**Document Requirements for Face Recognition Attendance Application**

**1. Project Overview**

* **Objective**: Automate attendance management using face recognition to enhance accuracy and efficiency.
* **Scope**: The system will recognize faces, mark attendance, generate reports, and provide administrative controls.

**2. Functional Requirements**

* **User Authentication**:
  + Admins: Manage users, attendance logs, and settings.
  + Employees/Students: Use face recognition for attendance marking.
* **Face Enrollment**:
  + Capture face images and store them securely.
* **Face Recognition**:
  + Real-time face detection and recognition.
* **Attendance Management**:
  + Automatically mark attendance with a timestamp.
  + Provide manual override for corrections.
* **Reporting**:
  + Generate daily, weekly, and monthly attendance reports.
  + Export reports in CSV or PDF formats.
* **Notifications**:
  + Notify users of successful attendance marking via email or app.
* **Data Storage**:
  + Store attendance logs and face data securely in a database.
* **Device Integration**:
  + Support for webcams, IP cameras, or mobile camera modules.

**3. Non-Functional Requirements**

* **Performance**:
  + Recognize faces in under 1 second.
* **Scalability**:
  + Handle up to X number of users (configurable based on the deployment size).
* **Security**:
  + Encrypt face data and attendance logs.
* **Availability**:
  + Ensure 99.9% uptime for critical operations.
* **Usability**:
  + Intuitive user interface for admins and users.

**4. Technical Requirements**

* **Hardware**:
  + Camera: High-resolution camera (HD or above).
  + System: Windows/Linux/MacOS-based server or workstation.
* **Software**:
  + Programming Language: Python, Java, or a language suited to the project.
  + Libraries/Frameworks:
    - OpenCV (for face recognition).
    - Dlib (for facial landmark detection).
    - TensorFlow/PyTorch (if deep learning is used).
  + Database: MySQL, PostgreSQL, or MongoDB.
  + Backend: Django, Flask, or Node.js.
  + Frontend: HTML, CSS, React.js, or Angular.js.
* **Network**:
  + Stable internet for remote data access and notifications.