**Assignment 1 & 2**

**Software Engineering (W4)**

**Submitted By**

**M. Hammad Rana**

**(F2023065280)**

**Submitted to**

**Kinza Sardar**



**Department of Software Engineering**

**University of Management and Technology Lahore**

**Proposal: Police Verification System**

* **Introduction**
* With the increasing demand for efficient and transparent police verification processes, this system aims to streamline verification for various purposes, such as employment, rental agreements, and background checks. Traditional methods are often time-consuming and prone to inefficiencies. This project proposes a digital solution to improve the speed, accuracy, and accessibility of police verification services.
* **Project Overview**
* The Police Verification System will provide an integrated platform for citizens and police departments. It will include features such as online application submission, status tracking, automated document verification, and secure communication. By leveraging modern technologies like AI, blockchain, and cloud computing, the system will ensure accuracy, transparency, and data security.
* **Problem Statement**

Current police verification processes face challenges such as:

* Lengthy application procedures.
* Lack of real-time status updates.
* Manual errors in data processing.
* Inadequate security measures for sensitive data
* **Objectives**

The primary objectives of the Police Verification System are:

1. To enable citizens to apply for police verification online.
2. To provide real-time status updates and notifications.
3. To automate document verification and background checks.
4. To ensure data security and compliance with regulations.
5. To enhance communication between applicants and police authorities

* **Domain Analysis**
* **Customer**: Individuals, organizations, and government entities requiring police verification services.
* **Stakeholders**:
  + **Primary Stakeholders**: Citizens, police officers, and verification authorities.
  + **Secondary Stakeholders**: IT developers, system administrators, and regulatory bodies.

#### Requirements Analysis

* **Functional Requirements**:
  + User registration and login.
  + Submission of verification applications.
  + Real-time status tracking and notifications.
  + Document upload and automated verification.
  + Secure communication with police departments.
* **Non-Functional Requirements**:
  + Usability: Intuitive interface for diverse users.
  + Scalability: Support for a large user base.
  + Security: Compliance with data protection standards.
  + Performance: Fast processing and response times.

#### List of Actors

* **Primary Actors**: Citizens, police officers.
* **Secondary Actors**: IT administrators, regulatory bodies.

#### Use Cases

1. User registration and login.
2. Submission of verification applications.
3. Status tracking of applications.
4. Automated document verification.
5. Communication with police departments.
6. Generation of verification reports.

#### Diagrams

* **System Use Case Diagram**: Illustrating interactions between users and the system.
* **Class Diagram**: Detailing system entities and their relationships.
* **Sequence Diagram**: Depicting the sequence of operations for key processes.
* **Architecture Diagram**: Highlighting the system's technical framework.
* **DFD Diagram**: Representing data flow within the system.

### ****Week 1–2: Requirements Gathering and Planning****

* Identify stakeholders (citizens, police officers, admins).
* Define system requirements (functional and non-functional).
* Prepare use case diagrams and tables.
* Develop project plan and timeline.

### ****Week 3–4: System Design****

* Create system architecture and database design.
* Design user interfaces (citizen portal, police portal, admin panel).
* Prepare test cases for initial design validation.

### ****Week 5–7: Development****

#### ****Week 5: Citizen Module****

* Develop citizen registration and login functionality.
* Implement verification request submission.
* Add status viewing feature for citizens.

#### ****Week 6: Police Module****

* Develop login and request processing features for police officers.
* Implement report generation functionality.

#### ****Week 7: Admin Module****

* Develop system administration panel for role management and settings.
* Integrate notifications for citizens.

### ****Week 8–9: Testing****

#### ****Week 8: Unit Testing****

* Test individual modules (citizen, police, admin).
* Validate all test cases for each module.

#### ****Week 9: Integration Testing****

* Test interactions between modules.
* Ensure data flows correctly across the system.

### ****Week 10: User Acceptance Testing (UAT)****

* Conduct testing with real users (citizens, police officers, admins).
* Gather feedback and identify any bugs or usability issues.

### ****Week 11: Deployment****

* Deploy the system on a staging server for final validation.
* Perform security testing and optimize performance.

### ****Week 12: Go-Live and Maintenance****

* Deploy the system to production.
* Monitor system performance and address post-deployment issues.
* Plan for periodic updates and enhancements.

The Police Verification System typically follows a **Three-Tier Architecture** model. This architecture separates the system into three distinct layers, ensuring scalability, maintainability, and security. Here's a breakdown of the layers:

### ****1. Presentation Layer (User Interface Layer)****

* **Purpose**: Interacts with the users (citizens, police officers, and admins).
* **Components**:
  + Web or mobile applications for citizens to register, submit requests, and view status.
  + Police officer portal for processing requests and generating reports.
  + Admin panel for managing system configurations and user roles.
* **Technologies**: HTML, CSS, JavaScript, React, Angular, or mobile frameworks like Flutter.

### ****2. Application Layer (Business Logic Layer)****

* **Purpose**: Handles the business logic and processes user requests.
* **Components**:
  + Processes verification requests.
  + Validates user inputs and enforces rules (e.g., required documents for verification).
  + Manages notifications and report generation.
* **Technologies**: Backend frameworks like Node.js, Django, Spring Boot, or .NET.

### ****3. Data Layer (Database Layer)****

* **Purpose**: Stores and retrieves data for the system.
* **Components**:
  + Citizen information (e.g., name, address, ID proofs).
  + Verification requests and their statuses.
  + Logs of activities and reports.
* **Technologies**: Relational databases like MySQL, PostgreSQL, or NoSQL databases like MongoDB.

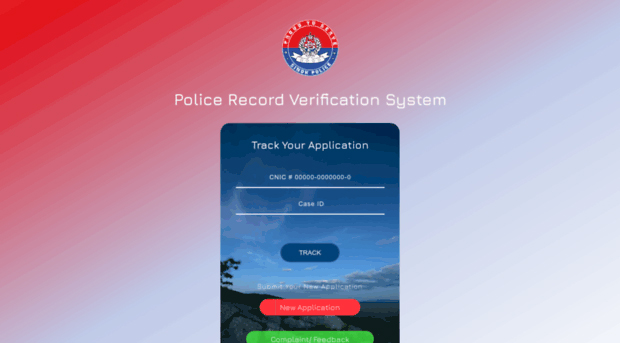
### ****Why Three-Tier Architecture?****

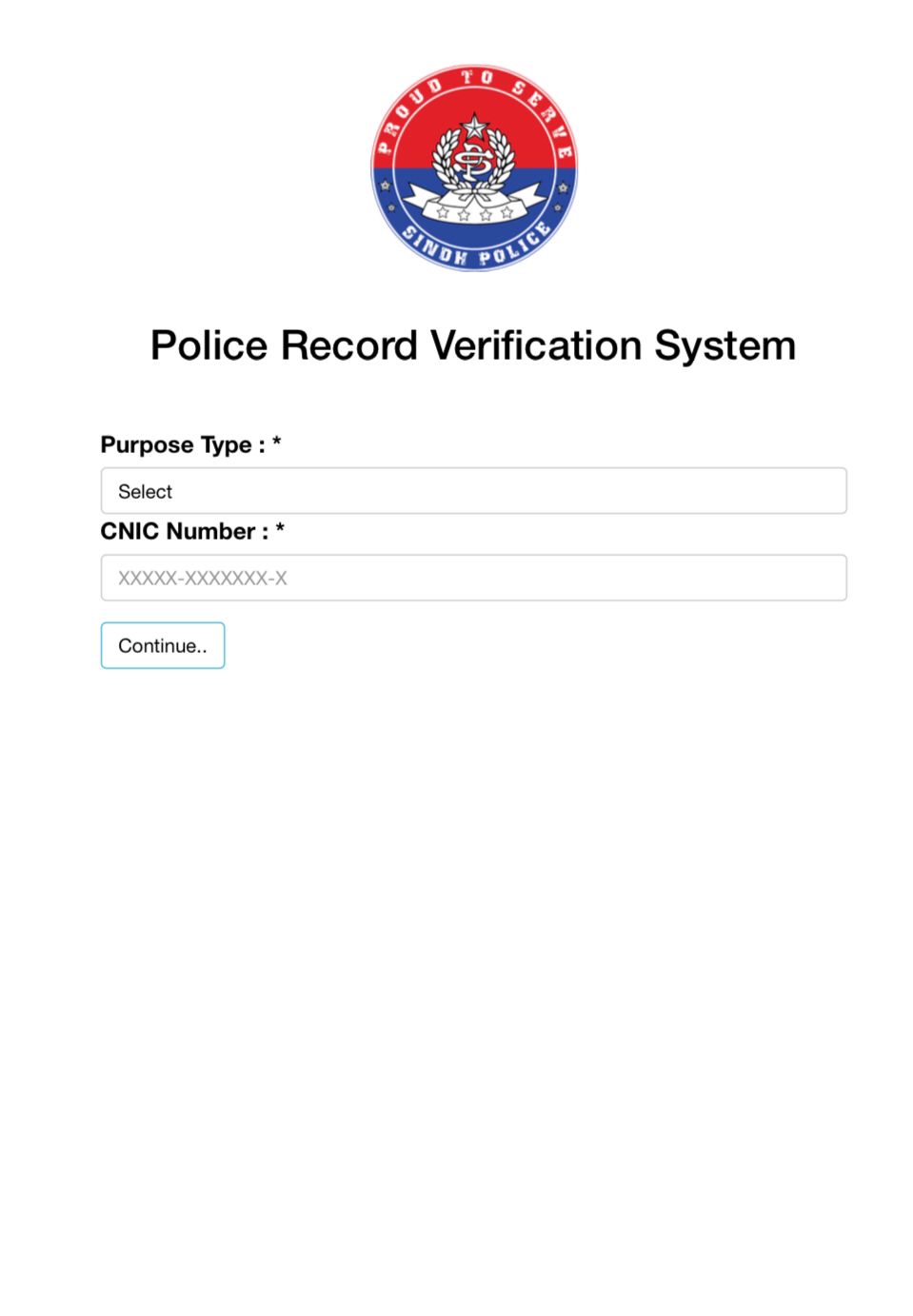
1. **Scalability**: Each layer can be scaled independently based on demand.
2. **Maintainability**: Changes in one layer (e.g., UI) don’t affect others (e.g., database).
3. **Security**: Sensitive data is managed in the database layer, isolated from the presentation layer.
4. **Flexibility**: The system can be deployed on different platforms (cloud, on-premises, or hybrid).

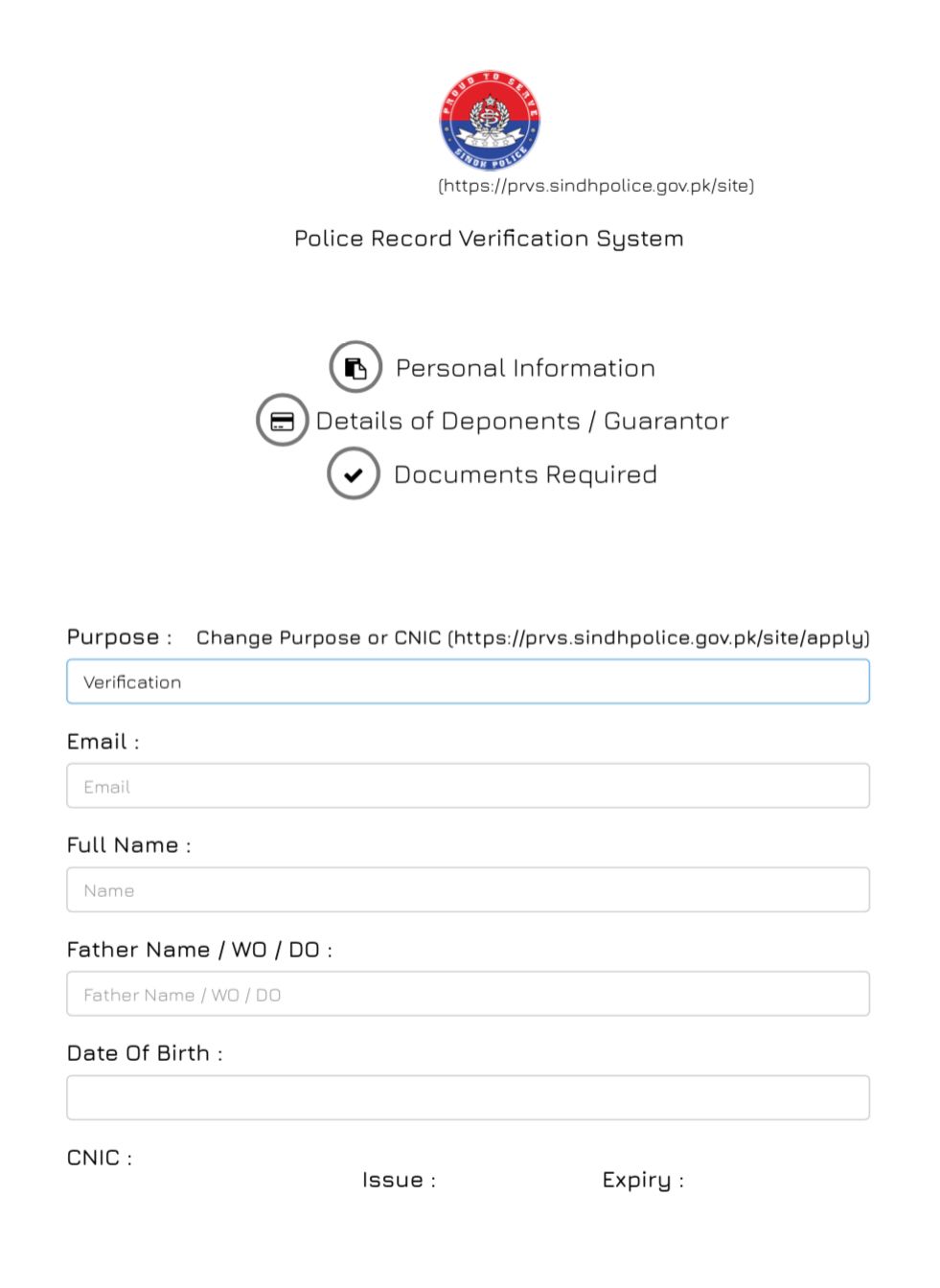
**Prototypes**

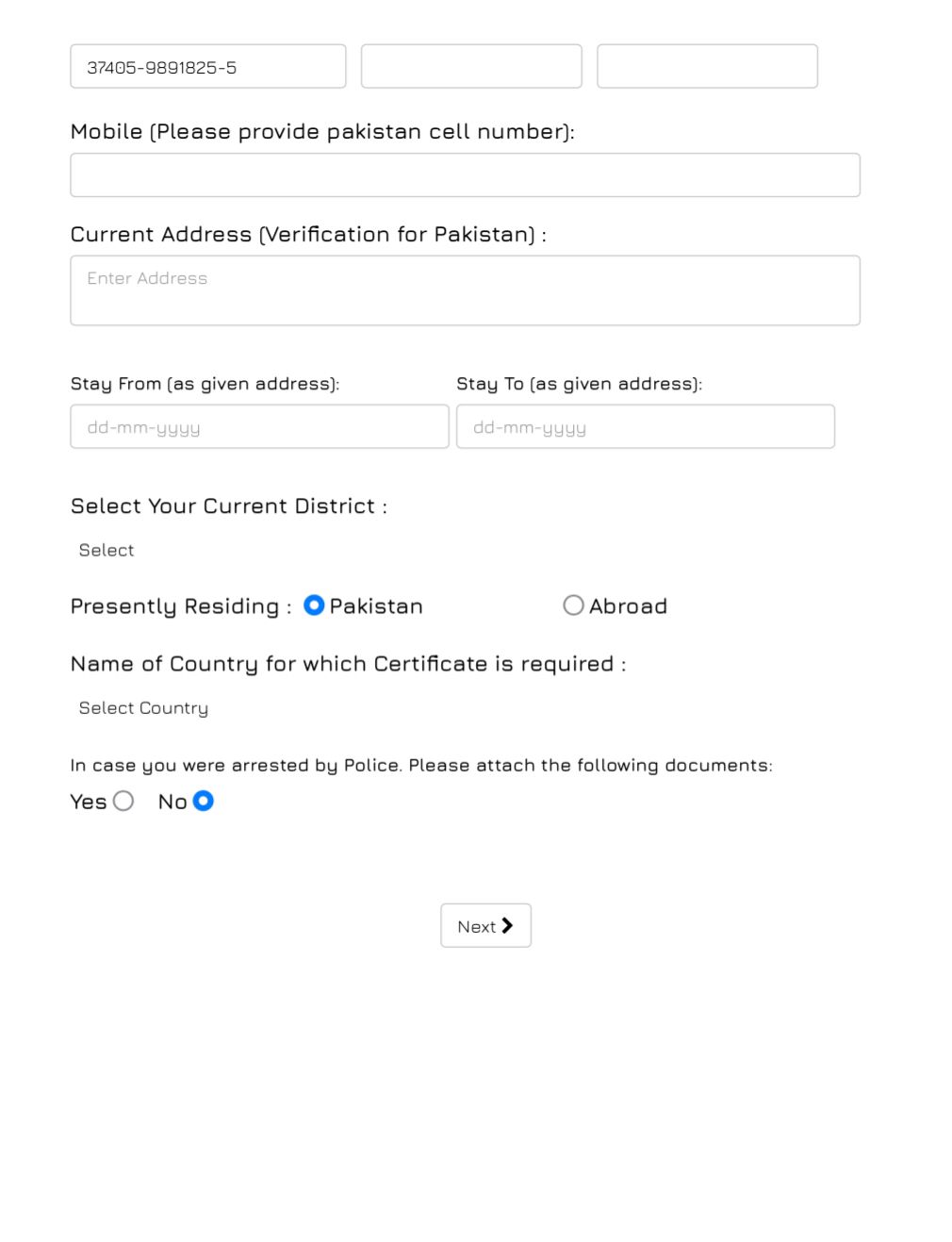
To create prototypes for the **Police Verification System**, we can include the following visual representations of key screens:

1. **Login Screen**: For user authentication.
2. **Application Submission Screen**: Where users submit verification requests.
3. **Status Tracking Screen**: To view the progress of their applications.
4. **Admin Dashboard**: For administrators to manage applications.
5. **Report Generation Screen**: To generate and view verification reports.

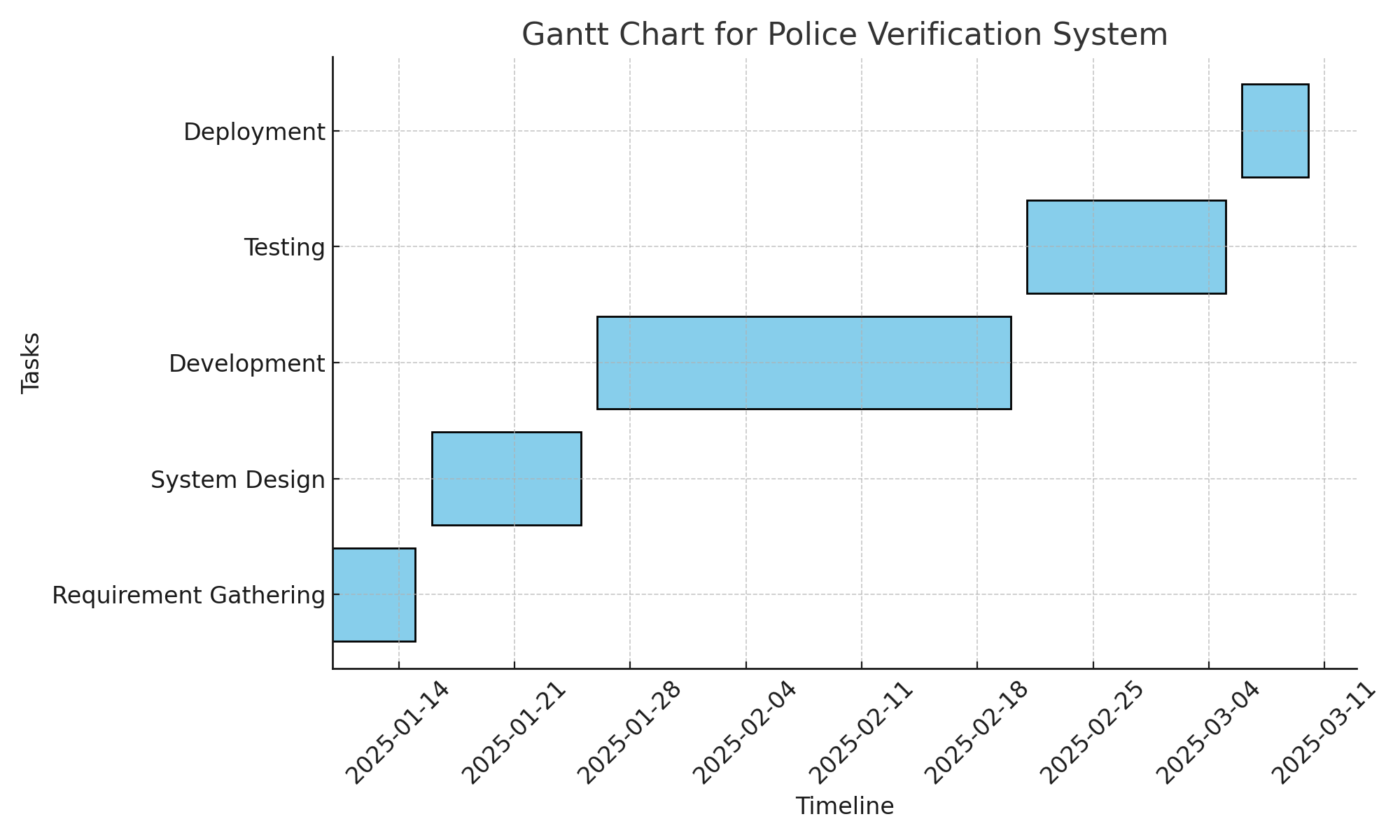


****

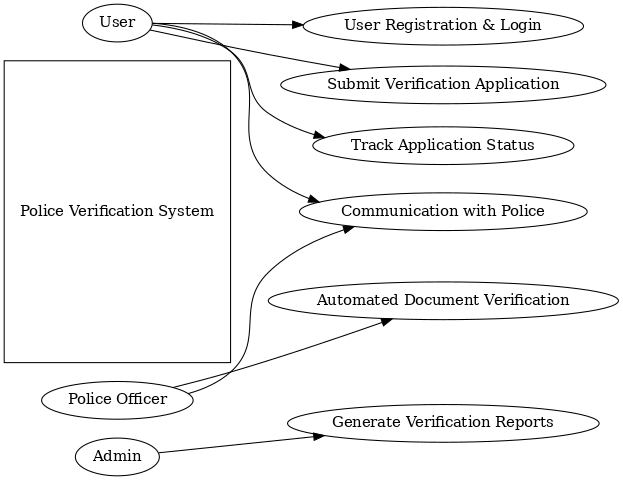




* **Gantt Chart**

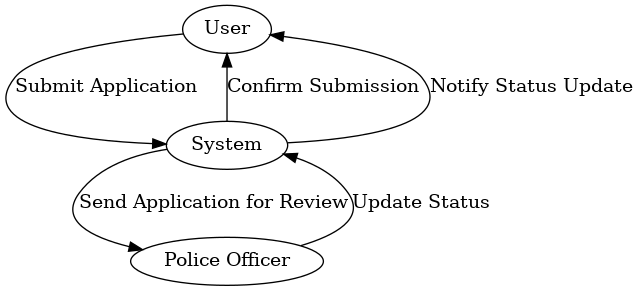
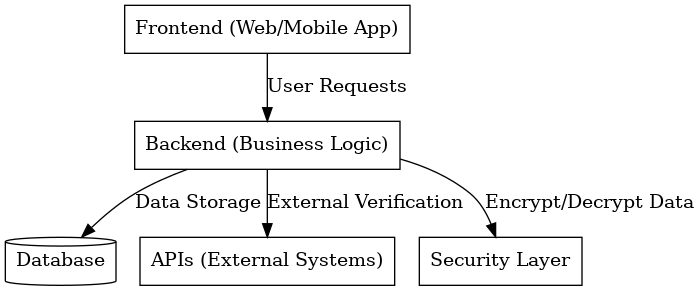
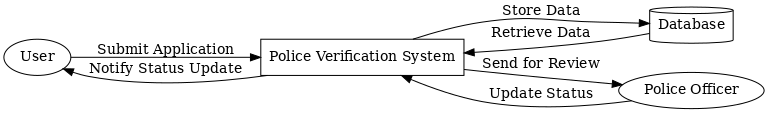


* **System Use Case Diagram**



* **Class Diagram:**



* **Sequence Diagram:**
* 
* **Architecture Diagram:**
* 
* **DFD Diagram:**
* 

**USE CASE TABLE**

| **Use Case ID** | **Use Case Name** | **Actors** | **Description** | **Preconditions** | **Postconditions** |
| --- | --- | --- | --- | --- | --- |
| UC01 | Register Citizen | Citizen | A citizen registers their details in the system for verification. | Citizen must have valid ID proof. | Citizen details are stored in the system. |
| UC02 | Submit Verification Request | Citizen | Citizen submits a request for police verification. | Citizen must be registered. | Request is submitted and logged. |
| UC03 | Process Verification | Police Officer | Police officer processes the verification request by checking documents and conducting inquiries. | Request must be submitted. | Verification status is updated. |
| UC04 | Generate Report | Police Officer | Generates a detailed verification report for the citizen. | Verification process must be complete. | Report is generated and stored. |
| UC05 | Notify Citizen | System | Notifies the citizen about the verification status or report availability. | Verification request must be processed. | Citizen is informed of the status. |
| UC06 | View Status | Citizen | Citizen views the status of their verification request. | Citizen must have an active request. | Status is displayed to the citizen. |
| UC07 | Administer System | Admin | Admin manages user roles, system settings, and logs. | Admin must be authenticated. | System configurations are updated. |

**TEST CASE TABLE**

| **Test Case ID** | **Test Case Name** | **Description** | **Test Steps** | **Expected Result** | **Actual Result** | **Status** |
| --- | --- | --- | --- | --- | --- | --- |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TC01 | Register Citizen | Verify if a citizen can successfully register in the system. | 1. Open registration page. 2. Enter valid details. 3. Submit form. | Citizen is registered, and confirmation is displayed. |  | Pass/Fail |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TC02 | Invalid Registration | Verify system behavior for invalid registration details. | 1. Open registration page. 2. Enter invalid/missing details. 3. Submit form. | Error message is displayed for invalid inputs. |  | Pass/Fail |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TC03 | Submit Verification Request | Verify if a citizen can submit a verification request. | 1. Login as a citizen. 2. Navigate to request page. 3. Fill form and submit. | Request is submitted, and acknowledgment is displayed. |  | Pass/Fail |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TC04 | Process Verification | Verify if a police officer can process a verification request. | 1. Login as a police officer. 2. Open pending requests. 3. Process and update status. | Status is updated successfully in the system. |  | Pass/Fail |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TC05 | Generate Report | Verify if a verification report can be generated. | 1. Login as a police officer. 2. Select completed request. 3. Generate and view report. | Report is generated and displayed successfully. |  | Pass/Fail |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TC06 | Notify Citizen | Verify if a citizen is notified about verification status. | 1. Process a verification request. 2. Check citizen notification. | Citizen receives a notification about the status. |  | Pass/Fail |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TC07 | View Status | Verify if a citizen can view the status of their request. | 1. Login as a citizen. 2. Navigate to status page. 3. View request status. | Status is displayed correctly. |  | Pass/Fail |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TC08 | Administer System | Verify if an admin can manage system settings. | 1. Login as an admin. 2. Navigate to admin panel. 3. Update settings and. | System settings are updated successfully. |  | Pass/Fail |