Sign Detection Software for Deaf and Dumb

LAB REPORT

Submitted by

Rishabh Agarwal [RA2111003011564]
Kousika S [RA2111003011577]
Pratham Garg [RA2111003011584]

Under the Guidance of

Dr. S. Gnanavel

Associate Professor, Computing Technologies

In partial satisfaction of the requirements for the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE ENGINEERING

with specialization in Computer Science and Engineering

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

MAY 2023

COLLEGE OF ENGINEERING & TECHNOLOGY
SRM INSTITUTE OF SCIENCE & TECHNOLOGY
S.R.M. NAGAR, KATTANKULATHUE – 603 203
Chengalpattu District

BONAFIDE CERTIFICATE

Register No. RA2111003011564 Certified to be the bonafide work done by Rishabh Agrawal of II Year/IV Sem B. Tech Degree Course in the **Practical Software Software Engineering and Project Management 18CSC206J** in **SRM INSTITUTE OF SCIENCE AND TECHNOLOGY,** Kattankulathur during the academic year 2022 – 2023.

4/5/23

LAB INCHARGE

Dr.S.Gnanavel

Associate Professor

SEPM-Course Faculty

Department of Computing Technologies

SRMIST – KTR.

Department of Computing Technology Kattankulathur 603 203 SRM IST

M. Pushpalatha

Head of the Department

Date: 15.5.23

ABSTRACT

The Sign Detection software for deaf and dumb is a revolutionary tool that aims to bridge the communication gap between people with hearing and speech impairments and the general population. This software utilizes machine learning algorithms and computer vision techniques to detect and interpret sign language gestures in real-time. The software has the ability to recognize a wide range of sign language gestures and convert them into spoken or written language. The Sign Detection software has been designed to provide a more efficient and accurate means of communication for people with hearing and speech impairments. The software can be used on a variety of devices, including smartphones and tablets, making it easily accessible to users. The software is also customizable, allowing users to add their own gestures and modify the recognition algorithms to suit their individual needs. The implementation of Sign Detection software has the potential to transform the way in which people with hearing and speech impairments interact with the world around them. It provides a simple and intuitive means of communication that is not reliant on the use of sign language interpreters or written notes. The software can be used in a variety of settings, including schools, workplaces, and public spaces, making it an invaluable tool for people with hearing and speech impairments.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

WBS	**Work Breakdown Structure**
ER	**Entity Relation**
DFD	**Data Flow Diagram**
UI/UX	**User Interface and User Experience**
IDE	**Integrated Development Environment**
OPD	**Outpatient Department**

LIST OF TABLES

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	1
Title of Experiment	SIGN DETECTION FOR DEAF AND DUMB
Name of the candidate	RISHABH AGRAWAL
Team Members	S.KOUSIKA, PRATHAM GARG
Register Number	RA211003011577, RA2111003011564, RA2111003011584
Date of Experiment	28.1.23

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To Frame a project team, analyze and identify the possible solutions for deaf and dumb people using virtual sign detection

Team Members:

S. No	Register No	Name	Role
1	RA2111003011564	RISHAB AGARWAL	Lead/Rep
2	RA2111003011577	KOUSIKA.S	Member
3	RA2111003011584	PRATHAM GARG	Member

Project Title: **SIGN DETECTION FOR DEAF AND DUMB**

Project Description

- Hand gesture is one of the methods used in sign language for non-verbal communication. It is most commonly used by deaf & dumb people who have hearing or speech problems to communicate among themselves or with normal people.
- Various sign language systems had been developed by many makers around the world but they are neither flexible nor cost-effective for the end users.
- Hence, it is a software which presents a system prototype that is able to automatically recognize sign language to help deaf and dumb people to communicate more effectively with each other or normal people.
- Dumb people are usually deprived of normal communication with other people in the society, also normal people find it difficult to understand and communicate with them. These people have to rely on an interpreter or on some sort of visual communication.
- An interpreter won't be always available and visual communication is mostly difficult to understand. Sign Language is the primary means of communication in the deaf and dumb community.
- As a normal person is unaware of the grammar or meaning of various gestures that are part of a sign language, it is primarily limited to their families and/or deaf and dumb community.

THE PROJECT

- Sign language is the mode of communication which uses visual ways like expressions, hand gestures, and body movements to convey meaning.

- Sign language is extremely helpful for people who face difficulty with hearing or speaking. Sign language recognition refers to the conversion of these gestures into words or alphabets of existing formally spoken languages.

- Thus, conversion of sign language into words by an algorithm or a model can help bridge the gap between people with hearing or speaking impairment and the rest of the world.

- Hand gesture recognition for human computer interaction is an area of active research in computer vision and machine learning. One of its primary goals is to create systems, which can identify specific gestures and use them to convey information or to control a device.

THE HISTORY

- The researches done in this field are mostly done using a glove based system. In the glove based system, sensors such as potentiometer, accelerometers etc. are attached to each of the finger. Based on their readings the corresponding alphabet is displayed.

- Christopher Lee and Yangsheng-Xu developed a glove-based gesture recognition system that was able to recognize 14 of the letters from the hand alphabet, learn new gestures and able to update the model of each gesture in the system in online mode.

- Over the years advanced glove devices have been designed such as the Sayre Glove, Dexterous Hand Master and Power Glove.

- The main problem faced by this gloved based system is that it has to be recalibrate every time whenever a new user on the finger-tips so that the fingertips are identified by the Image Processing unit. We are implementing our project by using Image Processing.

LIMITATIONS

- The user must be within a defined perimeter area, in front of the camera.

- The number of signs used in sign language is huge and collecting the dataset for each and every sign is very difficult.

- The model has to be trained using hundreds of images to increase accuracy and availability of this is very tough.

- The model designed is not totally accurate for every person. It works best for the person whose images have been used to train the model.

.

- Hand pose is defined with a bare hand and not occluded by other objects.

- The system must be used indoor, since the selected camera does not work well under sun light conditions.

APPROACH

- It is required to make a proper database of the gestures of the sign language so that the images captured while communicating using this system can be compared.

- A whole lot of libraries and pre-trained models are required.

- A webcam in the system is required for accessing live footage for testing.

BENEFITS

- This project will reduce the communication gap between the specially abled ones and the rest of the world by breaking down the barriers.

- Wearing of color bands is not required in our system. By that, securities could authorize an individual's identity depending on "who she is", and not "what she has" and "what she could remember". Two main classes can be found in biometrics:-

- · Physiological – It is associated with the body shape, includes all physical traits, iris, palm print, facial features, Fingerprints, etc.

- Behavioral – Related to the behavioral characteristics of a person. A characteristic widely used till today is signatures. Modern methods of behavioral studies are emerging such as keystroke dynamics and voice analysis.

Result: Thus, the project team formed, the project is described, the business case was prepared and the problem statement was arrived.

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	2
Title of Experiment	Identification of Process Methodology and Stakeholder Description
Name of the candidate	RISHABH AGRAWAL (RA2111003011564)
Team Members	KOUSIKA S (RA2111003011577) PRATHAM GARG (RA2111003011584)
Register Number	
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To identify the appropriate Process Model for the project and prepare Stakeholder and User Description.

Team Members:

Sl No	Register No	Name	Role
1	RA2111003011564	Rishabh Agarwal	Rep/Member
2	RA2111003011577	Kousika	Member
3	RA2111003011584	Pratham Garg	Member

Project Title:

Selection of Methodology

Agile methodology is an iterative and incremental approach to software development that emphasizes flexibility, collaboration, and customer involvement. It is best justified for a sign detection software because it allows for rapid prototyping and continuous testing and feedback from users. This ensures that the software is meeting the needs of the users and can quickly adapt to changes or new requirements. Additionally, Agile methodology promotes a flexible and adaptive approach, which can be beneficial in a field like sign detection where requirements may change frequently or be hard to predict.

Stakeholder Name	Activity/ Area /Phase	Interest	Influence	Priority (High/ Medium/ Low)
Owner	Setting the project's overall goals and objectives.	High	High	1
The software development team	This includes the project manager, developers, designers, and any other team members responsible for creating the software.	High	Low	2
The end-users	Provides feedback	High	Low	6
The sponsors or clients	These are the individuals or organizations that are providing funding or resources for the project. They may have specific requirements or constraints that need to be taken into consideration during the development process.	Low	High	3
Regulators and standardization bodies	They may need to review and approve the software before it can be used in certain applications	Low	High	4
Suppliers	These are companies or individuals who provide materials, equipment, or services that are necessary for the development or deployment of the software	Low	Low	5

Table 1

Result: Thus the Project Methodology was identified and the stakeholders were described.

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	3
Title of Experiment	System, Functional and Non-Functional Requirements of the Project
Name of the candidate	RISHABH AGRAWAL (RA2111003011564)
Team Members	KOUSIKA S (RA2111003011577) PRATHAM GARG (RA2111003011584)
Register Number	
Date of Experiment	

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To identify the system, functional and non-functional requirements for the project.

Team Members:

S No	Register No	Name	Role
1	RA2111003011564	Rishabh Agarwal	Rep/Member
2	RA2111003011577	Kousika S	Member
3	RA2111003011584	Pratham Garg	Member

Project Title: Sign Detection Software

System Requirements

1. Hardware requirements: The system will likely require high-performance computer hardware with a powerful processor, large amount of RAM, and dedicated graphics card. It may also require specialized hardware such as cameras or sensors for capturing images or video.

2. Operating system: The system will require a specific operating system such as Windows, Linux, or macOS. It should be compatible with the hardware and software requirements of the system.

3. Software requirements: The system will likely require specialized software such as computer vision libraries or image processing tools. It may also require programming languages such as Python, C++, or Java.

4. Storage: The system will require large storage capacity to store images and videos used for training and testing the system.

5. Network requirements: The system may require a high-speed internet connection for data transfer and communication with other systems.

6. Power supply: The system will require a stable power supply to ensure continuous operation.

7. Environmental requirements: The system may require specific environmental conditions such as temperature, humidity, and lighting to ensure proper functioning.

8. Safety and security: The system should be designed with security measures to protect sensitive data and prevent unauthorized access.

9. Scalability: The system should be designed to handle increasing amounts of data and processing power as the system grows.

10. Interoperability: The system should be designed to interact and integrate with other systems and platforms.

11. Accessibility: The system should be designed to be accessible to people with disabilities and those who have difficulty using standard computer interfaces

Functional Requirements

1. Image and video capture: The software should be able to capture images and videos from various sources such as cameras, smartphones, or drones.

2. Image and video processing: The software should be able to process images and videos to detect and recognize signs, such as traffic signs, road signs, or warning signs.

3. Object detection: The software should be able to detect and locate signs in images and videos using techniques such as machine learning or deep learning.

4. Object recognition: The software should be able to recognize the type of sign and provide information such as its shape, color, text, or symbols.

5. Real-time processing: The software should be able to process images and videos in real-time, providing immediate results to the user.

6. Data storage and management: The software should be able to store and manage images and videos used for training and testing the system, as well as the data generated by the system such as detected signs and their location.

7. User interface: The software should have a user-friendly interface that allows users to interact with the system and access its features and functions.

8. Alerts and notifications: The software should be able to generate alerts and notifications when a sign is detected, providing the user with information about the sign's location and type.

9. Reporting and analysis: The software should be able to generate reports and perform analysis on the data generated by the system, such as the number of signs detected, their location, and their type.

10. Integration with other systems: The software should be able to integrate with other systems and platforms, such as GPS systems, navigation systems, or traffic management systems, to provide additional information or functionality.

11. Customization: The software should be able to be configured to meet the specific needs of different users, such as different languages or different types of signs.

12. Security: The software should have strong security measures to protect sensitive data and prevent unauthorized access.

Non-Functional Requirements

1. Performance: The software should be able to process images and videos quickly and accurately, providing real-time results to the user.

2. Scalability: The software should be able to handle increasing amounts of data and processing power as the system grows.

3. Reliability: The software should be able to operate continuously and reliably, with minimal downtime or errors.

4. Usability: The software should be easy to use and understand, with a user-friendly interface and clear instructions.

5. Maintainability: The software should be easy to maintain, with clear documentation and well-structured code.

6. Portability: The software should be able to run on different hardware and software platforms, with minimal modification..

7. Compliance: The software should comply with relevant laws and regulations, such as data protection and privacy laws.

8. Accessibility: The software should be accessible to people with disabilities and those who have difficulty using standard computer interfaces.

9. Extensibility: The software should be designed to be easily extended or modified to add new features or functionality.

10. Compatibility: The software should be compatible with other systems and platforms that it may need to interact with.

11. Localization: The software should be able to support different languages and cultural conventions.

Result: Thus the requirements were identified and accordingly described.

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	4
Title of Experiment	Prepare Project Plan based on scope, Calculate Project effort based on resources and Job roles and responsibilities
Name of the candidate	RISHABH AGRAWAL (RA2111003011564)
Team Members	KOUSIKA S (RA2111003011577) PRATHAM GARG (RA2111003011584)
Register Number	RA2111003011564 RA2111003011577 RA2111003011584
Date of Experiment	06/02/2023

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To Prepare Project Plan based on scope, Calculate Project effort based on resources, Find Job roles and responsibilities

Team Members:

Sl No	Register No	Name	Role
1	RA2111003011564	Rishabh Agarwal	Lead
2	RA2111003011577	Kousika S	Member
3	RA2111003011584	Pratham Garg	Member

1. Project Management Plan

Describe the key issues driving the project. **[Min 3 Focus Areas]**

Focus Area	Details
Integration Management	Governance Framework Project Team Structure Roles & Responsibilities of Team Change Management (Change Control, Issue Management) Project Closure
Scope Management	Scope Statement Requirement Management (Gathering, Control, Assumption, Constraint Stakeholder) Define Deliverable Requirement Change Control Activities and Sub-Tasks
Schedule Management	Define Milestones Schedule Control
Cost Management	Estimate Effort Assign Team Budget Control
Quality Management	Quality Assurance: Quality assurance will be managed including governance, roles and responsibilities, tools and techniques and reporting Quality Control: Specify the mechanisms to be used to measure and control the quality of the work products
Resource Management	Estimate and Manage the need People: People & Skills Required Finance: Budget Required Physical: Facilities, IT Infrastructure
Stakeholder	Identifying, Analyzing, Engaging Stakeholders
Communication Management	Determine communication requirements, roles and responsibilities, tools and techniques. [Type of Communication, Schedule, Mechanism Recipient]
Risk Management	Identifying, analysing, and prioritizing project risks
Procurement Management	Adhering to organization procurement process

2. Estimation

2.1. Effort and Cost Estimation

Activity Description	Sub-Task	Sub-Task Description	Effort (in hours)	Cost in INR
Image is processed	E1R1A1T1 (Effort-Requirement-Activity-Task)	images are clicked	3	3000
	E1R1A1T2	signs in the images are highlighted for processing	6	6000
	E1R1A1T3	images are uploaded	1	1000
Model is trained	E2R1A1T1	model training libraries installed	1	1000
	E2R1A1T2	uploaded images are used for training	3	3000
Object detection libraries are installed	E3R1A1T1	images are processed using the installed libraries	2	2000
Testing of model	E4R1A1T1	model was tested	5	5000

Table 2

Effort (hr)	Cost (INR)
1	1000

2.2. Infrastructure/Resource Cost [CapEx]

Infrastructure Requirement	Qty	Cost per qty	Cost per item
Python	1	0 (open source)	0
Libraries	8	0 (open source)	0
Git terminal	1	0 (open source)	0
Anaconda (Jupyter)	1	0 (open source)	0

Table 3

2.3 Maintenance and Support Cost [OpEx]

Category	Details	Qty	Cost per qty per Quarter	Total cost
People	Technical Developer Project Manager Business Analyst Tester	4	10000 21000 15000 7000	53000
License	Operating System Database Middleware	4	0	0

	IDE			
Infrastructures	Server, Storage and Network	20	16 GB (open source)	0

Table 4

3. Project Team Formation

3.1. Identification Team members

Name	Role	Responsibilities
Kousika S Pratham Garg Rishabh Agarwal	Key Business User (Product Owner)	Provide clear business and user requirements
Pratham Garg	Project Manager	Manage the project
Kousika S	Business Analyst	Discuss and Document Requirements
Rishabh Agarwal	Technical Lead	Design the end-to-end architecture
Rishabh Agarwal	Frontend Developer	Develop user interface
Rishabh Agarwal	Backend Developer	Design, Develop and Unit Test Services/API/DB
Kousika S Pratham Garg	Tester	Define Test Cases and Perform Testing

Table 5

3.2. Responsibility Assignment Matrix

RACI Matrix	Team Members			
Activity	**Name (BA)**	**Name (Developer)**	**Name (Project Manager)**	**Key Business User**
User Requirement Documentation	Kousika S (A)	Rishabh Agarwal (C/I)	Pratham Garg I	Kousika S Pratham Garg Rishabh Agarwal (I)
planning documentation	Kousika S (A)	Rishabh Agarwal (C/I)	Pratham Garg I	Kousika S Pratham Garg Rishabh Agarwal (I)
Resource Management Documentation	Kousika S (A)	Rishabh Agarwal (C/I)	Pratham Garg I	Kousika S Pratham Garg Rishabh Agarwal (I)
Cost Estimation Management	Kousika S (A)	Rishabh Agarwal (C/I)	Pratham Garg I	Kousika S Pratham Garg Rishabh Agarwal (I)

Table 6

A	Accountable
R	Responsible
C	Consult
I	Inform

Result: Thus, the Project Plan was documented successfully done.

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	5
Title of Experiment	Prepare Work breakdown structure, Timeline chart, Risk identification table
Name of the candidate	RISHABH AGRAWAL (RA2111003011564)
Team Members	KOUSIKA S (RA2111003011577) Pratham Garg (RA2111003011584)
Register Number	RA2111003011564,RA2111003011577,RA2111003011584
Date of Experiment	13/02/2023

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To Prepare Work breakdown structure, Timeline chart and Risk identification table

Team Members:

Sl No	Register No	Name	Role
1	RA2111003011564	Rishabh Agarwal	Rep
2	RA2111003011577	Kousika S	Member
3	RA2111003011584	Pratham Garg	Member

WBS

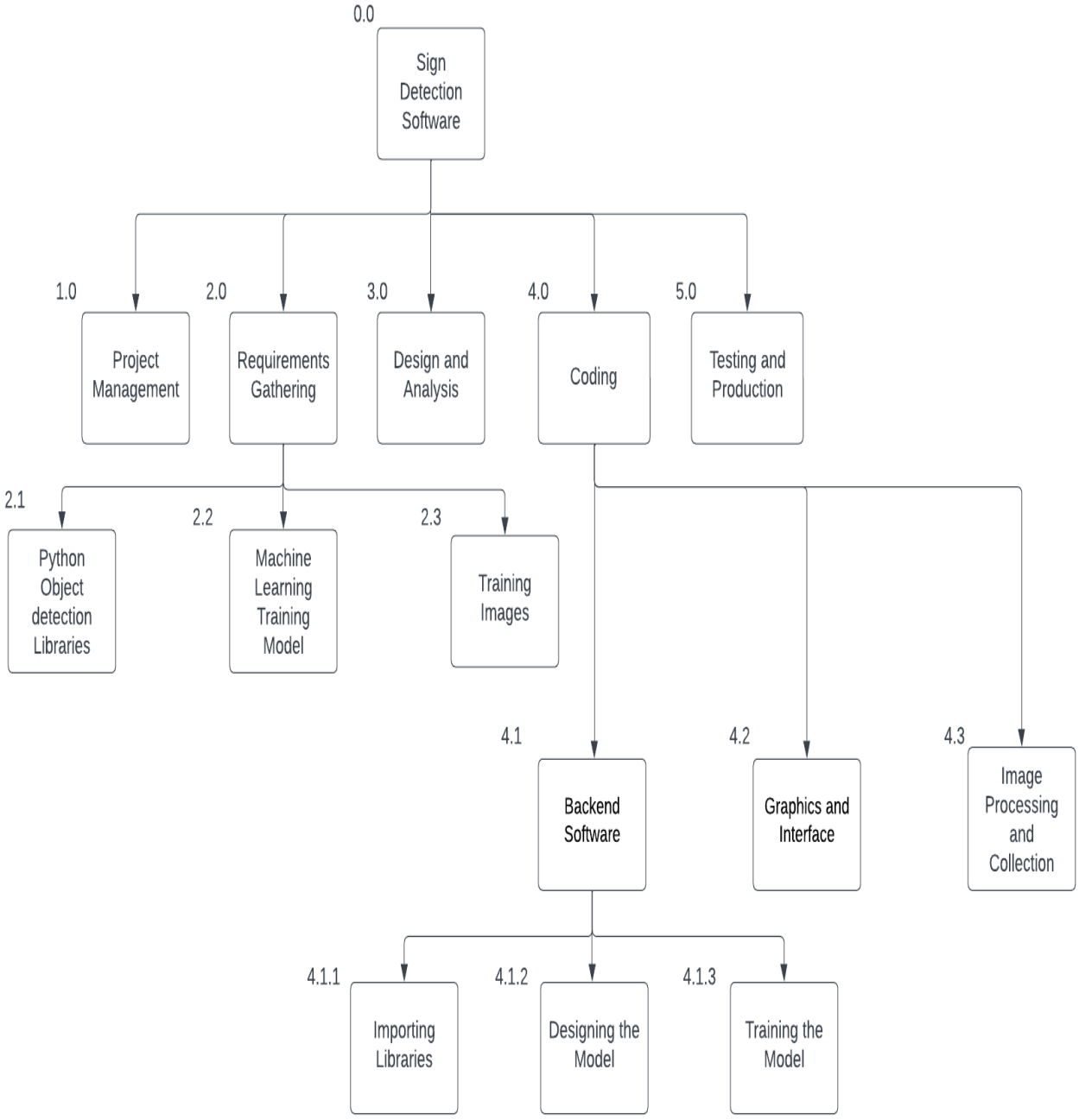

0.0
Sign Detection Software
1.0
Project Management
2.0
Requirements Gathering
3.0
Design and Analysis
4.0
Coding
5.0
Testing and Production
2.1
Python Object detection Libraries
2.2
Machine Learning Training Model
2.3
Training Images
4.1
Backend Software
4.2
Graphics and Interface
4.3
Image Processing and Collection
4.1.1
Importing Libraries
4.1.2
Designing the Model
4.1.3
Training the Model

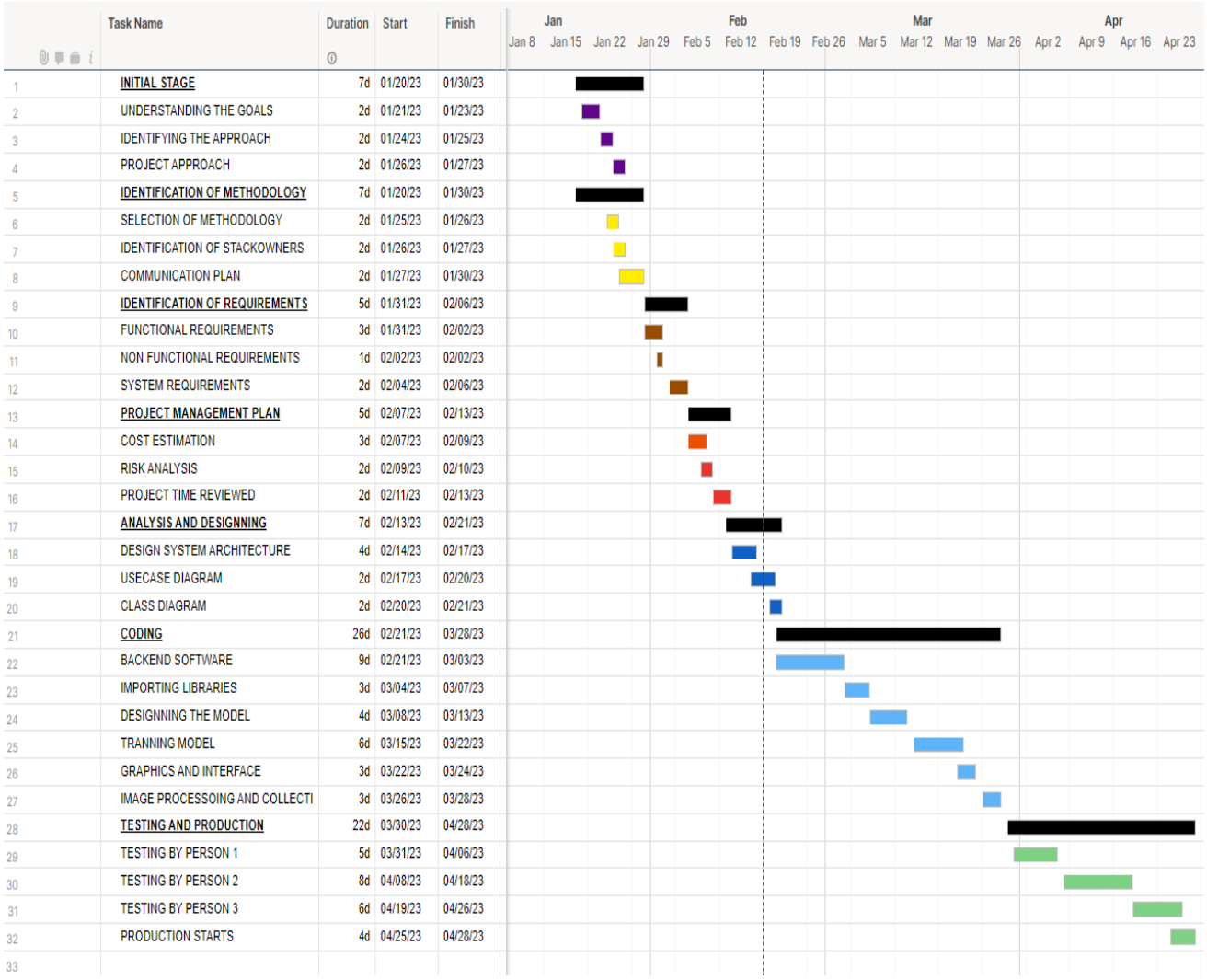
Fig no. 1

☒ 0.0 Sign Detection Software
☒ 1.0 Project Management
☒ 2.0 Requirements Gathering
- 2.1 Python Object detection Libraries
- 2.2 Machine Learning Training Model
- 2.3 Training Images

☒ 3.0 Analysis & Design
☒ 4.0 Coding
- 4.1 Backend Software
  - 4.1.1 Importing Libraries
  - 4.1.2 Designing the Model
  - 4.1.3 Training the Model
- 4.2 Graphics and Interface
- 4.3 Image Processing and Collection

☒ 5.0 Testing and Production

TIMELINE – GANTT CHART



	Task Name	Duration	Start	Finish
1	INITIAL STAGE	7d	01/20/23	01/30/23
2	UNDERSTANDING THE GOALS	2d	01/21/23	01/23/23
3	IDENTIFYING THE APPROACH	2d	01/24/23	01/25/23
4	PROJECT APPROACH	2d	01/26/23	01/27/23
5	IDENTIFICATION OF METHODOLOGY	7d	01/20/23	01/30/23
6	SELECTION OF METHODOLOGY	2d	01/25/23	01/26/23
7	IDENTIFICATION OF STACKOWNERS	2d	01/26/23	01/27/23
8	COMMUNICATION PLAN	2d	01/27/23	01/30/23
9	IDENTIFICATION OF REQUIREMENTS	5d	01/31/23	02/06/23
10	FUNCTIONAL REQUIREMENTS	3d	01/31/23	02/02/23
11	NON FUNCTIONAL REQUIREMENTS	1d	02/02/23	02/02/23
12	SYSTEM REQUIREMENTS	2d	02/04/23	02/06/23
13	PROJECT MANAGEMENT PLAN	5d	02/07/23	02/13/23
14	COST ESTIMATION	3d	02/07/23	02/09/23
15	RISK ANALYSIS	2d	02/09/23	02/10/23
16	PROJECT TIME REVIEWED	2d	02/11/23	02/13/23
17	ANALYSIS AND DESIGNNING	7d	02/13/23	02/21/23
18	DESIGN SYSTEM ARCHITECTURE	4d	02/14/23	02/17/23
19	USECASE DIAGRAM	2d	02/17/23	02/20/23
20	CLASS DIAGRAM	2d	02/20/23	02/21/23
21	CODING	26d	02/21/23	03/28/23
22	BACKEND SOFTWARE	9d	02/21/23	03/03/23
23	IMPORTING LIBRARIES	3d	03/04/23	03/07/23
24	DESIGNNING THE MODEL	4d	03/08/23	03/13/23
25	TRANNING MODEL	6d	03/15/23	03/22/23
26	GRAPHICS AND INTERFACE	3d	03/22/23	03/24/23
27	IMAGE PROCESSOING AND COLLECTI	3d	03/26/23	03/28/23
28	TESTING AND PRODUCTION	22d	03/30/23	04/28/23
29	TESTING BY PERSON 1	5d	03/31/23	04/06/23
30	TESTING BY PERSON 2	8d	04/08/23	04/18/23
31	TESTING BY PERSON 3	6d	04/19/23	04/26/23
32	PRODUCTION STARTS	4d	04/25/23	04/28/23
33				

Fig no. 2

SWOT ANALYSIS:

Strengths:	**Weaknesses:**
● **Sign detection technology can improve communication and understanding for the deaf and dumb community.** ● **This technology can reduce the communication gap between the hearing and non-hearing community.** ● **Sign detection can be used in various settings like schools, workplaces, and public places.**	● **Sign language has various dialects and can vary from country to country, making it challenging to create a universal sign detection system.** ● **Sign detection technology may not be accessible or affordable for all members of the deaf and dumb community.** ● **The accuracy of sign detection technology can be affected by environmental factors such as lighting, background, and noise.**
Opportunities:	**Threats:**
● **Sign detection technology can be integrated into various devices such as smartphones, tablets, and smartwatches, providing easy and convenient communication for the deaf and dumb community.** ● **The demand for sign detection technology is increasing, creating opportunities for innovation and development.** ● **Sign detection technology can be used for language translation, making it easier for the hearing community to communicate with the non-hearing community.**	● **The development and maintenance of sign detection technology can be expensive, making it difficult for small organizations and individuals to afford.** ● **The accuracy of sign detection technology may be affected by changes in sign language and the emergence of new signs, requiring frequent updates and maintenance.** ● **The adoption of sign detection technology may face resistance or discrimination from individuals or groups who do not understand or support the needs of the deaf and dumb community.**

Risk Mitigation, Monitoring, and Management

Response	Strategies
Avoid	• **Gather high-quality data**: High-quality data is essential for accurate sign detection. Ensure that the data used for training the algorithm is diverse, representative of the sign's appearance in different lighting conditions, angles, and backgrounds. • **User feedback:** Collect feedback from users of the software to identify any issues or errors that may arise in real-world use. • **Use multiple algorithms**: Use multiple algorithms to detect signs to avoid false positives and negatives. Combining the results of different algorithms can help increase the overall accuracy.
Transfer	• **Contractual agreements**: Use contractual agreements to transfer the risk of developing sign detection software to a third party. The agreement can outline the responsibilities of each party, including the allocation of risks and liabilities. • **Insurance:** Purchase insurance to cover the risks associated with developing sign detection software. The insurance can cover issues such as inaccurate detection, false positives, and false negatives. • **Outsourcing:** Consider outsourcing the development of sign detection software to a third-party vendor. This can transfer the risk of development to the vendor, who has the expertise and resources to develop the software.
Mitigate	• **Implement quality control measures**: Implement rigorous quality control measures to ensure that the sign detection software meets the required accuracy standards. This can include regular testing, validation, and verification. • **Conduct a risk assessment**: Conduct a risk assessment to identify potential risks associated with developing sign detection software. The assessment can identify areas where the software is most vulnerable and highlight potential issues that need to be addressed. • **Involve stakeholders:** Involve stakeholders in the development process to ensure that the software meets the needs of the target audience. This can include sign language experts, deaf and hard-of-hearing individuals, and other groups who may use the software.
Accept	• **Monitor and track risks**: Monitor and track risks associated with the development process and take appropriate measures to manage them. This can include implementing contingency plans, adjusting the software's features or functionality, or revising the risk management plan as needed. • **Accept the consequences of risks**: Accept the consequences of potential risks and be prepared to address any issues that arise

	during the development process. This can include accepting responsibility for any errors or inaccuracies in the software and taking steps to rectify them. ● **Set realistic expectations**: Set realistic expectations for the accuracy and performance of the sign detection software. This can help to manage user expectations and reduce the risk of disappointment or negative feedback.

Table 7

Result: Thus, the work breakdown structure with timeline chart and risk table were formulated successfully.

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	6
Title of Experiment	Design a System Architecture, Use Case and Class Diagram
Name of the candidate	RISHABH AGRAWAL (RA2111003011564)
Team Members	KOUSIKA S (RA2111003011577) PRATHAM GARG (RA2111003011584)
Register Number	RA2111003011564, RA2111003011577, RA2111003011584
Date of Experiment	20/02/2023

Mark Split Up

S.No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To Design a System Architecture, Use case and Class Diagram

Team Members:

Sl No	Register No	Name	Role
1	RA2111003011564	Rishabh Agarwal	Rep
2	RA2111003011577	Kousika S	Member
3	RA2111003011584	Pratham Garg	Member

SYSTEM ARCHITECTURE

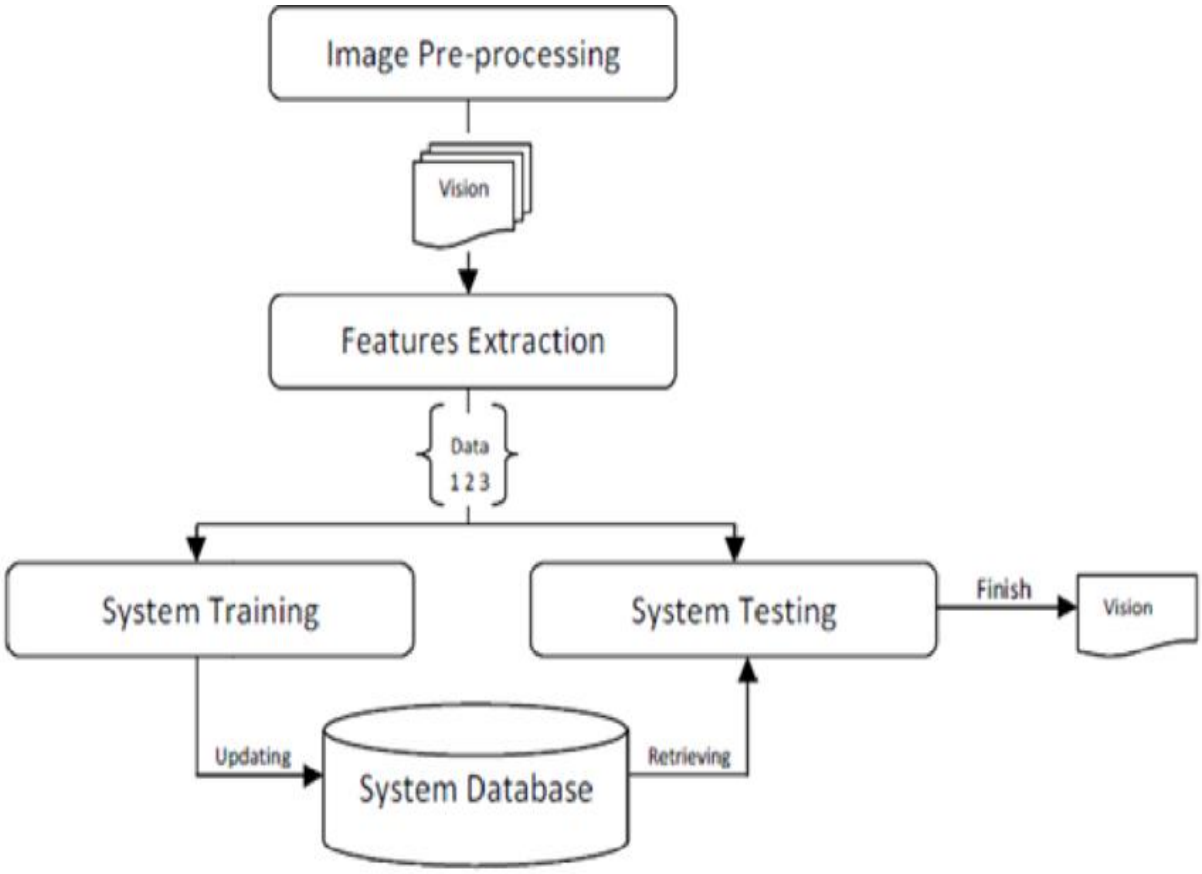


Fig no. 3

USE CASE DIAGRAM

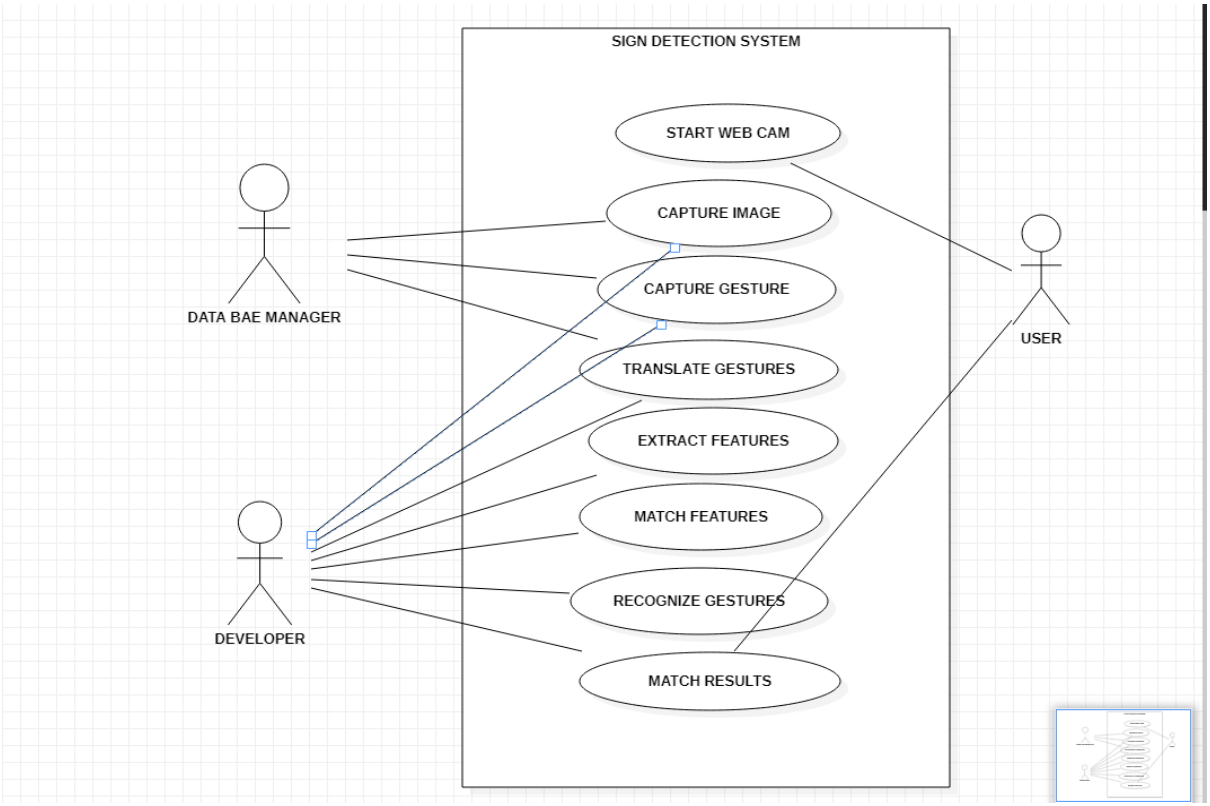


Fig no. 4

CLASS DIAGRAM

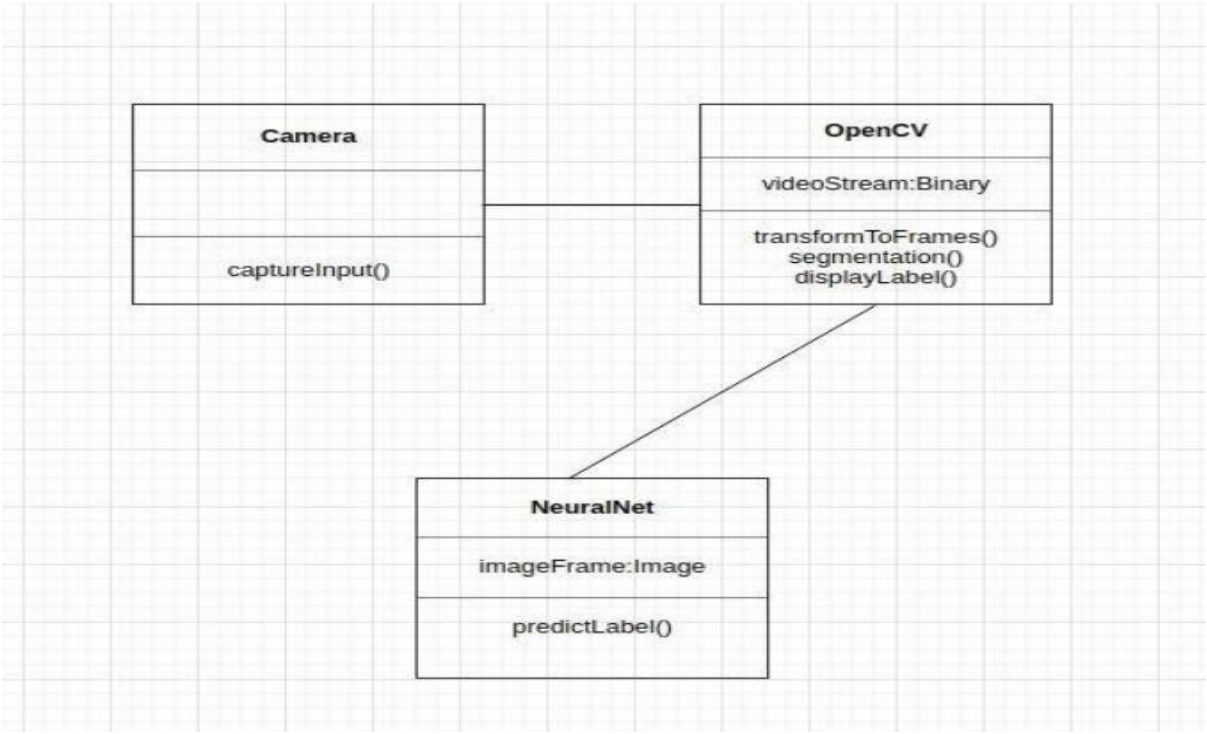


Fig no. 5

Result: Thus, the system architecture, use case and class diagram created successfully.

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	7
Title of Experiment	Design a Entity relationship diagram
Name of the candidate	Rishabh Agrawal
Team Members	Pratham Garg, Kousika.S
Register Number	RA2111003011577, RA2111003011564, RA2111003011584
Date of Experiment	6/3/23

Mark Split Up

S. No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To create the Entity Relationship Diagram

Team Members:

S No	Register No	Name	Role
1	RA2111003011564	Rishabh Agrawal	Rep
2	RA2111003011577	Kousika.S	Member
3	RA2111003011584	Pratham garg	Member

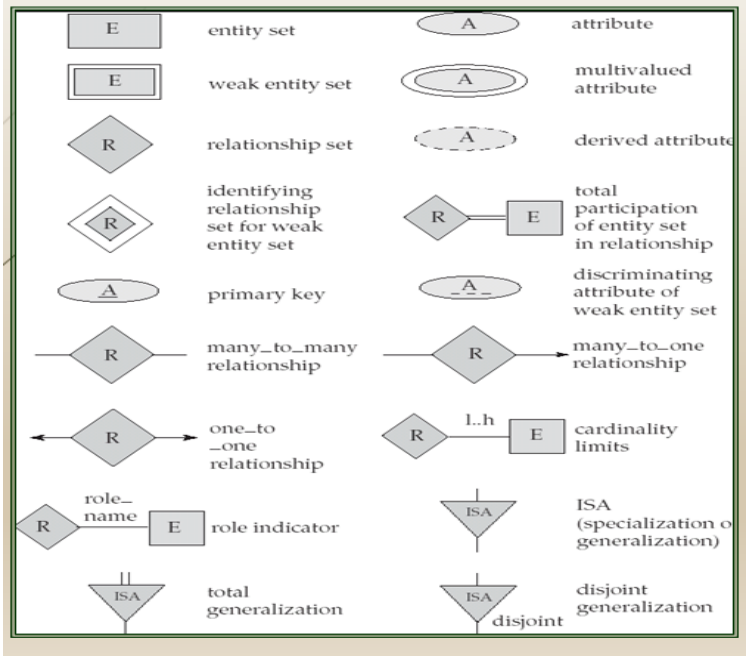


Fig no. 6

ER Diagram of sign language detection system:-

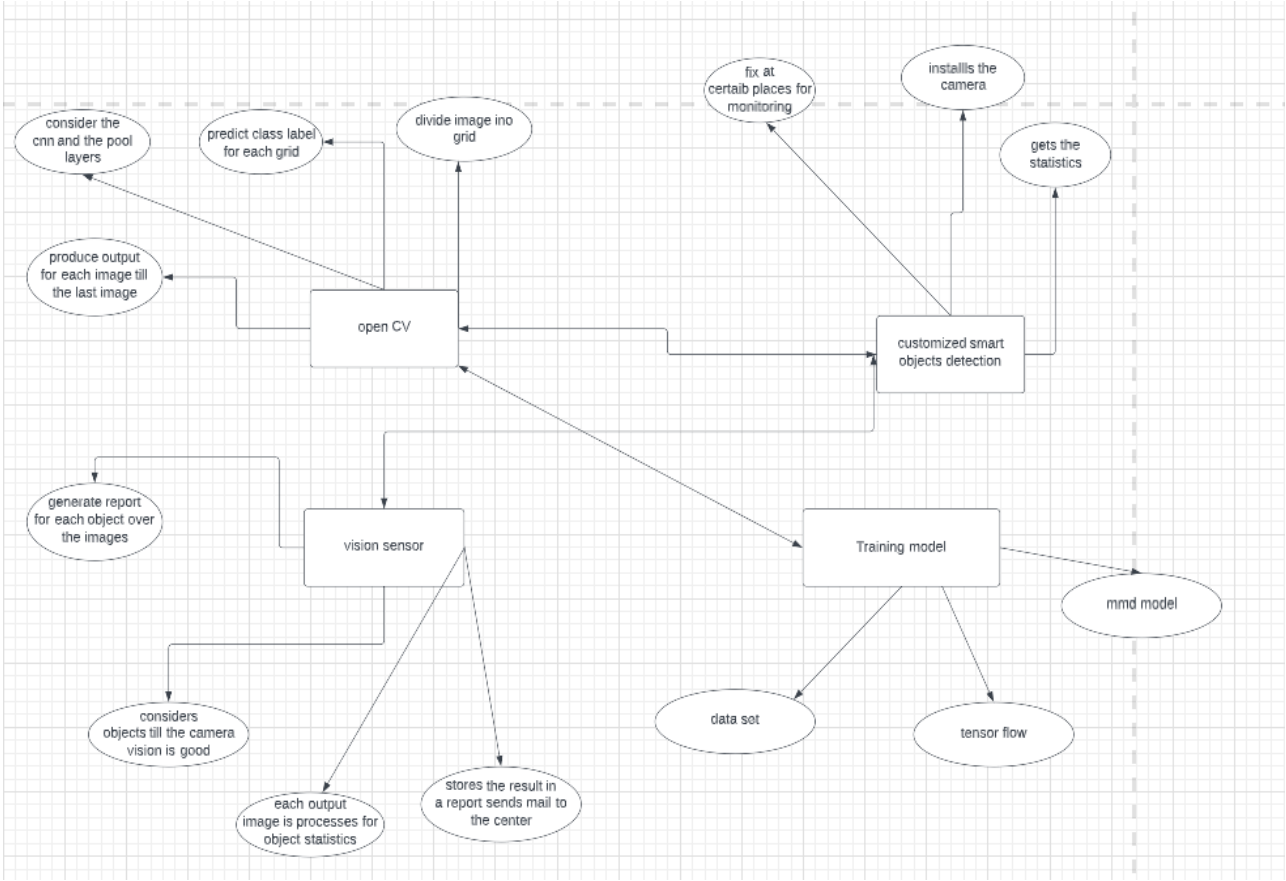


Fig no. 7

Result: Thus, the entity relationship diagram was created successfully.

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	8
Title of Experiment	Develop a Data Flow Diagram (Process-Up to Level 1)
Name of the candidate	RISHABH AGRAWAL (RA2111003011564)
Team Members	KOUSIKA S (RA2111003011577) PRATHAM GARG (RA2111003011584)
Register Number	RA2111003011584, RA2111003011577, RA2111003011564
Date of Experiment	

Mark Split Up

S. No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To develop the data flow diagram up to level 1 for the <project name>

Team Members:

S No	Register No	Name	Role
1	RA2111003011564	Rishabh Agarwal	Rep
2	RA2111003011584	Pratham Garg	Member
3	RA2111003011577	Kousika S	Member

Data Flow Diagram

The DFD takes an input-process-output view of a system. That is, data objects flow into the software, are transformed by processing elements, and resultant data objects flow out of the software. Data objects are represented by labeled arrows, and transformations are represented by circles (also called bubbles). The DFD is presented in a hierarchical fashion. That is, the first data flow model (sometimes called a level 0 DFD or context diagram) represents the system as a whole. Subsequent data flow diagrams refine the context diagram, providing increasing detail with each subsequent level.

The data flow diagram enables you to develop models of the information domain and functional domain. As the DFD is refined into greater levels of detail, you perform an implicit functional decomposition of the system. At the same time, the DFD refinement results in a corresponding refinement of data as it moves through the processes that embody the application.
A few simple guidelines can aid immeasurably during the derivation of a data flow diagram:
(1) Level 0 data flow diagram should depict the software/system as a
single bubble;
(2) Primary input and output should be carefully noted;
(3) Refinement should begin by isolating candidate processes, data objects, and data stores to be represented at the next level;
(4) All arrows and bubbles should be labeled with meaningful names;
(5) Information flow continuity must be maintained from level to level and
(6) One bubble at a time should be refined. There is a natural tendency to overcomplicate the data flow diagram. This occurs when you attempt to show too much detail too early or represent procedural aspects of the software in lieu of information flow.

DFD Level 0

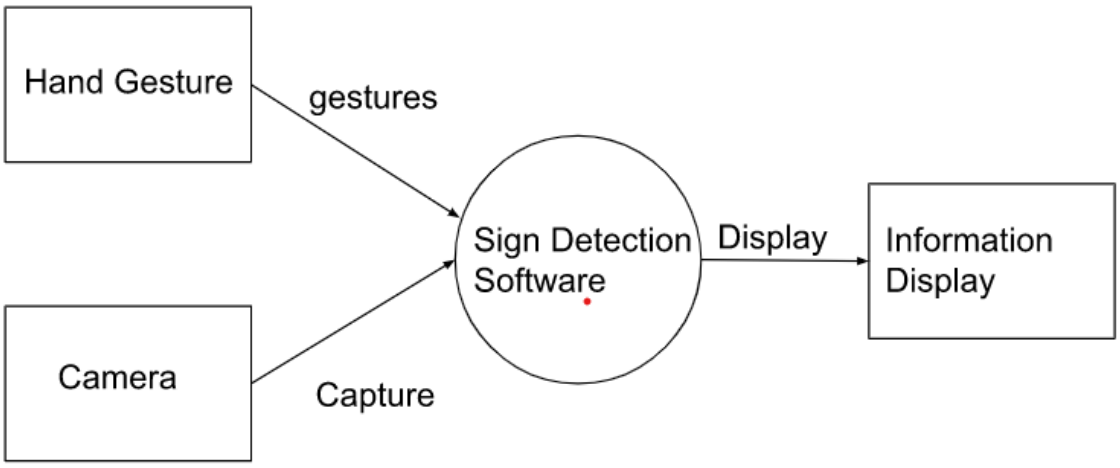


Fig no. 8

DFD Level 1

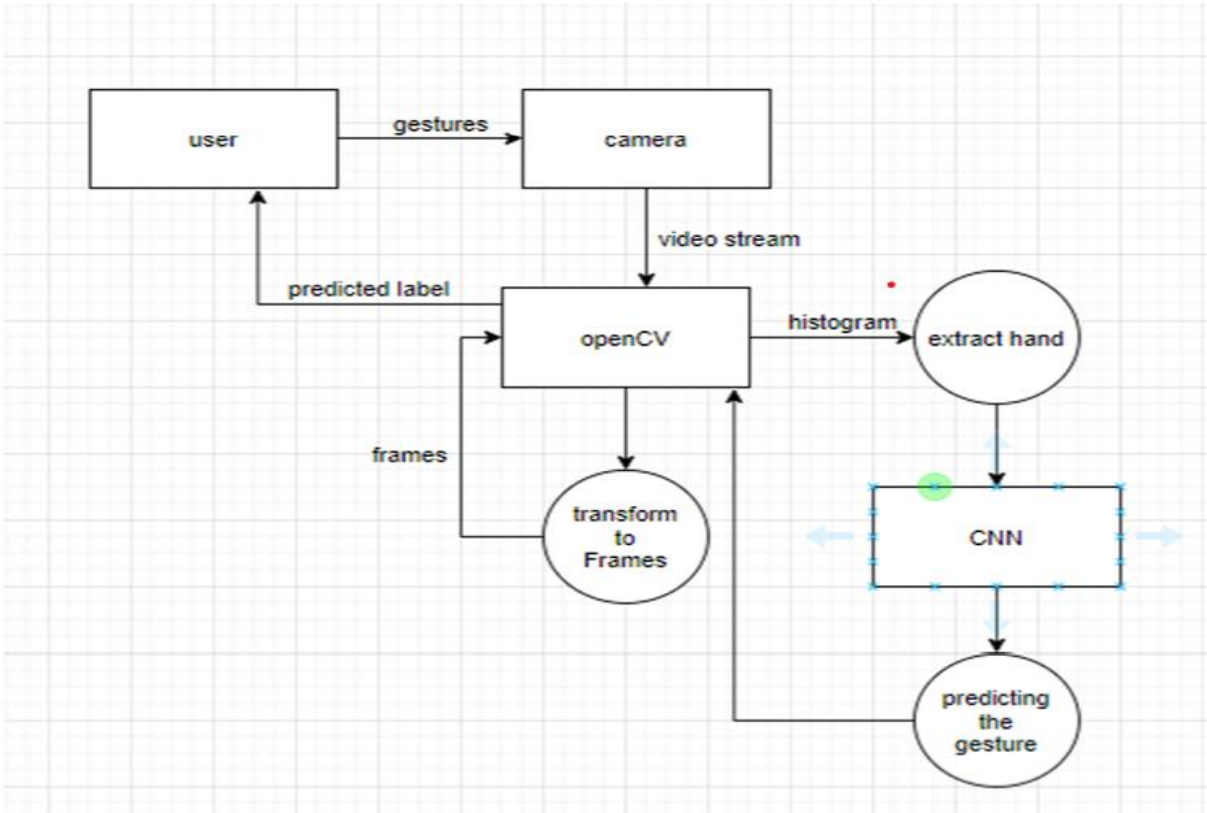


Fig no. 9

Result: Thus, the data flow diagrams have been created for the Sign Detection software.

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	9
Title of Experiment	Design a Sequence and Collaboration Diagram
Name of the candidate	RISHABH AGRAWAL (RA2111003011564)
Team Members	KOUSIKA S (RA2111003011577) PRATHAM GARG (RA2111003011584)
Register Number	RA2111003011564, RA2111003011577, RA2111003011584
Date of Experiment	14/03/2023

Mark Split Up

S. No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To create the sequence and collaboration diagram for the Sign Detection Software

Team Members:

S No	Register No	Name	Role
1	RA2111003011564	Rishabh Agarwal	Rep/Member
2	RA2111003011577	Kousika S	Member
3	RA211003011584	Pratham Garg	Member

Sequence Diagram

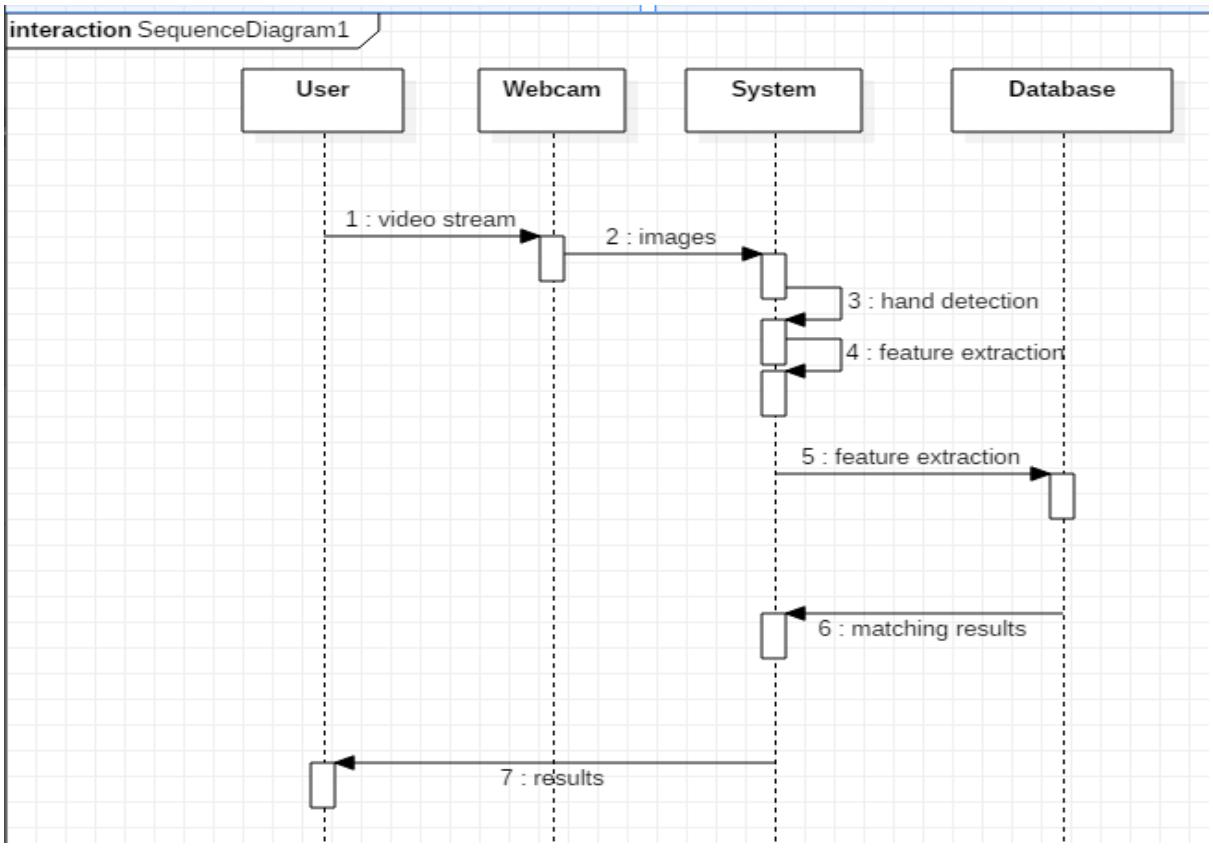


Fig no. 10

Collaboration Diagram

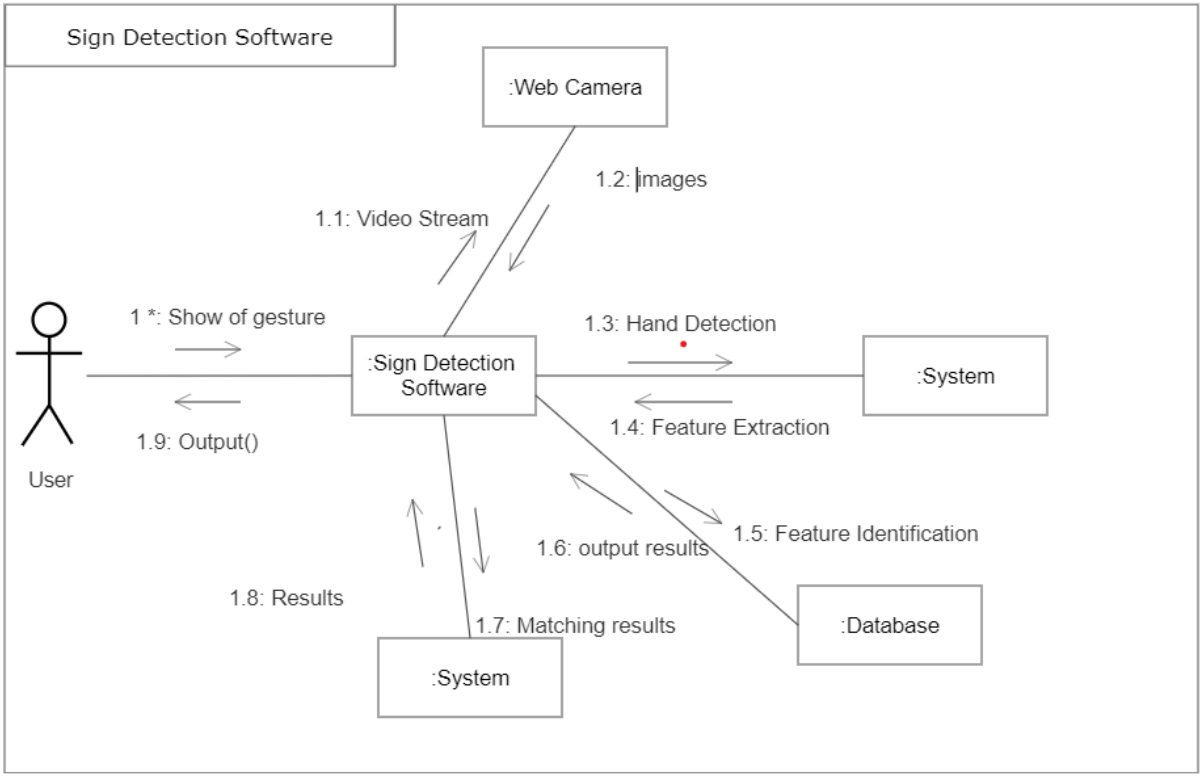


Fig no. 11

Result: Thus, the sequence and collaboration diagrams were created for the Sign Detection Software.

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	10
Title of Experiment	Develop a Testing Framework/User Interface
Name of the candidate	RISHABH AGRAWAL
Team Members	PRATHAM GARG , KOUSIKA.S
Register Number	RA2111003011564, RA2111003011584,RA2111003011577
Date of Experiment	29.03.2003

Mark Split Up

S. No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To develop the testing framework and/or user interface framework for the Sign language detection software.

Team Members:

S No	Register No	Name	Role
1	RA2111003011564	RISHABH AGRAWAL	Rep/Member
2	RA2111003011584	PRATHAM GARG	Member
3	RA2111003011577	KOUSIKA.S	Member

Open CV framework is used for user interface.

Executive Summary

Objective: The objective of testing sign detection software for deaf and dumb is to ensure its accuracy and reliability in recognizing and interpreting sign language gestures. The software should be able to detect a wide range of signs accurately, even with variations in lighting and background. Additionally, the software should be user-friendly and easy to operate for individuals with hearing and speech impairments. Ultimately, the goal is to create a tool that can facilitate communication and improve the quality of life for deaf and dumb individuals.

Approach: To test sign detection software for deaf and dumb, a comprehensive approach should be adopted. The software should be tested against a diverse range of sign languages and signs, ensuring that it can detect and interpret them accurately. Real-world scenarios should be simulated to test the software's reliability and responsiveness in different lighting conditions, distances, and angles. Testing should also include different hardware configurations to ensure compatibility with various devices.

Scope of Testing

Sign detection technology can enable deaf and mute individuals to communicate more effectively by recognizing and translating sign language into text or speech.

This technology has the potential to increase accessibility for people with hearing and speech disabilities in various settings such as schools, workplaces, and public spaces.

Sign detection can be used in real-time communication such as video calls, live events, and public announcements, improving the inclusivity and engagement of deaf and mute individuals in society.

The scope of sign detection technology is not limited to English or American Sign Language (ASL) but can be expanded to other sign languages used globally, enabling cross-cultural communication.

Additionally, sign detection can be integrated with other technologies such as machine learning and artificial intelligence, enhancing its accuracy and effectiveness over time.

Types of Testing, Methodology, Tools

Category	Methodology	Tools Required
FUNCTIONAL REQUIREMENTS 1. Image and video capture: 2. Image and video processing 3. Object detection: 4. Object recognition 5. Data storage	Manual Automated Manual Manual Manual	**Bugzilla**

NON FUNCTIONAL REQUIREMENTS 1. Usability: 2. Performance: 3. Reliability: 4. Accessibility: 5. Scalability:	Manual Manual Manual Manual Manual	

Table 8

Result: Thus, the testing framework/user interface framework has been created for the Sign language detection software.

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	11
Title of Experiment	Test Cases
Name of the candidate	RISHABH AGRAWAL
Team Members	KOUSIKA.S, PRATHAM GARG
Register Number	RA2111003011564,RA2111003011577,RA2111003011584,
Date of Experiment	5.4.23

Mark Split Up

S. No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To develop the test cases manual for the sign detection of the deaf and dumb

Team Members:

S No	Register No	Name	Role
1	RA2111003011564	RISHABH AGRAWAL	**Rep**
2	RA2111003011577	KOUSIKA.S	**Member**
3	RA2111003011584	PRATHAM GARG	**Member**

Test Case

Functional Test Cases

Test ID (#)	Test Scenario	Test Case	Execution Steps	Expected Outcome	Actual Outcome	Status	Remarks
1.	Sign detection accuracy	Test the accuracy of the system in detecting signs of gesture and orientations. This can be done by presenting a range of signs to the system and verifying if it correctly detects them.	1.compile the code 2.execute the code 3.open system camera 4.show hand gestures within camera frame 5.Test the detected sign	The system should detect the gestures with over 80% accuracy	detected with 80%accuracy	Pass	success
2.	Sign recognition speed	Test the speed at which the system recognizes the signs. This can be done by presenting a series of signs and measuring the time it takes for the system to detect and recognize them.	1.compile the code 2.execute the code 3.open system camera 4.show hand gestures within camera frame	The system should detect the speed with over 70% accuracy	detected with 70%accuracy	Pass	success

			5.Test the detected sign and its speed				
3.	Sign classification accuracy	Test the accuracy of the system in classifying signs into different categories such as alphabets, numbers, and words. This can be done by presenting a range of signs and verifying if the system correctly classifies them.	1.compile the code 2.execute the code 3.open system camera 4.show hand gestures within camera frame 5.Test the detected sign and its speed with accuracy	The system should detect the speed with over 85% accuracy	detected with 85%accuracy	Pass	success
4.	Sign recognition under different hand positions	Test the ability of the system to detect signs when the hands are in different positions. This can be done by presenting a range of signs with hands in different positions and verifying if the system correctly detects them.	1.compile the code 2.execute the code 3.open system camera 4.show hand gestures within camera frame 5.Test the detected sign and its speed with accuracy	The system should detect the hand position with over 85% accuracy	detected with 85%accuracy	Pass	success
5.	Sign recognition with different camera angles	Test the ability of the system to detect signs from different camera angles such as from above or from the side. This can be done by presenting a range of signs from different angles and verifying if the system correctly detects them.	1.compile the code 2.execute the code 3.open system camera 4.show hand gestures within camera frame 5.Test the detected sign and its speed with accuracy and with	The system should detect the different camera angles over 70%accuracy	detected with 70%accuracy	Pass	success

			different camera angles				

Table 9

Non-Functional Test Cases

Test ID (#)	Test Scenario	Test Case	Execution Steps	Expected Outcome	Actual Outcome	Status	Remarks
1.	Usability	Test the usability of the system by evaluating the ease of use and the user interface of the system. This can be done by conducting user testing sessions and collecting feedback from the users.	The usability of the system is checked by executing and working on the system	we get the expected usability over 70% accuracy	detected with 70%accuracy	pass	success
2.	Performance:	Test the performance of the system by evaluating the speed and responsiveness of the system. This can be done by measuring the time it takes for the system to detect and recognize signs and the time it takes for the system to respond to user inputs.	the user will change the gesture soo fast and check it	we get the expected performanceover 90% accuracy	detected with 90%accuracy	pass	success
3.	Scalability:	Test the scalability of the system by evaluating its ability to handle an increasing number of users and sign recognition requests.	the software is tested with different types of inputs with increasing complexity	we get the expected performance over 70% accuracy	detected with 80%accuracy	pass	success

		This can be done by testing the system with a large number of users simultaneously and monitoring the system's performance.					
4.	Security	Test the security of the system by evaluating its ability to protect the user's sign data and personal information. This can be done by conducting security testing and identifying potential vulnerabilities in the system.	the user checks the system security by conducting the security testing	we get the expected performanc e over 70% accuracy	detected with 40%accuracy	Fail	Failure
5.	Reliabilit y	Test the reliability of the system by evaluating its ability to detect signs accurately and consistently. This can be done by conducting regression testing and verifying if the system consistently detects signs over time.	the user checks the reliability by checking with the system if it detects the signs consistently	we get the expected reliability over 70% accuracy	detected with 70%accuracy	pass	success

Table 10

Result: Thus, the test case manual has been created for the sign detection for deaf and dumb

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	12
Title of Experiment	Manual Test Case Reporting
Name of the candidate	RISHABH AGRAWAL
Team Members	KOUSIKA.S, PRATHAM GARG
Register Number	RA2111003011564, RA2111003011577, RA2111003011584
Date of Experiment	12/04/2023

Mark Split Up

S. No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To prepare the manual test case report for the Sign Detection Software.

Team Members:

S No	Register No	Name	Role
1	RA2111003011564	RISHABH AGRAWAL	**Rep/Member**
2	RA2111003011577	KOUSIKA.S	**Member**
3	RA2111003011584	PRATHAM GARG	**Member**

Category	Progress Against Plan	Status
Functional Testing	Green	Completed
Non-Functional Testing	Amber	In progress

Table 11

Functional (Test ID)	Test Case Coverage (%)	Status
1	80%	Completed
2	70%	In progress
3	85%	Completed
4	85%	Completed
5	70%	In progress

Table 12

Non - Functional (Test ID)	Test Case Coverage (%)	Status
1	70%	Completed
2	90%	Completed
3	80%	Completed
4	40%	Not Required
5	70%	Completed

Table 13

Result: Thus, the test case report has been created for the Sign Detection Software.

School of Computing

SRM IST, Kattankulathur – 603 203

Course Code: 18CSC206J

Course Name: Software Engineering and Project Management

Experiment No	13
Title of Experiment	Provide the details of Architecture Design/Framework/Implementation
Name of the candidate	RISHABH AGRAWAL (RA2111003011564)
Team Members	KOUSIKA S (RA2111003011577) PRATHAM GARG (RA2111003011584)
Register Numbers	RA2111003011564, RA2111003011577, RA2111003011584
Date of Experiment	13/4/23

Mark Split Up

S. No	Description	Maximum Mark	Mark Obtained
1	Exercise	5	
2	Viva	5	
	Total	**10**	

Staff Signature with date

Aim: To provide the details of architectural design/framework/implementation of the project sign detection of the deaf and dumb

Team Members:

S No	Register No	Name	Role
1	RA2111003011564	Rishabh Agarwal	Rep/Member
2	RA2111003011577	Kousika S	Member
3	RA2111003011584	Pratham Garg	Member

SOFTWARE LAYOUT:-

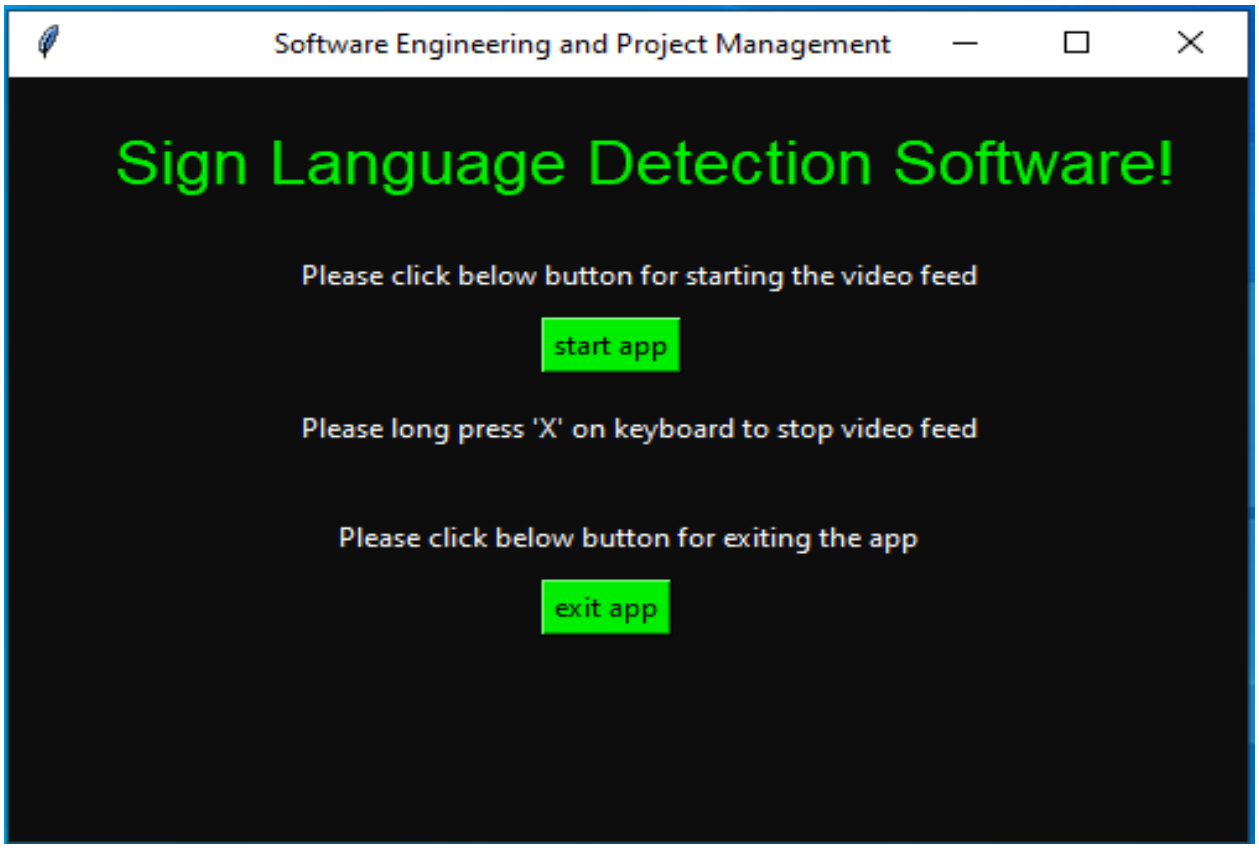


Fig no. 12

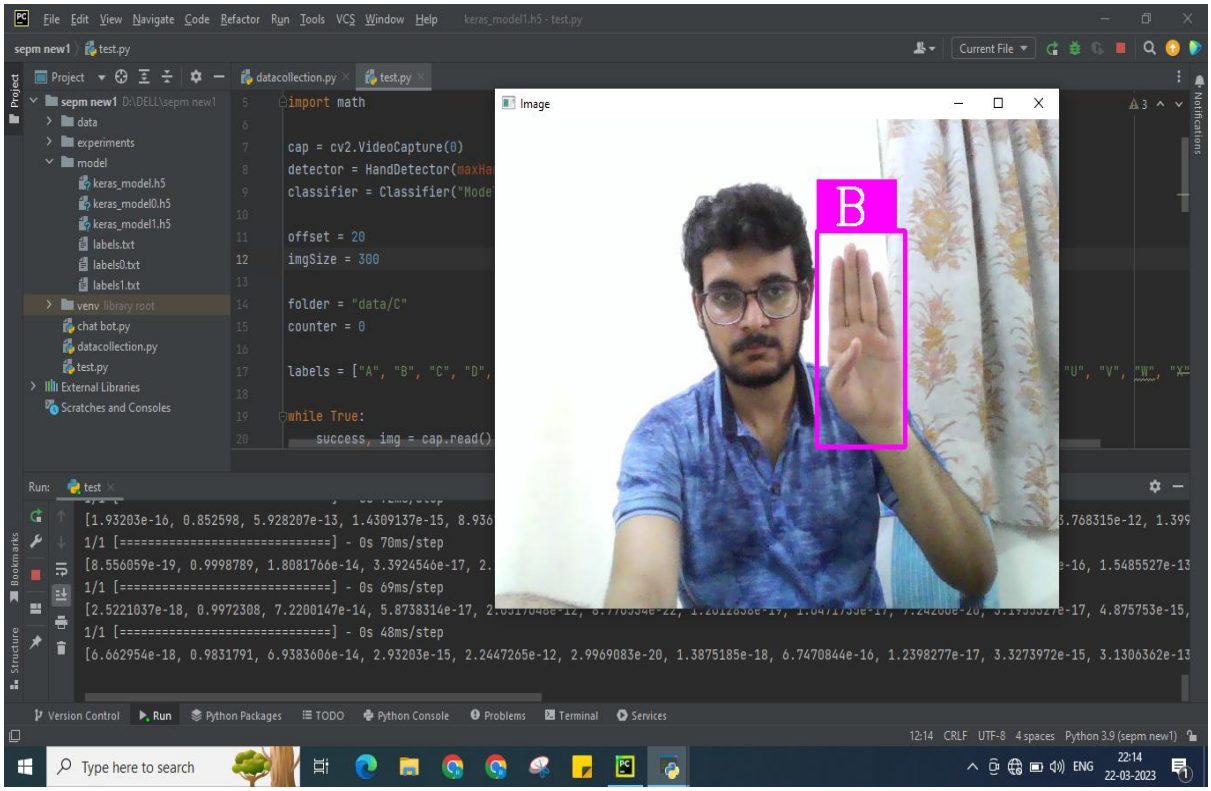


Fig no. 13

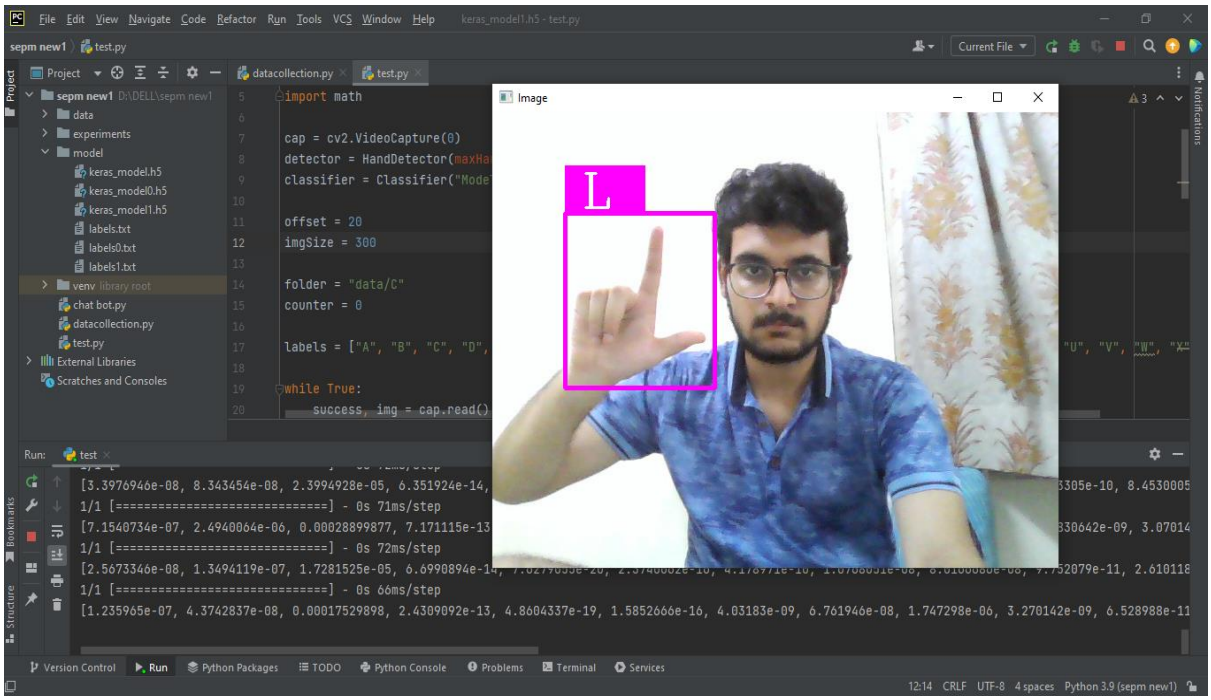


Fig no. 14

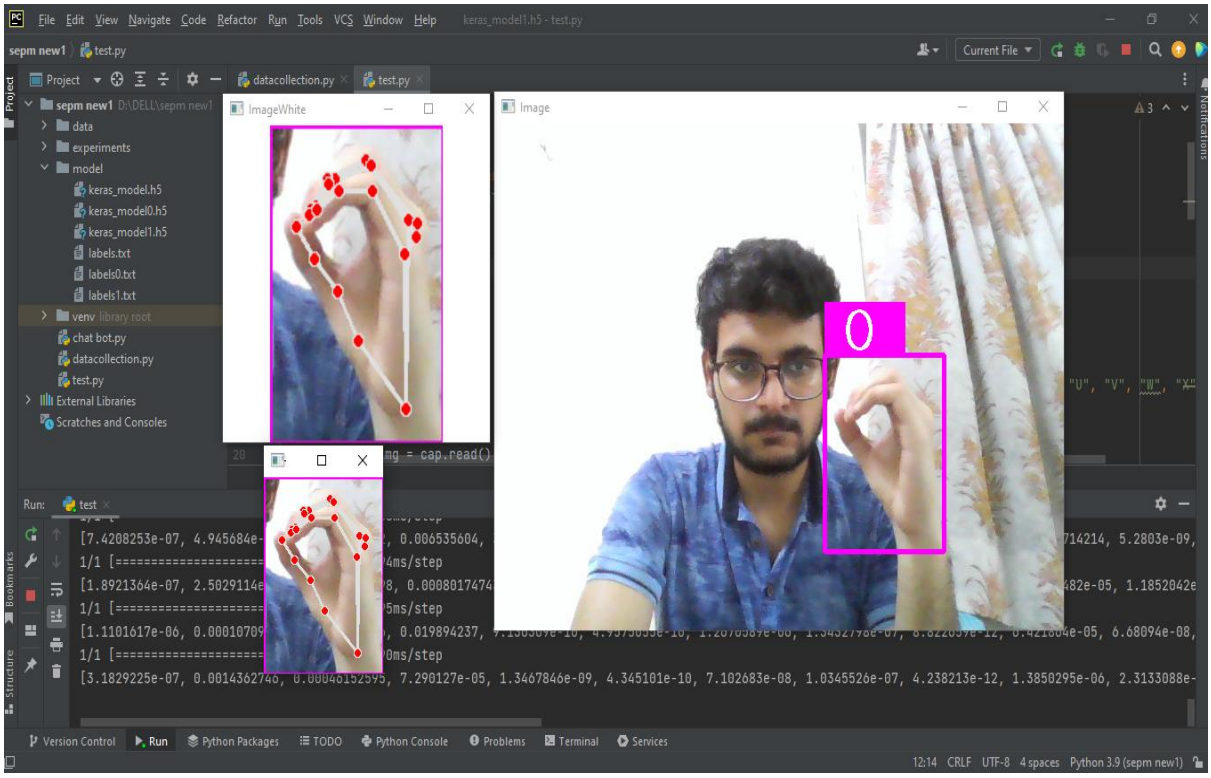


Fig no. 15

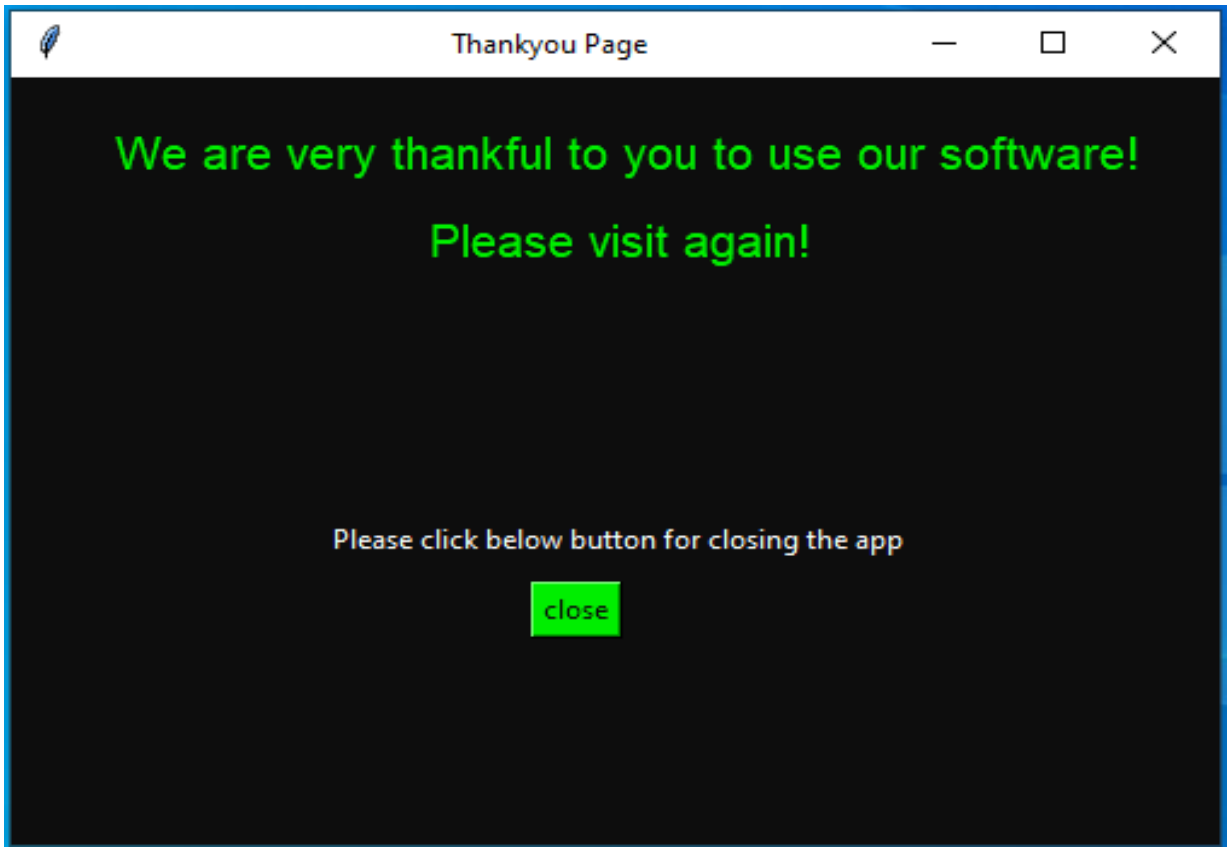
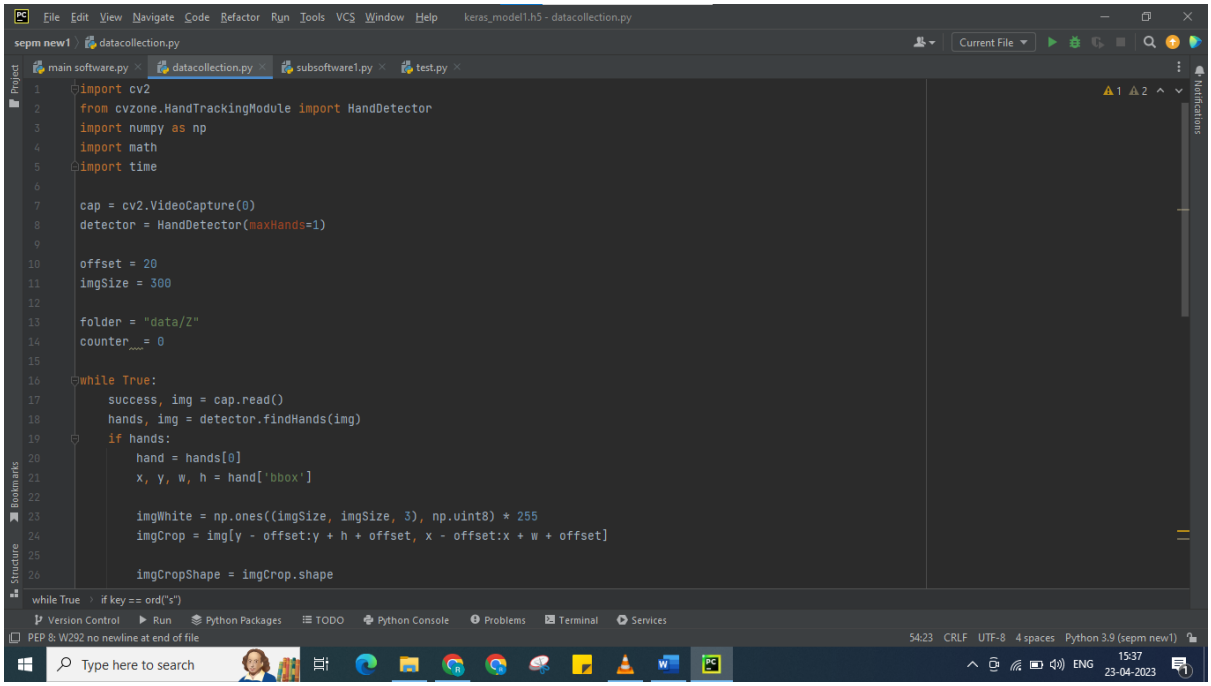


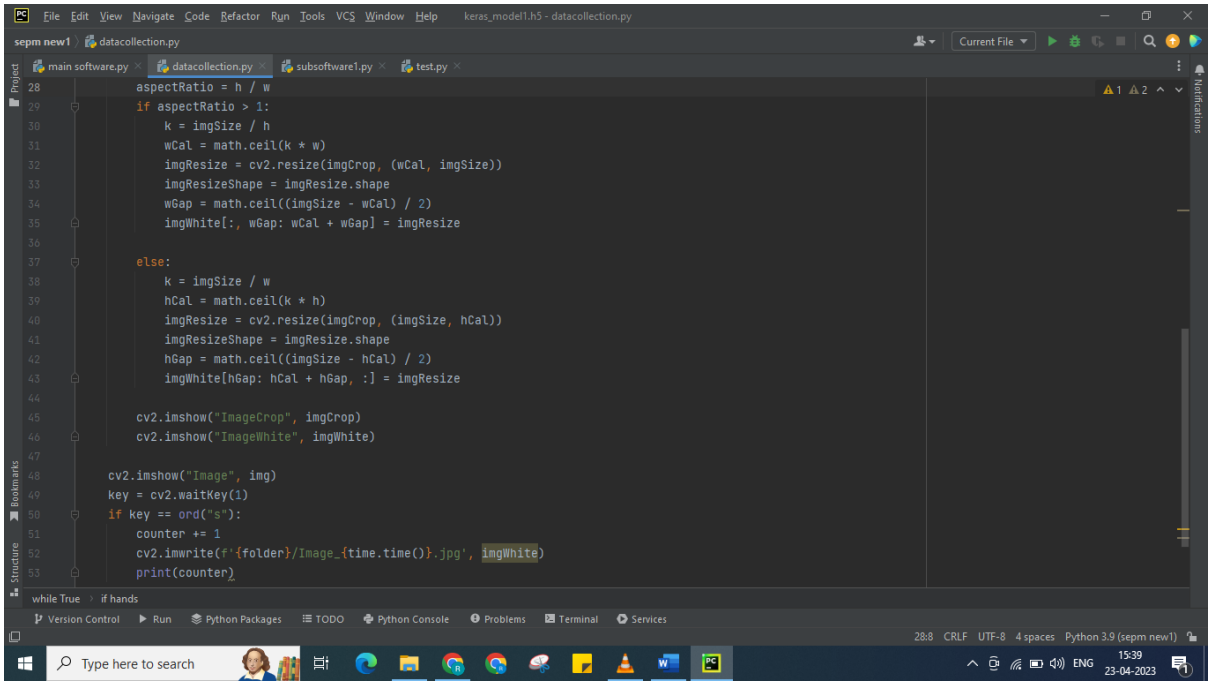
Fig. no. 16

CODE behind the SOFTWARE

Dataset collection code-

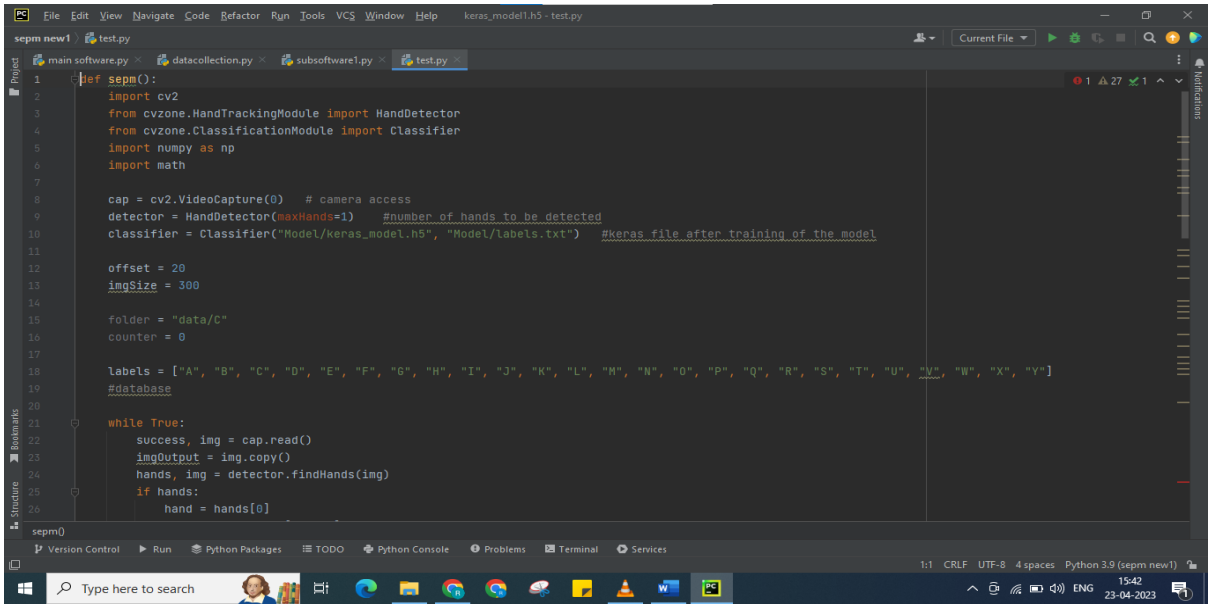


```
import cv2
from cvzone.HandTrackingModule import HandDetector
import numpy as np
import math
import time

cap = cv2.VideoCapture(0)
detector = HandDetector(maxHands=1)

offset = 20
imgSize = 300

folder = "data/Z"
counter = 0

while True:
    success, img = cap.read()
    hands, img = detector.findHands(img)
    if hands:
        hand = hands[0]
        x, y, w, h = hand['bbox']

        imgWhite = np.ones((imgSize, imgSize, 3), np.uint8) * 255
        imgCrop = img[y - offset:y + h + offset, x - offset:x + w + offset]

        imgCropShape = imgCrop.shape
```

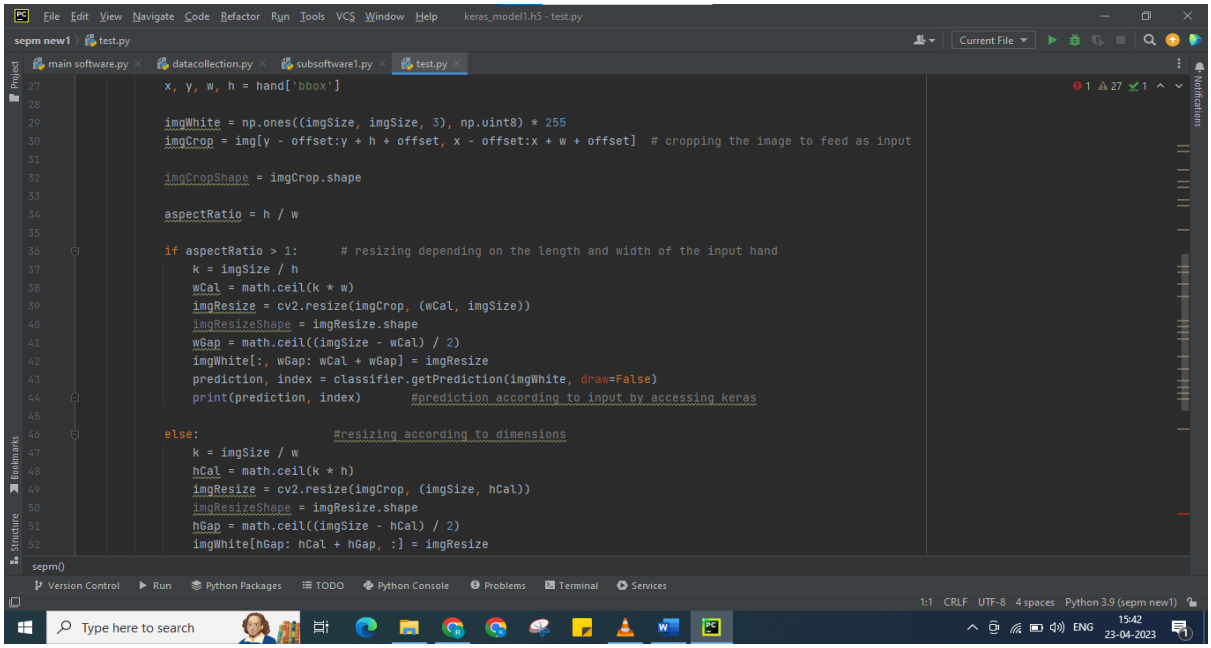


```
        aspectRatio = h / w
        if aspectRatio > 1:
            k = imgSize / h
            wCal = math.ceil(k * w)
            imgResize = cv2.resize(imgCrop, (wCal, imgSize))
            imgResizeShape = imgResize.shape
            wGap = math.ceil((imgSize - wCal) / 2)
            imgWhite[:, wGap: wCal + wGap] = imgResize

        else:
            k = imgSize / w
            hCal = math.ceil(k * h)
            imgResize = cv2.resize(imgCrop, (imgSize, hCal))
            imgResizeShape = imgResize.shape
            hGap = math.ceil((imgSize - hCal) / 2)
            imgWhite[hGap: hCal + hGap, :] = imgResize

        cv2.imshow("ImageCrop", imgCrop)
        cv2.imshow("ImageWhite", imgWhite)

    cv2.imshow("Image", img)
    key = cv2.waitKey(1)
    if key == ord("s"):
        counter += 1
        cv2.imwrite(f'{folder}/Image_{time.time()}.jpg', imgWhite)
        print(counter)
```

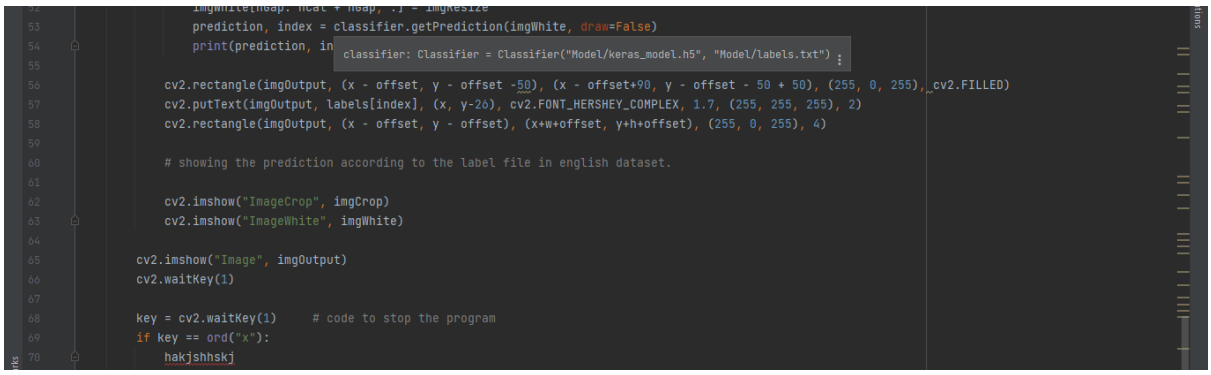
Software run code-



```
def sepm():
    import cv2
    from cvzone.HandTrackingModule import HandDetector
    from cvzone.ClassificationModule import Classifier
    import numpy as np
    import math

    cap = cv2.VideoCapture(0)   # camera access
    detector = HandDetector(maxHands=1)    #number of hands to be detected
    classifier = Classifier("Model/keras_model.h5", "Model/labels.txt")   #keras file after training of the model

    offset = 20
    imgSize = 300

    folder = "data/C"
    counter = 0

    labels = ["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"]
    #database

    while True:
        success, img = cap.read()
        imgOutput = img.copy()
        hands, img = detector.findHands(img)
        if hands:
            hand = hands[0]
```

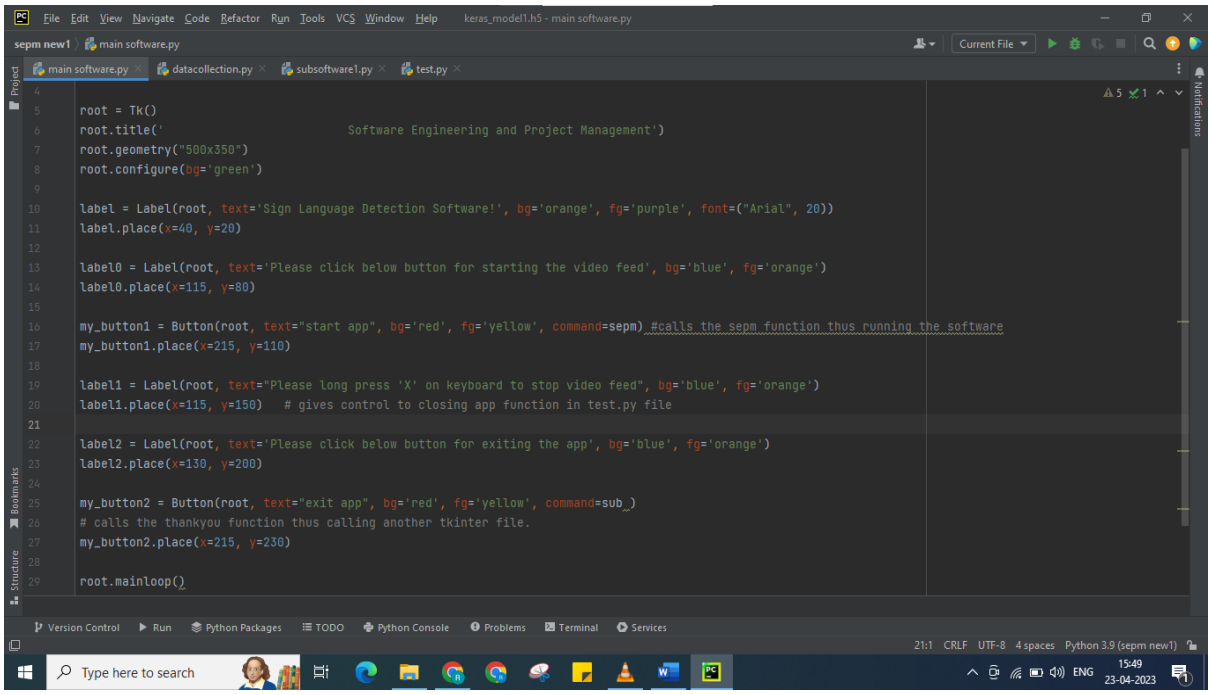


```
            x, y, w, h = hand['bbox']

            imgWhite = np.ones((imgSize, imgSize, 3), np.uint8) * 255
            imgCrop = img[y - offset:y + h + offset, x - offset:x + w + offset]  # cropping the image to feed as input

            imgCropShape = imgCrop.shape

            aspectRatio = h / w

            if aspectRatio > 1:      # resizing depending on the length and width of the input hand
                k = imgSize / h
                wCal = math.ceil(k * w)
                imgResize = cv2.resize(imgCrop, (wCal, imgSize))
                imgResizeShape = imgResize.shape
                wGap = math.ceil((imgSize - wCal) / 2)
                imgWhite[:, wGap: wCal + wGap] = imgResize
                prediction, index = classifier.getPrediction(imgWhite, draw=False)
                print(prediction, index)       #prediction according to input by accessing keras

            else:                   #resizing according to dimensions
                k = imgSize / w
                hCal = math.ceil(k * h)
                imgResize = cv2.resize(imgCrop, (imgSize, hCal))
                imgResizeShape = imgResize.shape
                hGap = math.ceil((imgSize - hCal) / 2)
                imgWhite[hGap: hCal + hGap, :] = imgResize
```



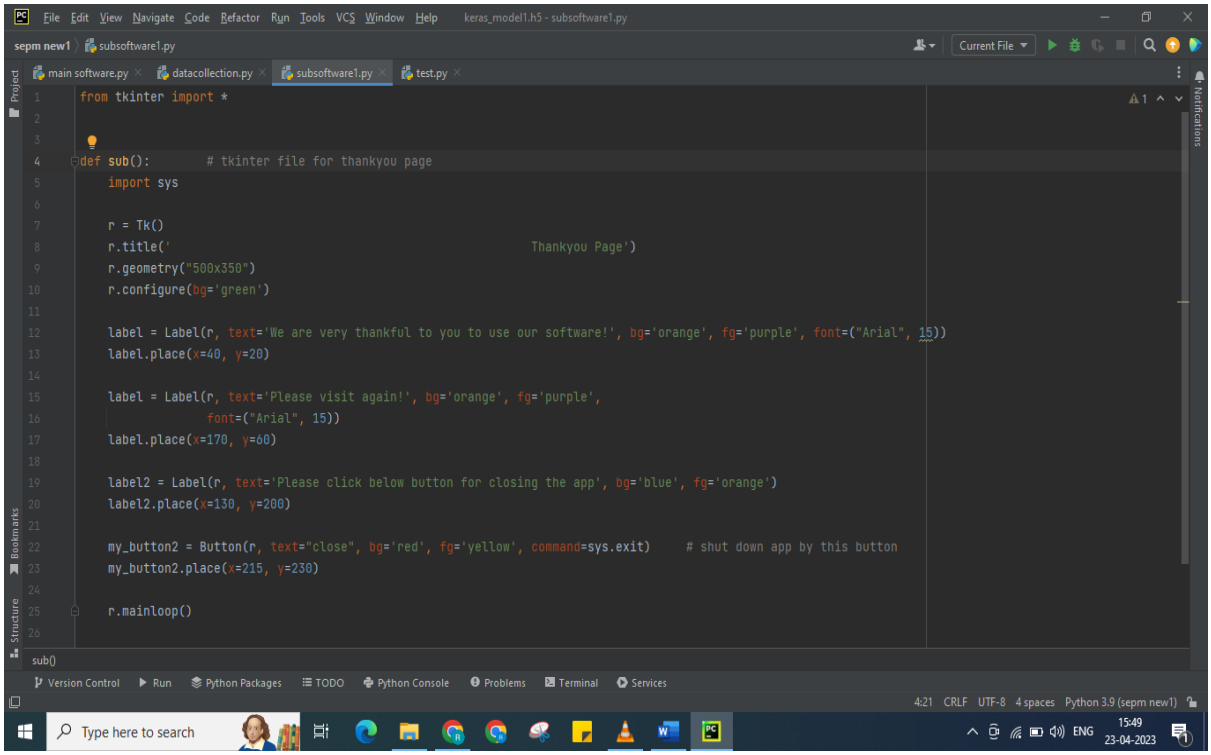
```
                prediction, index = classifier.getPrediction(imgWhite, draw=False)
                print(prediction, in
                classifier: Classifier = Classifier("Model/keras_model.h5", "Model/labels.txt")

            cv2.rectangle(imgOutput, (x - offset, y - offset -50), (x - offset+90, y - offset - 50 + 50), (255, 0, 255), cv2.FILLED)
            cv2.putText(imgOutput, labels[index], (x, y-26), cv2.FONT_HERSHEY_COMPLEX, 1.7, (255, 255, 255), 2)
            cv2.rectangle(imgOutput, (x - offset, y - offset), (x+w+offset, y+h+offset), (255, 0, 255), 4)

            # showing the prediction according to the label file in english dataset.

            cv2.imshow("ImageCrop", imgCrop)
            cv2.imshow("ImageWhite", imgWhite)

        cv2.imshow("Image", imgOutput)
        cv2.waitKey(1)

        key = cv2.waitKey(1)     # code to stop the program
        if key == ord("x"):
            hakjshhskj
```

Initial User interface to start the app code-



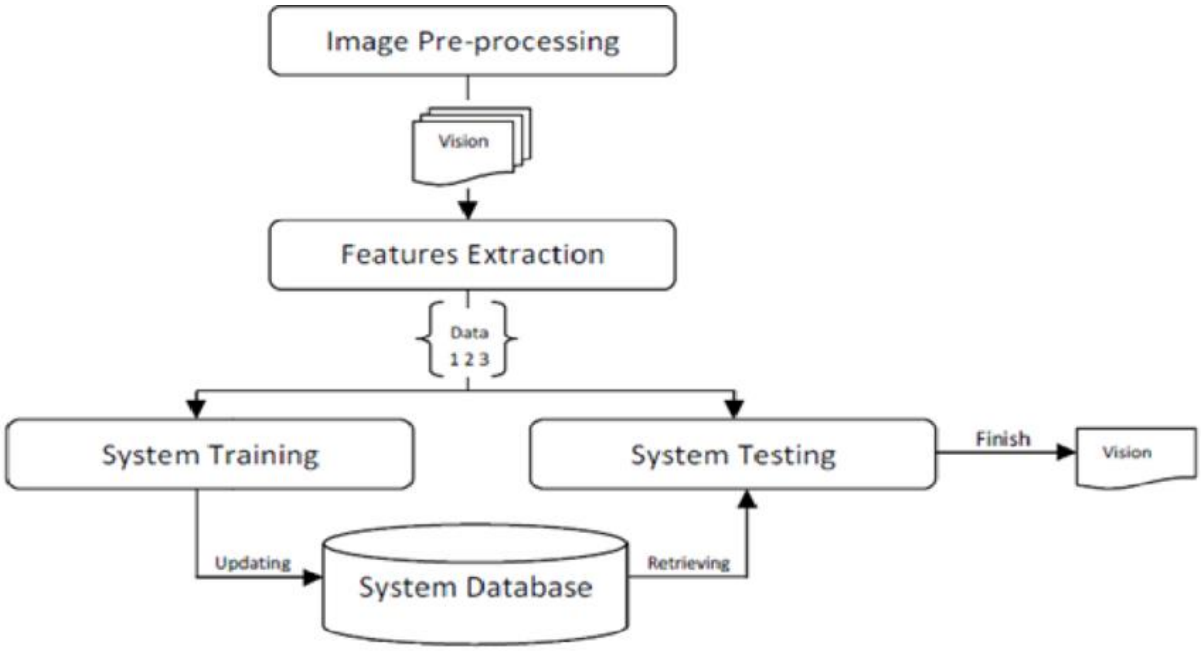
```
root = Tk()
root.title('                        Software Engineering and Project Management')
root.geometry("500x350")
root.configure(bg='green')

label = Label(root, text='Sign Language Detection Software!', bg='orange', fg='purple', font=("Arial", 20))
label.place(x=40, y=20)

label0 = Label(root, text='Please click below button for starting the video feed', bg='blue', fg='orange')
label0.place(x=115, y=80)

my_button1 = Button(root, text="start app", bg='red', fg='yellow', command=sepm) #calls the sepm function thus running the software
my_button1.place(x=215, y=110)

label1 = Label(root, text="Please long press 'X' on keyboard to stop video feed", bg='blue', fg='orange')
label1.place(x=115, y=150)   # gives control to closing app function in test.py file

label2 = Label(root, text='Please click below button for exiting the app', bg='blue', fg='orange')
label2.place(x=130, y=200)

my_button2 = Button(root, text="exit app", bg='red', fg='yellow', command=sub_)
# calls the thankyou function thus calling another tkinter file.
my_button2.place(x=215, y=230)

root.mainloop()
```

Final thankyou page while exiting code-



```
from tkinter import *


def sub():       # tkinter file for thankyou page
    import sys

    r = Tk()
    r.title('                                        Thankyou Page')
    r.geometry("500x350")
    r.configure(bg='green')

    label = Label(r, text='We are very thankful to you to use our software!', bg='orange', fg='purple', font=("Arial", 15))
    label.place(x=40, y=20)

    label = Label(r, text='Please visit again!', bg='orange', fg='purple',
                  font=("Arial", 15))
    label.place(x=170, y=60)

    label2 = Label(r, text='Please click below button for closing the app', bg='blue', fg='orange')
    label2.place(x=130, y=200)

    my_button2 = Button(r, text="close", bg='red', fg='yellow', command=sys.exit)     # shut down app by this button
    my_button2.place(x=215, y=230)

    r.mainloop()
```

SYSTEM ARCHITECTURE



1. Input devices: The system can use video cameras, depth cameras, or other sensors to capture sign language gestures performed by a user. These devices should be located in a well-lit environment to ensure that the captured images are of high quality.
2. Pre-processing: The captured image or video data may require pre-processing before it is used for sign detection. This may include image enhancement, noise reduction, and image segmentation to isolate the user's hand or other relevant features.
3. Feature extraction: The pre-processed image or video data is used to extract features that will be used for sign language recognition. This may include hand shape, hand motion, finger position, or other relevant features.
4. Sign language recognition: The extracted features are processed using a machine learning algorithm, such as a convolutional neural network (CNN), to recognize the sign language gesture. The algorithm may be trained on a large dataset of sign language samples to improve accuracy.
5. Output: Once the sign language gesture is recognized, the system can output the corresponding text or audio output, or any other appropriate output method, to the user.
6. Database: The system may use a database to store sign language gesture samples and their corresponding text or audio output. This can be used to improve the accuracy of the sign language recognition algorithm over time.

7. Overall, the system architecture for sign detection for the deaf and mute involves capturing input from the user, pre-processing the data, extracting features, using a machine learning algorithm to recognize the sign language gesture, outputting the corresponding text or audio output, providing feedback to the user, and using a database to improve accuracy over time.

Result: Thus, the details of architectural design/framework/implementation along with the screenshots were provided.

CONCLUSION

In conclusion, the sign detection software for deaf and dumb is a valuable tool that helps bridge the communication gap between hearing and speech-impaired individuals and the rest of society. The program, which is written in Python, uses computer vision techniques to recognize hand gestures and translate them into spoken or written language. The program has undergone rigorous testing and training to ensure that it can recognize a wide range of signs accurately. Its ease of use also means that even individuals with limited technical expertise can use it effectively. The potential applications of the software are vast. It can be integrated into video conferencing platforms, allowing for real-time communication between deaf and hearing individuals. It can also be used in classrooms, enabling deaf or dumb students to communicate with their teachers and peers. The software can even be used in public spaces, such as hospitals or government offices, where individuals with hearing or speech impairments may require assistance. Overall, the sign detection software for deaf and dumb is a remarkable achievement, and its potential impact cannot be overstated. With continued development and refinement, it has the potential to transform the lives of millions of people worldwide.

REFERENCES

1 Sunitha K. A, Anitha Saraswathi.P, Aarthi.M, Jayapriya. K, Lingam Sunny, "Deaf Mute Communication Interpreter- A Review", International Journal of Applied Engineering Research, Volume 11, pp 290-296, 2016.

2 Mathavan Suresh Anand, Nagarajan Mohan Kumar, Angappan Kumaresan, "An Efficient Framework for Indian SignLanguage Recognition Using Wavelet Transform" Circuits and Systems, Volume 7, pp 1874- 1883, 2016.

3 Mandeep Kaur Ahuja, Amardeep Singh, "Hand Gesture Recognition Using PCA", International Journal of Computer Science Engineering and Technology (IJCSET), Volume 5, Issue 7, pp. 267-27, July 2015.

4 Sagar P.More, Prof. Abdul Sattar, "Hand gesture recognition system for dumb people",

5 International Journal of Science and Research (IJSR)

6 Chandandeep Kaur, Nivit Gill, "An Automated System for Indian Sign Language Recognition", International Journal of Advanced Research in Computer Science and Software Engineering.

7 Pratibha Pandey, Vinay Jain, "Hand Gesture Recognition for Sign Language Recognition: A Review", International Journal of Science, Engineering and Technology Research (IJSETR), Volume 4, Issue 3, March 2015.

8 Nakul Nagpal,Dr. Arun Mitra.,Dr. Pankaj Agrawal, "Design Issue and Proposed Implementation of Communication Aid for Deaf & Dumb People", International Journal on Recent and Innovation Trends in Computing and Communication ,Volume: 3 Issue: 5,pp-147 – 149.

9 S. Shirbhate1, Mr. Vedant D. Shinde2, Ms. Sanam A. Metkari3, Ms. Pooja U. Borkar4, Ms. Mayuri A. Khandge/Sign-Language-Recognition-System.2020 IRJET Vol3 March,2020.

AWS Certificate