Cybercrime Reporting System

August 18, 2025

1. Introduction

Our Cybercrime Reporting System is a Java-based application designed to simplify cybercrime reporting for civilians and help law enforcement departments reach out to people online. It allows users to file cybercrime cases (for example, Online harassment, financial fraud etc.), and it allows admins to observe and manage the filed cases, analyze and update crime statistics, and track area-based crime records. The system prioritizes cases involving minors/seniors depending on their age. This creates an efficient and effective way to deal with online crimes and offences that are often left unattended. As a result, in a world where online presence controls almost every aspect of our life, the online space is becoming more hostile and lawless by the day. This project is built using Java Swing for the GUI and binary file storage for data persistence; it complies fully with the project's OOP requirements while focusing on digital crime management.

2. Methodology

Core Implementation:

The system was developed using core Object-Oriented Programming (OOP) and different concepts of Java such as, encapsulation, method-overloading, method-overriding etc. The design incorporated the following concepts:

Encapsulation: Various methods and classes were kept private to ensure the data safety and prevent data breach. This ensures that the method from other classes are hidden and prevents external code from triggering admin workflows accidentally etc.

Inheritance: The abstract User class was extended by both the Admin and Reporter classes, reducing code duplication and promoting hierarchy-based design.

Polymorphism: The getRole() method from the User class was overridden in the subclasses Admin and Reporter, allowing role-specific behavior while maintaining a consistent interface. Also, method-overloading is used in FileHandler.java in

saveData(Object data) and saveData(Object data, String filename) to input different types of datas in two methods with same name.

Interfaces: The ICasePriorityCalculator interface was implemented to define flexible case-priority calculation rules.

File Handling

To ensure data persistence, the system utilized binary files (cases.dat, crime_data.dat) for storing case and crime records. A dedicated FileHandler class was implemented to manage all file operations, including saving and loading, while incorporating error handling to prevent data corruption.

GUI Components

The user interface was developed using Java Swing, providing an interactive and user-friendly environment. The system included seven distinct windows, with separate dashboards for administrators and reporters, ensuring role-based functionality and clear navigation. It was made with the help of AI.

Exception Handling

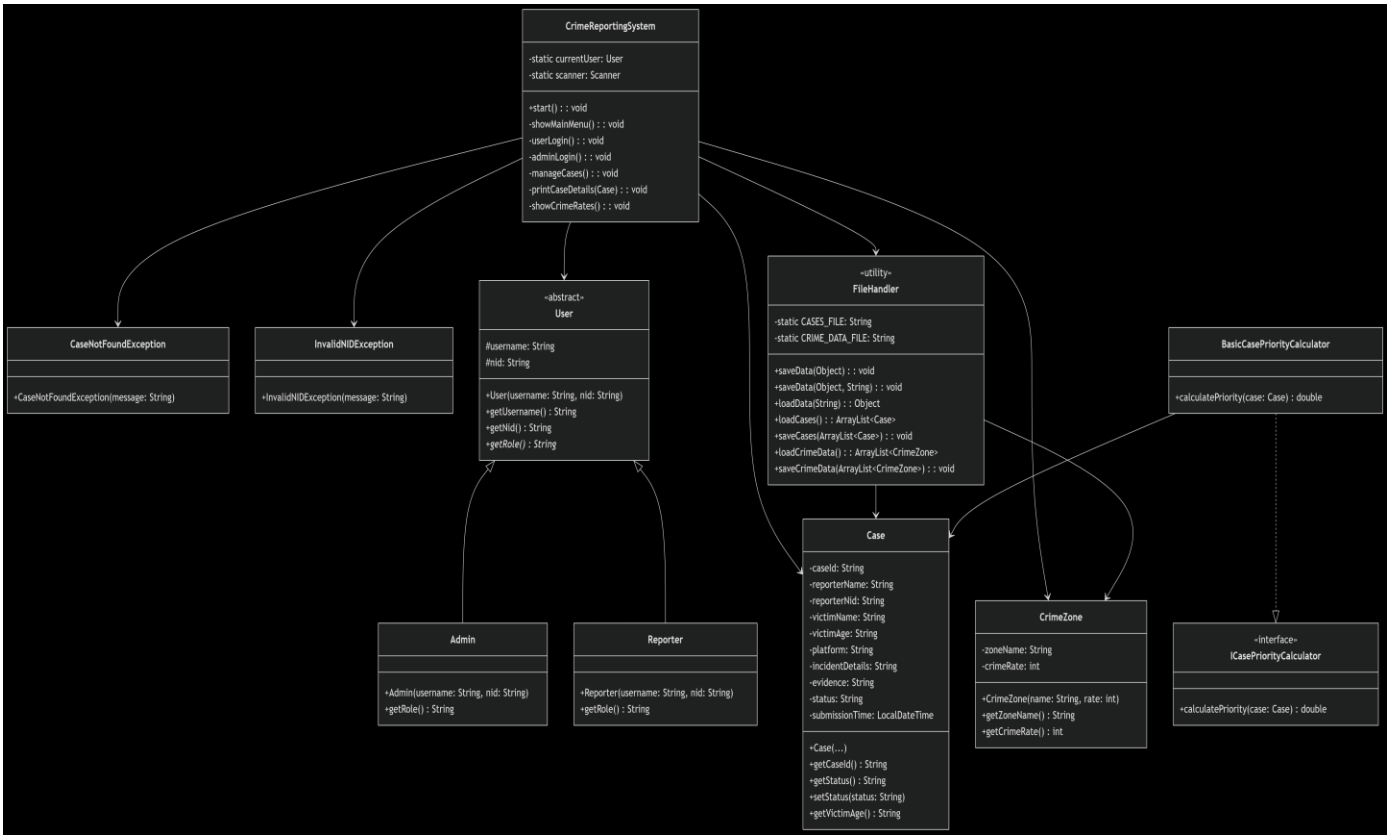
The system incorporated both custom and built-in exception handling to improve reliability and robustness. Custom exceptions included:

InvalidNIDException for handling invalid National ID formats. NID numbers must be 10 digits. Not less or not more.

CaseNotFoundException for invalid or missing case references.

Additionally, built-in exceptions were leveraged to handle file-related errors and invalid user inputs, ensuring smooth system performance and user feedback.

3. Class UML Diagram



4. Uses

Main usage of our code stands to be a small-scale cyberbullying or online crime reporting system. This connects people to the law enforcements in places where laws are often little to non-existent. Here are different usages and aspects of this program:

1. For Reporters (Citizens):

    - File new cybercrime or online crime cases with victim details and incident descriptions
    - Track existing case statuses with ease (Pending/Investigating/Resolved)
    - Simple login using username and NID number
    -

2. For Administrators:

- Update and view cases stored in the binary file and change status of cases according to the investigation
- View number of cases and status of cases ongoing, resolved, or waiting for approval
- Add and update crime rates of different areas in realtime

3. For General Public:

- Access crime rate heatmaps by geographical zone
- Identify high-risk areas with higher crime counts

5. Limitations

- GUI interface is unable to show crime rates based on zones.
- Cases are prioritized based on age for now which is ineffective in certain cases.
- The admin id and password are hardcoded into the code, which creates a security risk.
- Does not check if the NID inputs are digits or not.

6. Conclusion

The Cybercrime Reporting System successfully showcases core OOP concepts, file I/O operations, and GUI design. Key achievements include:

- Role-based access control (Admin/Reporter)
- Automated case prioritization for minors/seniors
- Crime statistics visualization via area-based records
- Successfully implementing FileIO to save and load data

Future Work:

1. Implementing the system with a database and turning it into a webpage system for ease of access.

2. Adding a hotline system for faster response.
3. Being able to take images as evidence and store them in a secured file.
4. Measuring priority of cases more accurately. Instead of using just age as a factor, time submitted report, degree of crime or disturbance will also play a role.
5. Adding another user role “Inspector” for police officers or professionals to be able to access cases on the go for better response time to situations.
6. More detailed area-based crime rate option for police officers(Inspector role) to stay up-to-date.

The system provides a base for efficient cybercrime reporting system while fully complying with the project’s technical and ethical requirements.