# AI ENHANCED PUBLIC MONITORING SYSTEM

*A Project Report*

*submitted to*

*the APJ Abdul Kalam Technological University*

*in partial fulfillment of the requirements for the degree of*

***Bachelor of Technology***

*by*

**ASHIK JHONSON (VML21CS071)**

**ARMOND JOSE (VML21CS068)**

**GERALD SIRIAC(VML21CS091)**

**ALFI SIBY (VML21CS042)**

*under the supervision of*

**Mr. RIJIN I K**

**Assistant Professor**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**VIMAL JYOTHI ENGINEERING COLLEGE CHEMPERI**

CHEMPERI P.O. - 670632, KANNUR, KERALA, INDIA

**March 2025**

VIMAL JYOTHI
ENGINEERING COLLEGE
JYOTHI NAGAR, CHEMPERI – 670632, KANNUR, KERALA
ACCREDITED BY IEI, NBA & NAAC ◆ ISO 9001:2015 CERTIFIED
AFFILIATED TO KTU ◆ APPROVED BY AICTE

NATIONAL BOARD OF ACCREDITATION
ACCREDITED

NAAC ACCREDITED

**DEPT. OF COMPUTER SCIENCE AND ENGINEERING**

## CERTIFICATE

This is to certify that the report entitled **AI ENHANCED PUBLIC MONITORING SYSTEM** submitted by **ASHIK JHONSON** (VML21CS071), **ARMOND JOSE** (VML21CS068), **GERALD SIRIAC** (VML21CS091) & **ALFI SIBY** (VML21CS042) to the APJ Abdul Kalam Technological University in partial fulfillment of the B.Tech. degree in Computer Science and Engineering is a bonafide record of the project work carried out by them under our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

**Mr. RIJIN I K**
(Project Guide)
Assistant Professor
Dept.of CSE
Vimal Jyothi Engineering College
Chemperi

**Ms. Divya B**
(Project Coordinator)
Assistant Professor
Dept.of CSE
Vimal Jyothi Engineering College
Chemperi

**Ms. Dinsha P.K**
(Project Coordinator)
Assistant Professor
Dept.of CSE
Vimal Jyothi Engineering College
Chemperi

Place: VJEC Chemperi
Date:27-3-2025

Head of the department

(Office Seal)

# DECLARATION

We hereby declare that the project report **AI ENHANCED PUBLIC MONITORING SYSTEM**, submitted for partial fulfillment of the requirements for the award of degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bona fide work done by us under supervision of **Mr. RIJIN I K**

This submission represents our ideas in our own words and where ideas or words of others have been included, we have adequately and accurately cited and referenced the original sources.

We also declare that we have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. We understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

**ASHIK JHONSON**<br>
**ARMOND JOSE**<br>
**GERALD SIRIAC**<br>
**ALFI SIBY**

CHEMPERI
27-3-2025

## ACKNOWLEDGEMENT

We are extremely grateful to **Dr. Benny Joseph**, Principal and **Ms. Divya B.** Head of Department, Department of CSE, and our project coordinators for providing all the required resources for the successful completion of our project.

We express my thanks to our project coordinators **Ms. Divya B**, Associate Professor, **Ms. Dinsha P.K**, Assistant Professor and all staff members and friends for all the help and co-ordination extended in bringing out this project successfully in time. Our thanks and appreciation also goes to the authors of the references and other literatures referred to in this project.

We are highly indebted to the guide **Mr. Rijin I.K**, Assistant Professor for his guidance and constant supervision, as well as for providing necessary information regarding the project and also for his support.

Last but not the least, We would like to express our gratitude towards our parents and friends for their kind cooperation and encouragement.

**ASHIK JHONSON**
**ARMOND JOSE**
**GERALD SIRIAC**
**ALFI SIBY**

# Abstract

The AI Enhanced Public Monitoring System is an innovative solution aimed at improving public safety through real-time detection and reporting of emergencies, such as accidents, fires, and criminal activities. Leveraging advancements in computer vision and artificial intelligence, the system integrates YOLO (You Only Look Once) for object detection and Optical Flow techniques to identify motion patterns, enabling detection of sudden changes or unusual movements that may indicate emergencies. To ensure secure and tamper-proof video evidence, blockchain technology is employed for storage, maintaining data integrity crucial for legal proceedings and investigations. Additionally, the system uses IPFS (InterPlanetary File System) for decentralized storage, ensuring scalability and security by eliminating reliance on traditional databases. Edge computing enables real-time processing close to the source, reducing latency for faster emergency detection and reporting. Designed to be scalable, the system can be deployed across large urban areas, smart cities, and high-risk zones, providing constant monitoring to enhance public safety. By automating the detection, classification, and reporting of emergencies, the AI Enhanced Public Monitoring System improves both the reliability and responsiveness of public monitoring infrastructure.

# Contents

**Abstract** **iii**

**List of Figures** **vii**

**Nomenclature** **viii**

**1 Introduction** **1**

1.1 Overview . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1

1.2 General Background . . . . . . . . . . . . . . . . . . . . . . . . . 2

1.3 Problem statement . . . . . . . . . . . . . . . . . . . . . . . . . 3

1.4 Objective . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

**2 Literature Review** **5**

2.1 VD-Net: An Edge Vision-Based Surveillance System for Violence Detection . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

2.2 Cognition Guided Video Anomaly Detection Framework for Surveillance Services . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

2.3 YOLO-SF: YOLO for Fire Segmentation Detection . . . . . . . . . . 6

2.4 Smart City Transportation: Deep Learning Ensemble Approach for Traffic Accident Detection . . . . . . . . . . . . . . . . . . . . . 8

2.5 A Secure, Reliable and Low-Cost Distributed Storage Scheme Based on Blockchain and IPFS for Firefighting IoT Data . . . . . . . . . . . 9

**3 Requirement Specification** **10**

3.1 Functional Requirements . . . . . . . . . . . . . . . . . . . . . . . . . 10
3.2 Non-Functional Requirements . . . . . . . . . . . . . . . . . . . . . . 10
3.3 Software Requirements . . . . . . . . . . . . . . . . . . . . . . . . . . 11
3.4 Hardware Requirements . . . . . . . . . . . . . . . . . . . . . . . . . 11

**4 Proposed system and Design 12**

4.1 Proposed system . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
4.2 Feasibility Study . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13
4.2.1 Technical Feasibility . . . . . . . . . . . . . . . . . . . . . . . 13
4.2.2 Operational Feasibility . . . . . . . . . . . . . . . . . . . . . . 14
4.2.3 Economic Feasibility . . . . . . . . . . . . . . . . . . . . . . . 14
4.3 Design . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
4.3.1 Architecture Diagram . . . . . . . . . . . . . . . . . . . . . . 15
4.3.2 Use Case Diagram . . . . . . . . . . . . . . . . . . . . . . . . 15
4.3.3 Data Flow Diagram . . . . . . . . . . . . . . . . . . . . . . . 16

**5 Implementation 19**

5.1 Implementation details . . . . . . . . . . . . . . . . . . . . . . . . . . 19
5.1.1 Loading pre trained object detection model . . . . . . . . . . 20
5.1.2 Perform prediction based on pre trained model . . . . . . . . 21
5.1.3 Sending Alert Notification . . . . . . . . . . . . . . . . . . . . 21
5.1.4 Back-End . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23
5.1.5 Blockchain . . . . . . . . . . . . . . . . . . . . . . . . . . . . 24
5.2 Modules Used . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25
5.2.1 Video Input Module (Live Surveillance Feed) . . . . . . . . . 25
5.2.2 Model Loading and Initialization Module . . . . . . . . . . . 25
5.2.3 Detection and Classification Module . . . . . . . . . . . . . . 26
5.2.4 Event Alert and Notification Module . . . . . . . . . . . . . . 26
5.2.5 Decentralized Storage Module . . . . . . . . . . . . . . . . . 26
5.2.6 Web Interface Module . . . . . . . . . . . . . . . . . . . . . 27

**6 Result and Discussion** **28**

6.1 Web Interface . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 28

6.2 Alert Message . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

**7 Conclusion** **32**

**References** **33**

# List of Figures

4.1 Architecture Diagram . . . . . . . . . . . . . . . . . . . . . . . . . 15
4.2 Use-Case Diagram . . . . . . . . . . . . . . . . . . . . . . . . . 16
4.3 DFD Level0 . . . . . . . . . . . . . . . . . . . . . . . . . . . 17
4.4 DFD Level1 . . . . . . . . . . . . . . . . . . . . . . . . . . . 17
4.5 DFD Level2 . . . . . . . . . . . . . . . . . . . . . . . . . . . 18

5.1 Load Model Image . . . . . . . . . . . . . . . . . . . . . . . . . 20
5.2 Detection Image . . . . . . . . . . . . . . . . . . . . . . . . . . 21
5.3 Alert Code . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22
5.4 Back End Image . . . . . . . . . . . . . . . . . . . . . . . . . . 23
5.5 Block Chain . . . . . . . . . . . . . . . . . . . . . . . . . . . 24

6.1 Web Image . . . . . . . . . . . . . . . . . . . . . . . . . . . . 28
6.2 Notification Image . . . . . . . . . . . . . . . . . . . . . . . . 30

# Nomenclature

| | |
|---|---|
| **AI** | Artificial Intelligence |
| **CBAM** | Convolutional Block Attention Module |
| **CNN** | Convolutional Neural Network |
| **DFD** | Data Flow Diagram |
| **IPFS** | InterPlanetary File System |
| **LSTM** | Long Short-Term Memory |
| **ST-TCN** | Special task temporal convolutional network |
| **VD-Net** | Violence Detection Network |
| **YOLO** | You Only Look Once |

# Chapter 1

# Introduction

## 1.1 Overview

The AI-Enhanced Public Monitoring System is designed to improve public safety by leveraging cutting-edge technologies such as computer vision, deep learning, and blockchain. The system operates in real-time, detecting and reporting emergencies like accidents, fires, and crimes in public spaces. At its core, YOLO (You Only Look Once) is employed for real-time object detection, identifying critical objects such as people, vehicles, or fire within video surveillance feeds. This is complemented by Optical Flow techniques, which analyze motion patterns to detect sudden or abnormal movements that might indicate emergency situations. Together, these technologies enable the system to recognize potentially dangerous situations, signaling authorities for timely intervention.A key differentiator of this system is its integration with blockchain technology and the InterPlanetary File System (IPFS) for secure and decentralized storage of video evidence. This approach eliminates the need for traditional databases by using blockchain's cryptographic hashes to ensure tamper-proof data, making it suitable for legal investigations. Large video files are distributed across multiple nodes via IPFS, providing scalability and reliability in data management. In the event of an emergency, the system immediately alerts authorities, providing real-time data to aid in rapid response.

The system's use of edge computing ensures that video processing occurs close to the data source (e.g., CCTV cameras), reducing latency for faster detection and reporting. By automating the detection, classification, and alerting processes, the system minimizes human error and enhances the efficiency of monitoring large public spaces. The combination of AI and blockchain provides a robust framework for trustworthy data handling, making the recorded data reliable for use as evidence.Designed for scalability, the AI-Enhanced Public Monitoring System can be deployed across urban environments, smart cities, or high-risk areas, where continuous monitoring and real-time response capabilities are crucial. The integration of YOLO for object detection, Optical Flow for motion analysis, and blockchain for data integrity ensures that the system is both effective and adaptable, addressing the complex needs of modern public safety infrastructure.

## 1.2 General Background

This project aims to develop an AI-powered, real-time public monitoring system capable of detecting and responding to emergencies like accidents, fires, and criminal activities. Traditional surveillance systems are often prone to human error, fatigue, and delayed response times. By incorporating advanced computer vision and machine learning techniques, such as YOLO for object detection and Optical Flow for motion analysis, the system autonomously detects emergencies and alerts authorities. Additionally, blockchain technology is utilized to securely store video evidence in a tamper-proof manner, maintaining data integrity for any necessary legal follow-up. IPFS enables decentralized storage, eliminating traditional database dependency while ensuring both scalability and security. Designed for deployment in smart cities and high-risk areas, this project presents a scalable, automated solution for real-time public safety monitoring that minimizes human supervision and speeds up emergency response.

## 1.3 Problem statement

Current public surveillance systems largely depend on human operators to monitor live video feeds, which introduces limitations like human error, fatigue, and delayed responses. These manual systems may be inadequate for detecting emergencies—such as accidents, fires, or crimes—in real time, potentially delaying interventions and allowing dangerous situations to escalate. Moreover, conventional storage methods for surveillance footage rely on centralized databases, which are vulnerable to tampering or data loss, compromising evidence integrity crucial for investigations. In complex urban settings, smart cities, and high-risk areas, there is an urgent need for an automated, intelligent monitoring system that can detect emergencies promptly and securely store video evidence. Developing a system that improves both the speed and accuracy of emergency detection while maintaining data integrity and security is essential. Traditional centralized systems also struggle with scalability, posing challenges in managing large volumes of data across vast public areas. This project addresses these issues by using artificial intelligence, real-time video processing, and blockchain technology to enhance public safety, reduce response times, and ensure the reliability of evidence.

## 1.4 Objective

The objective of the AI-Enhanced Public Monitoring System is to develop an advanced, automated solution for real-time detection and reporting of emergencies such as accidents, fires, and criminal activities. Leveraging artificial intelligence and computer vision, the system aims to detect and classify emergency situations with high accuracy through technologies like YOLO for object detection and Optical Flow for motion analysis. This approach enables the system to analyze video feeds from public surveillance cameras, identifying abnormal behaviors, sudden movements, or patterns that may signal an emergency. Another key objective is to ensure the tamper-proof storage of video evidence using blockchain technology, which guarantees data integrity and security, making it reliable for legal proceedings or investigations. The system also incorporates edge computing to process video data closer to its source, reducing latency for faster detection and response times. Scalability is crucial, ensuring that the system can be deployed in smart cities, urban environments, or high-risk areas with large-scale monitoring requirements. Overall, the system automates the detection, classification, and reporting of emergencies to significantly enhance public safety, facilitate timely responses, and maintain data integrity through a combination of AI, blockchain, and decentralized storage technologies.

# Chapter 2

# Literature Review

## 2.1 VD-Net: An Edge Vision-Based Surveillance System for Violence Detection

The automation of surveillance systems, driven by the rapid development of computer vision Technology, has significantly enhanced the analysis of surveillance videos, particularly in recognition Of human activity, including behavior analysis and violence detection, thereby bolstering public and Industrial security. Despite these advancements, detecting and analyzing violent actions remains challenging, Especially for real-time surveillance systems with limited computing power. We propose an artificial Intelligence- based framework called VD-Net (Violence Detection Network), enabled by Intelligent Internetof-Things (IIoT) to detect violent behavior in public and private spaces. The model utilizes lightweight Special task temporal convolutional network (ST-TCN) blocks and several bottleneck layers to focus on Salient features in the input sequence. The learned features passed from the classifier to discriminate Between violent and nonviolent actions. Additionally, our system is supposed to trigger an alert if violence Is detected, which is then communicated to relevant departments. We checked the robustness of our system By surveillance and non-surveillance datasets and ensured a 1-4 The-Art (SoTA) Accuracy.

## 2.2 Cognition Guided Video Anomaly Detection Framework for Surveillance Services

Existing approaches to video anomaly detection often face significant challenges due to the inherent imbalance between normal and abnormal data, frequently leading to poor generalization and models that overfit to specific scenes. This paper introduces a Cognition Guided Video Anomaly Detection (CG-VAD) framework, designed to enhance anomaly detection by integrating both explicit and implicit knowledge. The framework consists of three key components: a frame prediction network that models temporal information to predict expected outcomes in video sequences, an explicit knowledge embedding network that incorporates predefined relationships and rules to guide the model's understanding of typical behaviors, and an anomaly assessment module that evaluates predictions by comparing expected outcomes with actual observations. By leveraging cognition guidance, the CG-VAD framework effectively detects anomalies in diverse surveillance scenarios. The explicit knowledge component enhances detection capabilities by providing context-specific insights, while the implicit knowledge allows the model to adapt to unseen events based on patterns learned from extensive video data. This combination significantly improves generalization across various environments, reducing the risk of overfitting and enhancing the system's robustness. The framework's ability to assess prediction errors in real-time ensures immediate detection of unusual activities, making it a highly effective solution for video surveillance systems that require prompt responses to potential threats or incidents. Experimental results demonstrate that the proposed CG-VAD framework outperforms existing state-of-the-art methods, providing a robust solution for real-time anomaly detection in video surveillance applications.

## 2.3 YOLO-SF: YOLO for Fire Segmentation Detection

The increasing frequency of fires in urban environments underscores the urgent need for reliable fire detection systems, as existing algorithms often suffer from significant drawbacks, including missed detections, false positives, and low accuracy, which can compromise safety and emergency response effectiveness. To address these issues, we propose a novel segmentation detection algorithm called YOLO-SF, which integrates instance segmentation technology with the YOLOv7-Tiny object detection framework to enhance detection accuracy for fire incidents. The foundation of YOLO-SF lies in the creation of a comprehensive Fire Segmentation Dataset (FSD), comprising images that capture both fire and non-fire elements, enabling the model to learn intricate features that distinguish fire from other objects. By adopting the segmentation detection head from YOLO, we enhance the model's segmentation capabilities, allowing it to express finer details in fire imagery. The backbone network for YOLO-SF incorporates the MobileViTv2 module, which effectively reduces the number of parameters while maintaining robust feature extraction capability. Additionally, the Efficient Layer Aggregation Network (ELAN) is enhanced with the Convolutional Block Attention Module (CBAM) to broaden the model's receptive field and improve its focus on both channel and spatial information in fire images. To tackle inaccurate object positioning at the edges of fire images, we implement Varifocal Loss, which adjusts the loss function to emphasize challenging detections. The results show significant performance improvements over the existing YOLOv7-Tiny segmentation algorithm, with a precision increase of 5.9 % for box detection and 6.2% for mask detection, and recall improvements of 2.5% and 3.3%, respectively. The mean Average Precision (mAP) enhances by 4% for boxes and 6% for masks, and with a frame rate of 55.64 FPS, YOLO-SF meets the stringent requirements for real-time detection. Overall, the improved YOLO-SF algorithm demonstrates strong generalization performance and robustness, making it a reliable tool for fire detection in various applications.

## 2.4 Smart City Transportation: Deep Learning Ensemble Approach for Traffic Accident Detection

The dynamic and unpredictable nature of road traffic necessitates the development of effective accident detection methods to enhance safety and streamline traffic management in smart cities. In this paper, we provide a comprehensive exploration of prevailing accident detection techniques, shedding light on various state-of-the-art methodologies while offering a detailed overview of distinct traffic accident types, including rear-end collisions, T-bone collisions, and frontal impact accidents. To address the unique challenges of accident detection in urban environments, we introduce the I3D-CONVLSTM2D model architecture, a lightweight solution explicitly designed for accident detection in smart city traffic surveillance systems. This innovative model integrates RGB frames with optical flow information, enabling it to capture both spatial and temporal dynamics of traffic scenes effectively. Empirical analysis from our experimental study underscores the efficacy of the I3D-CONVLSTM2D architecture, with the RGB + Optical-Flow (trainable) model outperforming its counterparts by achieving an impressive 87% Mean Average Precision (MAP). Furthermore, our findings delve into the challenges posed by data imbalances, particularly when working with limited datasets, varying road structures, and diverse traffic scenarios, which can complicate the training process and affect detection accuracy. Ultimately, our research illuminates the path toward a sophisticated vision-based accident detection system that is primed for real-time integration into edge IoT devices within smart urban infrastructures, offering a significant step forward in improving road safety and enhancing traffic management efficiency.

## 2.5 A Secure, Reliable and Low-Cost Distributed Storage Scheme Based on Blockchain and IPFS for Firefighting IoT Data

Fire scene investigations rely on firefighting IoT data as key electronic evidence for event analysis and determining responsibility. In the traditional centralized storage setup, data is vulnerable to tampering and damage, undermining its reliability. As digital evidence plays a crucial role in investigations, a secure and tamper-proof system is essential. To address this, we propose a distributed storage scheme using blockchain technology, specifically the Hyperledger Fabric framework, integrated with IPFS and the Practical Byzantine Fault Tolerance (PBFT) algorithm. In this scheme, IPFS acts as off-chain storage for complete IoT data, while only the IPFS hash is recorded on the blockchain, reducing on-chain storage needs and enhancing data security. The Fabric framework, embedded with PBFT, ensures the reliability of consensus nodes, providing fault tolerance and improving blockchain availability. Security is further reinforced through AES and RSA encryption for data storage and transmission. Our system analysis and experimental results show that this solution meets the requirements for secure storage and traceability, offering greater storage efficiency, throughput, and lower latency compared to blockchain-only methods. The improved Fabric framework also supports Byzantine fault tolerance, enhancing data integrity and security for reliable fire scene investigations.

# Chapter 3

# Requirement Specification

## 3.1 Functional Requirements

- Anomaly Detection:The system must identify and classify anomalies like accidents, fires, and crimes in real-time from video feeds.
- Real-Time Video Analysis: The system must analyze video footage instantaneously to ensure prompt emergency detection.
- Blockchain Integration: The system must use blockchain technology for secure, tamper-proof evidence storage.
- Notification to Authorities:The system must automatically alert authorities upon detecting an anomaly for quick response.

## 3.2 Non-Functional Requirements

- Security: The system must protect sensitive data from unauthorized access.
- Scalability: The system must accommodate growing data and user demands without performance loss.
- Transparency: The system must ensure verifiable data handling and storage processes.

## 3.3 Software Requirements

- YOLO: The system must implement the YOLO object detection algorithm for real-time anomaly detection.
- IPFS: The system must utilize IPFS for decentralized and secure storage of video evidence.
- Twilio:Twilio API will be used to send real-time alert notifications to relevant authorities when an anomaly is detected.
- Frontend:CSS,JavaScript,HTML
- Backend:Python-Django,Solidity

## 3.4 Hardware Requirements

- CCTV Camera: High-resolution cameras are needed for capturing video footage in monitored areas.
- Computer: A powerful computer is required to process video data and run detection algorithms in real-time.

# Chapter 4

# Proposed system and Design

## 4.1 Proposed system

The AI Enhanced Public Monitoring System integrates advanced technologies for real-time anomaly detection and secure evidence management in urban environments. High-resolution CCTV cameras capture video footage, which is processed by a powerful computer running the YOLO algorithm for immediate identification of anomalies such as accidents, fires, and crimes. To enhance detection, the system employs Optical Flow techniques to analyze motion patterns. Upon detecting an anomaly, it automatically alerts relevant authorities for prompt response. Captured video footage is stored securely using IPFS, ensuring decentralized and tamper-proof evidence management. The integration of blockchain technology further guarantees data integrity, providing an immutable record throughout the evidence lifecycle. Designed for scalability, the system can accommodate growing data and user demands, ultimately enhancing public safety while ensuring secure handling of sensitive information in smart urban environments.

## 4.2 Feasibility Study

The feasibility study for the AI Enhanced Public Monitoring System evaluates its technical, operational, and economic viability. Technically, the integration of advanced technologies such as YOLO for object detection and IPFS for decentralized storage ensures that the system can meet real-time processing requirements, with readily available hardware like high-resolution CCTV cameras and powerful computers capable of supporting its demands. Operationally, the system is designed to enhance public safety by automating anomaly detection and reporting, which reduces reliance on human monitoring and allows for faster emergency response times, making it suitable for deployment in urban areas and smart cities; training personnel will be straightforward due to the user-friendly interface. Economically, the initial investment in hardware and software is justified by potential cost savings from reduced emergency response times and enhanced public safety, while integrating the system into existing infrastructure minimizes additional expenses. Overall, the AI Enhanced Public Monitoring System is deemed feasible across technical, operational, and economic dimensions, making it a valuable addition to modern urban safety initiatives.

### 4.2.1 Technical Feasibility

The technical feasibility of the AI Enhanced Public Monitoring System relies on the integration of advanced technologies that ensure efficient performance and reliability. Utilizing YOLO for real-time object detection, the system can accurately identify anomalies like accidents, fires, and crimes from high-resolution CCTV camera feeds. Powerful computers equipped with sufficient GPU resources will meet the processing demands of these algorithms. Additionally, IPFS enables secure, decentralized storage of video evidence, ensuring data integrity, while blockchain technology provides an immutable record of all stored evidence for verification. This combination supports real-time processing and scalability, allowing the system to handle growing data volumes in urban environments. Overall, the proposed system demonstrates strong technical feasibility for implementation in smart city initiatives.

### 4.2.2 Operational Feasibility

The operational feasibility of the AI Enhanced Public Monitoring System focuses on its capacity to enhance public safety through automation and efficient processes. By utilizing advanced technologies like YOLO and IPFS, the system can effectively detect anomalies in real time, enabling quicker responses to emergencies. This automation reduces reliance on human monitoring, which can be prone to error and fatigue, ensuring more reliable surveillance. The user-friendly interface will facilitate straightforward training for personnel, allowing for rapid adaptation and integration into current operations. Moreover, the system is designed to be scalable, accommodating the growing demands of urban environments and smart city initiatives. Overall, the operational feasibility assessment confirms that the proposed system can significantly improve safety and efficiency while being easily integrated into existing workflows.

### 4.2.3 Economic Feasibility

The economic feasibility of the AI Enhanced Public Monitoring System is supported by the potential for significant cost savings and long-term benefits. The initial investment in hardware, including high-resolution CCTV cameras and powerful computers, is justified by the anticipated reduction in emergency response times and enhanced public safety. By automating anomaly detection and reporting, the system reduces the reliance on manual monitoring, lowering operational costs associated with personnel. Additionally, integrating the system into existing infrastructure minimizes additional expenses, ensuring a cost-effective deployment. Long-term benefits include potential reductions in crime rates, which can lead to decreased costs related to crime prevention and response. Overall, the economic analysis indicates that the investment in the system will yield substantial returns in terms of safety, efficiency, and reduced operational costs over time.

## 4.3 Design

### 4.3.1 Architecture Diagram

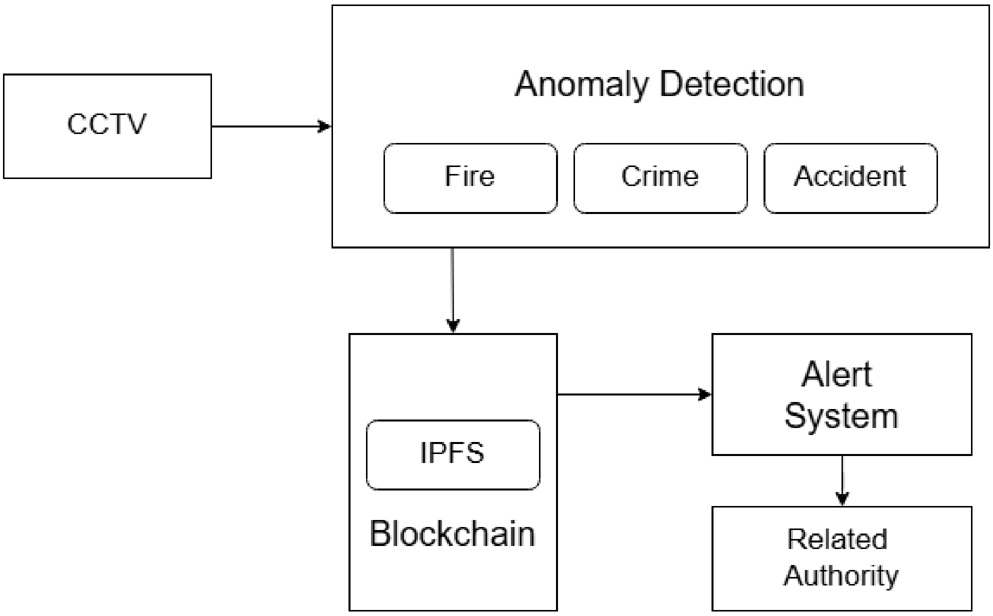


Figure 4.1: Architecture Diagram

### 4.3.2 Use Case Diagram

The diagram describes a surveillance-based platform for real-time detection of accidents, crimes, and fires. Surveillance cameras provide live video data to the system, which then analyzes the footage to detect incidents such as accidents, criminal activities, and fire hazards. Once an event is detected, the system securely stores the incident data using blockchain technology, ensuring data integrity and security. In addition, the platform features a notification system that sends alerts to relevant authorities, including hospitals, fire and rescue teams, and police stations. Overall, the platform offers a comprehensive solution for monitoring, detecting, storing, and responding to emergency situations, enhancing public safety and response efficiency.

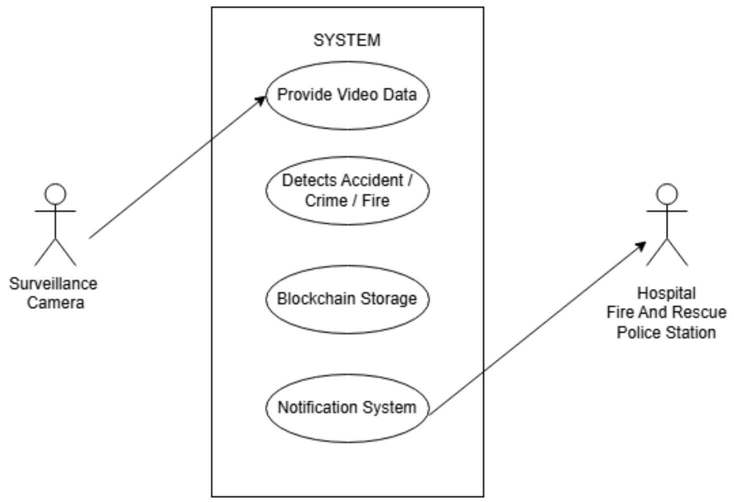


Figure 4.2: Use-Case Diagram

### 4.3.3 Data Flow Diagram

The diagram describes a surveillance-based system for detecting and responding to critical events such as accidents, crimes, and fire risks in real-time. The process begins with a video stream captured by surveillance cameras, which is sent for video preprocessing. This preprocessing stage improves the quality of the video frames, making them suitable for further analysis. The preprocessed frames are then fed into a feature extraction module, where key characteristics or features are identified.

Once features are extracted, they are analyzed by separate modules designed to detect specific entities or risks: human detection, vehicle detection, and fire risk detection. These detected entities and risk factors are then evaluated by an event rule matching system, which applies predefined rules to determine if an event of interest, such as a crime, accident, or fire, has occurred.

Upon detecting a critical event, the system generates an alert message that is directed to the required authority, such as police, fire department, or medical services, depending on the type of incident. Additionally, the detected data is securely stored on a blockchain, ensuring data integrity and providing a tamper-proof record of events. Overall, the system offers a robust pipeline for real-time detection, secure storage, and alert notification, enabling rapid response to various public safety incidents.

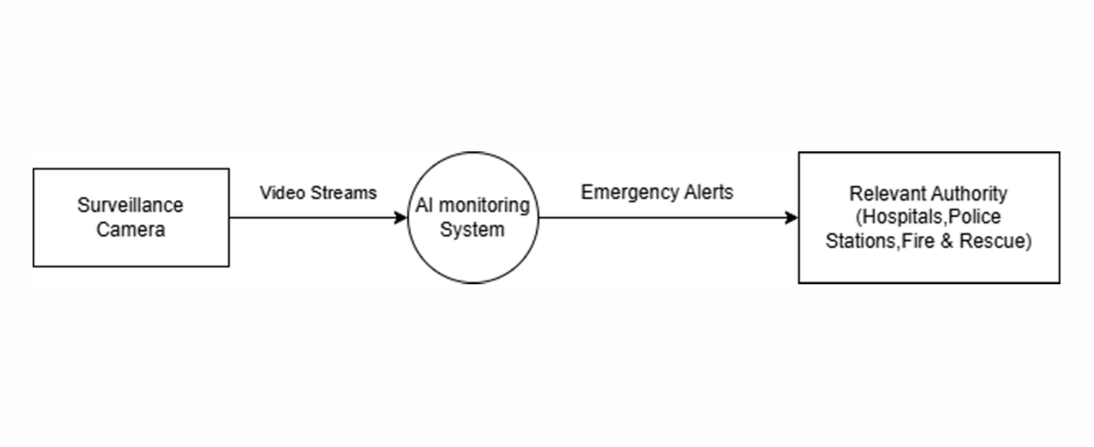


Figure 4.3: DFD Level0

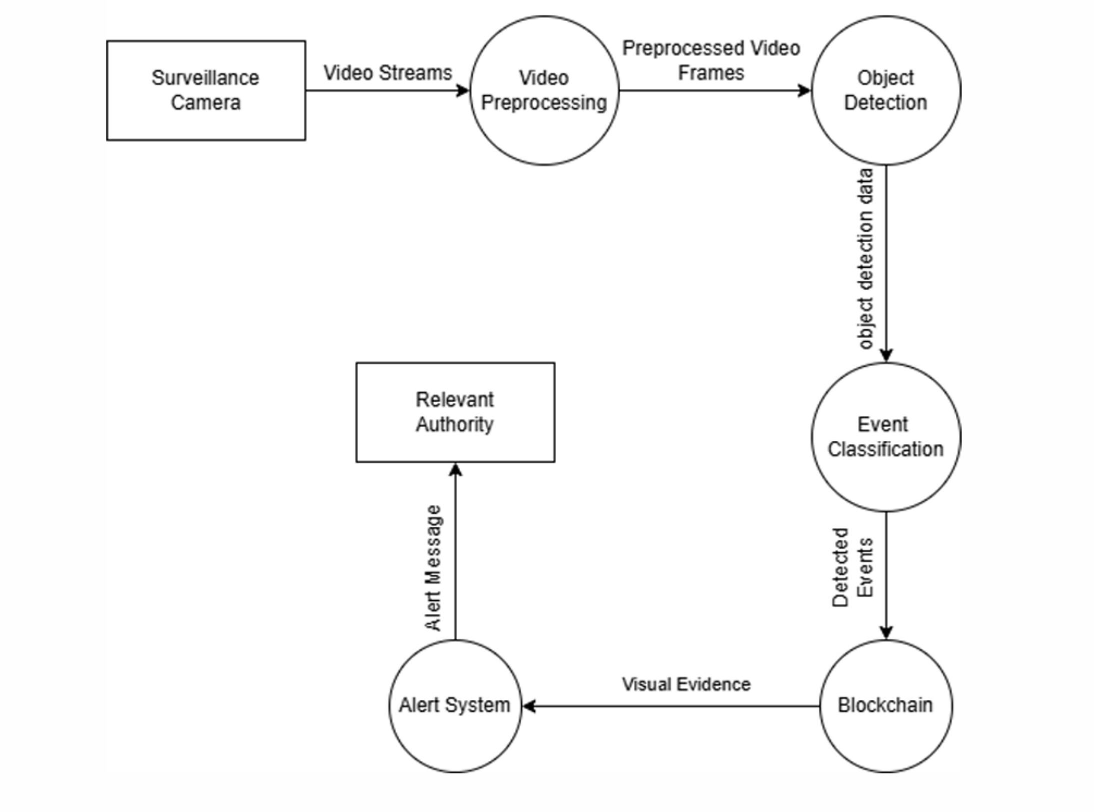


Figure 4.4: DFD Level1

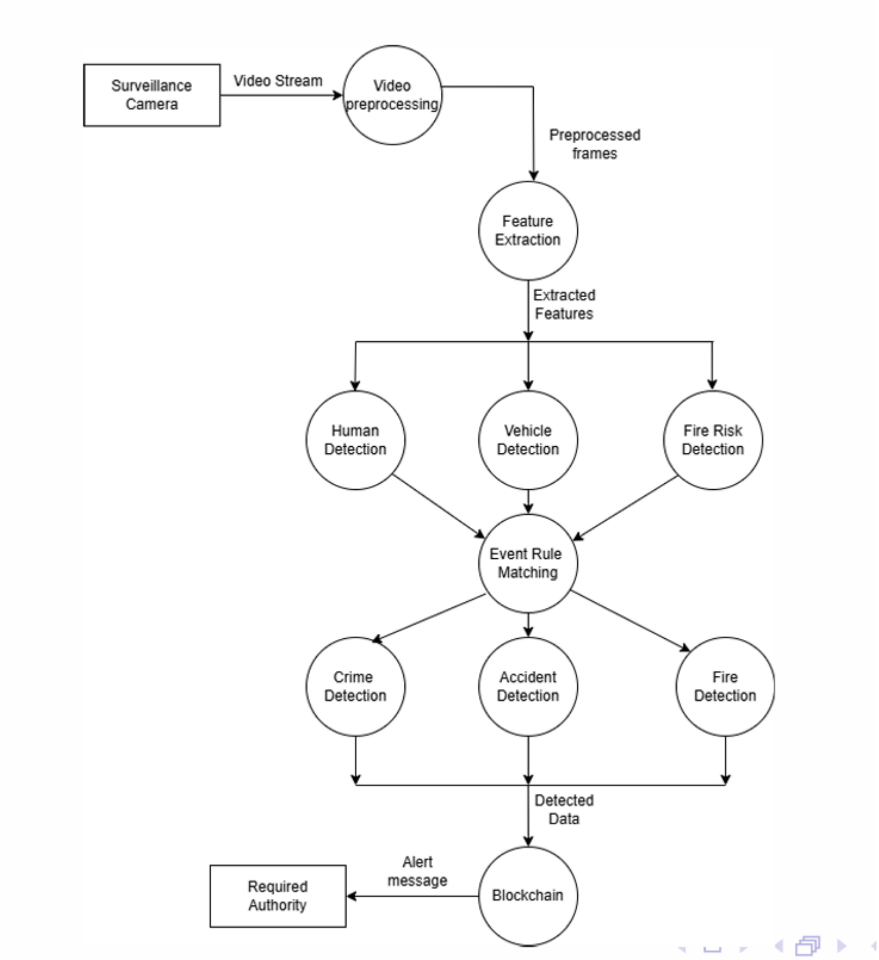


Figure 4.5: DFD Level2

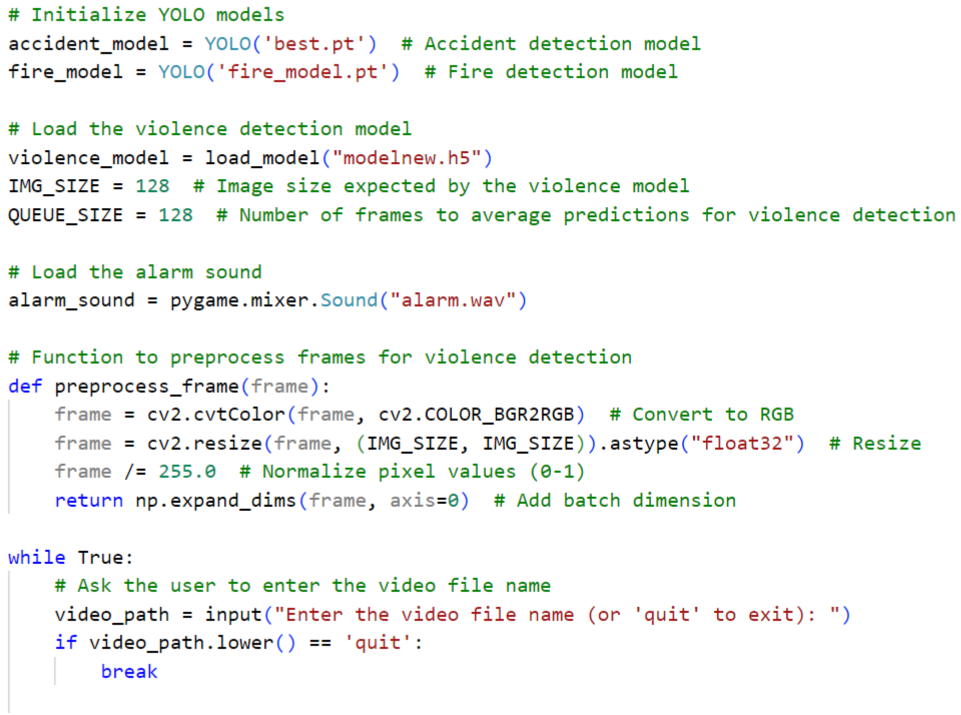
# Chapter 5

# Implementation

## 5.1 Implementation details

The ”AI Enhanced Surveillance System” integrates deep learning models (YOLOv5, CNN, and LSTM) for real-time detection of critical events—fire, crime (violence), and accidents—from live CCTV footage. The system stores the detected incident clips securely in IPFS (InterPlanetary File System) and registers their metadata on the Ethereum blockchain through a smart contract. A Flask-based frontend displays detected events and alerts, while the Node.js backend ensures secure blockchain transactions.

### 5.1.1 Loading pre trained object detection model

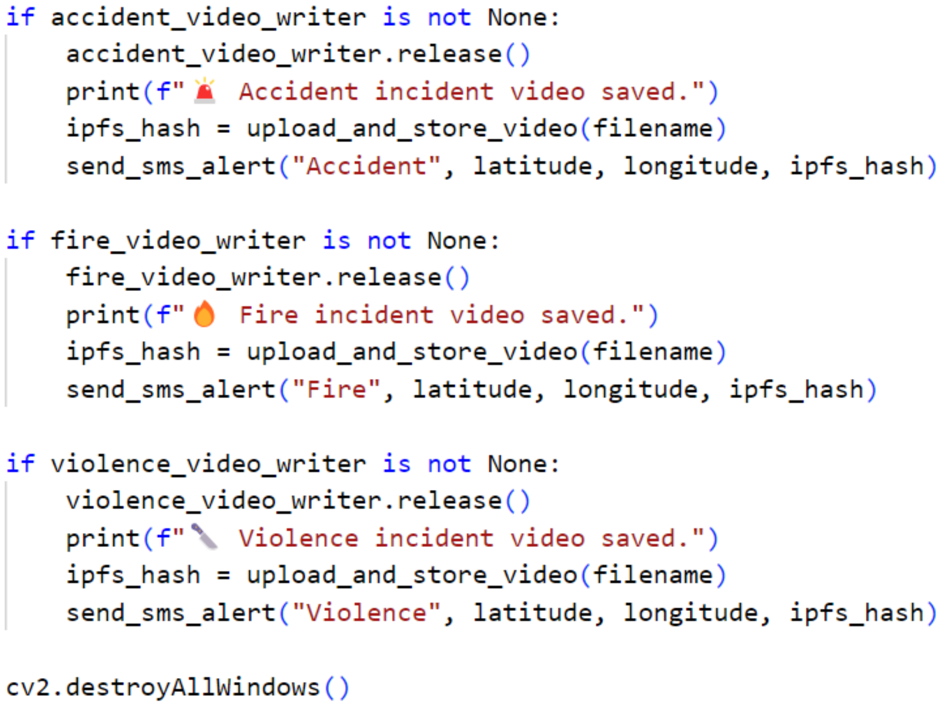


```
# Initialize YOLO models
accident_model = YOLO('best.pt')  # Accident detection model
fire_model = YOLO('fire_model.pt')  # Fire detection model

# Load the violence detection model
violence_model = load_model("modelnew.h5")
IMG_SIZE = 128  # Image size expected by the violence model
QUEUE_SIZE = 128  # Number of frames to average predictions for violence detection

# Load the alarm sound
alarm_sound = pygame.mixer.Sound("alarm.wav")

# Function to preprocess frames for violence detection
def preprocess_frame(frame):
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)  # Convert to RGB
    frame = cv2.resize(frame, (IMG_SIZE, IMG_SIZE)).astype("float32")  # Resize
    frame /= 255.0  # Normalize pixel values (0-1)
    return np.expand_dims(frame, axis=0)  # Add batch dimension

while True:
    # Ask the user to enter the video file name
    video_path = input("Enter the video file name (or 'quit' to exit): ")
    if video_path.lower() == 'quit':
        break
```

Figure 5.1: Load Model Image

The code represents the initialization and preprocessing stage of the multi-event detection system developed in this project. At this stage, various deep learning models are loaded, including YOLO models for detecting accidents and fires, and a violence detection model built using a combination of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks. These models are designed to identify both spatial features and temporal patterns in video footage. The system sets essential parameters such as image size and the number of frames to consider for stable predictions. An alarm sound is also loaded to provide real-time alerts when any critical event is detected. Additionally, a preprocessing function is defined to convert, resize, normalize, and format incoming video frames to match the input requirements of the models. The script includes a user-interactive loop that allows video file paths to be entered for analysis, serving as the entry point for real-time event detection from surveillance or recorded footage.

### 5.1.2 Perform prediction based on pre trained model



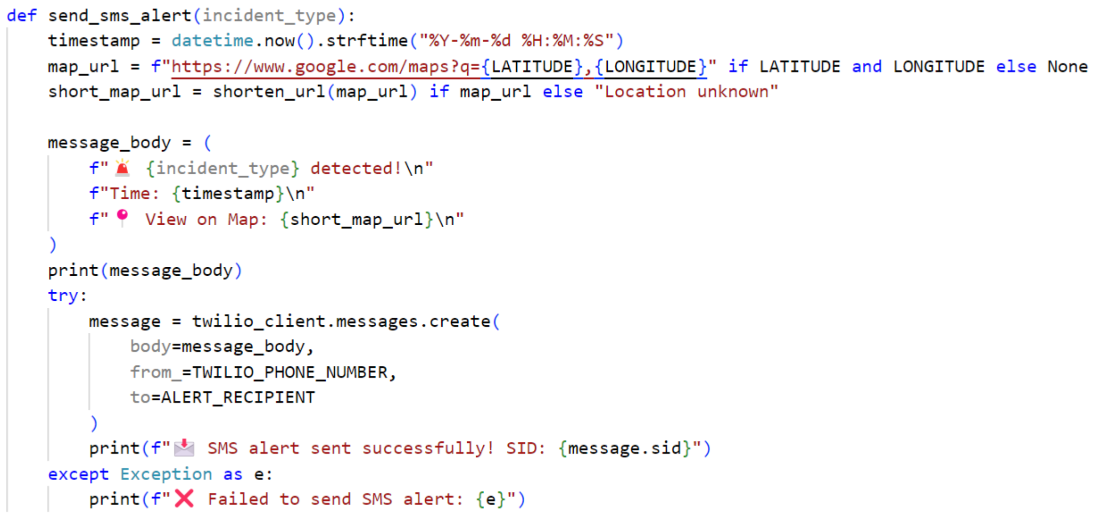
```
if accident_video_writer is not None:
    accident_video_writer.release()
    print(f"🚨 Accident incident video saved.")
    ipfs_hash = upload_and_store_video(filename)
    send_sms_alert("Accident", latitude, longitude, ipfs_hash)

if fire_video_writer is not None:
    fire_video_writer.release()
    print(f"🔥 Fire incident video saved.")
    ipfs_hash = upload_and_store_video(filename)
    send_sms_alert("Fire", latitude, longitude, ipfs_hash)

if violence_video_writer is not None:
    violence_video_writer.release()
    print(f"🔪 Violence incident video saved.")
    ipfs_hash = upload_and_store_video(filename)
    send_sms_alert("Violence", latitude, longitude, ipfs_hash)

cv2.destroyAllWindows()
```

Figure 5.2: Detection Image

The code illustrates the final stage of incident handling in the multi-event detection system. It shows how the system processes and stores video footage after detecting specific events such as accidents, fires, or violence. Once an incident is confirmed, the respective video writer object is safely released, ensuring that the recorded footage is properly saved. The system then uploads the saved video to a decentralized storage platform (likely using IPFS) and retrieves a unique hash for the file. This hash is used to reference the stored video when sending real-time SMS alerts, which include the type of incident along with the location coordinates (latitude and longitude). This mechanism ensures secure video storage and instant notification, enhancing the reliability and responsiveness of the overall surveillance system.

### 5.1.3 Sending Alert Notification

The code displays the function responsible for sending SMS alerts in the event detection system. This function generates and dispatches real-time notification

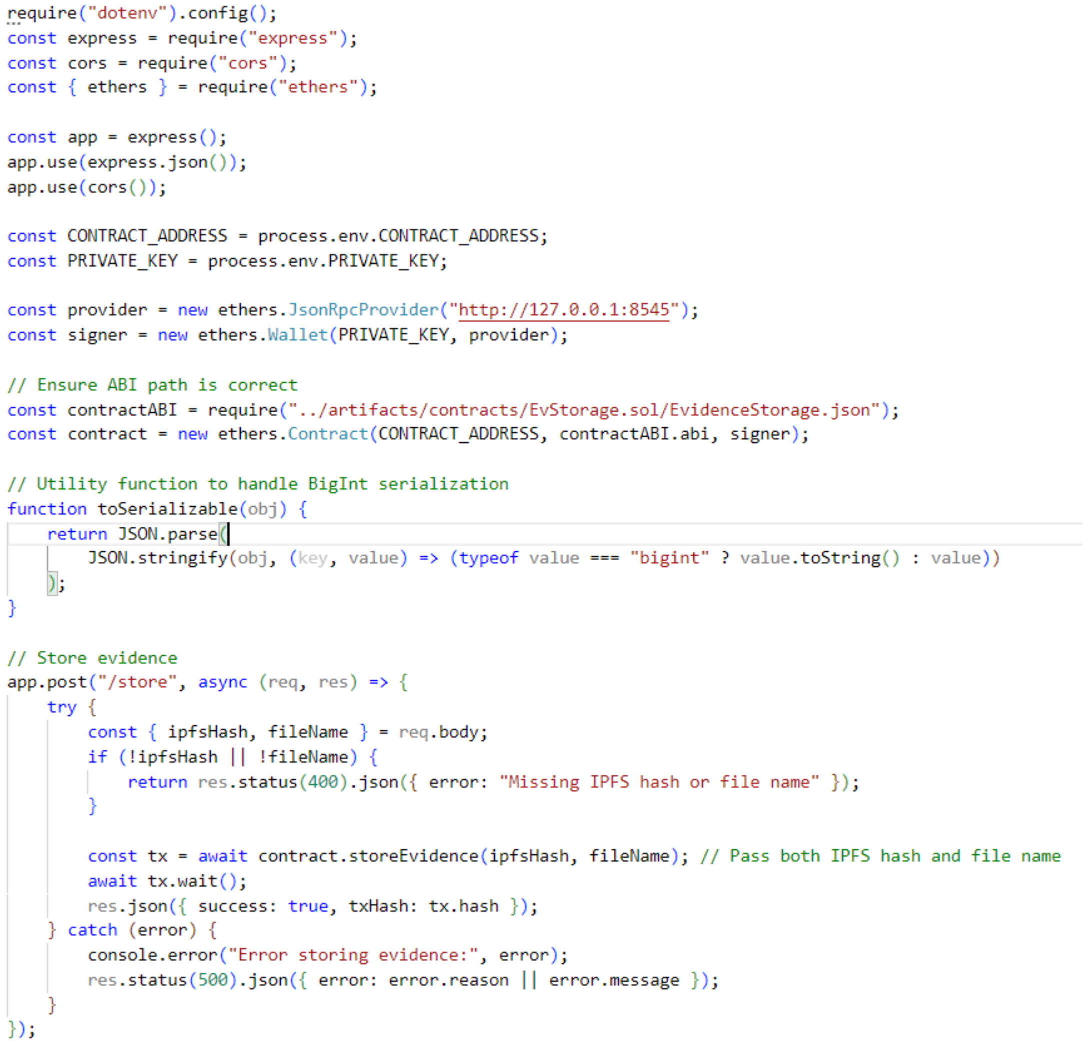


```python
def send_sms_alert(incident_type):
    timestamp = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    map_url = f"https://www.google.com/maps?q={LATITUDE},{LONGITUDE}" if LATITUDE and LONGITUDE else None
    short_map_url = shorten_url(map_url) if map_url else "Location unknown"

    message_body = (
        f"🚨 {incident_type} detected!\n"
        f"Time: {timestamp}\n"
        f"📍 View on Map: {short_map_url}\n"
    )
    print(message_body)
    try:
        message = twilio_client.messages.create(
            body=message_body,
            from_=TWILIO_PHONE_NUMBER,
            to=ALERT_RECIPIENT
        )
        print(f"📩 SMS alert sent successfully! SID: {message.sid}")
    except Exception as e:
        print(f"❌ Failed to send SMS alert: {e}")
```

Figure 5.3: Alert Code

messages to alert recipients when incidents such as accidents, fires, or violence are detected. It constructs a message body that includes the type of incident, the exact timestamp of detection, and a Google Maps link showing the location of the event (based on latitude and longitude data). The map URL is optionally shortened for better readability. Using the Twilio API, the system sends this message to a predefined recipient number. If the message is successfully sent, a confirmation with the message SID is printed; otherwise, any exceptions encountered during the process are caught and displayed. This alert mechanism ensures that emergency responses can be triggered quickly and with accurate situational awareness.

### 5.1.4 Back-End

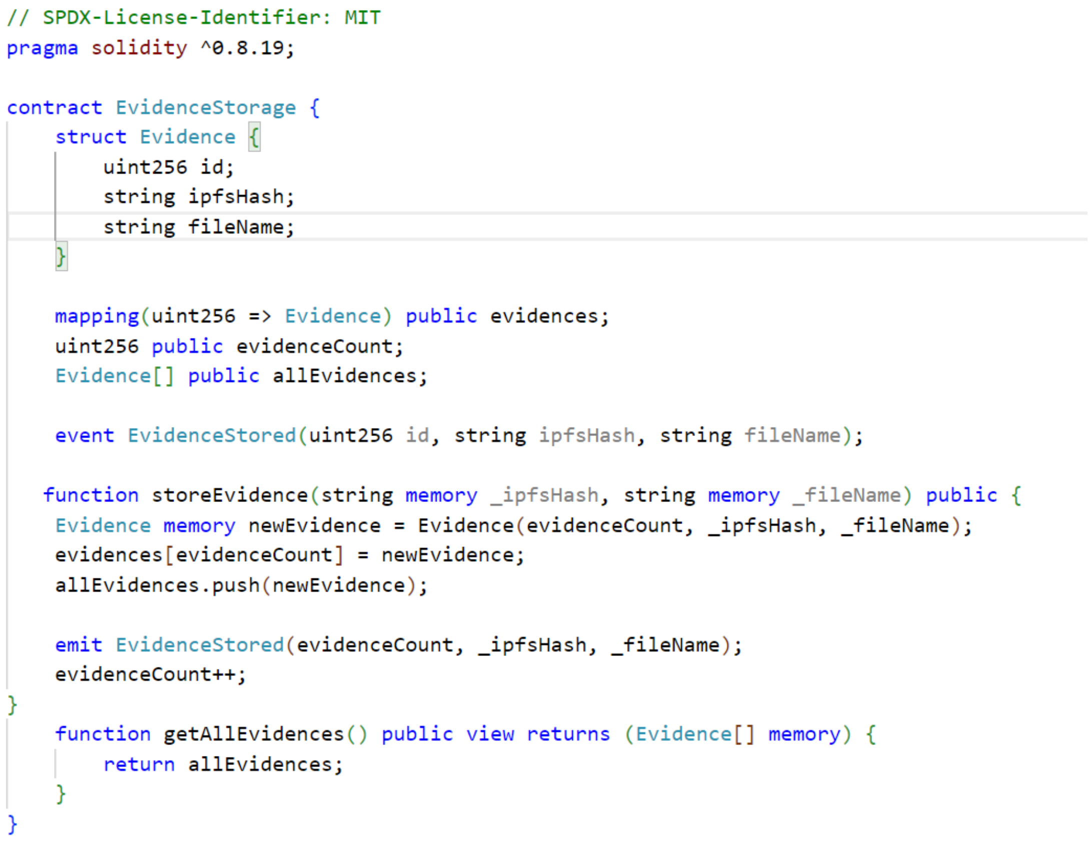


```
require("dotenv").config();
const express = require("express");
const cors = require("cors");
const { ethers } = require("ethers");

const app = express();
app.use(express.json());
app.use(cors());

const CONTRACT_ADDRESS = process.env.CONTRACT_ADDRESS;
const PRIVATE_KEY = process.env.PRIVATE_KEY;

const provider = new ethers.JsonRpcProvider("http://127.0.0.1:8545");
const signer = new ethers.Wallet(PRIVATE_KEY, provider);

// Ensure ABI path is correct
const contractABI = require("../artifacts/contracts/EvStorage.sol/EvidenceStorage.json");
const contract = new ethers.Contract(CONTRACT_ADDRESS, contractABI.abi, signer);

// Utility function to handle BigInt serialization
function toSerializable(obj) {
    return JSON.parse(
        JSON.stringify(obj, (key, value) => (typeof value === "bigint" ? value.toString() : value))
    );
}

// Store evidence
app.post("/store", async (req, res) => {
    try {
        const { ipfsHash, fileName } = req.body;
        if (!ipfsHash || !fileName) {
            return res.status(400).json({ error: "Missing IPFS hash or file name" });
        }

        const tx = await contract.storeEvidence(ipfsHash, fileName); // Pass both IPFS hash and file name
        await tx.wait();
        res.json({ success: true, txHash: tx.hash });
    } catch (error) {
        console.error("Error storing evidence:", error);
        res.status(500).json({ error: error.reason || error.message });
    }
});
```

Figure 5.4: Back End Image

The image illustrates the backend implementation of a blockchain-based evidence storage system using Node.js, Express, and Ethereum's smart contract interaction via the Ethers.js library. This backend is responsible for receiving evidence data, such as IPFS hashes and file names, and securely storing them on the blockchain. Environment variables are used to securely load the contract address and private key, while a local Ethereum provider is configured for blockchain interaction. The smart contract's ABI (Application Binary Interface) is imported to facilitate interaction with deployed contract functions. A utility function is also included to handle serialization of BigInt values for compatibility with JSON formatting. The Express route /store listens for POST requests, extracts the necessary data from the request body, and calls the storeEvidence function in the smart contract to immutably record the evidence. The response includes the transaction hash if successful, or a detailed error message if the operation fails. This backend module ensures that digital evidence is not only captured but also securely and transparently stored on the blockchain for future verification.

### 5.1.5 Blockchain



```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.19;

contract EvidenceStorage {
    struct Evidence {
        uint256 id;
        string ipfsHash;
        string fileName;
    }

    mapping(uint256 => Evidence) public evidences;
    uint256 public evidenceCount;
    Evidence[] public allEvidences;

    event EvidenceStored(uint256 id, string ipfsHash, string fileName);

   function storeEvidence(string memory _ipfsHash, string memory _fileName) public {
    Evidence memory newEvidence = Evidence(evidenceCount, _ipfsHash, _fileName);
    evidences[evidenceCount] = newEvidence;
    allEvidences.push(newEvidence);

    emit EvidenceStored(evidenceCount, _ipfsHash, _fileName);
    evidenceCount++;
}
    function getAllEvidences() public view returns (Evidence[] memory) {
        return allEvidences;
    }
}
```

Figure 5.5: Block Chain

The code showcases a Solidity smart contract named EvidenceStorage designed to store and manage digital evidence on the Ethereum blockchain. This contract defines a structured data type, Evidence, which holds an evidence ID, an IPFS hash for file storage, and the file name. It uses a mapping to associate each evidence entry with a unique ID and maintains a counter to track the number of entries. Additionally, an array stores all evidence records for easy retrieval. The storeEvidence function enables the addition of new evidence by accepting the IPFS hash and file name as inputs, then storing the data on-chain and emitting an EvidenceStored event for transparency and traceability. The getAllEvidences function allows users to fetch the entire collection of stored evidence. This contract ensures secure, immutable storage of digital proofs using blockchain technology, promoting data integrity and accessibility.

## 5.2 Modules Used

The proposed system consists of several interconnected modules working together to create an AI-powered surveillance system. The system leverages computer vision (YOLO), deep learning models (CNN + LSTM), and decentralized storage (IPFS) to monitor real-time visuals from CCTV cameras and generate alerts during abnormal activities such as violence or accidents. The major modules used in this system are elaborated below

### 5.2.1 Video Input Module (Live Surveillance Feed)

The Video Input Module is responsible for capturing live footage from CCTV cameras or webcam sources. It continuously reads frames using OpenCV, an open-source computer vision library. This module acts as the starting point of the surveillance pipeline, ensuring that every frame from the video stream is fed into the system in real-time. It plays a crucial role in maintaining the frame rate and resolution necessary for accurate detection, thus forming the foundation for further processing steps.

### 5.2.2 Model Loading and Initialization Module

This module handles the loading of pre-trained models necessary for detecting violence or abnormal events. It typically involves deep learning models like YOLO (You Only Look Once) for object detection and CNN-LSTM for activity recognition. The model loading process includes reading model weights, configuration files, and setting up necessary computation environments (e.g., GPU/CPU setup). By ensuring the model is properly initialized, this module prepares the system for accurate inference on incoming video data.

### 5.2.3 Detection and Classification Module

The Detection and Classification Module performs the core task of identifying abnormal events from the video stream. It uses YOLO to detect and localize objects within each frame and employs a combination of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks to analyze both spatial and temporal patterns. This hybrid approach enhances the system's ability to detect complex activities such as fights or accidents by understanding motion and object behavior over time.

### 5.2.4 Event Alert and Notification Module

Once an abnormal event is detected, the Event Alert and Notification Module takes over to inform the concerned personnel. This module is built using a backend framework like Flask, which handles the triggering and display of alerts. It can be integrated with various messaging APIs to send notifications via email, SMS, or other real-time communication channels. This ensures rapid dissemination of critical alerts, helping stakeholders respond quickly to incidents.

### 5.2.5 Decentralized Storage Module

To securely store evidence of detected events, the Decentralized Storage Module uploads the relevant video clips to IPFS (InterPlanetary File System). This module works by extracting short segments of footage from the video stream whenever an incident is confirmed and pushing them to IPFS using gateways like Pinata. Once uploaded, a unique hash (CID) is generated for each clip, ensuring immutability and transparency. This guarantees that event footage is tamper-proof and accessible for future verification.

### 5.2.6 Web Interface Module

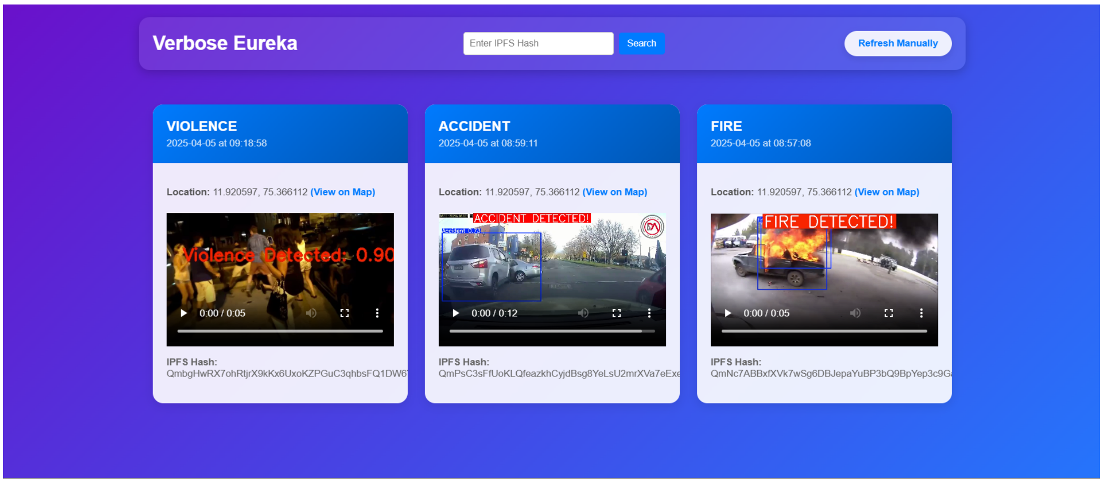
The Web Interface Module provides an interactive front-end for users to monitor the system's status and view alerts and stored clips. Built with HTML, CSS, and JavaScript, and supported by Flask on the backend, this module allows administrators to access the platform via a web browser. It displays real-time video feeds, detection logs, and links to IPFS-stored clips. This user-friendly dashboard enhances accessibility and control over the surveillance system.

# Chapter 6

# Result and Discussion

Each chapter is to begin with a brief introduction (in 4 or 5 sentences) about its contents. The contents can then be presented below organised into sections and subsections

## 6.1 Web Interface



**VIOLENCE**
2025-04-05 at 09:18:58

**Location:** 11.920597, 75.366112 **(View on Map)**

**IPFS Hash:**
QmbgHwRX7ohRtjrX9kKx6UxoKZPGuC3qhbsFQ1DW6

**ACCIDENT**
2025-04-05 at 08:59:11

**Location:** 11.920597, 75.366112 **(View on Map)**

**IPFS Hash:**
QmPsC3sFfUoKLQfeazkhCyjdBsg8YeLsU2mrXVa7eEx

**FIRE**
2025-04-05 at 08:57:08

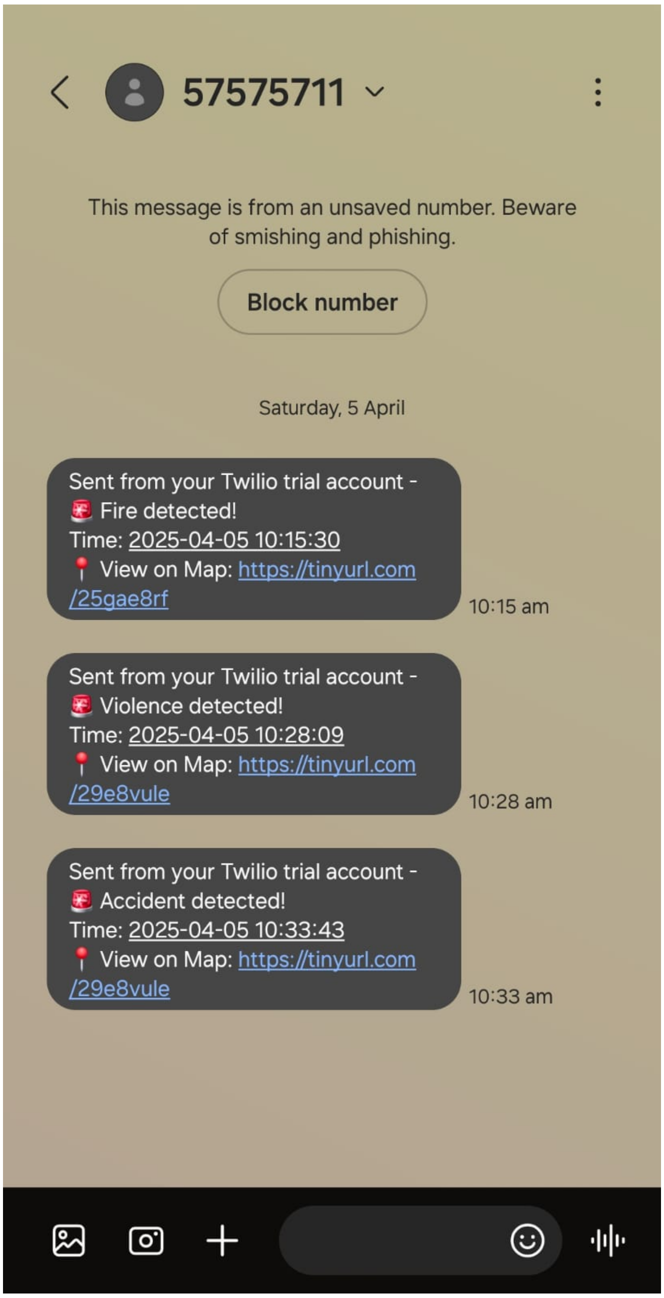
**Location:** 11.920597, 75.366112 **(View on Map)**

**IPFS Hash:**
QmNc7ABBxfXVk7wSg6DBJepaYuBP3bQ9BpYep3c9G

Figure 6.1: Web Image

The web interface serves as the user-facing portal of the AI-enhanced surveillance system, providing real-time access to crucial information such as live feeds, alert logs, and stored incident clips. Upon successful deployment, the interface is launched through a local or remote host address. The web page is designed to be intuitive, presenting users with clear options to navigate through the system's functions. At the top of the page, there's typically a navigation menu or a dashboard summary showcasing system health, number of active video feeds, and total alerts generated. The central part of the interface displays either the live stream from the CCTV camera or a list of recent detections, each associated with timestamps and short descriptions. When an incident is detected, the event is recorded and a hyperlink to the IPFS-hosted video is dynamically added to the web page, ensuring easy access to tamper-proof footage. The layout is developed using HTML and CSS, styled for readability and responsiveness across devices. Behind the scenes, Flask handles server-side operations and routing, ensuring seamless integration between the detection backend and user frontend. Overall, the interface significantly enhances system usability by ensuring that stakeholders can conveniently monitor and verify surveillance results in real-time.

## 6.2 Alert Message



57575711

This message is from an unsaved number. Beware of smishing and phishing.

Saturday, 5 April

Sent from your Twilio trial account -
🚨 Fire detected!
Time: 2025-04-05 10:15:30
📍 View on Map: https://tinyurl.com/25gae8rf
10:15 am

Sent from your Twilio trial account -
🚨 Violence detected!
Time: 2025-04-05 10:28:09
📍 View on Map: https://tinyurl.com/29e8vule
10:28 am

Sent from your Twilio trial account -
🚨 Accident detected!
Time: 2025-04-05 10:33:43
📍 View on Map: https://tinyurl.com/29e8vule
10:33 am

Figure 6.2: Notification Image

When an abnormal activity such as a fight or accident is detected by the system, a notification is automatically triggered and sent to the web portal. This real-time alerting mechanism is a crucial component of the surveillance system, aiming to reduce response time during emergencies. The notification message typically includes a clear warning such as “ALERT: Fight Detected!” or “Accident Detected!”, accompanied by the exact timestamp of the incident. Below the alert text, a hyperlink is provided that redirects the user to the IPFS location of the recorded clip. This ensures that authorized personnel can immediately review the footage for verification and further action. The alert is color-coded—often in red or orange—to emphasize its urgency and catch the viewer’s attention. The design of the notification panel is minimal yet effective, enabling clear visibility even in multi-alert scenarios. These messages are dynamically generated using backend logic, and the frontend is automatically refreshed or updated to display the latest notifications. This module not only ensures timely communication of threats but also serves as a log for historical data, which can be revisited for investigative or audit purposes. The integration of blockchain storage adds further trust, making the alerts both timely and verifiable.

# Chapter 7

# Conclusion

The report on the “AI-Enhanced Public Monitoring System” demonstrates that the project effectively addresses the challenges outlined in the problem statement. This system offers a comprehensive solution for public safety by enabling real-time detection and response to incidents such as accidents, fires, and crimes. Utilizing the YOLO algorithm, the system accurately and efficiently identifies key objects and activities, ensuring timely recognition of potential emergencies in monitored areas. Optical Flow further enhances detection capabilities by analyzing motion patterns, enabling the identification of sudden or unusual movements that may signal emergencies.To maintain data integrity and ensure tamper-proof evidence, blockchain technology is integrated into the system, providing a secure and immutable record of events. This feature is essential for legal or investigational purposes, as it upholds the trustworthiness of recorded data. Additionally, the system uses IPFS for decentralized storage, which eliminates reliance on traditional databases, providing both scalability and enhanced security.The AI-Enhanced Public Monitoring System is designed to operate autonomously, providing an efficient, scalable, and proactive monitoring solution suited for urban environments, smart cities, and high-risk areas. By automating anomaly detection, classification, and alerting, the system enables a faster response to incidents and reduces dependency on manual surveillance.

# References

[1] M. Khan, A. E. Saddik, W. Gueaieb, G. De Masi, and F. Karray, “Vd-net: An edge vision-based surveillance system for violence detection,” *IEEE Access*, vol. 12, pp. 43 796–43 808, 2024.

[2] M. Zhang, J. Wang, Q. Qi, Z. Zhuang, H. Sun, and J. Liao, “Cognition guided video anomaly detection framework for surveillance services,” *IEEE Transactions on Services Computing*, vol. 17, no. 5, pp. 2109–2123, 2024.

[3] X. Cao, Y. Su, X. Geng, and Y. Wang, “Yolo-sf: Yolo for fire segmentation detection,” *IEEE Access*, vol. 11, pp. 111 079–111 092, 2023.

[4] V. A. Adewopo and N. Elsayed, “Smart city transportation: Deep learning ensemble approach for traffic accident detection,” *IEEE Access*, vol. 12, pp. 59 134–59 147, 2024.

[5] L. Li, D. Jin, T. Zhang, and N. Li, “A secure, reliable and low-cost distributed storage scheme based on blockchain and ipfs for firefighting iot data,” *IEEE Access*, vol. 11, pp. 97 318–97 330, 2023.