**Selfie Project: A Browser-Based Real-Time Vision Control App**

**Project Summary**

The **Selfie Project** is a browser-based interactive application that uses a computer's webcam to track body movements, facial expressions, and hand gestures in real time. The system leverages computer vision techniques such as pose estimation and gesture recognition to enable users to interact with the system using their body.

Key applications include:

* Taking a selfie using a specific gesture
* Performing actions on the computer through hand signals
* Detecting posture and providing feedback
* Activity tracking (e.g., counting exercises)
* Triggering events like sounds or UI changes using facial or body cues

This application runs entirely in the browser using JavaScript and TensorFlow.js or MediaPipe, ensuring privacy and accessibility across platforms.

**Core Features**

| **Feature** | **Description** |
| --- | --- |
| Gesture Detection | Detect hand signals like open palm, thumbs-up, peace sign, and pointing |
| Pose Estimation | Identify body joints and postures for actions and fitness tracking |
| Gesture-Triggered Selfies | Capture a selfie automatically when a specific gesture is recognized |
| Face and Expression Detection | Use facial cues like smiling or yawning to trigger events |
| Signal-Based Triggers | Map custom gestures to specific browser-based actions |
| Posture Monitoring | Alert users when poor posture is detected |
| Activity Tracker | Detect and count physical activities like squats or jumping jacks |
| Voice Feedback | Provide spoken responses or alerts using speech synthesis APIs |

**Technology Stack**

**Frontend:**

* HTML, CSS, JavaScript

**Computer Vision / ML Libraries:**

* TensorFlow.js
* MediaPipe (Pose, Hands, FaceMesh)
* Optional: ml5.js (simplified wrapper for beginners)

**Other APIs and Tools:**

* getUserMedia (WebRTC) for webcam access
* HTML5 Canvas for rendering annotations
* Web Speech API or Howler.js for audio feedback
* Optional UI framework: React.js or plain JavaScript for interactivity

**Project Structure**

For a vanilla JavaScript implementation:

selfie-project/

│

├── index.html

├── style/

│ └── styles.css

├── scripts/

│ ├── main.js # Application entry point

│ ├── poseHandler.js # Pose and posture detection logic

│ ├── gestureHandler.js # Hand signal recognition and interpretation

│ ├── selfieModule.js # Selfie capture, preview, and storage

│ ├── ui.js # Canvas overlays and visual feedback

│ └── actions.js # Signal-to-action mapping logic

├── assets/

│ └── sounds/ # Sound clips for alerts or confirmations

│ └── icons/ # Optional UI icons

├── models/ # Local or remote machine learning models

├── data/

│ └── selfies/ # Folder for storing selfies (or use IndexedDB)

└── README.md

**Example Gestures and Mapped Actions**

| **Gesture** | **Triggered Action** |
| --- | --- |
| Peace sign | Take a selfie |
| Raised hand | Start a timer or open an app |
| Thumbs-up | Confirm an action or play a sound |
| Pointing left/right | Navigate through interface or images |
| Folded arms | Pause detection temporarily |
| Head tilt left/right | Navigate left/right through tabs |
| Squat or jump | Count fitness activity repetitions |

**Requirements**

**For Users:**

* Modern browser with support for getUserMedia (Chrome, Firefox, Edge, Safari)
* Functional webcam with decent lighting
* Stable internet (if models are loaded from CDN)

**For Developers:**

* Proficiency with JavaScript (ES6+)
* Understanding of asynchronous programming and event handling
* Experience with Canvas API, TensorFlow.js, or MediaPipe is recommended

**Libraries and Tools:**

* TensorFlow.js or MediaPipe (Pose, Hands, FaceMesh)
* HTML5 Canvas API
* WebRTC API for camera access
* Optional: React or Vue for UI, Vite or Webpack for bundling

**Modes of Operation**

| **Mode** | **Description** |
| --- | --- |
| Solo Mode | Single-person tracking for selfies, gestures, or posture |
| Group Mode | Optional multi-person detection and interaction |
| Focus Mode | Monitors seated posture and alerts on slouching |
| Fitness Mode | Detects and counts physical exercises based on body motion |
| Custom Actions Mode | User-defined gestures mapped to custom events or commands |

**Potential Extensions**

* Integrate smart home triggers (e.g., lights on with a gesture)
* Add voice confirmation for commands (e.g., "Did you mean take a selfie?")
* Export selfie images or posture logs to cloud storage (Google Drive, Dropbox)
* Introduce facial emotion recognition for sentiment tracking
* Build support for augmented reality overlays (filters, hats, etc.)
* Allow gesture customization and training

**Testing and Deployment**

**Testing:**

* Ensure webcam permissions and model loading work across browsers
* Test gesture responsiveness under different lighting and backgrounds
* Validate on desktop and mobile browsers

**Deployment Options:**

* GitHub Pages

**Debugging Tools:**

* Chrome DevTools for console and network monitoring
* FPS monitoring for performance optimization

**Development Plan**

| **Phase** | **Objective** |
| --- | --- |
| Phase 1 | Set up webcam streaming and display with canvas overlays |
| Phase 2 | Integrate pose and hand detection using TensorFlow.js or MediaPipe |
| Phase 3 | Implement gesture recognition and define action triggers |
| Phase 4 | Develop the selfie module and UI feedback system |
| Phase 5 | Add posture tracking, voice alerts, and save-to-disk options |
| Phase 6 | Final testing and deployment |