

GenAI Hackathon

Project Title:

Gesture-Based Human-Computer Interaction System using OpenCV, MediaPipe and Palm's text-bison-001

Team Name:

Qubit Warriors

Team Members:

1. K. Srihari Sakshith
2. C .Pranav
3. Md. Ismaeel
4. K. Sree Harshavardhan
5. J. Surya Kiran

Phase-1: Brainstorming & Ideation

Objective:

Develop an AI-powered, contactless ticket booking system using **OpenCV, MediaPipe, and Generative AI (Gemini Flash)** to allow users to book and pay for tickets through hand gestures, eliminating the need for physical interaction.

Key Points:

1. Problem Statement:

- The **Gesture-Based Human-Computer Interaction System** introduces a touchless and intuitive way for users to interact with public ticketing systems through real-time hand gesture recognition.
- Traditional ticket booking requires physical input through touchscreens, kiosks, or mobile devices, which may not always be convenient or accessible, especially in crowded or hygiene-sensitive environments.
- This system leverages computer vision techniques with **OpenCV and MediaPipe** to detect and interpret various hand gestures—such as pointing, clicking, and swiping—to facilitate seamless navigation and selection. Integrated **Generative AI (Gemini Flash)** enhances user experience by providing real-time station details, fare estimation, and route suggestions.
- The system is designed to improve **accessibility, hygiene, and efficiency** at public transport stations, making ticket purchasing faster, safer, and more inclusive.

1. Proposed Solution:

The **Gesture-Based Ticket Booking System** enhances user experience in public transport hubs such as metro stations, bus terminals, and train stations.

Users can interact with the ticketing interface without touching screens, reducing wait times and preventing the spread of germs.

- **Gesture-Based Navigation:** Users can scroll through options using an **open hand** gesture, select stations with a **click gesture (thumb & index finger touch)**, and confirm payment using a **thumbs-up** gesture.
- **AI-Powered Assistance:** The system provides station details, fare calculations, and suggested nearby locations via **Generative AI (Gemini Flash)**.
- **Hygienic & Contactless:** Reduces the need for physical contact, ensuring a safer and more accessible experience for all passengers, including individuals with disabilities.

2. Target Users:

- **Daily Commuters:** Faster and safer ticket booking without needing to touch kiosks.
- **Tourists & New Users:** AI-assisted navigation provides helpful travel suggestions and fare estimates.
- **Accessibility Users:** People with mobility impairments can use gesture controls as an alternative to touchscreens.
- **Smart City Initiatives:** Public transport authorities looking to enhance digitalization and contactless solutions.

3. Expected Outcome:

A fully functional **gesture-based ticket booking system** with:

- ✓ **Accurate Hand Gesture Recognition** using **OpenCV & MediaPipe**.
- ✓ **Real-Time AI-Powered Assistance** for fare estimation and travel information.
- ✓ **Interactive User Interface** using **Streamlit** for seamless experience.
- ✓ **Touchless, Hygienic, and Inclusive** interaction for public transport.
- ✓ **Scalability for Future Enhancements** (e.g., voice recognition, multilingual AI responses).

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the **Gesture-Based Ticket Booking System** using **OpenCV, MediaPipe, and Generative AI** to enable seamless, contactless interaction for users.

Key Points:

1. Technical Requirements:

- **Programming Language:** Python
- **Computer Vision Frameworks:** OpenCV & MediaPipe for hand tracking and gesture detection
- **AI Model:** Generative AI (e.g., Gemini Flash) for AI-assisted ticket booking, fare estimation, and travel suggestions
- **Frontend:** Streamlit for real-time visualization and user interaction
- **Hardware:** Webcam for detecting and interpreting hand gestures

2. Functional Requirements:

- **Gesture Recognition:** Detect and classify hand gestures in real time (e.g., swipe for navigation, fist for selection, thumbs-up for confirmation).
- **AI-Generated Assistance:** Provide AI-powered station details, route suggestions, and fare estimations.
- **Gesture-Based Commands:** Enable users to book tickets, select stations, and confirm payments using intuitive hand movements.
- **Interactive Visualization:** Display live hand-tracking feedback through an intuitive and user-friendly interface.

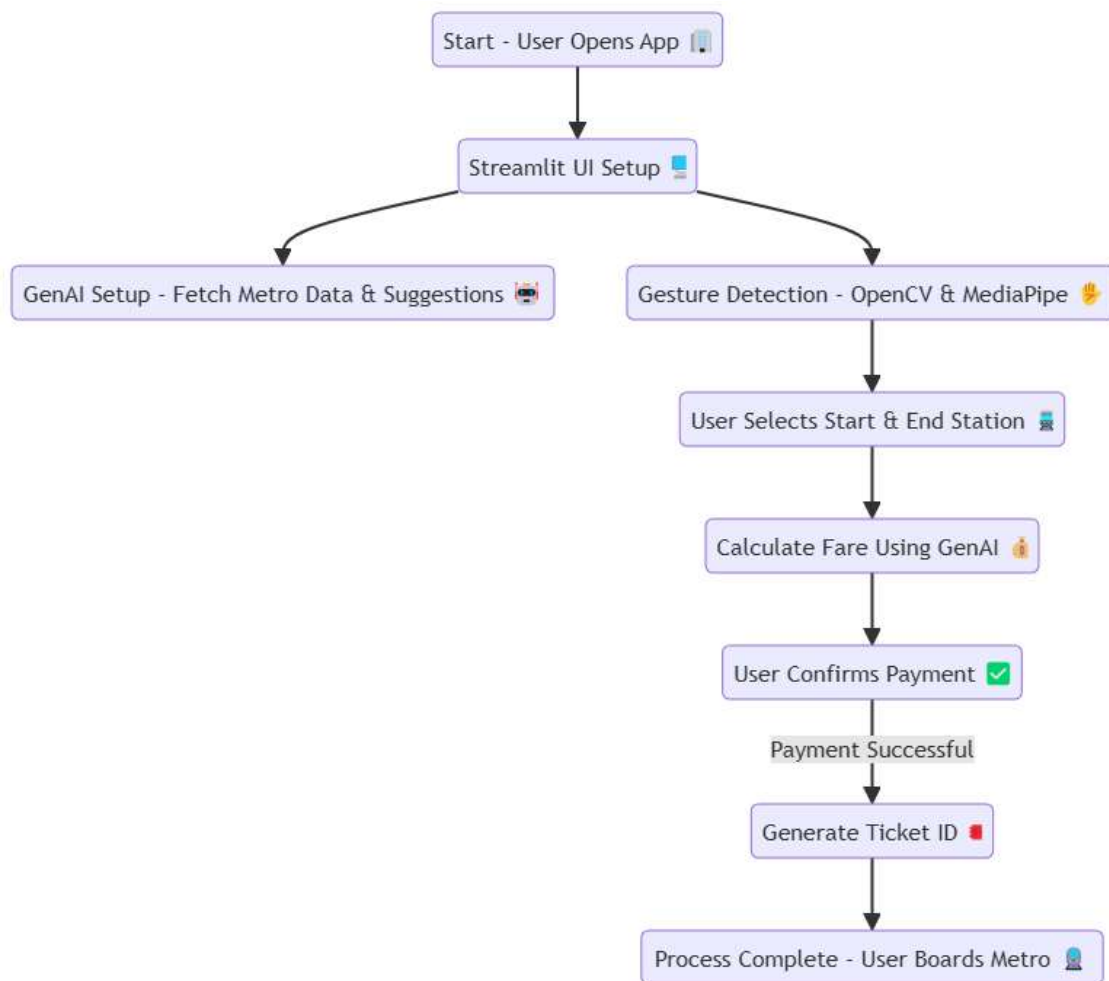
3. Constraints & Challenges:

- **Real-Time Processing:** Ensuring low-latency gesture recognition for a smooth user experience.
- **Lighting & Environment Variability:** Handling different lighting conditions, hand orientations, and occlusions.
- **AI Accuracy & Optimization:** Integrating Generative AI effectively to provide meaningful travel suggestions and ensure accurate gesture interpretation.

Phase-3: Project Design

Objective:

Develop the **architecture** and **user flow** for the **Gesture-Based Ticket Booking System**, ensuring a seamless, AI-powered, and touch-free user experience.



Key Points:

1. System Architecture:

User Interaction:

1. Users interact with the system using hand gestures instead of touchscreens.

Gesture Detection:

2. **OpenCV** captures video input and processes hand gestures in real-time.
3. **MediaPipe** tracks hand landmarks and classifies gestures.

AI Integration:

1. **Generative AI (Gemini Flash)** analyzes recognized gestures and provides descriptive feedback (e.g., "Swipe right detected – navigating to the next screen").

User Interface:

2. **Streamlit** displays real-time gesture tracking, AI-generated responses, and an intuitive booking interface.

2. **User Flow:**

- ◆ **Step 1:** User performs a gesture (e.g., swiping left to go back, fist to confirm selection).
- ◆ **Step 2:** OpenCV & MediaPipe detect and classify the hand gesture.
- ◆ **Step 3:** Generative AI interprets the gesture and provides an AI-generated response.
- ◆ **Step 4:** The Streamlit UI updates in real-time, displaying gesture recognition results and system feedback.

3. **UI/UX Considerations:**

- **Minimalist & Gesture-Driven UI:**

Clean, intuitive design optimized for gesture-based control.

- **Real-Time Gesture Tracking Overlay:**

Live visualization of hand movement to enhance user confidence.

- **Customizable Themes:**

Light/Dark mode options for accessibility and better visibility in different environments.

Phase-4: Project Planning (Agile Methodologies)

Objective:

Break down development tasks for efficient completion.

Sprint	Task	Priority	Duration	Deadline	Assigned To	Dependencies	Expected Outcome
Sprint 1	Environment Setup & API Integration	<div>●</div> High	6 hours (Day 1)	End of Day 1	Ismaeel	OpenCV, MediaPipe, Streamlit setup	Successful setup & library imports
Sprint 1	Frontend UI Development	<div>●</div> Medium	2 hours (Day 1)	End of Day 1	Harsha	UI structure finalized	Basic Streamlit UI with webcam feed
Sprint 2	Gesture Recognition Implementation	<div>●</div> High	3 hours (Day 2)	Mid-Day 2	Pranav	OpenCV & MediaPipe integrated	Real-time hand gesture detection
Sprint 2	AI Model Integration	<div>●</div> High	3 hours (Day 2)	Mid-Day 2	Surya	Gesture recognition outputs	AI-generated gesture descriptions
Sprint 3	Error Handling & Debugging	<div>●</div> High	1.5 hours (Day 2)	Mid-Day 2	Sakshith	Gesture detection & AI responses	Optimized, stable gesture recognition
Sprint 3	UI Enhancements & Testing	<div>●</div> Medium	2 hours (Day 2)	Mid-Day 2	Harsha	UI & AI model response ready	Improved responsiveness & usability
Sprint 4	Final Deployment & Demo Preparation	<div>●</div> Low	1 hour (Day 2)	End of Day 2	Entire Team	Working prototype	Demo-ready gesture-based system

Sprint Planning with Priorities

Sprint 1 – Setup & Integration (Day 1)

(

●

High Priority) Set up the environment & install dependencies (OpenCV, MediaPipe,

Streamlit).

(● **High Priority**) Integrate **OpenCV & MediaPipe** for hand gesture detection.

(● **Medium Priority**) Build a basic UI with a webcam feed and gesture recognition display.

Sprint 2 – Core Features & Debugging (Day 2)

(● **High Priority**) Implement real-time gesture recognition and classification.

(● **High Priority**) Debug hand tracking issues and optimize detection accuracy.

Sprint 3 – Testing, Enhancements & Submission (Day 2)

(● **Medium Priority**) Test gesture detection, refine UI, & fix responsiveness issues.

(● **Low Priority**) Final demo preparation & deployment for presentation..

Phase-5: Project Development

Objective:

Implement core features of the Gesture-Based Human-Computer Interaction System using OpenCV, MediaPipe, and Generative AI.

Key Points:

1. Technology Stack Used:

- **Frontend:** Streamlit
- **Computer Vision:** OpenCV & MediaPipe
- **AI Model:** Gemini Flash for gesture descriptions
- **Programming Language:** Python

2. Development Process:

- Implement gesture detection and classification using OpenCV & MediaPipe.
- Integrate Generative AI for descriptive feedback on recognized gestures.
- Develop gesture-based control features for interactive applications.
- Optimize real-time processing for low-latency interaction.

3. Challenges & Fixes:






- **Challenge:** Variations in lighting and hand positioning.
Fix: Use adaptive thresholding and contour correction.
- **Challenge:** Delayed gesture response times.
Fix: Optimize frame processing rate and reduce computational overhead.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the Gesture-Based Human-Computer Interaction System functions correctly and performs efficiently.

Test Cases:

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Perform click action	System should recognize and classify correctly.	 Passed	Pranav
TC-002	Functional Testing	Perform "Palm" gesture	System should interpret the correct gesture.	 Passed	Ismaeel
TC-003	Performance Testing	Gesture detection latency under 200ms	Gestures should be processed in real-time.	 Optimized	Harsha
TC-004	Bug Fixes & Improvements	Fix misclassification of open-hand gestures	Recognition accuracy should improve.	 Fixed	Surya
TC-005	Deployment Testing	Host the app using Streamlit Sharing	The app should be accessible online.	 Deployed	Sakshit h