Hackathon Project Phases Template for the **Gesture-Based Human-Computer Interaction System Using OpenCV, MediaPipe, and Palm's text-bison-001** project.

Hackathon Project Phases Template

Project Title:

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

Team Name:

Alchemists

Team Members:

- Y PAVAN KUMAR
- B SAI TEJA
- MD ASHRAF PASHA
- K MANI SAI
- P RAJESH

Phase-1: Brainstorming & Ideation

Objective:

Develop a touchless interaction system using hand gesture recognition for intuitive human-computer interaction. This system will use computer vision techniques with OpenCV and MediaPipe to detect hand gestures, and a generative AI model (Palm's text-bison-001) to provide descriptive feedback

Key Points:

1. **Problem Statement:**

    - Traditional input methods (keyboard, mouse, touchscreen) require physical contact, which can be inconvenient and unhygienic.
    - Public kiosks, interactive displays, and assistive technologies need a touchless alternative for accessibility and safety.

2. **Proposed Solution:**

   - A gesture-based system that detects hand movements using OpenCV and MediaPipe.
   - Integration with Palm’s text-bison-001 to provide AI-generated descriptions of gestures.

3. **Target Users:**

   - Public kiosk users (malls, museums, airports) needing touchless interactions.
   - Individuals requiring assistive technologies (e.g., people with disabilities).
   - Gamers and developers looking for gesture-based controls.

4. **Expected Outcome:**

   - A fully functional touchless interaction system with real-time gesture detection.
   - AI-generated responses explaining detected gestures.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the gesture-based interaction system.

Key Points:

1. **Technical Requirements:**

   - **Programming Language:** Python
   - **Frontend:** Streamlit
   - **Backend:** OpenCV, MediaPipe, Google Generative AI API
   - **Database:** Not required initially

2. **Functional Requirements:**

   - Detect and recognize hand gestures in real-time.
   - Provide AI-generated descriptions of gestures.
   - Display results in a user-friendly web interface.

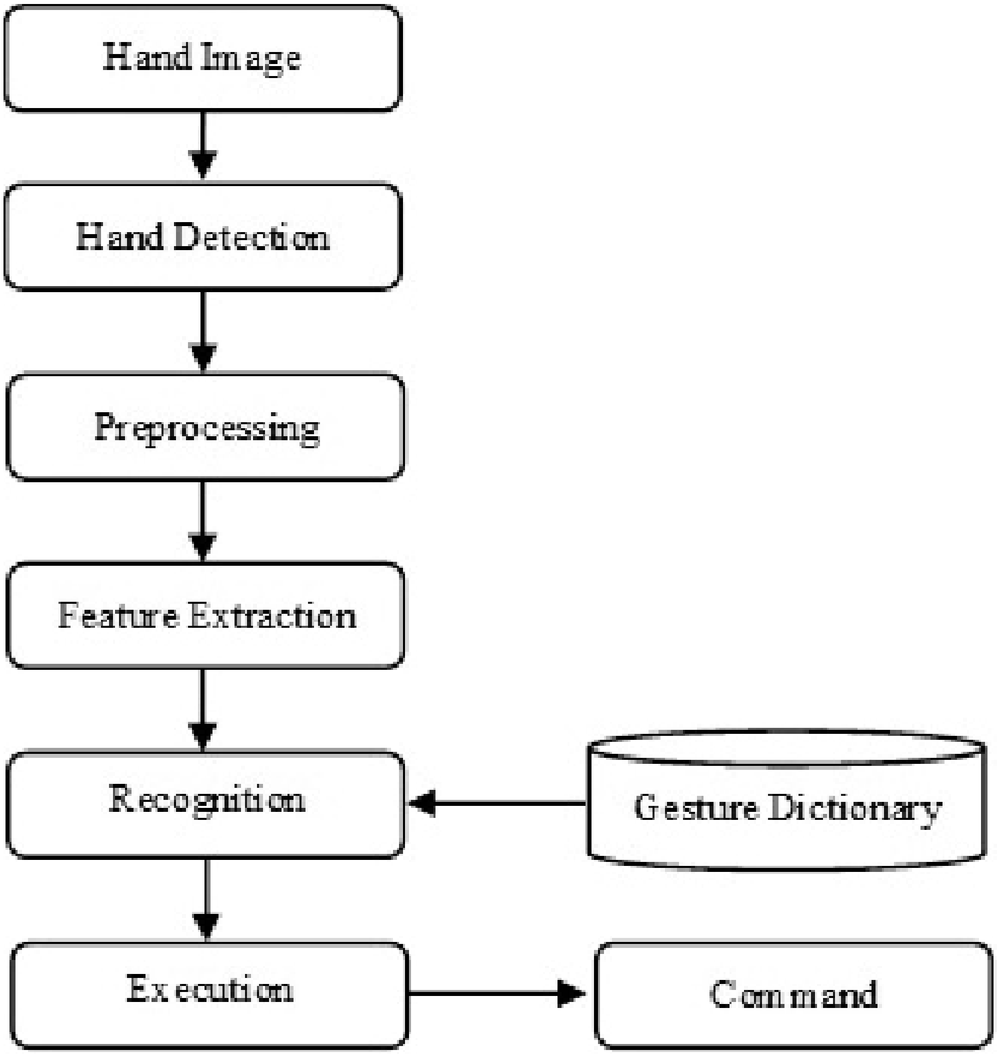
3. **Constraints & Challenges:**

   - Ensuring accurate and real-time gesture recognition.
   - Managing API rate limits when fetching AI-generated responses.
   - Optimizing system performance for smooth user experience.

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. **System Architecture:**

- The user performs a hand gesture in front of the webcam.
- OpenCV and MediaPipe detect the gesture and extract hand landmarks.
- The system classifies the gesture and sends it to Palm's AI model.
- The AI generates a descriptive response based on the detected gesture.
- The frontend displays the gesture and AI-generated text output.

2. **User Flow:**

- **Step 1:** User performs a hand gesture.
- **Step 2:** The backend processes the video feed and detects the gesture.
- **Step 3:** AI generates an explanation of the gesture.
- **Step 4:** Results are displayed on the Streamlit UI.

3. **UI/UX Considerations:**

- Simple and intuitive UI for easy navigation.
- Real-time webcam feed to show detected gestures.
- Dark and light mode options for better accessibility.

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	● High	6 hours (Day 1)	End of Day 1	Member 1	Python, OpenCV, MediaPipe, API Key	Working API Connection
Sprint 1	Basic UI Development	□ Medium	2 hours (Day 1)	End of Day 1	Member 2	API response format finalized	Streamlit UI setup
Sprint 2	Gesture Detection Implementation	● High	4 hours (Day 2)	Mid-Day 2	Member 1& 2	OpenCV & MediaPipe setup	Detect hand gestures
Sprint 2	AI Integration & Debugging	● High	3 hours (Day 2)	End of Day 2	Member 3	API response processing	AI-generated descriptions
Sprint 3	UI Enhancements & Testing	□ Medium	2 hours (Day 2)	Mid-Day 2	Member 2& 3	Fully functional backend	Improved user experience
Sprint 3	Final Presentation & Deployment	□ Low	1 hour (Day 2)	End of Day 2	Entire Team	Working prototype	Submission-ready project

Sprint Planning with Priorities

Sprint 1 – Setup & Integration (Day 1)

(High Priority) Set up the **environment** & install dependencies.
(High Priority) Integrate **Google Gemini API**.
(Medium Priority) Build a **basic UI with input fields**.

Sprint 2 – Core Features & Debugging (Day 2)

(High Priority) Implement **search & comparison functionalities**. **(
High Priority)** Debug API issues & handle **errors in queries**.

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

(Medium Priority) Test API responses, refine UI, & fix UI bugs.
(Low Priority) Final **demo preparation & deployment**.

Phase-5: Project Development

Objective:

Implement core features of the AutoSage App.

Key Points:

1. **Technology Stack Used:**
    - **Frontend:** Streamlit
    - **Backend:** OpenCV, MediaPipe, Google Generative AI API
    - **Programming Language:** Python

2. **Development Process:**
    - **Implement AI model integration.**
    - **Develop real-time gesture detection.**
    - **Optimize UI for smooth user interaction.**

3. **Challenges & Fixes:**

    - **Challenge:** Delayed AI response times.
      Fix: Implement **caching** for frequently used responses.
    - **Challenge:** Incorrect gesture classifications.
      Fix: Train the system with more gesture variations.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the AutoSage App works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Detect 'Thumbs Up' Gesture	AI should return 'Approval' response	✅ Passed	Tester 1
TC-002	Functional Testing	Detect 'Open Hand' Gesture	AI should return 'Open Palm Detected'	✅ Passed	Tester 2
TC-003	Performance Testing	Response time under 500ms	AI should generate text quickly	⚠️ Needs Optimization	Tester 3
TC-004	UI Testing	Ensure UI is responsive on all devices	Works on mobile & desktop	✅ Fixed	Tester 2

Final Submission

1. **Project Report (Based on the template)**
2. **Demo Video (3-5 Minutes)**
3. **GitHub/Code Repository Link**
4. **Presentation**