**SourceCode:**

**Overview:**

**1.** **Dataset:** You will need a collection of images for each individual: Jayapriya, Dhanachezhiyan, Barath, Siva, and Bharathi.

**2. Face Recognition:** This can be done using \*\*TensorFlow.js\* and the \*BlazeFace\* model.

**3. Firebase Integration:** Firebase will be used to store the attendance logs for each user when they "punch in."

**Project Structure :**

/face-recognition-punching-system

│

├── index.html

├── styles.css

├── app.js

├── firebase-config.js

├── images/

│ ├── jayapriya.jpg

│ ├── dhanachezhiyan.jpg

│ ├── barath.jpg

│ ├── siva.jpg

│ └── bharathi.jpg

**Index.html** - The Web Interface

html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Face Recognition Punching System</title>

<link rel="stylesheet" href="styles.css">

<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs"></script>

<script src="https://cdn.jsdelivr.net/npm/@tensorflow-models/blazeface"></script>

<script src="https://www.gstatic.com/firebasejs/9.0.2/firebase-app.js"></script>

<script src="https://www.gstatic.com/firebasejs/9.0.2/firebase-firestore.js"></script>

</head>

<body>

<div class="container">

<h1>Face Recognition Punching System</h1>

<video id="video" width="640" height="480" autoplay></video>

<div id="status">Waiting for face detection...</div>

<button id="punch-button" disabled>Punch In</button>

<div id="attendance-log"></div>

</div>

<script src="firebase-config.js"></script>

<script src="app.js"></script>

</body>

</html>

**Styles.css** - Simple Styling for the Interface

css

body {

font-family: Arial, sans-serif;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

margin: 0;

background-color: #f4f4f9;

}

.container {

text-align: center;

}

video {

border: 2px solid #4CAF50;

margin-bottom: 20px;

}

button {

padding: 10px 20px;

background-color: #4CAF50;

color: white;

border: none;

cursor: pointer;

font-size: 16px;

}

button:hover {

background-color: #45a049;

}

#status {

font-size: 18px;

margin-top: 20px;

}

#attendance-log {

margin-top: 20px;

font-size: 14px;

color: #333;

}

**Firebase-config.js** - Firebase Configuration

This file initializes Firebase and connects to Firestore for logging attendance.

javascript

// Your web app's Firebase configuration

const firebaseConfig = {

apiKey: "YOUR\_API\_KEY",

authDomain: "YOUR\_PROJECT\_ID.firebaseapp.com",

projectId: "YOUR\_PROJECT\_ID",

storageBucket: "YOUR\_PROJECT\_ID.appspot.com",

messagingSenderId: "YOUR\_SENDER\_ID",

appId: "YOUR\_APP\_ID"

};

// Initialize Firebase

const app = firebase.initializeApp(firebaseConfig);

const db = firebase.firestore();

**App.js** - JavaScript Logic for Face Recognition and Punch In

javascript

let model;

let videoElement = document.getElementById('video');

let punchButton = document.getElementById('punch-button');

let statusElement = document.getElementById('status');

let attendanceLog = document.getElementById('attendance-log');

// Preloaded faces for matching

const knownFaces = {

"Jayapriya": "images/jayapriya.jpg",

"Dhanachezhiyan": "images/dhanachezhiyan.jpg",

"Barath": "images/barath.jpg",

"Siva": "images/siva.jpg",

"Bharathi": "images/bharathi.jpg"

};

// Load the BlazeFace model

async function loadModel() {

model = await blazeface.load();

statusElement.innerText = 'Model Loaded. Waiting for Face...';

startVideo();

}

// Start the webcam video stream

function startVideo() {

navigator.mediaDevices.getUserMedia({

video: true

}).then((stream) => {

videoElement.srcObject = stream;

}).catch((err) => {

statusElement.innerText = 'Error accessing webcam: ' + err;

});

}

// Detect faces in real-time using BlazeFace

async function detectFace() {

const predictions = await model.estimateFaces(videoElement, false);

if (predictions.length > 0) {

statusElement.innerText = 'Face Detected!';

punchButton.disabled = false;

// Find the closest match

checkFaceMatch(predictions[0].topLeft, predictions[0].bottomRight);

} else {

statusElement.innerText = 'No Face Detected';

punchButton.disabled = true;

}

requestAnimationFrame(detectFace);

}

// Match face with the known dataset

async function checkFaceMatch(topLeft, bottomRight) {

// Use TensorFlow.js to compare face embeddings

// This is a simplified logic, use real embeddings comparison in a production system

// Placeholder for face recognition with known images

for (let person in knownFaces) {

const img = new Image();

img.src = knownFaces[person];

img.onload = function() {

const faceImage = tf.browser.fromPixels(img);

// Compare logic would go here (embedding-based)

if (isMatch(topLeft, bottomRight, faceImage)) {

punchButton.disabled = false;

return;

}

};

}

}

// Placeholder for face comparison logic

function isMatch(topLeft, bottomRight, faceImage) {

// Actual face matching algorithm should go here (e.g., compare embeddings)

return true; // Simplified for this example

}

// Firebase Punch In Function

async function punchIn() {

const userId = 'user\_' + new Date().getTime(); // Unique ID for the punch-in session

const timestamp = new Date().toLocaleString();

// Log attendance in Firebase Firestore

db.collection("attendance").doc(userId).set({

punchInTime: timestamp,

userId: userId

}).then(() => {

attendanceLog.innerHTML = `Punch In Successful: ${timestamp}`;

}).catch((error) => {

console.error("Error logging attendance: ", error);

});

}

// Add event listener to punch button

punchButton.addEventListener('click', punchIn);

// Start the model and video stream

loadModel();

detectFace();

**How to Set Up Your Dataset:**

You need images for each person in your dataset (Jayapriya, Dhanachezhiyan, Barath, Siva, Bharathi). These images should be in the /images/ directory and named as follows:

- jayapriya.jpg

- dhanachezhiyan.jpg

- barath.jpg

- siva.jpg

- bharathi.jpg

These images will be used for comparison when a face is detected by the webcam.

**Key Points:**

1. **Face Detection with BlazeFace**: This is used to detect faces in real-time from the webcam feed.

2. **Image Dataset**: Each user's image is stored in the knownFaces object with their name as the key and image file path as the value.

3. **Punch In**: When a face is detected and matched, the "Punch In" button becomes active. Upon clicking it, Firebase logs the punch-in timestamp.

4. **Firebase Integration**: Attendance is logged in \*\*Firebase Firestore\* with a timestamp when the user punches in.

5. **Face Recognition**: In a real-world application, face recognition would be done using embeddings and cosine similarity. The current example is simplified.

**How to Run the System:**

1. **Set Up Firebase**:

* Create a Firebase project in the Firebase Console.
* Enable Firestore and get your Firebase config, replacing the placeholder values in firebase-config.js.

2. **Host the Application**:

You can host this app using \*Firebase Hosting\* or any static web hosting service (e.g., GitHub Pages, Netlify).

3. **Install TensorFlow.js and BlazeFace**:

The required libraries for face detection (BlazeFace) and TensorFlow.js are linked via CDN in the index.html.

4. **Run the App**:

Open the index.html file in your browser. If everything is set up correctly, the webcam will display, and upon detecting a face, the \*"Punch In"\* button will