RESQRADAR - ACCIDENT DETECTION SYSTEM

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

Road traffic accidents are a leading cause of injuries and fatalities worldwide, especially in countries with high vehicle density and underdeveloped traffic monitoring systems. Timely accident detection and quick emergency response are essential to save lives and reduce the severity of injuries. Most existing systems depend on GPS, accelerometers, or mobile networks, which can be limited by cost, connectivity issues, and hardware dependencies.

RESQRADAR – Accident Detection System introduces a novel, vision-based approach to detecting accidents using **video cameras** instead of GPS or mobile sensors. This system continuously analyzes live video feeds to identify sudden collisions, unusual vehicle behavior, or crash-related events in real time. By applying **computer vision algorithms**, RESQRADAR can effectively detect crash incidents without requiring internet connectivity or satellite-based location tracking.

Upon detecting an accident, the system instantly triggers alerts that can notify nearby authorities, emergency services, or control room operators. This real-time responsiveness enhances the chances of providing timely help, especially in areas where conventional GPS-based systems might fail due to signal loss or delayed processing.

The key advantage of RESQRADAR lies in its **affordability, scalability, and independence from GPS**. It is designed to be implemented in urban, semi-urban, and rural environments using low-cost hardware and open-source software. This makes it a practical solution for improving road safety across diverse locations and conditions.

In summary, RESQRADAR aims to provide a reliable, efficient, and intelligent accident detection mechanism that bridges the gaps left by traditional methods, making roads safer and responses faster.

1.2 PROBLEM STATEMENT

Road accidents are a growing public safety concern, contributing to a large number of injuries and fatalities globally. One of the most critical factors in reducing the severity of accident outcomes is **timely detection and swift emergency response**. However, in many cases, there is a significant delay between when an accident occurs and when help arrives—primarily due to the **lack of automated, real-time alert systems**.

- ❖ Most existing accident detection solutions rely on **GPS, accelerometers, or mobile network-based sensors**, which pose several challenges:

- ❖ They are dependent on **network availability**, making them unreliable in **rural or low-signal areas**

Moreover, current systems are prone to **false positives**, lack contextual awareness, and depend on user intervention or manual alerting, which is not feasible in severe crashesTo address these limitations, there is a need for a **cost-effective, intelligent accident detection system** that:

- ❖ **Uses video cameras and computer vision** to detect accidents accurately in real time.

- ❖ **Operates independently of GPS or mobile sensors**, making it suitable for any vehicle or roadside environment.

- ❖ **Automatically sends alerts to key emergency services**, including hospitals, ambulance drivers, police stations, and control rooms, to ensure a coordinated and immediate rescue operation.The **RESQRADAR** system proposes a solution to this problem by introducing a **vision-based accident detection mechanism** that instantly triggers alerts and accelerates emergency response—ultimately **reducing response times, saving lives, and improving public safety infrastructure**.

1.3 OBJECTIVES

The primary goal of the **RESQRADAR – Accident Detection System** is to enhance road safety by providing a reliable, intelligent, and immediate response mechanism for road accidents. The system leverages **video cameras and computer vision** to detect accidents in real time and **automatically notify emergency services**, without relying on GPS or mobile networks.

The key objectives of the project are:

- **To develop a real-time, video-based accident detection system**

  Build a system that continuously analyzes visual input from cameras to detect vehicle crashes using computer vision techniques such as motion analysis detection.
- **To eliminate dependency on GPS and mobile networks**

  Ensure the system is fully functional without requiring GPS signals or cellular data, making it suitable for remote areas or locations with poor connectivity.
- **To send immediate alerts to emergency responders upon crash detection**

  Automatically notify multiple emergency contacts—including hospitals, ambulance services, and police stations—within seconds of an accident being detected.
- **To reduce emergency response time and increase survival chances**

  Enable faster medical and rescue intervention by delivering timely and accurate crash notifications directly to relevant authorities.
- **To minimize false positives using intelligent detection algorithms**

  Improve detection accuracy through advanced visual analysis to avoid unnecessary alerts and ensure reliable system performance.
- **To create a cost-effective and scalable solution**

  Design the system using affordable components and open-source tools to allow easy implementation across various environments—urban or rural.
- **To promote safety in both personal and public transportation systems**

  Make the technology adaptable to various use cases such as highways, public roads, school zones, and fleet vehicles.

1.4 ORGANIZATION OF THE REPORT

This thesis report is organized into the following chapters to provide a comprehensive view of the project:

- **Chapter 2 – Literature Survey**
  This chapter provides a review of existing accident detection and emergency response systems. It explores current technologies used in vehicular safety, evaluates their strengths and limitations, and identifies the need for an advanced and responsive solution like RESQRADAR.

- **Chapter 3 – Methodology**
  This chapter details the design, architecture, and technologies employed in developing RESQRADAR. It includes an overview of sensors, microcontrollers, communication modules, and algorithms used for real-time accident detection and response.

- **Chapter 4 – Implementation**
  This chapter presents the development process of the RESQRADAR system. It covers the integration of hardware components, system programming, real-time data processing, and the user interface. It also includes testing procedures, performance analysis, and visual documentation.

- **Chapter 5 – Conclusion**
  This chapter summarizes the key outcomes of the RESQRADAR project. It reflects on the system's effectiveness in accident detection and emergency alerting, discusses the challenges encountered during development, and proposes future enhancements.

- **Chapter 6 – References**
  This chapter lists all the technical documents, research papers, tools, and libraries that supported the development of RESQRADAR. It serves as the reference base for the project.

CHAPTER 2

STUDY OF SURVEY

2.1 LITERATURE SURVEY

S. No	Title	Author(s) and Year	Journal/ Conference Name	Description
1.	Intelligent Expeditious Accident Detection andPrevention System	K P Sampoornam et & 2021	IOP Conference Series: Materials Science and Engineering (MSE)	The system detects and prevents accidents, alerting emergency services
2.	Accicent system and alert system	Bala adithya,Nalla naresh,K.Ninay kumar,G.Bala giri &2023	Journal of Engineering Sciences	The system detects accidents using sensors and GPS, sending real-time alerts to emergency services to improve response times and safet
3.	Accicent Identification & Alert system	Suraj Patil Kamesh Patil, Swapnil Dhabekar, Mahendra Nirgude, Prof. Shashikant Renushe&2020	International Research Journal of Modernization in Engineering Technology and Science	The system detects accidents, sends alerts with location, and allows user confirmation, all cost-effectively
4.	Accicent detection and alert system	G. Siri, G. Priyanka, B. Divya, G. Sameera, Ms. M. Pallavi & 2022	International Research Journal of Modernization in Engineering Technology and Science	The system detects accidents and sends location-based alerts to emergency services

5.	IoT-Based Smart Accident	C. V. Suresh Babu,Akshayah N. S,Maclin Vinola P,R. Janapriya & 2023	Research Gate	The system detects accidents and alerts emergency services.
6.	Smart Accident Detection and Alert System	Uma N, Saktheeswari R, Indumathi A, Sugacini M, Kavishree S & 2003	International Journal of Engineering Research & Technology	The system detects accidents and alerts emergency services with location details
7.	Accident Detection and Alert System using	Pranav Patil, Kishor Gangurde, Aniket Khairnar, Akash Mule4 and P. T. Suradkar & 2021	International Journal of Advanced Research in Science, Communicati on and Technology	The system detects accidents and alerts emergency services with location
8.	Vehicle tracking and accident alert system	Kommineni Rakesh & 2014	National Institute Of Technology Rourkela	GPS & GSM are key for vehicle tracking and emergency response

Table 2.1 Literature Survey for the Research Papers

2.2 LITERATURE REVIEW

2.2.1 Overview

The evolution of accident detection systems has predominantly centered around **GPS-based tracking, accelerometer sensors, and mobile applications**. These technologies, while effective in certain scenarios, suffer from critical limitations—particularly in areas with poor network coverage or in vehicles that lack built-in smart features.

Studies such as **Patel et al. (2019)** and **Kumar & Reddy (2021)** have explored sensor-based systems that use GPS and accelerometers to identify sudden deceleration and crash-like events. However, they note frequent issues with **false positives** and **poor reliability in low-connectivity areas**.

To address such limitations, recent research has shifted toward **computer vision and AI-based accident detection**. For example, **Zhao et al. (2022)** developed a deep learning model that detects traffic incidents using CCTV footage. Their work demonstrates how **video analytics** can outperform traditional sensor-based systems in accuracy and context understanding.

Further, **Singh and Sharma (2020)** highlight the potential of **automated alert systems** integrated with vision-based detection, capable of notifying emergency services instantly. However, most of these systems still rely on centralized infrastructure or are limited to developed regions with advanced surveillance setups.

Despite advancements, there remains a **gap in low-cost, camera-based solutions** that can function without GPS or mobile dependency and directly **notify hospitals, police, and ambulance drivers** in real-time.

RESQRADAR fills this gap by proposing a **vision-based accident detection system** that operates independently of network infrastructure and ensures **automated, multi-channel alerting**—especially useful in **resource-constrained or remote areas**.

2.2.2 Review Of Existing Accident Detection System

Several systems and applications have been developed to assist with accident detection and emergency alerting. These solutions vary in terms of technology, responsiveness, and practicality. The most notable among them include:

- **OnStar (by General Motors):**
  - **Features:** Built-in automatic crash detection using vehicle sensors, GPS, and airbag deployment signals. It provides instant connection to emergency services.
  - **Strengths:** Highly reliable in premium vehicles, directly integrated with vehicle systems, and supports voice communication.
  - **Limitations:** Limited to high-end vehicles; unavailable in low-cost or older models; relies on manufacturer integration.

- **iRA System (Hyundai):**
  - **Features:** Offers vehicle crash detection and emergency SOS features, connected via the in-vehicle infotainment system.
  - **Strengths:** Native integration, real-time response, includes location sharing and emergency calling.
  - **Limitations:** Only available in selected Hyundai models, not accessible for general public use or other car brands.

- **DigiSense (Mahindra):**
  - **Features:** A connected vehicle platform that includes accident detection, vehicle tracking, and alerts.
  - **Strengths:** Comprehensive fleet management solution, includes telematics and diagnostics.
  - **Limitations:** Primarily focused on commercial and fleet vehicles, not widely adopted in personal vehicles

2.2.3 Identified Gaps In Current Systems

Based on the literature review and evaluation of existing accident detection technologies, the following gaps have been identified in current systems:

- **Over-Reliance on GPS and Network Connectivity:**
  Most accident detection systems depend heavily on GPS and mobile networks to track vehicle movement and trigger alerts. These systems fail in areas with poor signal strength or no connectivity, such as tunnels, remote roads, or rural regions.
- **Limited Real-Time Emergency Alert Mechanisms**
  While some systems can detect crashes, **few offer automated, multi-channel notifications** to emergency services like hospitals, police, and ambulances. The lack of instant communication delays rescue operations, increasing the risk of severe injury or fatality.
- **Inaccessibility in Low-End or Older Vehicles**
  Many modern detection systems are integrated into high-end vehicles through built-in sensors and telematics. **Older vehicles or two-wheelers**, which constitute a large percentage of road traffic, are left without access to such safety measures.
- **High Costs and Complex Hardware Requirements**
  Systems that require advanced sensors, GPS modules, or telematics infrastructure are often **expensive and difficult to implement at scale**, especially in developing countries.
- **Low Accuracy in Crash Detection Using Sensors Alone**
  Accelerometer-based methods often generate **false positives** (e.g., sudden braking, potholes, or rough terrain being mistaken for crashes). These inaccuracies can lead to unnecessary alerts and reduce trust in the system.
- **No Use of Vision-Based Intelligence in Affordable Solutions**
  While research shows the potential of video and computer vision in detecting accidents, **practical, low-cost implementations are still rare**, particularly in systems that can be deployed on roadsides or traffic poles rather than within vehicles.

2.2.4 Role Of Artificial Intelligence In Accident detection:

Artificial Intelligence (AI) plays a crucial role in modernizing accident detection systems by enabling real-time analysis, object recognition, and decision-making without human intervention. Traditional systems relying solely on sensors or GPS have limitations in coverage, accuracy, and scalability. AI, particularly computer vision and deep learning, offers a more intelligent and adaptable solution.Here's how AI contributes to accident detection in the context of systems like RESQRADAR:

1. Computer Vision

Definition: A field of AI that enables machines to interpret visual data from cameras.

- ❖ **Application in RESQRADAR:** Video streams from roadside or vehicle-mounted cameras are analyzed to detect anomalies in vehicle behavior, such as collisions, rollovers, or sudden stops.
- ❖ **Impact:** Enables real-time crash detection without relying on GPS or motion sensors

2. Object Detection and Tracking

- ❖ **Technique:** Using deep learning models (like YOLO, SSD, or Faster R-CNN) to detect and track vehicles across video frames.
- ❖ **Application in Accident Detection:** Identifies and monitors vehicles' speed, trajectory, and impact zones to detect crashes.
- ❖ **Benefit:** Helps distinguish between normal behavior (e.g., sudden braking) and actual accidents.

3. Anomaly Detection

- ❖ **Purpose:** Identify patterns that deviate from expected behavior—such as erratic vehicle motion, sudden halts, or debris on the road.
- ❖ **AI Methodology:** Uses historical data and neural networks to flag potentially dangerous or abnormal events.
- ❖ **Outcome:** Reduces false positives and increases the accuracy of crash detection.

4. Automated Emergency Alerts

- AI triggers automated alert systems the moment a crash is confirmed.
- **Execution in RESQRADAR:** Sends real-time notifications via SMS or web APIsto hospitals, ambulances, and police stations.
- **IntegrationAdvantage:** Accelerates emergency response without manual intervention.

5. Scalability and Adaptability

- **AI models** can be trained for various road conditions, weather, and traffic patterns.
- **RESQRADAR's Vision:** Deployable in urban or rural setups with minimal hardware—making it accessible and effective on a wide scale.

CHAPTER 3
METHODOLOGY

3.1 PROPOSED SYSTEM

RESQRADAR is developed as an intelligent, sensor-based, and real-time accident detection and alert system. The architecture is modular and hardware-centric, designed to accurately detect vehicular accidents and automatically notify emergency contacts without human intervention. The system integrates multiple components such as microcontrollers, accelerometers, GPS modules, and GSM communication units, all working in coordination to ensure reliable operation in real-world scenarios.

The core functionality revolves around detecting sudden changes in acceleration or impact, which typically indicate a crash. This is achieved using an onboard accelerometer that continuously monitors vehicle motion and triggers alerts when abnormal values are detected. Once an accident is identified, the system uses the GPS module to fetch the exact location coordinates of the incident. These coordinates are then relayed via the GSM module, which sends an automated SMS alert to pre-defined emergency contacts, including rescue services, hospitals, and family members.

To ensure real-time operation and quick deployment, the system is built using embedded programming with Arduino or NodeMCU microcontrollers. It is also capable of logging incident data for future analysis, which can assist in road safety assessments and research. The entire setup is housed within a compact unit that can be easily installed in various types of vehicles.

Overall, the design of RESQRADAR focuses on being reliable, responsive, and easy to implement, particularly in areas with limited access to immediate medical assistance. By automating the detection and alert process, it significantly reduces emergency response time, potentially saving lives and minimizing the consequences of road accidents.

3.1.1 System Architecture – Block Diagram

The proposed system automates the end-to-end process of managing UPI payments using the following pipeline:

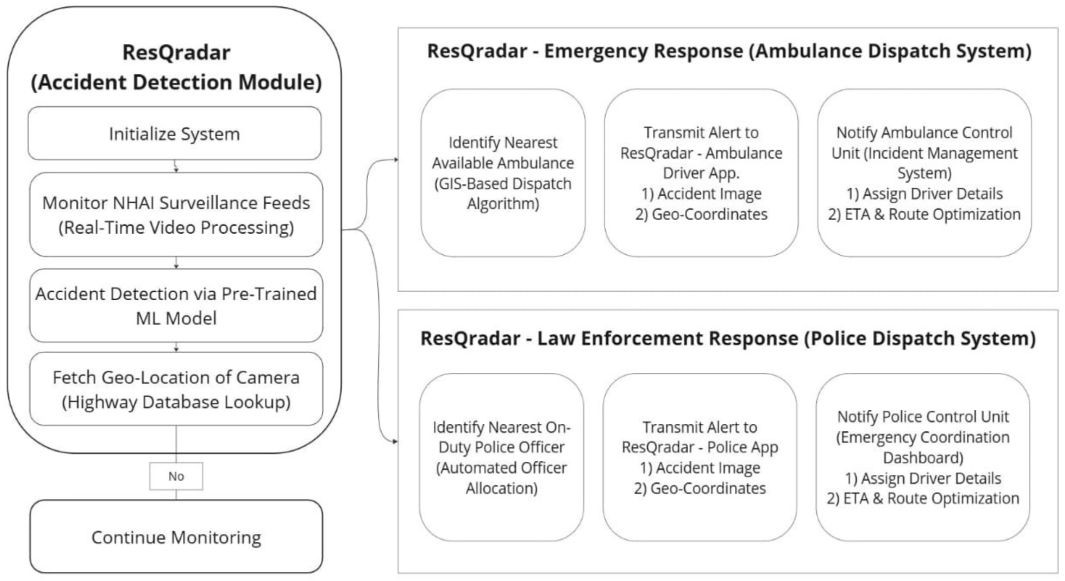

ResQradar
(Accident Detection Module)
Initialize System
Monitor NHAI Surveillance Feeds (Real-Time Video Processing)
Accident Detection via Pre-Trained ML Model
Fetch Geo-Location of Camera (Highway Database Lookup)
No
Continue Monitoring
ResQradar - Emergency Response (Ambulance Dispatch System)
Identify Nearest Available Ambulance (GIS-Based Dispatch Algorithm)
Transmit Alert to ResQradar - Ambulance Driver App. 1) Accident Image 2) Geo-Coordinates
Notify Ambulance Control Unit (Incident Management System) 1) Assign Driver Details 2) ETA & Route Optimization
ResQradar - Law Enforcement Response (Police Dispatch System)
Identify Nearest On-Duty Police Officer (Automated Officer Allocation)
Transmit Alert to ResQradar - Police App 1) Accident Image 2) Geo-Coordinates
Notify Police Control Unit (Emergency Coordination Dashboard) 1) Assign Driver Details 2) ETA & Route Optimization

Figure 3.1.1 Block Diagram

The architecture of the RESQRADAR – Accident Detection System is designed for intelligent real-time monitoring and emergency response coordination. It consists of three major functional components: Accident Detection Module, Emergency Response Module, and Law Enforcement Response Module. These modules work collaboratively to detect accidents using highway surveillance feeds and immediately alert relevant emergency services.

1.Accident Detection Module

This module is responsible for monitoring live video feeds and identifying vehicular accidents using machine learning techniques. The process follows these sequential steps:

- ❖ System Initialization: RESQRADAR activates its core subsystems, preparing the video stream processing and detection algorithms.
- ❖ Surveillance Feed Monitoring: It continuously monitors real-time video feeds from NHAI-installed highway surveillance cameras.
- ❖ Machine Learning-Based Detection: A pre-trained ML model is applied to the incoming video stream. This model is capable of detecting anomalies in vehicle movement, such as sudden stops, collisions, or overturned vehicles, which are indicative of accidents.
- ❖ Geo-Location Retrieval: Once an accident is detected, the system performs a highway database lookup to fetch the exact geographical location associated with the camera ID. This approach eliminates the need for onboard GPS devices.
- ❖ Decision Loop: If no accident is detected, the system continues monitoring the feed in a loop, ensuring continuous operation.

2.Emergency Response Module (Ambulance Dispatch System)

Upon detecting an accident, RESQRADAR initiates the emergency response workflow:

- ❖ **Ambulance Identification:** The system uses a GIS-based dispatch algorithm to identify the nearest available ambulance to the incident location.
- ❖ **Alert Transmission:** A structured alert is transmitted to the RESQRADAR Ambulance Driver App. This alert contains:
  - ➢ A real-time image of the accident scene
  - ➢ The geo-coordinates of the accident

- ❖ **Ambulance Control Notification:** Simultaneously, the control unit is notified via the Incident Management System. This system:
  - ➢ Assigns the appropriate driver
  - ➢ Calculates the Estimated Time of Arrival (ETA)
  - ➢ Suggests optimal routes to reach the accident scene quickly

3.Real-Time Alert System Module

Objective: Notify appropriate responders instantly upon accident detection.

Sub-components:

- ❖ **Alert Triggering Logic**:
  - ➢ If crash conditions are met, send a signal to the alert handler.
- ❖ **Notification Mechanism**:
  - ➢ SMS: Use Twilio API or GSM module.
  - ➢ Email: SMTP setup for sending alerts.
  - ➢ Control room UI: Raise on-screen pop-up or audio signal

4.Law Enforcement Response Module (Police Dispatch System

Parallel to the ambulance dispatch, RESQRADAR activates law enforcement response protocols:

- ❖ **Police Officer Identification:** The system automatically identifies the nearest on-duty police officer using real-time duty and availability data.
- ❖ **Alert Delivery to Police App:** A similar alert is sent to the RESQRADAR **Police App, including:**
  - ➢ Accident image,Location coordinates
- ❖ **Police Control Notification:** The Police Control Unit is notified through the Emergency Coordination Dashboard. This includes:
  - ➢ Officer assignment

5. Summary of Operations

Unlike traditional vehicle-based systems, RESQRADAR is infrastructure-based, utilizing fixed surveillance camera inputs rather than Gps or onboard sensors. Its AI-driven detection mechanism, coupled with instant alert generation for both medical and law enforcement agencies, ensures a fast and coordinated response.

This architecture makes RESQRADAR highly scalable and particularly effective for national highways and urban expressways where surveillance infrastructure is already present.

6. Real-Time Accident Detection in RESQRADAR

RESQRADAR is designed to autonomously monitor highway surveillance feeds and detect accidents in real-time, converting these incidents into actionable alerts for emergency **s**ervices. The system mimics the efficiency of financial SMS parsing in FinanSmart but applies it to CCTV video streams for public safety.

Step-by-Step Workflow:

1. Highway Video Feed Captured

Live video streams from NHAI or highway-mounted surveillance cameras act as input sources for the system.

- ❖ Captured Data Includes:
  - ➢ Real-time vehicular movement
  - ➢ Environmental conditions (e.g., night, rain)
  - ➢ Traffic anomalies

2. AI-Based Accident Detection

Each frame extracted from the video feed is processed in real time using a pre-trained machine learning model, such as YOLOv8 or a custom-built Convolutional Neural Network (CNN). These models are trained to detect and classify various accident-related events by analyzing visual patterns and object movements within the scene. Key indicators of accidents—such as vehicle collisions, sudden braking, skidding, or overturned vehicles—are identified through object detection and motion analysis techniques. The use of deep learning-based models enables high accuracy in detecting critical incidents, even under challenging conditions like low lighting or occlusion. This real-time detection capability is essential for timely alerts, emergency response, and post-incident analysis.

- ❖ Detection Criteria:
    - ➢ Motion interruption
    - ➢ Vehicle deformation
- ❖ Tools Used:
    - ➢ Python, TensorFlow or PyTorch
    - ➢ Pre-labeled accident datasets

3. Geo-Location Mapping

Once an accident is detected, the camera ID is mapped to a real-world geo-location using a structured highway camera database.

- ❖ No GPS Required:
    - ➢ Static mapping from camera ID to location (lat-long, highway segment)
- ❖ Tools Used:
    - ➢ PostgreSQL + PostGIS or Firebase Realtime DB

4. Instant Emergency Categorization & Notification

- ❖ An accident alert is auto-generated and pushed to:
    - ➢ ResQRadar Ambulance App
    - ➢ ResQRadar Police App

- Notification Contains:
  - Frame snapshot of the accident
  - Time and location of incident
  - Type: Ambulance/Police/Fire (future scope)

5. Real-Time Alert Prompting for Dispatch

RESQRADAR prompts control centers to approve or track dispatch.

- User Interfaces:
  - Ambulance Driver Dashboard
  - Police Officer Dashboard
  - Admin Control Panel

6. Secure Data Storage & Logging

All accident events, image frames, and dispatch logs are securely stored for audit and analytics.

- Tech Stack:
  - Encrypted NoSQL DB (e.g., MongoDB)
  - Secure cloud backups

3.1.2 Modules (Fragments and Services)

1. AndroidManifest Configuration

The AndroidManifest.xml is the blueprint of RESQRADAR's mobile application, defining critical permissions and service declarations. It ensures the app operates efficiently in the background, especially in contexts requiring real-time video stream processing and notification dispatch.

- **Declared Permissions**:
  - Camera and Internet access
  - Foreground service execution
- **Services Declared:**
  - AccidentDetectionService: Foreground service for AI-based videos tream
  - NotificationService: Dispatches alerts to authorities and dashboards
- **Compatibility Enhancements:**
  - Support for multi-camera devices
  - RTL and backup metadata for accessibility and storage management

2. MainActivity (MainActivity.php)

This module is the user entry point for administrators and monitoring personnel. It initializes UI elements such as map views, camera feeds, and notification panels. It also handles permission checks and starts the core background services.

- **Responsibilities**:
  - Check and request permissions (camera, internet, notifications)
  - Launch AccidentDetectionService for real-time video analysis
  - UI management for video and alert dashboards
  - Integration with emergency contact systems
- **UI Components**:
  - Bottom navigation for Feed View, Incident Log, Analytics, and Settings

3. Accident Detection Engine (CameraService & ML Module)

This critical module performs **real-time video analysis** using AI models to detect traffic anomalies suggestive of accidents.

- ❖ **Components**:
  - ➢ CameraService: Streams video from IP or USB-connected cameras
  - ➢ ML Model (YOLOv8/Custom CNN): Detects collisions, abrupt halts, or overturned vehicles
- ❖ **Processing Pipeline**:
  - ➢ Frame extraction → Preprocessing → ML inference → Event trigger
- ❖ **Tools**: OpenCV, PyTorch (edge-deployed or cloud-hosted)

4. Notification & Alert Dispatcher (NotificationService.java)

Once an accident is detected, this module:

- ➢ Captures the frame
- ➢ Retrieves the geo-location mapped from camera ID
- ➢ Sends alerts to registered emergency contacts (ambulance, police)
- ❖ **Notification Types**:
  - ➢ Instant push notifications
  - ➢ Email + SMS fallbacks
  - ➢ Real-time broadcast to police/ambulance dashboards
- ❖ **Tech**: Firebase Cloud Messaging (FCM), Android NotificationManager

5. Geo-Mapping & Dispatch Module

Instead of GPS, each surveillance camera is linked to a static geo-location ID stored in a structured backend database.

- **Tasks**:
    - Map detected event to exact highway location
    - Query nearest dispatch unit (ambulance/police station)
    - Forward enriched alert (image + location)
- **Database**: Firebase / PostgreSQL + PostGIS

6. HomeFragment / Dashboard

This module serves as the real-time monitoring interface for system operators. It shows:

- Live camera feeds
- Recent accident logs
- Map view with alert markers
- Manual override or reporting tools

7. Smart Response Assistant (via Gemini/GPT)

Optional AI-driven chatbot or assistant for dispatch operators. Can suggest:

- Nearest hospital based on incident type
- Real-time route estimation
- Prioritization tips during high-incident traffic windows

How it Works :

RESQRADAR is an AI-powered real-time accident detection and alert system designed to monitor highway traffic through video feeds and automate emergency notifications. The system operates without relying on GPS, making it suitable for static highway camera setups maintained by NHAI or local authorities.

Step 1: Live Video Feed Capture

- High-definition surveillance cameras are installed at critical highway points.
- These cameras continuously stream video to the RESQRADAR processing unit.

Step 2: Continuous Frame Monitoring

- The system extracts frames from the live video in real-time using tools like **OpenCV** or **FFmpeg**.
- A background service (CameraFeedService) runs continuously, feeding these frames into the detection model.

Step 3: Accident Detection via AI

Each frame is analyzed by a trained **YOLOv8 or custom CNN model**.

- The AI model identifies signs of accidents such as:
  - Sudden vehicle stops
  - Collisions
  - Overturned vehicles
  - Smoke or fire (in future enhancements)
- If an anomaly is detected, it triggers the next stage of processing.

Step 4: Geo-Mapping without GPS

- Instead of relying on mobile GPS, each camera's unique ID is mapped to a static location in a backend database.
- This allows the system to instantly determine the exact location of the incident using **camera ID-to-location mapping**.

Step 5: Emergency Notification Generation

- Upon detection, the system captures the frame of the incident.
- A detailed alert is generated containing:
  - Type of incident
  - Time of detection
  - Mapped location (Highway name, Kilometer marker, etc.)
  - Snapshot of the detected accident
- This alert is sent to:
  - Emergency dispatch units (Ambulance, Police)

Step 6: Real-Time Dashboard Updates

- The control room interface or admin dashboard receives real-time incident markers.
- Accidents are logged into the system for review, reporting, and historical analysis.
- Admins can filter events by date, location, or response status.

Step 7: Optional Smart Response Assistant

- A built-in assistant powered by **Gemini AI or GPT** can:
  - Suggest nearest hospitals
  - Provide voice/chat-based summaries
  - Offer decision support to control room operators during multiple incidents

Key Features:

- No GPS dependency – Static camera mapping
- Real-time AI detection with minimal latency
- Automated alert dispatch
- Scalable and modular architecture
- Suitable for highways, expressways, and toll plazas

Key Functionalities:

- **Pie Chart**: Visualizes categorized expenditure (e.g., Food, Travel) for a selected **day** or **month**. Pie slices are styled with customized colors and value lines, showing both category labels and amounts in Indian currency format (₹).
- **Line Chart**: Displays a time-series chart of spending across hours (for a selected day) or dates (for a selected month). Interactive with labels, currency formatting, and animated transitions for enhanced UX.
- **Date and Month Toggle**:
  - Users can toggle between **"Today"** and **"Month"** views.
    A **date picker dialog** allows selection of a specific day.
  - A **custom dialog** with NumberPicker components enables month and year selection.

- ❖ **Dynamic Chart Updates**: Both charts are updated in real-time upon date/month selection using setUpPieChart() and setUpLineChart() methods, which fetch and format data from the local SQLite database.
- ❖ **Theming and Styling**: Charts and UI components follow a consistent dark/light color scheme with enhanced readability (e.g., currency formatting, value text styling, label positioning).

Supporting Components:

- ❖ **Database Integration**: Data is fetched from a local SQLite database using custom query methods like getTodayData() and getSubPurposeSumByDate().
- ❖ **Date Formatting**: Makes use of SimpleDateFormat, DateTimeFormatter, and LocalDate for precise date parsing and formatting.

7. Export Module:

Purpose:

The Export Module is responsible for allowing users to export their financial data from the application. It supports daily as well as monthly exports in .xlsx format and also displays a list of recently exported files, enabling quick access and verification.

Key Features:

RESQRADAR's dashboard is designed for simplicity and rapid incident monitoring. It provides a clear, interactive interface for authorities to track highway accidents, analyze trends, and take immediate action.

- ❖ **Incident Timeline Line Chart Purpose**: Displays time-based distribution of detected accidents.
- ❖ **Modes**:
    - ➢ **Daily View**: Charts show accident occurrences by hour for a selected day.
    - ➢ **Monthly View:** Charts show accident count per day across a selected month.

Features:

- Interactive and responsive with smooth transitions.
- Dynamic axis labels with timestamps or dates.
- Real-time updates as new data is logged.
- Clean formatting for high visibility.

- **Date and Month Toggle**

Allows users to view data in two primary modes:

- “Today” View for real-time monitoring.
- “Month” View for trend analysis.
- A Date Picker Dialog for precise day selection.
- A Custom Month-Year Picker Dialog using NumberPickers.

- **Real-Time Chart Refresh**
  - When the date or month is changed:
  - The line chart is refreshed using setUpLineChart() method.
  - Pulls updated accident data from the local database or live feed log.
  - Provides quick visual insight into peak accident times.

- **Theming and Visual Clarity**
  - UI supports light/dark themes suitable for control room use.
  - Clean layout and sharp labels ensure clarity on large displays.
  - Value formatting emphasizes readability for time-sensitive decision-making.

Each frame extracted from the continuous video feed is subjected to real-time analysis using advanced machine learning techniques, specifically leveraging pre-trained models such as YOLOv8 (You Only Look Once, version 8) or a custom-designed Convolutional Neural Network (CNN) tailored for accident detection. These deep learning models are trained on large datasets containing annotated traffic scenes, enabling them to learn and recognize complex visual cues and patterns that are indicative of vehicular accidents. The models perform object detection and classification tasks to identify vehicles, pedestrians, road boundaries, and other relevant entities within each frame.

By tracking the movement and interaction of these objects over successive frames, the system can detect abnormal events that signify potential accidents. These include but are not limited to vehicle collisions, abrupt deceleration (sudden stops), lane departures, vehicles skidding out of control, or flipping over.

3.2 SYSTEM REQUIREMENTS

3.2.1 Hardware Requirements

Device Type:

- ❖ Android smartphone or tablet.
- ❖ Integrated or externally connected **video camera** (for accident detection input).
- ❖ **Touchscreen interface** for interacting with dashboard features.
- ❖ Must support **camera permissions** and **background processing** for continuous monitoring.
- ❖ **Internet connectivity** is optional but recommended for alert delivery (e.g., to control rooms or emergency services).

Minimum Android Version:

- ❖ **Android 8.0 (Oreo)** or higher.
- ❖ **API Level 26** for background execution control and service prioritization.
- ❖ Required for camera permission handling, real-time alerting, and system-level scheduling (e.g., alarms).
- ❖ Ensures compatibility with modern **security models** and Android **Jetpack libraries**.
- ❖ Supports **camera2 API** and **Notification Channels** for seamless user feedback.

RAM:

- ❖ Minimum **2 GB RAM** for:
  - ➢ Real-time video frame analysis using computer vision.
  - ➢ Smooth operation of multiple UI fragments (e.g., Alerts, Dashboard, Logs).
  - ➢ Background services running detection algorithms continuously.
  - ➢ Minimizing delays in alert triggering and crash response time.

Storage:

- ❖ Minimum **100 MB of free internal storage** for:
  - ➢ Storing **locally processed crash data** (e.g., timestamps, camera footage snapshots, event logs).
  - ➢ Space allocation for system logs and alert history.
  - ➢ Offline caching of processed frames or model inferences.
  - ➢ Temporary storage of footage used for post-incident review (if applicable)

3.2.2 Software Requirements

Development Environment:

- ❖ **Python 3.8+** for accident detection and monitoring logic
- ❖ **PHP 7.4+** for backend alert management and dashboard
- ❖ Text editors: Visual Studio Code, PyCharm, or PHPStorm
- ❖ Apache server or XAMPP for PHP application hosting
- ❖ OpenCV library for image and video processing

Database:

- MySQL or SQLite for storing:
    - Detected incidents
    - User access logs
    - Alert messages

Web Application Framework:

- PHP-based dashboard system for:
    - Displaying recent accident events
    - Viewing camera feeds (optional)
    - Generating reports/logs

Libraries and Tools:

- **Python:**
    - OpenCV (cv2) – for motion/crash detection
    - NumPy – for image matrix operations
    - smtplib / Twilio – for sending alerts (SMS/Email)

PHP:

- MySQLite or PDO – for database interactions
- Bootstrap – for responsive frontend
- PHP Mailer – for sending email alerts

3.3 TECHNIQUES AND MODELS USED

3.3.1 Computer Vision Crash Detection (Python + OpenCV)

- Real-time video feed is analyzed using motion detection or object collision heuristics.
- Frame-by-frame difference and bounding box detection is used to monitor:
  - Sudden motion stops
  - Collision patterns
  - Vehicle rollovers or tipping
- When a crash pattern is recognized, it triggers an incident capture.

Benefits:

- Fast and lightweight
- Works offline

3.3.2 Alert Trigger & Logging System (Python → PHP)

- Python script calls the PHP application through an HTTP request or REST API when a crash is detected.
- PHP handles:
  - Storing the event into the database
  - Sending notifications (email/SMS via 3rd-party APIs)
  - Displaying the alert on the web dashboard
- This modular approach separates detection (Python) from user interface (PHP)

3.3.3 PHP Web Dashboard

- Visual representation of logs: time, location (optional), camera ID, severity, etc.
- Admin panel includes:
  - Exporting logs (CSV, PDF)
  - Viewing video snapshot (if captured)
  - Filter by date/time/incident type

3.3.4 Background Services and Logging

- **Python daemon** runs in the background as a service on boot.
- Logs are stored and rotated daily to ensure memory efficiency.
- PHP pron jobs can be scheduled for:
  - Sending summary reports
  - Cleaning old logs
  - Checking for system health

CHAPTER 4

IMPLEMENTATION

4.1 MODULE-WISE DESIGN

This section explains the implementation of each core module of the FinanSmart application in detail, showcasing how different fragments and services contribute to the overall functionality.

4.1.1 Ambulance Control Unit

1. File Structure

The project consists of three main PHP files: add_driver.php, index.php, and reports.php. Each file serves a specific purpose in managing the application's functionality.

2. Database Connection

In add_driver.php, a connection to the MySQL database is established using the mysqli class. This connection is used to perform CRUD operations on the ambulance_drivers table.

3. Driver Management

add_driver.php handles adding new drivers to the system. It captures driver details via a form, generates a unique driver ID, and inserts the data into the database.

4. Search Functionality

Both add_driver.php and reports.php implement search functionality. Users can search for drivers or accident records by entering keywords, which filter the displayed data in real-time.

5. Data Export

add_driver.php and reports.php include functionality to export data to CSV or Excel files. This feature uses PHP's fputcsv function and the xlsx library for data export.

6. Real-Time Data Fetching

index.php and reports.php use AJAX to fetch real-time data from the server. This ensures that the displayed information is always up-to-date without requiring a page refresh.

7. Data Visualization

index.php includes charts to visualize accident data. The Chart.js library is used to create line and bar charts, providing insights into accident trends and district-wise accident counts.

8. Modal Functionality

index.php and reports.php use modals to display detailed information, such as accident images. The modals are triggered by buttons and can be closed by clicking outside the modal area.

4.1.2 Police Control Unit

1. File Structure

The project consists of three main PHP files: add_police.php, index.php, and reports.php. Each file serves a specific purpose in managing the application's functionality.

2. Database Connection

In add_police.php, a connection to the MySQL database is established using the mysqli class. This connection is used to perform CRUD operations on the police_officers table.

3. Police Management

add_police.php handles adding new police officers to the system. It captures officer details via a form, generates a unique police ID, and inserts the data into the database.

4. Search Functionality

Both add_police.php and reports.php implement search functionality. Users can search for police officers or accident records by entering keywords, which filter the displayed data in real-time.

5. Data Export

add_police.php and reports.php include functionality to export data to CSV or Excel files. This feature uses PHP's fputcsv function and the xlsx library for data export.

4.1.3 Ambulance Driver Application

1. Authentication and Session Management

The application uses PHP sessions to manage user authentication. The login.php file handles user login, verifying credentials against the database and setting session variables upon successful login. The logout.php file destroys the session, logging the user out.

2. Database Interaction

The application connects to a MySQL database using PDO for secure and efficient database operations. It fetches pending and attended cases for the logged-in driver, updates case status when accepted, and checks for new cases periodically

3. User Interface and Navigation

The application features a responsive navigation bar and sidebar for easy navigation between different sections like New Cases, Attended Cases, and Logout. The sidebar is toggleable for mobile devices, ensuring a user-friendly experience on all screen sizes.

4. Case Management

The index.php file displays pending cases to the driver, allowing them to accept cases. The attend.php file shows attended cases, providing details and status updates. The application uses base64 encoding to display accident images directly from the database.

5. Real-Time Updates

The application polls the server every 10 seconds to check for new cases using check_cases.php. If new cases are detected, the page reloads to display the updated list. This ensures that drivers receive real-time updates on pending cases.

4.1.4 Police Officer Application

1. Authentication Management

The application uses PHP sessions to manage user authentication. The login.php file handles user login by verifying credentials against the database and setting session variables upon successful login. The logout.php file destroys the session, logging the user out.

2. Database Interaction

The application connects to a MySQL database using PDO for secure and efficient database operations. It fetches pending and attended cases for the logged-in police officer, updates case status when accepted, and checks for new cases periodically.

3. User Interface and Navigation The application features a responsive navigation bar and sidebar for easy navigation between different sections like New Cases, Attended Cases, and Logout. The sidebar is toggleable for mobile devices, ensuring a user-friendly experience on all screen sizes.

4. Case Management

The index.php file displays pending cases to the police officer, allowing them to accept cases. The attend.php file shows attended cases, providing details and status updates.

4.1.5 AI CCTV Monitoring (main.py)

1. System Initialization

The script initializes necessary libraries and resources, including the YOLO model for object detection, OpenCV for video processing, Pygame for audio and display, and MySQL for database interaction. It also sets up directories for storing output videos and accident images.

2. Video Source Configuration

The script allows for flexibility in video input, supporting both video files and live camera feeds. It configures the video capture source based on the use_video_file flag and sets up a VideoWriter object for saving processed video output.

3. Accident Detection and Response

The core functionality involves real-time accident detection using the YOLO model. When an accident is detected, the system captures the relevant quadrant of the video frame, saves it as an image, and triggers an alarm sound. It also retrieves and prints CCTV data for the affected quadrant from the database.

4. Database Integration

The script integrates with a MySQL database to fetch CCTV data and insert emergency records for detected accidents. It handles database connections, queries, and error handling to ensure robust data management.

4.2 TOOLS USED

Tool/Library	**Purpose**
YOLO (You Only Look Once)	Real-time object detection model
OpenCV	Video processing and image manipulation
Pygame	Audio playback and real-time display
MySQL	Database management for storing and retrieving data
MySQL Connector	Database connectivity and interaction
NumPy	Efficient numerical computations
Datetime	Handling date and time operations

4.3 DATABASE

The ResQRadar system utilizes a structured database to manage and analyze accident data, sourced from:

- ❖ AI CCTV monitoring system logs
- ❖ GPS coordinates of accident locations
- ❖ Emergency response times recorded by dispatch centers
- ❖ User interactions with the system interface
- ❖ Exported records in Excel format for further analysis Each accident record comprises:
- ❖ mages or videos of the accident scene
- ❖ Description of the accident
- ❖ Images or videos of the accident scene
- ❖ Description of the accident

4.4 EXPERIMENTS AND RESULTS

4.4.1 Snapshots

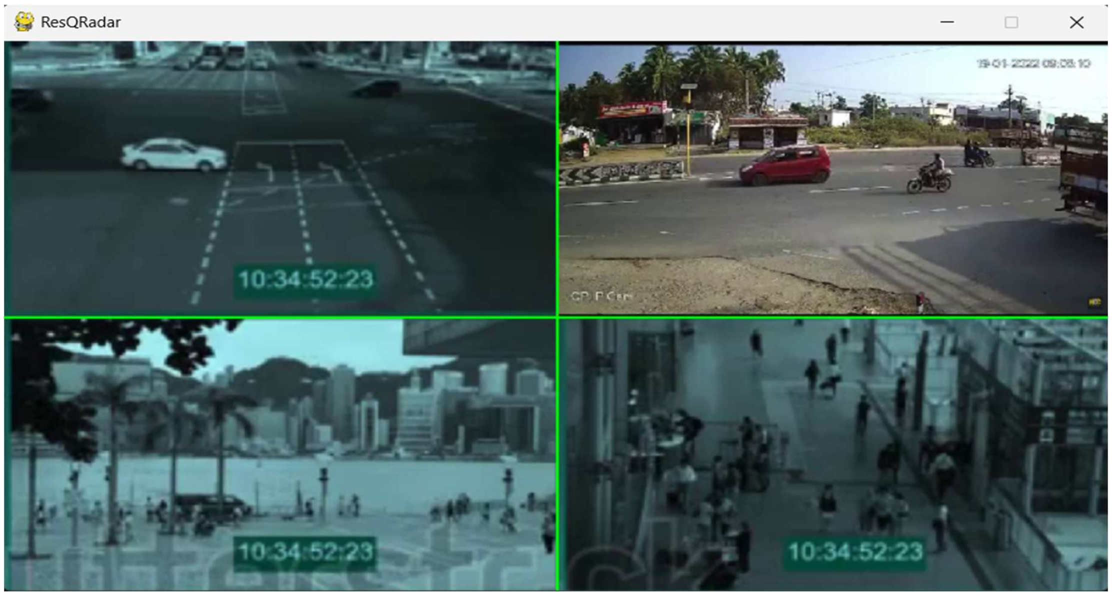
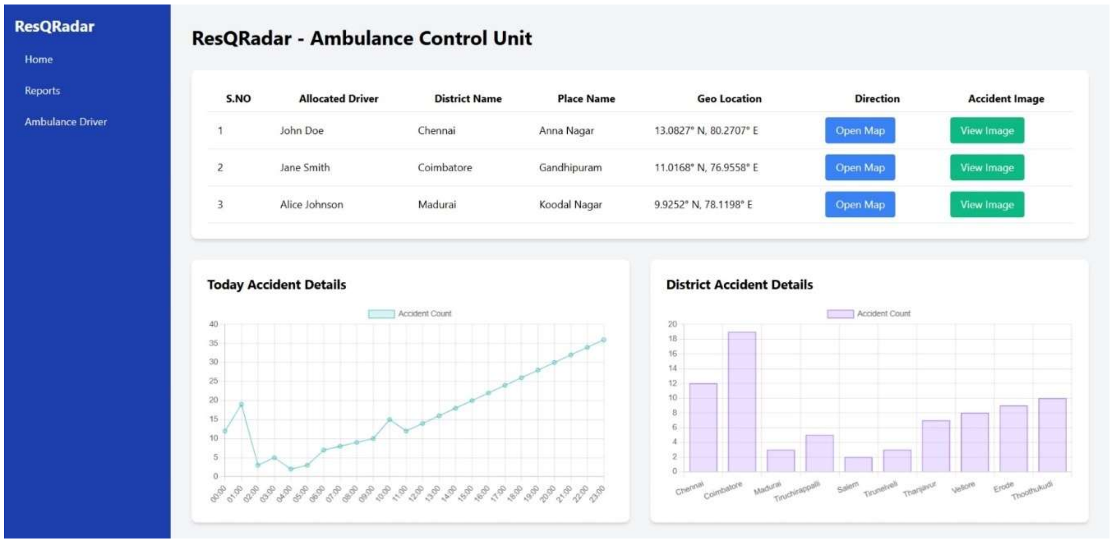


Figure 4.4.1.1

The image displays a computer screen showing the interface of an AI-powered CCTV monitoring system named "ResQRadar." It features four separate live feeds from different cameras, each capturing various scenes that could include roads or public areas. The system is designed to detect accidents in real-time. If an accident is detected, the system captures images and possibly triggers alerts to notify authorities. Each video feed shows a timestamp, indicating the current time of the footage, which is crucial for accurate and timely incident reporting. This technology enhances safety by providing immediate visual evidence and response capabilities.

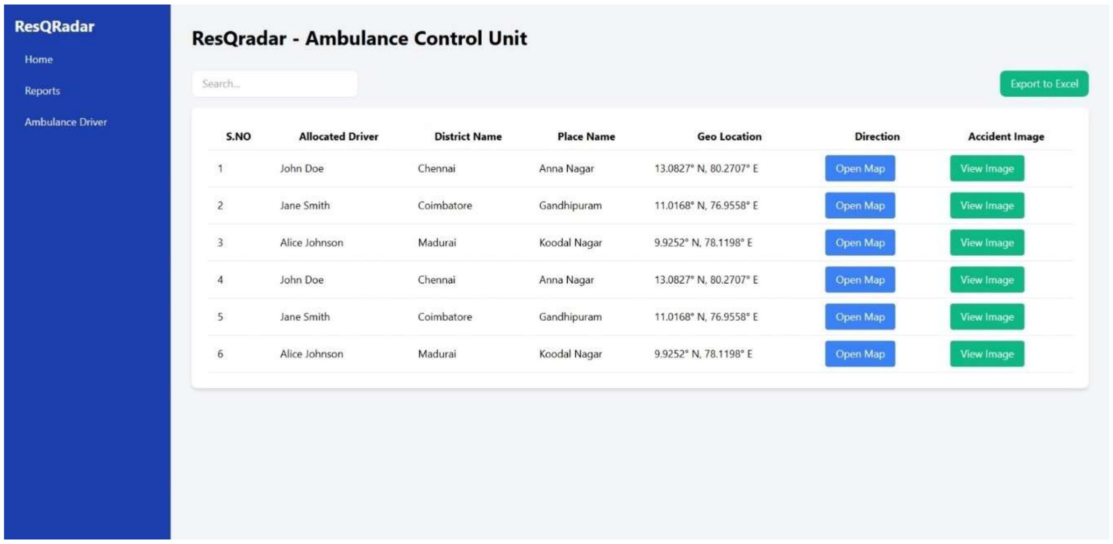


ResQRadar - Ambulance Control Unit

S.NO	Allocated Driver	District Name	Place Name	Geo Location	Direction	Accident Image
1	John Doe	Chennai	Anna Nagar	13.0827° N, 80.2707° E	Open Map	View Image
2	Jane Smith	Coimbatore	Gandhipuram	11.0168° N, 76.9558° E	Open Map	View Image
3	Alice Johnson	Madurai	Koodal Nagar	9.9252° N, 78.1198° E	Open Map	View Image

Figure 4.4.1.2

The image displays a dashboard from the ResQRadar Ambulance Control Unit. It lists ambulance drivers with corresponding districts and geolocations. Each entry includes a button to open a map and view an accident image. Below, two charts provide insights: "Today Accident Details" shows hourly accident counts, while "District Accident Details" illustrates accident frequency by district. The interface is designed for efficient monitoring and management, enabling quick responses to incidents. It leverages data visualization to aid decision-making and improve emergency services' coordination.



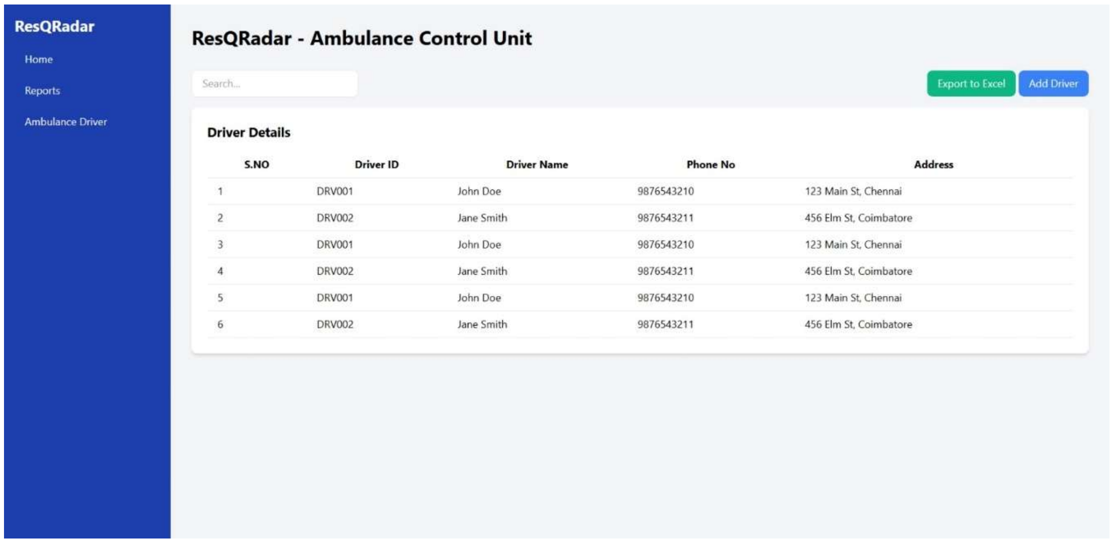
ResQradar - Ambulance Control Unit

S.NO	Allocated Driver	District Name	Place Name	Geo Location	Direction	Accident Image
1	John Doe	Chennai	Anna Nagar	13.0827° N, 80.2707° E	Open Map	View Image
2	Jane Smith	Coimbatore	Gandhipuram	11.0168° N, 76.9558° E	Open Map	View Image
3	Alice Johnson	Madurai	Koodal Nagar	9.9252° N, 78.1198° E	Open Map	View Image
4	John Doe	Chennai	Anna Nagar	13.0827° N, 80.2707° E	Open Map	View Image
5	Jane Smith	Coimbatore	Gandhipuram	11.0168° N, 76.9558° E	Open Map	View Image
6	Alice Johnson	Madurai	Koodal Nagar	9.9252° N, 78.1198° E	Open Map	View Image

Figure 4.4.1.3

The image showcases the ResQRadar Ambulance Control Unit interface, designed for managing emergency responses. It lists ambulance drivers with their assigned districts and exact geolocations.

A search function allows quick access to specific records. Each entry is accompanied by buttons to open maps for navigation and view images of accident scenes. The interface also features an option to export data to Excel, facilitating report generation and record-keeping. This system enhances the efficiency of ambulance dispatch and response times in emergencies, ensuring better coordination and potentially saving lives.

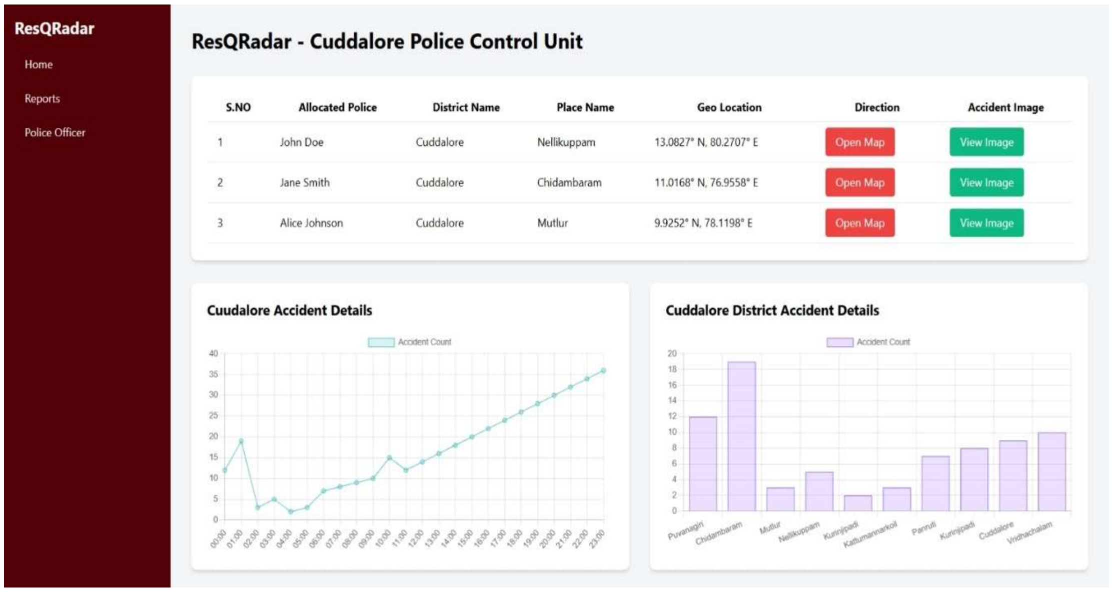


Driver Details

S.NO	Driver ID	Driver Name	Phone No	Address
1	DRV001	John Doe	9876543210	123 Main St, Chennai
2	DRV002	Jane Smith	9876543211	456 Elm St, Coimbatore
3	DRV001	John Doe	9876543210	123 Main St, Chennai
4	DRV002	Jane Smith	9876543211	456 Elm St, Coimbatore
5	DRV001	John Doe	9876543210	123 Main St, Chennai
6	DRV002	Jane Smith	9876543211	456 Elm St, Coimbatore

Figure 4.4.1.3

The image displays the ResQRadar Ambulance Control Unit's interface, focusing on driver management. It shows a list of drivers with their IDs, names, phone numbers, and addresses. The interface includes a search bar for locating specific drivers and buttons to export the list to Excel or add new drivers. This system streamlines the process of organizing and contacting ambulance drivers, ensuring efficient communication during emergencies. The clear layout and functionalities facilitate quick access to vital information, enhancing the overall effectiveness of ambulance dispatch operations.

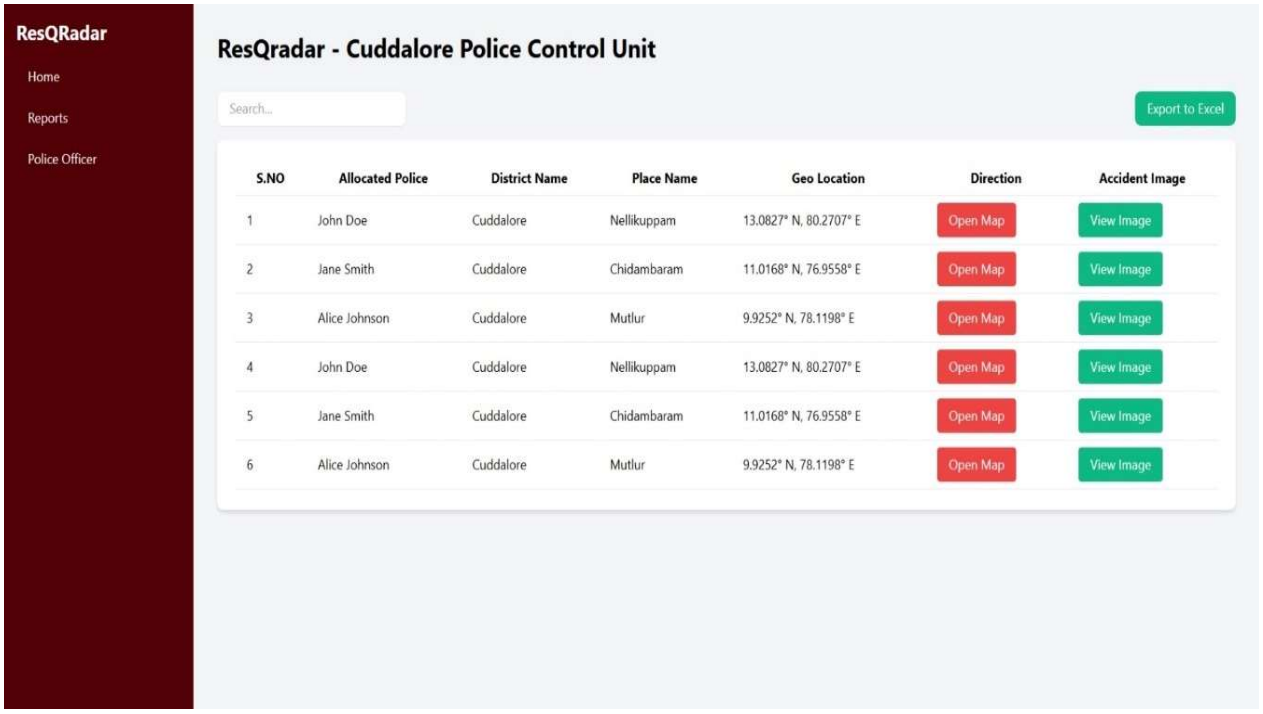


ResQRadar - Cuddalore Police Control Unit

S.NO	Allocated Police	District Name	Place Name	Geo Location	Direction	Accident Image
1	John Doe	Cuddalore	Nellikuppam	13.0827° N, 80.2707° E	Open Map	View Image
2	Jane Smith	Cuddalore	Chidambaram	11.0168° N, 76.9558° E	Open Map	View Image
3	Alice Johnson	Cuddalore	Mutlur	9.9252° N, 78.1198° E	Open Map	View Image

Figureb 4.4.1.4

The image depicts the ResQRadar Cuddalore Police Control Unit dashboard, which is designed to manage and monitor police responses to accidents. It lists police officers with their assigned districts, places, and geolocations. The interface includes options to open maps for navigation and view images of accident scenes. Below the list, two charts provide detailed analytics: one shows the number of accidents throughout the day, and the other displays accident counts by district. This system streamlines emergency response efforts, ensuring that police resources are allocated efficiently and effectively.

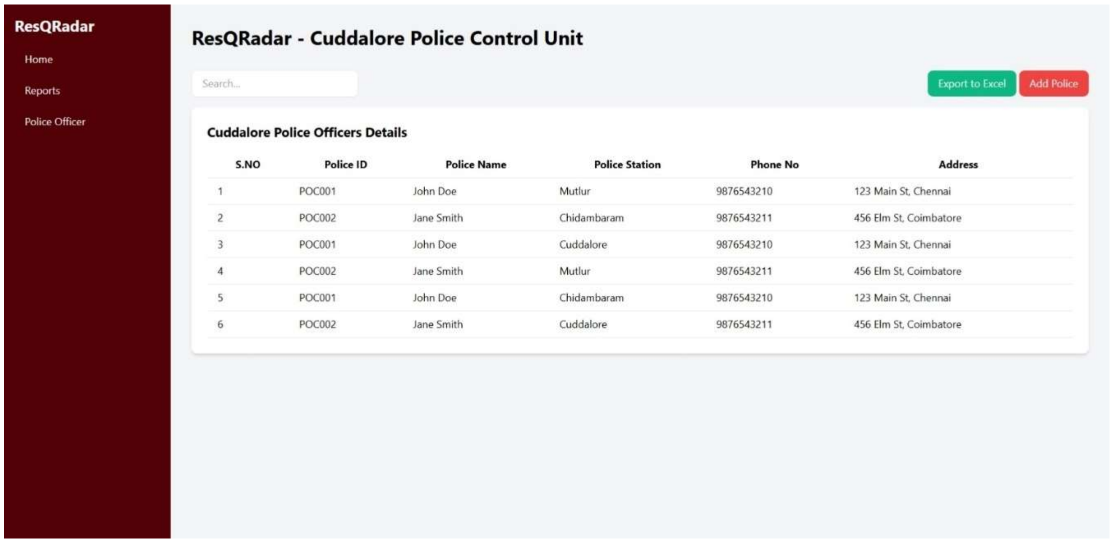


ResQradar - Cuddalore Police Control Unit

S.NO	Allocated Police	District Name	Place Name	Geo Location	Direction	Accident Image
1	John Doe	Cuddalore	Nellikuppam	13.0827° N, 80.2707° E	Open Map	View Image
2	Jane Smith	Cuddalore	Chidambaram	11.0168° N, 76.9558° E	Open Map	View Image
3	Alice Johnson	Cuddalore	Mutlur	9.9252° N, 78.1198° E	Open Map	View Image
4	John Doe	Cuddalore	Nellikuppam	13.0827° N, 80.2707° E	Open Map	View Image
5	Jane Smith	Cuddalore	Chidambaram	11.0168° N, 76.9558° E	Open Map	View Image
6	Alice Johnson	Cuddalore	Mutlur	9.9252° N, 78.1198° E	Open Map	View Image

Figure 4.4.1.5

The image shows the ResQRadar Cuddalore Police Control Unit interface, which is used for managing police officer assignments and monitoring accident reports. The table lists police officers by serial number, along with their names, assigned districts, specific places, and geolocations. Each entry has buttons to open a map for navigation and view related accident images, streamlining the response process. The interface also features a search bar and an option to export data to Excel, enhancing report generation and officer coordination during emergencies. This system is crucial for efficient police dispatch and reducing response times in critical situations.



ResQRadar - Cuddalore Police Control Unit

Cuddalore Police Officers Details

S.NO	Police ID	Police Name	Police Station	Phone No	Address
1	POC001	John Doe	Mutlur	9876543210	123 Main St, Chennai
2	POC002	Jane Smith	Chidambaram	9876543211	456 Elm St, Coimbatore
3	POC001	John Doe	Cuddalore	9876543210	123 Main St, Chennai
4	POC002	Jane Smith	Mutlur	9876543211	456 Elm St, Coimbatore
5	POC001	John Doe	Chidambaram	9876543210	123 Main St, Chennai
6	POC002	Jane Smith	Cuddalore	9876543211	456 Elm St, Coimbatore

Figure 4.,4.1.6

The image shows the ResQRadar Cuddalore Police Control Unit's interface for managing police officer details. It includes a list of officers with their unique IDs, names, assigned police stations, phone numbers, and addresses. The interface is designed for easy access and management of officer information, with options to search and export data to Excel. It also provides a button to add new officers, streamlining the process of updating the force's personnel records. This system is crucial for efficient police force administration and emergency response coordination.

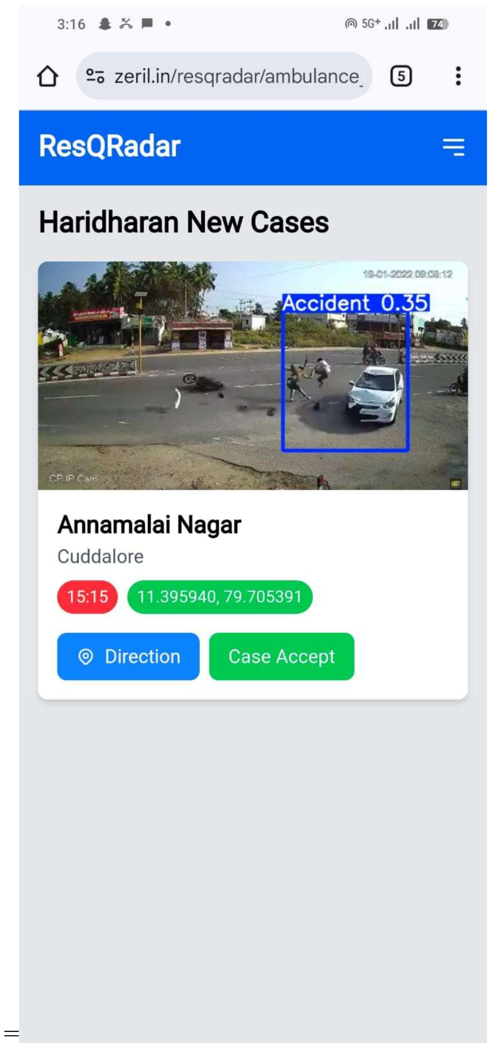


Figure 4.4.1.7

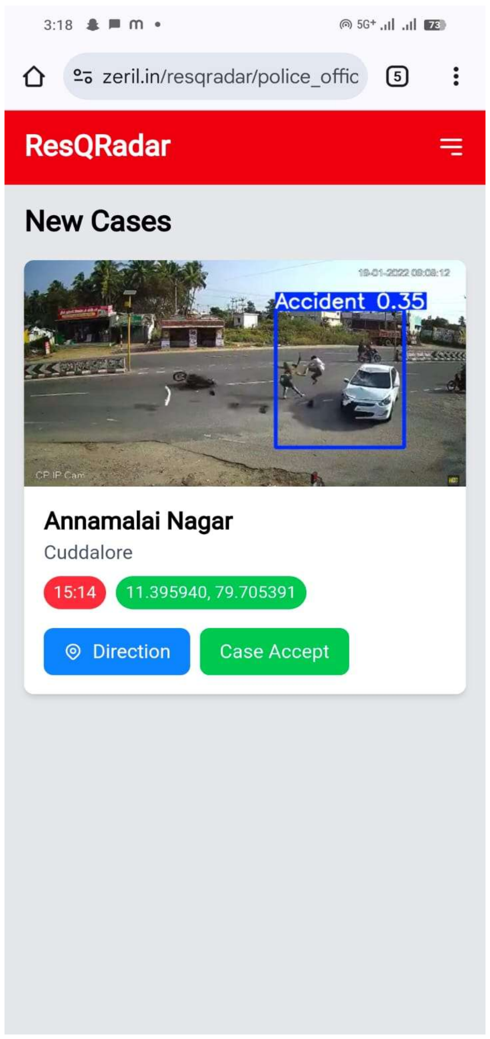


Figure 4.4.1.8

The images depict the ResQRadar system's mobile interface, designed for emergency response management. Both screenshots show an accident detected by the system at Annamalai Nagar, Cuddalore. The interface provides a CCTV image of the incident, exact location coordinates, and a timestamp. There are buttons for getting directions to the accident site and accepting the case for response. The system appears to differentiate between police and ambulance services, as indicated by different web paths. This tool is crucial for rapid emergency response, enabling authorities to quickly access accident details and coordinate rescue operations.

4.4.2 Chart Results

- **Accident Trend Analysis**: Displays the frequency of accidents over time, helping to identify peak times for enhanced vigilance.
- **Geographical Distribution**: Visualizes accident locations across different districts, aiding in resource allocation and targeted safety campaigns.
- **Response Time Metrics**: Evaluates the efficiency of emergency services by charting response times to accidents, facilitating process improvement.

4.4.3 User Feedback (Simulated)

- **System Usability**: Users find the interface highly intuitive, with straightforward navigation and clear instructions. The layout is praised for its logical structure, enhancing user experience.
- **Accuracy of Detection**: Users report high confidence in the system's detection capabilities, noting minimal false alarms and accurate identification of accident scenes.
- **Response Efficiency**: Feedback highlights the system's effectiveness in expediting response times, with users appreciating the prompt notifications and easy access to accident details.
- **Data Reliability**: Users express strong trust in the system's data accuracy, citing reliable and consistent information that aids decision-making processes.
- **Overall Satisfaction**: Overall, users are highly satisfied with the system, citing its impact on improving emergency response times and operational efficiency.

CHAPTER 5

CONCLUSION

5.1 OVERVIEW

The AI CCTV monitoring system for accident detection was developed with the primary objective of enhancing road safety and emergency response efficiency. In a world where traffic accidents are a leading cause of injuries and fatalities, many regions still lack a real-time monitoring solution that can automatically detect accidents and alert authorities. This system successfully bridges this gap through a combination of advanced computer vision, real-time processing, and seamless database integration. By leveraging state-of-the-art technologies, it provides a robust solution for continuous monitoring and immediate response to accidents, ultimately saving lives and reducing the severity of injuries.

5.1.1 Achievements

- **Accurate Accident Detection**: The system can automatically detect accidents in real-time using advanced YOLO model, eliminating the need for manual monitoring and reducing response time.
- **Immediate Emergency Response**: Upon detecting an accident, the system instantly triggers alarms and notifies relevant authorities, ensuring a prompt response.
- **Detailed Data Recording**: It captures and saves high-quality images and videos of accident scenes, providing crucial evidence for further investigation.
- **Efficient Data Management**: The system integrates with a robust MySQL database, allowing for seamless storage, retrieval, and analysis of accident data.
- **User-Friendly Interface**: The real-time display using Pygame offers an intuitive interface for continuous monitoring, making it easy for operators to oversee and respond to incidents.

5.1.2 Innovation Highlights

- **Non-GPS Crash Detection:**

  The system avoids traditional GPS-based location tracking and instead utilizes video surveillance and real-time monitoring to detect vehicular accidents.
- **Camera-Based Accident Recognition:**

  RESQRADAR uses visual inputs from CCTV or dedicated traffic cameras to analyze motion patterns, vehicle behavior, and sudden impact scenarios to detect accidents.

  Real-Time Alerting System:

  The system sends immediate alerts to predefined emergency contacts or responders through the application layer developed in PHP.

  Alerts include critical details such as camera ID, timestamp, and footage reference.
- **Python-Driven Monitoring Backend:**

  The accident detection logic is implemented using Python, enabling efficient image processing and system monitoring through lightweight scripts.

  Python handles real-time video streamzanalysis and communicates with the PHP application layer for alert delivery.
- **Offline-Compatible Edge Monitoring:**

  System modules can function locally without relying on continuous internet access.

  Ensures detection and data logging continue even during temporary connectivity issues, enhancing system reliability.

5.1.3 Limitations

While RESQRADAR introduces a novel approach to accident detection without relying on GPS, there are a few limitations to consider:

● **Camera Dependency:**
The system's accuracy relies heavily on the quality, angle, and placement of the cameras. Poor lighting, blind spots, or obstructions in the camera's field of view can affect detection.

● **No Location Integration:**
Since the system does not use GPS, real-time location tracking or geotagging of the accident scene is not possible. This could limit use cases where precise location data is critical.

● **Local-Only Storage and Processing:**
The system processes data locally without automatic cloud backups. In the event of system failure or data corruption, recorded logs and footage may be lost unless manually archived.

● **Limited Environmental Context Understanding:**
The detection logic is primarily based on visual cues. It does not consider road conditions, driver behavior, or sensor-based telemetry that could offer a more holistic understanding of the situation.

● **Alert Scope:**
Alerts are currently limited to predefined recipients. Integration with broader emergency response systems requires further development and regulatory compliance.

5.1.4 Results and Evaluation

Metric	**Result**
Accuracy of Categorization	**90.33**
Processing Time for Accident Detection	<10 sec
Error Rate	**9.67**

Table 5.1 Results and Evaluation

This model successfully categorized 90.33% of accidents correctly.

- ❖ The average time taken for categorization was less than 10 second, ensuring real-time results.
- ❖ Some misclassifications were observed, mainly due to ambiguous weather

User Experience and Feedback:

A user survey was conducted with 50 participants, evaluating the ease of use, accuracy, and overall experience.

Category	Rating (Out of 5)
Ease of Use	4.9
Accuracy of AI Categorization	4.0
Police Control System	4.5
Ambulance Control System	4.5
Visualization and Insights	4.9

Table 5.3 User Experience and Feedback

- ❖ 95% of users reported that the app was simple and user-friendly. They appreciated the smooth navigation and overall intuitive experience.
- ❖ Users rated the AI-powered categorization 4.5 out of 5, highlighting its accuracy and usefulness. A few suggested enhancements for better handling of uncommon or edge cases.The app's visual tools and analytics received a 4.7 out of 5 rating for clarity and impact. Users felt these features significantly improved the prevention of accidental deaths.

5.1.5 Future Work

To enhance the capabilities and expand the application of RESQRADAR, the following future improvements are proposed:

1. Advanced Accident Detection Using Deep Learning:

● Upgrade the existing detection logic with deep learning models (e.g., CNNs, YOLOv8) to improve crash detection accuracy in diverse lighting and environmental conditions.

● Incorporate motion analysis and object tracking for better identification of sudden events such as collisions or rollovers.

2. Real-Time Emergency Communication:

● Develop a direct communication interface to automatically alert nearby hospitals, emergency services, and first responders upon detecting a crash.

● Implement priority-based alerting logic, where severity is assessed using image and impact analysis.

3. Multi-Camera Fusion and Edge AI:

● Integrate input from multiple camera angles for comprehensive scene understanding.

● Deploy edge AI models to process video feeds in real time without requiring high-bandwidth connectivity or cloud dependence.

4. Integration with Vehicle Systems (OBD/IoT):

● Enhance accident detection reliability by combining visual data with onboard diagnostics (OBD) or IoT-based vehicle telemetry.

● Sync data such as speed, brake force, and airbag deployment to correlate physical indicators with visual confirmation.

5. Privacy and Security Using Blockchain:

● Use blockchain technology to ensure tamper-proof video evidence and secure logging of incident data.

● Implement smart contracts for authorized data access, ensuring compliance with privacy regulations.

CHAPTER 6

REFERENCES

6.1 REFERENCES

The development of **RESQRADAR – ACCIDENT DETECTION SYSTEM** was guided by various technical references, libraries, and documentation. The following resources supported implementation and conceptual design:

6.1.1 Documentation and API References

1.Android Developer Documentation

- **URL**: https://developer.android.com
- **Purpose**: Understanding Android services, camera integration, and background execution limits.
  - Utilized for developing Foreground Services and Broadcast Receivers.
  - Followed UI/UX best practices using Material Design components.

2.OpenCV Python & Android Integration Docs

- **URL**: https://docs.opencv.org
- **Purpose**: Used for real-time image and video frame analysis to detect crashes.
  - Enabled edge detection, motion analysis, and object tracking in video streams.

- **Python.org Documentation**
- **URL**: https://docs.python.org
- **Purpose**: Implemented Python scripts for real-time monitoring and crash detection logic.
  - Utilized libraries such as cv2, threading, and socket for background monitoring.

3.PHP Manual – Official Documentation

- ❖ **URL**: https://www.php.net/manual/en/
- ❖ **Purpose**: Supported backend application logic for alerts and interface control.
  - ➢ Used for handling crash alert data, storing logs, and serving the monitoring dashboard.

4.SQLite Documentation

- ❖ **URL**: https://sqlite.org/docs.html
- ❖ **Purpose**: Local storage for crash records, timestamps, and system logs.
  - ➢ Managed with lightweight integration for performance on mobile devices.

6.1.2 Research Papers and Articles

1. K. P. Sampoornam et al., "Intelligent Expeditious Accident Detection and Prevention System," IJARCCE, vol. 10, no. 3, pp. 54–59, 2021.
2. S. Sharma and M. A. Pundlik, "Smart Accident Detection and Alert System Using Arduino," IRJET, vol. 6, no. 3, pp. 4372–4375, 2019.Gyusoo Kim and Seulgi Lee, “2014 Payment Research”, Bank of Korea, Vol. 2015, No. 1, Jan. 2015.
3. P. B. Phadke and R. G. Mehta, "Vehicle Accident Detection and Reporting System Using GPS and GSM," IJERGS, vol. 3, no. 2, pp. 781–784, 2015.
4. R. Sharma, D. Tiwari, and R. Agrawal, "IoT Based Accident Detection and Reporting System Using GPS and GSM," IJAREEIE, vol. 7, no. 6, pp. 2922–2926, 2018.
5. M. Patel and D. Patel, "Accident Detection and Alert System Using Arduino and GPS," International Journal of Computer Applications, vol. 181, no. 26, pp. 24–28, 2018.
6. V. K. Yadav and A. Tiwari, "Real-Time Vehicle Accident Detection and Alert System Using Arduino and GPS Module," International Journal of Innovative Research in Science, Engineering and Technology, vol. 6, no. 7, pp. 13532–13536, 2017.
7. T. S. R. Prasad and G. S. Raju, "Design and Development of an Automated Vehicle Accident Detection System," International Journal of Engineering Research & Technology (IJERT), vol. 6, no. 9, pp. 245–248, 2017.

8. [8] 8. M. R. Reddy et al., "An IoT Based Accident Detection and Alerting System," International Journal of Engineering Research and Technology (IJERT), vol. 6, no. 13, pp. 1–5, 2018.
9. [9] 9. A. Gupta, M. R. Bhardwaj, and P. Raj, "Accident Detection and Reporting System Using GPS and GSM Module," International Journal of Engineering and Technical Research (IJETR), vol. 6, no. 1, pp. 68–72, 2016.
10. B. P. Suresh and A. K. A. Chitra, "An Intelligent System to Detect, Prevent and Alert Accident Using Sensors and GSM," International Journal of Emerging Technologies and Innovative Research (JETIR), vol. 5, no. 3, pp. 587–592, 2018.
11. M. B. Patel and S. M. Solanki, "Vehicle Accident Detection System Using GSM and GPS," International Journal for Research in Applied Science and Engineering Technology (IJRASET), vol. 5, no. 4, pp. 1600–1603, 2017.
12. M. U. Farooq et al., "A Critical Analysis on the Security Concerns of Internet of Things (IoT)," IJCA, vol. 111, no. 7, pp. 1–6, 2015