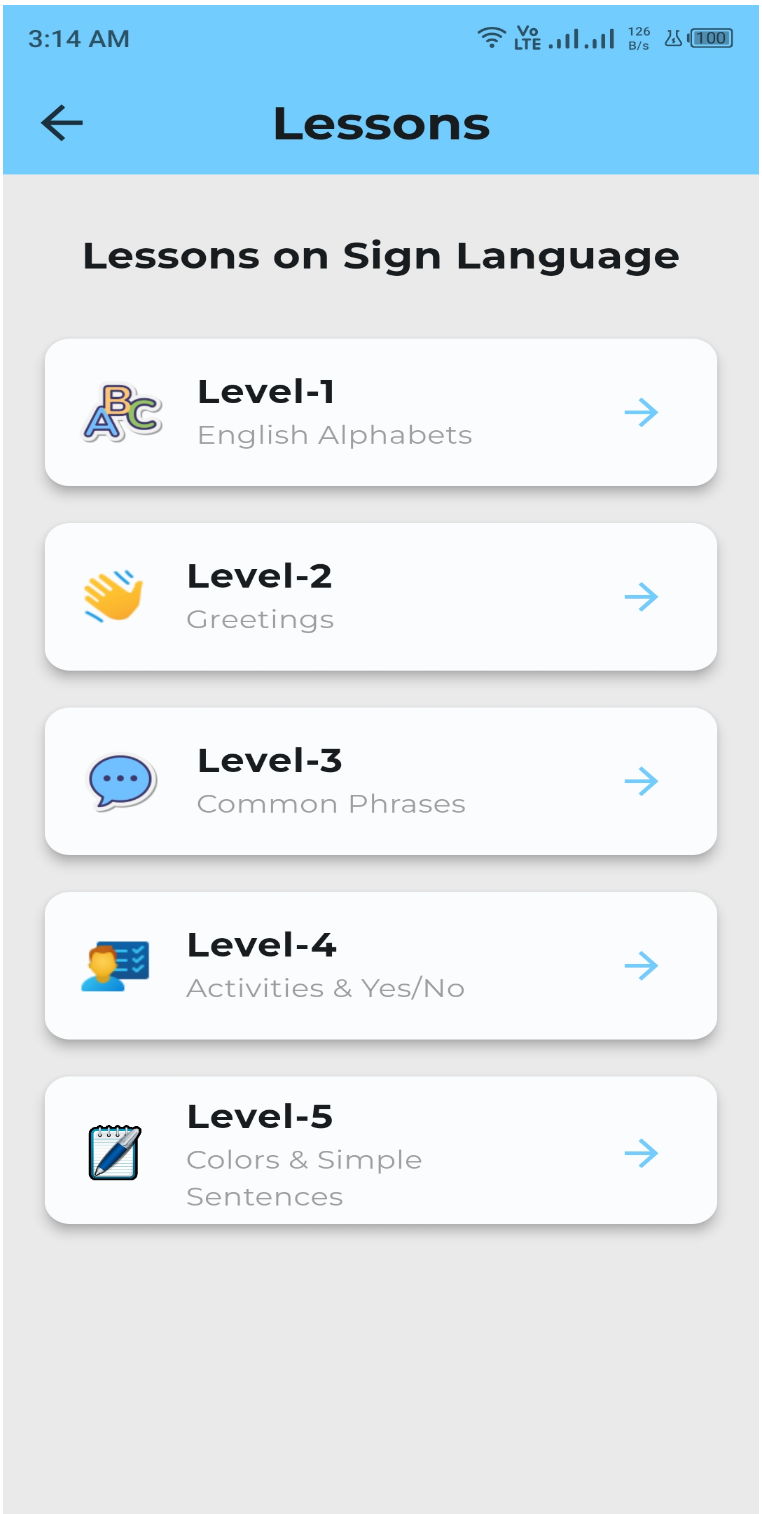

FIGURE 4.7: Level Page

4.3.8 Learning and Detection

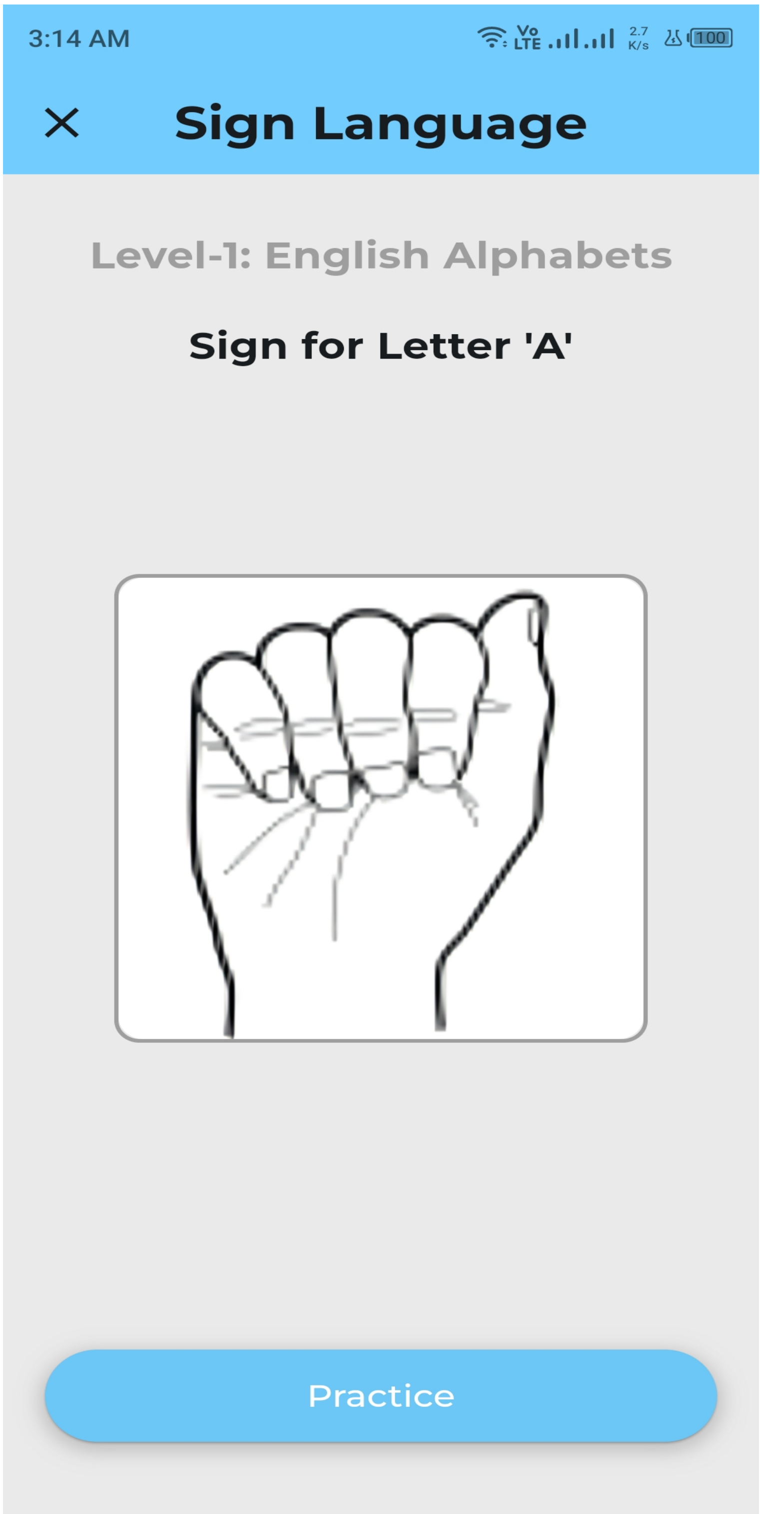

FIGURE 4.8: Learning

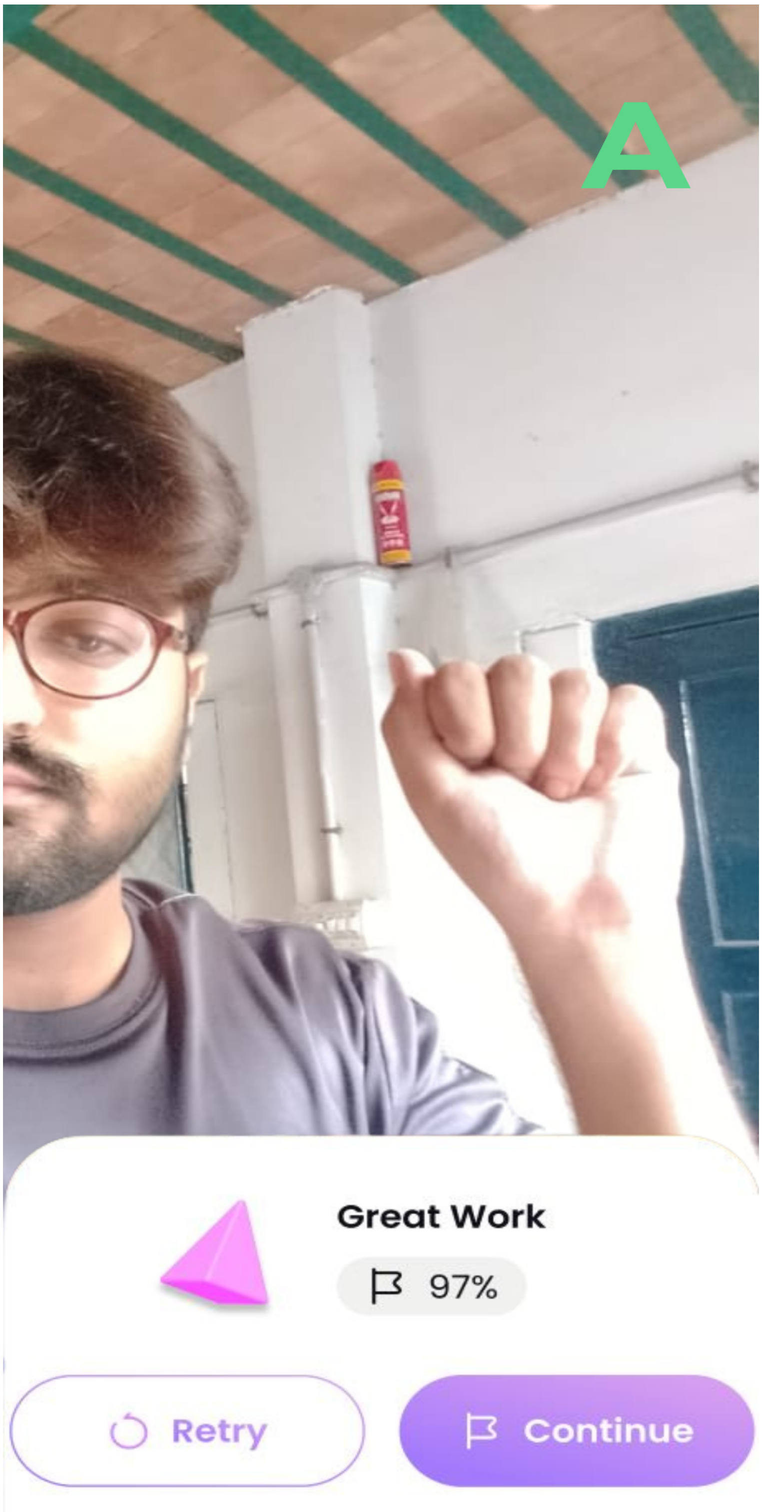

FIGURE 4.9: Detection

4.3.9 General Important Symbols Learning

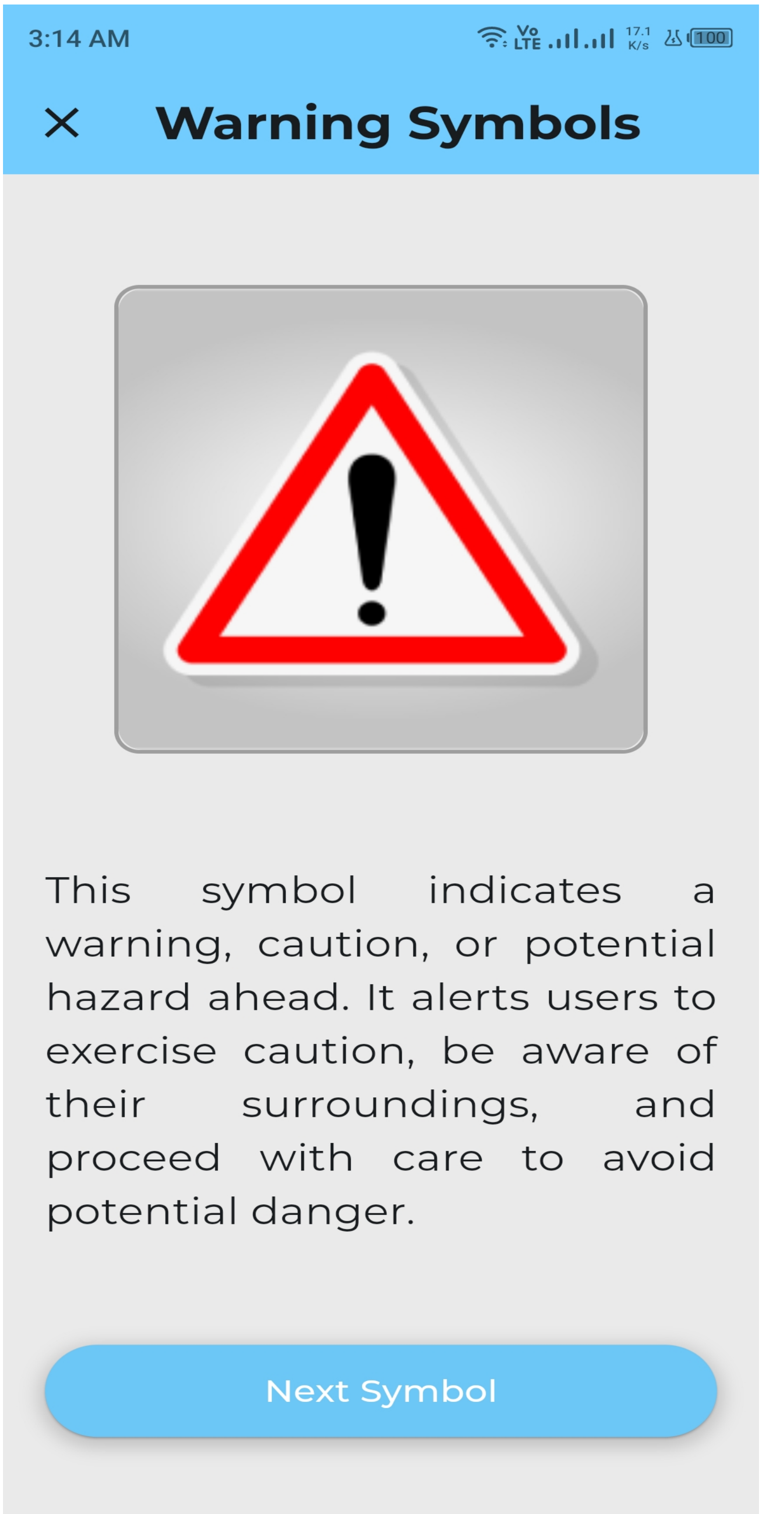

FIGURE 4.10: General Symbols

4.3.10 About Us

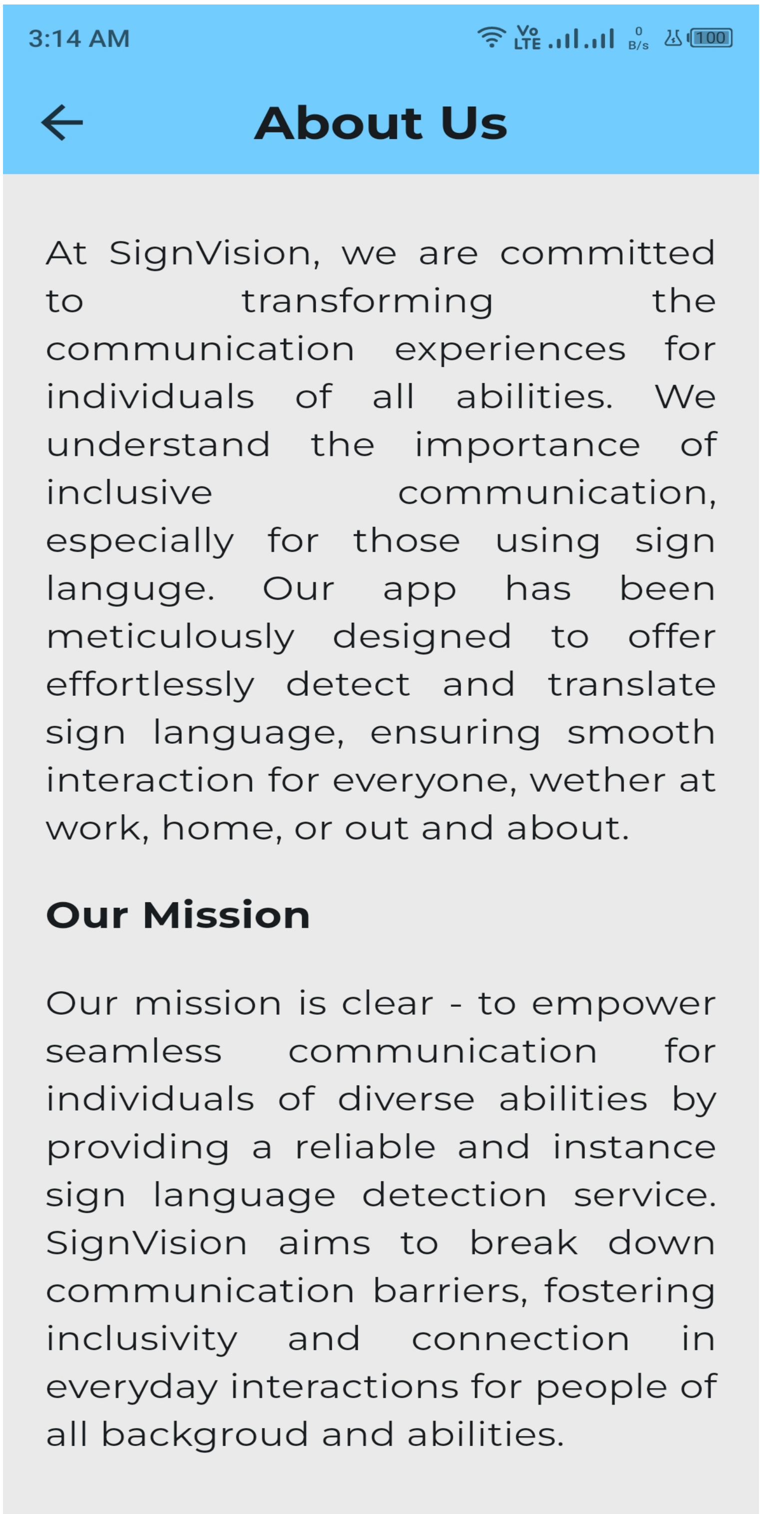

FIGURE 4.11: About Us Page

4.4 Unit Testing

4.4.1 Test Case For SignUp:

Test Case 1	Sign Up
Test Case No	TC-01
Enviroment	Samsung
Priority	High
Application	SignVision
Actors	User
Pre-condition	User is on the sign-up screen
Post-condition	User account is created
Input	Valid email, name, and password
Expected Result	User must successfully login
Actual Result	User has logged in successfully
Test Case Result	Pass

4.4.2 Test Case For SignIn:

Test Case 2	Sign In
Test Case No	TC-02
Enviroment	Samsung
Priority	High
Application	SignVision
Actors	User
Pre-condition	User is on the sign-in screen
Post-condition	User is logged in successfully
Input	Valid email and password
Expected Result	User should be logged in successfully and redirected to the home screen
Actual Result	User has logged in successfully
Test Case Result	Pass

4.4.3 Test Case For Forgot Password Functionality:

Test Case 3	Forgot Password
Test Case No	TC-03
Enviroment	Samsung
Priority	High
Application	SignVision
Actors	User
Pre-condition	User has forgotten the password
Post-condition	Password reset email is sent
Input	Valid email address
Expected Result	A password reset email should be sent to the user
Actual Result	User has reset the password successfully
Test Case Result	Pass

4.4.4 Test Case For Real-Time Hand Gesture Recognition:

Test Case 4	Real-Time Sign Detection
Test Case No	TC-04
Enviroment	Samsung
Priority	High
Application	SignVision
Actors	User
Pre-condition	User has working camera
Post-condition	Correct sign meaning is displayed
Input	Valid sign
Expected Result	The app should detect the sign correctly and display the corresponding meaning
Actual Result	App has correctly detected the sign and displayed its meaning
Test Case Result	Pass

4.4.5 Test Case For Invalid Hand Gesture Detection:

Test Case 5	Invalid Sign Detection
Test Case No	TC-06
Enviroment	Samsung
Priority	High
Application	SignVision
Actors	User
Pre-condition	User is in the real-time sign detection practice mode
Post-condition	Error message or retry prompt is displayed
Input	Invalid or unclear sign
Expected Result	The app should display an error message or prompt the user to try again
Actual Result	App displayed error message and ask user to sign again
Test Case Result	Pass

4.4.6 Test Case For Accurate Result Display Verification:

Test Case 6	Correct Result Display
Test Case No	TC-07
Enviroment	Samsung
Priority	High
Application	SignVision
Actors	User
Pre-condition	User has performed a valid sign detection
Post-condition	Correct result is displayed on the screen
Input	Valid sign detection
Expected Result	The correct sign meaning should be displayed on the screen
Actual Result	The screen is displaying correct sign detection
Test Case Result	Pass

4.4.7 Test Case For Live Video Hand Gesture Detection Performance:

Test Case 7	Live Video Sign Detection
Test Case No	TC-08
Enviroment	Samsung
Priority	High
Application	SignVision
Actors	User
Pre-condition	User has open camera is on the live video detection screen
Post-condition	Detected sign meaning is displayed in real-time
Input	Real-time video feed
Expected Result	The app should process the live video feed, detect signs accurately, and display the corresponding meanings in real-time
Actual Result	App detected the sign accurately and displayed the meaning in real-time
Test Case Result	Pass

4.4.8 Test Case For Efficient Symbol Learning:

Test Case 8	Efficient Symbols Learning
Test Case No	TC-09
Enviroment	Samsung
Priority	High
Application	SignVision
Actors	User
Pre-condition	User has selected the general symbols learning option from home screen
Post-condition	Screen displays the symbol with its meaning and button to move to next symbol
Input	Click on valid symbol learning
Expected Result	The app displays the symbols along with their meanings and showcast a button to move next symbol
Actual Result	Screen display the symbol along with its meaning
Test Case Result	Pass

4.5 BlackBox Testing

On the back end, black box testing is employed to ensure that the system yields some outcome. For instance, to show the right result, the camera has to be able to identify real-time sign language symbols and match them to the model dataset. This procedure validates the system but does not analyze the system's components or how the system works. In addition, it helps in identifying any variance between the planned and actual outcomes, thus ensuring reliability and effectiveness.

4.6 Path Testing

In path testing, all the possible program execution paths are identified by searching through the source code. This will be one of the application's execution routes, to switch on the camera so as to capture the user making a sign in real-time. By conducting thorough testing of these paths, we can ensure that all the paths in the code are working as expected and as designed, which enhances the efficiency and reliability of the system.

4.7 Compatibility Testing

Our application is designed to work on any Android device. The back-end of the system was developed using TensorFlow and Python programming language and the back-end has been tested rigorously to ensure that it works as expected and is compatible with the other components of the system. This rigorous testing helps to avoid problems on different Android devices and versions during the interaction.

4.8 GUI Testing

The GUI of this application has been evaluated and it has been concluded that there are no errors in this application. This enhances the usability and user satisfaction since it provides a positive and uncomplicated user experience.

4.9 Integration Testing

With integrated and tested modules the project was built. This modular design ensures that each part is functioning correctly and contributes positively to the performance and reliability of the program.

4.10 System Testing

During the system testing, a third party tests the modules to confirm that all the features are functional. Such confirmation adds credibility to the testing process as it is done independently and with great attention to detail.

Chapter 5

Conclusion & Future Work

5.1 Conclusion

In conclusion, SignVision software is a great advancement in enhancing the user accessibility, efficiency and interest in learning sign language. By incorporating Python, Keras, OpenCV, MediaPipe, TensorFlow, and Flutter, we have built a comprehensive environment that encompasses both an effective learning process and sign language recognition in real time. We have been able to make small enhancements to the application and ensure that it is in harmony with the needs of the user through the use of the agile approach. The current functionalities are as follows, which respond to the problems of sign language education and communication in the present functions: user identification, general symbol learning, and real-time sign language practice. SignVision reduces the learning barriers and provides immediate feedback on the signs made by the user, thus helping him or her to become more fluent in sign language. This enhances interaction among the deaf and hard of hearing people.

5.2 Future Work

In the future, several changes should be made to SignVision in order to enhance its performance and usefulness. Another one of the upcoming innovations will be the integration of a sign language chatbot that will help make communication with the user more engaging. By the help of this function, users will be able to practice the signs in a conversation as if they are in a real life situation hence enhancing their fluency.

We also plan to integrate real time street symbol and traffic detection. Using this function, the users will be able to direct the cameras of their devices to the unfamiliar symbols that people encounter in everyday life, and the application will immediately explain what these symbols mean. This feature will enhance the practical applicability of learning sign language while at the same time enhancing the safety of the users.

With these upcoming enhancements, we believe that the communication barriers and the issue of diversity will be addressed because SignVision will become a necessity to the users and learners of sign languages. SignVision is committed to continuous improvement and advancement of sign language education technology and therefore will always be ahead of time.

Bibliography

Chen, J. (2011). Sign Language Recognition with Unsupervised Feature Learning. https://cs229.stanford.edu/proj2011/ChenSenguptaSundaram-SignLanguageGestureRecognitionWithUnsupervisedFeatureLearning.pdf

Chitampalli, Prof. Mrs. M., Takalkar, D., Pillai, G., Gaykar, P., & Khubchandani, S. (2023). REAL TIME SIGN LANGUAGE DETECTION. https://www.irjmets.com/uploadedfiles/paper/issue_4_april_2023/36648/final/fin_irjmets1684077417.pdf

Clason, D. (2016, January 20). Why you should learn sign language. Healthy Hearing; Healthy Hearing. https://www.healthyhearing.com/report/52606-Why-you-should-learn-sign-language-in-the-new-year

Deza, A., & Hasan, D. (2018). Sign Language Recognition. MIE324.

Layman, L., Williams, L., & Cunningham, L. (2006). Motivations and measurements in an agile case study. Journal of Systems Architecture, 52(11), 654–667. https://doi.org/10.1016/j.sysarc.2006.06.009

PATEL, 20it105 P. (2023, August 4). Sign Language Recognition Using Python. Medium. https://medium.com/@20it105/sign-language-recognition-using-python-74ef7ea43181

SignSchool. (2020). SignSchool | Learn American Sign Language Online. SignSchool. https://www.signschool.com/

Ur Rehman, M., Ahmed, F., Attique Khan, M., Tariq, U., Abdulaziz Alfouzan, F., M. Alzahrani, N., & Ahmad, J. (2022). Dynamic Hand Gesture Recognition Using 3D-CNN and LSTM Networks. Computers, Materials & Continua, 70(3), 4675–4690. https://doi.org/10.32604/cmc.2022.019586

WATANABE, Y., SUZUKI, S., SUGIHARA, M., & SUEOKA, Y. (2002). AN EXPERIMENTAL STUDY OF PAPER FLUTTER. Journal of Fluids and Structures, 16(4), 529–542. https://doi.org/10.1006/jfls.2001.0435

World. (2019, March 20). Deafness and hearing loss. Who.int; World Health Organization: WHO. https://www.who.int/en/news-room/fact-sheets/detail/deafness-and-hearing-loss

SignVision Model Training code

```
import os
import cv2
import numpy as np
from matplotlib import pyplot as plt
from sklearn.model_selection import train_test_split
import tensorflow as tf
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.layers import Dense, Flatten, Conv3D, MaxPooling3D, LSTM,
TimeDistributed, Dropout, BatchNormalization
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.metrics import Precision, Recall, BinaryAccuracy, F1Score
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay

from google.colab import drive
drive.mount('/content/drive')

!unzip /content/drive/MyDrive/dataset.zip

# Define the list of actions
actions =
['red','eat','howcanihelpyou','sandwich','dream','ketchup','drink','timewhat','whatdidyousay'
,'sleep','blue','no','bread','cereal','nothing','green','seeyoulater','howoldareyou','nicetomeet
you','whereisthebathroom','yes','goodbye', 'hello', 'hi', 'howareyou', 'ok', 'thanks', 'fine',
'sorry', 'whatsup', 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T',
'U', 'V', 'W', 'X', 'Y', 'Z']

# Define the directory where the frames are saved
base_dir = '/content/dataset'

# Initialize lists to store the frames and labels
frames = []
labels = []

# Loop for each action
for i, action in enumerate(actions):
    # Loop for each sequence
    try:
        for sequence in os.listdir(os.path.join(base_dir, action)):
            # Loop for each frame
            for frame_name in os.listdir(os.path.join(base_dir, action, sequence)):
```

```
            # Load the frame
            frame = cv2.imread(os.path.join(base_dir, action, sequence, frame_name))
            # Resize the frame to a smaller size (e.g., 112x112 pixels)
            frame = cv2.resize(frame, (112, 112))
            # Normalize pixel values
            frame = frame / 255.0
            # Append the frame to the frames list
            frames.append(frame)
            # Append the label to the labels list
            labels.append(i)
    except KeyboardInterrupt:
        print("Execution interrupted by user.")

# Convert the frames and labels to numpy arrays
frames = np.array(frames)
labels = np.array(labels)

# One-hot encode the labels
labels = to_categorical(labels)

# Reshape the data to fit the 3D CNN input requirement
frames = frames.reshape((frames.shape[0], 1, frames.shape[1], frames.shape[2],
frames.shape[3]))

# Split the data into training and testing sets
frames_train, frames_test, labels_train, labels_test = train_test_split(frames, labels,
test_size=0.2)

# prompt: show some of the resized frames

# Display some resized frames
plt.figure(figsize=(10, 10))
for i in range(9):
    plt.subplot(3, 3, i + 1)
    plt.imshow(frames_train[i][0])
    plt.title(f"Label: {np.argmax(labels_train[i])}")
    plt.axis('off')
plt.show()

# Define a 3DCNN model
model = Sequential([
    Conv3D(32, (3, 3, 3), activation='relu', strides=(1, 2, 2), padding='same',
input_shape=(None, 112, 112, 3)),
    MaxPooling3D((1, 2, 2), strides=(1, 2, 2)),
```

```
    Conv3D(64, (3, 3, 3), activation='relu', padding='same'),
    MaxPooling3D((1, 2, 2), strides=(1, 2, 2)),
    Conv3D(128, (3, 3, 3), activation='relu', padding='same'),
    Conv3D(128, (3, 3, 3), activation='relu', padding='same'),
    MaxPooling3D((1, 2, 2), strides=(2, 2, 2)),
    Conv3D(128, (3, 3, 3), activation='relu', padding='same'),
    Conv3D(128, (3, 3, 3), activation='relu', padding='same'),
    MaxPooling3D((1, 2, 2), strides=(2, 2, 2)),
    BatchNormalization(),
    TimeDistributed(Flatten()),
    LSTM(64, return_sequences=True),
    LSTM(128),
    Dropout(0.5),
    Dense(64, activation='relu'),
    Dense(128, activation='relu'),
    Dense(len(actions), activation='softmax')
])

# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Define an ImageDataGenerator object with the desired transformations
datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    horizontal_flip=True)

# Define a generator function that will apply transformations to each sequence
def sequence_generator(frames, labels, batch_size):
    while True:
        indices = np.random.choice(frames.shape[0], batch_size)
        batch_frames = frames[indices]
        batch_labels = labels[indices]
        batch_transformed_frames = np.zeros((batch_size, *frames.shape[1:]))
        for i in range(batch_size):
            for j in range(frames.shape[1]):
                img = batch_frames[i, j]
                img = datagen.random_transform(img)  # Apply random transformation
                batch_transformed_frames[i, j] = img
        yield batch_transformed_frames, batch_labels

# Define the batch size
batch_size = 16
```

```
# Create the sequence generators for the training and testing sets
train_gen = sequence_generator(frames_train, labels_train, batch_size)
test_gen = sequence_generator(frames_test, labels_test, batch_size)

# Compute the number of steps per epoch
steps_per_epoch = frames_train.shape[0] // batch_size
validation_steps = frames_test.shape[0] // batch_size

model.summary()

# ... (preceding code)

# Plot some images from the training generator
x, y = next(train_gen)
fig, axes = plt.subplots(nrows=2, ncols=4, figsize=(10, 5))
axes = axes.flatten()
for i in range(8):
  axes[i].imshow(x[i][0])
  axes[i].set_title(f"Label: {np.argmax(y[i])}")
  axes[i].axis('off')
plt.tight_layout()
plt.show()

# Define EarlyStopping callback
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)

# Train the model using the sequence generators
history = model.fit(train_gen, epochs=30, steps_per_epoch=steps_per_epoch,
validation_data=test_gen, validation_steps=validation_steps)
```