# ABSTRACT

The Auto FIR Registration System is an innovative software application designed to facilitate the efficient filing of First Information Reports (FIRs) for a range of incidents, including vehicle-related cases, theft, assault, murder, and suicide. This application addresses the need for a streamlined process that reduces bureaucratic delays and improves the responsiveness of law enforcement agencies. Developed using PyQt5, the user-friendly interface allows authorized personnel to register FIRs with minimal effort, thereby enhancing the accessibility of critical reporting mechanisms for serious incidents.

The system employs an SQLite database for secure data management, ensuring that all reports are stored and retrieved reliably. Key features include a secure login system, automatic email notifications to relevant stakeholders upon the registration of new FIRs or significant updates, and a multi-page navigation system for seamless access to different functionalities. The application also incorporates robust reporting capabilities, allowing users to view historical records and analyze data on incidents such as murder and suicide. Additionally, the application provides insights into financial statistics related to police operations through interactive charts, aiding decision-making and resource allocation.

Ultimately, the Auto FIR Registration System seeks to enhance the efficacy of law enforcement efforts, promote transparency, and foster community trust by providing a reliable and efficient platform for reporting serious incidents.

# ACKNOWLEDGEMENT

On this page, we express our sincere gratitude and appreciation to those individuals who have played a pivotal role in shaping the course of this project. Their unwavering guidance and assistance have been instrumental in ensuring the smooth and efficient execution of this dissertation.

We would like to extend our heartfelt thanks to **Ms Shruthi Patil, Assistant Professor**, Department of Intelligent Computing & Business System, for her invaluable guidance and insightful suggestions. Her contributions have greatly contributed to the successful completion of this project.

Our profound gratitude goes to **Dr Shreenath Acharya, Head of the Department,** Intelligent Computing & Business System, for providing us with his kind consent and expert guidance, which have been instrumental in the successful culmination of this endeavour.

We also wish to express our deep appreciation to our esteemed **Director, Rev. Fr. Wilfred Prakash D'Souza,** our respected **Principal, Dr Rio D'Souza**, and our **Assistant Director, Rev. Fr. Kenneth Crasta,** for their unwavering support and encouragement throughout this journey.

Our heartfelt thanks are extended to all the dedicated faculty and staff members of ICBS who have consistently stood by our side, offering their support, invaluable suggestions, guidance, and unwavering encouragement at every juncture.

Additionally, we would like to convey our gratitude to our friends and family members for their continuous and unwavering support, which has been a source of strength and motivation throughout this project.

# TABLE OF CONTENTS

**Abstract** iii

**Acknowledgement** iv

**Table of contents** v

**Chapter 1: Introduction** ..... **1**

1.1 Problem definition ..... 2

1.2 Methodology ..... 2

1.3 Application Modules Overview ..... 3

1.4 Application Functionalities ..... 4

**Chapter 2: Requirement Analysis** ..... **6**

2.1 System Requirements ..... 6

2.1.1 Functional Requirements ..... 6

2.1.2 Non Functional Requirements ..... 7

2.1.3 Software Requirements ..... 8

2.1.4 Hardware Requirements ..... 10

**Chapter 3: System Design** ..... **11**

3.1 ER Diagram ..... 11

3.2 User Flow Diagram ..... 11

**Chapter 4: Implementation** ..... **12**

**Chapter 5: Screen Outputs** ..... **14**

**Chapter 6: Conclusion** ..... **17`**

**Chapter 7: Future Enhancement** ..... **18**

**References** ..... **19**

# CHAPTER – 1

# INTRODUCTION

The Auto FIR Registration Application is a cutting-edge solution designed to revolutionize the way First Information Reports (FIRs) are filed with law enforcement agencies. In today's rapidly evolving world, where quick response times and efficient communication are paramount, this application addresses the longstanding challenges associated with traditional FIR filing processes.

This application empowers users to report incidents swiftly and conveniently, eliminating the need for cumbersome paperwork and reducing the time spent waiting for assistance at police stations. By leveraging technology, the Auto FIR Registration Application enhances accessibility, allowing authorized personnel to register FIRs from anywhere at any time, thus ensuring that critical information is captured and processed in real-time.

In addition to streamlining the registration process, the application promotes transparency and accountability within the law enforcement system. It facilitates better communication between citizens and authorities, fostering trust and cooperation in addressing community safety concerns. The user-friendly interface and secure access ensure that sensitive information is handled with care, making the application suitable for users with varying levels of technical expertise.

By adopting this modern approach to FIR registration, law enforcement agencies can significantly improve their operational efficiency, leading to quicker responses to incidents and enhanced community engagement. Ultimately, the Auto FIR Registration Application not only simplifies the FIR filing process but also plays a vital role in building a safer and more informed society.

## 1.1 Problem definition

The traditional process of filing First Information Reports (FIRs) is often cumbersome and time-consuming, leading to delays in law enforcement response. The existing manual systems are prone to errors, miscommunication, and loss of critical information, particularly in urgent cases such as theft, assault, murder, and suicide. This inefficiency can hinder timely investigations and affect public trust in the police system. The Auto FIR Registration System addresses these issues by providing a digital platform that simplifies and accelerates the FIR filing process, ensuring that reports are accurately recorded and quickly accessible to relevant authorities.

## 1.2 Methodology

- **Object-Oriented Programming (OOP)**:

  The application is built using an object-oriented paradigm, defining classes for key components such as User, FIR, and Notification.

  Each class encapsulates properties (attributes) and behaviors (methods) relevant to its functionality.

  For example, the FIR class has attributes like incident details, timestamp, and status, along with methods for validating and submitting the FIR.

- **Modular Design**:
  - The application is organized into various modules, each responsible for specific functionalities, promoting code reusability and maintainability.
  - The main modules identified include:
    - User Authentication Module
    - FIR Management Module
    - Email Notification Module
    - Database Interaction Module
    - User Interface Module
    - Record Viewing Module

- **Separation of Concerns**:
  - The code adheres to the principle of separation of concerns by dividing functionality into distinct classes and modules.
  - For instance, the FIR Management Module handles FIR-related operations,

while the Email Notification Module manages communication with stakeholders.
    - This separation results in organized, maintainable, and easier-to-understand code.

- **Event-Driven Programming**:
    - The application employs an event-driven programming approach, continuously checking for user interactions (e.g., button clicks, form submissions) and updating the application state accordingly.
    - Signal-slot connections are used to handle relevant events, ensuring smooth user interactions.

- **Main Application Loop**:
    - The main application loop is the core of the application, managing user interactions, processing FIR submissions, and updating the user interface.
    - This loop is implemented in the main function and runs until the application is closed, ensuring a responsive user experience.

- **Use of External Libraries**:
    - The application utilizes the PyQt5 library for creating the graphical user interface and SQLite for database management.
    - These libraries provide essential functionalities for rendering UI components, managing user input, and interacting with the database.

- **Code Documentation**:
    - The code includes comments and documentation to clarify the purpose and functionality of each component, aiding in understandability and maintainability.
    - Descriptive variable and function names are used to further enhance code readability.

## 1.3 Application Modules Overview

**User Authentication Module:**

- Manages user registration and login processes.
- Ensures secure access to the application using credentials.

**Incident Reporting Module:**

- Allows users to file FIRs by entering incident details, including type, victim information, and

incident date.

- Captures location data using the geocoder to enhance incident reporting accuracy.
- Provides options to upload supporting documents or images.

**Email Notification Module:**

- Sends automated email notifications to stakeholders upon FIR submission or updates.
- Manages email communications through SMTP protocols.

**Data Management Module:**

- Handles storage and retrieval of FIR data in a secure database.
- Implements CRUD (Create, Read, Update, Delete) operations for FIR records.

**Log Time Module:**

- Allows users to view log times of their activities or incident reports within the application.
- Provides an overview of user interactions for tracking purposes.

**Image Processing Module:**

- Utilizes OpenCV for any image processing tasks related to incident reporting (e.g., analyzing images submitted with FIRs).
- Enables features such as extracting text from images using Optical Character Recognition (OCR) with Tesseract.

**User Interface Module:**

- Provides a user-friendly interface for interacting with the application.
- Implements navigation features (navbar, buttons) for easy access to different functionalities.

## 1.4 Application Functionalities

**User Registration and Login:**

- Users can create accounts and log in to securely access the application.

**FIR Filing:**

- Users can file a new FIR by entering incident details, victim information, and incident date.
- Location data is captured using the geocoder to improve the accuracy of the FIR submission.
- Users can attach supporting documents or images to their FIR submissions.

**View and Manage FIRs:**

- Users can view their filed FIRs and track the status of each case.
- Authorized personnel can update FIR information and change case stages.

**Email Notifications:**

- Automatic email notifications are sent to relevant stakeholders upon FIR submission or status changes.

**Log Time Viewing:**

- Users can view log times for their activities within the application, providing insights into their interactions.

**Image Processing:**

- Users can utilize OpenCV for image-related tasks, such as analyzing or processing images associated with FIRs.

# CHAPTER - 2

# REQUIREMENT ANALYSIS

## 2.1 System Requirements:

### 2.1.1 Functional Requirements:

**User Interface (UI):**

- The application must provide a graphical user interface built with PyQt5 that allows users to interact with the system easily.
- UI elements should include buttons, text fields, and dialogs for various functionalities like file selection, FIR registration, and report generation.

**File Management:**

- The application must allow users to open and save files using a file dialog (QFileDialog).
- Users must be able to import CSV files containing FIR data for processing and viewing.

**Email Notifications:**

- Users must be able to send emails using the SMTP protocol.
- The application must allow users to specify recipients, subject lines, and message content.
- Users must receive notifications upon successful email sending or error messages if sending fails.

**Geolocation Services:**

- The application should utilize geolocation services (geocoder) to capture and display geographical data related to FIRs, such as the location of incidents.

**PDF Report Generation:**

- The application must generate PDF reports (using ReportLab) that summarize FIR details and allow users to save them locally.
- Users must be able to specify the filename and destination for the generated PDF reports.

**Image Processing and Text Recognition:**

- The application must allow users to upload images (e.g., images of number plates).

- The application must utilize OpenCV and Tesseract to perform Optical Character Recognition (OCR) on uploaded images to extract text information.

- Users should be able to view the recognized text and save it if necessary.

**Logging:**

- The application must implement logging (using the logging module) to track errors and important events during application execution.

- The logs should include timestamps, error descriptions, and other relevant information for debugging purposes.

**Data Handling:**

- The application must manage and store FIR data effectively, ensuring that users can view, edit, and delete records as needed.

- Data should be saved in a structured format, such as CSV, for easy access and reporting.

### 2.1.2 Non – Functional Requirements:

**Performance:**

- The application must respond to user inputs within 2 seconds for most operations (e.g., loading pages, submitting forms).

- The system should handle at least 100 simultaneous users without significant performance degradation.

**Usability:**

- The user interface must be intuitive and easy to navigate, allowing users to complete tasks with minimal training.

- All UI elements should be clearly labeled, and the application should provide tooltips or help messages for additional guidance.

**Reliability:**

- The application must maintain an uptime of at least 99.5%, ensuring that users can access it whenever needed.

- The system should be able to recover gracefully from crashes, preserving user data and states where applicable.

**Security:**

- User credentials (passwords) must be stored securely using hashing and salting techniques.

- The application should implement SSL/TLS encryption for data transmitted over the network, especially for email communications and user authentication.

- Access control measures should be in place to restrict unauthorized users from accessing sensitive data.

**Scalability:**

- The application must be designed to scale efficiently as the number of users increases.

- The system should support the addition of new features and functionalities without significant rework.

**Maintainability:**

- The codebase must be well-documented, allowing for easier updates and maintenance by developers.

- The application should follow best practices in coding standards and architecture, making it easier to identify and fix bugs**.**

**Portability:**

- The application should be compatible with major operating systems (Windows, macOS, Linux) without requiring significant modifications.

- Users should be able to run the application from different environments, such as local machines or cloud-based platforms.

**Compliance:**

- The application must comply with relevant legal and regulatory standards, such as data protection laws (e.g., GDPR, HIPAA) regarding user data and privacy.

- The application should also adhere to industry standards for software development and security.

**Localization:**

- The application should support multiple languages to cater to users from different regions.

- It should allow users to switch languages easily without requiring a restart**.**

**Monitoring and Logging:**

- The application must include monitoring tools to track performance metrics and user activity.

- Logs should be generated for all significant events (e.g., errors, transactions) and should be easily accessible for troubleshooting**.**

## 2.1.3 Software Requirements:

**Operating System:**

- The application should be compatible with major operating systems:
  - Windows 10 or later
  - macOS Mojave (10.14) or later

- Linux distributions (Ubuntu 20.04 LTS or later)

**Programming Language:**

- Python 3.7 or later should be used as the primary programming language for **development.**

**Libraries and Frameworks:**

- GUI Framework: PyQt5 for building the graphical user interface.
- Email Handling: smtplib and email.mime libraries for sending emails.
- File Handling: csv for reading and writing CSV files.
- Geolocation: geocoder library for retrieving geographical data.
- PDF Generation: ReportLab for creating PDF reports.
- Image Processing: OpenCV for image handling and manipulation.
- OCR: Tesseract (via pytesseract) for optical character recognition.
- Logging: Python logging module for tracking application events and errors.

**Database:**

- SQLite or any other relational database management system (RDBMS) for storing user information and FIR data.
- Database should support SQL queries for data retrieval and manipulation.

**Development Environment:**

- Integrated Development Environment (IDE) such as:
  - PyCharm
  - Visual Studio Code
  - Jupyter Notebook (for prototyping)

**Dependencies:**

- List of required Python packages, which can be managed using pip and should be specified in a requirements.txt file:
  - PyQt5
  - smtplib (part of the standard library, no need to install separately)
  - geocoder
  - reportlab
  - opencv-python
  - pytesseract
  - logging (part of the standard library, no need to install separately)

**Version Control:**

- Git for version control and collaboration.
- A repository hosted on platforms like GitHub or GitLab for code management and issue tracking.

**Testing Framework:**

- Use unittest or pytest for unit testing and ensuring the reliability of the application.

**Documentation Tools:**

- Sphinx or MkDocs for generating documentation for the application.
- Markdown or reStructuredText for writing documentation files.

**Deployment Environment:**

- A suitable environment for deploying the application, which could be a local machine, a server, or a cloud platform.

- Optional: Docker for containerization to ensure consistent environments different machines.

### 2.1.4 Hardware Requirements:

**Processor:**

- Minimum: Intel Core i3 or equivalent
- Recommended: Intel Core i5 or higher

**RAM:**

- Minimum: 4 GB
- Recommended: 8 GB or more

**Storage:**

- Minimum: 500 MB of free disk space for installation
- Recommended: 1 GB or more for additional data storage and application files

**Graphics:**

- Minimum: Integrated graphics (for basic UI)
- Recommended: Dedicated graphics card (for advanced image processing features)

**Network:**

- Required for email notifications and data synchronization
- Minimum: Broadband internet connection

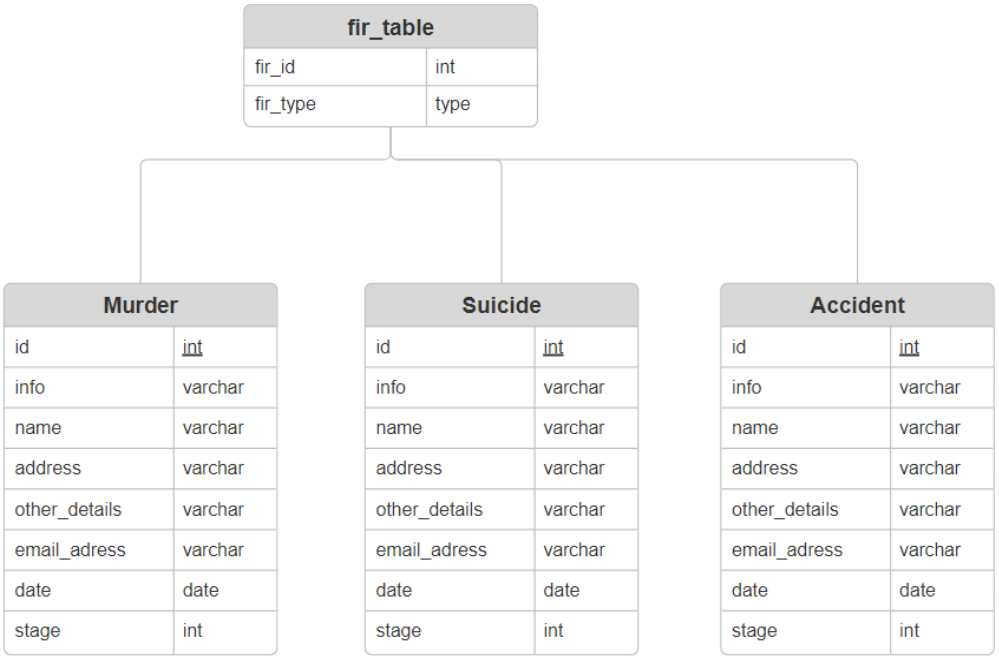
**Peripheral Devices:**

- Webcam (if implementing features for capturing images)
- Scanner (for digitizing documents, if applicable)
- Printer (for printing FIR reports, if required)

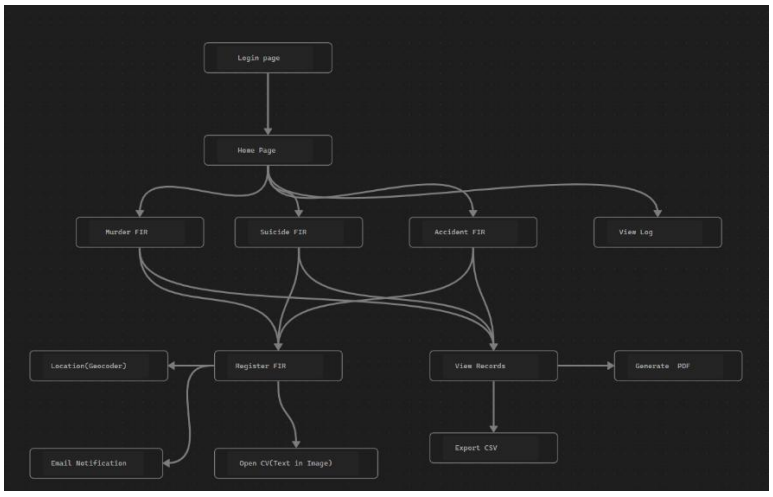
# CHAPTER – 3

# SYSTEM DESIGN

## 3.1 ER Diagram:




## 3.2 User Flow Diagram:



# CHAPTER – 4

# IMPLEMENTATION

The main application initializes the necessary components for the user interface and backend functionalities. The PyQt5 framework is used to build the GUI, while modules like sqlite3, csv, and geocoder support data management and location services. Email capabilities are integrated using smtplib and related classes, and the application is capable of generating PDF reports with reportlab. Image processing and OCR are handled using cv2 and pytesseract, respectively. Logging is configured to track application events and errors.
Next, the code defines three classes representing the game objects:

- Import Statements
  - **import sys**: Used for system-specific parameters and functions.
  - **import sqlite3**: For database operations.
  - **from PyQt5 import QtCore, QtGui, QtWidgets**: Imports PyQt5 modules for GUI creation.
  - **from PyQt5.QtWidgets import QFileDialog**: Provides file dialog functionality.
  - **import smtplib**: For sending emails.
  - from email.mime.multipart import MIMEMultipart and from email.mime. text import **MIMEText**: For creating email messages.
  - **import csv**: For handling CSV files.
  - **import geocoder**: For geographical location services.
  - **from reportlab.pdfgen import canvas and from reportlab.lib.pagesizes import letter**: For generating PDF documents.
  - **import os**: For interacting with the operating system.
  - **import cv2**: For image processing.
  - **import pytesseract**: For Optical Character Recognition (OCR).
  - **import logging**: For logging application events.

- **Login Page class:**
  - LoginPage inherits from QtWidgets.QWidget, making it a QWidget that can be used in a PyQt5 application.
  - Login Page class creates a login interface for a PyQt5 application.
  - It includes text fields for the username and password, a login button, and other UI elements. The class also handles basic layout and styling, and it emits a signal when a successful login attempt is made.
  - The __init__method initializes the paddle's position, dimensions, and color.

- **Home Page class:**

- HomePage inherits from QtWidgets.QWidget, making it a QWidget that can be used in a PyQt5 application.
- It includes a navigation bar, a title, and three cards for different types of FIR (First Information Report) records.
- Each card contains buttons for viewing and registering FIRs. with corresponding signals for navigation.
- The class also manages layout, styling, and signal connections for navigation.
- The design ensures a structured and interactive user interface with custom styling and layout.

- **RegisterFIRDialog class:**

  - The RegisterFIRDialog class creates a dialog for registering a First Information Report (FIR) for different types of incidents (e.g., murder, suicide, accident).
  - This dialog collects FIR details, handles image uploads for number plates, sends confirmation emails, and inserts the data into a database.
  - The RegisterFIRDialog class provides a comprehensive dialog for registering FIRs with various input fields, image upload functionality for number plates, geolocation fetching, email validation, and sending confirmation emails.
  - It also handles database insertion and emits a signal upon successful FIR registration.
  -

- **RecordsPage class:**

  - The RecordsPage class creates a user interface (UI) for displaying, updating, and exporting First Information Report (FIR) records of a specified type.
  - It includes functionality for saving changes to the records, generating PDFs for individual records, and exporting all records to a CSV file.
  - The __init__ Method Initializes the RecordsPage with the specified fir_type.
  - The methods handle database operations, email notifications, and file generation to facilitate smooth record management.

- **Navbar class:**

  - Provides a navigation bar with buttons for home, viewing logs, and logging out. It uses signals to notify the main application of button clicks.
  - **ViewLogTimePage**: Displays log times from a file in a table and includes a button to go back to the home page. It reads the log file, parses the log lines, and displays them in a table.

- **MainWindow class:**

  - MainWindow class that serves as the main application window for a PyQt application.
  - This window uses a QStackedWidget to switch between different pages, such as login, home, records, and log viewing pages.
  - The create_connection function establishes a connection to an SQLite database and creates a table if it doesn't already exist.

# CHAPTER - 5

# SCREEN OUTPUTS

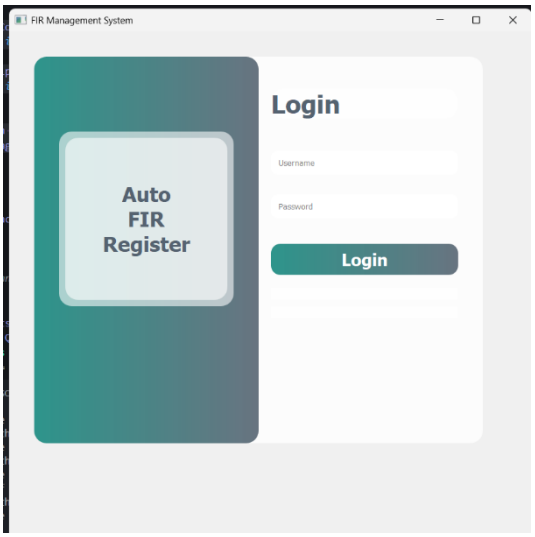



**Figure 7.1: Login Page**

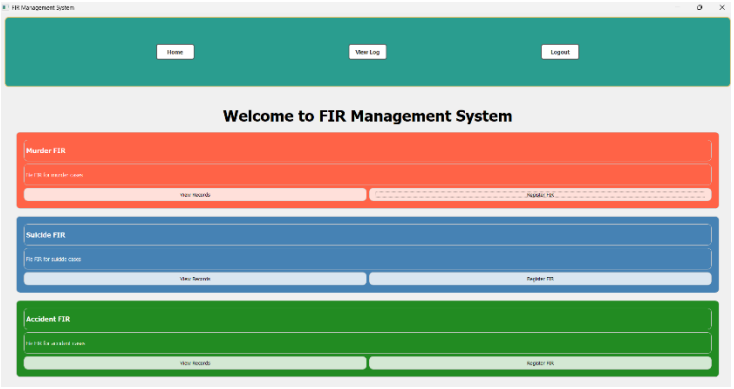



**Figure 7.2: Home Page**

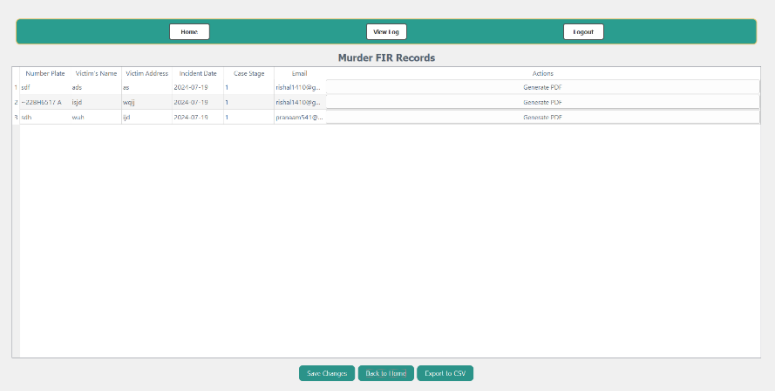



**Figure 7.2: Records Page**

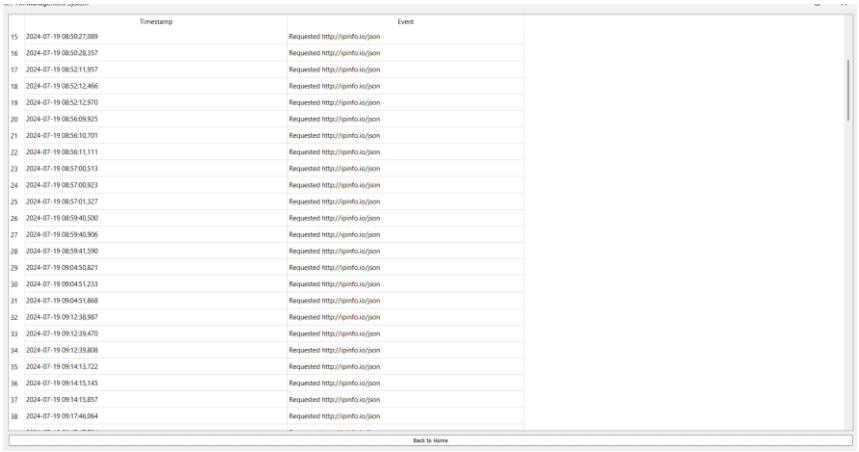



**Figure 7.4: Log Page**

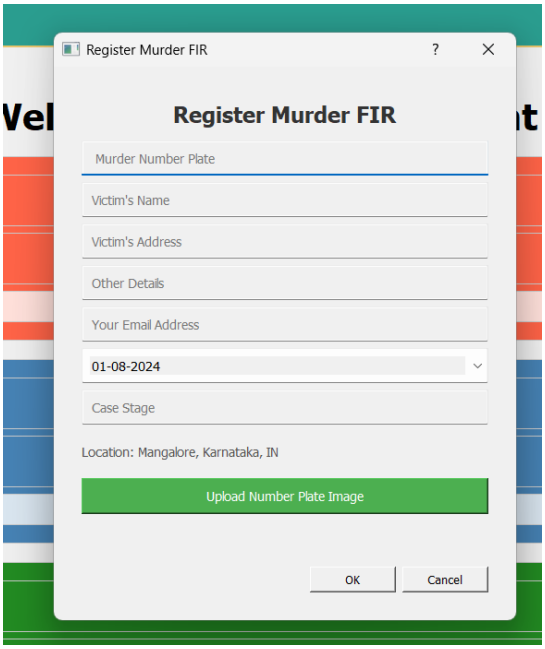



**Figure 7.5: Register Dialog**

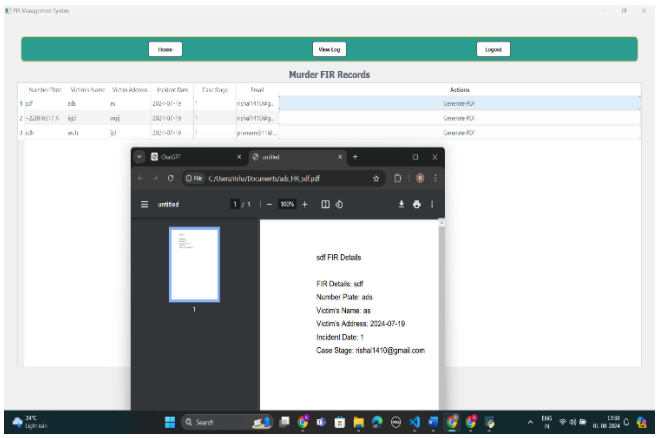



**Figure 7.6: PDF Generation**

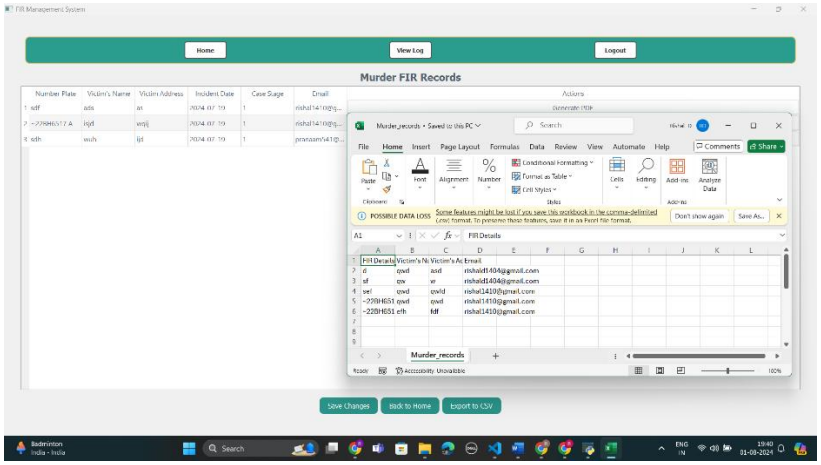



**Figure 7.6: CSV File**

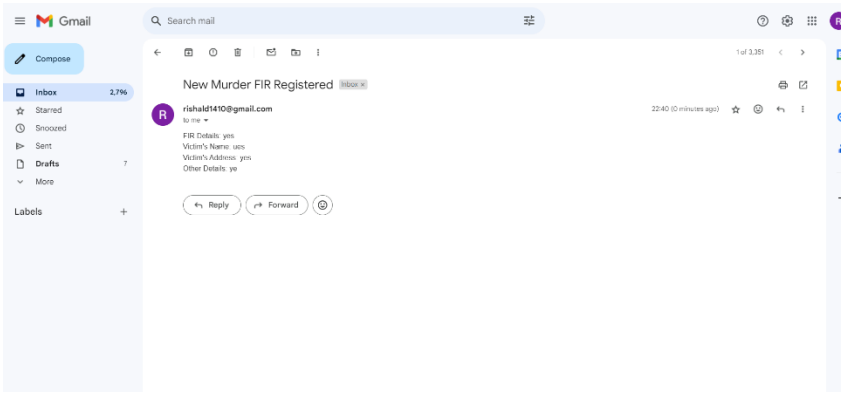



**Figure 7.6: Email Notification**

# CHAPTER - 6
# CONCLUSION

The Auto FIR Registration Application represents a significant advancement in the way First Information Reports (FIRs) are filed and managed within law enforcement agencies. By leveraging modern technology, this application streamlines the FIR registration process, making it more accessible, efficient, and user-friendly.

Through its robust functionalities, including user authentication, incident reporting, and automated email notifications, the application enhances communication between citizens and law enforcement, fostering transparency and trust within the community. The modular design and object-oriented programming principles ensure that the application is maintainable and extensible, allowing for future enhancements and adaptations to meet evolving needs.

Moreover, by implementing data integrity and security measures, the application safeguards sensitive information while providing users with a reliable platform for reporting incidents. The insights gained through reporting and analytics further empower law enforcement agencies to make informed decisions and respond effectively to community safety concerns. The ability to track case stages and incident history provides a comprehensive view of ongoing investigations, facilitating better resource allocation and operational efficiency.

In addition, the user-friendly interface promotes ease of use for both citizens and law enforcement personnel, reducing the time and effort required to file and process FIRs. The application's capacity for integrating with existing databases and systems enhances interoperability within law enforcement agencies, creating a more unified approach to public safety.

Furthermore, the application can be adapted to accommodate various types of incidents, allowing law enforcement to categorize and prioritize cases based on urgency and severity. This flexibility not only improves response times but also ensures that critical incidents receive the attention they deserve.

In conclusion, the Auto FIR Registration Application not only simplifies the FIR filing process but also contributes to a more efficient and responsive law enforcement system, ultimately promoting a safer and more informed society. By bridging the gap between citizens and law enforcement, this application paves the way for a collaborative approach to crime prevention and community safety, reinforcing the foundational principles of justice and accountability.

# CHAPTER - 7

# FUTURE ENHANCEMENTS

While the FIR Management System project achieves its primary objectives, there are several areas for potential enhancement:

- **Advanced Search and Filters**:

  Implementing advanced search capabilities and filters to allow users to find specific FIRs quickly.

- **User Roles and Permissions**:

  Adding a more sophisticated user management system with different roles and permissions to further secure the application.

- **Data Analytics and Reporting**:

  Integrating data analytics tools to generate insightful reports on FIR records and trends.

- **Cloud Integration**:

  Transitioning to a cloud-based database solution to enable real-time access and updates acrossmultiple

# REFERENCES


1. Wilson, K. (2023, July 31). *Python Made Easy: A First Course in Computer Programming Using Python*. Programming Applications Works.
2. Matthes, E. (2019). *Python Crash Course* (2nd ed.)