1. Purpose of This Document

This document serves as an **instructional and communication manual** for all team members involved in the Vision-Based Detection Project. It defines what the project is about, how it is structured, how components interact, and how work should be executed, tracked, and validated.

The documentation consolidates elements of a **Project Requirement**, **Process Requirements Document (PRD)**, **Functional Requirements Document (FRD)**, **System/Software Design Document (SDD/SSD)**, and **Process Maps** into a single, coherent and future reference.

This is written to **guide the team clearly and simply** through the Vision-Based Detection Project. Our main goal is to:

- Understand what we are building
- Know what tools to install
- Know who is responsible for what
- Work together efficiently (even with vibe coding)

Think of this document as **our shared playbook**. If you read this, you should understand the project without asking too many questions.

2. What This Project Is About: Face Recognition–Based Attendance System (Web-Based)

This project is about building a **web-based attendance system** that uses **face recognition** instead of manual sign-ins or paper attendance.

In simple terms:

- A camera captures students' faces
- The system checks if the face matches a registered student
- If there is a match, the student is **automatically marked present**
- Attendance records are stored and viewed through a **web application**

What the system actually does:

1. Uses a camera (webcam or laptop camera) to capture faces
2. Detects faces in real time
3. Recognizes registered students from a stored dataset
4. Automatically marks attendance
5. Displays and manages attendance records on a web interface

3. System Architecture

This section explains **how the system is structured** and **how each part works together**. Think of the system as a **pipeline**: the camera captures data, the system processes it step by step, and the results are shown on the web app.

The system is made up of **six main components** that work in sequence:

Camera → Face Detection → Face Recognition → Backend → Database → Web Interface

Each component has a clear role, and no component works in isolation.

3.2 System Components Explained

1. Image Acquisition

- Uses a **webcam or laptop camera**
- Captures real-time video frames
- Sends each frame to the face detection module

2. Face Detection Module

- Scans each video frame
- Detects where faces are located
- Draws bounding boxes around detected faces
- Uses models like **OpenCV, Haar Cascade** or **YOLO**

3. Face Recognition Module

- Takes detected faces as input
- Extracts unique facial features
- Compares them with stored facial data

4. Backend Server

- Acts as the **brain of the system**
- Controls the attendance logic
- Decides when a student is marked present
- Communicates between recognition module and database

5. Database

- Stores:
    - Student details (ID, name, face data)
    - Attendance records (date, time, status)
- Ensures attendance data is persistent and retrievable

6. Web Interface

- Displays:
    - Attendance records
    - Student list
    - System status (camera on/off, recognition running)
- Allows instructors or admins to view attendance easily

3.3 Architecture Summary (Plain Language)

1. Camera captures a face
2. System detects the face
3. System recognizes who it is
4. Backend marks attendance
5. Database saves the record
6. Web app shows the result

If you understand this flow, you understand the entire system.

4. Technology Stack

This section explains the **tools and technologies** we will use, why we chose them, and **what team members need to install** before development begins.

4.1 Computer Vision & Face Recognition

These tools handle face detection and recognition.

OpenCV

- Used for image and video processing
- Handles webcam input and frame manipulation

Face Recognition Model

We will use **one of the following** (depending on implementation simplicity):

- **LBPH Face Recognizer**
    - Simple and lightweight
    - Works well for small datasets

OR

- **FaceNet (via `face_recognition` library)**
    - More accurate
    - Uses deep learning–based face embeddings

Haar Cascade

- Used for basic face detection
- Lightweight and easy to implement

4.2 Backend (Server Logic)

The backend controls:

- Face recognition logic
- Attendance rules
- Database operations
- Communication with the frontend

Python

- Main programming language
- Easy to understand and implement
- Strong support for computer vision and web frameworks

Web Framework (We'd Choose One)

- **Flask** → simpler, ideal for beginners
- **FastAPI** → more structured, modern, but optional
- **FireBase.**

4.3 Frontend (User Interface)

The frontend is what users interact with in the browser.

HTML

- Structure of web pages

CSS

- Styling and layout

JavaScript

- Handles interactions and dynamic updates

Bootstrap (Optional..but we'd check it out)

- Pre-built UI components
- Makes pages look clean quickly
- Not mandatory

4.4 Database

The database stores student data and attendance records.

We'd make use of SQLite OR MySQL

4.5 Required Installations (Everyone Must Do This)

Before starting development, **every team member must install** the following:

System Requirements

- **Python 3.8 – 3.11**
- **Webcam** (built-in or external)
- **Code Editor** (VS Code…for now)
- **GitHub(**For Collaboration**)**
- **LINKS: Tutorial, Python, Vscode, Github**

Python Libraries

Install using pip:

```
pip install opencv-python
pip install face-recognition
pip install flask
pip install numpy
pip install sqlite3
```

Important Note to the Team

If you have **installation issues**, report them early.
Do **not wait until coding starts**.

5. Communication Rules

Clear and simple rules to keep the team efficient:

- **If you're stuck → say it**
  Don't struggle in silence — ask for help early.
- **If something breaks → share it**
  Let the team know immediately so we can fix it together.
- **If you fix a bug → explain it**
  Share what went wrong and how you solved it — others will learn from it.

6. What Success Looks Like

By the end of this project:

- The system runs on at least one machine
- Faces are detected correctly
- Known faces are recognized
- The team understands how it works

7. Final Note to the Team

This project is meant to be **fun, educational, and collaborative**. Don’t overthink it. Follow the process, let’s help each other, and learn by doing.

If you understand this document, you already understand the project.