ONLINE CRIME REPORTING SYSTEM

Submitted in fulfillment of the Requirements for the award of the Degree of

### BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

#### By

Ammai Lagishetti & Tanuja Gotekar

**Under the esteemed guidance of**

**Ms. Anjali Bunkar**

## Department of Information Technology



**B. K. BIRLA COLLEGE, KALYAN**

#### (EMPOWERED AUTONOMOUS STATUS)

***(Affiliated to University of Mumbai)***

**KALYAN, 421304 MAHARASHTRA**

**A.Y. 2023-2024**

**B. K. BIRLA COLLEGE, KALYAN ‘EMPOWERED AUTONOMOUS STATUS’**

***(Affiliated to University of Mumbai)***

### DEPARTMENT OF INFORMATION TECHNOLOGY



**CERTIFICATE**

This is to certify that the project entitled**, "Online Crime Reporting System"**, is bonafied work of **Ammai Lagishetti and Tanuja Gotekar** bearing Seat. No**: 66241061 & 66241104** submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from the University of Mumbai.

#### Internal Guide Head of Department

**External Examiner**

#### Date: College Seal

**PROFORMA FOR THE APPROVAL OF PROJECT PROPOSAL**

***(Note: All entries of the proforma of approval should be filled up with appropriate and***

***complete information. Incomplete proforma of approval in any respect will be summarily rejected )***

#### PRN No : Roll no :

1. Name of the Student
2. Title of the Project
3. Name of the Guide
4. Is this your First submission? Yes No

Signature of the Student Signature of the Guide

Signature of the Coordinator Date:

# Abstract

The Online Crime Reporting System (OCRS) is a web-based application designed to facilitate a streamlined and efficient mechanism for the reporting and management of criminal incidents by citizens and their subsequent handling by law enforcement agencies. Developed using the Java programming language, this innovative platform offers a dual-module interface catering to both users (complainants) and administrators (law enforcement officials).

On the administrative front, the OCRS equips law enforcement officials with a comprehensive toolkit for effective case management. Administrators have the capability to view submitted complaints, update case statuses, and proceed with legal actions such as registering FIRs and charge sheets. A pivotal feature includes assigning specific officers to cases, thereby ensuring a specialized and focused approach to each incident. The system further enhances operational efficiency by generating consolidated crime reports, which include all pertinent data from complaints to final charge sheets.

To bolster communication and provide continuous support, the OCRS incorporates an integrated email system and a chatbot, facilitating direct interaction between users and administrators. While the email feature is exclusive to administrators for secure communication, the chatbot serves as an instant support mechanism for users. This innovative project not only simplifies the crime reporting process for citizens but also significantly aids law enforcement in their investigative and procedural duties, marking a significant advancement in the application of technology for public safety and justice.

# ACKNOWLEDGEMENT

I would like to express my special thanks of gratitude to my mentor Ms. Anjali Bunkar, who gave us the golden opportunity to do this wonderful project of Information Technology on “ ONLINE CRIME REPORTING SYSTEM”, and who also helped me in completing my project. I came to know about so many new things I am thankful for them.

Thanking you

# DECLARATION

I hereby declare that the project entitled, “Online Crime Reporting System” done at kalyan where the project is done, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY) to be submitted as final semester project as part of our curriculum.

Tanuja Gotekar Ammai Lagishetti

# TABLE OF CONTENTS

[Chapter 1: Introduction 1](#_TOC_250020)

* 1. [Background 2](#_TOC_250019)
  2. [Objectives 2](#_TOC_250018)
  3. Purpose and Scope 4
     1. [Purpose 4](#_TOC_250017)
     2. [Scope 4](#_TOC_250016)
     3. [Applicability 4](#_TOC_250015)
  4. [Achievements 4](#_TOC_250014)
  5. Organization 5

[Chapter 2: System Analysis 6](#_TOC_250013)

* 1. [Existing System 7](#_TOC_250012)
  2. [Proposed System 8](#_TOC_250011)
  3. [Requirement Analysis 9](#_TOC_250010)
  4. Software Requirements 9
  5. Hardware Requirements 11
  6. Justification of selection of Technology 11

[Chapter 3: System Design 16](#_TOC_250009)

* 1. [Module Division 17](#_TOC_250008)
  2. [Data Dictionary 18](#_TOC_250007)
     1. [Gantt Chart 21](#_TOC_250006)
  3. [Conceptual Models 21](#_TOC_250005)

[Chapter 4: Implementation and Testing 28](#_TOC_250004)

* 1. Code (Place Core Segments) 29
  2. Testing Approach 50
     1. Unit Testing (Test cases and Test Results) 50
     2. Integration System (Test cases and Test Results) 52

Chapter 5: Results and Discussion 54

[Chapter 6: Conclusion and Future Work 61](#_TOC_250003)

* 1. [Conclusion 62](#_TOC_250002)
  2. [Future Work 62](#_TOC_250001)

[Chapter 7: References 63](#_TOC_250000)

**List of figures**

### Fig 2.1: Data Flow Architecture 15

Fig 3.1: Gantt Chart 21

### Fig 3.3.1: Flow Diagram 22

Fig 3.3.2: ER Diagram 23

### Fig 3.3.3: Level 1-DFD 25

Fig 3.3.4: Level 2-DFD 25

### Fig 3.3.5: Use Case Diagram 26

Fig 3.3.6: Class Diagram 27

### Fig 5.1: Home Page 55

Fig 5.2: Admin Dashboard 55

### Fig 5.3: User Dashboard 56

Fig 5.4: UI for Display Complaint Table 56

### Fig 5.5: User Interface for FIR Details 57

Fig 5.6: Chargesheet Details 57

### Fig 5.7: User Interface for Chargesheet Details 58

Fig 5.8: User Interface for Report 58

### Fig 5.9: Case Report 59

Fig 5.10: Email Form 60

Fig 5.11: User Interface for Chatbot 60

# Chapter 1 : Introduction

## Background:

## The Online Crime Reporting System is a digital platform designed to revolutionize the process of reporting and managing criminal complaints. With two distinct modules for users and administrators, the system ensures a seamless experience for both parties involved. Registered users can easily log in, file complaints, and provide evidence details, all within a secure online environment. They can then track the progress of their complaints, receiving updates on the status of FIR registration and charge sheet generation. On the admin side, authorized personnel can efficiently manage incoming complaints, update their status, register FIRs, generate charge sheets, and assign police personnel to handle specific cases. Ultimately, the system aims to enhance accessibility, efficiency, and transparency in the reporting and resolution of criminal incidents.

## By leveraging digital technology, the Online Crime Reporting System streamlines administrative processes, saving time and resources for law enforcement agencies. Automation of tasks such as complaint registration and report generation ensures swift and accurate handling of cases. Moreover, the system promotes transparency by allowing users to monitor the progress of their complaints in real-time. Comprehensive crime reports generated upon the completion of cases provide valuable insights into criminal activities and investigations, enabling informed decision-making and strategic analysis. Overall, the Online Crime Reporting System represents a significant step forward in modernizing law enforcement efforts, empowering both citizens and administrators in the fight against crime.

## 1.2 Objectives

The project aims to reduce the paperwork by registering crime and storing the data in a database which helps the police to increase the speed of investigation, enabling them to solve the crime more efficiently with the help of crime reports which is useful in future for identifying crime hotspot. Also, it includes crime-related news articles from which public awareness can be taken. The authentication and authorization were crosschecked at all the relevant stages.

Following are objectives:

* + - Easy to use
    - Public awareness
    - Paperless Reporting
    - Real time communication
    - Improved law enforcement

## Purpose, Scope and Applicability

### Purpose

### The Online Crime Reporting System is to provide a comprehensive and streamlined platform for users to report crimes and for administrators to efficiently manage and track these reports. For users, the system offers a secure login mechanism where they can register complaints along with detailed evidence. This ensures that law enforcement agencies receive accurate and thorough information to initiate investigations promptly. Users can also stay informed about the status of their complaints, FIRs (First Information Reports), and chargesheets, fostering transparency and trust in the law enforcement process.

#### On the administrator side, the system enables efficient handling of reported complaints. Administrators can view and update complaint statuses, assign police personnel to specific cases, and oversee the registration of FIRs and chargesheets. By consolidating all data into a single crime report format, the system facilitates seamless communication and collaboration between users, administrators, and law enforcement agencies, ultimately enhancing the effectiveness of crime management efforts and promoting public safety.

#### The purpose of this system is threefold:

* + - * **Efficient Crime Management**: Streamline the process of reporting and managing crimes, allowing law enforcement to respond swiftly and effectively to ensure public safety.
      * **Enhanced User Engagement**: Empower the public to actively participate in crime reporting and monitoring, fostering a sense of community involvement and security.
      * **Data-Driven Insights**: Generate comprehensive crime reports for analysis, aiding law enforcement in identifying trends and patterns, which is invaluable for future crime prevention strategies and enhancing overall public safety.

### Scope

The scope of the Online Crime Management System is broad and encompasses various critical aspects of crime management and law enforcement. Firstly, it addresses the pressing issue of crime data management. By transitioning from manual record-keeping to a digital platform, the

project streamlines the recording and tracking of crime-related data. This not only eliminates the inefficiencies and errors associated with paperwork but also ensures that a comprehensive and accurate record of crimes is maintained.

Secondly, the project significantly enhances law enforcement efficiency. It achieves this by introducing a FIR (First Information Report) system that allows for the swift registration of serious crimes, such as murder. This expedited process ensures that law enforcement agencies can respond promptly to critical incidents, ultimately improving the resolution of cases.

Additionally, the system's scope extends to crime report generation, facilitating the creation of detailed reports that enable law enforcement to identify crime patterns and hotspots. This data- driven approach empowers authorities to proactively implement measures for crime prevention.

Lastly, the project prioritizes security through authorization measures. By implementing stringent access controls, it ensures that only authorized personnel can access and manage sensitive crime data, safeguarding its confidentiality and integrity.

Functionality provided by Online Crime Management System are as follows:

* It tracks all the crime related data.
* Generate the crime report.
* Provide the mail feature.
* Manage the information of crime.
* Transparency and User Engagement

### 1.3.3 Applicability

The online crime reporting system offers a user-friendly platform for complainants to submit complaints with evidence. Administrators can efficiently manage cases, assign police personnel, and track progress. Real-time updates ensure transparency for users, while standardized reports streamline documentation and analysis. Its applicability spans law enforcement, judiciary, and community safety sectors, enhancing efficiency and fostering public trust.

## Achievements

1. Efficient Crime Reporting Process: The system streamlines the process of reporting crimes by allowing users to easily submit complaints with detailed evidence, reducing the time and effort required for traditional reporting methods.
2. Enhanced User Experience: Users can conveniently log in using their registered email and password, providing a seamless experience. The interface is user-friendly, making it easy for individuals to navigate and submit complaints.
3. Improved Transparency and Accountability: The system promotes transparency by allowing users to track the status of their complaints, FIRs, and chargesheets in real-time. This transparency enhances accountability within the law enforcement agencies involved.
4. Faster Resolution of Cases: By digitizing the entire process, from complaint submission to case closure, the system accelerates the resolution of cases. Assigning police personnel to specific cases and monitoring their progress ensures timely action.
5. Centralized Data Management: All data related to complaints, FIRs, chargesheets, assigned police personnel, and case reports are stored in a centralized database. This facilitates easy access, retrieval, and management of information for both users and administrators.

## Organization of report

A short overview for the chapters which are included in this report is as followed;

* Chapter 1: Introduction

This chapter focus on the overview and background of the Online Crime Management System Project.

* Chapter 2: Survey of technologies

In this chapter the major focus is on the brief overview about used technologies in the project.

* Chapter3: Requirement and Analysis

These chapter focuses on requirements what user is required or supposed from systemto do. During development phase re-validating the requirements with actual requirement comparison noted in these chapter.

* Chapter 4: System Design

What the system will contained, what are the different modules are discussed in the section.

* Chapter 5; Implementation and Testing

This chapter is about the Implementation of various modules. As the project is proceeding with the testing various results are expected during the testing of the project.

# 

# Chapter 2 : System Analysis

A web-based application, often referred to as a web app, is a software program that operates within a web browser. It runs on remote servers and is accessed by users through the Internet. Web apps are versatile, as they work on various devices and operating systems, eliminating the need for installation. They can serve various purposes, from simple tasks like email and document editing to complex functions like online banking and e-commerce. Web apps are known for their accessibility, ease of updates, and the ability to deliver real-time information, making them a popular choice for businesses and individuals seeking flexibility and wide-ranging usability.

### How Web application work?

Web-based applications function through a client-server architecture. When a user accesses a web-based application in their web browser, the client-side, or user's device, sends a request to the application's server. This request contains a URL, indicating which web page or function the user wants to access. Upon receiving this request, the server processes it by retrieving data from a database, performing computations, or executing other necessary tasks based on the application's design. The server then sends back the processed data and instructions in the form of a web page, typically composed of HTML, CSS, and JavaScript.

Once the web page is received by the user's browser, it's interpreted and displayed as a visual interface. This interface enables users to interact via clickable links, buttons, and forms, facilitating navigation, data input, and function triggering. User actions are then sent to the server, maintaining a seamless interaction with the remote application. This architecture ensures accessibility across devices with internet connectivity, making web-based applications versatile and popular for tasks ranging from email and social media to e-commerce and productivity tools.

* 1. **Existing System**

In the existing system, access to information about police stations in a state is limited and typically available only to authorized personnel. This restricted access often places a heavy workload on the authorized individuals responsible for managing and processing the information. This means that the general public has limited means of directly engaging with the law enforcement system to report crimes or complaints about specific incidents, cities, or individuals.

In contrast, the proposed system offers a significant advancement in terms of user engagement and access. By allowing users to register on the platform, it empowers individuals to actively participate in the process of reporting crimes and registering complaints. This fundamental shift democratizes the system, providing a direct channel for the public to contribute essential information, fostering community involvement in law enforcement activities.

The proposed system not only lightens the workload for authorized personnel but also facilitates a more transparent and efficient crime reporting process. Users can submit crime reports and complaints about specific cities or individuals, which are then channeled directly to the relevant authorities. This direct engagement with the public and streamlining of the reporting process enhances the responsiveness of law enforcement agencies, ultimately contributing to a safer and more engaged community.

## 2.2 Proposed System

The Crime Reporting System is a robust web-based application employing JSP, Servlets, and JDBC technologies to enhance the management of criminal complaints. It consists of two main modules: the User Module and the Admin Module, designed to facilitate seamless interaction between users and administrators.

### User Module:

1. User Registration and Complaint Submission: Users can register complaints by providing detailed information about the crime, including crime type, area, location, and date.
2. Complaint Status: Users can track the status of their complaints, enabling them to stay informed about the progress.
3. FIR Request: In cases of severe crimes like murder or rape, the system allows administrators to contact users via email to discuss the need for registering an FIR.
4. FIR Report and Chargesheet: Users can access and view FIR reports generated by the admin and the associated chargesheets.
5. Data Contribution: User complaint details feed into a database for admin to generate reports.

### Admin Module:

1. Complaint Management: Administrators can view and manage user complaints, assign police officers to specific cases, and update complaint statuses.
2. FIR Registration: Admins can register FIRs for serious offenses after consulting with the users, ensuring a proper legal process is followed.
3. Assign Police: Admin can assign the police to particular case after acceptance of complaint.
4. Chargesheet Registration: Admin can register the chargesheet after the completion of FIR.
5. Report Generate: The system offers administrators tools to generate reports via MySQL queries and fetch data using JSP, facilitating comprehensive case tracking in a single report.
6. News module: Admins update early news to inform users and visitors about crimes.
7. Email: Admin can email users to communicate regarding specific cases.

This comprehensive system enhances the efficiency of crime reporting, ensuring better communication between users and administrators, enabling data-driven decisions, and ultimately contributing to safer communities by addressing and preventing criminal activities effectively.

## Requirement Analysis

Requirement analysis is a pivotal phase in project management, focused on understanding and documenting the specific needs, objectives, and constraints of a project. It involves a systematic process of gathering, categorizing, and prioritizing both functional and non-functional requirements.

Functional requirements:

* User
* Hardware
* Software
* Communications
* System Features

Non-Functional Requirements:

* Safety
* Security
* Performance

## Software

* Programming Language: Java

Java's platform independence ensures that the application can run on various operating systems without modification, providing flexibility and accessibility to a wide range of users, including law enforcement personnel who may use diverse devices.

* Web Technology: HTML, CSS, JDBC, JSP

JDBC (Java Database Connectivity) is a fundamental Java technology that serves as a bridge between Java applications and relational databases. It enables Java applications to connect, interact, and exchange data with databases such as MySQL, Oracle, and PostgreSQL. JDBC is a crucial component for database-driven applications, providing a standardized and robust interface for managing database operations.

JavaServer Pages (JSP) is a technology used in web development to create dynamic and interactive web pages. It allows developers to embed Java code within HTML pages, enabling the execution of server-side logic and the generation of dynamic content. JSP pages are processed on the server, making it possible to generate dynamic content and interact with databases or other resources. This technology simplifies the development of web applications and is commonly used in conjunction with Java servlets to build robust and feature-rich web applications.

* Operating System: Windows 10, Windows 11.

This project work is done on the windows 11, which is the operating system. An operating system is a set of software tools designed to make it easy for people or programmers to make optimum use of the computer. People who use computers have different levels of needs and interest. These peoples can be separated can be two groups, users and programmers. The user wants a convenient set of commands to manage files of data or programs, copy and run application package while a programmer used as a set of tools that can be held together and debug programs.

* Database: MySql.

MySQL is a popular, easy-to-use open-source database system widely used for web applications. It's known for its speed, reliability, and works well with languages like PHP, Python, and Java. With MySQL, you can store and manage structured data and perform complex queries. It's secure, supports transactions, and scales from small projects to large enterprises, making it versatile for various needs.

* Software: Netbeans.

NetBeans is an integrated development environment (IDE) that provides a versatile and user-friendly platform for software development. It supports various programming languages, including Java, PHP, C++, and HTML, making it suitable for a wide range of application development tasks. NetBeans offers features like code editing, debugging, and version control, facilitating efficient software development. It also supports a wide array of plugins and extensions, allowing developers to customize their development environment to suit their specific needs and preferences.

## 2.4 Hardware

* Processor – i3.
* Hard disk- 1tp
* Memory – 4GB RAM.
* Processor Speed – Dual Core 2.40 GHz

## Justification of Selection of Technologies:

Java is a small, simple, safe, object-oriented, interpreted or dynamically optimized, byte-coded, architectural, garbage-collected, multithreaded programming language with a strongly typed exception handling for writing distributed and dynamically extensible programs. Java supports this application and the following features make it one of the best programming languages.

* + - It is simple and object-oriented
    - It helps to create user-friendly interfaces.
    - It is platform-independent
    - It is highly secure and robust.
    - It supports Internet programming

Firstly, Java's platform independence ensures that the application can run on various operating systems without modification, providing flexibility and accessibility to a wide range of users, including law enforcement personnel who may use diverse devices. This enables seamless access to the system from different locations, promoting efficient crime data management.

Java's robust security features, such as sandboxing and extensive libraries, are crucial for handling sensitive crime-related data in the online crime reporting system. Its access controls and encryption mechanisms ensure data confidentiality and integrity, essential for law enforcement. With a vast developer community and resources, Java simplifies system development and maintenance, ensuring longevity and adaptability to evolving needs. Overall, Java technology provides the necessary tools and security measures to create a reliable and accessible online crime reporting system that meets modern demands.There are following technologies used in project:

* JDBC:

1. JDBC connects Java apps with databases like MySQL, Oracle, PostgreSQL, facilitating smooth data exchange. Its standardized interface simplifies database tasks, crucial for building database-driven Java software.
2. Driver Manager: Driver Manager handles a list of database drivers in JDBC, facilitating connection establishment by selecting the suitable driver from the list based on the database's URL.
3. PreparedStatement in JDBC: Precompiled SQL with parameter placeholders for security and improved query performance.
4. Batch Processing: JDBC supports batch processing, which allows multiple SQL statements to be executed as a single batch. This feature is useful for optimizing performance when performing a series of database operations.
5. Exception Handling: JDBC handles exceptions and errors generated during database interactions. Proper exception handling ensures that the application can gracefully recover from database-related issues.
6. Transaction Management: JDBC offers transaction management capabilities, enabling the grouping of multiple database operations into a single transaction. This ensures that either all operations within a transaction succeed or fail as a single unit, maintaining data integrity.
7. Type Mapping: JDBC provides type mapping between Java data types and SQL data types. It helps in seamlessly transferring data between Java and the database without loss of precision or data integrity.
8. Data Source: JDBC supports data sources that provide connection pooling and connection management features. Using a data source can simplify database access in applications.

In summary, JDBC is a versatile and powerful technology for database access in Java applications. It simplifies database connectivity, supports transaction management, improves security, and enhances performance. JDBC is a critical component for developing robust and data-driven applications that interact with relational databases.

* **Java Server Pages (JSP)**

Java Server Pages technology is the Java platform technology for building applications containing dynamic Web content such as HTML, DHTML and XML. The Java Server Pages technology enables the authoring of Web pages that create dynamic content easily but with maximum power and flexibility.

The Java Server Pages technology offers a number of advantages:

* Write Once, Run Anywhere Properties: JSP Platform-neutral tech for dynamic web pages and server components. Develop anywhere, deploy on any server, access from any browser.
* High Quaity Tool Support: JSP enables choice of top tools due to Write Once, Run Anywhere feature. Design aims for high-quality, portable tool creation.
* Support for scripting and actions: JSP supports scripting elements for flexibility and actions for encapsulating functionality. Scripts enable per-page functionality assembly, while actions offer convenient manipulation by tools.
* Tomcat Server:Open-source server for Java web apps, essential for deployment and execution. Integrated seamlessly with NetBeans IDE 8.2 for streamlined development.

1. Servlet Container and Web Server:

Tomcat is primarily known as a servlet container and web server. It provides a runtime environment for Java Servlets, JavaServer Pages (JSP), and other Java-based web technologies. When integrated with NetBeans 8.2, developers can create, develop, and deploy dynamic web applications with ease. Tomcat acts as the engine that processes incoming web requests, executes servlets, and delivers dynamic content to clients' web browsers.

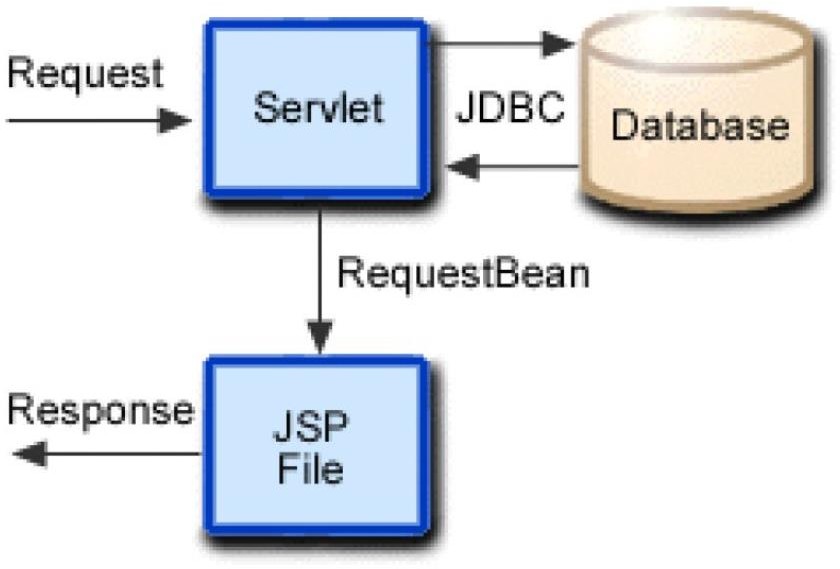
1. Java EE Compatibility:

Tomcat is designed to be lightweight and focuses on the implementation of Java Servlet and JavaServer Pages technologies. While it doesn't fully support the entire Java EE (Enterprise Edition) stack, it provides a robust platform for developing web applications without the complexities associated with more comprehensive Java EE application servers. This makes it an excellent choice for projects with simpler requirements.

1. Development and Debugging:

NetBeans 8.2 offers seamless integration with Tomcat, allowing developers to develop, test, and debug web applications within the IDE. Developers can set breakpoints, inspect variables, and perform step-by-step debugging in their Java web applications running on the Tomcat server directly from the NetBeans IDE.

Apache Tomcat integrated with NetBeans 8.2 offers a developer-friendly platform for Java-based web app creation and testing. Its lightweight design and Java EE compatibility make it popular, particularly for projects with simpler needs. This integration ensures smooth development and debugging for Java web applications.

* Web development using JSP, JDBC and Servlet

*Fig 2.1 DATA FLOW ARCHITECTURE*

The Servlet-JSP-JDBC data flow architecture is a fundamental structure for building dynamic web applications in Java. Servlets act as the controller, receiving and processing incoming HTTP requests from clients. They can perform tasks like handling user input, processing business logic, and interacting with a database. JSP, on the other hand, is used to create dynamic web pages with embedded Java code, allowing developers to generate HTML content based on data retrieved from the database through JDBC. JDBC (Java Database Connectivity) serves as the bridge between the Java application and the database, enabling the retrieval and manipulation of data, as well as the insertion or updating of information in the database. This architecture creates a powerful and flexible system for building web applications that can interact with databases to deliver dynamic and data- driven content to users.

# Chapter 3 : System Design

## Module Division:

A crime management system is designed to efficiently handle various aspects of managing criminal cases and investigations. It comprises several essential modules to ensure effective functionality.

### Modules:

* Registered users
* Admin

#### The registered user module includes

* Add Complaint: This module helps the user to report online complaints.
* Edit Complaint: This module helps the user to edit his complaint details.
* Edit Account: This module helps the user to update his or her profile.
* View complaint status: This module allows us to view the status of all complaint that you have posted earlier.
* View News: User can view the latest crime related news.
* Add FIR: This module helps the user to add the FIR for the crimes like murder, rep,etc.
* View crime status: This module allows us to view the status of the all crimes that you have posted earlier.
* Chat: This module help the user to chat with the administrator or with other registered users
* Mail: This module helps the user to send mail to the administrator.

#### The administrator module includes

* View and reply user complaint: This module helps the admin to view and reply user’s complaint details
* View complaint status: This module allows us to view the status of all complaint that you have posted earlier.
* Add and delete latest news: This module helps the admin to add and delete latest news.
* Add and view FIR: This module helps the admin to add and view Fir reports
* Chat: This module help the admin to chat with the administrator or with other registered users
* Mail: This module helps the user to send mail to the user.
* Assign police: admin can assign the police.

## Data Dictionary:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Constraints** |
| id | int(200) | NO | PRI | NULL | auto\_increment |
| name | varchar(20) | NO |  | NULL |  |
| email | varchar(20) | NO | UNI | NULL |  |
| password | varchar(20) | NO |  | NULL |  |
| date\_column | date | YES |  | 2023-01-01 |  |

#### *Table 3.1: reg*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Constraints** |
| id | int(11) | NO | PRI | NULL | auto\_increment |
| email | varchar(30) | NO |  | NULL |  |
| password | varchar(30) | NO |  | NULL |  |
| code | varchar(20) | NO |  | NULL |  |

*Table 3.2: admin*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Constraints** |
| fid | int(11) | NO | PRI | NULL | auto\_increment |
| id | int(11) | YES | MUL | NULL |  |
| section | varchar(255) | NO |  | NULL |  |
| complaint\_type | varchar(255) | NO |  | NULL |  |
| reg\_date | date | NO |  | NULL |  |
| fir\_detail | text | NO |  | NULL |  |
| fir\_start\_date | varchar(50) | NO |  | NULL |  |
| fir\_end\_date | varchar(50) | NO |  | NULL |  |
| status | varchar(50) | YES |  | Pending |  |

#### *Table 3.3: fir*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Constraint** |
| id | int(11) | NO | PRI | NULL | auto\_increment |
| user\_id | int(11) | NO | MUL | NULL |  |
| name | varchar(255) | NO |  | NULL |  |
| email | varchar(255) | NO |  | NULL |  |
| phone\_no | varchar(15) | NO |  | NULL |  |
| police\_station\_city | varchar(255) | NO |  | NULL |  |
| police\_station\_state | varchar(255) | NO |  | NULL |  |
| complaint\_detail | text | NO |  | NULL |  |
| area | varchar(255) | NO |  | NULL |  |
| incident\_city | varchar(255) | NO |  | NULL |  |
| type\_of\_crime | varchar(255) | NO |  | NULL |  |
| date | date | NO |  | NULL |  |
| accused\_name | varchar(255) | NO |  | NULL |  |
| accused\_address | varchar(255) | NO |  | NULL |  |
| accused\_phone\_no | varchar(15) | NO | UNI | NULL |  |
| victim\_name | varchar(255) | NO |  | NULL |  |
| victim\_address | varchar(255) | NO |  | NULL |  |
| victim\_phone\_no | varchar(15) | NO | UNI | NULL |  |
| evidence\_detail | text | NO |  | NULL |  |
| imageFileName | varchar(255) | YES |  | NULL |  |
| status | varchar(255) | YES |  | NULL |  |

*Table 3.4: Complaint*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Constraints** |
| id | int(11) | NO | PRI | NULL | auto\_increment |
| fid | int(11) | YES | MUL | NULL |  |
| section | varchar(255) | NO |  | NULL |  |
| briefdesc | varchar(255) | NO |  | NULL |  |
| filename | varchar(255) | NO |  | NULL |  |
| offense | varchar(255) | NO |  | NULL |  |
| accused | varchar(255) | NO |  | NULL |  |
| charge\_date | date | NO |  | NULL |  |
| status | varchar(50) | NO |  | NULL |  |

#### *Table 3.5: chargesheet*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Constraints** |
| pid | int(11) | NO | PRI | NULL | auto\_increment |
| name | varchar(255) | NO |  | NULL |  |
| email | varchar(255) | NO |  | NULL |  |
| station | varchar(255) | NO |  | NULL |  |
| designation | varchar(255) | NO |  | NULL |  |
| pfilename | varchar(255) | NO |  | NULL |  |
| gender | varchar(10) | NO |  | NULL |  |
| cont\_no | varchar(15) | NO |  | NULL |  |
| pstatus | varchar(20) | NO |  | NULL |  |
| id | int(11) | YES | MUL | NULL |  |

*Table 3.6: police*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Constraints** |
| id | int(11) | NO | PRI | NULL | auto\_increment |
| chid | int(11) | YES | MUL | NULL |  |
| crdate | date | NO |  | NULL |  |
| pname | varchar(255) | NO |  | NULL |  |
| section | varchar(255) | NO |  | NULL |  |
| cdetail | text | NO |  | NULL |  |
| paddress | varchar(255) | NO |  | NULL |  |
| pimgfilename | varchar(255) | YES |  | NULL |  |
| note | text | YES |  | NULL |  |

#### *Table 3.7: report*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Constraints** |
| id | int(20) | NO | PRI | NULL | auto\_increment |
| title | varchar(150) | NO |  | NULL |  |
| date\_time | varchar(40) | NO |  | NULL |  |
| content | varchar(1000) | NO |  | NULL |  |

*Table 3.8: news*

#### 3.2.1 Gantt Chart

#### 

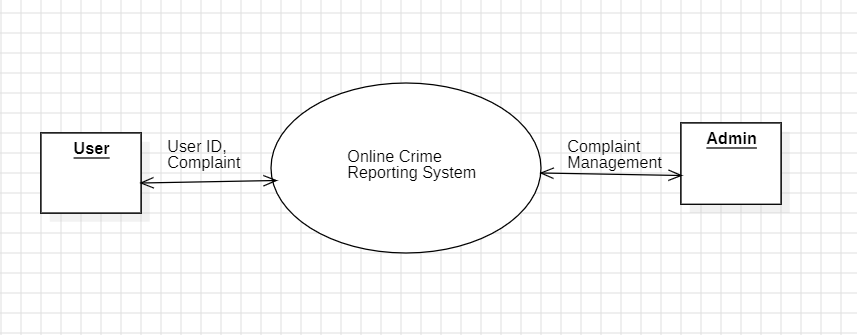
*Fig 3.1 Gantt Chart*

## Conceptual Models

In the realm of software development, a conceptual model may be used to represent relationships of entities within a database. A conceptual model can easily represent abstract concepts of the relationships between objects in the system, such as Users and their relationships to accounts.

1. Flow Diagram:

A flow diagram shows how data is processed within a system based on inputs and outputs. Visual symbols are used to represent the flow of information, data sources and destinations, and where data is stored. Data flow diagrams are often used as a first step toward redesigning a system. They provide a graphical representation of a system at any level of detail, creating an easy-to-understand picture of what the system does.



*Fig 3.1 Flow Diagram*

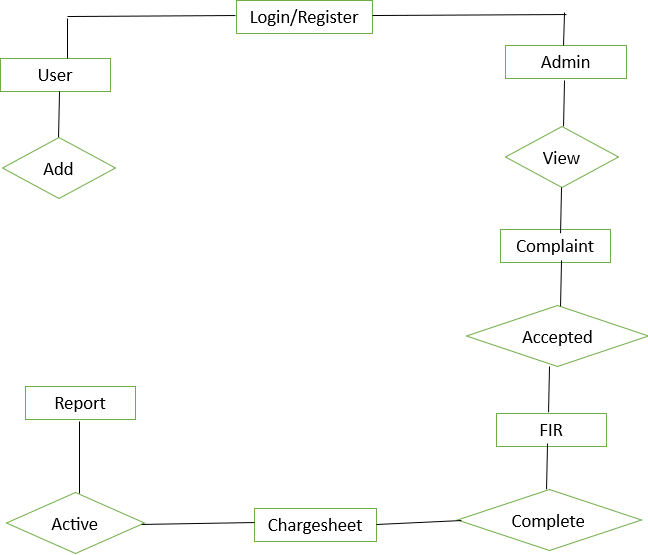
1. ER Diagram

ER diagram stands for the entity relationship diagrams that displays relationship of the entity sets stored in the database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that defines its properties. In software engineering an ER model is commonly formed to represent things that a business needs to remember in order to perform business processes. The below chart represents the symbol of the ER diagram.

#### Table 3.9: symbols of ER diagram

|  |  |  |  |
| --- | --- | --- | --- |
| Sr.No. | Name | Symbol | Description |
|  | Rectangle |  | Represent Entity Set |
|  | Double Rectangle |  | Represents Weak Entity Set |
|  | Ellipse |  | Represents Attribute |
|  | Double Ellipse |  | Represents Multi- valued Attributes |

|  |  |  |  |
| --- | --- | --- | --- |
|  | Diamond |  | Represents Relationship Set |
|  | Double Lines |  | Represents Total Participation |



*Fig 3.2 ER Diagram*

1. DFD

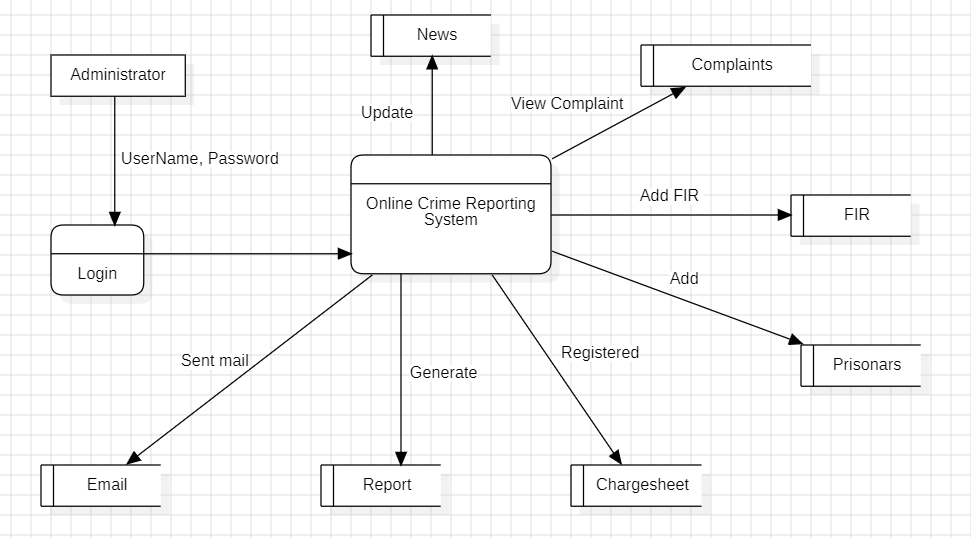
**DFD** is the abbreviation for **Data Flow Diagram**. The flow of data of a system or a process is represented by DFD. It also gives insight into the inputs and outputs of each entity and the process itself. DFD does not have control flow and no loops or decision rules are present.

Specific operations depending on the type of data can be explained by a flowchart. It is a

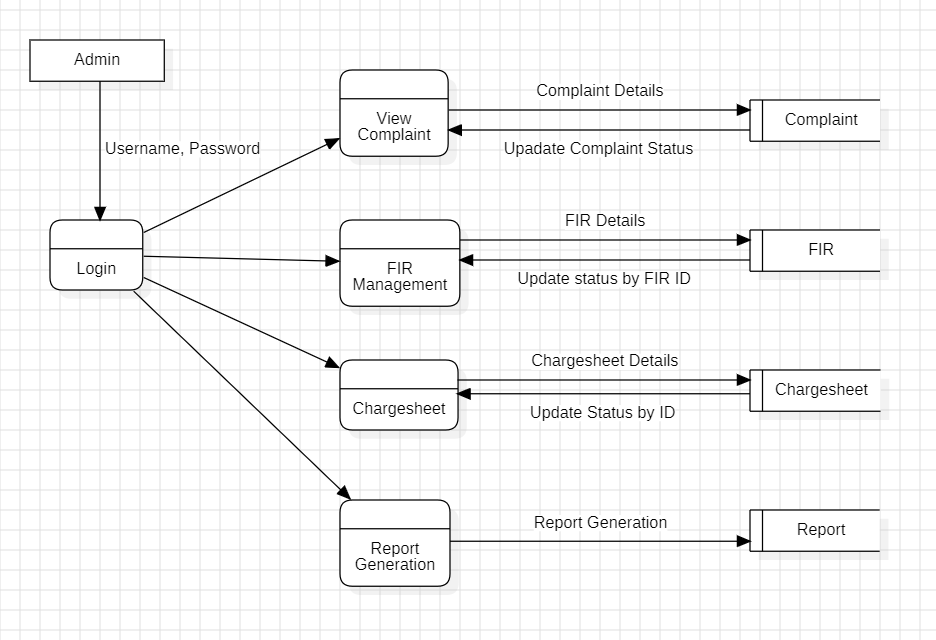
graphical tool, useful for communicating with users ,managers and other personnel. it is useful for analyzing existing as well as proposed system.

#### Table 3.10: Symbols of DFD

|  |  |  |  |
| --- | --- | --- | --- |
| Sr.No. | Name | Symbol | Description |
| 1. | Process |  | Process transformation incoming data flow into outgoing ata flow. |
| 2. | Data Flow |  | Data flow are pipelines through which packets of information |
| 3. | Data Store |  | Data sores are repositories of data in system. |
| 4. | External Entity |  | External entities are objects outside system, with which the system communicate. |



*Fig 3.3 Level 1-DFD*

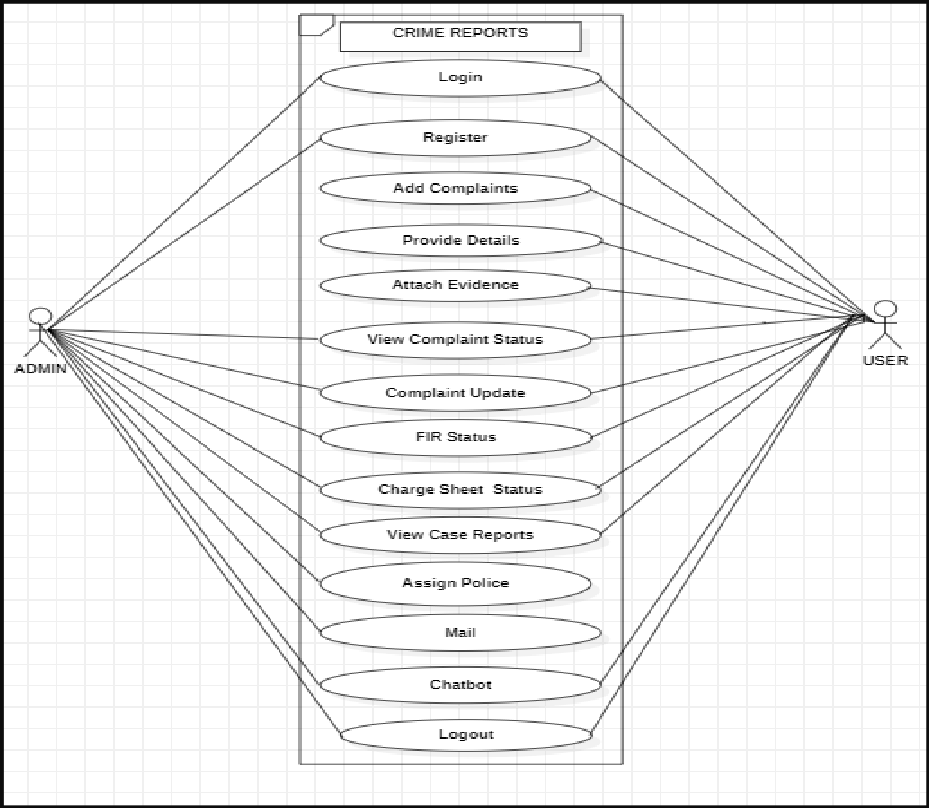


*Fig 3.4 Level 2-DFD*

1. Use Case Diagram

A use case diagram illustrates a system's dynamic behavior, showcasing its functionality through use cases, actors, and their relationships. It represents the tasks, services, and functions needed by a system or subsystem, providing a high-level overview of its functionality and user interactions. **Table 3.11: Symbols of Use Case**

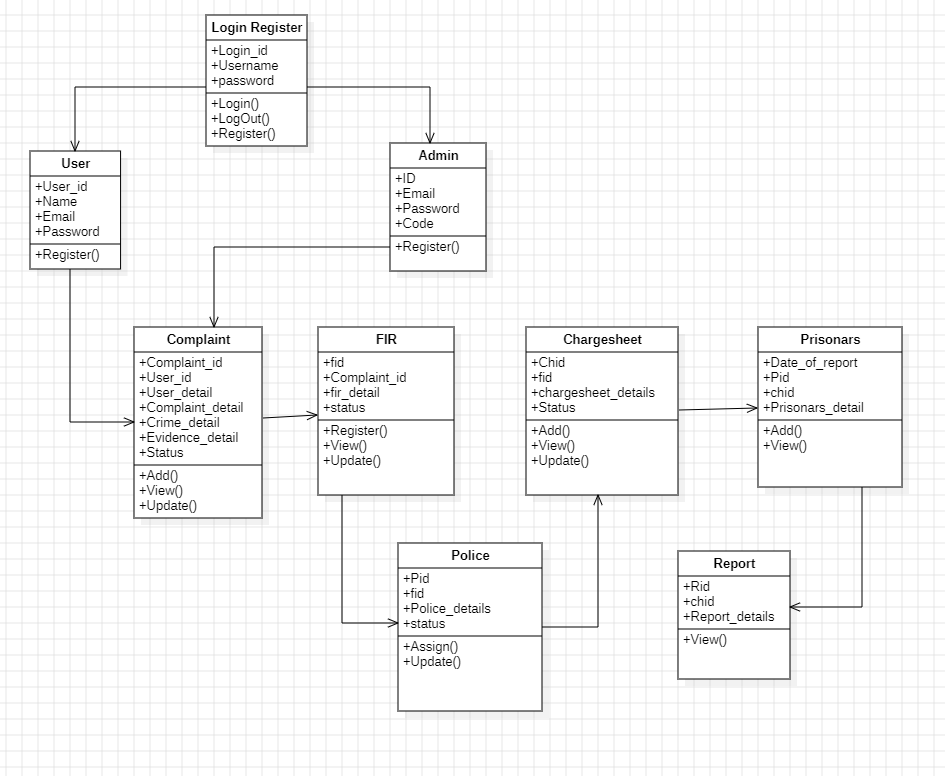
|  |  |  |  |
| --- | --- | --- | --- |
| Sr.No | Name | Symbol | Description |
| 1. | Actor |  | Actor represents user or another system that will interact with the system you are modelling. |
| 2. | Use Case |  | A user case is an external view of the system that represents some action the user might perform in order to complete a task. |



*Fig 3.5 Use Case Diagram*

6] Class Diagram:

Class diagrams are a type of UML(Unified Modeling Language) diagram used in software engineering to visually represent the structure and relationships of classes in a system. UML is a standardized modeling language that helps in designing and documenting software systems. They are an integral part of the software development process, helping in both the design and documentation phases.



*Fig 3.6 Class Diagram*

# Chapter 4 : Implementation and Testing

## Code

### regcomplaint.jsp

<script>

$(document).ready(function() {

// Function to validate phone numbers

function validatePhoneNumber(inputId) {

var phoneNumber = $("#" + inputId).val();

if (phoneNumber.length !== 10) {

alert("Phone number must be 10 digits long.");

return false;

}

return true;

}

// Event listener for form submission

$("form").on("submit", function(event) {

// Validate phone numbers before submitting the form

if (!validatePhoneNumber("n3") || !validatePhoneNumber("n13") || !validatePhoneNumber("n16")) {

event.preventDefault(); // Prevent form submission

}

});

});

</script>

<script>

function validatePhoneNumber(phoneNo, inputId) {

if (phoneNo.length !== 10) {

alert("Phone number must be 10 digits long.");

document.getElementById(inputId).focus();

return false;

}

return true;

}

function validateForm() {

var phoneNo = document.getElementById("n3").value; // Replace "n3" with the actual input id for phone\_no

var accusedPhoneNo = document.getElementById("n13").value; // Replace "n13" with the actual input id for accused\_phone\_no

var victimPhoneNo = document.getElementById("n16").value; // Replace "n16" with the actual input id for victim\_phone\_no

if (!validatePhoneNumber(phoneNo, "n3") || !validatePhoneNumber(accusedPhoneNo, "n13") || !validatePhoneNumber(victimPhoneNo, "n16")) {

return false;

}

// Additional form validation logic can be added here

return true;

}

</script>

<main class="p-15" style="padding-bottom: 50px;">

<div class="container">

<div class="row justify-content-end">

<div class="col-md-11 mt-4 ml-auto">

<div class="card mx-auto">

<div class="card-header text-center text-black">

<h3>

Register Complaint here

</h3>

</div>

<div class="card-body">

<form action="complaint" method="POST" onsubmit="return validateForm();" enctype="multipart/form-data" required>

<div class="form-row">

<input type="hidden" name="id">

<!--<input type="hidden" name="uid">-->

<div class="form-group col-md-4">

<input type="text" name="name" class="form-control" id="n1" placeholder="Your Name" required>

</div>

<div class="form-group col-md-4">

<input type="email" name="email" class="form-control" id="n2" placeholder="Your Email" required>

</div>

<div class="form-group col-md-4">

<input type="text" name="phone\_no" class="form-control" id="n3" placeholder="Your Phone No" required>

</div>

</div>

<div class="form-row">

<div class="form-group col-md-6">

<input type="text" name="pcity" class="form-control" id="n4" placeholder="Police\_Station city" required>

<input type="text" name="pstate" class="form-control" id="n5" placeholder="Police\_Station state" required>

<!--<textarea class="form-control" name="police\_station" id="exampleFormControlTextarea1" rows="3" placeholder="Police\_Station State <% out.println(); %>Police\_Station City"></textarea>-->

</div>

<div class="form-group col-md-6">

<textarea class="form-control" name="complaint\_detail" id="n6" rows="3" placeholder="Complaint Title <% out.println(); %>Complaint Detail" required></textarea>

</div>

</div>

<div class="form-row">

<div class="form-group col-md-3">

<select id="n7" name="area" class="form-control" placeholder="Area" required>

<option selected>Incident Area</option>

<option>Bandra</option>

<option>Dadar</option>

<option>Powai</option>

<option>Andheri</option>

<option>Malad</option>

<option>Vasant Kunj</option>

<option>Dwarka</option>

<option>Chanakyapuri</option>

<option>Saket</option>

<option>Pitampura</option>

</select>

</div>

<div class="form-group col-md-3">

<select id="n8" name="icity" class="form-control" placeholder="City" required>

<option selected>Incident City</option>

<option>Mumbai</option>

<option>Delhi</option>

</select>

</div>

<div class="form-group col-md-3">

<select class="form-control" id="n9" name="tcrime" required>

<option>Type Of Crime</option>

<option>Theft</option>

<option>Burglary</option>

<option>Assault</option>

<option>Robbery</option>

<option>Fraud</option>

<option>Cyber crime</option>

<option>Homicide</option>

<option>Drug Offense</option>

</select>

</div>

<div class="form-group col-md-3">

<input type="date" class="form-control" name="date" id="n10" step="1" required>

</div>

</div>

<div class="form-row">

<div class="form-group col-md-6">

<input type="text" name="aname" class="form-control" id="n11" placeholder="Accused Name" required>

<input type="text" name="aadd" class="form-control" id="n12" placeholder="Accused Address" required>

<input type="text" name="aphone" class="form-control" id="n13" placeholder="Accused Phone\_no" required>

<!--<textarea class="form-control" name="accused\_detail" id="exampleFormControlTextarea1" rows="3" placeholder="Accused Name <% out.println(); %>Victim Address <% out.println(); %>Accused Ph\_no"></textarea>-->

</div>

<div class="form-group col-md-6">

<input type="text" name="vname" class="form-control" id="n14" placeholder="Victim Name" required>

<input type="text" name="vadd" class="form-control" id="n15" placeholder="Victim Address" required>

<input type="text" name="vphone" class="form-control" id="n16" placeholder="Victim Phone\_no" required>

<!--<textarea class="form-control" name="victim\_detail" id="exampleFormControlTextarea1" rows="3" placeholder="Victim Name <% out.println(); %>Victim Address <% out.println(); %>Victim Ph\_no"></textarea>-->

</div>

</div>

<div class="form-row">

<div class="form-group col-md-12">

<textarea class="form-control" name="evidence\_detail" id="n17" rows="2" placeholder="Any Evidence" required></textarea>

</div>

</div>

<div class="form-row">

<div class="form-group col-md-6">

<input type="file" name="files" id="n18"><br>

<small>Choose image as well as video file.</small>

</div>

</div>

<div>

<select id="inputState" class="form-control" name="status">

<option>---</option>

<option>Accepted</option>

<option>Rejected</option>

</select>

</div>

<button type="submit" class="btn btn-primary col-md-3 mx-auto d-block">Register</button>

</form>

</div>

</div>

</div>

</div>

</div>

</main>

#### Viewreport.jsp

<%@page import="com.entities.FIR"%>

<%@page import="com.dao.FIRDao"%>

<%@page import="com.entities.Complaint"%>

<%@page import="com.dao.ComplaintDao"%>

<%@page import="com.entities.chargesheet"%>

<%@page import="com.dao.chargesheetDao"%>

<%@page import="com.entities.report"%>

<%@page import="java.util.List"%>

<%@page import="com.conn.DbConnect"%>

<%@page import="com.dao.reportDao"%>

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>JSP Page</title>

<%@include file="allcomponet/allCss.jsp" %>

</head>

<body>

<%@include file="navbar.jsp"%>

<center><h1><b>CASE REPORT</b></h1></center>

<center>

<div class="container-fluid">

<div class="row p-2">

<div class="col-md-5 offset-md-4">

<div class="card" style="box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);">

<div class="card-body">

<div class="row">

<div class="center-table">

<table border='1'>

<%

int id = Integer.parseInt(request.getParameter("reportId")); System.out.println(id);

reportDao dao = new reportDao(DbConnect.getConn()); List<report> list = dao.isReportReg1(id);

%>

<%

for(report r: list) {

int chid = r.getChid();

chargesheetDao dao1 = new chargesheetDao(DbConnect.getConn()); chargesheet ch = dao1.getchargesheetById(chid);

int fid = ch.getFid();

FIRDao dao2 = new FIRDao(DbConnect.getConn()); FIR f = dao2.getFIRById(fid);

int cid = f.getId();

ComplaintDao dao3 = new ComplaintDao(DbConnect.getConn()); Complaint c = dao3.getComplaintById(cid);

%>

<!--report-->

<h3>CASE DETAIL</h3>

Case ID: <%= r.getRId() %> <br>

Title: <%= c.getComplaint\_detail() %><br> Date: <%= r.getCdate() %><br> Description: <%= r.getCdetail() %><br>

<hr>

<div class="form-row">

<div class="form-group col-md-6">

<b>ACCUSED DETAIL:</b><br>

<%= c.getAccused\_name() %><br><%= c.getAccused\_add() %><br><%= c.getAccused\_phno() %>

</div>

<div class="form-group col-md-6">

<b>VICTIM DETAIL:</b><br>

<%= c.getVictim\_name() %><br><%= c.getVictim\_add() %><br><%= c.getVictim\_phno() %>

</div>

</div><hr>

<!--fir-->

<div class="form-row">

<div class="form-group col-md-6">

<b>FIR DETAILS</b> <br>

FIR No.: <%= f.getFid() %> <br>

Date: <%= f.getFir\_start\_date() %> <%= "-" %> <%= f.getFir\_end\_date() %><br> Section: <%= f.getSection() %><br>

Description: <%= f.getFir\_detail() %><br>

</div>

<!--complaint-->

<div class="form-group col-md-6">

<b>COMPLAINT DETAIL</b> <br>

Complaint No. : <%= c.getId() %><br> Complaint Date: <%= c.getDate() %><br> Proof:

</div>

</div>

<hr>

<div class="form-row">

<div class="form-group col-md-6">

<!--chargesheet-->

<b>CHARGEHSEET DETAIL:</b> <br>

Chargesheet No: <%= ch.getId() %><br> Chargesheet Date: <%= ch.getDate() %><br>

Chargesheet Description: <%= ch.getBriefdesc() %><br>

</div>

<div class="form-group col-md-6">

<b>PRISONARS DETAIL:</b> <br>

Prisonor Name: <%= r.getName() %> <br> Section: <%= r.getSection() %> <br> Prisonars Address: <%= r.getAdd() %> <br>

</div>

</div>

<% } %>

</table>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</center>

</body>

</html>

**NewsServlet.java**

@WebServlet("/news")

public class NewsServlet extends HttpServlet {

public void doPost(HttpServletRequest request, HttpServletResponse response) throws IOException

{

PrintWriter out = response.getWriter();

String title = request.getParameter("title");

String date\_time = request.getParameter("date\_time");

String content = request.getParameter("content");

News n = new News(title, date\_time, content);

NewsDAO dao=new NewsDAO(DbConnect.getConn());

//boolean f=dao.addNews(n);

boolean f = dao.updateNews(n);

HttpSession session=request.getSession();

if(f){

out.println("<html><head><script type='text/javascript'>");

out.println("alert('News Updated successful;ly..!!');");

out.println("window.location.href='admin.jsp'");

out.println("</script></head></html>");

}else{

out.println("<html><head><script type='text/javascript'>");

out.println("alert('Something went wrong!');");

out.println("window.location.href='admin.jsp'");

out.println("</script></head></html>");

}

}

}

## Testing Approach Test Design

Software testing is an essential and important technique for assessing the quality of a particular

software product/service. In software testing, test cases and scenarios play an inevitable and a pivotal role. A good strategic design and technique help to improve the quality of the software testing process. The process improves the quality of the product/service and ensures effectiveness. Software testing is the process of analysing a software item to know the differences between the existing and required conditions (bugs). Testing helps to evaluate the features of the software, to ensure it is free of bug. It is an activity that is carried out in co ordinance with the development cycle and before the deployment.

Test case design techniques are crucial verifications steps that are created to design a software or application that is free from various kinds of defects and issues. The purpose of these techniques is to test the functionalities and features of the software with the assistance of some effective testcases.

#### The two categories of test design techniques are:

* + 1. Unit Testing.
    2. Integration Testing.
    3. **Unit Testing**

Unit testing in the context of an online crime management system, or any software application, is essential to ensure the system works correctly, is reliable, and can handle various scenarios efficiently.

Identify Units: Break down the codebase of your crime management system into smaller, testable units. These units might include functions, methods, or classes responsible for specific functionalities (e.g., user authentication, case creation, evidence management, etc.).

Isolate Units: Unit tests should focus on testing individual units in isolation. This means you should isolate the unit being tested from the rest of the system, using techniques like mocking or stubbing for external dependencies, such as databases or external APIs.

Test Cases: Write test cases for each unit that cover a wide range of scenarios, including normal use cases, edge cases, and error conditions. In the context of a crime management system, consider scenarios like creating cases with different types of crimes, updating case statuses, or handling large amounts of evidence data.

Test Automation: Automate your unit tests so that they can be run regularly as part of your continuous integration (CI) process. This helps ensure that any changes or new code additions do not break existing functionality.

Assertions: Use assertions in your unit tests to verify that the unit under test produces the

expected outputs for given inputs. This is crucial for detecting regressions in your code.

Code Coverage: Measure code coverage to ensure that your tests are exercising a significant

Portion of your codebase. This helps identify areas of your code that may not be adequately

tested.

Edge Cases: Pay special attention to testing edge cases and boundary conditions. In a crime management system, this might include testing the system's behavior when handling extreme data inputs or unexpected user interactions.

Performance and Scalability: Depending on the scale of your system, consider adding performance and scalability tests to ensure that it can handle the expected load efficiently. For a crime management system, this might involve testing the system's ability to handle a large number of concurrent cases or users.

Security Testing: Unit tests can also include security testing, such as checking for vulnerabilities like SQL injection or authentication issues, especially if your system deals with sensitive crime- related data.

Test Documentation: Document your unit tests clearly, including the purpose of each test, the expected results, and any dependencies or setup required.

Regression Testing: Whenever a bug is discovered or a change is made, add a unit test that reproduces the issue before fixing it. This becomes part of your regression testing suite, ensuring that the issue doesn't reappear in the future.

Continuous Improvement: Regularly review and update your unit tests as the codebase evolves. Ensure that your tests remain aligned with the current system functionality.

**Test Plans and Test Cases:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No | Test Name | Expected Output | | | Actual Output | Remark |
| 1. | Register(put correct details) | Account created successfully | | | Registered | Passed |
| 2. | Login: Enter Valid Details | Login Successful | | | Login Success | Passed |
| 3. | Login: Enter Invalid Details | Invalid credentials | | | Invalid | Passed |
| 4. | Add Complaint | Entered valid  details of the complaint. | | | Complaint added | Passed |
| 5. | View Complaint | | Viewed all details | Successfully displaying details. | | Passed | |
| 6. | Registered FIR | | Enter the correct details of FIR. | FIR registered successfully. | | Passed | |
| 7. | View FIR | | Viewed all details of FIR. | displaying details. | | Passed | |
| 8. | Generate Chargesheet | | Add all valid details of the chargesheet. | Successfully added details. | | Passed | |
| 9. | View chargesheet details. | | Viewed all the details. | Succesfully displaying details. | | Passed | |
| 10. | View Report | | Viewed all case details. | displaying all details in one report. | | Passed | |
| 11. | Edit Complaint status | | Update status | Updated | | Passed | |
| 12. | Edit FIR status | | Update status | Updated | | Passed | |
| 13. | Edit chargesheet status | | Update status | Updated | | Passed | |
| 14. | User see all case statuses. | | Display all updated status | Dislaying | | Passed | |

### Integration Testing

Integration testing in an online crime management system is essential to ensure that different components or modules of the system work together seamlessly and to detect any issues that may arise when these components are integrated.

Define Integration Points: Identify the key integration points in your crime management system. These are the interfaces or interactions between different components, modules, or services.

Integration Scenarios: Create integration test scenarios that represent real-world use cases. Consider scenarios such as creating a case, assigning investigators, updating case status, managing evidence, and generating reports.

Test Data: Prepare a set of test data that simulates various situations, including valid and invalid inputs, boundary conditions, and edge cases. This data should cover different types of crimes, user roles, and system states.

Testing Environments: Set up separate testing environments, if possible, that mimic the

Production environments.

Positive and Negative Testing: Test positive scenarios to ensure that components work as expected when they cooperate. Also, test negative scenarios to check how the system handles errors or unexpected inputs.

Data Flow Testing: Verify that data flows correctly between different modules, and check for data consistency and accuracy as it moves through the system.

Interface Testing: Pay close attention to the interfaces between modules. Verify that data is correctly passed between them and that the APIs or protocols used for communication are reliable and secure.

Concurrency and Multi-User Testing: Simulate concurrent users and transactions to assess how the system handles multiple requests and data consistency in a multi-user environment.

Performance and Load Testing: Evaluate the system's performance under different loads to ensure that it can handle the expected number of cases, users, and data efficiently.

Security Testing: Include security checks in your integration testing to verify that sensitive data is protected, and that the system is resistant to common security threats, such as SQL injection or unauthorized access.

Error Handling and Logging: verify that appropriate error messages are generated and logged.

# Chapter 5 : Result and Discussion

index.jsp

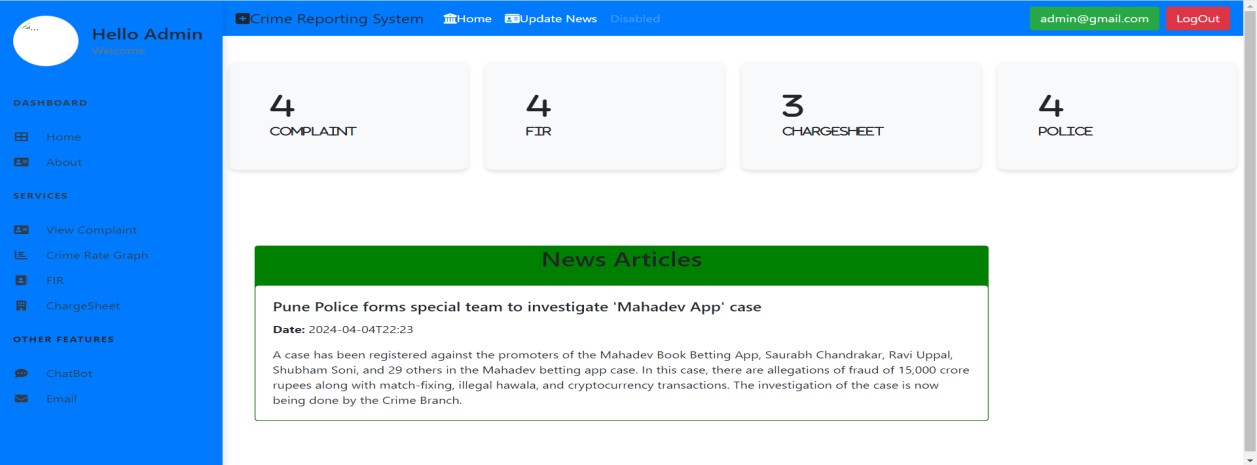


*Fig 5.1 Home Page*

The index page serves as the initial view for users and includes a navbar with buttons for Home, About us, Login, and Sign up. Users can choose to sign up as either a user or admin, or log in.

Additionally, the page features an early news section and an About us section providing website details. A separate login page is available for both admins and users.

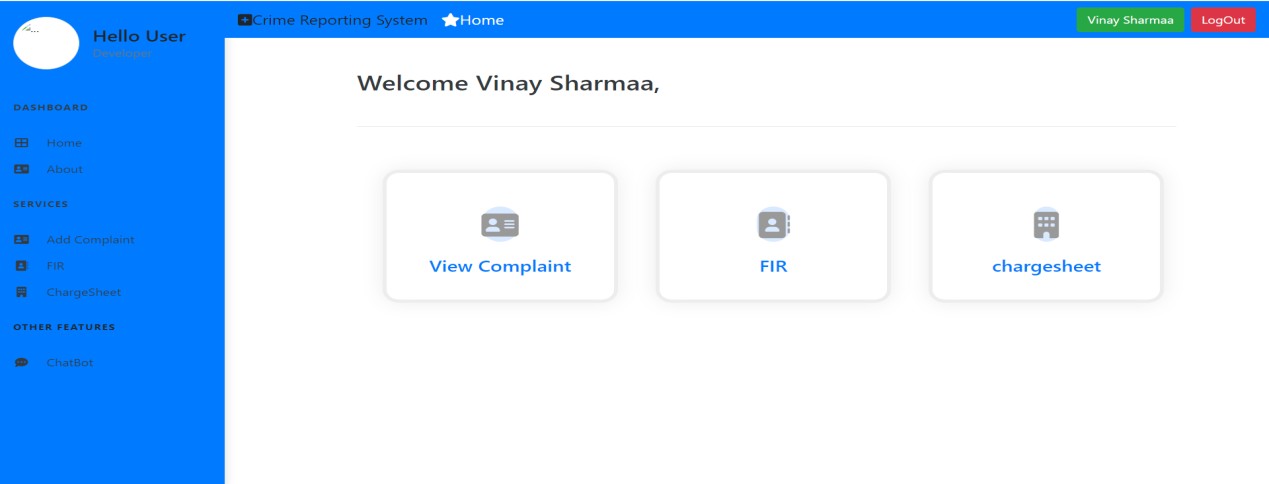
admin.jsp



*Fig 5.2 Admin Dashboard*

The admin page presents a dashboard with a left-sided vertical navbar for navigation to sections like complaints, FIR, and chargesheets. It also includes an Update News button in the top navbar and a news section at the bottom, along with four cards offering analysis on complaint, FIR, chargesheet, and police data count, improving data accessibility and management.

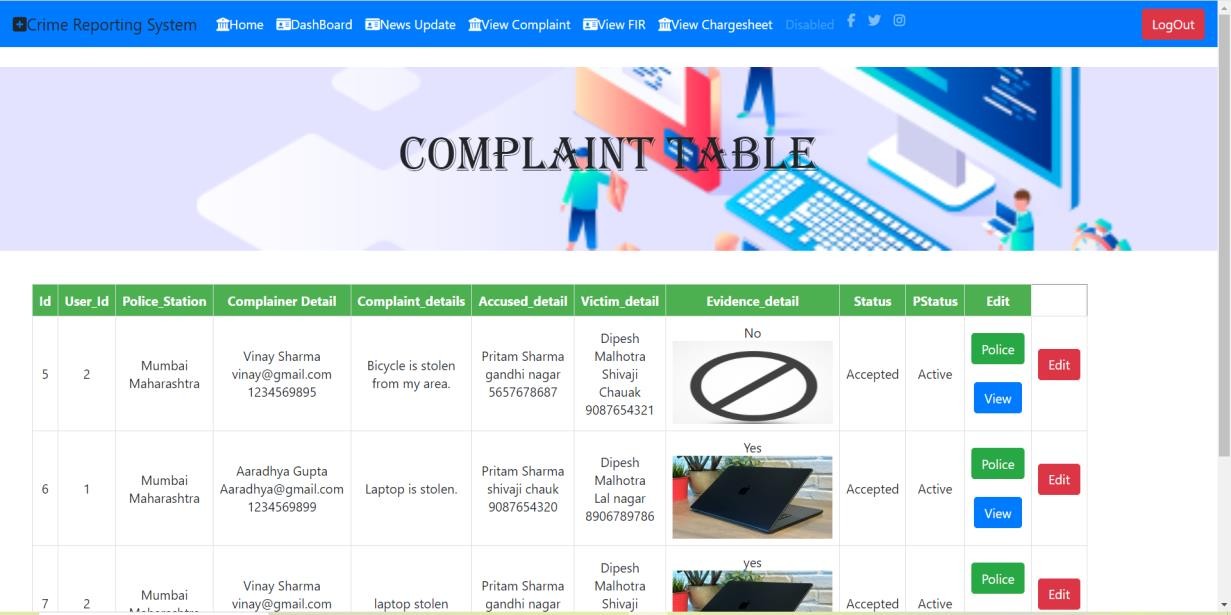
profile.jsp



*Fig 5.11 User Dashboard*

The user dashboard has a vertical navbar with key links and three prominent cards for quick access to complaints, FIR, and chargesheets, providing an efficient and user-friendly interface for incident tracking.

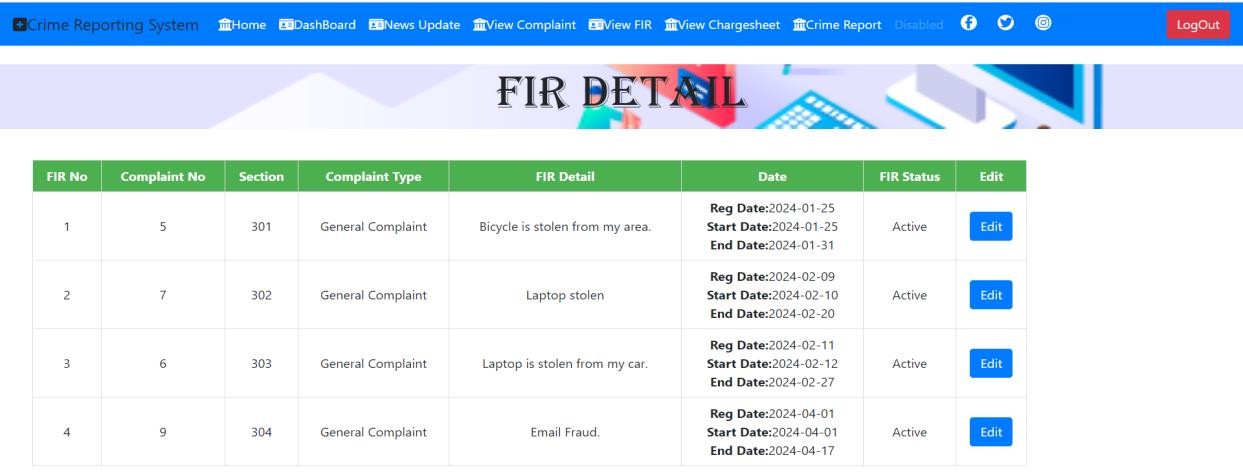
displayadmin.java



*Fig 5.3 UI for Display Complaint Table*

The Display Admin page presents user-added complaints in a table format, with buttons for Police, View, and Edit. Clicking "View" opens a Bootstrap modal displaying complaint details. Admins update complaint status with the "Edit" button and assign police using the "Add" button, which then converts to a "Police" button for status updates. A navbar at the top provides navigation to other pages.

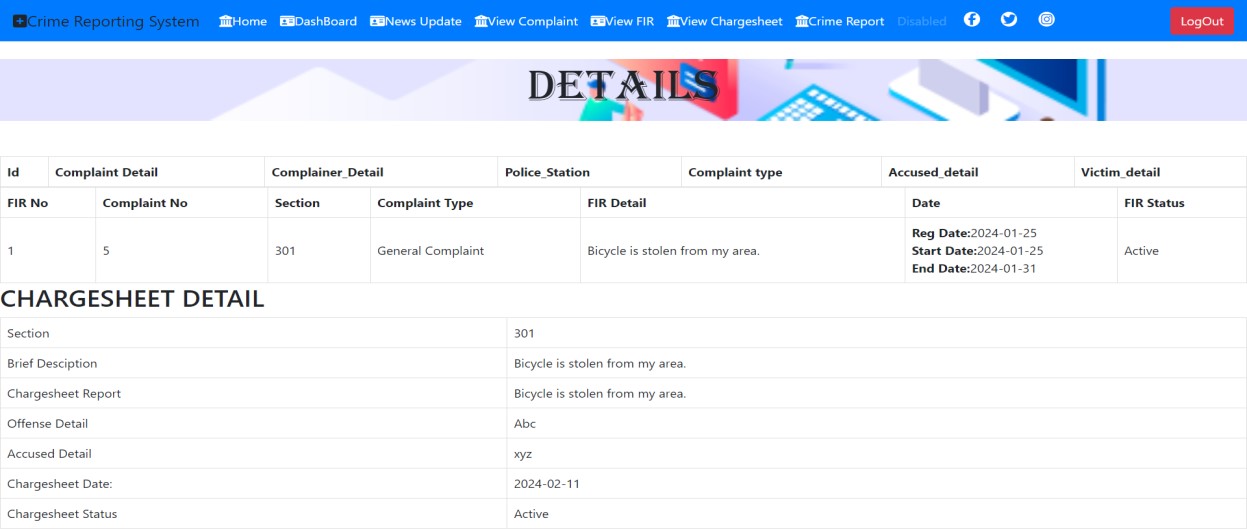
displayFir.jsp



*Fig 5.5 User Interface for FIR Detail*

The displayFIR page displays all the details of FIRs added at the time of FIR registration. In the table, an edit button is present, allowing admins to update the status of the FIR. Admins can update the status as Under Process, Active, or Inactive, which helps them track the case details easily.

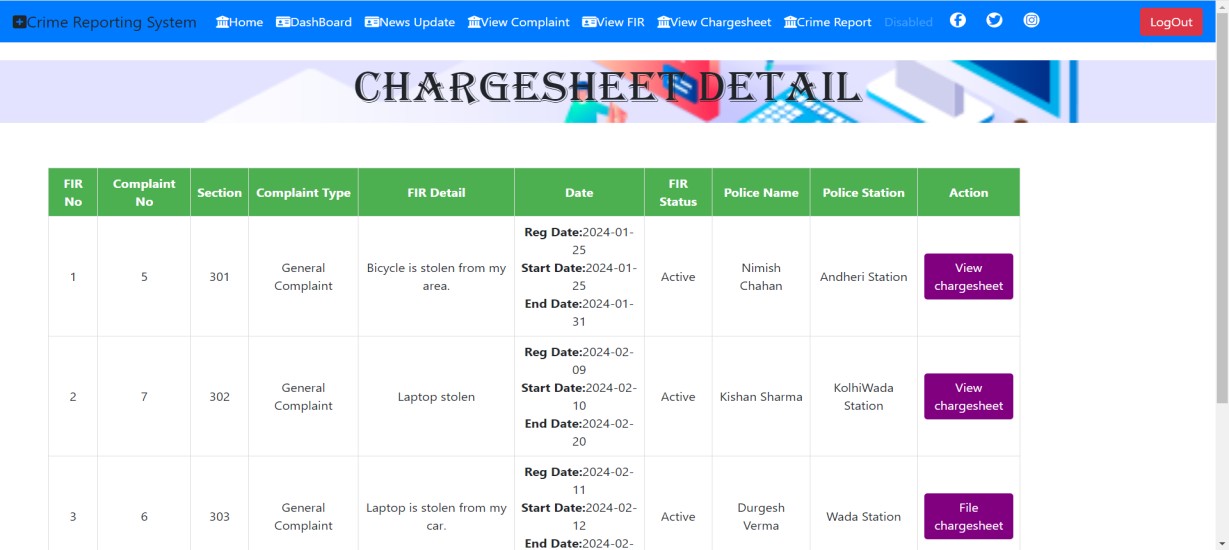
Displaycharge.jsp



*Fig 5.7 Chargesheet details*

The displaychargesheet page includes the chargesheet data of a particular case. At the top of the webpage, it shows the navbar, which contains links to other pages.

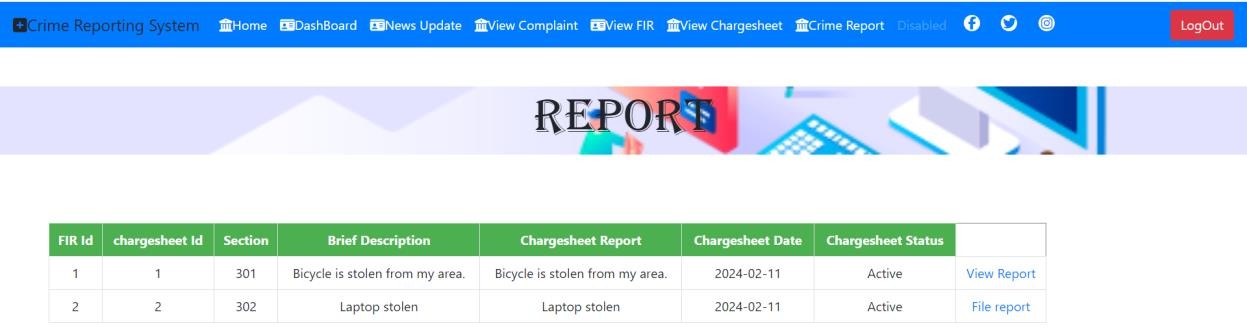
displayAccptedFir.jsp



*Fig 5.6 User Interface for chargesheet Detail*

This page will display the chargesheet details. Only if the FIR status is completed, the admin can generate the chargesheet. The chargesheet details are displayed in table form, which also includes the police details. The table also contains the "View Chargesheet" button, which redirects to the chargesheet page, showing the chargesheet of the particular case.

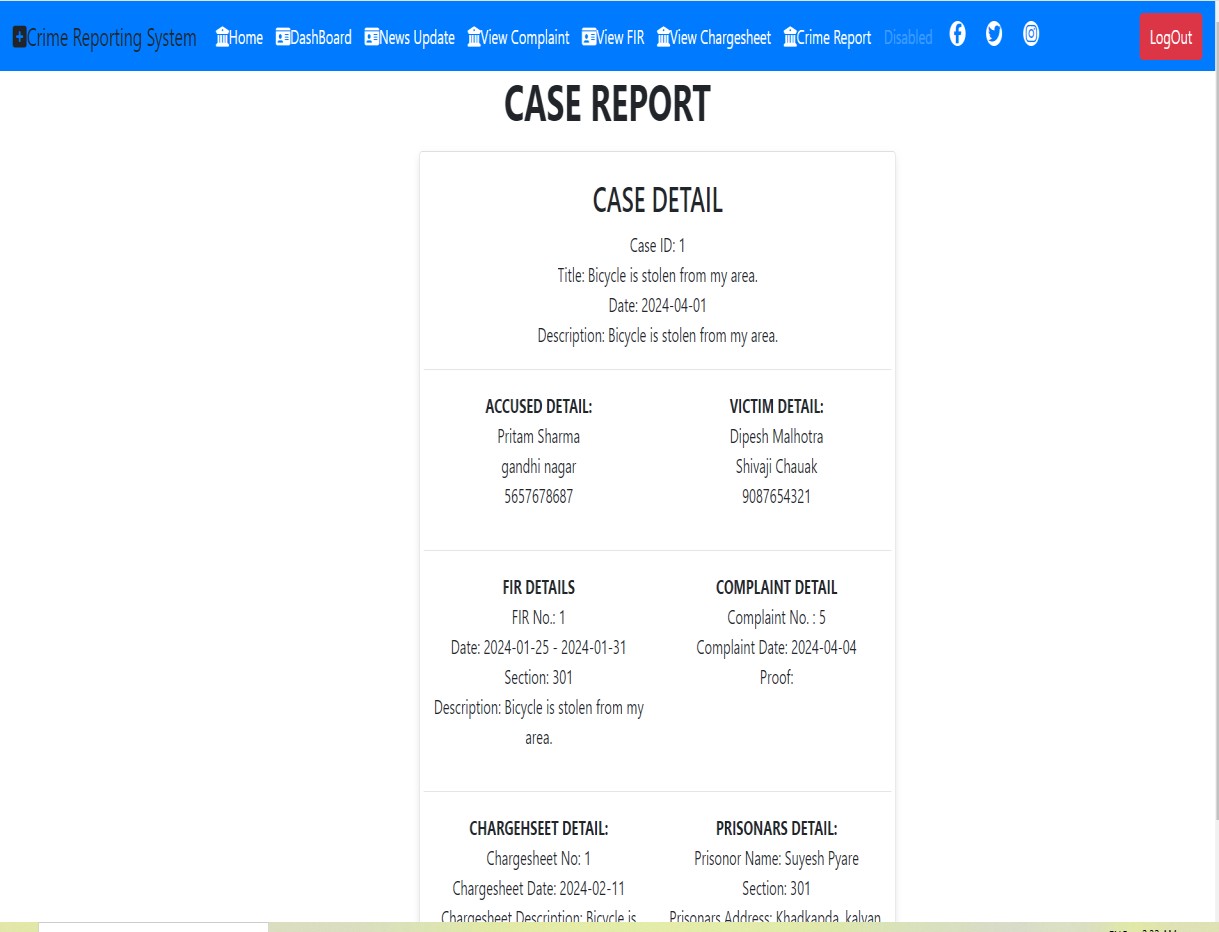
displayacceptedcharge.jsp



*Fig 5.8 User Interface for Report*

The Display Accepted Charges page presents active chargesheets in a table format, enabling admins to file or view reports with ease. Admins can register reports via "File Report" and access crime reports through "View Report," streamlining administrative tasks.

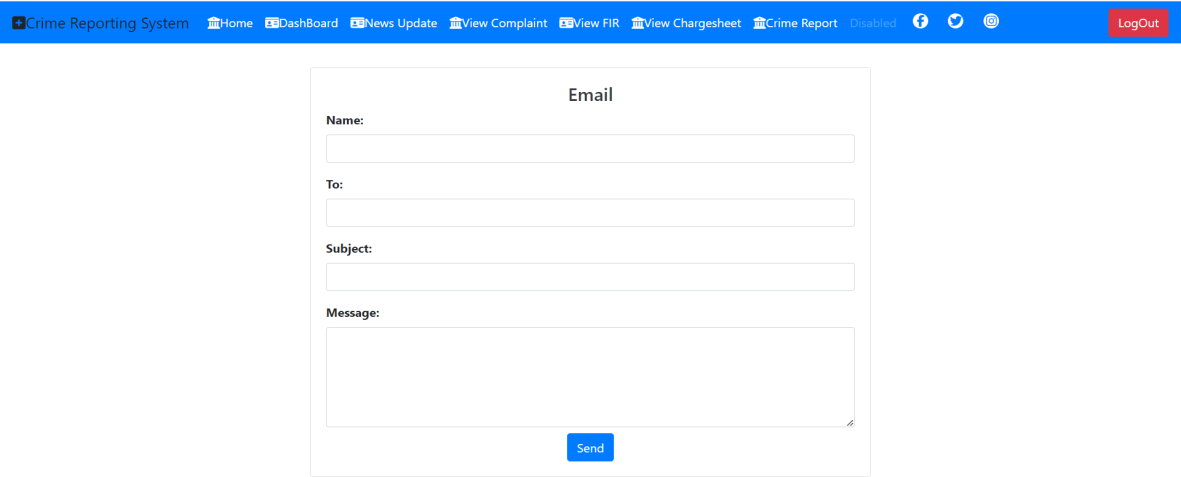
Viewreport.jsp



*Fig 5.10 Case Report*

The "View Report" page provides a central interface presenting comprehensive case details, including report dates, prisoner information, and associated evidence. Offering a holistic perspective, it streamlines case review by presenting all pertinent information in a single, easily navigable location. Its intuitive design enhances user experience, facilitating efficient retrieval and analysis of case data for administrators and authorized users.

email.jsp



*Fig 5.12 Email Form*

Efficiently communicate updates to users via SMTP-enabled email form, enhancing case management transparency and user engagement. Facilitate timely and clear correspondence between admins and users, improving incident reporting and resolution processes.

Chatbot.jsp



*Fig 5.13 User Interface for Chatbot*

The chatbot interface allows admins and users to input crime-related queries, providing relevant information and assistance in a user-friendly manner. Enhance communication and streamline information exchange between users and administrators, fostering a seamless user experience for accessing crime data and website-related inquiries.

# Chapter 6 : Conclusion and Future Work

* 1. **Conclusion**:

The proposed online crime reporting system represents a significant leap forward in law enforcement technology. It offers tailored modules for users and administrators, ensuring accurate and reliable reporting through secure authentication mechanisms. Administrators can efficiently manage cases, from reviewing complaints to allocating resources, thanks to robust functionalities. By consolidating data into a unified format, the system enhances case management efficiency, provides valuable insights for decision-making, and improves crime prevention efforts. Its user-friendly interface, strong administrative features, and data integration capabilities make it a crucial asset for modern law enforcement, albeit requiring continuous refinement to adapt to evolving needs and technologies

.**Limitations**:

* + - Chatbot functionality may not fully replicate real-life conversations.
    - Complexity of crime reporting process may pose challenges.
    - Resource constraints, including budget and technical expertise, may impact development.
    - Regulatory compliance with data protection and privacy laws is essential but challenging.
  1. **Future Work:**
     + Enhancing Chatbot Capabilities: Future iterations could focus on improving the chatbot's functionality by enhancing natural language processing algorithms, generating more intelligent responses, and personalizing interactions.
     + Strengthening User Authentication and Security Measures: Prioritizing the implementation of robust user authentication and security measures to ensure the confidentiality and integrity of user data.
     + Implementing Data Analytics: Integrating data analytics for crime trend analysis to identify patterns and insights that can aid in crime prevention efforts.
     + Developing a Mobile Application: Creating a mobile application for convenient reporting, allowing users to submit complaints and access system functionalities on the go.

# Chapter 7 : References

1. Ku, C.H.; Iriberri, A.; Leroy, G. Crime Information Extraction from Police and Witness Narrative Reports. In Proceedings of the 2008 IEEE International Conference on Technologies for Homeland Security, Westin Hotel, Waltham, MA, USA, 12–13 May 2008; pp. 12–13. [[**Google Scholar**](https://scholar.google.com/scholar_lookup?title=Crime%2BInformation%2BExtraction%2Bfrom%2BPolice%2Band%2BWitness%2BNarrative%2BReports&conference=Proceedings%2Bof%2Bthe%2B2008%2BIEEE%2BInternational%2BConference%2Bon%2BTechnologies%2Bfor%2BHomeland%2BSecurity%2C%2BWestin%2BHotel&author=Ku%2C%2BC.H.&author=Iriberri%2C%2BA.&author=Leroy%2C%2BG.&publication_year=2008&pages=12%E2%80%9313)]
2. Iriberri, A.; Leroy, G. Natural Language Processing and e-Government: Extracting Reusable Crime Report Information. In Proceedings of the IEEE International Conference on Information Reuse and Integration, Las Vegas, NV, USA, 13–15 August 2007; pp. 221–226. [[**Google Scholar**](https://scholar.google.com/scholar_lookup?title=Natural%2BLanguage%2BProcessing%2Band%2Be-Government%3A%2BExtracting%2BReusable%2BCrime%2BReport%2BInformation&conference=Proceedings%2Bof%2Bthe%2BIEEE%2BInternational%2BConference%2Bon%2BInformation%2BReuse%2Band%2BIntegration&author=Iriberri%2C%2BA.&author=Leroy%2C%2BG.&publication_year=2007&pages=221%E2%80%93226)]
3. Subrata Chowdhury, M. Sreekanth, L. Yeshwanth, "Online Crime File Management System", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 8 Issue 6, pp. 34-41, November-December 2022. Journal URL : <https://ijsrcseit.com/CSEIT228562>