"Real-time Hand Gesture Recognition using TensorFlow & OpenCV"

AN EXPERENTIAL LEARNING PROJECT REPORT SUBMITTED TO

THE NATIONAL INSTITUTE OF ENGINEERING MYSURU

(An Autonomous Institute under VTU, Belagavi)

In partial fulfillment of the requirements for the award of degree of

Bachelor of Engineering in Computer Science and Engineering

Submitted by

Mohammed Arman Ali (4NI22CS123)
Madan A (4NI22CS111)
Harshith M (4NI22CS071)
Harshith M(4NI22CS072)

<u>Under the guidance of</u>

Mr. G B Janardhana Swamy
Assistant professor, CSE dept

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERINGTHE NATIONAL INSTITUTE OF ENGINEERING

Mysuru-570 008

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING THE NATIONAL INSTITUTE OF ENGINEERING
Mysore-570 008
2024-2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
THE NATIONAL INSTITUTE OF ENGINEERING

ESTD : 1946

CERTIFICATE

This is to certify that the project work entitled "**Real-time Hand Gesture Recognition using TensorFlow & OpenCV**" is a work carried out by **Mohammed Arman Ali (4NI22CS123), Madan A (4NI22CS111), Harshith M (4NI22CS071), Harshith M(4NI22CS072)**in partial fulfillment for the SPM-project work **(BCS501),** fifth semester, Computer Science & Engineering, The National Institute of Engineering (Autonomous Institution under Visvesvaraya Technological University, Belagavi) during the academic year 2024-2025. It is certified that all corrections and suggestions indicated for the Internal Assessment have been incorporated in the report deposited in the department library. The SPM project work report has been approved in partial fulfillment as per academic regulations of The National Institute of Engineering, Mysuru.

Signature of the Internal Guide	**Signature of the HoD**
______________	______________
Mr. G B Janardhana Swamy Assistant Professor Dept. of CSE, NIE	Dr. Anitha R Professor and Head Dept. of CSE, NIE

TABLE OF CONTENTS

Chapter 1

Introduction

In the modern era, human-computer interaction has undergone a rapid transformation with the advent of advanced technologies and intuitive interfaces. Traditional input methods, such as keyboards and mice, are increasingly being complemented or replaced by natural and gesture-based interfaces. However, developing real-time and accurate gesture recognition systems has long posed significant challenges due to the complexity of hand motion, variations in lighting conditions, and diverse user behaviors.With advancements in machine learning and computer vision, the implementation of gesture recognition technology has become more accessible. These systems enable seamless interaction between users and machines, providing innovative solutions in fields such as sign language translation, virtual reality, gaming, and robotics. By automating gesture recognition, these systems reduce the dependency on traditional input devices and offer a more intuitive, hands-free user experience.The "**Real-Time Hand Gesture Recognition System**" is a cutting-edge solution designed to address these challenges. This application leverages the **MediaPipe** framework **and TensorFlow's** deep learning capabilities, combined with **OpenCV** for real-time video processing, to detect and classify hand gestures efficiently. It utilizes pre-trained models to identify and interpret hand gestures in real-time, providing a platform for interactive applications.By adopting this system, developers and researchers can simplify complex hand gesture recognition tasks, enhance user interaction, and create scalable solutions across various industries. This platform not only improves operational efficiency but also provides real-time feedback, ensuring high accuracy and adaptability to dynamic user environments.The Real-Time Hand Gesture Recognition System represents a significant advancement in gesture-based interfaces, enabling intuitive, efficient, and reliable human-computer interaction. As the demand for hands-free and immersive technologies continues to grow, this system empowers users to explore innovative applications and bridge the gap between traditional methods and modern technological capabilities

Chapter 2

Stakeholders and Team member's roles

Stakeholders in the Real-Time Hand Gesture Recognition System Project

1. **Developers and Researchers**
    1. Use the system to create and test gesture-based applications across multiple domains.
    2. Rely on pre-trained models and customizable frameworks for faster and more efficient prototyping.
    3. Seek detailed documentation, sample datasets, and integration tools for streamlined development.
2. **End Users (Gamers, Virtual Reality Enthusiasts, and Accessibility Advocates)**
    1. Enjoy intuitive, hands-free interaction in gaming, virtual reality (VR), and augmented reality (AR) environments.
    2. Depend on accurate and responsive gesture detection for seamless user experiences.
    3. Require consistent performance in dynamic environments and various conditions like lighting or background noise.
3. **Industries (Healthcare, Robotics, and Automation)**
    1. Leverage gesture recognition for applications like robotic control, telemedicine, and hands-free device navigation.
    2. Use the technology to create accessible interfaces in high-stakes scenarios.
4. **Technology Providers (MediaPipe, TensorFlow, OpenCV)**
    1. Deliver robust and efficient libraries and frameworks for gesture detection and recognition.
    2. Provide regular updates to ensure compatibility with evolving technologies and industry needs.
    3. Facilitate community support and resources to encourage innovation and address technical challenges.

Team Members and Responsibilities

Mohammed Arman Ali (Team Lead)

1. Oversees the project, ensuring collaboration among team members.
2. Manages timelines, deliverables, and communication with stakeholders.

Madan A (Computer Vision Specialist)

1. Implements hand detection and landmark recognition using MediaPipe.
2. Integrates OpenCV for real-time frame processing and gesture visualization.
3. Develops preprocessing pipelines for optimal gesture recognition performance.

Harshith M (Machine Learning Engineer)

1. Trains and fine-tunes the neural network model for gesture classification.
2. Optimizes TensorFlow-based implementations for speed and accuracy.
3. Prepares and validates datasets to improve model robustness.

Harshith M (System Architect and Tester)

1. Designs the overall system architecture, ensuring seamless integration of MediaPipe, TensorFlow, and OpenCV components.
2. Implements rigorous testing strategies, including unit testing, integration testing, and performance validation.

Chapter 3:

Problem Statement, Scope, and Limitations

3.1 Problem Statement:

The advancement of technology has paved the way for innovative human-computer interaction methods, yet traditional input devices like keyboards and mice remain a barrier to seamless, intuitive interaction. The Real-Time Hand Gesture Recognition System addresses the need for an efficient, hands-free, and natural way of interaction. Existing gesture-based systems often struggle with issues such as accuracy, latency, and adaptability to diverse environments and use cases.
This project aims to develop a robust solution that leverages advanced frameworks such as MediaPipe, TensorFlow, and OpenCV to enable accurate and responsive gesture recognition. By providing real-time functionality and a user-friendly experience, this system can be utilized across various industries, including gaming, healthcare, robotics, and accessibility, bridging the gap between traditional interaction methods and modern technological advancements.

3.2 Scope of the Project :

The Real-Time Hand Gesture Recognition System focuses on the following key areas:

1. Hand Detection and Landmark Recognition:
   i. Utilizes MediaPipe to detect hands and extract 21 key landmarks, ensuring accurate gesture recognition.
2. Real-Time Gesture Recognition:
   i. Implements a TensorFlow-based model to identify gestures in real-time with high accuracy and minimal latency.
3. User-Friendly Interface:
   i. Provides an intuitive platform that ensures seamless interaction for users, including developers, researchers, and end-users.
4. Cross-Industry Applications:
   i. Supports diverse use cases such as gaming, robotic control, virtual reality interfaces, sign language translation, and healthcare.
5. Customization:
   i. Allows developers to train models for specific gestures or integrate the system into

existing applications.

6. Platform Independence:
    i. Compatible with various devices and operating systems to ensure accessibility and usability.
7. Scalability:
    i. Designed to handle multiple users or devices without significant performance degradation.

3.3 Limitations :

Despite its innovative approach and features, the Real-Time Hand Gesture Recognition System has certain limitations:

1. Environmental Constraints:
    i. Performance may be affected by external factors such as poor lighting, complex backgrounds, or occluded hand positions.
2. Gesture Limitations:
    i. The system initially recognizes a limited set of gestures, with additional gesture training requiring time and computational resources.
3. Processing Requirements:
    i. Real-time gesture recognition requires devices with adequate computational power, which may limit usability on low-end hardware.
4. Internet Dependency (Optional Features):
    i. If cloud-based processing or updates are implemented, stable internet connectivity will be required.
5. Security and Privacy Concerns:
    i. Continuous video feed processing raises potential privacy concerns, requiring careful data handling and compliance with security standards.
6. Initial Deployment Features:
    i. Advanced functionalities such as multi-hand gesture detection, environment-specific optimizations, and gesture prediction using AI may not be available in the initial version.

Chapter 4:

Problem Identification and Stakeholders Input

Role Play:

Mohammed Arman Ali (Team Lead):

Issues like latency, poor recognition in varied lighting or backgrounds, and limited gestures hinder usability. To make these systems practical, they must offer real-time, accurate recognition and adapt to diverse environments, enabling applications in gaming, robotics, healthcare, and accessibility. Engaging key stakeholders developers, researchers, end-users, and industry representatives throughout development will ensure the system meets diverse needs."

Madan A (Frontend/Backend Developer):

"I agree, Arman. To address these challenges, we should develop a platform that supports accurate hand detection and recognition in real-time. The user interface must be intuitive, catering to both developers and end-users. On the backend, we need a model capable of recognizing multiple gestures efficiently and allowing users to customize gestures as needed.

Harshith M (Database Administrator):

"From the database perspective, we'll need an optimized structure to store gesture models, user preferences, and application logs. Quick retrieval and updates will be critical, especially for real-time recognition and user customization. Additionally, maintaining data consistency and security will be essential, particularly when integrating the system with external applications or services."

Harshith M (System Architect and Tester):

"The architecture must be scalable and capable of handling high-performance requirements. We need a robust backend to ensure real-time processing without lag. During testing, I'll focus on evaluating performance under different conditions, such as varying lighting, or complex backgrounds. I'll also collaborate with the frontend team to ensure the system provides a smooth and engaging user experience, especially during application integration and gesture interaction."

Chapter 5:

Defining the Problem Statement

Problem Identification and Stakeholders' Input

Mohammed Arman Ali (Team Lead):

"One of the biggest challenges in hand gesture recognition systems is ensuring real-time performance, accuracy, and adaptability across diverse environments. Current systems often struggle with latency, inconsistent detection in varying conditions, and limited gesture libraries. Our goal is to develop a robust and versatile platform that delivers reliable real-time hand gesture recognition, enabling seamless interaction across applications in gaming, robotics, healthcare, and accessibility."

Madan A (Frontend/Backend Developer):

"From a technical perspective, the system should provide an intuitive user experience for both developers and end-users. I'll focus on designing a user-friendly interface where users can test gestures and customize recognition settings. On the backend, I'll implement a highly efficient and adaptable recognition model that works across different devices and environments. Scalability and cross-platform compatibility will be essential to ensure the system caters to a wide audience."

Harshith M (Database Administrator):

"To support this system, we'll need a well-structured database to store gesture libraries, user preferences, and system logs. I'll focus on optimizing data retrieval to ensure real-time performance and accuracy. Role-based access will be implemented to ensure that only authorized users can modify gesture definitions or access sensitive data. Additionally, we'll need to incorporate data analytics to track performance and improve user experience over time."

Harshith M (System Architect and Tester):

"The architecture must prioritize scalability and performance. I'll design a backend that supports real-time gesture recognition and adapts to varying network conditions. Testing will be a critical component of development; I'll employ extensive strategies to validate the system's performance in diverse environments, ensuring it works seamlessly across devices and lighting conditions. The focus will also be on minimizing latency to deliver instant feedback during gesture interactions."

Mohammed Arman Ali (Team Lead):

"These are excellent contributions. Throughout the development process, we'll actively involve key stakeholders such as developers, end-users, and industry experts to gather feedback and refine the system. Their insights will help us create a versatile and impactful platform that meets the needs of diverse applications and use cases."

Chapter 6:

Requirements Elicitation

Role Play:

Mohammed Arman Ali (Team Lead):

"To ensure our system meets the real needs of users, we must gather input from key stakeholders, including developers, end-users, and industries relying on gesture recognition. Madan, as a developer, what are the critical features the system should include to ensure usability and efficiency? Excellent insights. To further refine these requirements, we'll engage with stakeholders such as application developers, VR users, and industry experts through surveys and interviews."

Madan A (Frontend/Backend Developer):

"The system should provide an intuitive user interface where users can interact with and test hand gestures effortlessly. Features like a real-time gesture tracking visualization, customizable gesture libraries, and seamless integration with third-party applications will be crucial. Additionally, it should allow developers to adjust recognition sensitivity and thresholds for better control. Scalability for multiple gestures and environments is a must."

Harshith M (Database Administrator):

"To support these features, the database must be robust and secure. It should handle gesture libraries, user preferences, and system logs efficiently while enabling fast access to data for real-time processing. Role-based access will ensure only authorized users can modify gesture definitions or access sensitive information. I'll also implement analytics capabilities to monitor system performance and user interaction patterns."

Harshith M (System Architect and Tester):

"The system architecture must prioritize real-time performance and scalability. A lightweight and responsive backend will be critical to ensure low latency during gesture recognition. I'll focus on designing tests for various scenarios, such as differing lighting conditions, incomplete hand gestures, and multiple users interacting simultaneously. It's important to ensure that the system performs consistently across devices and networks."

Chapter 7:

Development Process Using Agile

Role Play:

Mohammed Arman Ali (Team Lead):

"To ensure a structured and iterative approach, we'll adopt Agile methodology for this project. Each sprint will focus on building and delivering specific functionalities of the system. Let's identify what we'll achieve in the first sprint."

Madan A :

"In Sprint 1, I'll develop the basic interface for real-time webcam input and gesture display. This includes setting up the camera feed and ensuring smooth integration with the MediaPipe framework for hand detection."

Harshith M (Database Administrator):

"For Sprint 1, I'll create a structure to manage gesture data, including class names and corresponding model outputs. This will also include setting up storage for training data and model files."

Harshith M (System Architect and Tester):

"I'll ensure the system architecture is scalable and robust from the start. During Sprint 1, I'll conduct unit testing on the MediaPipe integration and validate the accuracy of hand keypoint detection."

Mohammed Arman Ali (Team Lead):

"Great start. Sprint 2 will focus on integrating the pre-trained TensorFlow model with the system to classify gestures. This sprint will also include the development of functionality to process landmarks and predict gestures in real time."

Madan A

"In Sprint 2, I'll develop the gesture prediction and display mechanism. This will involve displaying gesture labels on the live camera feed and ensuring smooth performance."

Harshith M (Database Administrator):

"In Sprint 2, I'll work on optimizing the data processing pipeline for real-time predictions. I'll also ensure efficient management of gesture classes and configurations."

Harshith M (System Architect and Tester):

"I'll conduct integration testing during Sprint 2 to validate the interaction between the TensorFlow model and the MediaPipe framework. Additionally, I'll simulate various scenarios to ensure consistent gesture recognition accuracy."

Mohammed Arman Ali (Team Lead):

"In Sprint 3, we'll focus on enhancing the user experience by adding advanced functionalities, such as support for custom gestures and visualizing confidence levels for predictions. We'll also gather feedback from users to fine-tune the system."

Takeaways for the Project

The Agile methodology ensures a flexible and iterative approach to developing the Hand Gesture Recognition System, incorporating continuous feedback for refinement.

1. **Sprint 1:**

   Focuses on setting up the webcam feed, integrating MediaPipe for hand detection, and structuring gesture-related data.

2. **Sprint 2:**

   Introduces core functionalities like real-time gesture recognition using the TensorFlow model and displaying the results on the interface.

3. **Sprint 3:**

   Adds advanced features, including custom gesture support, improved confidence visualization, and user feedback-based refinements.

Testing follows each sprint to ensure features are functional, scalable, and aligned with user expectations. This iterative process ensures the development of a robust, efficient, and user-friendly hand gesture recognition system.

Chapter 8:

Analysis Model

Role Play:

Mohammed Arman Ali (Team Lead):

"I've drafted a Use Case Diagram to represent how various stakeholders will interact with the system. The primary use cases include initializing the system, detecting gestures, classifying gestures, and providing feedback for accuracy improvements. Each stakeholder—users, developers, and administrators—will have specific roles and functionalities tailored to their needs."

Madan A (Frontend/Backend Developer):

"We should also incorporate a user-friendly interface for gesture visualization. This interface will display live feed with detected gestures and provide options for users to train the system with new gestures or view prediction details like confidence levels."

Harshith M (Database Administrator):

"To ensure the system supports these functionalities, I'll develop a robust data model. This model will include tables for storing gesture classes, training data, model parameters, and user feedback. Relationships between tables will facilitate updates and real-time gesture classification."

Harshith M (System Architect and Tester):

"I'll create an Activity Diagram to map out backend processes. For instance, when a user performs a gesture, the system will capture the input, process it with MediaPipe, classify it using the TensorFlow model, and display the output in real time. Similarly, feedback loops for gesture corrections will ensure continuous model improvement."

Mohammed Arman Ali (Team Lead):

"Excellent. Let's include these diagrams—Use Case and Activity Diagrams—in our analysis. They will provide clarity on system interactions and processes, reducing the risk of errors during development and ensuring the system meets all user expectations effectively."

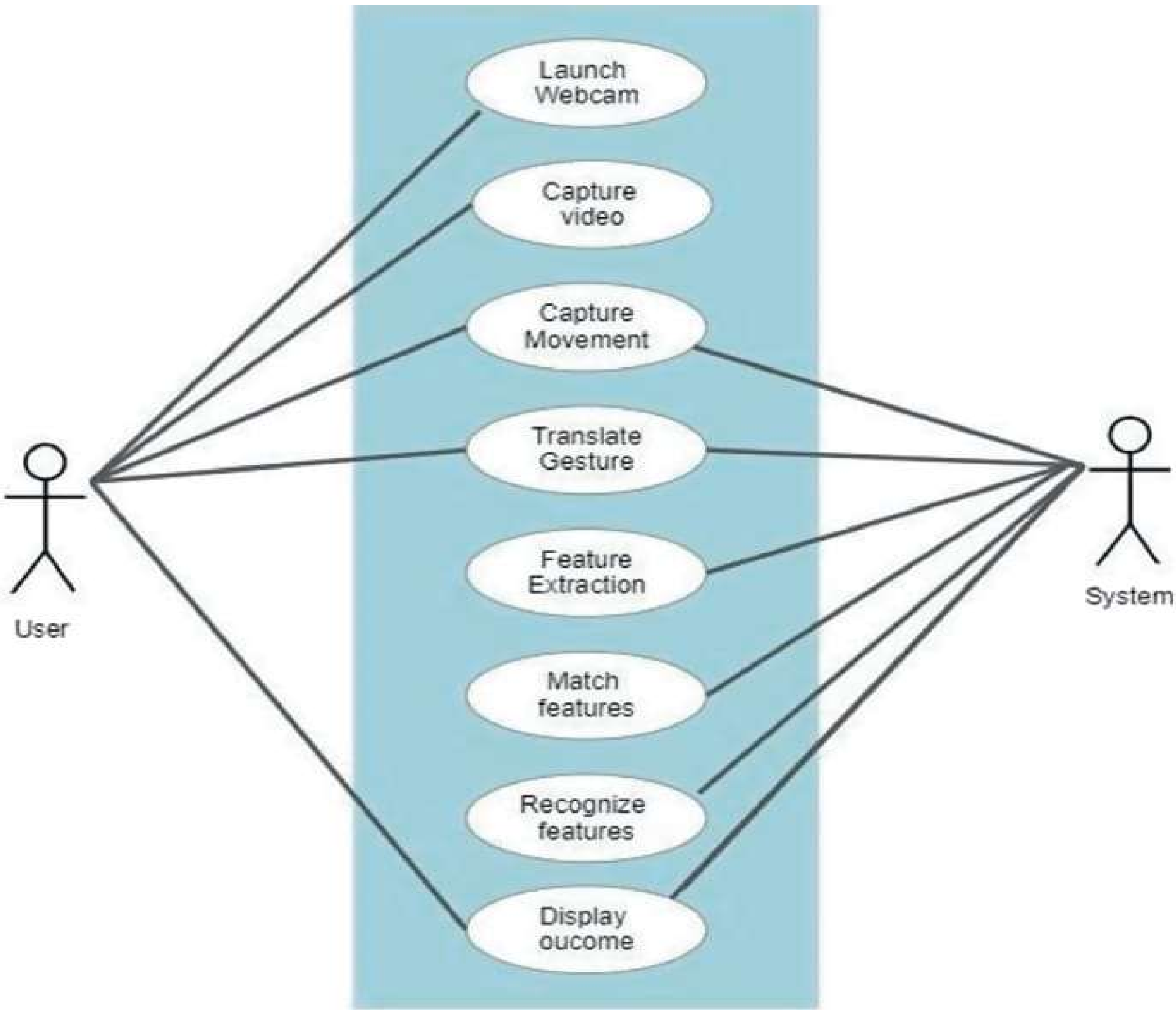
Use case diagram:

The Use Case Diagram for the Hand Gesture Recognition System outlines primary interactions between the user and the system. It captures the following key functionalities:

i. Initializing the system (launching the interface and setting up the webcam).
ii. Detecting and visualizing gestures in real time.
iii. Classifying gestures based on trained data.
iv. Training the system with new gestures.
v. Providing feedback for improving gesture classification accuracy.

Actors:

i. User: Initiates and interacts with the system for gesture recognition.
ii. Admin/Developer: Manages training data and oversees the system’s performance.
iii. System: Processes input, performs gesture recognition, and displays results.



Chapter 9:

Architectural Design

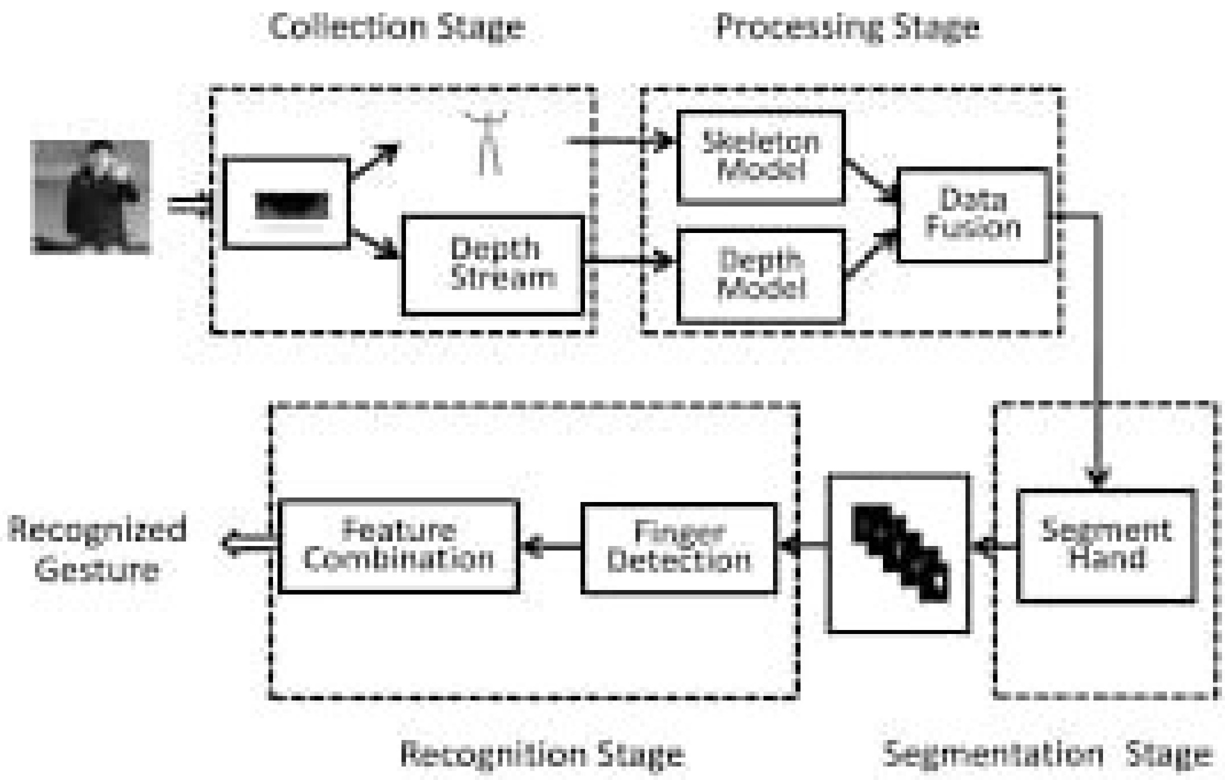
Overview:

The architecture of the Hand Gesture Recognition System ensures smooth integration between the user interface, gesture processing logic, and the backend database. This system utilizes modern technologies to capture, classify, and manage gestures, offering a responsive and scalable solution.

The system is divided into three main layers:

1. **Frontend (User Interface)**
2. **Backend (Business Logic)**

These layers work in synergy to provide an efficient and user-friendly platform for gesture recognition and management.



Key Components:

1. Frontend Layer (User Interaction):

i. **User Interface:**
   1. Displays live video feed, detected gestures, and classification results

ii. **Functionalities:**
   1. Users can interact with the system via live webcam feed.
   2. Visualize gesture classification results in real time.
   3. Access options to train the system with new gestures and provide feedback.

iii. **Mobile Optimization:**
   1. Ensures compatibility with mobile and tablet devices for convenient access.

---

2. Backend Layer (Business Logic):

i. **Gesture Recognition Framework:**
   1. Leverages MediaPipe for hand detection and TensorFlow for gesture classification.
   2. Implements preprocessing pipelines to extract hand landmarks from video feeds.

ii. **Core Features:**
   1. Real-time gesture detection and classification.
   2. Custom gesture training and feedback loops for improving accuracy.
   3. Role-based access for users, admins, and developers to manage and monitor functionalities

Data Flow Process:

i. **Gesture Detection:**
   1. The user performs a gesture captured via a live webcam feed through the interface.
   2. The backend processes the input using MediaPipe to extract hand landmarks.
   3. The extracted features are classified using TensorFlow, and results are displayed to the user.

ii. **Custom Gesture Training:**
   1. Users provide labeled data for new gestures through the interface.
   2. The backend validates and stores this data in the database.
   3. The TensorFlow model is retrained periodically with updated data.

iii. **Feedback Integration:**

1. Users provide feedback on misclassified gestures.
2. This feedback is logged in the database and used to improve model accuracy during retraining.

Architectural Design Diagrams:

Use Case Diagram:

i. Illustrates the interaction between actors (users, developers, and admins) and the system's functionalities, such as gesture detection, custom training, and feedback.

Activity Diagram:

ii. Describes key processes, such as gesture detection and classification, showcasing the workflow from input capture to result display and feedback management.

iii. This architecture ensures that the Hand Gesture Recognition System is robust, scalable, and aligned with user needs. Let me know if you'd like assistance with creating the diagrams!

Chapter 10:

Test Case Design

Overview:

The test case design for the Hand Gesture Recognition System focuses on validating functionality, accuracy, and usability. Each test case ensures that the system meets the specified requirements, providing a seamless experience for users interacting with gesture recognition, training, and feedback features.

Test Case Examples

Test Case ID: TC001

Objective: Verify that the gesture recognition function correctly identifies predefined gestures.

Input: User performs a predefined gesture (e.g., "Thumbs Up").

Steps:

1. Launch the Hand Gesture Recognition System.
2. Open the live camera feed interface.
3. Perform the "Thumbs Up" gesture in front of the camera.
4. Observe the recognition result displayed on the screen.

   Expected Result:

   The system recognizes the "Thumbs Up" gesture and displays the correct classification label.

   Actual Result: TBD

   Status: TBD

Test Case ID: TC002

Objective: Verify that the custom gesture training feature allows users to add and classify new gestures.

Input: User provides a new gesture labelled "Wave" with sample images.

Steps:

1. Launch the Hand Gesture Recognition System.
2. Navigate to the "Custom Gesture Training" feature.
3. Perform the new gesture ("Wave") multiple times to capture training data.
4. Label the gesture as "Wave" and save the training data.

5. Test the newly added gesture by performing it in front of the camera.
   Expected Result:
   The system recognizes the "Wave" gesture and displays the correct classification label.
   Actual Result: TBD
   Status: TBD

Testing Approach

1. Unit Testing:

i. Test individual modules such as hand landmark detection, preprocessing, gesture classification, and feedback logging to ensure they function independently.

2. Integration Testing:

1. Verify interactions between components:
   i. The frontend captures video feeds correctly.
   ii. Backend processes gestures accurately.
   iii. Database stores and retrieves custom gesture data effectively.

3. System Testing:

1. Evaluate the entire system to ensure it meets both functional and non-functional requirements, including:
   i. Real-time gesture detection performance.
   ii. Scalability for handling multiple custom gestures.
   iii. System accuracy across various lighting conditions and hand orientations.

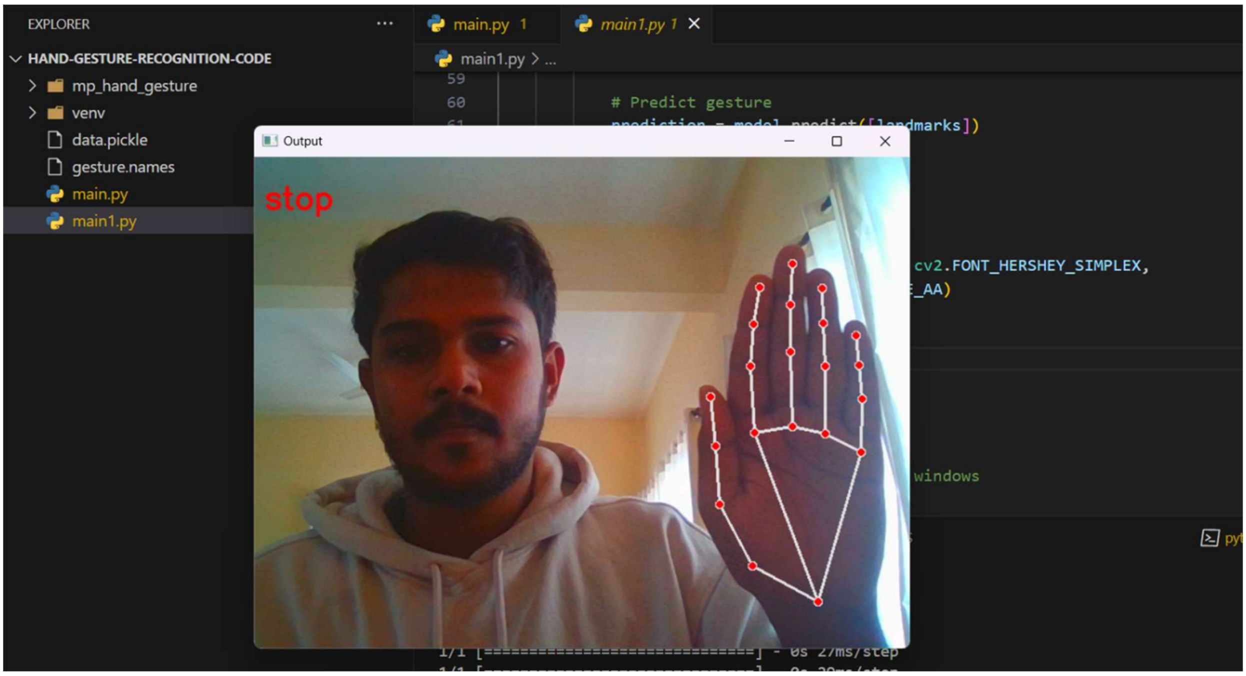
4. User Acceptance Testing (UAT):

1. Collect feedback from users such as educators, developers, and accessibility experts to validate usability.
2. Conduct real-world usage scenarios:
   i. Testing in environments with varying lighting conditions.
   ii. Evaluating system behavior with users of different hand sizes and gestures.

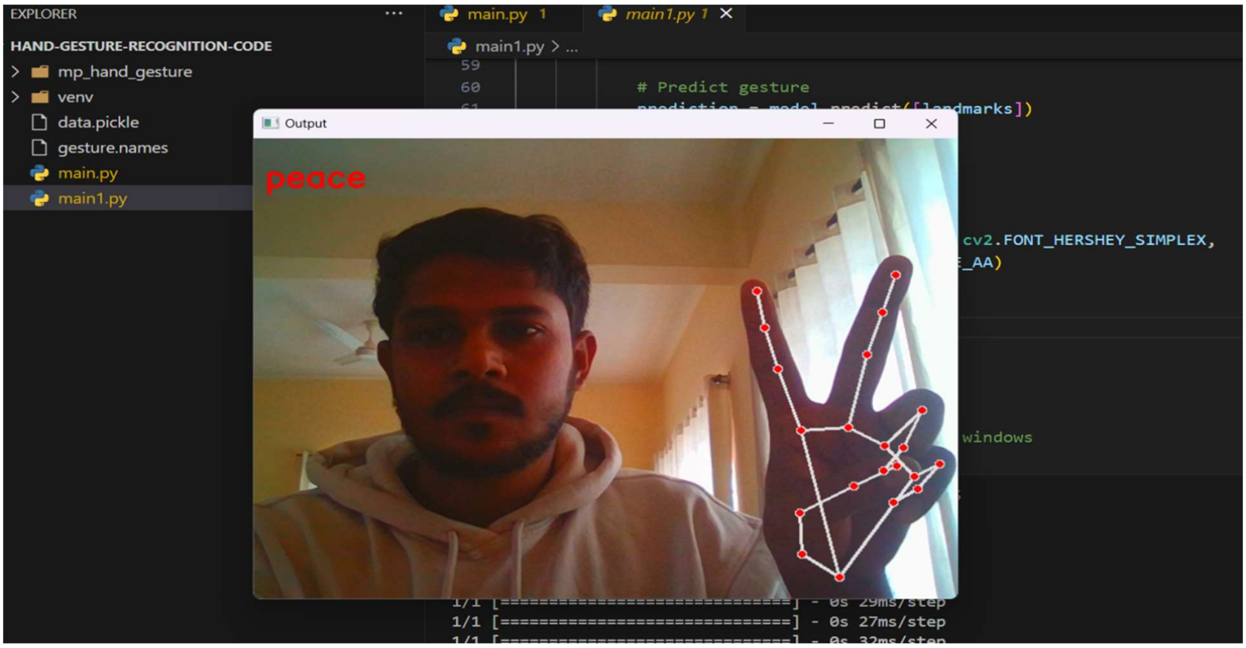
This testing strategy ensures that the Hand Gesture Recognition System is functional, reliable, and user-centric. Let me know if you need further details or additional test cases!

Real Time Testing (Output) Alpha Testing :

Testcase 1 :



Testcase2 :



Chapter 11:

Project Summary

The **Hand Gesture Recognition System** is an innovative platform designed to facilitate intuitive interaction between users and digital systems using hand gestures. This system integrates core features such as gesture detection, real-time recognition, customizable gesture training, and feedback mechanisms to provide a seamless experience for diverse applications such as accessibility, gaming, and human-computer interaction.

Developed using an Agile methodology, the project was completed iteratively, with each sprint focusing on critical components such as gesture recognition algorithms, training modules for custom gestures, real-time processing, and user feedback. Testing was performed at each phase to ensure that the system met functional requirements, provided accurate recognition, and delivered a smooth and responsive user experience.

The architecture of the system is built to support scalability, accuracy, and performance, ensuring that it can handle a growing library of gestures and work effectively in various environments. The backend processes gestures efficiently, leveraging machine learning models for classification, while the frontend offers an intuitive interface that is responsive across devices, ensuring ease of use for all users, including those with limited technical expertise.

This Hand Gesture Recognition System empowers users by providing a natural and efficient way to interact with digital systems. It enables developers, educators, and accessibility advocates to leverage gesture-based controls in their applications, simplifying user experiences and expanding the possibilities of human-computer interaction. The successful implementation of this system demonstrates the transformative role of technology in enhancing accessibility, productivity, and user engagement across various domains.

Chapter 12:

REFERENCES

W3Schools. Python Tutorial. Retrieved from https://www.w3schools.com/python/

OpenCV. Open Source Computer Vision Library. Retrieved from https://opencv.org/

TensorFlow Documentation. Retrieved from https://www.tensorflow.org/

Mediapipe. A Cross-Platform Framework for Building Perception Pipelines. Retrieved from https://mediapipe.dev/

Retrieved from IEEE Xplore: Research Articles on Hand Gesture Recognition Systems https://ieeexplore.ieee.org/

Towards Data Science. Hand Gesture Recognition with Machine Learning. Retrieved from https://towardsdatascience.com/

GeeksforGeeks. OpenCV Python Tutorial. Retrieved from https://www.geeksforgeeks.org/