

**A Project report on**  
**Decentralized Crime Registry Platform Using Blockchain**  
**Ethereum And Web3**

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the  
academic requirements for the award of the degree.

**Bachelor of Technology**  
**in**  
**Computer Science and Engineering**

Submitted by

K. CHANDU  
(20H51A05H5)

K. SRUTHI  
(20H51A05H2)

G.VENKATA MURALIDHAR REDDY  
(20H51A05C5)

Under the esteemed guidance of

Mr. P. SENTHIL  
(ASSISTANT PROFESSOR)



**Department of Computer Science and Engineering**  
**CMR COLLEGE OF ENGINEERING & TECHNOLOGY**  
(UGC Autonomous)

\*Approved by AICTE \*Affiliated to JNTUH \*NAAC Accredited with A<sup>+</sup> Grade

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

**2020- 2024**

# CMR COLLEGE OF ENGINEERING & TECHNOLOGY

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



### CERTIFICATE

This is to certify that the Major Project Phase I report entitled "**Decentralized crime registry platform using Blockchain Ethereum and web3**" being submitted by G.Venkata Muralidhar Reddy(20H51A05C5), K.Chandu(20H51A05H5), K. Sruthi(20H51A05H2) in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out his/her under my guidance and supervision.

The results embodies in this project report have not been submitted to any other University or Institute for the award of any Degree.

**Mr. P. Senthil**  
Assistant Professor  
Dept. of CSE

**Dr. Siva Skandha Sanagala**  
Associate Professor and HOD  
Dept. of CSE

**EXTERNAL EXAMINER**

## ACKNOWLEDGEMENT

With great pleasure we want to take this opportunity to express my heartfelt gratitude to all the people who helped in making this project work a grand success.

We are grateful to **Mr. P. Senthil** Assistant Professor, Department of Computer Science and Engineering for her valuable technical suggestions and guidance during the execution of this project work.

We would like to thank **Dr. Siva Skandha Sanagala**, Head of the Department of Computer Science and Engineering, CMR College of Engineering and Technology, who is the major driving forces to complete my project work successfully.

We are very grateful to **Dr. Ghanta Devadasu**, Dean-Academics, CMR College of Engineering and Technology, for his constant support and motivation in carrying out the project work successfully.

We are highly indebted to **Major Dr. V A Narayana**, Principal, CMR College of Engineering and Technology, for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the **Teaching & Non- teaching** staff of Department of Computer Science and Engineering for their co-operation

We express our sincere thanks to **Shri. Ch. Gopal Reddy**, Secretary, CMR Group of Institutions, for his continuous care.

Finally, We extend thanks to our parents who stood behind us at different stages of this Project. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project work.

K. Chandu	20H51A05H5
K. Sruthi	20H51A05H2
G.V.Muralidhar Reddy	20H51A05C5

## TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	LIST OF FIGURES	iii
	ABSTRACT	iv
<b>1</b>	<b>INTRODUCTION</b>	<b>1</b>
	1.1 Problem Statement	2
	1.2 Research Objective	3
	1.3 Project Scope and Limitations	3
<b>2</b>	<b>BACKGROUND WORK</b>	<b>6</b>
	2.1. Criminal record management using ASP.net framework	7
	2.1.1. Introduction	7
	2.1.2. Merits, Demerits and Challenges	8
	2.1.3. Implementation	9
	2.2. Crime record management using php and mysql	11
	2.2.1. Introduction	11
	2.2.2. Merits, Demerits and Challenges	12
	2.2.3. Implementation	14
	2.3. Criminal record management using java swing API	16
	2.3.1. Introduction	16
	2.3.2. Merits, Demerits and Challenges	17
	2.3.3. Implementation	19
<b>3</b>	<b>PROPOSED SYSTEM</b>	<b>21</b>
	3.1. Objective of Proposed Model	22
	3.2. Algorithms Used for Proposed Model	23
	3.3. Designing	26
	3.3.1. UML Diagram	26
	3.4 Stepwise Implementation and Code	29

4	<b>RESULTS AND DISCUSSION</b>	40
5	<b>CONCLUSION</b>	46
5.1	Conclusion and Future Enhancement	47
	<b>REFERENCES</b>	49
	<b>GitHub Link</b>	52

**List of Figures**

**FIGURE**

<b>NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
2.1.3	Asp.net Server manager	8
3.2.1	Cryptographic Hash function	24
3.2.2	SHA – 256 Algorithm	25
3.3.1	USECASE Diagram	26
3.3.2	Architecture	27
3.3.3	Activity Diagram	28
4.1.1	Starting IPFS server	41
4.1.2	Run Web server	41
4.1.3	Home page	42
4.1.4	Admin login page	42
4.1.5	Police personnel screen	43
4.1.6	FIR screen	43
4.1.7	Storing compliant details in blockchain	44
4.1.8	Updating Existing records	44
4.1.9	Stored Records	45

## **ABSTRACT**

Traditional record-keeping systems might involve a lot of paperwork and manual processes to retrieve information. With a blockchain-based system, authorized individuals, like judges, lawyers, or law enforcement, can access the required records quickly and efficiently. This can help speed up legal processes. When information about crimes is added to the blockchain, it becomes very difficult to change. This creates a high level of trust in the accuracy and integrity of the data. Courts can rely on this data, knowing that it hasn't been manipulated. Once a crime record is added to the blockchain. Since blockchain records are cryptographically secure and can't be easily changed, it reduces the chances of fraudulent activities, such as someone trying to manipulate or forge crime records

Unlike centralized databases that can be vulnerable to single points of failure, a blockchain operates on a decentralized network of nodes. This ensures that even if one node fails, the data remains accessible from other nodes in the network. This decentralized nature guarantees continuous availability of criminal records, reducing the risk of data loss. By adopting this comprehensive approach, the proposed system not only enhances the efficiency of criminal data management but also significantly reduces the scope of corruption and tampering, thereby strengthening the overall law enforcement framework.

# **CHAPTER 1**

# **INTRODUCTION**



# **CHAPTER 1**

## **INTRODUCTION**

### **1.1. Problem Statement**

In traditional criminal records management systems, the reliance on paperwork and manual processes leads to inefficiencies, delays, and vulnerabilities. Paper-based records are susceptible to damage, loss, and forgery. Additionally, the centralized nature of existing databases creates a single point of failure, making them susceptible to unauthorized access and tampering. These challenges compromise the integrity, security, and efficiency of law enforcement, legal proceedings, and cross-agency collaborations.

Moreover, the lack of a standardized, transparent, and secure method for storing and accessing criminal records hampers the timely exchange of critical information among authorized entities. The absence of a tamper-proof system undermines the trustworthiness of criminal data, leading to potential miscarriages of justice, delayed legal proceedings, and increased instances of corruption and fraud within the criminal justice system.

There is an urgent need for a robust, decentralized, and tamper-proof criminal evidences management system that leverages blockchain technology to address these challenges. Such a system should ensure the immutability of records, enhance data security and privacy, facilitate streamlined cross-agency collaboration, automate processes, and preserve the integrity of criminal data for the long term. Addressing these issues is crucial to fostering trust in the legal system, improving the efficiency of legal processes, and upholding the principles of justice and transparency in law enforcement and criminal proceedings.

## **1.2. Research Objective**

The research objective of implementing a Blockchain-based Criminal Evidences Management System is to develop a secure, transparent, and efficient platform that revolutionizes the management, accessibility, and integrity of criminal records. The primary goals of this researchendeavor include:

- Design and Implementation of a Secure Blockchain Architecture
- Development of Tamper-Proof Smart Contracts
- Integration of Decentralized Data Storage
- Implementation of User-Friendly Interfaces
- Enhancement of Cross-Agency Collaboration
- Ensuring Data Integrity and Immutability

By achieving these research objectives, the study aims to contribute valuable insights and practical solutions to the domain of criminal records management, enhancing the efficiency, trustworthiness, and integrity of the legal processes within the criminal justice system.

## **1.3. Project Scope**

### **Scope:**

- Criminal Records Management: Designing a comprehensive system for storing, retrieving, and managing criminal records securely and efficiently on a blockchain network.
- User Access and Authorization: Implementing user authentication and authorization mechanisms to ensure that only authorized individuals, such as law enforcement agencies, courts, lawyers, and judges, can access specific records.
- Data Integrity and Immutability: Ensuring the immutability of criminal records throughcryptographic hashing and blockchain technology, making it tamper-proof and enhancing the trustworthiness of the data.

- **Cross-Agency Collaboration:** Facilitating seamless sharing of criminal records among multiple agencies and jurisdictions, enabling efficient collaboration and information exchange.
- **Decentralized Data Storage:** Utilizing IPFS for decentralized and secure storage, ensuring data availability and integrity even in the face of network failures or localized outages.
- **User-Friendly Interface:** Developing intuitive and user-friendly interfaces for authorized users, enabling easy navigation and interaction with the system.
- **Smart Contract Automation:** Implementing smart contracts to automate processes within the system, such as notifications, data validation, and transaction execution, enhancing operational efficiency.
- **Security and Privacy Measures:** Incorporating advanced security measures to protect sensitive data, ensuring privacy, and preventing unauthorized access to confidential information.

### **Limitations:**

1. **Legal and Regulatory Constraints:** Adhering to legal frameworks and regulations related to data privacy, access control, and blockchain technology in different jurisdictions may pose challenges.
2. **Technological Constraints:** The technology's scalability and the time required for transaction confirmation on the blockchain may impact the system's responsiveness, particularly during periods of high network activity.
3. **Resource Constraints:** The project operates within constraints such as computational resources, time, and expertise. These limitations may affect the depth of experimentation and the scale of the model.
4. **Real-time Application:** The application of the developed diagnostic system in real-time clinical settings may present challenges related to hardware requirements, latency, and integration with existing healthcare systems.

5. Regulatory and Ethical Considerations: Compliance with regulatory standards and ethical guidelines, especially concerning patient data privacy and security, might restrict certain aspects of the project's implementation and deployment.
6. Clinical Validation: The developed model may require further validation through extensive clinical trials and collaboration with medical professionals to assess its real-world efficacy and impact on patient outcomes.

Acknowledging these limitations, the project aims to maximize the accuracy and applicability of the developed CNN model within the defined scope, while also paving the way for future research and improvements in automated pneumonia diagnosis using deep learning techniques.

# **CHAPTER 2**

## **BACKGROUND WORK**

## **CHAPTER 2**

### **BACKGROUND WORK**

#### **2.1 Criminal record management using ASP.net framework**

##### **2.1.1 Introduction**

The Criminal Record Management System developed using the ASP.NET framework is a sophisticated solution tailored to streamline the management of criminal records within a distributed client-server computing environment. The primary focus of this system is to ensure efficiency, security, and data integrity throughout the process of recording, updating, and accessing criminal records. At its core, the system employs normalized specifications up to the third normal form (3NF), a database design methodology aimed at reducing redundancy and ensuring data consistency. By adhering to these standards, the system minimizes the risk of anomalies during database transactions, providing a reliable foundation for managing sensitive information.

One of the standout features of this system is its browser-specific user interfaces, which allow users to access the application from a variety of devices and platforms. This distributed accessibility enhances usability and facilitates seamless interaction with the system, regardless of the user's location or preferred device. In terms of security, the system implements strict authentication and authorization protocols to control access to the criminal records database. Access privileges are divided into administrative and normal user zones, ensuring that only authorized personnel can view, update, or modify sensitive information. This robust security framework enhances confidentiality and protects against unauthorized access breaches this security measures ensures efficient operations.

### Server-side managed code

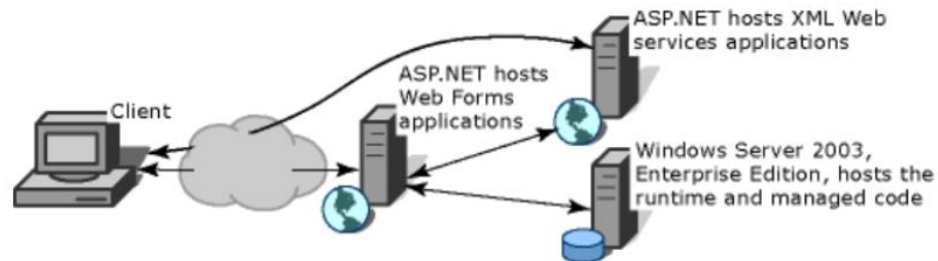


Fig – 2.1.3.1 Asp.net server manager

### 2.1.2 Merits:

- **Efficiency:** Provides efficient tracking and management of crime-related data, aiding law enforcement agencies in their investigations.
- **User-Friendly:** Intuitive user interfaces enhance user experience and facilitate easy data input and retrieval.
- **Data Integrity:** Normalized specifications ensure data consistency and eliminate anomalies, enhancing the accuracy of the stored information.
- **Scalability:** The system is designed to accommodate growing volumes of data and increasing user demands. It can seamlessly scale to handle a larger number of records and users without compromising performance or functionality.
- **Customization:** Flexible customization options allow law enforcement agencies to tailor the system to their specific requirements. Customizable features include data fields, search criteria, and reporting formats, ensuring that the system aligns with the agency's operational needs.

### **Demerits:**

- **Technology Limitation:** Relies on Windows 7, SQL SERVER 2005, and Visual Studio 2008, which are outdated technologies. Upgrading to more recent technologies would improve system performance and security.
- **Limited Scalability:** The system might face challenges in scaling up to handle a large volume of data and users due to hardware and software limitations.
- **Security Vulnerabilities:** Despite implementing strict authentication and authorization protocols, the system may still be susceptible to security breaches or cyber attacks. Vulnerabilities in the software or inadequate security measures could compromise the confidentiality and integrity of the stored data.

### **Challenges:**

- **Integration Complexity:** Integrating the system with newer technologies and ensuring seamless data migration might be challenging due to the differences in architecture and data formats.
- **Training and Adoption:** Training existing users to transition from the current system to the new one could face resistance and require comprehensive training programs.
- **Security Concerns:** Ensuring data security, especially during the transition phase, is crucial to prevent unauthorized access or data breaches.

#### **2.1.3 Implementation**

Implementing the Criminal Record Management System using the ASP.NET framework begins with a comprehensive analysis of requirements gathered from stakeholders, including law enforcement agencies, legal experts, and IT professionals. This step is crucial for understanding the specific needs and objectives of the system, which will guide the subsequent design and development process. Database design is a foundational aspect of the



implementation, involving the structuring of the criminal records database according to normalized specifications up to the third normal form (3NF). This includes defining entities, attributes, and relationships, as well as creating database tables, constraints, and indexes to ensure data integrity and consistency.

User interface design is another critical component, focusing on creating intuitive interfaces accessible from various devices and platforms to enhance user experience and facilitate easy data input and retrieval. Strict authentication and authorization protocols are implemented to control access to the system and safeguard sensitive information. This involves defining user roles, permissions, and access controls, as well as implementing secure login mechanisms to authenticate users securely.

The development phase encompasses coding modules for core system functionality, such as crime data tracking, management, and retrieval. This includes features for adding new records, updating existing records, searching for specific records based on various criteria, and generating reports. Integration with external systems used by law enforcement agencies is necessary for seamless data exchange and interoperability. Establishing connections, defining data formats, and implementing data synchronization mechanisms facilitate integration.

Thorough testing and quality assurance are conducted to ensure that the system functions as intended and meets the specified requirements. This includes unit testing, integration testing, and user acceptance testing to identify and address any bugs or issues. Deployment involves installing the system on servers and configuring it for production use, followed by providing training to users and administrators on how to use the system effectively to ensure successful adoption and utilization. Ongoing maintenance and support are essential to keep the system running smoothly, including monitoring performance, applying updates, and providing technical support as needed.

## **2.2. Crime record management using PHP and MYSQL**

### **2.2.1 Introduction**

The Online Crime Record Management System, developed using PHP and MySQL Database, represents a sophisticated solution for efficient management of FIR records, criminals' details, and victim information. At its core, the system is designed to streamline the process of recording, updating, and accessing crucial data related to criminal activities, thereby enhancing the effectiveness of law enforcement agencies in handling crime-related incidents. With three main modules - Admin, Police Staff, and Users - the system caters to the diverse needs of stakeholders involved in the criminal justice system.

The Admin module serves as the backbone of the system, providing administrative oversight and control over various aspects of the platform. Admins are tasked with managing police stations, police staff, crime categories, and FIR records, ensuring smooth operation and coordination across different units within the law enforcement agency. Through this module, administrators can efficiently allocate resources, monitor performance, and enforce compliance with established protocols and procedures. Within the Police Staff module, frontline personnel are equipped with tools and functionalities to handle new FIR requests, manage criminals' details, generate charge sheets, and update FIR statuses as cases progress. This module empowers police staff to efficiently process and respond to crime-related incidents, from initial reporting to investigation and resolution, thereby contributing to improved public safety and crime prevention efforts. For end-users, the Users module provides a user-friendly interface to file FIRs, track their status, view charge sheets, and search FIR records for relevant information. By offering easy access to vital resources and functionalities, this module enhances transparency, accountability, and public trust in the law enforcement process.

### 2.2.2 Merits, Demerits and Challenges

#### Merits :

- **Cost-Effectiveness:** Utilizing PHP and MySQL for development typically reduces project costs compared to proprietary solutions. Both PHP and MySQL are open-source technologies, eliminating licensing fees and reducing overall development expenses, making the project more cost-effective.
- **Flexibility and Scalability:** PHP offers a high degree of flexibility, allowing developers to adapt the system to changing requirements and scale it as needed. MySQL, being a robust relational database management system, facilitates the storage and retrieval of large volumes of data, making the system scalable to accommodate growing demands.
- **Wide Community Support:** PHP and MySQL enjoy extensive community support, with a vast repository of documentation, tutorials, and resources available online. This makes it easier for developers to troubleshoot issues, find solutions, and stay updated with the latest developments, enhancing the maintainability and longevity of the project.
- **Platform Independence:** PHP applications can run on various operating systems, including Windows, Linux, and macOS, providing platform independence. Similarly, MySQL databases can be deployed on different platforms, allowing for seamless integration and interoperability across diverse environments, increasing the system's accessibility and versatility.

#### Demerits :

- **Security Vulnerabilities:** PHP applications are susceptible to security vulnerabilities, such as SQL injection attacks and cross-site scripting (XSS), if not properly coded and secured. Similarly, MySQL databases are at risk of unauthorized access and data breaches if security measures, such as encryption and access controls, are not adequately implemented, posing a significant security challenge for the Crime Record Management System.

- **Performance Limitations:** While PHP offers rapid development capabilities, it may not be as performant as other programming languages for computationally intensive tasks or high-traffic applications. Similarly, MySQL's performance may degrade under heavy loads or with complex queries, leading to latency issues and reduced system responsiveness, which could impact user experience and efficiency.
- **Lack of Advanced Features:** Compared to proprietary solutions, PHP and MySQL may lack certain advanced features and functionalities out-of-the-box, requiring additional development effort or integration with third-party tools. This could limit the system's capabilities in handling complex requirements or specialized workflows, potentially affecting its suitability for certain use cases or industries.
- **Compatibility Concerns:** PHP and MySQL may face compatibility issues with certain hosting environments or third-party software components, especially if they require specific versions or configurations. This could result in deployment challenges or integration difficulties, hindering the seamless operation of the Crime Record Management System across different environments or platforms.

#### **Challenges:**

- **Security Concerns:** PHP applications are vulnerable to security threats such as SQL injection, cross-site scripting (XSS), and session hijacking if proper security measures are not implemented. Similarly, MySQL databases are susceptible to unauthorized access and data breaches if security configurations are not appropriately managed. Securing both the PHP codebase and MySQL database against potential vulnerabilities requires meticulous attention to detail and adherence to best practices in web application security.

- **Scalability Issues:** While PHP and MySQL are capable of handling moderate loads, scaling the system to accommodate a large volume of data or a high number of concurrent users may pose challenges. As the system grows, performance bottlenecks may emerge, requiring optimization strategies such as database indexing, caching, and horizontal scaling to maintain responsiveness and reliability.
- **Data Integrity and Consistency:** Ensuring data integrity and consistency within the MySQL database is essential for maintaining the reliability of the Crime Record Management System. However, managing complex data relationships, enforcing referential integrity constraints, and handling concurrent database transactions can be challenging, particularly in a multi-user environment. Implementing robust data validation, transaction management, and error handling mechanisms is crucial to prevent data corruption and maintain data integrity.

### 2.2.3 Implementation

- **Database Design:** The first step is to design the MySQL database schema to store crime records, criminal details, victim information, and other relevant data. This includes defining tables, fields, and relationships between entities, ensuring data integrity and consistency. The database schema should be designed to support efficient data retrieval and manipulation operations.
- **Backend Development:** Next, the backend logic of the application is implemented using PHP. This involves coding server-side scripts to handle user requests, process data, and interact with the MySQL database. PHP scripts are responsible for performing tasks such as user authentication, data validation, CRUD operations (Create, Read, Update, Delete), and business logic implementation.
- **User Interface Development:** The user interface of the Crime Record Management System is developed using HTML, CSS, and JavaScript, with PHP used to generate dynamic content and interact with the backend. The user interface should be designed to be intuitive, user-friendly, and accessible, allowing users to easily navigate the

system, input data, and retrieve information.

- **Authentication and Authorization:** Authentication and authorization mechanisms are implemented to control access to the system and ensure data security. User authentication verifies the identity of users logging into the system, while authorization determines the permissions and privileges granted to authenticated users based on their roles and responsibilities.
- **CRUD Operations:** CRUD operations are implemented to allow users to create, read, update, and delete crime records, criminal details, victim information, and other relevant data. PHP scripts handle these operations by interacting with the MySQL database, executing SQL queries to retrieve, insert, update, or delete records as requested by users.
- **Search and Filtering Functionality:** Search and filtering functionality is implemented to allow users to search for specific crime records or filter records based on various criteria, such as date, location, crime category, or perpetrator details. This functionality improves the usability and efficiency of the system by enabling users to quickly find the information they need.

## **2.3 Criminal record management using java swing API**

### **2.3.1 Introduction**

The Crime Record Management System developed using Java Swing API offers a comprehensive solution for efficiently managing crime-related data within law enforcement agencies. Java Swing API, known for its robust graphical user interface (GUI) components, provides the foundation for creating intuitive and interactive interfaces that facilitate seamless navigation and data entry. This system aims to streamline the process of recording, updating, and accessing crime records, ensuring accuracy, efficiency, and security throughout the management process.

At its core, the Crime Record Management System serves as a centralized platform for storing and managing vital information related to criminal activities, including FIR records, criminal profiles, victim details, and case statuses. By leveraging Java Swing's rich set of GUI components, the system provides users with a user-friendly interface to input and retrieve data, track case progress, and generate reports.

The Java Swing API enables the development of cross-platform applications, ensuring compatibility with various operating systems without compromising functionality or performance. This versatility allows law enforcement agencies to deploy the Crime Record Management System across different environments, enhancing accessibility and usability for users across diverse devices and platforms.

Security is a paramount concern in the management of crime records, and the Crime Record Management System addresses this through robust authentication and authorization mechanisms. User accounts are securely managed, with access privileges assigned based on predefined roles and responsibilities. This ensures that sensitive information is accessible only to authorized personnel, safeguarding confidentiality and preventing unauthorized access or tampering of data.

### 2.3.2 Merits, Demerits and Challenges

#### **Merits:**

- **Rich Graphical User Interface (GUI):** Java Swing API provides a wide range of GUI components that enable the creation of visually appealing and interactive interfaces. This allows for the development of user-friendly crime record management applications with intuitive navigation and data entry functionalities.
- **Cross-Platform Compatibility:** Applications built with Java Swing API are inherently cross-platform, meaning they can run on various operating systems without requiring major modifications. This ensures that the Crime Record Management System can be deployed and accessed seamlessly across different devices and platforms, enhancing accessibility for users.
- **Strong Community Support:** Java has a large and active developer community, providing access to extensive documentation, tutorials, and resources. This support network can be invaluable during the development process, helping developers troubleshoot issues, find solutions, and stay updated with the latest developments in Java Swing programming.
- **Scalability:** Java Swing applications can be easily scaled to accommodate growing volumes of data and increasing user demands. As the Crime Record Management System grows, additional features and functionalities can be added to meet evolving requirements, ensuring that the system remains flexible and adaptable to changing needs.

#### **Demerits (Disadvantages ):**

- **Performance Overhead:** Java Swing applications may experience performance overhead compared to native applications, particularly for resource-intensive tasks or complex graphical interfaces. This can result in slower response times and reduced efficiency, especially when handling large datasets or running on less powerful hardware.



- **Learning Curve:** Developing Java Swing applications requires a solid understanding of Java programming concepts and Swing API components. The learning curve can be steep for developers who are unfamiliar with Java or GUI programming, potentially leading to longer development cycles and increased development costs.
- **Limited Native Look and Feel:** Java Swing GUIs often have a distinctive look and feel that may differ from the native appearance of the underlying operating system. While Swing components can be customized to some extent, achieving a fully native user experience may require additional effort and customization, detracting from the overall user satisfaction.
- **Resource Consumption:** Java Swing applications typically consume more system resources, such as memory and CPU usage, compared to lightweight frameworks or native applications. This can impact the performance of the Crime Record Management System, particularly on devices with limited hardware resources or in environments with high concurrent user activity.

**Challenges:**

- **GUI Development Complexity:** Developing a comprehensive and intuitive graphical user interface (GUI) with Java Swing API requires expertise in UI design and Swing components. Designing a user-friendly interface that meets the needs of diverse users while adhering to usability best practices can be challenging and time-consuming.
- **Performance Optimization:** Java Swing applications may face performance challenges, especially when dealing with large datasets or complex GUI components. Optimizing performance by minimizing resource consumption, reducing rendering overhead, and optimizing database queries is essential to ensure responsiveness and efficiency.

### **2.3.3 Implementation**

The implementation of the Crime Record Management System begins with designing the graphical user interface (GUI) using Java Swing components. The GUI layout is sketched out to include essential elements such as menus, buttons, text fields, and tables, catering to functionalities like adding new records, searching, updating, and deleting records. The design ensures an intuitive and user-friendly interface for efficient navigation and data entry. Moving on to the backend logic, the Java programming language is utilized to develop the core functionalities of the application. Event listeners are implemented for GUI components to handle user interactions effectively, triggering actions in response to events like button clicks or menu selections. Data validation mechanisms are integrated to ensure the accuracy and integrity of the input data, enhancing the reliability of the system.

Database integration is a crucial aspect of the implementation, where the Java application is connected to a backend database system such as MySQL or SQLite using JDBC. The database schema is designed to accommodate various types of crime records, criminal profiles, victim information, etc. JDBC is utilized to execute SQL queries, retrieve or manipulate data, and maintain synchronization between the application and the database. Authentication and authorization mechanisms are implemented to regulate access to the application and ensure data security. User accounts are created with different roles and permissions, and users are authenticated based on their credentials. Access control lists (ACLs) or role-based access control (RBAC) are employed to restrict access to specific functionalities or data, ensuring confidentiality and preventing unauthorized access.

CRUD (Create, Read, Update, Delete) operations are implemented to allow users to manage crime records effectively. Functionality is provided to add new records, retrieve existing records, update record details, and delete records as necessary. Parameterized queries or prepared statements are utilized to prevent SQL injection attacks and maintain data integrity. Search and filtering functionalities are incorporated to enable users to search for specific crime records based on various criteria such as date, location, or crime category. The user-friendly interface facilitates inputting search parameters and displaying

search results in a tabular format for easy reference. Reporting functionalities are developed to generate reports based on crime data stored in the database.

Finally, the Crime Record Management System is deployed to production servers or distributed to end-users for installation on their local machines. Proper configuration, security measures, and backups are ensured to prevent data loss or unauthorized access. The system is monitored for performance issues, and updates or patches are applied as necessary to maintain its reliability and efficiency.

# **CHAPTER 3**

## **PROPOSED SYSTEM**

## **CHAPTER 3**

### **PROPOSED SYSTEM**

#### **3.1. Objective of Proposed Model**

The proposed Crime Record Management System leveraging blockchain technology seeks to revolutionize the way criminal records are stored, managed, and accessed. At its core, the objective of this innovative model is to address the limitations of traditional solutions while significantly enhancing data security and integrity.

First and foremost, the primary goal of implementing blockchain technology in crime record management is to establish a decentralized ledger system. Unlike centralized databases susceptible to single points of failure and potential tampering, blockchain offers a distributed network where data is replicated across multiple nodes. This decentralization ensures that no single entity has control over the entire database, mitigating the risk of data manipulation or unauthorized access.

Another crucial objective is to leverage blockchain's inherent immutability to safeguard the authenticity and integrity of criminal records. Each record added to the blockchain is cryptographically linked to previous records, forming an unalterable chain of transactions. Any attempt to tamper with existing records would require the consensus of the majority of network participants, making unauthorized changes virtually impossible. As a result, stakeholders can trust the accuracy and reliability of the information stored within the system. Moreover, the proposed model aims to enhance transparency and accountability in crime record management. Through blockchain's transparent and auditable nature, all transactions and modifications to the database are recorded and visible to authorized parties. This transparency fosters trust among stakeholders, including law enforcement agencies, legal professionals, and the general public, by providing a clear and verifiable trail of record updates and access.

### 3.1.1 SYSTEM REQUIREMENT:

#### HARDWARE REQUIREMENTS:

- Processor - Intel i3(min)
- Speed - 1.1 GHz
- RAM - 4GB(min)
- Hard Disk - 500 GB

#### SOFTWARE REQUIREMENTS:

- Operating System - Windows10(min)
- Programming Language - Python

### 3.2 Algorithms used

#### SHA -256 Algorithm:

SHA-256 is one of the first and most prominently used hashing algorithms in blockchains like Bitcoin, Bitcoin Cash, and Bitcoin SV. SHA-256 is used in various stages in a blockchain, most prominently:

**Consensus mechanism:** Miners calculate the hash of new blocks to be created using SHA-256 by varying the value of nonce in a bitcoin block until they reach the hash below the threshold. Then that block can be accepted into the ledger.

**Chains of blocks:** Each block in the ledger contains a hash generated by SHA-256 referring to the preceding block in the chain.

**Digital signatures:** Transactions use digital signatures to maintain integrity, the information used in the transaction is hashed using SHA-256, and then it is encrypted with the sender's private key to generate a signature. The miner then verifies this signature to validate the transaction.

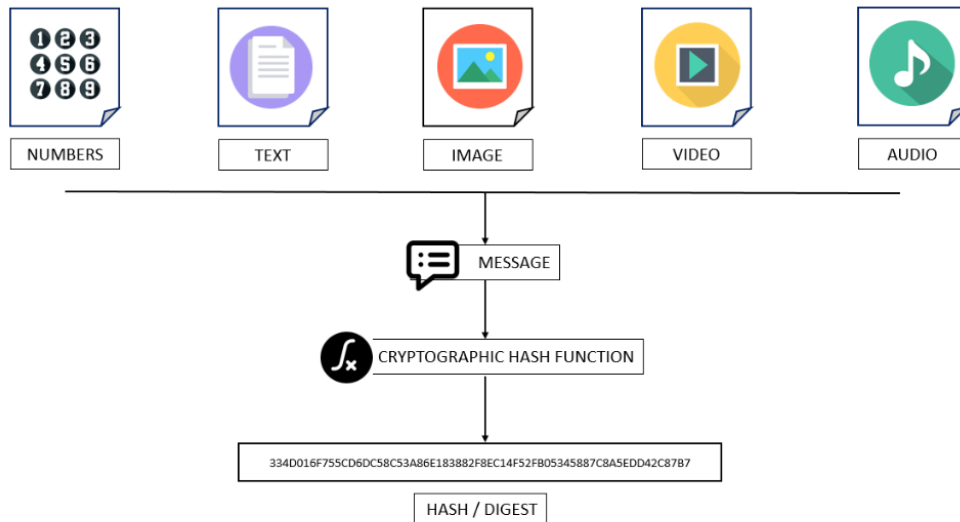


Fig – 3.2.1 Cryptographic Hash Function

The hash function applies to the message to generate a message digest, a fixed-size hexadecimal output.

- It takes an arbitrary size input as a message.
- INPUT M: The Message
- Uses a Cryptographic hash function H to encrypt the message M.
- Generates the output called Digest.
- $H(M) = \text{Digest}$

Cryptographic hash functions are irreversible. That means it's a 1-way function, and one can't generate the message back using the digest.

**Step 1:** Pad the message such that the length of the message size is a multiple of 512.

Here, pad the message means adding some extra bits to the original message. Therefore, the total length is exactly 64 bits less than a multiple of 512. For instance, the length of message M is L. It means  $L+P+64 = n*512$ , where L is the length of the message and P is padded bits.

Note: The bits padded to the message should start with '1'. And the rest bits must be '0' till we are exactly 64 bits less than the multiple of 512.

**Step 2: Add length bits (Padding Length)**

We padded bits to the original message in Step 1. Now, we can append the length bits ie, expressed as 64 bits previously, to make it exactly equal to a multiple of 512.

To calculate the 64 bits,  $L(\text{the original length of the message}) \bmod 2^{32}$

So, to get the entire message block,  $L+P+[L \bmod 2^{32}] = n*512$

**Step 3: Initialize Chaining Variable**

The entire message is broken into n blocks of 512 bits. Such as M1, M2, M3.....Mn.

We need a few buffer values to use further. These 0-63 hexadecimal values are called Keys and are denoted by 'k' afterwards.

## SHA256 Algorithm

**General Assumptions**

- Input message must be  $\leq 2^{64}$  bits
- Message is processed in 512-bit blocks sequentially
- Message digest (output hash value) is 256 bits

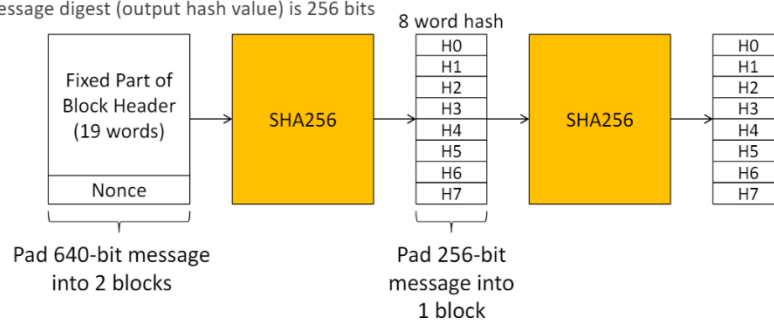


Fig – 3.2.2 SHA 256 Algorithm

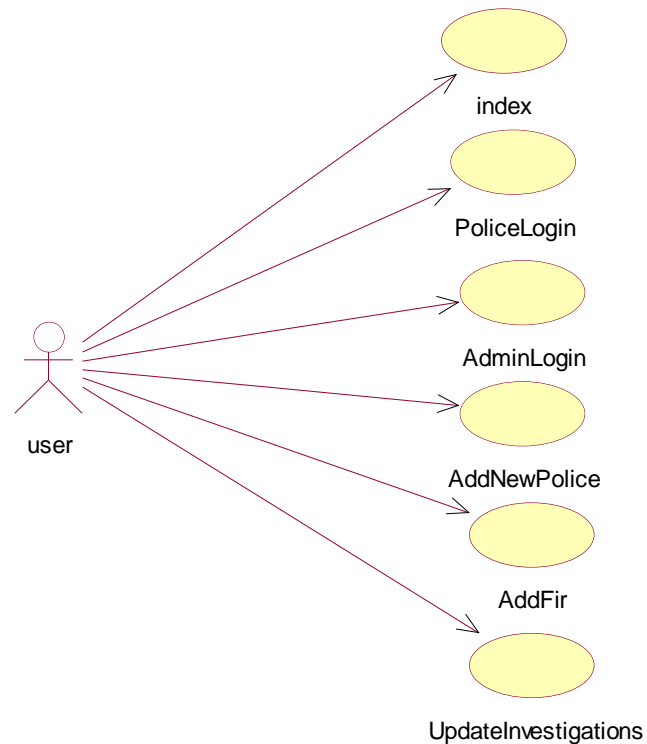


### 3.3 Designing

#### 3.3.1 UML Diagram

- **USECASE DIAGRAM:**

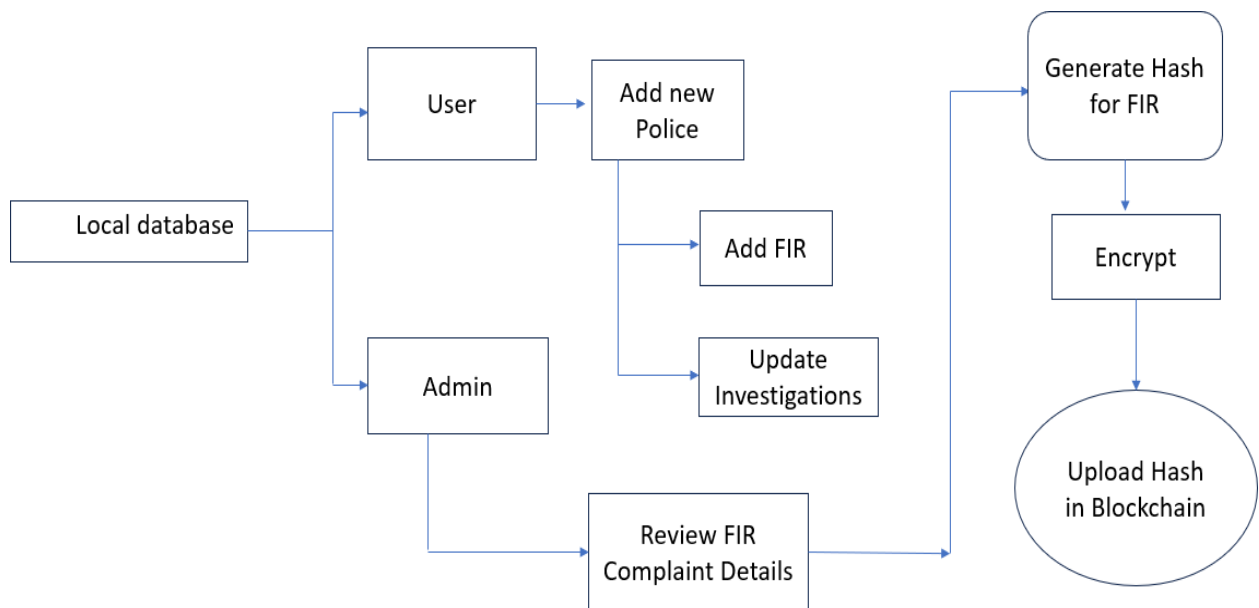
A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well



Fig(3.3.1) USECASE Diagram

- **ARCHITECTURE**

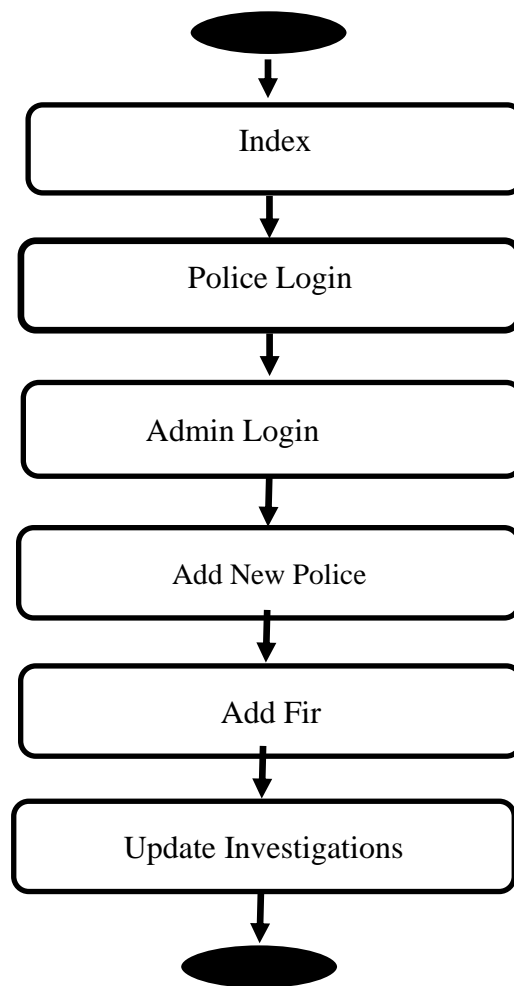
A sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



Fig(3.3.2) Architecture

- **ACTIVITY DIAGRAM:**

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent



Fig(3.3.3) Activity Diagram

### 3.4 Stepwise Implementation and Code

To implement this project first you need to run the Blockchain.bat in node modules file

- 1) Generate smart contract : when You run the blockchain.bat file a smart contract is generated with particular contract address for deployment of contracts
- 2) Update views File: Copy the deployment contract address and paste in views.py file where you need to choose initial migration and deployment address and paste the copied address.
- 3) Start IPFS: Now start the Interplanetary File system (IPFS) in order to store the data. When you start IPFS daemon will start and ready for storing the data.
- 4) Generate Browser URL: After starting the IPFS start clicking run button this will generate browser URL where you can execute your files.
- 5) Deploy Browser URL: After generating browser URL you should deploy it with index.html then check for the modules admin module, police module, FIR etc.,
- 6) Admin module: Login into admin module with proper credentials now you see login of a particular police officer using his credentials
- 7) Generating FIR: After logging of police officer with his credentials then he can generate FIR and add that record to the blockchain with particular hashcode

### 3.4.1 Source Code :

#### Views.py

```
from django.shortcuts import render
from django.template import RequestContext
from django.contrib import messages
from django.http import HttpResponse
from django.core.files.storage import FileSystemStorage
import os
from datetime import date
import os
import json
from web3 import Web3, HTTPProvider
import ipfsApi
import os
from django.core.files.storage import FileSystemStorage
import pickle
from datetime import date
import pyaes, pbkdf2, binascii, os, secrets
import base64
import urllib, mimetypes
from django.http import HttpResponse
global details, username
details=""
global contract
api = ipfsApi.Client(host='http://127.0.0.1', port=5001)
def readDetails(contract_type):
    global details
    details = ""
```

```
print(contract_type+"=====")
blockchain_address = 'http://127.0.0.1:9545' #Blockchain connection IP
web3 = Web3(HTTPProvider(blockchain_address))
web3.eth.defaultAccount = web3.eth.accounts[0]
compiled_contract_path = 'Police.json' #Police contract code
deployed_contract_address = '0x1DD4fb45C1cdC8C3f32cbaA60464c8107D4D4058'
with open(compiled_contract_path) as file:
    contract_json = json.load(file) # load contract info as JSON
    contract_abi = contract_json['abi'] # fetch contract's abi - necessary to call its functions
file.close()
contract = web3.eth.contract(address=deployed_contract_address, abi=contract_abi)
if contract_type == 'addusers':
    details = contract.functions.getUsers().call()
if contract_type == 'complaints':
    details = contract.functions.getComplaints().call()
if contract_type == 'investigations':
    details = contract.functions.getInvestigation().call()
print(details)
def saveDataBlockChain(currentData, contract_type):
    global details
    global contract
    details = ""
    blockchain_address = 'http://127.0.0.1:9545'
    web3 = Web3(HTTPProvider(blockchain_address))
    web3.eth.defaultAccount = web3.eth.accounts[0]
    compiled_contract_path = 'Police.json' #Police contract file
    deployed_contract_address = '0x1DD4fb45C1cdC8C3f32cbaA60464c8107D4D4058' #contract
    with open(compiled_contract_path) as file:
        contract_json = json.load(file) # load contract info as JSON
```

```
contract_abi = contract_json['abi'] # fetch contract's abi - necessary to call its functions
file.close()
contract = web3.eth.contract(address=deployed_contract_address, abi=contract_abi)
readDetails(contract_type)
if contract_type == 'addusers':
    details+=currentData
    msg = contract.functions.addUsers(details).transact()
    tx_receipt = web3.eth.waitForTransactionReceipt(msg)
if contract_type == 'complaints':
    details+=currentData
    msg = contract.functions.addComplaints(details).transact()
    tx_receipt = web3.eth.waitForTransactionReceipt(msg)
if contract_type == 'investigations':
    details+=currentData
    msg = contract.functions.addInvestigation(details).transact()
    tx_receipt = web3.eth.waitForTransactionReceipt(msg)
def index(request):
    if request.method == 'GET':
        return render(request, 'index.html', {})
def PoliceLogin(request):
    if request.method == 'GET':
        return render(request, 'PoliceLogin.html', {})
def AdminLogin(request):
    if request.method == 'GET':
        return render(request, 'AdminLogin.html', {})
def AddNewPolice(request):
    if request.method == 'GET':
        return render(request, 'AddNewPolice.html', {})
```

```
def AddFir(request):
    if request.method == 'GET':
        return render(request, 'AddFir.html', { })

def getKey(): #generating key with PBKDF2 for AES
    password = "s3cr3t*c0d3"
    passwordSalt = '76895'
    key = pbkdf2.PBKDF2(password, passwordSalt).read(32)
    return key

def encrypt(plaintext): #AES data encryption
    aes=pyaes.AESModeOfOperationCTR(getKey(),
    pyaes.Counter(311295470350000473029524339676541953981242398445663228841721636378460
    56248223))
    ciphertext = aes.encrypt(plaintext)
    return ciphertext

def decrypt(enc): #AES data decryption
    aes=pyaes.AESModeOfOperationCTR(getKey(),
    pyaes.Counter(311295470350000473029524339676541953981242398445663228841721636378460
    56248223))
    decrypted = aes.decrypt(enc)
    return decrypted

def UpdateInvestigations(request):
    if request.method == 'GET':
        global username
        output = '<tr><td><font size="" color="black">Complaint&nbsp;ID</font></td><td><select
name="t1">'
        readDetails("complaints")
        rows = details.split("\n")
        for i in range(len(rows)-1):
```



```
    arr = rows[i].split("#")
    output+= '<option value="'+arr[0]+'>'+arr[0]+'</option>'
    output += "</select></td></tr>"
    context= {'data1': output}
    return render(request, 'UpdateInvestigations.html', context)

def UpdateInvestigationsAction(request):
    if request.method == 'POST':
        global username
        today = date.today()
        complaint = request.POST.get('t1', False)
        investigation = request.POST.get('t2', False)
        today = date.today()
        data = complaint+"#" +username+"#" +investigation+"#" +str(today)+"#\n"
        saveDataBlockchain(data,"investigations")
        output = "Investigation Details Submitted Under Complaint No : "+str(complaint)
        context= {'data': output}
        return render(request, 'UserScreen.html', context)

def ViewInvestigations(request):
    if request.method == 'GET':
        global username
        output = '<table border=1 align=center>'
        output+= '<tr><th><font size=3 color=black>Complaint No</font></th>'
        output+= '<th><font size=3 color=black>Complaint Details</font></th>'
        output+= '<th><font size=3 color=black>Complainer Name</font></th>'
        output+= '<th><font size=3 color=black>Complainer Contact No</font></th>'
        output+= '<th><font size=3 color=black>Address</font></th>'
        output+= '<th><font size=3 color=black>Criminal Name</font></th>'
```

```
output+='<th><font size=3 color=black>Contact No</font></th>'
output+='<th><font size=3 color=black>Address</font></th>'
output+='<th><font size=3 color=black>Case Type</font></th>'
output+='<th><font size=3 color=black>Station Details</font></th>'
output+='<th><font size=3 color=black>IPFS Storage Hashcode</font></th>'
output+='<th><font size=3 color=black>Document Name</font></th>'
output+='<th><font size=3 color=black>Complaint Date</font></th>'
output+='<th><font size=3 color=black>Inspector Name</font></th>'
output+='<th><font size=3 color=black>Investigation Details</font></th>'
output+='<th><font size=3 color=black>Download Documents</font></th></tr>'
readDetails("investigations")
investigations = details.split("\n")
readDetails("complaints")
rows = details.split("\n")
for i in range(len(rows)-1):
    arr = rows[i].split("#")
    output+='<tr><td><font size=3 color=black>'+arr[0]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[1]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[2]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[3]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[4]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[5]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[6]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[7]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[8]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[9]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[10]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[11]+'</font></td>'
```

```
output+='\<td>\<font size=3 color=black>'+arr[12]+'</font></td>'
output+='\<td>\<font size=3 color=black>'+arr[13]+'</font></td>'
invest = ""
for k in range(len(investigations)):
    ar = investigations[k].split("#")
    if ar[0] == arr[0]:
        invest += ar[2]+"\\n"
output+='\<td>\<font size=3 color=black>'+invest+'</font></td>'
output+='\<td>\<a href=\"DownloadAction?file='+arr[0]+'"\>\<font size=3
color=black>Click Here</font></a></td></tr>'
    output += "</table><br/><br/><br/><br/>"
    context= {'data': output}
    return render(request, 'UserScreen.html', context)
def ViewReports(request):
    if request.method == 'GET':
        global username
        output = '<table border=1 align=center>'
        output+='\<tr>\<th>\<font size=3 color=black>Complaint No</font></th>'
        output+='\<th>\<font size=3 color=black>Complaint Details</font></th>'
        output+='\<th>\<font size=3 color=black>Complainer Name</font></th>'
        output+='\<th>\<font size=3 color=black>Complainer Contact No</font></th>'
        output+='\<th>\<font size=3 color=black>Address</font></th>'
        output+='\<th>\<font size=3 color=black>Criminal Name</font></th>'
        output+='\<th>\<font size=3 color=black>Contact No</font></th>'
        output+='\<th>\<font size=3 color=black>Address</font></th>'
        output+='\<th>\<font size=3 color=black>Case Type</font></th>'
        output+='\<th>\<font size=3 color=black>Station Details</font></th>'
        output+='\<th>\<font size=3 color=black>IPFS Storage Hashcode</font></th>'
        output+='\<th>\<font size=3 color=black>Document Name</font></th>'
```

```
output+='<th><font size=3 color=black>Complaint Date</font></th>'
output+='<th><font size=3 color=black>Inspector Name</font></th>'
output+='<th><font size=3 color=black>Investigation Details</font></th>'
output+='<th><font size=3 color=black>Download Documents</font></th></tr>'
readDetails("investigations")
investigations = details.split("\n")
readDetails("complaints")
rows = details.split("\n")
for i in range(len(rows)-1):
    arr = rows[i].split("#")
    output+='<tr><td><font size=3 color=black>'+arr[0]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[1]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[2]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[3]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[4]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[5]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[6]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[7]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[8]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[9]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[10]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[11]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[12]+'</font></td>'
    output+='<td><font size=3 color=black>'+arr[13]+'</font></td>'
    invest = ""
    for k in range(len(investigations)):
        ar = investigations[k].split("#")
        if ar[0] == arr[0]:
            invest += ar[2]+"\\n"
```

```
output+='<td><font size=3 color=black>'+invest+'</font></td>'
output+='<td><ahref=\'DownloadAction?file='+arr[0]+'\'><fontsize=3color=black>ClickHere</font>
</a></td></tr>'
    output += "</table><br><br><br><br>"
    context= {'data': output}
    return render(request, 'AdminScreen.html', context)
def AddFirAction(request):
    if request.method == 'POST':
        global username
        today = date.today()
        complaint = request.POST.get('t1', False)
        complainer_name = request.POST.get('t2', False)
        complainer_contact = request.POST.get('t3', False)
        complainer_address = request.POST.get('t4', False)
        criminal_name = request.POST.get('t5', False)
        criminal_contact = request.POST.get('t6', False)
        criminal_address = request.POST.get('t7', False)
        case_type = request.POST.get('t8', False)
        station = request.POST.get('t9', False)
        filedata = request.FILES['t10'].read()
        filename = request.FILES['t10'].name
        filedata = encrypt(filedata)
        complaint_id = 0
        readDetails("complaints")
        rows = details.split("\n")
        if len(rows) == 0:
            complaint_id = 1
```

```
    else:
        complaint_id = len(rows)
        filedata = pickle.dumps(filedata)
        hashcode = api.add_pyobj(filedata)
        data = str(complaint_id)+"#" + complaint+"#" + complainer_name+"#" + complainer_contact+"#" + complainer_
address+"#" + criminal_name+"#" + criminal_contact+"#"
        data = criminal_address+"#" + case_type+"#" + station+"#" + hashcode+"#" + filename+"#" + str(today)+"#" + user
name+"\n"
        saveDataBlockChain(data,"complaints")
        output = "Complaint Details saved in Blockchain with IPFS storage hashcode :
"+hashcode+"<br/>Complaint No : "+str(complaint_id)
        context= {'data': output}
        return render(request, 'UserScreen.html', context)

def DownloadAction(request):
    if request.method == 'GET':
        global username
        complaint = request.GET['file']
        fileName = ""
        hashcode = ""
        readDetails("complaints")
        rows = details.split("\n")
        response = HttpResponse(content,content_type="application/octet-stream")
        response['Content-Disposition'] = "attachment; filename=%s" % fileName
        return response
```

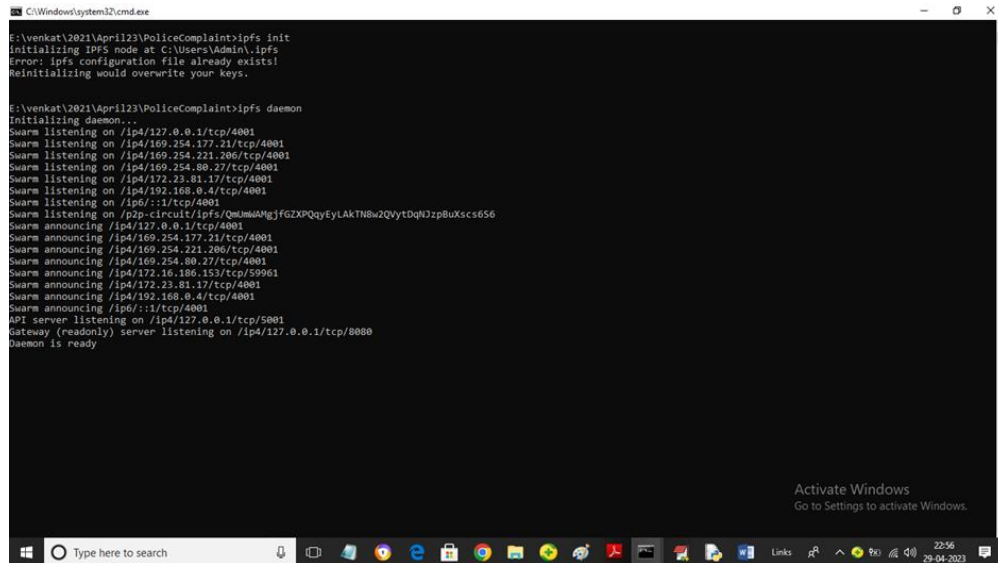
# **CHAPTER 4**

## **RESULTS AND DISCUSSION**

## RESULTS AND DISCUSSION

### 4.1 Output Screens :

- 1) To run the project first you need to start IPFS server

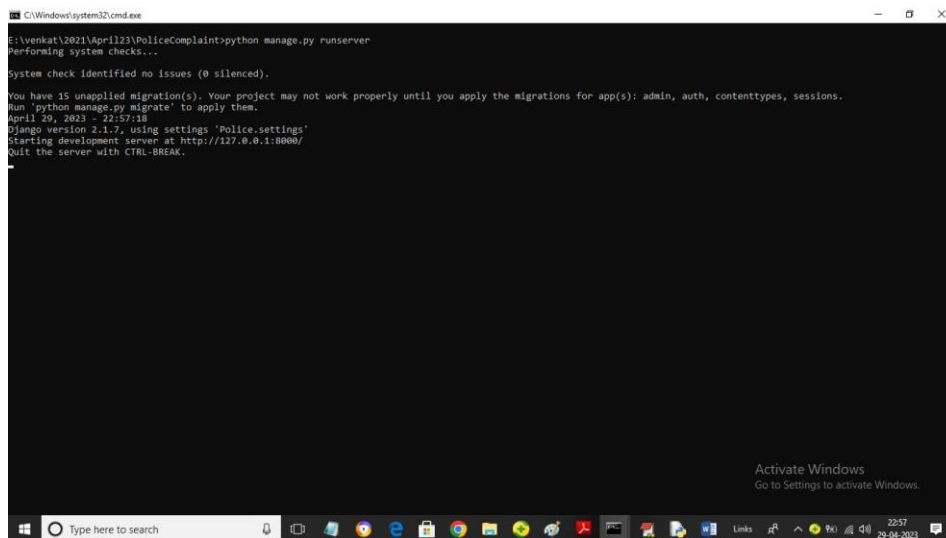


```
C:\Windows\system32\cmd.exe
E:\venkat\2021\April23\PoliceComplaint>ipfs init
initializing IPFS node at C:\Users\Admin\ipfs
Error: ipfs configuration file already exists!
Reinitializing would overwrite your keys.

E:\venkat\2021\April23\PoliceComplaint>ipfs daemon
initializing daemon...
Swarm listening on /ip4/127.0.0.1/tcp/4001
Swarm listening on /ip4/169.254.177.21/tcp/4001
Swarm listening on /ip4/169.254.221.206/tcp/4001
Swarm listening on /ip4/169.254.80.27/tcp/4001
Swarm listening on /ip4/172.23.81.17/tcp/4001
Swarm listening on /ip4/192.168.0.4/tcp/4001
Swarm listening on /ip6:::1/tcp/4001
Swarm listening on /p2p-circuit/ipfs/QmmbuMgjfGZXPQqEyLakTNBw2QytdqHJzPbUxscs656
Swarm announcing /ip4/127.0.0.1/tcp/4001
Swarm announcing /ip4/169.254.177.21/tcp/4001
Swarm announcing /ip4/169.254.221.206/tcp/4001
Swarm announcing /ip4/169.254.80.27/tcp/4001
Swarm announcing /ip4/172.16.186.153/tcp/59961
Swarm announcing /ip4/172.23.81.17/tcp/4001
Swarm announcing /ip4/192.168.0.4/tcp/4001
Swarm announcing /ip6:::1/tcp/4001
API server listening on /ip4/127.0.0.1/tcp/5001
Gateway (readonly) server listening on /ip4/127.0.0.1/tcp/8080
Daemon is ready
```

Fig-4.1.1 Starting IPFS server

- 2) Click run.bat to start the python web server



```
C:\Windows\system32\cmd.exe
E:\venkat\2021\April23\PoliceComplaint>python manage.py runserver
Performing system checks...

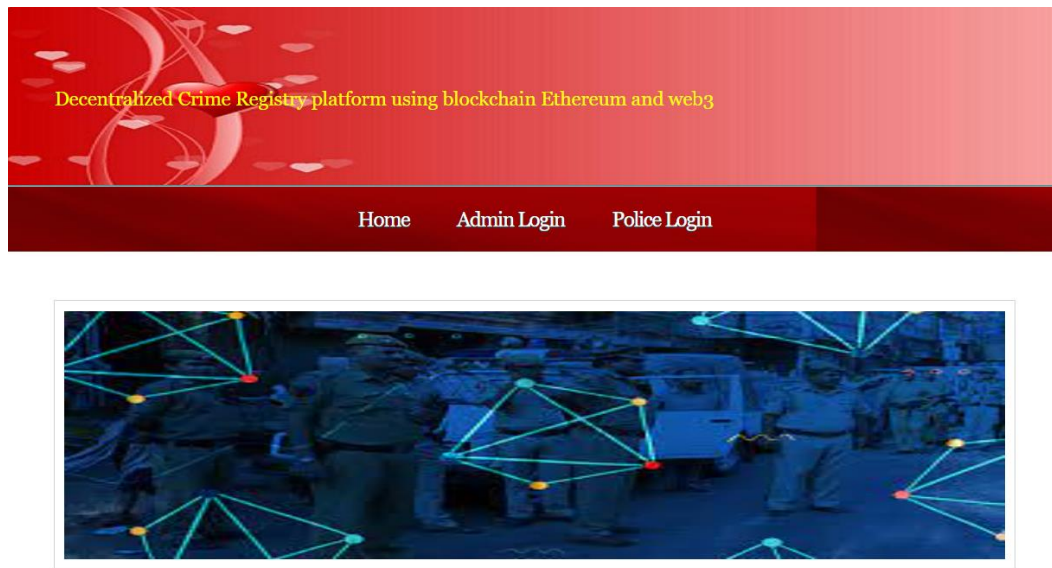
System check identified no issues (0 silenced).

You have 15 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
April 29, 2023 - 22:57:18
Django version 2.1.7, using settings 'Police.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.


```

Fig-4.1.2 Run Web server





Abstract:-Decentralized Crime Registry platform using blockchain Ethereum and web3

Fig -4.1.3 Home page

The image shows the "Admin Login Screen" of the web application. It features a dark red navigation bar at the top with the same three links: "Home", "Admin Login", and "Police Login". Below the navigation bar is a large image of police officers in uniform, overlaid with a network diagram. The login form is centered on the page and consists of the following elements:


**Admin Login Screen**

Username

Password

Fig -4.1.4 Admin login page


[Add New Police Users](#) [View Police Personnel](#) [View Reports](#) [Logout](#)



### Add New Police Personnel Screen

Username	<input type="text" value="sruthi"/>
Password	<input type="password" value="*****"/>
Contact No	<input type="text" value="9876543219"/>
Email ID	<input type="text" value="sruthi@gmail.com"/>
Address	<input type="text" value="hyd"/>

Fig -4.1.5 Police personnel screen



### Add Fir Screen

Complaint Details

Complainer Name	<input type="text" value="sruthi"/>
Contact No	<input type="text" value="7894561230"/>
Address	<input type="text" value="hyd"/>
Suspect Name	<input type="text" value="xyz"/>
Contact No	<input type="text" value="7894563214"/>
Address	<input type="text" value="hyd"/>
Report Type	<input type="text" value="FIR"/>
Register Under Station	<input type="text" value="hyd"/>
Supporting Document/ Image	<div><input type="button" value="Browse..."/> IEEE FINAL DOCS.docx</div> <div><input type="button" value="Submit"/></div>

Fig -4.1.6 FIR screen

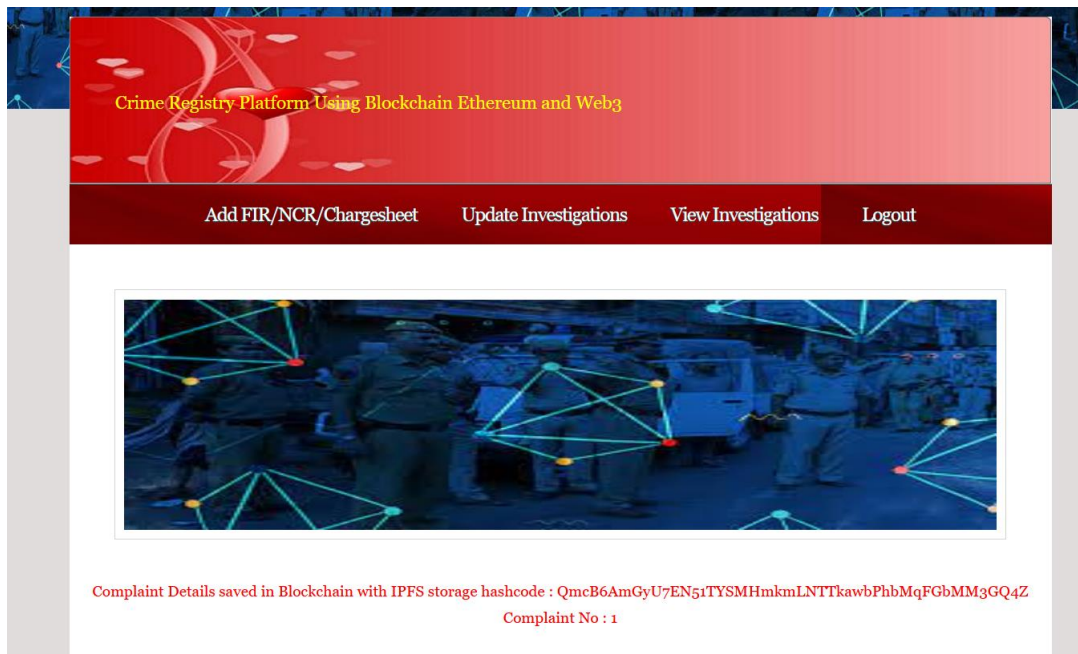


Fig -4.1.7 Storing compliant details in blockchain

The 'Update Investigations Screen' features a dark red navigation bar with buttons for 'Add FIR/NCR/Chargesheet', 'Update Investigations', 'View Investigations', and 'Logout'. Below the navigation bar is a blue-tinted image of police officers with a green network overlay. The form contains a 'Complaint ID' dropdown menu with '1' selected, an 'Investigation Details' text area containing 'theft case and murder', and a 'Submit' button.

Fig -4.1.8 Updating Existing records

Decentralized Crime Registry platform using blockchain Ethereum and web3											
Add FIR/NCR/Chargesheet    Update Investigations    View Investigations    Logout											
											
Complaint No	Complaint Details	Complainer Name	Complainer Contact No	Address	Criminal Name	Contact No	Address	Case Type	Station Details	IPFS Storage Hashcode	
1	theft	chandu	7801070652	hyderabad	rakesh	7801040654	hyd	FIR	hyd	QmSBNbX7anZCUS5mkTXUtx8vecYggREjcNTVgJqDXAs13l	

Fig -4.1.8 Stored Records

# CHAPTER 4

## CONCLUSION

## **CHAPTER 4**

### **CONCLUSION**

In conclusion, the integration of blockchain technology, specifically leveraging Ethereum blockchain and Web3 framework, in the crime record management system represents a significant leap forward in enhancing the security, transparency, and integrity of managing criminal records. By harnessing the decentralized nature of blockchain, the system ensures that data is stored across a distributed network of nodes, eliminating single points of failure and greatly reducing the risk of data tampering or unauthorized access. This decentralized approach not only enhances the resilience of the system but also instills trust among stakeholders, including law enforcement agencies, legal professionals, and the general public, in the authenticity and reliability of the stored records.

Furthermore, the use of cryptographic algorithms within the Ethereum blockchain ecosystem, such as hashing, encryption, and digital signatures, adds an extra layer of security to the crime record management system. These algorithms ensure the immutability of data, protect sensitive information from unauthorized disclosure, and enable the verification of transaction authenticity, thereby fortifying the overall integrity of the system. Additionally, the implementation of smart contracts facilitates the automation of various processes, including record creation, access control, and validation rules, streamlining operations and reducing administrative overhead.

Moreover, the seamless integration with the Web3 framework empowers users to interact with the crime record management system through user-friendly interfaces and decentralized applications (DApps). Through Web3.js, users can securely access and query blockchain data, execute transactions, and interact with smart contracts, enhancing the accessibility and usability of the system. Overall, the adoption of blockchain technology, Ethereum blockchain, and Web3 framework in the crime record management system not only addresses the limitations of traditional solutions but also sets a new standard for security, transparency, and trustworthiness in managing sensitive criminal records.

## **FUTURE SCOPE :**

- Implementation of blockchain-based digital identity solutions to improve the accuracy and reliability of identity verification processes.
- Integration with emerging technologies like Internet of Things (IoT) devices for gathering additional data for crime prevention and investigation.
- Adoption of blockchain-based voting systems for community engagement and decision-making on system upgrades and policy changes.
- Integration of machine learning algorithms for predictive analytics to identify crime trends and patterns, aiding in proactive law enforcement strategies.
- Expansion of functionalities to include real-time data updates and notifications for law enforcement agencies.

# REFERENCES



## REFERENCES

- [1] Gupta, Antra and D. Vilchez Jose. "FIR System using Blockchain.".(IJRTE) ISSN: 2247-3858, Volume-7, Issue-1, May 2021.
- [2] K. Tabassum, H. Shaiba, S. Shamrani and S. Otaibi," e-Cops: An Online Crime Reporting and Management System for Riyadh City," 2018 1st International Conference on Computer Applications Information Security (ICCAIS).
- [3] P.A.K.S.Y. K. S. , Shivaganesh Pillai, "Online Fir Registration and Sos System", int. jour. eng. com. sci, vol. 5,no. 4, Dec. 2017.
- [4] Giova, G. (2011) "Improving chain of custody in forensic investigation of electronic digital systems." *International Journal of Computer Science and Network Security* **11** (1): 1-9.
- [5] Xu, X, Weber I, Staples M, et al. (2017) A taxonomy of blockchain-based systems for architecture design. In: *2017 IEEE international conference on software architecture (ICSA)* 243-252. IEEE.
- [6] Du, X, Le-Khac N-A and Scanlon M. (2017) "Evaluation of digital forensic process models with respect to digital forensics as a service." *arXiv preprint arXiv:170801730*.
- [7] Prayudi, Y and Sn A. (2015) "Digital chain of custody: State of the art." *International Journal of Computer Applications* **114**: 1-9.
- [8] Nakamoto, S. (2008) "Bitcoin: A peer-to-peer electronic cash system." *Bitcoin-URL: <https://bitcoin.org/bitcoin.pdf>*.
- [9] Lone, AH and Mir RN. (2018) "Forensic-chain: ethereum blockchain based digital forensics chain of custody." *Sci Pract Cyber Secur J* **1** (2): 21-27.
- [10] Pilkington, M. (2016) "11 Blockchain technology: principles and applications." *Research handbook on digital transformations* **225**: 225-253.
- [11] Buterin, V. (2014) "A next-generation smart contract and decentralized application platform." *white paper* **3** (37).

- [12] Ølnes, S, Ubacht J and Janssen M. (2017) "Blockchain in government: Benefits and implications of distributed ledger technology for information sharing." *Government Information Quarterly* **34** (3): 355-364.
- [13] "Online Criminal Record Management System" by Pratibha Mishra, Ghousiya Bee. N, Mohsina S, MubashshiraSultana, Surbhi Singh, IJESC Volume 9 Issue No. 05 2019.
- [14] "Blockchain Based Crime Record Management System" by Bhushan Dube, Mahesh Gangarde, Ankit Singh, Jitendra Pawar, Sagar Dhanake. JAC: A Journal Of Composition Theory. ISSN: 0731-6755.
- [15] "A Method to Secure FIR System using Blockchain" by Antra Gupta, Deepa V. Jose. International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8, Issue-1, May 2019.
- [16] Blockchain Definition: What You Need to Know. (n.d.). Retrieved May 25, 2022, from <https://www.investopedia.com/terms/b/blockchain.asp>
- [17] Cryptography in Blockchain Explained | by Amarpreet Singh | Brandlitic | Medium. (n.d.). Retrieved May 25, 2022, from <https://medium.com/brandlitic/cryptography-in-blockchain-explained-df11fe1bd0f7>
- [18] What Is SHA-256 Algorithm: How it Works and Applications [2022 Edition] | Simplilearn. (n.d.). Retrieved May 25, 2022, from <https://www.simplilearn.com/tutorials/cyber-security-tutorial/sha-256-algorithm>
- [19] Smart Contracts and Chaincode — hyperledger-fabricdocs main documentation. (n.d.). Retrieved May 25, 2022, from <https://hyperledger-fabric.readthedocs.io/en/release-2.2/smartcontract/smartcontract.html>
- [20] Iyer A, Kathale P, Gathoo S and Surpam N 2016 E-Police System-FIR Registration and Tracking through Android Application International Research Journal of Engineering and Technology 3(2) 1176-1179. P. A. K. S. Y. K. S. , Shivaganesh Pillai, "Online Fir Registration and Sos System", int. jour

**GITHUB LINK :** <https://github.com/Chandu987654>

**IJRASET**  
International Journal For Research in  
Applied Science and Engineering Technology



**INTERNATIONAL JOURNAL  
FOR RESEARCH**  
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

**Volume:**      **Issue:**      **Month of publication:**  
**DOI:**

**[www.ijraset.com](http://www.ijraset.com)**  
**Call:** ☎ 08813907089      |      **E-mail ID:** [ijraset@gmail.com](mailto:ijraset@gmail.com)



# Crime Registry Platform using Blockchain Ethereum and web3

K. Chandu<sup>1</sup>, G.V. Muralidhar<sup>2</sup>, K. Sruthi<sup>3</sup>, P. Senthil<sup>4</sup>

<sup>1, 2, 3</sup>UG Student, Department of CSE, CMR College of Engineering & Technology, Hyderabad, Telangana

<sup>4</sup>Assistant Professor, Department of CSE, CMR College of Engineering & Technology, Hyderabad, Telangana

**Abstract:** Criminal activities in India are on the rise, with many incidents going unreported. Despite the availability of an online portal for storing the First Information Reports (FIRs) handwritten FIRs always will remain common due to some of the traditional methods and practices. And probably in most cases, the complainants should personally visit the police stations in order to file a offense report.

Crime and Criminal Tracking Network and Systems (CCTNS) was launched in 2010 for national wide e-governance, it will operates on a centralized system and it is particularly limited to some individual states. Therefore, there is a need for the decentralized solution in order to ensure failure of a single point and secure managing of criminal complaints from the unauthorized access.

Our project aims to address all this by depicting blockchain technology. Whatever the FIRs that are filed by police will be encrypted and stored on the InterPlanetary File System (IPFS), with the hash code or the hash value added to the blockchain network. This approach will provides strong and proper evidence against any attempts from the police to ignore or to the denial of the complaints. Storing records on an blockchain database which is immutable and it eliminates the risk of tampering, ensuring transparency and accountability in managing FIRs and NCRs.

**Keywords:** FIR, CCTNS, IPFS, NCR, CRYPTOGRAPHY

## I. INTRODUCTION

In India, generally the legal system divides offenses into two main categories: There are some offenses where police have an authority to arrest without any warrant in another case relatively police cannot make an arrest without a warrant. This captivates the crimes like murder, theft, kidnapping, and rape, among others. According to Section 2 (c) the police have the authority to arrest the suspect without any type of warrant. Then the inspector can begin the investigation process without the need of courts they can figure out there self. In some cases of cognizable offenses, the (FIR) is registered at the police station, which can be filed by any individual who is either a victim or a witness to the offense. The FIR contains crucial details such as the complainant's name and address, the date and time of the incident, the location, and also a description of the facts. Following the registration of the FIR, the police officer handling the case compiles a chargesheet report.

## II. RELATED WORK

**A. E-Cops is an innovative online Crime ReportingManagement:**

E-Cops an online platform designed for the Cities, addressing concerns about visiting police stations. It allows people to report crimes conveniently and securely from their devices, offering anonymity if desired. Its main goal is to strengthen the connection between citizens and law enforcement, enabling easier sharing of information and evidence. This digital solution enhances collaboration, helps track criminals, and improves crime reporting accessibility and efficiency in that place.

**B. FIR Registration and Trackingthrough Android Application:**

The E-Police System is a modernization initiative within the Indian Police Department to replace outdated manual processes with digital solutions. Traditionally, filing complaints and tracking cases involved physically visiting police stations, which is no longer efficient given the current volume of cases. To address this, an Android application is being developed to facilitate the registration of First Information Reports (FIR) and track cases. This application collects complainant data and transmits it to the police department's web portal, enabling seamless information exchange. The goal is to improve efficiency and transparency in the complaint registration and tracking process by leveraging digital technology.



### C. Implementation of an E-Police System

The police system in India, a country with a population of around 1.3 billion, is facing significant challenges marked by degradation and a surge in criminal activities such as murder, theft, robbery, assassinations, and corruption. In order to address all these issues, the growing focus on implementing electronic policing (E-police) systems, with the emerging Internet technologies. E-police, adopted by law enforcement agencies across all over the world, aims to enhance the delivery of law enforcement services to people. However, the Internet technology and electronic-based systems a large infrastructure with some initial costs and rapid development of the application capabilities.

## III. METHODS AND EXPERIMENTAL DETAILS

### A. Designing a Data Model

Choose a better blockchain platform for designing a data model you can choose platforms like Ethereum corda etc., Now Create the structure and schema for storing crime-related information in blockchain. This involves specifying which kind of data to be included in platform with necessary metadata.

### B. Developing the Smart Contracts

- 1) *Setting Up Development Environment:* Set up a development environment using tools like Truffle or Remix IDE. Use the Solidity programming language in order to a smart contract that will defines the structure and functionality of the crime registry platform define functions for adding new crime reports, updating the status of existing reports, retrieving crime data, and managing access control
- 2) *Testing, Storing in IPFS and Compilation:* Test the smart contract thoroughly using Truffle's testing framework or Remix IDE. Compile the smart contract code into bytecode using the Solidity compiler.
- 3) *Deployment:* Use Truffle or command-line interfaces provided by Ethereum to deploy the compiled smart contract onto the Ethereum blockchain network.

### C. Integration with Web3 and Ethereum

- 1) *Connecting to Ethereum Node:* Choose an Ethereum node provider (e.g., Infura) and obtain an API key or connection URL. Initialize Web3 instance in your web application using the connection URL.

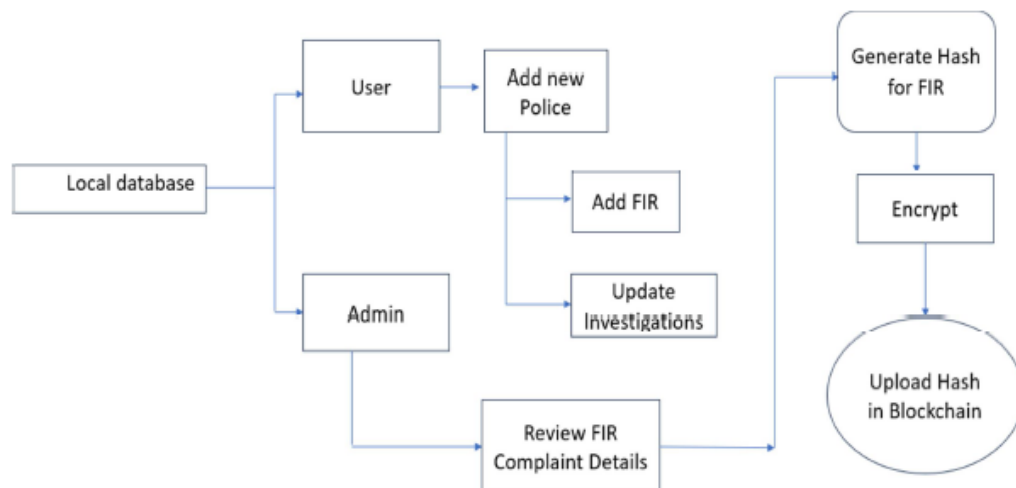


Fig-1 Block Diagram of crime registry platform using blockchain



*D. UI Integration*

Integrate Web3.js it will call into the UI of the web application and also it is used to allow users to interact with the crime registry platform.

Display the crime data that has retrieved from the smart contract on the UI. And then deploy the web application to a hosting server.

*E. Algorithm 1 (SHA 256 Algorithm)*

- 1) *Input:* Data: The input data has to be hashed
- 2) *Output:* Hash: The resulting SHA-256 hash value
- 3) *Procedure:*
  - a) Initialize the SHA-256 hash function.
  - b) Convert the input data into a binary representation.
  - c) Pad the binary representation to ensure its length is a multiple of 512 bits.
  - d) Break the padded binary data into 512-bit blocks.
  - e) Initialize the eight working variables (a-h) with the initial hash values.
  - f) Prepare the message schedule array (W[0..63]) from the block.
  - g) Initialize working variables (a-h) with the previous hash value.
  - h) Perform 64 rounds of hashing using a set of bitwise operations and constant values.
  - i) Update working variables (a-h) based on the results of each round.
  - j) Compute the final hash value by concatenating the values of (a-h) in the specified order.
  - k) Return the resulting hash value.

*F. Algorithm 2 (ECDSA)*

- 1) *Input:* Message: The message to be signed
- 2) *Output:* Signature: The resulting digital signature
- 3) *Procedure:*
  - a) Generate a random value  $k$  (nonce) that is a positive integer less than the order of the curve's basepoint.
  - b) Calculate the point  $(x_1, y_1) = k * G$ , where  $G$  is the basepoint of the elliptic curve.
  - c) Calculate  $r = x_1 \bmod n$ , where  $n$  is the order of the elliptic curve's base point.
  - d) If  $r = 0$ , go back to step 1 and choose a different value for  $k$ .
  - e) Calculate  $s = (k^{-1} * (\text{Hash}(\text{Message}) + \text{PrivateKey} * r)) \bmod n$ , (e.g., SHA-256).
  - f) If  $s = 0$ , go back to step 1 and choose a different value for  $k$ .
  - g) The resulting pair  $(r, s)$ .
  - h) Return the signature.

## IV. RESULT AND DISCUSSIONS

The implementation of a crime registry platform using the blockchain technology it represents an important step to forward in improving the security, and efficiency of criminal information. By acknowledging the decentralized nature of the blockchain, this platform will ensure that the criminal data will stay tamper-proof and instilling good positive trust among the people, government officers, and the public.

This high-level of security is not only protects the integrity and scalability of crime records but also manages the risk of unauthorized modifications, and thereby boosting the reliability of the information stored on the platform.

However, the adoption of blockchain technology for managing criminal records also raises important considerations, particularly regarding privacy issues, legal and regular compliance, and the usability of the platform.

Addressing all these challenges it requires an implementation of robust privacy measures, such as encryption and access control mechanisms, to protecting sensitive information while ensuring compliance with particular related data and regulations. Additionally, user-friendly interfaces and seamless integration with existing systems are essential for facilitating widespread adoption and usability among stakeholders.



International Journal for Research in Applied Science & Engineering Technology (IJRASET)  
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538  
Volume 12 Issue III Mar 2024- Available at [www.ijraset.com](http://www.ijraset.com)



Fig-2 Starting IPFS Server



Fig-3 Admin Login Screen



Fig-4 Police Personnel Screen

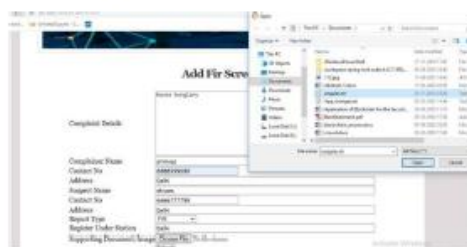


Fig-5 Adding FIR files





Fig-6 Adding Files into Blockchain

## V. PERFORMANCE ANALYSIS

The table presents a comparative analysis of the performance of two prominent blockchain platforms, Ethereum and Hyperledger, focusing on their transaction processing capabilities and support for smart contracts.

S.N	Blockchain Platform	Type	Smart contract	Tps
1	Ethereum	Public/private	5	30Tps
2	Hyperledger	private	0	80Tps

## VI. CONCLUSION

In conclusion, the Indian Police Services play a vital role in our nation, handling over 50 lakh complaints related to cognizable crimes annually. Despite the availability of online systems for managing complaints, there persists a reliance on handwritten reports, and societal apprehensions hinder the filing of complaints. Efficiently managing police complaints is crucial due to the sensitivity of the data involved.

The proposed system aims to address these challenges by offering transparency and ensuring the confidentiality of stored data. It is anticipated to encourage individuals to report complaints with the confidence, knowing that their concerns will not be overlooked. Additionally, the system aims to streamline the reporting process for police officers, reducing the burden of tasks like filing FIRs. By adopting a decentralized network, the proposed system minimizes reliance on trust factors among stakeholders. Overall, the system is designed to safeguard against corrupt police activities, ensuring justice is served from the outset.

## REFERENCES

- [1] Gupta, Antra and D. Vilchez Jose. "FIR System using Blockchain". (IJRTE) ISSN: 2247-3858, Volume-7, Issue-1, May 2021.
- [2] K. Tabassum, H. Shaiba, S. Shamrani and S. Otaibi. "e-Cops: An Online Crime Reporting and Management System for Riyadh City," 2018 1st International Conference on Computer Applications Information Security (ICCAIS).
- [3] P.A.K.S.Y. K. S., Shivaganes Pillai, "Online Fir Registration and Sos System", int. jour. eng. com. sci. vol. 5, no. 4, Dec. 2017.
- [4] M.SUDHAKAR, Published a paper entitled "EMOTION BASED MUSIC PLAYER" In (Volume 21, May-2022 Issue 05, YMER).



# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 📞 (24\*7 Support on Whatsapp)







