Attendance Logging System Report

1. Database Design

Tables:

employee Table:-Stores employee data.

Columns:

id (INT) | name (VARCHAR) | designation (VARCHAR)

- Sample Data:

0 | Arslan | Data Researcher
1 | Praful | Senior Data Analyst

attendance Table:

- Records daily check-ins

- *Columns:*

id | employee_id (INT) | date (DATE) | time_in (TIME)

- *Sample Entry:*

1 | 0 | 2024-05-20 | 09:15:00

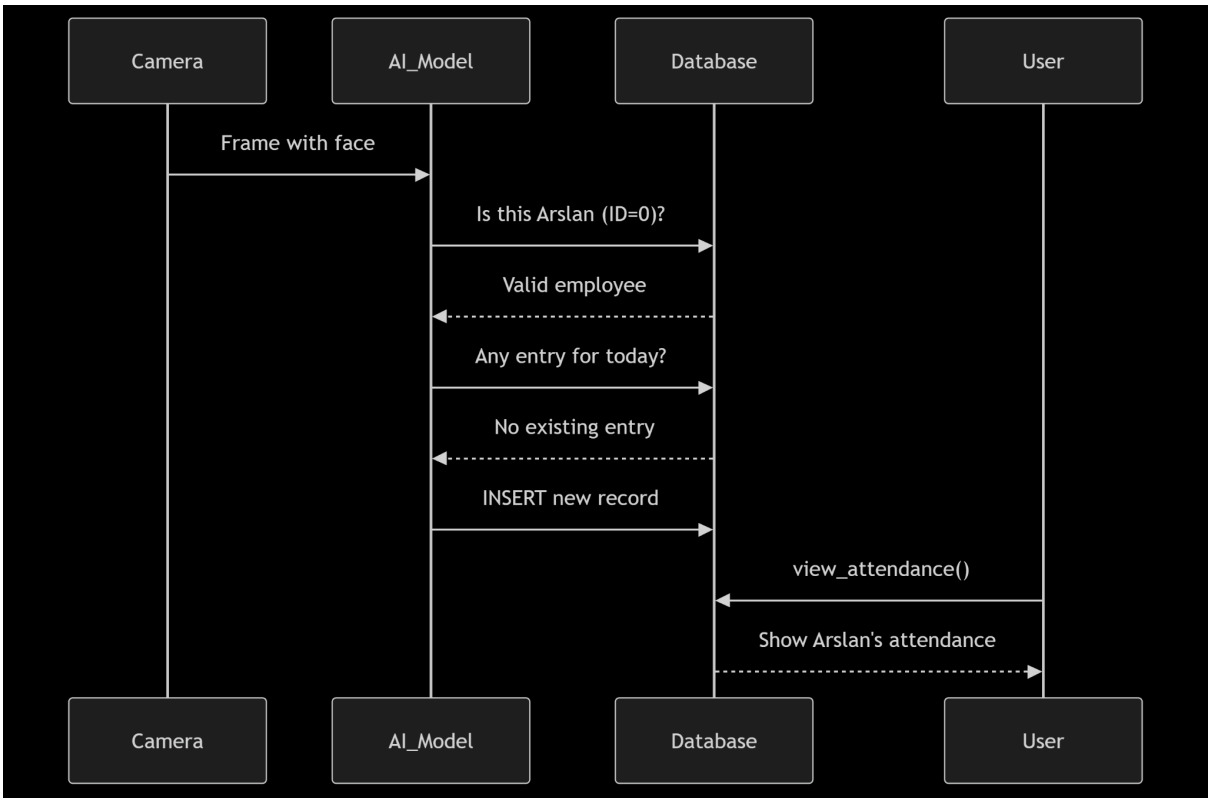
- *Relationship:*

attendance.employee_id → employee.id (Foreign Key Constraint)

2. Key Features

Feature	Implementation	Purpose
Duplicate Prevention	SELECT check before INSERT	1 entry/day/employee
Cooldown Period	5-second time buffer	Prevent rapid re-scans
Employee Validation	Verify ID exists in employee	Data integrity
Error Handling	Try-Except blocks	Prevent crashes

3. Attendance Workflow



4. Technical Implementation

Mark Attendance function:-

```
def mark_attendance(employee_id):
    # Check existing entry
    cursor.execute("""
        SELECT * FROM attendance
        WHERE employee_id = %s AND date = %s
    """, (employee_id, current_date))

    # Insert if new
    cursor.execute("""
        INSERT INTO attendance
        (employee_id, date, time_in)
        VALUES (%s, %s, %s)
    """, (employee_id, current_date, current_time))
```

Recognition → Logging

```
if max_prob > 0.7:  # Recognition threshold
    if mark_attendance(employee_id):
        last_marked[employee_id] = time.time()  # Cooldown
```

5. Performance Metrics

Metric	Value	Description
Processing Speed	~15 FPS	Real-time video analysis
Recognition Accuracy	70%+ Confidence	Configurable threshold
Data Integrity	Foreign Key Constraints	Valid employee checks
Entry Speed	<1s	From detection to DB insert