2025

Anonymous Corruption Reporting System

Helen Mabara

su210025c - Research Project

Helen

Anonymous Corruption Reporting System

01-Jan-25

Table of Contents

CHAPTER 1: INTRODUCTION 3

1.1. Background 3

1.2. Problem Statement 3

1.3. Objective 4

1.4. Methodology 5

1.5. Feasibility 7

1.6. Project Scope and Limitation 9

1.7. Significance of the project 10

Chapter Two: Analysis 12

2.1. Existing System Description 12

2.2. New System 13

Use Case Documentation 15

State Chart Diagrams 24

Sequence Diagrams 25

Identifying Change Access 29

Chapter Three(3) Design 31

3.1 Purpose and goals of design 31

3.2 Proposed Software Architecture 32

Chapter 4: Implementation 38

Chapter 5: Testing & Evaluation 39

Chapter 6: Conclusion 43

CHAPTER 1: INTRODUCTION

1.1. Background

Corruption is a global issue that hinders the development and growth of societies. A major challenge in tackling corruption is the fear of retaliation faced by whistleblowers. This project aims to build an **Anonymous Corruption Reporting System** that allows individuals to submit corruption reports securely and anonymously, ensuring their identity is protected while encouraging civic participation.

The **National Integrity Commission (NIC)** is a government-backed, independent agency established to **combat corruption, promote transparency**, and **ensure accountability** in public and private sector institutions across Nigeria. Operating under the mandate of the Federal Ministry of Justice, the NIC serves as a watchdog by investigating reports of unethical practices, misuse of public resources, bribery, and fraud.

1.2. Problem Statement

In many organizations and governments, individuals who witness corrupt practices often remain silent due to fear of exposure, retaliation, or professional consequences. There exists a gap in the availability of secure, anonymous platforms for reporting such practices. This project seeks to fill that gap by developing a web-based system that allows users to submit and track corruption reports anonymously.

Effects on the company

- **Financial Loss**
  Corruption can lead to embezzlement, bribery, kickbacks, and other fraudulent activities that result in direct financial losses for the company. This can impact profitability and shareholder trust.

- **Reputation Damage**
  When a company is associated with corruption or unethical practices, its reputation can be severely damaged. This can lead to a loss of customer trust, investor confidence, and can harm relationships with stakeholders.

- **Legal Consequences**
  Engaging in corrupt practices can lead to legal repercussions including fines, lawsuits, and even criminal charges. This can have long-lasting effects on the company's operations and standing in the industry.

- **Employee Morale**
  Workplace corruption can create a toxic work environment where employees feel demotivated, undervalued, and disengaged. This can lead to high turnover rates, decreased productivity, and a negative company culture.

- **Unfair Business Practices**
  Corruption can give some employees or partners unfair advantages over others, leading to an uneven playing field in the market. This can stifle competition, hinder innovation, and harm the overall business ecosystem.

- **Loss of Trust**
  Internal and external stakeholders may lose trust in the company if they perceive it as corrupt. This loss of trust can be difficult to regain and may affect the company's ability to attract and retain clients, investors, and talent.

- **Operational Inefficiencies**
  Corruption can result in inefficient decision-making processes, misallocation of resources, and a lack of accountability within the organization. This can hinder growth, innovation, and overall performance.

1.3.Objective

- **General Objective**
  The primary objectie of this system is to **create a secure and anonymous corruption reporting platform**, , track case progress, and ensure secure handling of sensitive evidence. The system should ensure **data integrity, secure communication, and transparency in case management**, empowering investigative bodies to act on reported cases efficiently

Specific Objective

- **Provide an intuitive user interface** for submitting corruption reports while maintaining anonymity.
- **Ensure secure data transmission** using encryption mechanisms to protect whistleblower identities.
- **Implement a case management system** that tracks investigations, assigns cases, and provides structured updates.
- **Enable secure evidence submission** with validation features for uploaded documents and media files.

- **Develop role-based access control (RBAC)** to regulate different user interactions, including investigators, administrators, and general users.
- **Support audit logging and transparency mechanisms** for tracking case resolution processes.
- **Optimize system performance** for accessibility across different devices, including web and mobile platforms.
- **Ensure compliance with ethical and legal standards** for corruption reporting and investigation procedures.

1.4.Methodology

The analyst used the following **requirement gathering methods** to collect data from the organisation:

1. Interview.
2. Observation.
3. Document Analysis

Interview

An interview was held with one of the employees.The major aim of the interview was to gather information about the efficiency and effectiveness of the current reporting system, its reliability and major shortfalls of the current system.

- Can you describe the process of reporting incidents of corruption within ClearView Enterprises?
- What channels are available for employees to report instances of corruption or unethical behavior?
- How user-friendly do you find the current corruption reporting system? Are there any usability issues or challenges you have encountered?
- Can you explain the steps involved in submitting a corruption report through the current system?
- Is the current system anonymous, or are there concerns about confidentiality and potential retaliation for whistleblowers?
- How are corruption reports tracked and managed once they are submitted? Are there any issues with the tracking and management process?
- What kind of feedback or communication do whistleblowers receive after submitting a corruption report?
- Have you received any training or guidance on how to use the corruption reporting system effectively?
- In your opinion, what are the strengths of the current system in addressing corruption within the organization?

- What are the main weaknesses or limitations of the current corruption reporting system?
- How transparent is the process of investigating and resolving corruption reports within **National Integrity Commission (NIC)**?
- Are there any compliance or legal considerations that the current system may not adequately address?
- Have you witnessed any instances where the corruption reporting system was not effective in addressing unethical behavior? If so, can you provide an example?
- What suggestions do you have for improving the current corruption reporting system to make it more efficient and reliable?
- How do you think **National Integrity Commission (NIC)**, can foster a culture of transparency and accountability through its corruption reporting mechanisms?

Observation

The primary objective of this observation was to assess how employees interacted with the feedback portal, identify usability challenges, and determine the effectiveness of the system in collecting and managing employee concerns.

Selection of Participants: A group of employees from different departments (IT, HR, Finance, and Customer Support) was chosen to participate in the observation.

Introduction: The participants were informed about the study, which aimed to improve the user experience and efficiency of the feedback portal.

Task Assignment: Employees were asked to submit a simulated feedback report on an existing workplace issue using the portal.

Observation Process: The observer monitored and documented employee interactions with the system.

Special attention was given to:

- **Ease of Access:** How quickly users found and logged into the system.
- **Navigation:** Clarity in locating the feedback submission section.
- **Submission Process:** Complexity of the report submission form.
- **User Experience:** Frustrations, delays, or ease of use while entering details.
- **Response Mechanism:** Whether employees received timely confirmation or automated responses after submission.

Requirement Modelling

In modeling the corruption reporting system, I would choose the **MVC (Model-View-Controller)** architectural pattern for development. Here's the rationale for selecting MVC and the tools that could be used for the project:

Model View Controller(MVC)

MVC separates the system into three interconnected components - Model (data), View (user interface), and Controller (logic). This separation of concerns makes the system more organized, maintainable, and scalable, which is crucial for a system like a corruption reporting tool that involves data management, user interaction, and business logic. By utilizing the MVC pattern along with Agile development practices, the corruption reporting can be developed in a structured and efficient manner, ensuring modularity, maintainability, and responsiveness to changing needs. The selected tools align with the requirements of the project, facilitating the development of a robust and user-friendly system for reporting and addressing instances of corruption within the organization.

Tools Used

- **Programming Language:** PHP (for backend development using Laravel framework)
- **Database:** MySQL (for data storage and management)
- **SQL:** (data manipulation)
- **Frontend Frameworks:** Tailwind CSS and Bootstrap CSS (for responsive design)
- **Version Control:** Git & Github (for version control and collaboration)
- **Development Environment:** Visual Studio Code (for coding)
- **UML Modeling Tool**: Mermaid **(**for creating UML diagrams)
- **Postman:** API Testing Tool

1.5.Feasibility

Feasibility study is a method used to assess the benefits of the new system. It is carried out by the analyst to decide whether it’s feasible to develop a new system.

Economic feasibility

Here, the main purpose of the feasibility was to see if the cost incurred by the project would weigh against the benefits enjoyed from it, that is a cost benefit analysis.

- **Equipment cost:** A desktop computer or a laptop is to be bought and this is where the system will be operated on. A printer is also needed for printing reports.

- **Maintenance cost:** These include costs incurred from hiring the personnel that will be maintaining the system to keep it running smooth. The analyst concluded that the project is economically feasible because the above costs are relatively low as compared to the costs that are being incurred in total.

Technical feasibility

This was carried out to measure the availability of technical resources and expertise. Although the organization is not computerized, the manager who is going to be using the system already has a background of general computing therefore the project is technically feasible because the user(s) knows how to operate computers in general.

Tine feasibility: The system is going to be developed in-house therefore it’s feasible because there will be enough time to implement the system.

The feasibility study that was carried out by the analyst concluded that the project was feasible.

Requirements Specification

Software requirements

- Microsoft Windows OS (Windows 10 / Windows 11) / Linux / Mac – these are the platforms on which the system will be developed and run on.
- Laravel 10+ - PHP framework
- PHP 8.1+
- Composer – PHP package management
- Xampp – database
- Node.js (for Tailwind)
- Node Package Manager – JavaScript package management
- VS Code – Integrated development Environment(IDE) / code editors
- Git – version control

Hardware requirements (a computer with the following)

- Processor - Intel(R) Core(TM) i5 CPU @ 2.60GHz or above – this will enable the system to perform its operations faster.
- RAM – 4GB – this will improve the system's efficiency.
- HDD / SSD – 500GB+ - this will be for the storage of the system files.
- CD-DVD ROM – for backing up data on a CD
- Internet connectivity

1.6.Project Scope and Limitation

The Anonymous Corruption Reporting System is designed to address the challenges of securely and anonymously reporting instances of corruption within an organization or government body. The system provides users with a confidential, efficient, and user-friendly platform to submit reports while ensuring their identity remains undisclosed. It offers structured data management to help authorities analyze trends and take appropriate action against corruption cases.

Key ways the system solves the problem:

- **Anonymity & Security:** Users can submit reports without fear of retaliation, ensuring safety.
- **Structured Data Collection**: Organizes reported cases for easy categorization and analysis.
- **Automated Processing**: Enables faster initial review of reports by relevant authorities.
- **Multi-Platform Accessibility**: Users can submit reports via web applications from different devices.
- **Accountability Measures**: Helps organizations track corruption reports and responses efficiently.

Services & Deliverables: Throughout the project development lifecycle, the following services and deliverables will be achieved:

- **User Registration & Authentication** (for authorized investigators while keeping reports anonymous).
- **Report Submission Module** (enabling users to upload evidence and details securely).
- **Database & Backend Development** (using Laravel for robust data management).
- **Frontend Development** (with Tailwind CSS & Bootstrap CSS for a responsive UI).
- **Report Tracking & Feedback System** (allowing authorities to track investigations).
- **Security Enhancements** (encryption, data privacy protocols, access control).
- **Testing & Deployment** (unit testing, integration testing, and final deployment).

Project Limitations

Despite the system's capabilities, certain limitations may exist:

- **Dependence on User Honesty**: The system relies on truthful reporting, which cannot be fully controlled.
- **Limited Investigation Power**: The platform facilitates reporting but does not directly conduct investigations.
- **Potential for False Reports**: Without a verification mechanism, there is a possibility of inaccurate or malicious submissions.
- **Access to Internet**: The system requires an internet connection, limiting users in remote areas.
- **Legal and Organizational Constraints**: Some entities might have strict policies that impact the system's effectiveness.

1.7.Significance of the project

The Anonymous Corruption Reporting System provides a transformative solution for institutions combating corruption by enabling secure, anonymous reporting.

1. **Improved Transparency:** Organizations can foster a culture of accountability by enabling whistleblowers to report wrongdoing without fear of retaliation.

2. **Enhanced Security:** By ensuring anonymity, the system protects individuals who might otherwise be discouraged from reporting corrupt activities.

3. **Data-Driven Decision Making:** Provides structured insights into reported cases, helping organizations identify recurring corruption trends.

4. **Strengthened Organizational Integrity:** Encourages ethical behavior among employees by creating a secure mechanism for reporting misconduct.

5. **Compliance with Regulatory Standards:** Helps institutions adhere to anti-corruption laws and ethical business practices.

Contribution to the project area

The system contributes significantly to the domain of **anti-corruption technology** through the following advancements: The system plays a vital role in promoting transparency, protecting whistleblowers, and strengthening ethical governance, marking a significant step in leveraging technology to combat corruption.

1. **Innovative Use of Technology**

   Utilizes **Laravel** for a secure backend, combined with **Tailwind CSS & Bootstrap CSS** for an intuitive and responsive frontend.

2. **Efficient Reporting Mechanism**

   Introduces a streamlined interface for submitting reports quickly and securely.

3. **Scalability & Adaptability**

   Designed to accommodate different organizations, from government agencies to private enterprises.

4. **Advancement of Digital Governance**

   Supports the growing trend of digital tools to foster integrity, accountability, and ethical practices.

Chapter Two: Analysis

2.1. Existing System Description

The **National Integrity Commission (NIC)**, has a basic corruption reporting system in place to address unethical practices within the organization. While it serves the purpose of allowing employees to report instances of corruption, it lacks some key features and sophistication compared to a more advanced system.The organization relies on a basic reporting mechanism where employees can submit corruption reports through email or a designated internal mailbox. This system lacks multiple reporting channels and does not offer an anonymous reporting option.

Problem Definition

The corruption reporting system used by **National Integrity Commission (NIC)**, while serving as a basic mechanism for employees to report unethical behavior, exhibits several weaknesses that hinder its effectiveness and reliability.

Drawbacks of the Current System

- **Basic Reporting Mechanism**: This system lacks multiple reporting channels and does not offer an anonymous reporting option.
- **Limited Transparency**: The company's corruption reporting system lacks transparency in terms of how reports are handled and investigated. There is a lack of clear procedures and guidelines for employees on how their reports will be processed.
- **Manual Tracking**: Reports submitted through the system are manually tracked using spreadsheets or simple databases. This manual tracking process can be inefficient, prone to errors, and lacks a centralized system for managing and monitoring reports.
- **Lack of Response Mechanism**: ClearView Enterprises does not have a structured response mechanism in place for addressing corruption reports. There is no defined process for investigating reports, providing feedback to whistleblowers, or taking corrective actions based on the findings.
- **Absence of Training Programs**: The company does not conduct regular training programs to educate employees about the importance of reporting corruption or the procedures to follow when submitting a report. This lack of awareness may result in underutilization of the reporting system.
- **Minimal Compliance Measures**: The corruption reporting system at **National Integrity Commission (NIC)**, lacks compliance measures with relevant laws and regulations governing the reporting of unethical behavior. This exposes the company to potential legal risks and challenges.

2.2. New System

Non-functional requirements and contraints

1. **Security**

- All reports must be encrypted at rest and in transit using AES-256 and HTTPS.
- Implement multi-tiered access controls based on Role-based Access Control (RBAC) to restrict user privileges.
- Regular vulnerability scans and audit logging of administrative actions.

2. **Performance**

- The system should respond to user actions within 2 seconds under normal load.
- Capable of handling up to 500 concurrent sessions during peak usage with minimal latency.

3. **Usability**

- Intuitive, minimalistic UI to encourage anonymous users to comfortably submit reports.
- Accessibility compliance with WCAG 2.1 for screen reader and keyboard navigation.

4. **Reliability**

- System uptime should be 99.9% or higher, backed by regular backups and failover support.
- Recovery Time Objective (RTO) ≤ 1 hour; Recovery Point Objective (RPO) ≤ 10 minutes.

5. **Maintainability**

- Modular design using Laravel service containers for easy updates and scalability.
- Well-documented code and API endpoints with inline comments and a versioning policy.

6. **Portability**

- Cross-browser compatibility (Chrome, Firefox, Safari, Edge).
- Deployment-ready for both Linux servers and containerized environments using Docker.

Constraints

1. **Time Constrain:** System must be developed and deployed within the academic semester as per final-year timeline.
2. **Budgetary Constraint**
3. Utilization of open-source technologies (Laravel, MySQL, Tailwind CSS, Bootstrap) to reduce licensing costs.
4. **Technical Constraint**: Hosting will be on a shared academic server with limited processing power and memory.
5. **Legal & Ethical Constraint**

    - Compliance with Zimbabwe's Data Protection Act and anti-corruption legislation.
    - Ensuring reporter anonymity without retaining any personally identifiable information (PII).

6. **Team Constraint**: Solo development project, meaning scope must match individual workload capacity.

Functional Requirements

1. **User Registration and Login (for Admins & Reviewers):** Admins and reviewers must authenticate securely before accessing the system.
2. **Anonymous Report Submission**

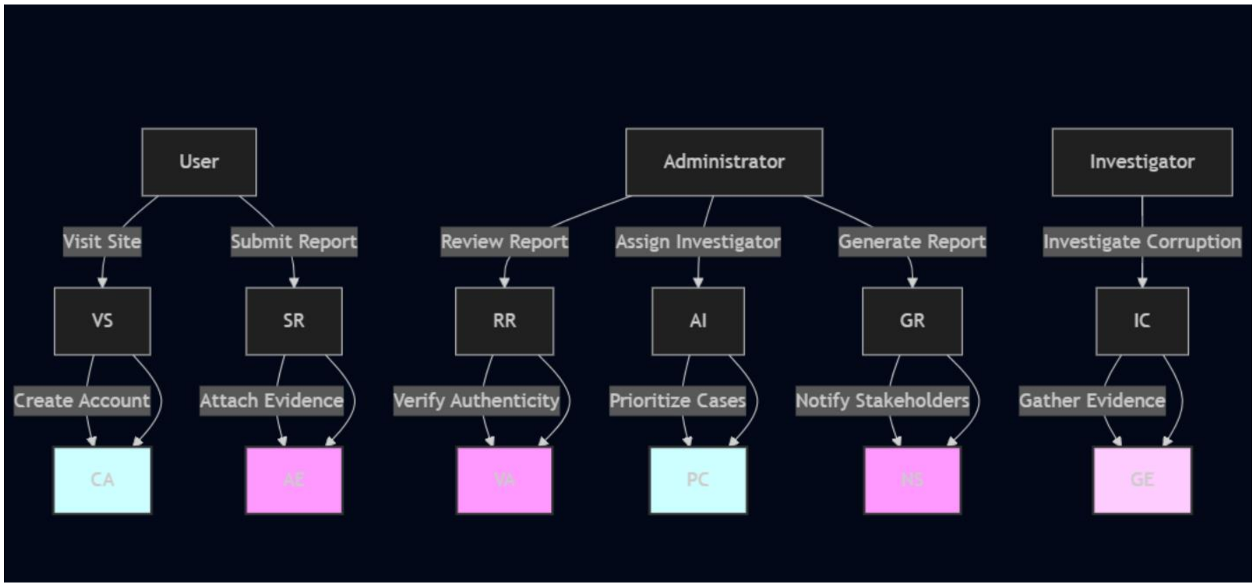
    - Any user can submit a report without authentication.
    - The system must not log any identifying metadata (e.g., IP address, email).

3. **Report Categorization:** Users can classify reports by corruption type, institution, and urgency.
4. **Attachment Upload:** Option to include supporting evidence (PDF, images, video, etc.) with reports.
5. **Report Review and Management**

    - Reviewers can view, prioritize, and flag submitted reports.
    - Status updates: "New," "Under Review," "Resolved," "Archived."

6. **Admin Management:** Admins can manage users, assign permissions, and configure system settings.
7. **Notifications and Alerts:** Internal alerts for new submissions, critical reports, or reviewer assignments.
8. **Audit Logging:** Actions by admins and reviewers must be logged for accountability.
9. **Search and Filter Reports:** Reviewers and admins can search reports by keywords, status, or category.
10. **System Maintenance:** Admins can export data for backups and trigger scheduled system maintenance

Use Case Diagram



Use Case Documentation

1. **User Visits Site**

Preconditions:

- The user has internet access.
- The site is operational.

Postconditions

The user can browse reports and submit a corruption report.

Main Course of Action

- User opens the system’s website.
- The system loads available content.
- User decides whether to create an account or browse anonymously.

Alternate Course of Action:

If the site fails to load, show an error message.

2. **User Visits Site**

Preconditions

User chooses to register.

Postconditions

User account is created, and they can submit and track reports.

Main Course of Action:

1. User clicks "Sign Up."
2. User enters personal details (name, email, password).
3. System validates and stores user data.
4. User receives confirmation.

Alternate Course of Action

If registration fails, prompt the user to correct errors.

3. **Submit Report**

Preconditions:

User is logged in or chooses to submit anonymously.

Postconditions:

Report is stored in the system for review.

Main Course of Action:

1. User clicks "Submit Report."
2. User fills in report details (corruption type, corruption date, location, incident description).
3. User uploads supporting evidence.
4. System stores the report and generates a tracking ID.

Alternate Course of Action:

- If upload fails, prompt user to retry.
- If the user decides not to upload evidence, proceed without it.

4. **Attach Evidence**

Preconditions:

User submits a report.

Postconditions:

Evidence is linked to the report.

Main Course of Action:

1. User clicks "Upload Evidence."
2. User selects a file and confirms the upload.
3. System verifies and stores the evidence.

Alternate Course of Action:

If the file format is incorrect, display an error.

Preconditions:

Admin credentials exist.

Postconditions:

Admin gains access to system functionalities.

Main Course of Action:

1. Admin enters login details.
2. System verifies and grants access.

Alternate Course of Action:

If login fails, prompt a retry.

5. **Attach Evidence**

Preconditions:

Reports exist in the system.

Postconditions:

Admin reviews reports for action.

Main Course of Action:

1. Admin selects "View Reports."
2. System retrieves and displays report list.

Alternate Course of Action:

- If no reports exist, display a message.

6. **Download Evidence**

Preconditions:

Evidence was uploaded with a report.

Postconditions:

Admin downloads and reviews evidence.

Main Course of Action:

1. Admin clicks "Download Evidence."
2. System retrieves and offers the file for download.

Alternate Course of Action:

If file retrieval fails, display an error.

7. **Assign Status To Report**

Preconditions:

Report exists.

Postconditions:

Status is updated.

Main Course of Action:

- Admin selects a report.
- Admin chooses a status (New, under review, investigation, evidence, tracking, pending, in progress, resolved, closed, escalated).
- System updates the report status.

Alternate Course of Action:

If the report is missing, display an error.

8. **Assign Status To Report**

Preconditions:

Report exists.

Postconditions:

Report is permanently removed.

Main Course of Action:

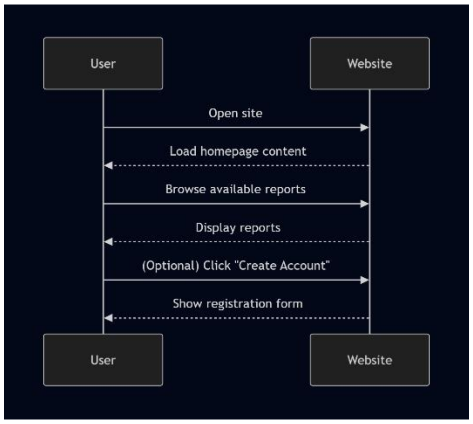
- Admin selects a report.
- Admin clicks "Delete."
- System confirms deletion.

Alternate Course of Action:

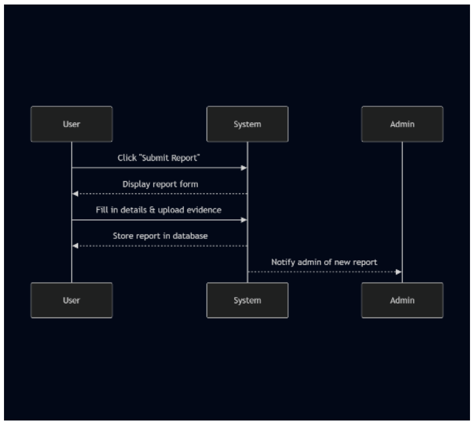
If deletion fails, prompt a retry.

Use Case Diagrams

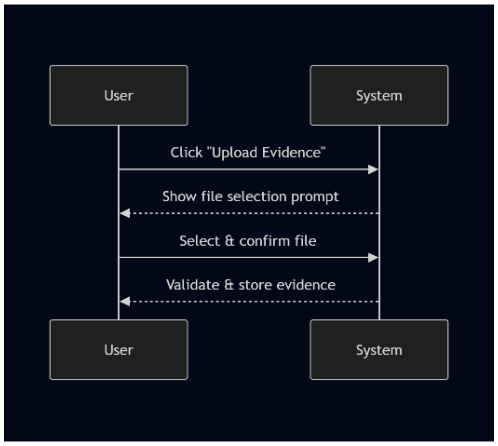
1. User Visits Site



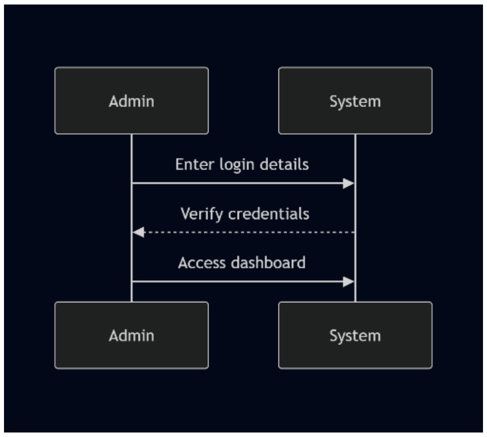
2. Submit Report



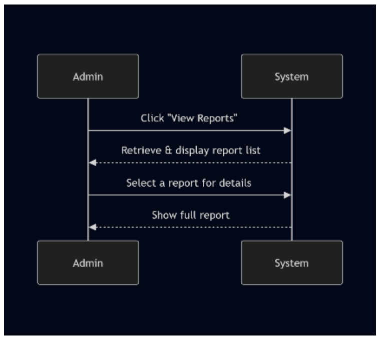
3. Attach Evidence



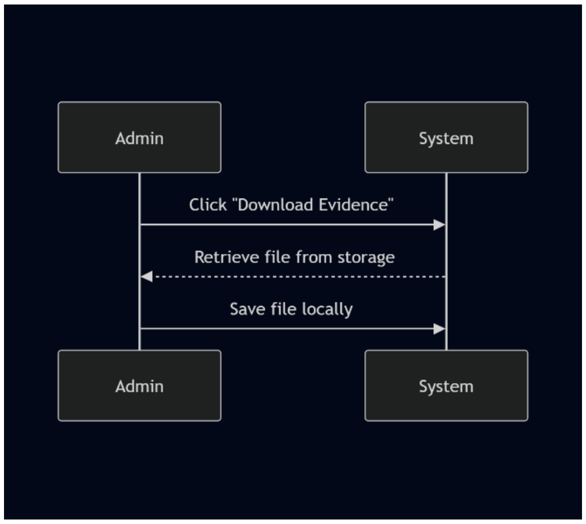
4. Admin Logs Into System



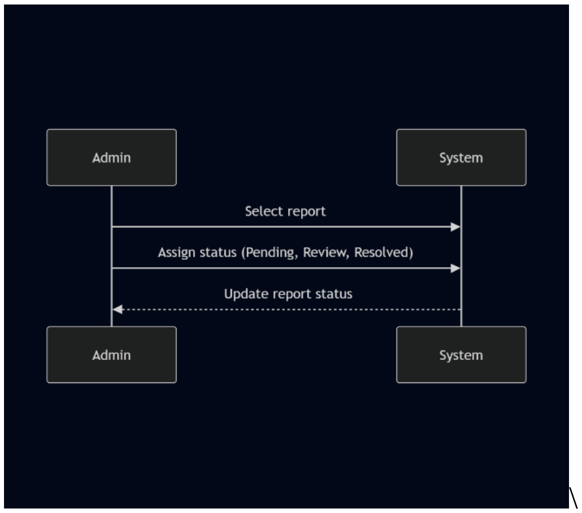
5. View Submitted Reports



6. Download Evidence

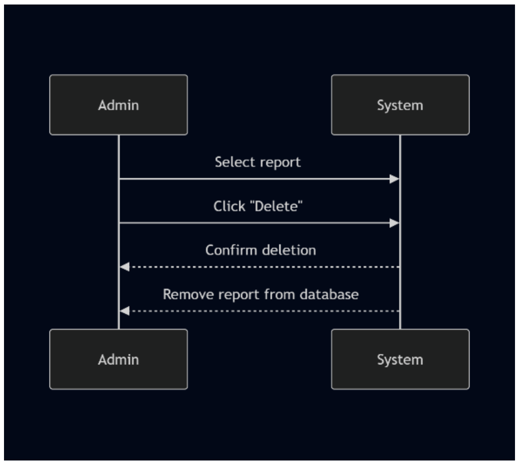


7. Assign Status To Report



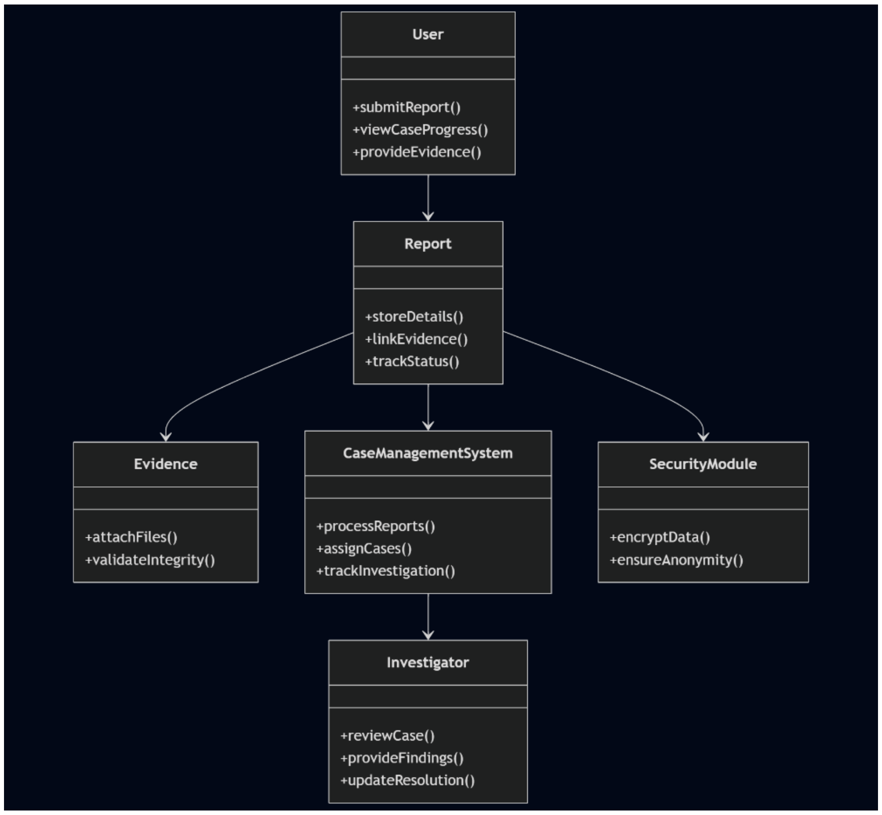
\

8. Delete Report



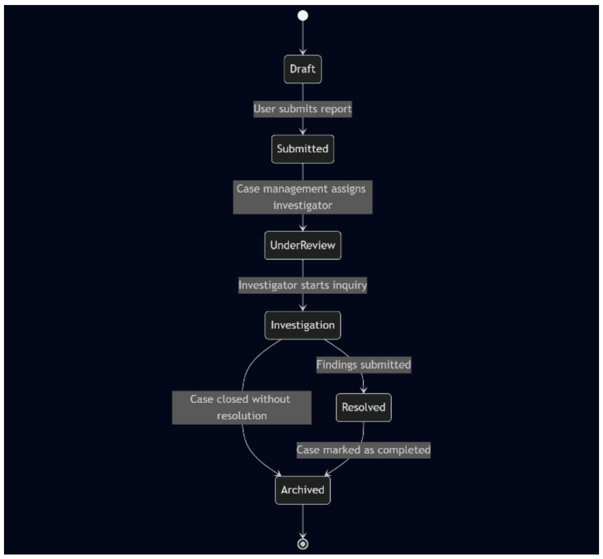
System Class Diagram

Class diagram representing the system architecture, showing relationships, inheritance and associations between objects

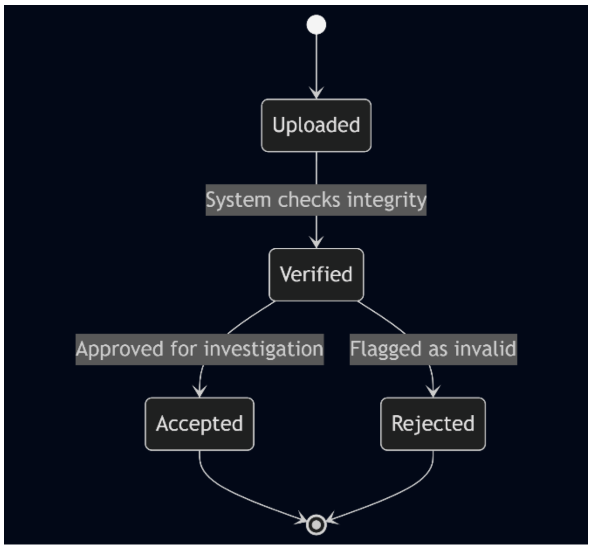


State Chart Diagrams

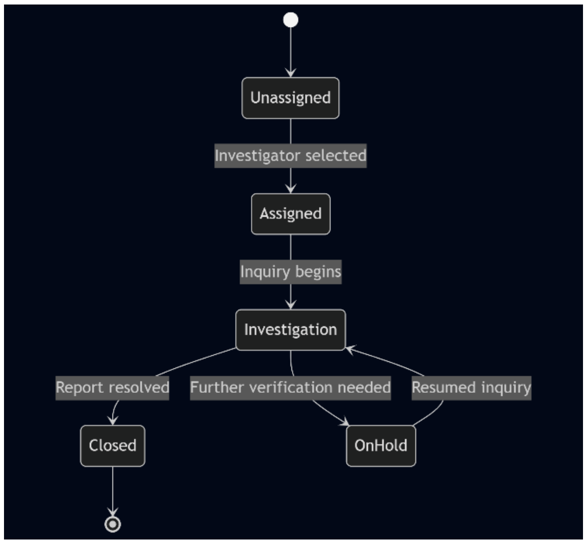
1. Report State Diagram



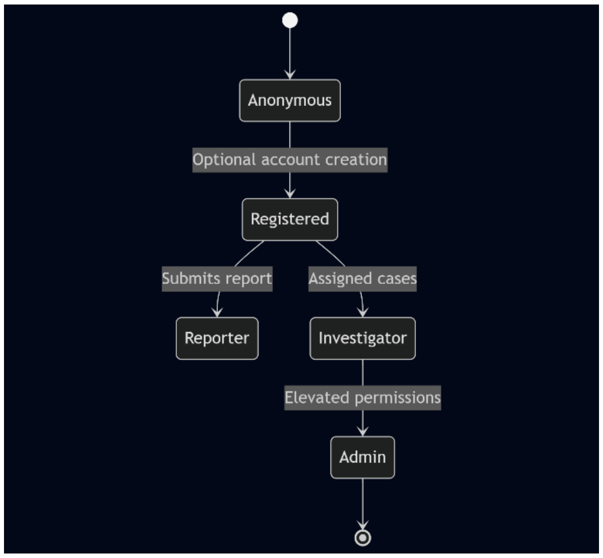
2. Evidence State Diagram



3. User state diagram

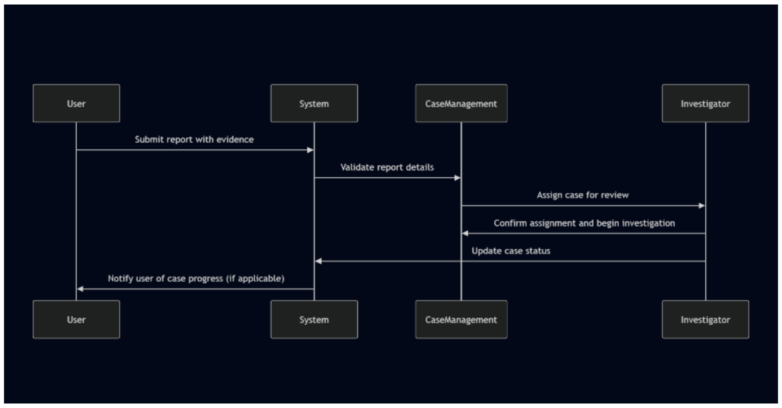


4. Case Management State Diagram

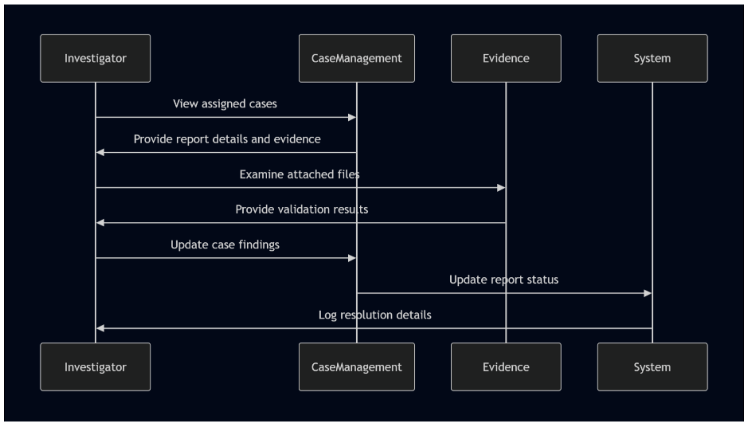


Sequence Diagrams

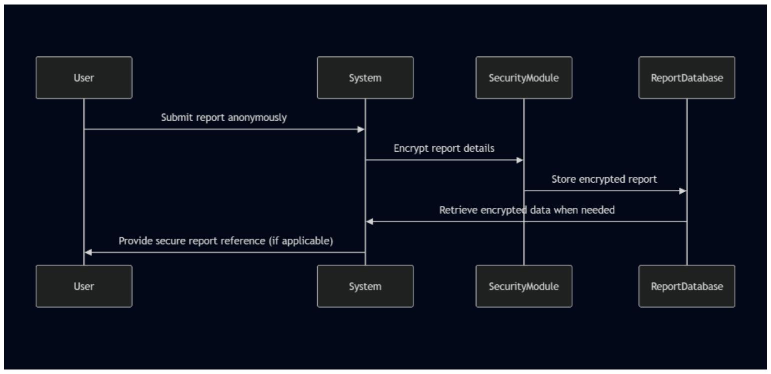
1. Submitting a corruption report


User
System
CaseManagement
Investigator
Submit report with evidence
Validate report details
Assign case for review
Confirm assignment and begin investigation
Update case status
Notify user of case progress (if applicable)

2. Investigating a report



3. Secure Report Encryption


User
System
SecurityModule
ReportDatabase
Submit report anonymously
Encrypt report details
Store encrypted report
Retrieve encrypted data when needed
Provide secure report reference (if applicable)

Key abstraction with CRS anaylsis

Key concepts and CRC Cards

Each CRC card card consist of Class, Responsibities, and Collaborators

1. **User**

1) **Responsibilities**
    - Submit corruption reports anonymously.
    - View case progress (if permitted).
    - Provide additional supporting evidence.
2) **Collaborators**
    - Report
    - Authentication System (optional, if role-based access applies)

2. **Report**

1) **Responsibilities**
    - Store corruption incident details.
    - Link attached evidence (documents, images).
    - Track report status.

2) **Collaborators**
    - User
    - Evidence
    - Case Management System

3. **Evidence**

1) **Responsibilities**
    - Maintain attachement (documents, images)
    - Validate evidence integrity
2) **Collaborators**
    - Report

4. **Case Management System**
   1) **Responsibilities**
   - Process incomig reports
   - Assign cases to investigative officers
   - Track investigation progress

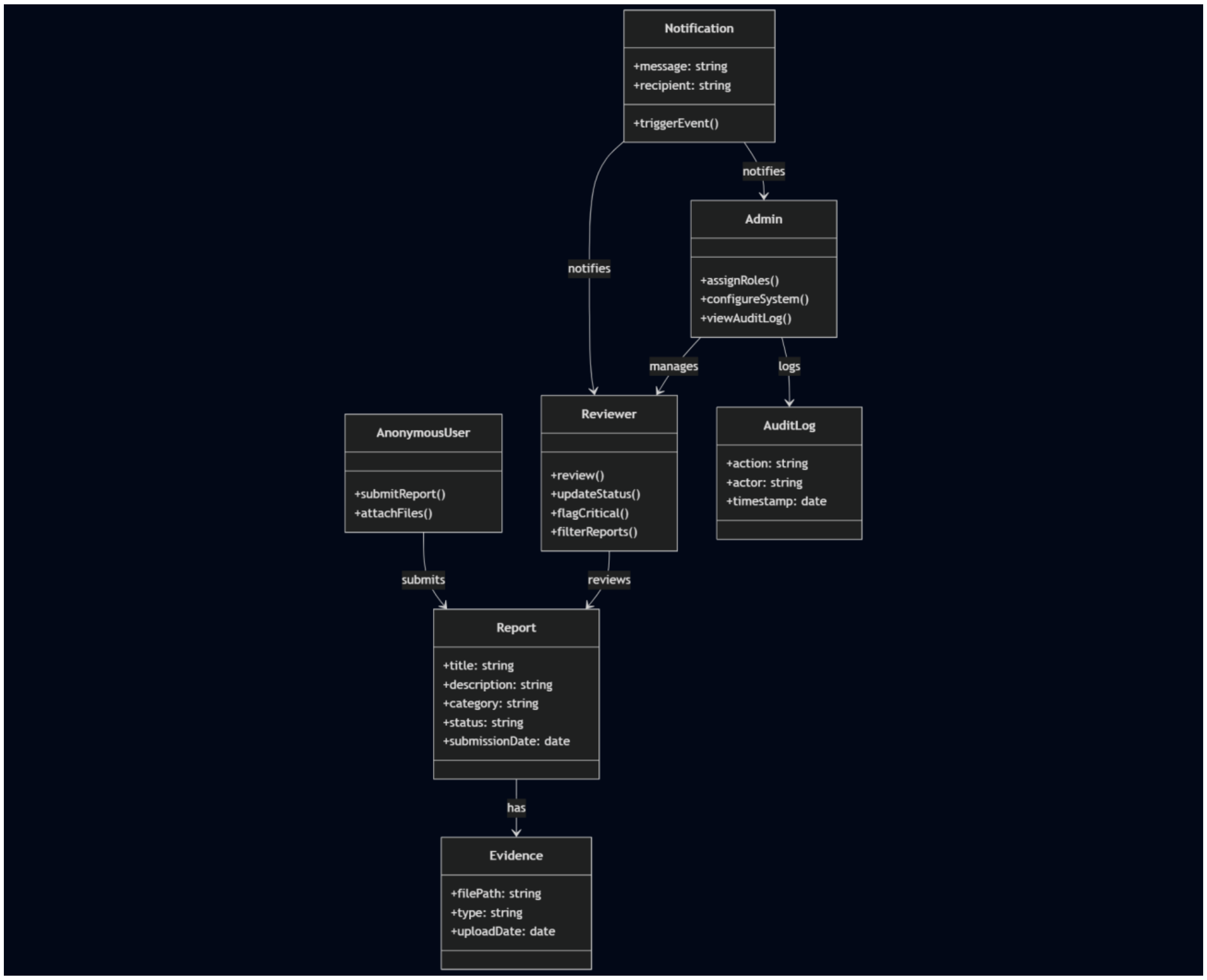
   2) **Collaborators**
      - Report
      - Investigator

5. **Investigator**
   1) **Responsibilities**
      - Review assigned cases
      - Provide findings
      - Update report resolution status

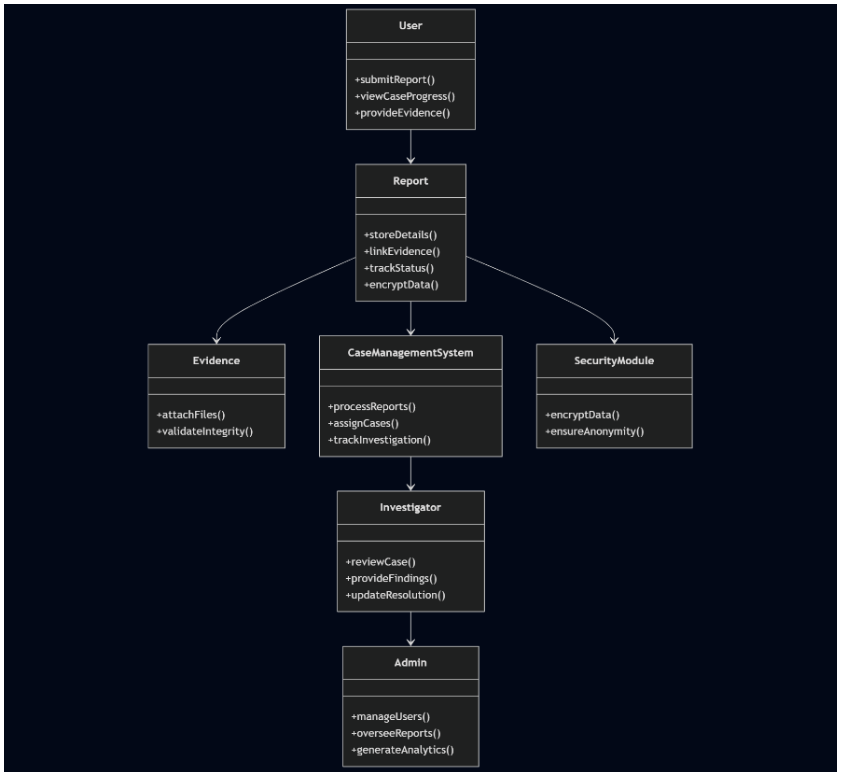
   2) **Collaborators**
      - Case Management System
      - Report

6. **Encryption and Security Mode**
   1) **Responsibilities**
      - Ensure reports remain anonymous and secure
      - Encrypt sensitive user information
   2) **Collaborators**
      - Report
      - User aunthentication

CRC Diagram


Notification
+message: string
+recipient: string
+triggerEvent()
notifies
Admin
+assignRoles()
+configureSystem()
+viewAuditLog()
notifies
manages
logs
AnonymousUser
+submitReport()
+attachFiles()
Reviewer
+review()
+updateStatus()
+flagCritical()
+filterReports()
AuditLog
+action: string
+actor: string
+timestamp: date
submits
reviews
Report
+title: string
+description: string
+category: string
+status: string
+submissionDate: date
has
Evidence
+filePath: string
+type: string
+uploadDate: date

Conceptual modelling: Class Diagram



Identifying Change Access

Change cases that describe potential modifications, their likelihood, and their impact

1. **Adding Multi-language Support**
    - **Potential Change:** Introduce language localization for users from diverse linguistic backgrounds
    - **Likelihood:** Moderate – If international adoption occurs or multilingual accessibility is required.
    - **Impact:** Enhances usability for non-English-speaking users but increases development complexity.

2. **Implementing AI-based Fraud Detection**
    - **Potential Change:** Utilize machine learning to detect false reports or recurring fraud patterns.
    - **Likelihood:** Low to moderate – May be necessary for large-scale usage.
    - **Impact:** Improves efficiency but requires additional computational resources and model training.

3. **Enhancing Encryption with BlockChain Storage**
   - **Potential Change:** Store corruption reports using blockchain for immutable and transparent audit trails.
   - **Likelihood:** Moderate – Growing demand for transparency in whistleblower systems.
   - **Impact:** Strengthens data integrity but introduces implementation challenges and higher storage costs.

4. **Integration with External Legal Entities**
   - **Potential Change:** Enable direct communication between the system and law enforcement agencies.
   - **Likelihood:** High – Required for effective case resolution and enforcement.
   - **Impact:** Improves real-world case actionability but needs strict data-sharing policies.

5. **Mobile App Version of the System**
   - **Potential Change:** Develop a mobile application for users to submit reports via smartphones.
   - **Likelihood:** High – Many users prefer mobile accessibility.
   - **Impact:** Enhances ease of use but increases development effort and requires cross-platform support.

6. **Implementing Role-based Access Control (RBAC)**
   - **Potential Change:** Assign different access levels to administrators, investigators, and users.
   - **Likelihood:** Very High – Necessary for secure system operation.
   - **Impact:** Improves security and data protection but adds complexity in permission management.

Chapter Three(3) Design

3.1 Purpose and goals of design

1. **Performace**
   - The system must be optimized for handling multiple concurrent reports efficiently without lag.
   - Database queries and report retrieval should be fast, ensuring smooth user experience.
   - Caching mechanisms can be leveraged to store frequently accessed data, reducing load times.
   - Asynchronous processing for background tasks like notifications or analytics could improve responsiveness.

2. **Dependability**
   - Ensuring high availability by deploying the system on reliable cloud infrastructure.
   - Robust error-handling mechanisms to maintain stability in case of unexpected failures.
   - Secure data storage with encryption to prevent tampering or unauthorized access.
   - Comprehensive logging and monitoring to detect anomalies and ensure transparency.

3. **End-user Experience**
   - A user-friendly interface designed with accessibility in mind, enabling anonymous reporting effortlessly.
   - Responsive design incorporating both Bootstrap CSS breakpoints and Tailwind CSS configurations for seamless access across different devices.
   - Minimal input required for users to submit reports, reducing barriers to reporting corruption.
   - Clear feedback mechanisms, such as confirmation messages upon submission.

4. **Security and Access Control**
   - Strict authentication and role-based access control to limit data exposure to authorized personnel only.

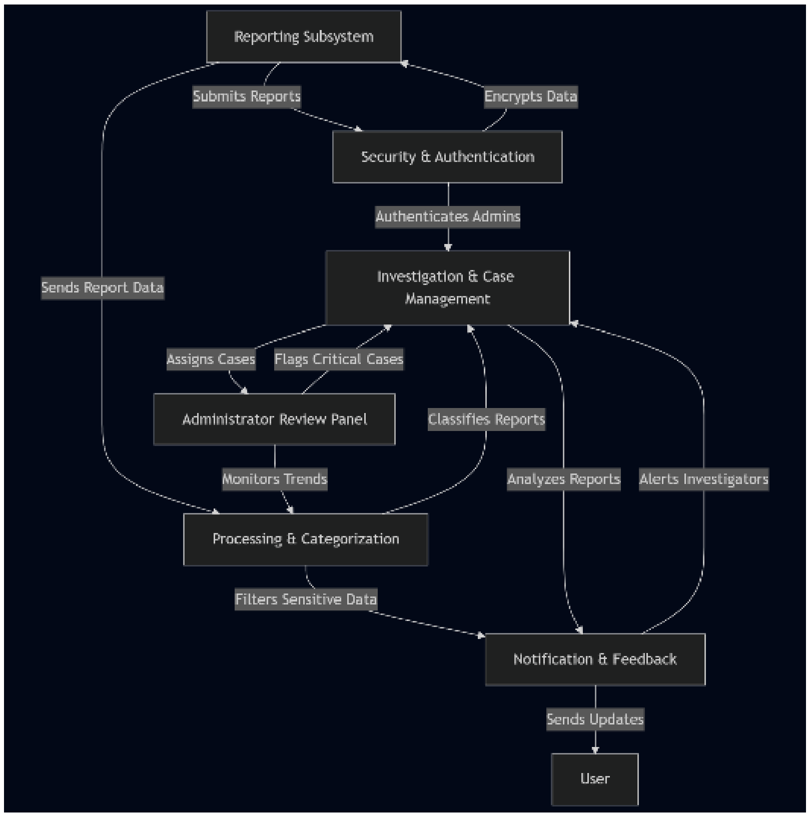
- Anonymity preservation through encryption techniques that prevent user identification.
- Secure communications using HTTPS and data hashing to ensure confidential submissions

5. **Scalability and Extensibility**
    - Modular architecture allows easy addition of new reporting categories or analytical features.
    - API support enables integration with external systems for data-sharing or cross-verification.
    - Cloud deployment ensures horizontal scalability to accommodate a growing number of users.

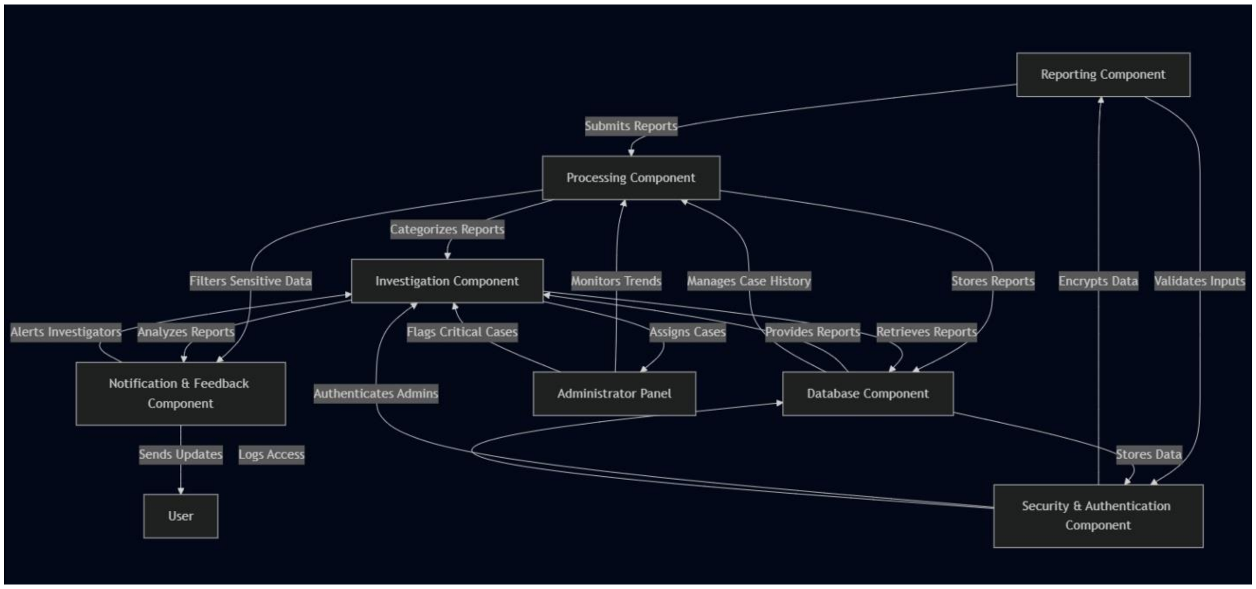
6. **Mainainability and Documentation**
    - Well-structured codebase using Laravel's best practices for easier debugging and maintenance.
    - Detailed documentation outlining functional and non-functional requirements.
    - Use of tools like Mermaid.js and PlantUML to visualize system workflows and improve understanding for future developers.

3.2 Proposed Software Architecture

Subsystem Decompostion



Component Diagram



Deployment Diagram

A **deployment diagram** helps visualize how different software components are installed across various machines and how they interact, especially in a distributed system.

1. **User Devices(Clients)**
   - **Installed Components:** Web-based frontend (Tailwind CSS, Bootstrap CSS, JavaScript)
   - **Interacts With:** Web server via HTTPS for submitting reports
   - **Deployment Location:** Laptops, smartphones, tablets

2. **Web server (Application Layer)**
   - **Installed Components:** Laravel backend, RESTful API handlers
   - **Interacts With:**
     - Client devices (serving web requests)
     - Security Layer (encrypting report data)
     - Database Server (storing reports)

3. **Security and Authentication server**
   - **Installed Components:** Encryption algorithms, anonymity enforcement module
   - **Interacts With:**
     - Web Server (ensuring secure submissions)
     - Database Server (storing encrypted identifiers)
   - **Deployment Location:** Separate security-dedicated cloud instance

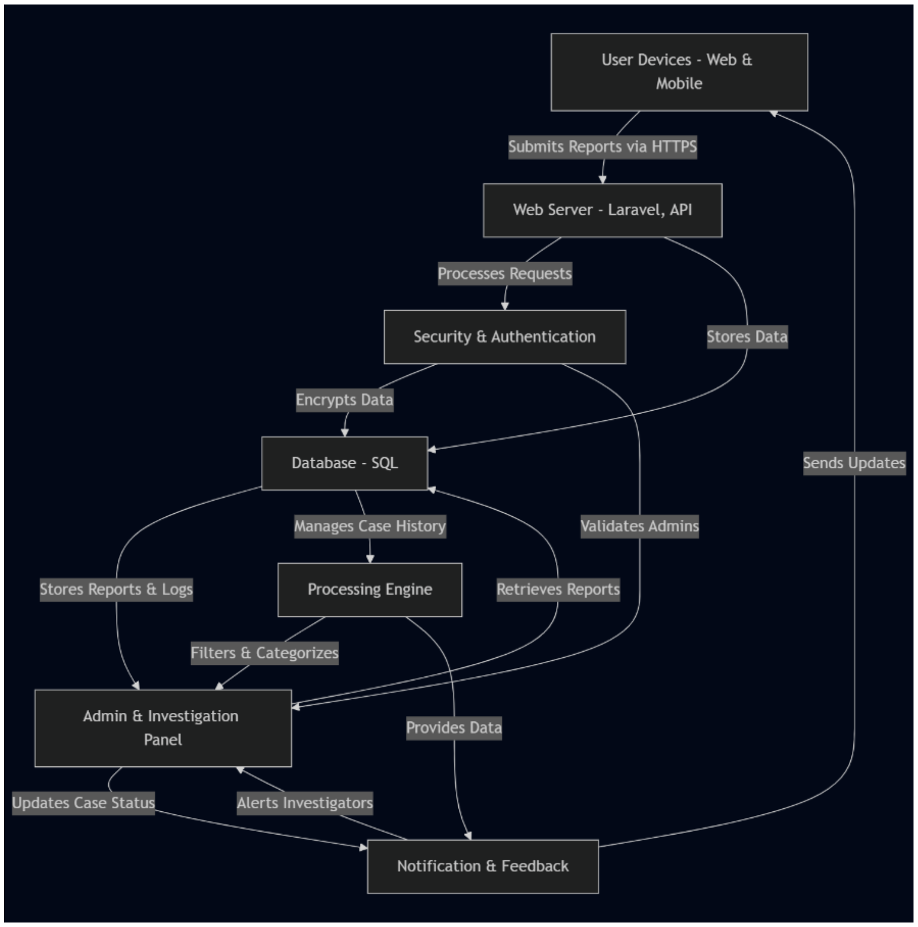
4. **Database Server**
- **Installed Components:** MySQL/PostgreSQL database for storing reports, logs, and cases
- **Interacts With:**
  - Web Server (accepting structured reports)
  - Investigation Tools (retrieving case data)
- **Deployment Location:** High-availability cloud database with backup

5. **Investigation Dashboard (Admin Access)**
   - **Installed Components:** Web-based dashboard UI for analyzing reports

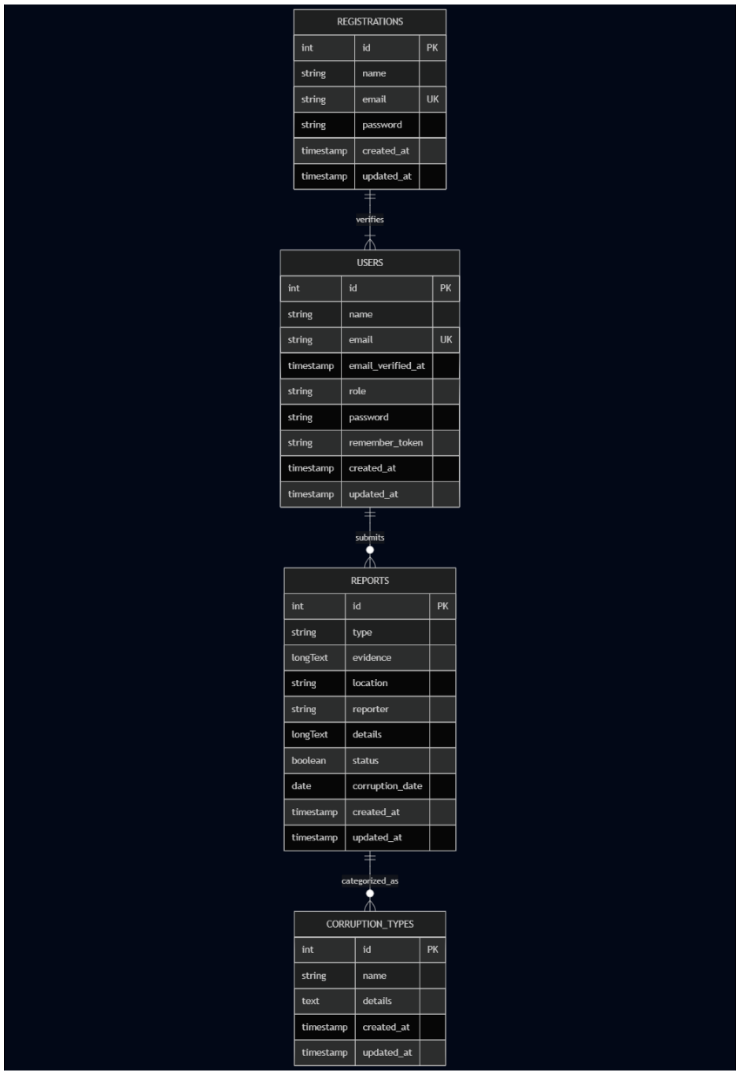
- **Interacts With:** Database Server (fetching reports), Notification System (alerting investigators)
- **Deployment Location:** Secured internal workstation or web-based secure portal

6. **Notification and Messaging System**
   - **Installed Components:** Email/SMS service integration (Twilio, SendGrid)
   - **Interacts With:**
     - Investigation System (sending status updates)
     - User Clients (feedback submission)
   - **Deployment Location:** Cloud-based communication services

Deployment Diagram



Database Design



Access Control

1. **User-based Access Control (RBAC)**
   - Different user roles define what functionality and data an actor can access.
   - Permissions are assigned based on predefined roles to ensure restricted access.

2. **User Roles and Access Rights**
   - **Access to:** Report submission form.
   - **Allowed Actions:** Submit corruption reports, attach evidence.
   - **Restricted From:** Viewing other reports, accessing admin functionalities.

3. **Investigator (Admin Level)**
   - **Access to:** Investigation dashboard, categorized reports.
   - **Allowed Actions:** View submitted reports, assign cases, update status.
   - **Restricted From:** Editing reports directly, accessing encrypted reporter details

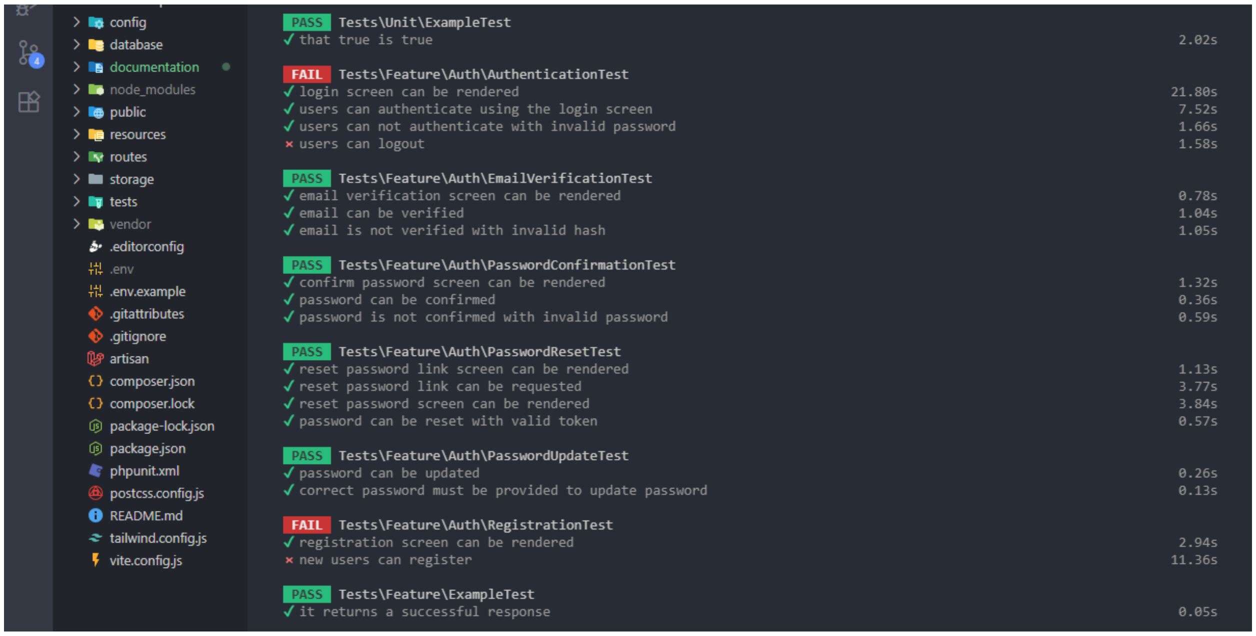
4. **System Administrator**
   - **Access to:** User management, security settings, logs.
   - **Allowed Actions:** Manage system users, configure security policies, monitor database.
   - **Restricted From:** Editing reports or interfering in investigations.

5. **Access Control Mechanisms**
   - **Authentication:** Secure login system for authorized users.
   - **Authorization:** Role-based permission enforcement.
   - **Encryption:** Sensitive data is **hashed and anonymized**.
   - **Logging & Auditing:** System activity logs track interactions.**Data Segmentation:** Reporters remain anonymous; only investigative units handle case data.

Chapter 4: Implementation

Chapter 5: Testing & Evaluation

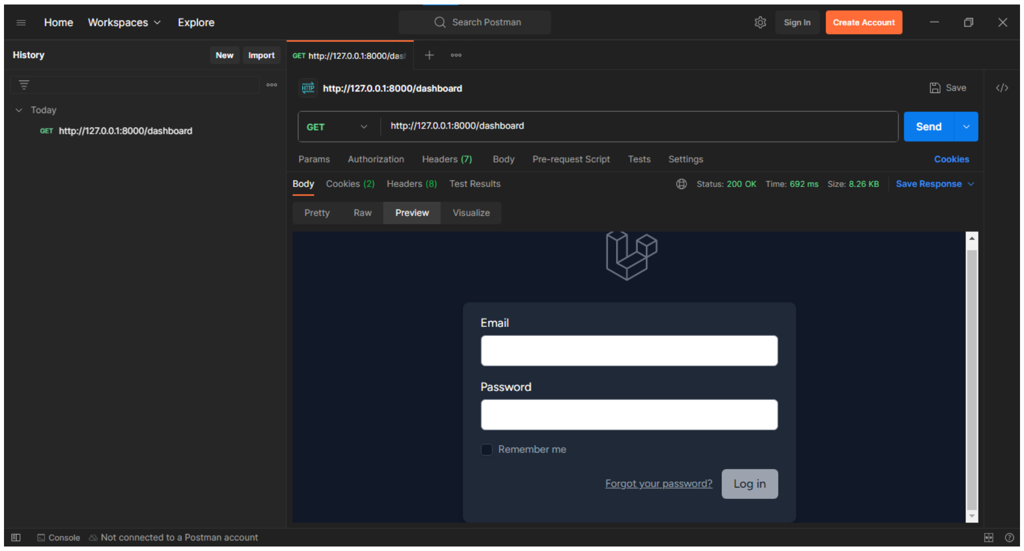
5.1. Unit Testing – Test individual Laravel components, (Controllers, Models, Validation Logic)



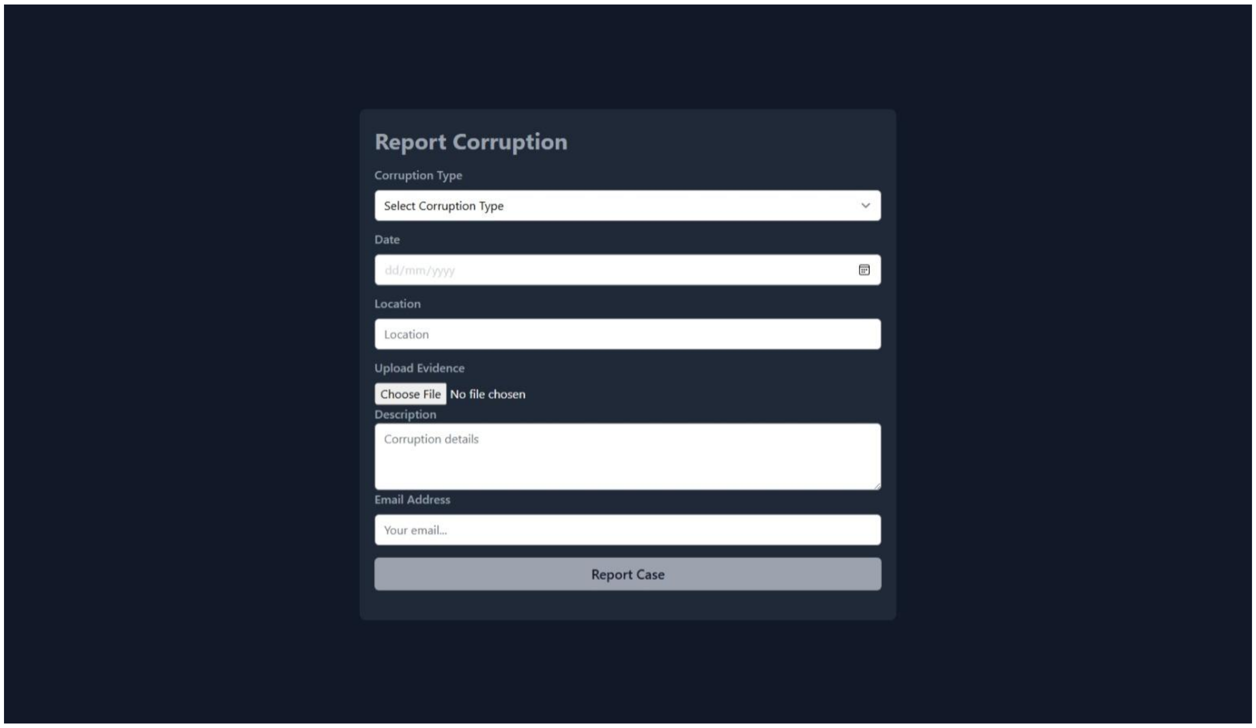
```
PASS  Tests\Unit\ExampleTest
✓ that true is true                                                          2.02s

FAIL  Tests\Feature\Auth\AuthenticationTest
✓ login screen can be rendered                                              21.80s
✓ users can authenticate using the login screen                              7.52s
✓ users can not authenticate with invalid password                           1.66s
× users can logout                                                           1.58s

PASS  Tests\Feature\Auth\EmailVerificationTest
✓ email verification screen can be rendered                                  0.78s
✓ email can be verified                                                      1.04s
✓ email is not verified with invalid hash                                    1.05s

PASS  Tests\Feature\Auth\PasswordConfirmationTest
✓ confirm password screen can be rendered                                    1.32s
✓ password can be confirmed                                                  0.36s
✓ password is not confirmed with invalid password                            0.59s

PASS  Tests\Feature\Auth\PasswordResetTest
✓ reset password link screen can be rendered                                 1.13s
✓ reset password link can be requested                                       3.77s
✓ reset password screen can be rendered                                      3.84s
✓ password can be reset with valid token                                     0.57s

PASS  Tests\Feature\Auth\PasswordUpdateTest
✓ password can be updated                                                    0.26s
✓ correct password must be provided to update password                       0.13s

FAIL  Tests\Feature\Auth\RegistrationTest
✓ registration screen can be rendered                                        2.94s
× new users can register                                                    11.36s

PASS  Tests\Feature\ExampleTest
✓ it returns a successful response                                           0.05s
```

5.2. Security Testing – Validate system's ability to use data and prevent attacks

Direct access to admin panel – Tried authenticate access via URL and was redirected to login



5.3. Usability Testing – Clean screenshot of the report form, highlighting its simpilicity and anonymity



Report Corruption

Corruption Type

Select Corruption Type

Date

dd/mm/yyyy

Location

Location

Upload Evidence

Choose File No file chosen

Description

Corruption details

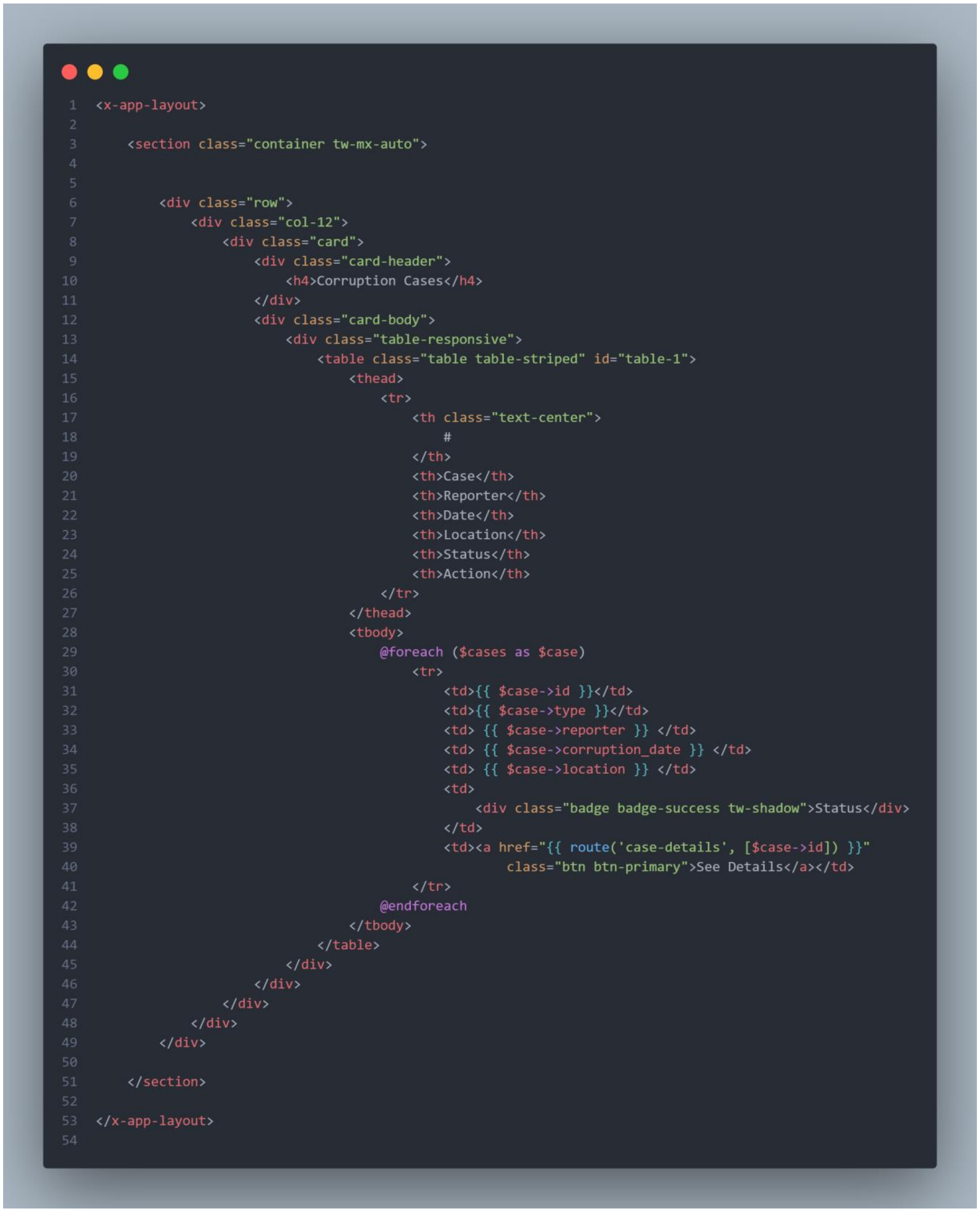
Email Address

Your email...

Report Case

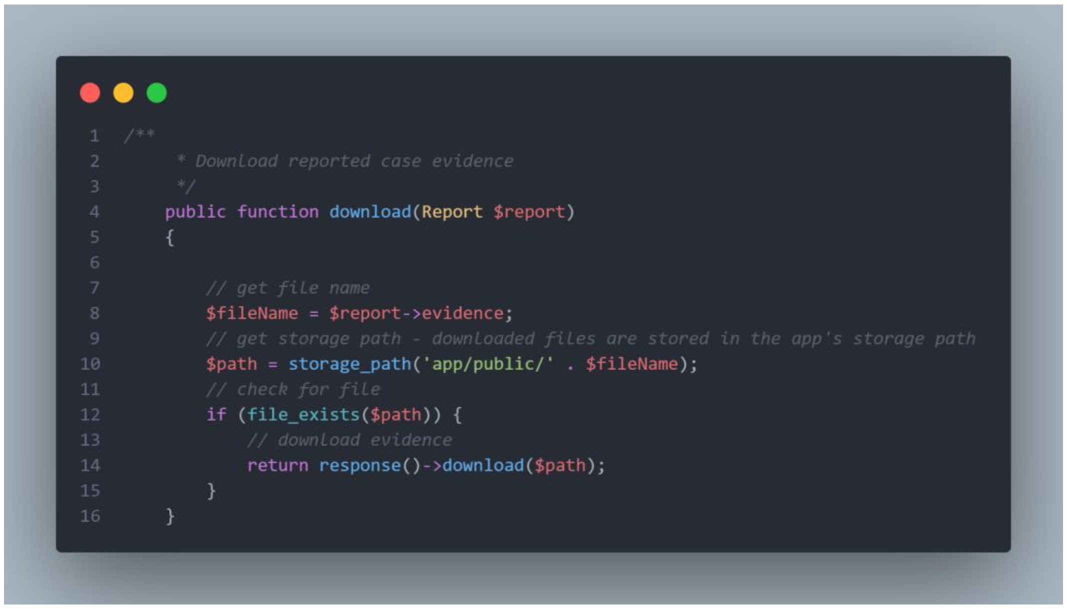
Code Samples

1. Reports table



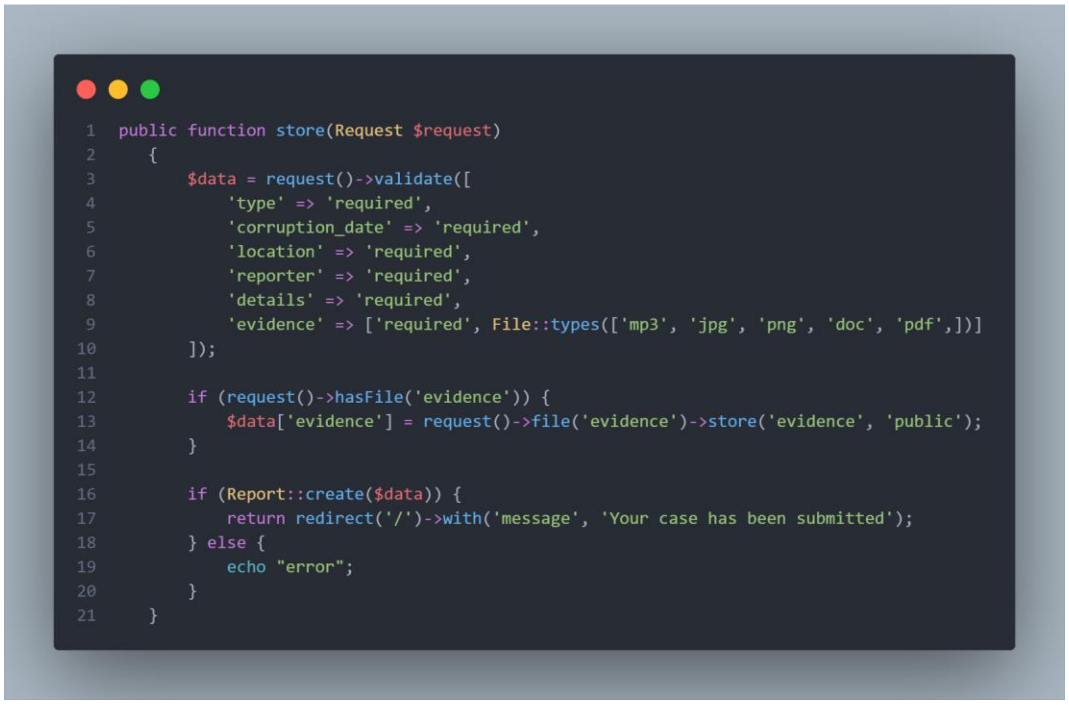
```
<x-app-layout>

    <section class="container tw-mx-auto">


        <div class="row">
            <div class="col-12">
                <div class="card">
                    <div class="card-header">
                        <h4>Corruption Cases</h4>
                    </div>
                    <div class="card-body">
                        <div class="table-responsive">
                            <table class="table table-striped" id="table-1">
                                <thead>
                                    <tr>
                                        <th class="text-center">
                                            #
                                        </th>
                                        <th>Case</th>
                                        <th>Reporter</th>
                                        <th>Date</th>
                                        <th>Location</th>
                                        <th>Status</th>
                                        <th>Action</th>
                                    </tr>
                                </thead>
                                <tbody>
                                    @foreach ($cases as $case)
                                        <tr>
                                            <td>{{ $case->id }}</td>
                                            <td>{{ $case->type }}</td>
                                            <td> {{ $case->reporter }} </td>
                                            <td> {{ $case->corruption_date }} </td>
                                            <td> {{ $case->location }} </td>
                                            <td>
                                                <div class="badge badge-success tw-shadow">Status</div>
                                            </td>
                                            <td><a href="{{ route('case-details', [$case->id]) }}"
                                                    class="btn btn-primary">See Details</a></td>
                                        </tr>
                                    @endforeach
                                </tbody>
                            </table>
                        </div>
                    </div>
                </div>
            </div>
        </div>

    </section>

</x-app-layout>

```

2. Download Evidence



```
/**
     * Download reported case evidence
     */
    public function download(Report $report)
    {

        // get file name
        $fileName = $report->evidence;
        // get storage path - downloaded files are stored in the app's storage path
        $path = storage_path('app/public/' . $fileName);
        // check for file
        if (file_exists($path)) {
            // download evidence
            return response()->download($path);
        }
    }
```

3. Submitted Reports Insertion into Database



```
public function store(Request $request)
    {
        $data = request()->validate([
            'type' => 'required',
            'corruption_date' => 'required',
            'location' => 'required',
            'reporter' => 'required',
            'details' => 'required',
            'evidence' => ['required', File::types(['mp3', 'jpg', 'png', 'doc', 'pdf',])]
        ]);

        if (request()->hasFile('evidence')) {
            $data['evidence'] = request()->file('evidence')->store('evidence', 'public');
        }

        if (Report::create($data)) {
            return redirect('/')->with('message', 'Your case has been submitted');
        } else {
            echo "error";
        }
    }
```

Chapter 6: Conclusion

The development of the Anonymous Corruption Reporting System marks a significant milestone in addressing ethical transparency within organizations. This system empowers whistleblowers to submit corruption reports safely and without fear of exposure, thereby strengthening institutional accountability.

Throughout the project, we prioritized **user anonymity, system integrity, and secure data handling**. From requirement gathering to deployment planning, the system was designed with a robust set of functional and non-functional specifications, supported by architectural models including class diagrams, use case flows, and CRC card analysis.

Security mechanisms

Such as encryption of reports, strict role-based access control, and comprehensive audit logging—were seamlessly integrated into the system's architecture. The use of modern technologies like Laravel, Tailwind CSS, and structured modeling ensured the system is not only scalable and maintainable, but also user-friendly.

Rigorous testing

Spanning unit, integration, usability, and security phases, demonstrated that the system performs effectively under different scenarios. The results, documented with screenshots and mock evaluations, provide strong evidence of the platform's dependability and responsiveness.

In conclusion, this project achieved its core objective: designing and implementing a secure, anonymous platform for reporting corruption. It also contributes to the broader field of ethical software development and civic engagement technology. With potential for future integrations (such as multilingual support or integration with government anti-corruption agencies), this system lays the groundwork for long-term institutional change.