

# **IOT BASED ACCIDENT DETECTION AND RESCUE SYSTEM**

Submitted in partial fulfillment of the requirements for the award of

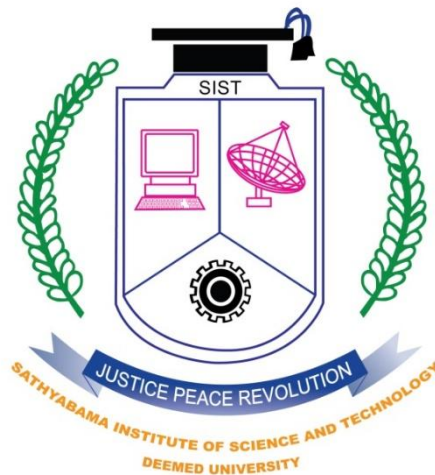
Bachelor of Engineering Degree in  
Electrical and Electronics Engineering

by

**BENIN JM (Register No. 39140008)**

**DINESH M (Register No. 39140013)**

**BREEZE JM (Register No.39140010)**



**DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING  
SCHOOL OF ELECTRICAL AND ELECTRONICS**

## **SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119**

**APRIL 2023**



**SATHYABAMA**

INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE  
[www.sathyabama.ac.in](http://www.sathyabama.ac.in)

DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

**BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafied work of BENIN JM (Register No. 39140008), DINESH M (Register No. 39140013) and BREEZE JM (Register No. 39140010) who carried out the project entitled "IOT BASED ACCIDENT DETECTION AND RESCUE SYSTEM" under our supervision from November 2022 to April 2023.

*[Signature]*  
25/4/23

DR.D.GODWIN IMMANUEL  
Internal Guide

*[Signature]*  
25/4/23  
Dr.V.SIVACHIDAMBARANATHAN  
Head of the Department

Submitted for Viva voce Examination held on 26/4/23

Name: Dr. S. RAGHAVA

Signature: *[Signature]*

Internal Examiner

*[Signature]*  
Dr. V. Balaraman

*[Signature]*

External Examiner

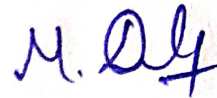
## DECLARATION

We BENIN JM (Register No. 39140008), DINESH M (Register No. 39140013) and BREEZE JM (Register No.39140010) hereby declare that the Project Report entitled "IOT BASED ACCIDENT DETECTION AND RESCUE SYSTEM" done by us under the guidance of Dr.D.GODWIN IMMANUEL is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Electrical and Electronics Engineering.

1.



2.



3.



DATE: 25/04/23

PLACE: Chennai

SIGNATURE OF THE CANDIDATES

## ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr. N. M. Nandhita., Dean, School of Electrical and Electronics Engineering** and **Dr. V. Sivachidambaranathan, Head of the Department, Dept. of Electrical and Electronics Engineering** for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Dr. D. Godwin Immanuel** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the Department of Electrical and Electronics Engineering who were helpful in many ways for the completion of the project.

## **ABSTRACT**

A lot of accidents in the highways, remote areas or during night times go unnoticed. Almost 40% of deaths due to accidents can be prevented if the victim is given treatment within limited time after the mishap. An increase in population and per capita incomes has led to an increase in ownership and the presence of vehicles on roads. Greater traffic volumes, over-speeding, reckless and drunken driving, driver fatigue, poor road infrastructure, and the presence of animals on roads are some reasons that are responsible for road fatalities. According to the World Health Organization (WHO), the percentage of road accident fatalities to the total number of deaths world- wide has increased by 2.2%. Approximately 1.35 million people die due to road accidents every year. In many cases, human lives are lost in road accidents due to delays in emergency medical assistance. The percentage of people who die before reaching the hospital in low- and middle-income countries is more than twice as compared to high-income countries. In this project we make use of a hardware setup consisting of an accelerometer sensor, and ESP32 module fixed to a bike. Two web applications and a mobile application is developed in this project. A web application using reactJS is developed for the admin to enter the list of all hospitals and another web application is developed for the hospitals to view the list of all accident cases in the vicinity. A mobile application is developed using react native for the user to enter his/her details with vital information such as blood group, medical history, emergency contact number, etc. Once an accident is detected by the hardware module, the user's location pushed to the database, all the hospitals in the vicinity will be effectively notified and the one that approves of the request will send an ambulance to save the patient.

## TABLE OF CONTENTS

| CHAPTER<br>NO | TITLE                                    | PAGE<br>NO. |
|---------------|------------------------------------------|-------------|
|               | <b>ABSTRACT</b>                          | vi          |
|               | <b>LIST OF FIGURES</b>                   | ix          |
|               | <b>LIST OF SYMBOLS AND ABBREVIATIONS</b> | x           |
| 1             | <b>INTRODUCTION</b>                      | 1           |
|               | 1.1 GENERAL INTRODUCTION                 | 1           |
|               | 1.2 TECHNOLOGIES USED                    | 2           |
|               | 1.2.1 Embedded System                    | 2           |
|               | 1.2.2 ReactJS                            | 4           |
|               | 1.2.3 React native app development       | 5           |
|               | 1.3 OBJECTIVES OF THE PROJECT            | 7           |
|               | 1.4 SCOPE OF THE PROJECT                 | 7           |
| 2             | <b>LITERATURE SURVEY</b>                 | 8           |
|               | 2.1 INTRODUCTION                         | 8           |
|               | 2.2 LITERATURE SURVEY                    | 8           |
|               | 2.3 IEEE STANDARDS                       | 17          |
| 3             | <b>PROPOSED SYSTEM</b>                   | 18          |
|               | 3.1 EXISTING SYSTEM                      | 18          |
|               | 3.2 DISADVANTAGES OF EXISTING SYSTEM     | 18          |
|               | 3.3 PROPOSED SYSTEM                      | 18          |

|   |       |                                   |    |
|---|-------|-----------------------------------|----|
|   | 3.4   | ADVANTAGES OF PROPOSED SYSTEM     | 19 |
|   | 3.5   | SYSTEM ARCHITECTURE               | 19 |
|   | 3.6   | WORKING                           | 20 |
|   | 3.7   | CASE STUDY                        | 21 |
| 4 |       | <b>SYSTEM ANALYSIS</b>            | 23 |
|   | 4.1   | MODULE DESCRIPTION                | 23 |
|   | 4.1.1 | Power Supply                      | 23 |
|   | 4.1.2 | ESP32 Module                      | 24 |
|   | 4.1.3 | Accelerometer sensor              | 25 |
|   | 4.1.4 | Database integration              | 27 |
|   | 4.1.5 | Mobile application development    | 28 |
|   | 4.1.6 | Web application development       | 29 |
| 5 |       | <b>SOFTWARE DESCRIPTION</b>       | 31 |
|   | 5.1   | ARDUINO                           | 31 |
|   | 5.2   | VISUAL STUDIO                     | 32 |
|   | 5.3   | EXTREME BURNER                    | 33 |
|   | 5.4   | ANDROID STUDIO                    | 34 |
| 6 |       | <b>RESULTS AND DISCUSSION</b>     | 36 |
|   | 6.1   | INTRODUCTION                      | 36 |
|   | 6.2   | RESULTS OBTAINED                  | 36 |
| 7 |       | <b>CONCLUSION AND FUTURE WORK</b> | 41 |
|   | 7.1   | CONCLUSION                        | 41 |
|   |       | REFERENCES                        | 42 |

## LIST OF FIGURES

| <b>FIGURE<br/>NO</b> | <b>NAME OF THE FIGURE</b>              | <b>PAGE<br/>NO.</b> |
|----------------------|----------------------------------------|---------------------|
| 1.1                  | Structure of embedded systems          | 4                   |
| 1.2                  | ReactJS architecture                   | 5                   |
| 3.1                  | Hardware block diagram                 | 20                  |
| 3.2                  | Proposed system architecture           | 21                  |
| 4.1                  | Power Supply                           | 24                  |
| 4.2                  | Microcontroller ESP32                  | 25                  |
| 4.3                  | Accelerometer sensor                   | 27                  |
| 4.4                  | MongoDB database integration           | 28                  |
| 4.5                  | Mobile application development         | 30                  |
| 4.6                  | Web application development            | 31                  |
| 5.1                  | Android Studio                         | 37                  |
| 6.1                  | Complete hardware setup of the project | 39                  |
| 6.2                  | Accident login page                    | 40                  |
| 6.3                  | Accident home page                     | 41                  |
| 6.4                  | Hospital login page                    | 41                  |
| 6.5                  | Hospital home page                     | 42                  |
| 6.6                  | Login page of mobile application       | 43                  |
| 6.7                  | Home page of mobile application        | 44                  |

## **LIST OF SYMBOLS AND ABBREVIATIONS**

|        |                                    |
|--------|------------------------------------|
| IOT    | Internet Of Things                 |
| NODEJS | Node Java Script                   |
| RTOS   | Real Time Operating System         |
| HW     | Hardware                           |
| SW     | Software                           |
| CPU    | Central Processing Unit            |
| BHS    | British Hypertension Society       |
| RAM    | Random Access Memory               |
| IDE    | Integrated Development Environment |
| IOT    | Internet Of Things                 |
| NODEJS | Node Java Script                   |

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 GENERAL INTRODUCTION**

India is one of the busiest countries in the world in terms of road traffic. The automotive industry across the south Asian country became the fourth largest in the world in 2017. In 2019, there were almost three million new car registrations in the country. The Indian Road network, spanning over five million kilometres, carried almost 90 percent of the country's passenger traffic and about 65 percent of the goods. With the rapid increase in the number of cars and the mercilessly congested Indian roads, road safety has turned into a factor of utmost importance for the country's citizens.

Accidents on road have become a major concern for the people as well as the government. About three to five percent of the GDP was invested in road accidents each year. India roughly accounts for just about one percent of the global vehicle population. However, it accounted for about six percent of the total global road accidents. In 2018, there were around 151 thousand deaths due to road accidents in India. One of the contributing factors could be the ever-increasing vehicle population. In the last decade, the road network across the country grew by about a third of its original length. Vehicle registrations on the contrary, increased by almost three times. The majority of the accidents involved two-wheelers, that also dominate the Indian automotive industry in terms of production and sales. Over-speeding was another significant contributor to road accidents in India. Driving under alcohol influence, hit and run cases, and general traffic violations resulted in almost 80 percent of the accidents being the driver's fault. Citizens aged between 18 and 45 years were involved in about 70 percent of the road accidents.

Lack of proper infrastructure for vehicles as well as pedestrians has presumably caused a spike in road accidents across India. The capital city of Delhi reported the highest number of deaths across the major cities, at approximately 1.4 thousand in 2018. Pedestrians in the capital region were the leading victims of road accidents that year. Considerable efforts were made by the Delhi Traffic Police to

curb down traffic incidents. Over 141 thousand on-the-spot prosecutions were made by the police for over-speeding. The highest number of traffic violations and prosecution slips were handed to two-wheeler riders in the capital.

Motor vehicle insurance is one of the ways for the victims to be compensated. Under the Motor Vehicle Act of 1988, third-party insurance was made compulsory for all motor vehicles. The insurance has unlimited liability and the premium amounts were calculated by courts based on victims age and financial earnings. Between 2009 and 2014, there was an increase of about 250 percent in the third-party insurance earned premium.

To further reduce damage to lives, the fine for speeding was increased by ten times under the new Motor Vehicles Act of 2019. The fines for drunk driving were also raised. A possibility for imprisonment for illegal street racing was also introduced by the law. Strict enforcement of these laws, however, will likely be a key factor in mitigating the road accident cases across India. Proper driver training and meticulous planning of the fast-growing road networks could also be key factors towards improving road safety in India.

## **1.2 TECHNOLOGIES USED**

### ***1.2.1 Embedded system***

An embedded system is a microprocessor-based computer hardware system with software that is designed to perform a dedicated function, either as an independent system or as a part of a large system. At the core is an integrated circuit designed to carry out computation for real-time operations.

Complexities range from a single microcontroller to a suite of processors with connected peripherals and networks; from no user interface to complex graphical user interfaces. The complexity of an embedded system varies significantly depending on the task for which it is designed.

Embedded system applications range from digital watches and microwaves to hybrid vehicles and avionics. As much as 98 percent of all microprocessors manufactured are used in embedded systems.

Embedded systems are managed by microcontrollers or digital signal processors (DSP), application-specific integrated circuits (ASIC), field-programmable gate arrays (FPGA), GPU technology, and gate arrays. These processing systems are integrated with components dedicated to handling electric and/or mechanical interfacing.

Embedded systems programming instructions, referred to as firmware, are stored in read-only memory or flash memory chips, running with limited computer hardware resources. Embedded systems connect with the outside world through peripherals, linking input and output devices.

### ***Structure of embedded systems***

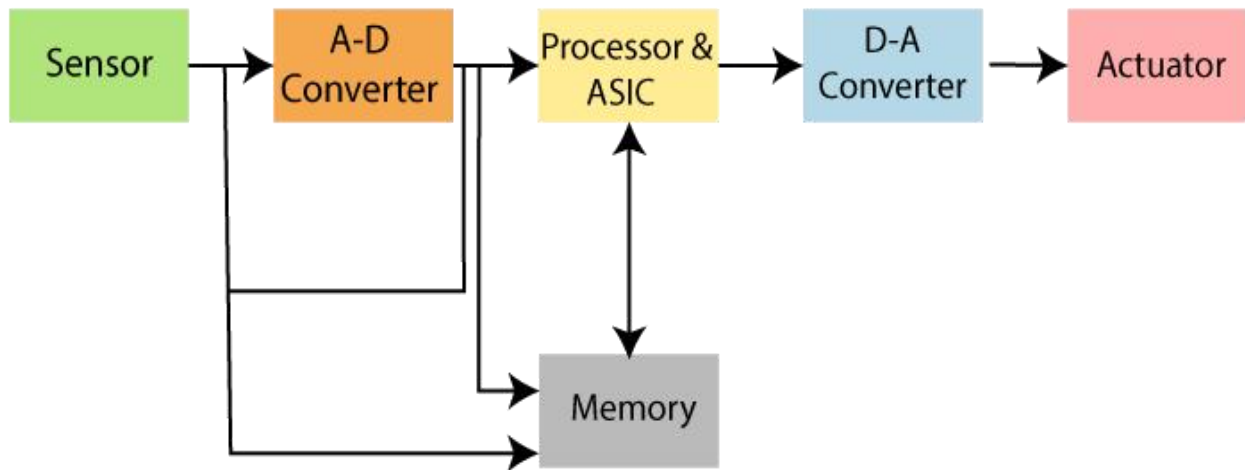
The basic structure of an embedded system includes the following components:

**Sensor:** The sensor measures and converts the physical quantity to an electrical signal, which can then be read by an embedded systems engineer or any electronic instrument. A sensor stores the measured quantity to the memory.

**A-D Converter:** An analog-to-digital converter converts the analog signal sent by the sensor into a digital signal. **Processor & ASICs:** Processors assess the data to measure the output and store it to the memory.

**D-A Converter:** A digital-to-analog converter changes the digital data fed by the processor to analog data

**Actuator:** An actuator compares the output given by the D-A Converter to the actual output stored and stores the approved output.



**Fig. 1.1 Structure of embedded systems**

### ***Benefits of using embedded systems***

- Small size and faster to load
- More specific to one task
- Easy to manage
- Low cost
- Spend less resources
- This operating system is dedicated to one device so performance is good and use less resources like memory and micro-processors

### **1.2.2 ReactJS**

ReactJS basically is an open-source JavaScript library which is used for building user interfaces specifically for single page applications. It's used for handling view layer for web and mobile apps. React also allows us to create reusable UI components. React was first created by Jordan Walke, a software engineer working for Facebook. React first deployed on Facebook's newsfeed in 2011 and on Instagram.com in 2012.

React allows developers to create large web applications which can change data, without reloading the page. The main purpose of React is to be fast, scalable, and simple. It works only on user interfaces in application. This corresponds to view

in the MVC template. It can be used with a combination of other JavaScript libraries or frameworks, such as Angular JS in MVC.

#### Features

- **JSX:** JSX stands for JavaScript XML. It is an XML/ HTML like syntax used by React. It extends the ECMAScript so that XML/ HTML like text can co-exist along with JavaScript react code. This syntax is used by the pre-processors like *Babel* to transform HTML like text found in JavaScript files into standard JavaScript objects. With JSX, we can go a step further by again embedding the HTML code inside the JavaScript. This makes HTML codes easy to understand and boosts JavaScript's performance while making our application robust.



**Fig. 1.2 ReactJS architecture**

#### **1.2.3 React native app development**

Cross-platform development has become a great alternative to fully native mobile app development. Following the native mobile development approach, you create separate apps for Android and iOS. The cross-platform development allows you to cut expenses and save time by using the same code across both platforms. The React Native framework is a rising mobile solution and is considered the future of cross-platform mobile app development.

React Native is an open-source framework that allows to build a mobile app with only JavaScript. It was introduced by Jordan Walke, a Facebook software engineer, as a new technology for simpler development and a better user experience.

The main distinctive of this framework is that React Native apps function just like native apps. They don't differ from apps built on Java, Objective-C or Swift and they use the same UI building blocks as native iOS or Android apps. But with React Native, building a mobile app is much faster and less expensive.

### Community Matters

React Native is an open-source platform. That means all documentation related to this technology is open for everyone and is available for free to everyone in React Native community. There's a great advantage to using a community-driven technology. For example, if you face any issue related to React Native development, anyone can get help from community experts or find information online.

### Reusable code & pre-built components

This is probably the most important advantage of React Native. With the convenience of this framework, developers don't need to build a separate mobile app for each platform, as almost 90 percent of React Native code can be reused between iOS and Android.

### Simplified UI

React Native is all about the mobile UI. Comparing this framework to AngularJS or MeteorJS, it looks more like a JavaScript library than a framework. It's important to create a sequence of actions when building a mobile app, and React Native makes an implementing order just perfect. In addition, user interfaces designed in React Native are more responsive, decrease load time, and provide a smoother feel.

### Third-party plugin support

The React Native framework is still advancing, so it might lack some components in the core framework. To fill this gap, React Native provides two types of third-party plugins: native modules and JavaScript modules.

### **1.3 OBJECTIVE OF THE PROJECT**

- The objective of the project is to integrate hospitals into a platform to help and save lives of an accident victim.
- To develop two web application using reactJS for the hospital and the admin.
- To develop a mobile application using react native for the user to fill form of user details.

### **1.4 SCOPE OF THE PROJECT**

- It can be used for general road safety.
- It can be used in hospitals.
- It can be used in clinics.
- It can be used for government agencies.

## **CHAPTER - 2**

### **LITERATURE SURVEY**

#### **2.1 INTRODUCTION**

The following shows survey did for Human Action Recognition technology. The most popular of the existing techniques is been discussed as follows.

#### **2.2 LITERATURE SURVEY**

##### **Near Real-Time Freeway Accident Detection**

In this paper, the problem of detecting accidents using speed sensors distributed spatially on a freeway is considered. Due to the significant impact of road accidents on health and development, early and accurate detection of accidents is crucial. To address this issue, a novel Bayesian quickest change detection formulation is introduced, which considers both average detection delay and false alarm rate. The optimum strategy is derived via dynamic programming and shown to compare a recursively computed statistic with a function of false alarm to identify accidents as they happen. Considering that post-accident conditions are typically not known, two methods are proposed that recursively estimate multiple unknown parameters during the accident detection process. Further, four aggregation methods are proposed to improve performance exploiting spatial correlations between sensors. Extensive evaluation results demonstrate improvement up to 65.2% and 87.2% in average detection delay and false alarm rate, respectively, against prior work.

##### **Machine Learning for Severity Classification of Accidents Involving Powered Two Wheelers**

Road traffic safety is one of the major challenges for the future of smart cities and transportation networks. Despite several solutions exist to reduce the number of fatalities and severe accidents happening daily in our roads, this reduction is smaller than expected and new methods and intelligent systems are needed. The emergency Call is an initiative of the European Commission aimed at providing rapid assistance to motorists thanks to the implementation of a unique emergency number. In this work, we study the problem of classifying the severity of accidents involving Powered Two Wheelers, by exploiting machine learning systems based on features that could be reasonably collected at the moment of the accident. An extended study on the set of features allows to identify the most important factors that enable to distinguish accident severity. The system we develop achieves around 90% of precision and recall on a large, publicly available corpus, using only a set of eleven features.

### **Haptic Feedback to Assist Bus Drivers for Pedestrian Safety at Low Speed**

Buses and coaches are massive Passenger Transportation Systems (PTS), because they represent more than half of land PTS in the European Union. Despite of that, bus accident figures are lower than other means of transport, but its size and weight increase the severity of accidents in which buses are involved, even at low speed. In urban scenarios, turnings and manoeuvres around bus stops are the main causes of accidents, mostly due to low visibility, blind spots or driver's distractions. Therefore, there is an increasing interest in developing driving assistance systems to avoid these situations, among others. However, even though there are some solutions on the market, they are not meant to work in urban areas at low speed and with the sole purpose of preventing collisions with pedestrians. In this sense, the paper proposes an active safety system for buses in manoeuvres at low speed. The safety system consists of haptic feedback devices together with collision avoidance and risk evaluation systems based on detected people nearby the bus. The performance of the active safety system has been validated in a simulated urban scenario. Our results show that driver's reaction time is reduced and time to collision increased due to the proposed low-speed active safety system. In particular, it is shown that there is a reduction in the number of high-risk cases and collisions, which implies a considerable improvement in safety terms. In addition to this, a brief

discussion about current regulations for innovative safety systems on a real vehicle is carried out.

### **Inevitable Collision States for Motorcycle-to-Car Collision Scenarios**

This paper presents a method to identify inevitable collision states (ICS) specifically for a motorcycle when interacting with an opponent passenger car in typical traffic scenarios. Previous ICS methods were applied to passenger cars or generic vehicles; however, the peculiarities of motorcycles urge the definition of specific methods for these vehicles. The findings extend the applicability of previous algorithms to include all motorcycle-to-car collisions, irrespective of collision configurations. ICS identification can be adopted as a triggering criterion for more invasive safety systems such as motorcycle autonomous emergency braking (MAEB), which require a last-resort approach in their initial phases of development. In this regard, this paper also presents an evaluation of an idealized MAEB through experiments simulating real-world crashes in a computer-based virtual environment.

### **Rear-End Collision Avoidance-Based on Multi-Channel Detection**

With multi-sensor-based collision avoidance systems (CASs) being adopted in today's automobiles, a new method that enables collaborative decision-making with preceding vehicle detection under various external environments is needed. In this paper, spatial-temporal correlations of multi-channel signals that are collected by multiple sensors on the host vehicle are considered, and a multi-channel detection technique with a stochastic model is introduced for automobile collision avoidance. We propose an accurate and robust multi-channel, generalized likelihood ratio test (GLRT)-based detection and collaborative decision-making scheme, with a vehicle kinematic analysis for avoiding rear-end collisions. The results of simulations and physical experiments demonstrated that our detector expands the detection range with a high detection rate and that our proposed scheme obtains good performance under varying operating and environmental conditions.

### **Collision Detection by Networked Sensors**

The advances in sensor technologies enable real-time collection of high-fidelity spatiotemporal data on transportation networks of major cities. We consider a set of speed sensors that are spatially distributed along a street and can

communicate via an exogenously determined network. In this paper, we address the problem of detecting in real-time collisions that occur within a certain distance from each sensor. The speed sensors observe the average speed value of the cars at regular time intervals and adopt a threshold-based approach to generate local predictions. Each sensor exchanges its local predictions with its neighbours and aggregates the local predictions it receives using a weighted majority aggregation rule to generate a final prediction. Since collisions are eventually reported (e.g., by a police officer or by crowd-sourced information), we assume that the information about the real occurrence of a collision is eventually given to the sensors. We propose an online learning rule that exploits this feedback to adapt the weights that each sensor gives to different local predictions. In the realizable case, i.e., when there exist unknown weights that would allow the sensors to distinguish between collisions and normal traffic behaviours, we determine an upper bound for the worst-case misdetection and false alarm probabilities of our scheme. We evaluate our scheme with traffic datasets collected from the segment of the 405 freeway that passes through Los Angeles County and the results show the efficacy of the proposed approach.

## **A Pythagorean-Type Fuzzy Deep Denoising Autoencoder for Industrial Accident Early Warning**

Early warning is crucial for preventing industrial accidents and mitigating damage, but current methods are often time-consuming, error-prone, and incompetent to deal with uncertainty. The paper presents a fuzzy deep neural network for early warning of industrial accidents, which equips the classical deep denoising autoencoder (DDAE) model with Pythagorean type fuzzy parameters in order to enhance the model's representation ability and robustness. To efficiently train the fuzzy deep model, we propose a hybrid algorithm combining Hessian-free (HF) optimization and biogeography-based optimization (BBO) metaheuristic to balance global search and local search. Experiments on datasets from several industrial zones in China show that the proposed Pythagorean-type fuzzy DDAE (PFDDAE) can achieve much higher accuracy of accident risk classification than the classical DDAE and the fuzzy DDAE using regular fuzzy parameters, and the proposed hybrid learning algorithm exhibits significant performance advantage over some other learning algorithms in training PFDDAE. In particular, a test on the 2014

Kunshan aluminum dust explosion accident shows that the deep learning model would be very likely to prevent the accident if it was adopted in advance.

### **Frailty Models for the Estimation of Spatiotemporally Maximum Congested Impact Information on Freeway Accidents**

The objective of this paper is to develop models for the estimation of the temporal and spatial extent of congestion impact caused by accidents. Although there have been various approaches based on the deterministic queuing diagrams and kinematic wave (or shockwave) theory, only a few studies have been able to estimate the spatiotemporal congested region based on field data, such as ubiquitous loop detector data. Accordingly, this paper applies a previously developed procedure to capture the spatiotemporal accident impacts based on binary integer programming (BIP). The procedure provides a foundation for models of the following: 1) maximum spatial distance to the end of the congestion region affected by each accident and 2) maximum time affected by congestion resulting from each accident. Based on these models, the objective of this paper is to estimate two statistical models for providing maximum congested distance and time information due to freeway accidents. Since various observations from BIP were censored with respect to time and space, survival analysis—specifically, frailty models to account for unobserved heterogeneity—is applied to identify factors critical to spatiotemporal congestion impacts of freeway accidents.

### **Blockchain-Enabled Certificate-Based Authentication for Vehicle Accident Detection and Notification in Intelligent Transportation Systems**

As the communications among the vehicles, the Road-Side Units (RSU) and the Edge Servers (ES) take place via wireless communication and the Internet, an adversary may take the opportunity to tamper with the data communicated among various entities in an Internet of Vehicles (IoV) environment. Therefore, it demands secure communication among the involved entities in an IoV-based Intelligent Transportation System (ITS) deployment. In this work, we design a new blockchain-enabled certificate-based authentication scheme for vehicle accident detection and notification in ITS, called BCASVADN. In BCAS-VADN, through the authentication process, each vehicle can securely notify accident related transactions to its nearby Cluster Head (CH), if an accident is detected on roads either by its own or neighbour

vehicle(s). The CH then securely sends the transactions received from the vehicles to its RSU and subsequently, these transactions are also received secretly by the ESs. The ES is responsible for preparing partial block containing transactions and Merkle tree root, and a digital signature on those information, and then forwarding to its associated Cloud Server (CS) in the Blockchain Centre (BC) for complete block creation, verification and addition of the block using the designed consensus process. Due to blockchain technology usage, it is shown that BCAS-VADN is not only secure against various potential attacks, but also maintains transparency, immutability and decentralization of the information. Furthermore, a comprehensive comparative analysis reveals that BCAS-VADN achieves better security and more functionality attributes, and has low communication and computational overheads as compared to other competitive authentication schemes in IoV. In addition, the practical demonstration using the blockchain technology has been also provided.

### **Deep Spatio-Temporal Representation for Detection of Road Accidents Using Stacked Autoencoder**

Vision-based detection of road accidents using traffic surveillance video is a highly desirable but challenging task. In this paper, we propose a novel framework for automatic detection of road accidents in surveillance videos. The proposed framework automatically learns feature representation from the spatiotemporal volumes of raw pixel intensity instead of traditional hand-crafted features. We consider the accident of the vehicles as an unusual incident. The proposed framework extracts deep representation using denoising autoencoders trained over the normal traffic videos. The possibility of an accident is determined based on the reconstruction error and the likelihood of the deep representation. For the likelihood of the deep representation, an unsupervised model is trained using one class support vector machine. Also, the intersection points of the vehicle's trajectories are used to reduce the false alarm rate and increase the reliability of the overall system. We evaluated our proposed approach on real accident videos collected from the CCTV surveillance network of Hyderabad City in India. The experiments on these real accident videos demonstrate the efficacy of the proposed approach.

## **Driver Behavior Analysis for Safe Driving**

Driver drowsiness and distraction are two main reasons for traffic accidents and the related financial losses. Therefore, researchers have been working for more than a decade on designing driver inattention monitoring systems. As a result, several detection techniques for the detection of both drowsiness and distraction have been proposed in the literature. Some of these techniques were successfully adopted and implemented by the leading car companies. This paper discusses and provides a comprehensive insight into the well-established techniques for driver inattention monitoring and introduces the use of most recent and futuristic solutions exploiting mobile technologies such as smartphones and wearable devices. Then, a proposal is made for the active of such systems into car-to-car communication to support vehicular ad hoc network's (VANET's) primary aim of safe driving. We call this approach the dissemination of driver behavior via C2C communication. Throughout this paper, the most remarkable studies of the last five years were examined thoroughly in order to reveal the recent driver monitoring techniques and demonstrate the basic pros and cons. In addition, the studies were categorized into two groups: driver drowsiness and distraction. Then, research on the driver drowsiness was further divided into two main subgroups based on the exploitation of either visual features or nonvisual features. A comprehensive compilation, including used features, classification methods, accuracy rates, system parameters, and environmental details, was represented as tables to highlight the (dis)advantages and/or limitations of the aforementioned categories. A similar approach was also taken for the methods used for the detection of driver distraction.

## **Accelerated Evaluation of Automated Vehicles in Car-Following Maneuvers**

The safety of automated vehicles (AVs) must be assured before their release and deployment. The current approach to evaluation relies primarily on 1) testing AVs on public roads or 2) track testing with scenarios defined in a test matrix. These two methods have completely opposing drawbacks: the former, while offering realistic scenarios, takes too much time to execute and the latter, though it can be completed in a short amount of time, has no clear correlation to safety benefits in the

real world. To avoid the aforementioned problems, we propose accelerated evaluation, focusing on the car-following scenario. The stochastic human-controlled vehicle (HV) motions are modeled based on 1.3 million miles of naturalistic driving data collected by the University of Michigan Safety Pilot Model Deployment Program. The statistics of the HV behaviors are then modified to generate more intense interactions between HVs and AVs to accelerate the evaluation procedure. The importance sampling theory was used to ensure that the safety benefits of AVs are accurately assessed under accelerated tests. Crash, injury and conflict rates for a simulated AV are simulated to demonstrate the proposed approach. Results show that test duration is reduced by a factor of 300 to 100 000 compared with the non-accelerated (naturalistic) evaluation. In other words, the proposed techniques have great potential for accelerating the AV evaluation process.

### **Multimodal Features for Detection of Driver Stress and Fatigue**

Driver fatigue and stress significantly contribute to higher number of car accidents worldwide. Although, different detection approaches have been already commercialized and used by car producers (and third-party companies), research activities in this field are still needed in order to increase the reliability of these alert systems. Also, in the context of automated driving, the driver mental state assessment will be an important part of cars in future. This paper presents state-of-the-art review of different approaches for driver fatigue and stress detection and evaluation. We describe in details various signals (biological, car and video) and derived features used for these tasks and we discuss their relevance and advantages. In order to make this review complete, we also describe different datasets, acquisition systems and experiment scenarios.

### **Assignment of Legal Responsibilities for Decisions by Autonomous Cars Using System Architectures**

Increasing automation in road vehicles moves decision making from humans to algorithms. This gives a lack of clarity in assigning responsibility for the consequences of the resultant actions, presenting problems for ethicists, lawyers,

and engineers. This article applies proven methods based on the 4D/RCS reference model architecture to derive proposed architectures for autonomous vehicles with J3016 autonomy levels 2–5. The architectures have a hierarchy of nodes, which can be either a human or an automated process, giving unambiguous allocation of authority to act and consequent responsibilities to individual nodes in the system. A concept called authorized power is introduced for every node, giving clear limits on each node's ability to act. Setting node requirements gives a firm basis for engineers and lawyers to agree where responsibilities lie, whether with the user, owner, or in the supply chain. It also provides a framework to assess engineering risk and the wider financial implications of the new technologies as they are introduced.

## **Towards Next-Generation Vehicles Featuring the Vehicle Intelligence**

Safe driving and minimizing the number of casualties are the main motivations of researchers and car companies for decades. They also care very much on saving fuel consumption and high comfort-level trips. With the help of advanced driver assistance systems (ADAS) applications, safer, more comfortable, and greener trips are very likely at the present time. However, today humankind is very close to make a very old dream, namely, driverless vehicles, to come true. In this paper, we address the concept of next-generation vehicles, their requirements, challenges, advantages, and problems. Regarding Society of Automotive Engineers (SAE), levels (1–5), we first define the crucial contexts for next-generation vehicles. We then discuss existing ADAS, their abilities, and available platforms which they run on from past to present. Next, we introduce a novel vehicle intelligence (VI) architecture consisting of ADAS modules and VI services which would pave the way for fully autonomous vehicles regarding not only driving issue, but also human-centric new demands such as entertainment and comfort level of the journey. The proposed conceptual design is built on sensors, vehicle ad hoc networks (VANETs), and big data. Afterward, we describe how current ADAS applications would transform on the way toward SAE Level 5 cars. We finally discuss the open issues for next-generation vehicles.

## 2.3 IEEE STANDARDS

**IEEE 802.15.4:** This standard specifies the physical and MAC layers for low-rate wireless personal area networks (LR-WPANs). This standard is particularly relevant for IoT-based systems that use wireless communication technologies such as Bluetooth, Zigbee, or Wi-Fi to transmit data from the accident detection sensors to the central control system.

**IEEE 802.11:** This standard specifies the protocols and procedures for wireless local area networks (WLANs). This standard is particularly relevant for IoT-based systems that use Wi-Fi to transmit data from the accident detection sensors to the central control system.

**IEEE 802.3:** This standard specifies the protocols and procedures for wired Ethernet networks. This standard is particularly relevant for IoT-based systems that use Ethernet cables to connect the accident detection sensors to the central control system.

**IEEE 1451:** This standard defines a family of smart transducer interface standards for sensors and actuators. This standard is particularly relevant for IoT-based systems that use sensors to detect accidents and emergencies and actuate emergency response mechanisms.

**IEEE 1901:** This standard specifies the protocols and procedures for high-speed powerline communication (PLC) networks. This standard is particularly relevant for IoT-based systems that use PLC to transmit data from the accident detection sensors to the central control system.

## **CHAPTER - 3**

### **PROPOSED SYSTEM**

#### **3.1 EXISTING SYSTEM**

In this paper, the problem of detecting accidents using speed sensors distributed spatially on a freeway is considered. Due to the significant impact of road accidents on health and development, early and accurate detection of accidents is crucial. To address this issue, a novel Bayesian quickest change detection formulation is introduced, which considers both average detection delay and false alarm rate. The optimum strategy is derived via dynamic programming and shown to compare a recursively computed statistic with a function of false alarm to identify accidents as they happen. Considering that post-accident conditions are typically not known, two methods are proposed that recursively estimate multiple unknown parameters during the accident detection process. Further, four aggregation methods are proposed to improve performance exploiting spatial correlations between sensors. Extensive evaluation results demonstrate improvement up to 65.2% and 87.2% in average detection delay and false alarm rate, respectively, against prior work.

#### **3.2 DISADVANTAGES OF EXISTING SYSTEM**

- The system does not focus on facilitating the hospitals to be notified when an accident occurs.
- The system cannot be used for real time implementation.

### **3.3 PROPOSED SYSTEM**

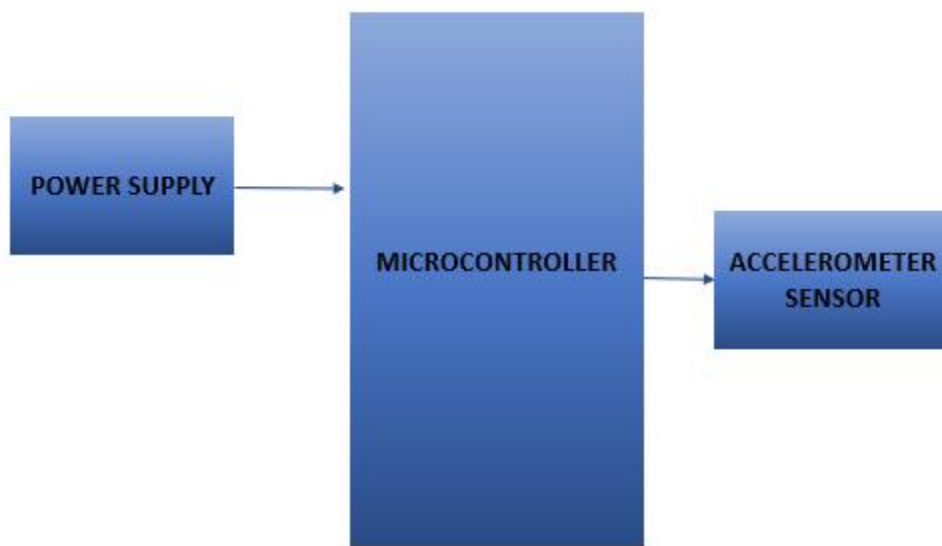
In this project we make use of a hardware setup consisting of an accelerometer sensor, and ESP32 module fixed to a bike. Two web applications and a mobile application is developed in this project. A web application using reactJS is developed for the admin to enter the list of all hospitals and another web application is developed for the hospitals to view the list of all accident cases in the vicinity. A mobile application is developed using react native for the user to enter his/her details with vital information such as blood group, medical history, emergency contact number, etc. Once an accident is detected by the hardware module, the user's location pushed to the database, all the hospitals in the vicinity will be effectively notified and the one that approves of the request will send an ambulance to save the patient.

### **3.4 ADVANTANGES OF PROPOSED SYSTEM**

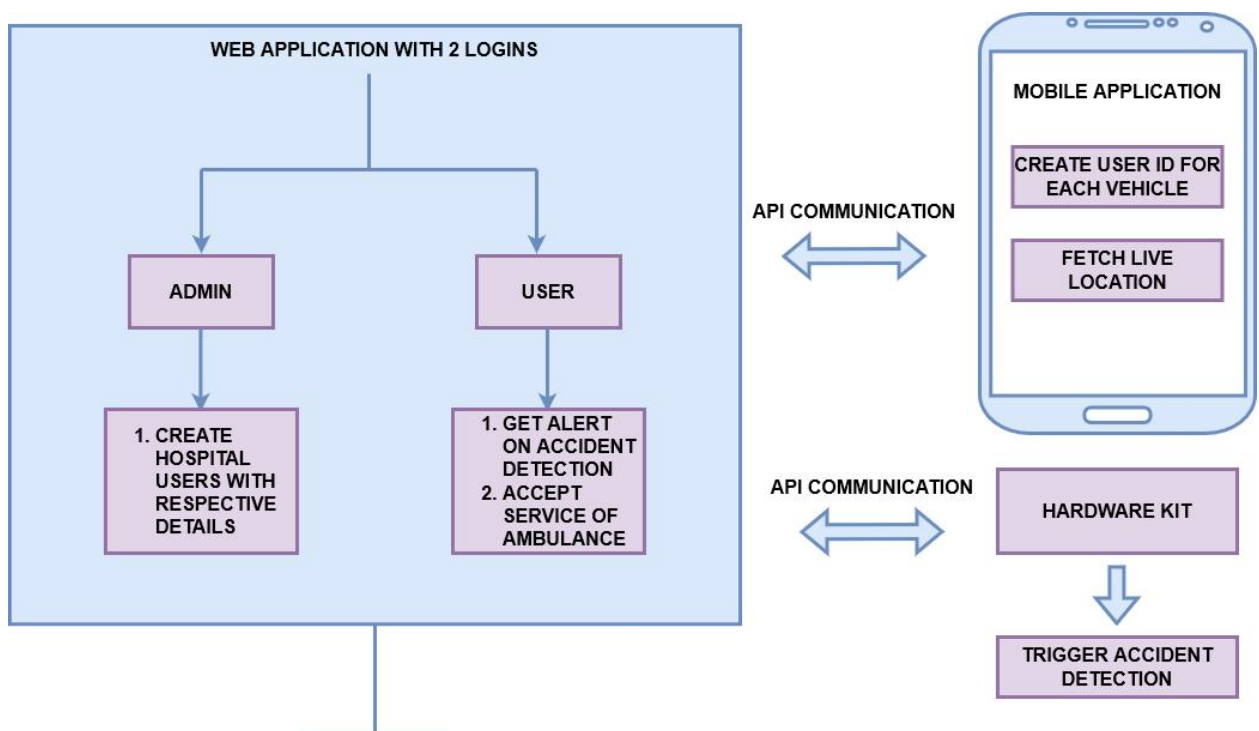
- Helps in saving lives
- Helps to notify hospitals nearby in the vicinity of accident has occurred.
- Two web applications and a mobile application is developed to facilitate the project.

### **3.5 SYSTEM ARCHITECTURE**

The hardware block diagram and proposed system architecture diagrams are shows as below respectively.



**Fig. 3.1 Hardware block diagram**



**Fig. 3.2 Proposed system architecture**

### 3.6 WORKING

We use a hardware setup for this project that consists of a bike-mounted accelerometer sensor and an ESP32 module. Both of these components are

attached to the bike. This project involves the development of three applications: two web applications and a mobile application. One web application, written in reactJS, is developed for the administrator to use in order to enter the list of all hospitals, and another web application is developed in order for hospitals to use in order to view the list of all accident cases in the area. Using react native, a mobile application is developed that allows the user to enter his or her details along with essential information such as blood group, medical history, emergency contact number, etc. After an accident has been identified by the hardware module and the user's location has been added to the database, all of the nearby hospitals will be effectively notified, and the hospital that grants the request will dispatch an ambulance to save the patient.

### **3.7 CASE STUDY**

#### **Introduction**

An accident detection and rescue system is a technology that can be used to monitor and detect accidents or emergencies and alert the relevant authorities for prompt response. In this case study, we will explore the effects of such a system after its implementation in a city with a high incidence of road accidents.

#### **Scenario**

Consider a busy city with a high volume of traffic, where road accidents are a common occurrence. Despite efforts to improve road safety, accidents still happen due to various reasons such as driver error, pedestrian negligence, and reckless driving. In this scenario, the implementation of an accident detection and rescue system can significantly improve the city's road safety and emergency response capabilities.

#### **Implementation**

The accident detection and rescue system can be designed to monitor the city's roads using various sensors and cameras. The sensors can detect any sudden changes in the traffic flow such as a vehicle slowing down or stopping abruptly, or a

pedestrian crossing the road unexpectedly. The cameras can capture images and videos of the accident site which can be used to analyze the severity of the accident and determine the required response.

The system can be connected to a centralized control center where trained personnel can monitor the traffic and respond to any emergencies. The control centre can receive real-time alerts from the sensors and cameras and dispatch emergency services such as ambulances, fire trucks, or police as required.

### **Effects after Implementation**

After the implementation of the accident detection and rescue system, the city experienced a significant reduction in road accidents and fatalities. The system's real-time monitoring and prompt response capability have enabled emergency services to respond to accidents and emergencies much faster, which has saved lives and prevented injuries.

The data collected by the system has also enabled authorities to identify patterns and trends in accidents, which has helped them to develop more targeted and effective road safety measures. The insights gained from the data analysis have led to improvements in road infrastructure, signage, and regulations.

Moreover, the implementation of the accident detection and rescue system has increased public confidence in the city's road safety and emergency response capabilities. Citizens are more aware of the system's presence and functionality, and they are more likely to report accidents and emergencies, which has further improved the system's effectiveness.

### **Conclusion**

The implementation of an accident detection and rescue system in the city has significantly improved road safety and emergency response capabilities. The system's real-time monitoring and prompt response capability have enabled emergency services to respond to accidents and emergencies much faster, which has saved lives and prevented injuries. The data collected by the system has enabled authorities to identify patterns and trends in accidents, which has helped them to develop more targeted and effective road safety measures. Overall, the implementation of the accident detection and rescue system has created a safer and more efficient city road system, which has benefited the citizens and the city as a whole.

## **CHAPTER – 4**

### **SYSTEM ANALYSIS**

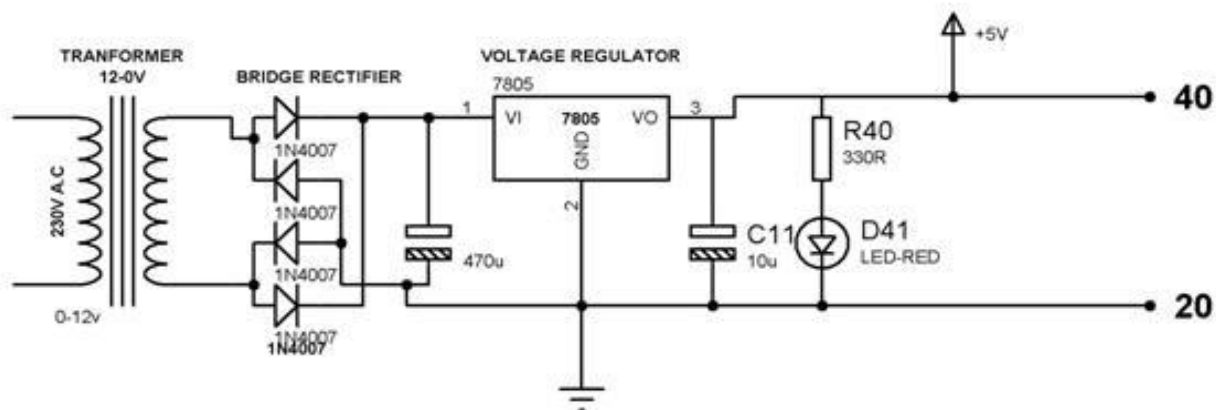
#### **4.1 MODULE DESCRIPTION**

- Power supply Module
- ESP32 Module
- Accelerometer sensor
- Database integration
- Mobile application development
- Web application development

##### ***4.1.1 Power Supply***

The power supply circuit consists of step-down transformer which is 230v step down to 12v. In this circuit 4 diodes are used to form bridge rectifier which delivers pulsating dc voltage & then fed to capacitor filter the output voltage from rectifier is fed to filter to eliminate any A.C components present even after rectification.

The filtered DC voltage is given to regulator to produce 12v constant DC voltage. 230V AC power is converted into 12V AC (12V RMS value wherein the peak value is around 17V), but the required power is 5V DC; for this purpose, 17V AC power must be primarily converted into DC power then it can be stepped down to the 5V DC. AC power can be converted into DC using one of the power electronic converters called as Rectifier. There are different types of rectifiers, such as half-wave rectifier, full-wave rectifier and bridge rectifier. Due to the advantages of the bridge rectifier over the half and full wave rectifier, the bridge rectifier is frequently used for converting AC to DC.



**Fig. 4.1 Power Supply**

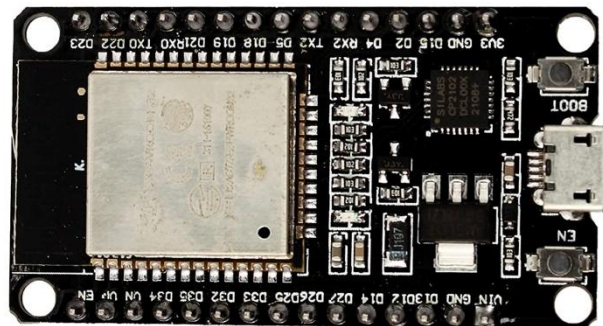
#### **4.1.2 ESP32 Module**

ESP32 is a series of low-cost, low-power system on a chip microcontroller with integrated Wi-Fi and dual-mode Bluetooth. The ESP32 series employs either a Tensilica Xtensa LX6 microprocessor in both dual-core and single-core variations, Xtensa LX7 dual-core microprocessor or a single-core RISC-V microprocessor and includes built-in antenna switches, RF balun, power amplifier, low-noise receive amplifier, filters, and power-management modules. ESP32 is created and developed by Espressif Systems, a Shanghai-based Chinese company, and is manufactured by TSMC using their 40 nm process.[2] It is a successor to the ESP8266 microcontroller.

Since the release of the original ESP32, a number of variants have been introduced and announced. They form the ESP32 family of microcontrollers. These chips have different CPUs and capabilities, but all share the same SDK and are largely code-compatible. Additionally, the original ESP32 was revised (see ESP32 ECO V3, for example). ESP32 is capable of functioning reliably in industrial environments, with an operating temperature ranging from  $-40^{\circ}\text{C}$  to  $+125^{\circ}\text{C}$ . Powered by advanced calibration circuitries, ESP32 can dynamically remove external circuit imperfections and adapt to changes in external conditions.

Engineered for mobile devices, wearable electronics and IoT applications, ESP32 achieves ultra-low power consumption with a combination of several types of proprietary software. ESP32 also includes state-of-the-art features, such as fine-

grained clock gating, various power modes and dynamic power scaling. ESP32 is highly-integrated with in-built antenna switches, RF balun, power amplifier, low-noise receive amplifier, filters, and power management modules. ESP32 adds priceless functionality and versatility to your applications with minimal Printed Circuit Board (PCB) requirements.



**Fig. 4.2 Microcontroller ESP32**

#### **4.1.3 Accelerometer sensor**

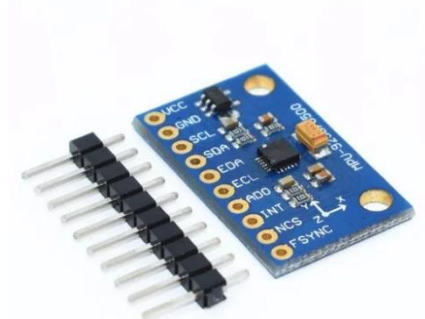
An accelerometer sensor is a tool that measures the acceleration of any body or object in its instantaneous rest frame. It is not a coordinate acceleration. Accelerometers can be used to measure vibration on cars, machines, buildings, process control systems and safety installations. They can also be used to measure seismic activity, inclination, machine vibration, dynamic distance and speed with or without the influence of gravity. An accelerometer sensor reports the acceleration of the device along the three sensor axes. The measured acceleration includes both the physical acceleration (change of velocity) and the gravity.

An accelerometer is an electronic sensor that measures the acceleration forces acting on an object, in order to determine the object's position in space and monitor the object's movement. Acceleration, which is a vector quantity, is the rate of

change of an object's velocity (velocity being the displacement of the object divided by the change in time).

There are two types of acceleration forces: static forces and dynamic forces. Static forces are forces that are constantly being applied to the object (such as friction or gravity). Dynamic forces are “moving” forces applied to the object at various rates (such as vibration, or the force exerted on a cue ball in a game of pool). This is why accelerometers are used in automobile collision safety systems, for example. When a car is acted on by a powerful dynamic force, the accelerometer (sensing a rapid deceleration) sends an electronic signal to an embedded computer, which in turn deploys the airbags.

There are three different types of accelerometers, and they are each designed to efficiently function in their intended environments. The three types are: piezoelectric, piezoresistance and capacitive. A piezoelectric accelerometer utilizes the piezoelectric effect (piezoelectric materials produce electricity when put under physical stress) to sense change in acceleration. Piezoelectric accelerometers are most commonly used in vibration and shock measurement. Piezoresistance accelerometers are much less sensitive than piezoelectric accelerometers, and they are better suited to vehicle crash testing. A piezoresistance accelerometer increases its resistance in proportion to the amount of pressure applied to it. The third and most commonly used type of accelerometer is the capacitive accelerometer. Capacitive accelerometers use change in electrical capacitance to determine an object's acceleration. When the sensor undergoes acceleration, the distance between its capacitor plates changes as the diaphragm of the sensor moves.



**Fig. 4.3 Accelerometer sensor**

#### **4.1.4 Database integration**

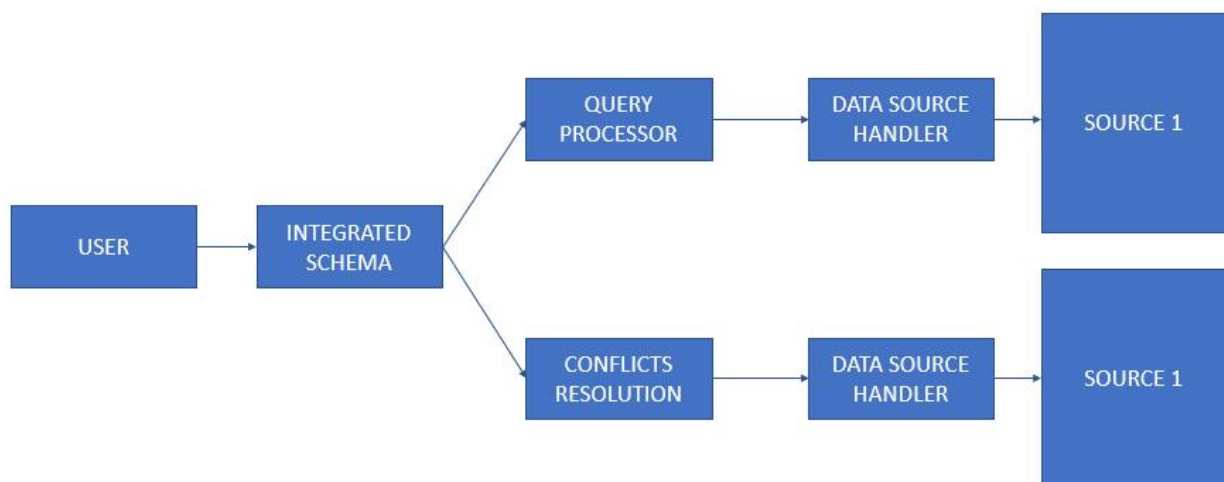
MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server-Side Public License (SSPL) which is deemed non-free by several distributions. MongoDB is a member of the MACH Alliance.

MongoDB supports field, range query, and regular-expression searches. Queries can return specific fields of documents and also include user-defined JavaScript functions. Queries can also be configured to return a random sample of results of a given size. Fields in a MongoDB document can be indexed with primary and secondary indices or index.

MongoDB provides high availability with replica sets. A replica set consists of two or more copies of the data. Each replica-set member may act in the role of primary or secondary replica at any time. All writes and reads are done on the primary replica by default. Secondary replicas maintain a copy of the data of the primary using built-in replication. When a primary replica fails, the replica set automatically conducts an election process to determine which secondary should become the primary. Secondaries can optionally serve read operations, but that data is only eventually consistent by default.

If the replicated MongoDB deployment only has a single secondary member, a separate daemon called an arbiter must be added to the set. It has a single responsibility, which is to resolve the election of the new primary. As a consequence, an idealized distributed MongoDB deployment requires at least three separate servers, even in the case of just one primary and one secondary.

MongoDB scales horizontally using sharding. The user chooses a shard key, which determines how the data in a collection will be distributed. The data is split into ranges (based on the shard key) and distributed across multiple shards. (A shard is a master with one or more replicas.) Alternatively, the shard key can be hashed to map to a shard – enabling an even data distribution. MongoDB can run over multiple servers, balancing the load or duplicating data to keep the system up and running in case of hardware failure.



**Fig. 4.4 MongoDB database integration**

#### ***4.1.5 Mobile application development***

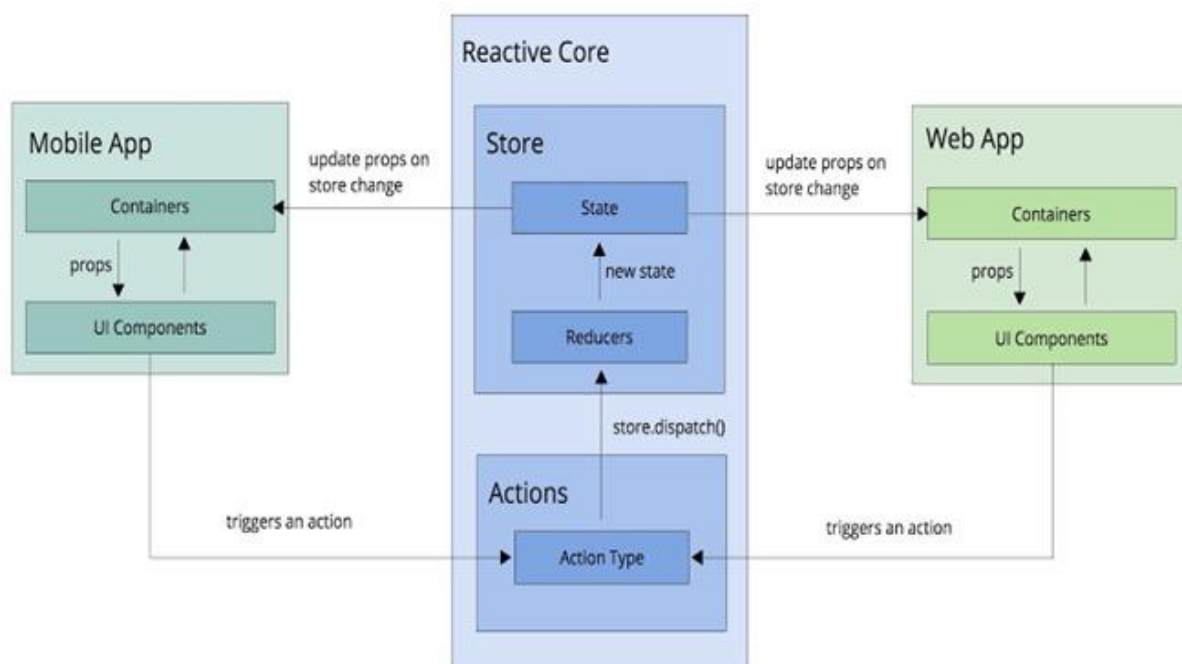
Flask algorithms in python are made use to communicate with the mobile app so that the images can be transmitted and received to predict the disease presence. When an input image is given the features of the image is compared with all other features of the model file. The most predicted feature will be estimated and the results will be displayed. A camera integration or an image input will be used for final outcome of the results in mobile app.

A native mobile app is developed as an interface using a JavaScript framework react native. The mobile app enables the user to look for available slots for parking in the preferred location. It also enables the user to book the parking slot of his/her location. React Native (also known as RN) is a popular JavaScript-based mobile app framework that allows you to build natively-rendered mobile apps for iOS and Android. The framework lets you create an application for various platforms by using the same codebase.

React Native was first released by Facebook as an open-source project in 2015. In just a couple of years, it became one of the top solutions used for mobile development. React Native development is used to power some of the world's leading mobile apps, including Instagram, Facebook, and Skype. We discuss these and other examples of React Native-powered apps further in this post.

There are several reasons behind React Native's global success. Firstly, by using React Native, companies can create code just once and use it to power both their iOS and Android apps. This translates to huge time and resource savings. Secondly, React Native was built based on React – a JavaScript library, which was already hugely popular when the mobile framework was released. Thirdly, the framework empowered frontend developers, who could previously only work with web-based technologies, to create robust, production-ready apps for mobile platforms.

React Native is written with a mixture of JavaScript and JXL, a special markup code resemblant of XML. The framework has the ability to communicate with both realms – JavaScript-based threads and existent, native app threads.



**Fig. 4.5 Mobile application development**

#### **4.1.6 Web application development**

ReactJS is JavaScript library used for building reusable UI components. According to React official documentation, following is the definition –

React is a library for building composable user interfaces. It encourages the creation of reusable UI components, which present data that changes over time. Lots of people use React as the V in MVC. React abstracts away the DOM from you,

offering a simpler programming model and better performance. React can also render on the server using Node, and it can power native apps using React Native. React implements one-way reactive data flow, which reduces the boilerplate and is easier to reason about than traditional data binding.

## JSX

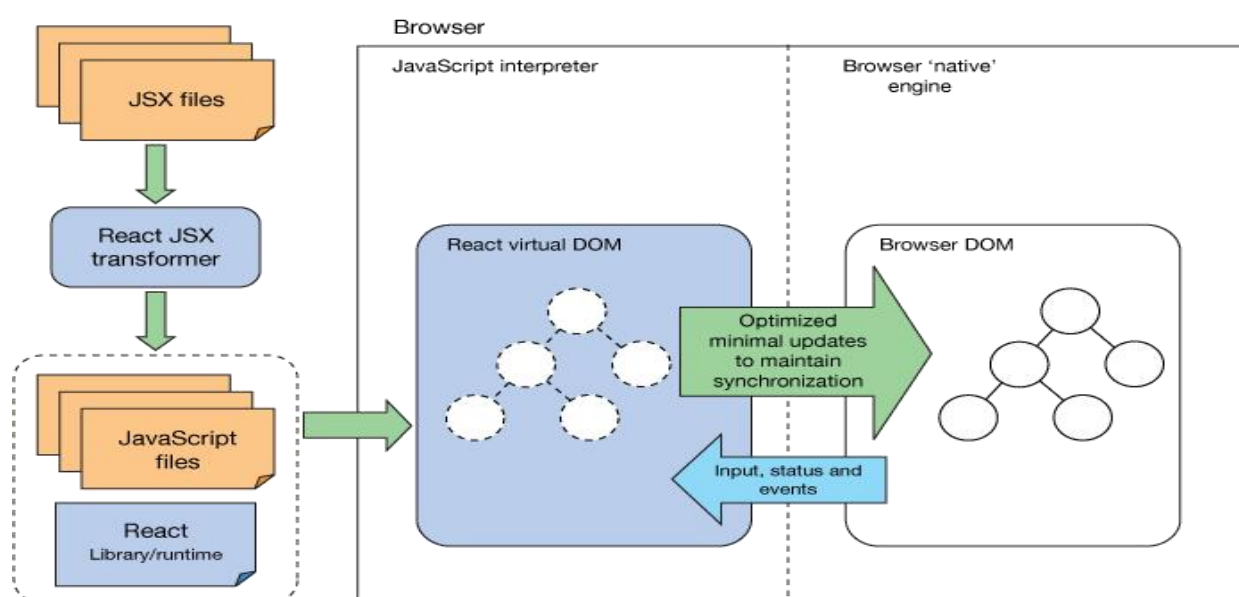
JSX stands for JavaScript XML. JSX allows us to write HTML in React. JSX makes it easier to write and add HTML in React. It is faster than normal JavaScript as it performs optimizations while translating to regular JavaScript. It makes it easier for us to create templates. Instead of separating the markup and logic in separated files, React uses components for this purpose. We will learn about components in details in further articles.

## Components

Components are independent and reusable bits of code. They serve the same purpose as JavaScript functions, but work in isolation and return HTML via a render function. Components come in two types, Class components and Function components, in this tutorial we will concentrate on Class components.

## Unidirectional data flow and Flux

React implements one-way data flow which makes it easy to reason about your app. Flux is a pattern that helps keeping your data unidirectional.



## **Fig. 4.6 Web application development**

### **CHAPTER – 5**

#### **SOFTWARE DESCRIPTION**

The purpose of the Software Requirement Specification is to produce the specification of the analysis task and also to establish complete information about the requirement, behavior and also the other constraint like functional performance and so on. The main aim of the Software Requirement Specification is to completely specify the technical requirements for the software product in a concise and in unambiguous manner.

##### **5.1 ARDUINO**

In this project the Arduino software is used.

The Arduino software is easy-to-use for beginners, yet flexible enough for advanced users. It runs on Mac, Windows, and Linux. Teachers and students use it to build low cost scientific instruments, to prove chemistry and physics principles, or to get started with programming and robotics. Designers and architects build interactive prototypes, musicians and artists use it for installations and to experiment with new musical instruments. Makers, of course, use it to build many of the projects exhibited at the Maker Faire, for example. Arduino is a key tool to learn new things. Anyone - children, hobbyists, artists, programmers - can start tinkering just following the step-by-step instructions of a kit, or sharing ideas online with other members of the Arduino community.

There are many other microcontrollers and microcontroller platforms available for physical computing. Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard, and many others offer similar functionality. All of these tools take the messy details of microcontroller programming and wrap it up in an easy-to-use package. Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems:

- Inexpensive - Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than \$50
- Cross-platform - The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.
- Simple, clear programming environment - The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.
- Open source and extensible software - The Arduino software is published as open source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, we can add AVR-C code directly into our Arduino programs if we want to.
- Open source and extensible hardware - The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money.

## **5.2 VISUAL STUDIO**

In this project we make use of the Microsoft visual studio as the IDE.

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code. Visual Studio includes a code editor supporting IntelliSense (the code completion component) as well as code refactoring. The integrated debugger works

both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that enhance the functionality at almost every level—including adding support for source control systems (like Subversion and Git) and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Azure DevOps client: Team Explorer).

Visual Studio supports 36 different programming languages and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++, C++/CLI, Visual Basic .NET, C#, F#, JavaScript, TypeScript, XML, XSLT, HTML, and CSS. Support for other languages such as Python, Ruby, Node.js, and M among others is available via plug-ins. Java (and J#) were supported in the past. The most basic edition of Visual Studio, the Community edition, is available free of charge. The slogan for Visual Studio Community edition is "Free, fully-featured IDE for students, open-source and individual developers".

### **5.3 EXTREME BURNER**

The Extreme Burner- AVR is a full graphical user interface (GUI) AVR series of MCU that supports several types of clock sources for various applications.

An AVR or Alf and Vegard's RISC processor programmer is a device used to program an AVR microcontroller. The AVR microcontroller is based off an 8-bit RISC (reduced instruction set computing) microcontroller. An AVR programmer allows the user to place operating instructions on the microcontroller that tell it to perform a specific task. AVR microcontrollers are often used in hardware development and robotics. An AVR programmer is typically expensive compared to the price of the microcontroller itself and can be purchased by itself or as a kit. The AVR programmer puts code from the computer onto the AVR microcontroller. The code is compiled on the computer into binary. This code is typically saved as a HEX file and can then be uploaded onto the microcontroller. The AVR programmer is usually plugged into the serial port of the computer, and a special program is required to get the code from the computer to the microcontroller.

The code from the computer is then transferred to the microcontroller's flash memory. The programmer is connected to certain pins on the microcontroller that allow it to be programmed. Most AVR microcontrollers have only a few kilobytes of memory, so programs put on the microcontroller must be fairly small. This process of transferring the program from the computer to the microcontroller is called flashing.

AVR microcontrollers are useful for many applications. A microcontroller is essentially a tiny computer and can be programmed to do many things. Input and output are handled through the numerous pins on the microcontroller. It is possible to hook up sensors, switches, lights, motors, and many other items to the pins, which allow it a great range of uses. Thus, AVR microcontrollers are often used for prototyping and robotics projects.

## **5.4 ANDROID STUDIO**

In this project android studio is used as the environment for mobile app development.

Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems or as a subscription-based service in 2020. It is a replacement for the Eclipse Android Development Tools (E-ADT) as the primary IDE for native Android application development.

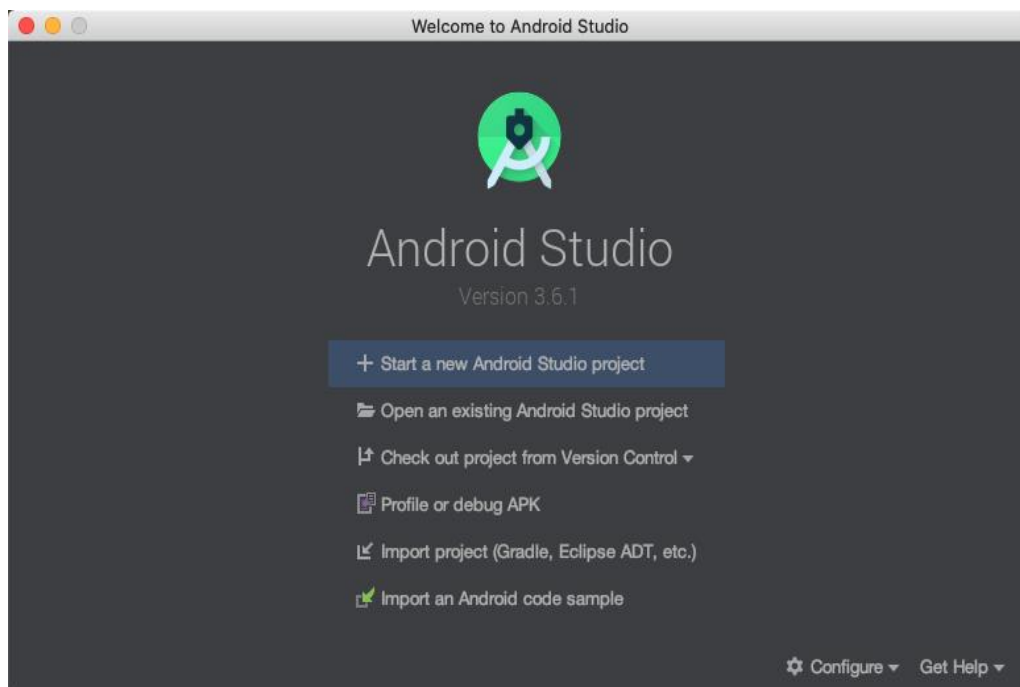
Android Studio was announced on May 16, 2013 at the Google I/O conference. It was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0.

Android Studio supports all the same programming languages of IntelliJ (and CLion) e.g. Java, C++, and more with extensions, such as Go; and Android Studio 3.0 or later supports Kotlin and "all Java 7 language features and a subset of Java 8 language features that vary by platform version." External projects backport some Java 9 features. While IntelliJ states that Android Studio supports all released Java versions, and Java 12, it's not clear to what level Android

Studio supports Java versions up to Java 12 (the documentation mentions partial Java 8 support). At least some new language features up to Java 12 are usable in Android.

## Features

- Gradle-based build support
- Android-specific refactoring and quick fixes
- Lint tools to catch performance, usability, version compatibility and other problems
- ProGuard integration and app-signing capabilities
- Template-based wizards to create common Android designs and components
- A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations
- Support for building Android Wear apps
- Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine
- Android Virtual Device (Emulator) to run and debug apps in the Android studio.



**Fig. 5.1 Android Studio**

## **CHAPTER- 6**

### **RESULTS & DISCUSSIONS**

#### **6.1 INTRODUCTION**

This chapter discusses about the practical results obtained while implementing the project.

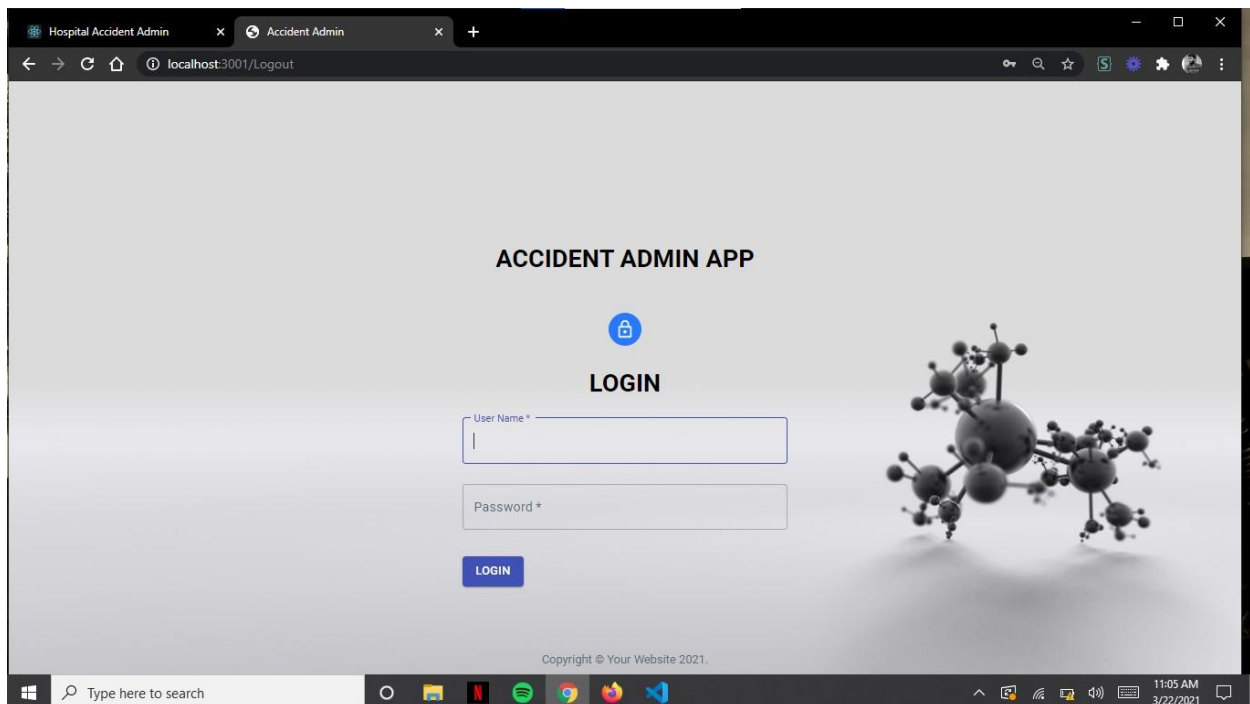
#### **6.2 RESULTS OBTAINED**

The below image shows the complete hardware setup of the project:



**Fig. 6.1 Complete hardware setup of the project**

A web application using a JavaScript framework reactJS is developed for the accident admin page of the project that is used to enter all the hospital details. The below image shows the login page of the web application:



**Fig. 6.2 Accident login page**

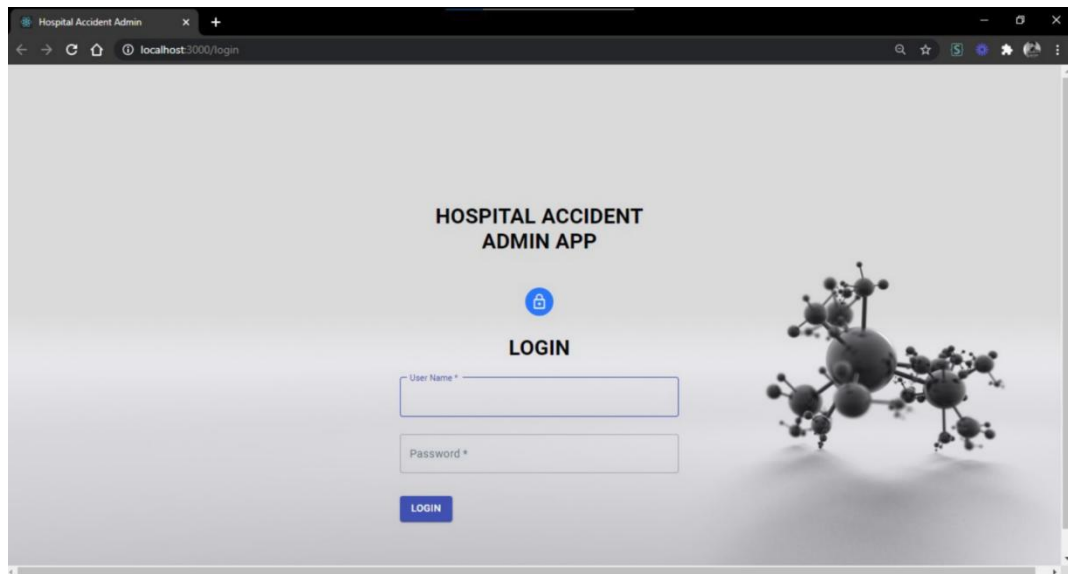
The below image shows the home page of the accident admin web application on successful login displaying the list of hospitals:

A screenshot of the 'ACCIDENT ADMIN APP' home page. The page has a blue header with the app name. On the left, there is a vertical sidebar with a blue-to-green gradient, containing three menu items: 'Add Hospital' (with a plus icon), 'View Hospital' (with a list icon), and 'Log Out' (with a logout icon). The main content area is white and features a 'Hospital Registration Form'. The form has six input fields: 'Hospital Id \*', 'Hospital Name \*', 'Password \*', 'Phone Number \*', 'Latitude \*', and 'Longitude \*'. There is also a single-line input field for 'Address \*'. A blue 'SAVE DETAILS' button is positioned below the form fields.

**Fig.**

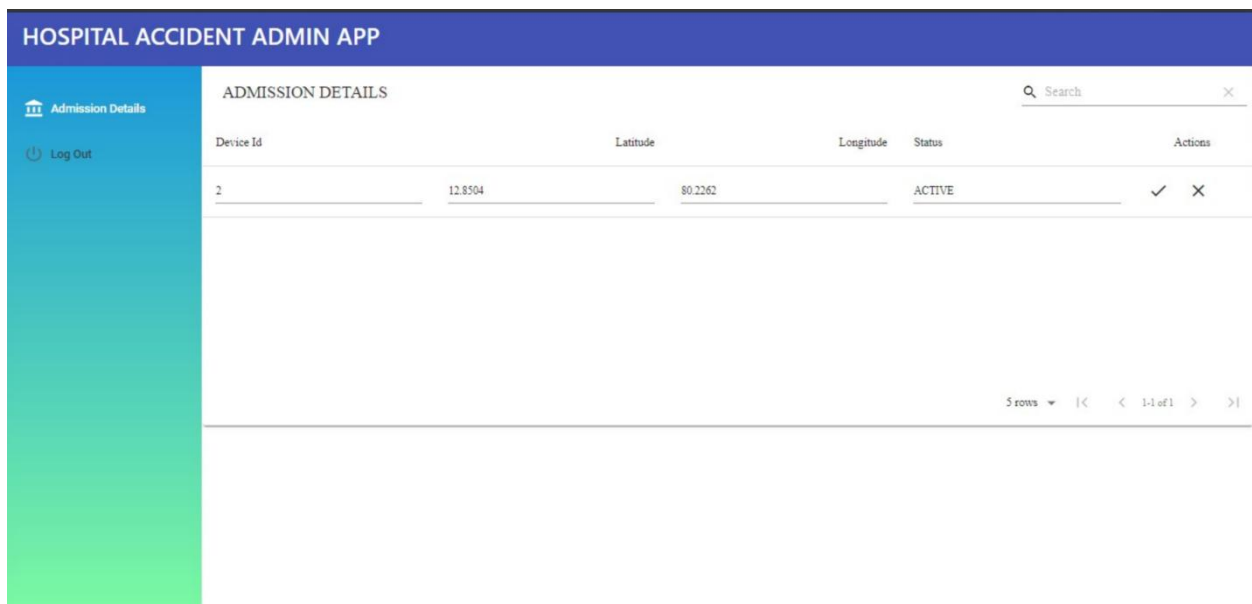
**6.3 Accident home page**

The below image shows the login page of the hospital web application:



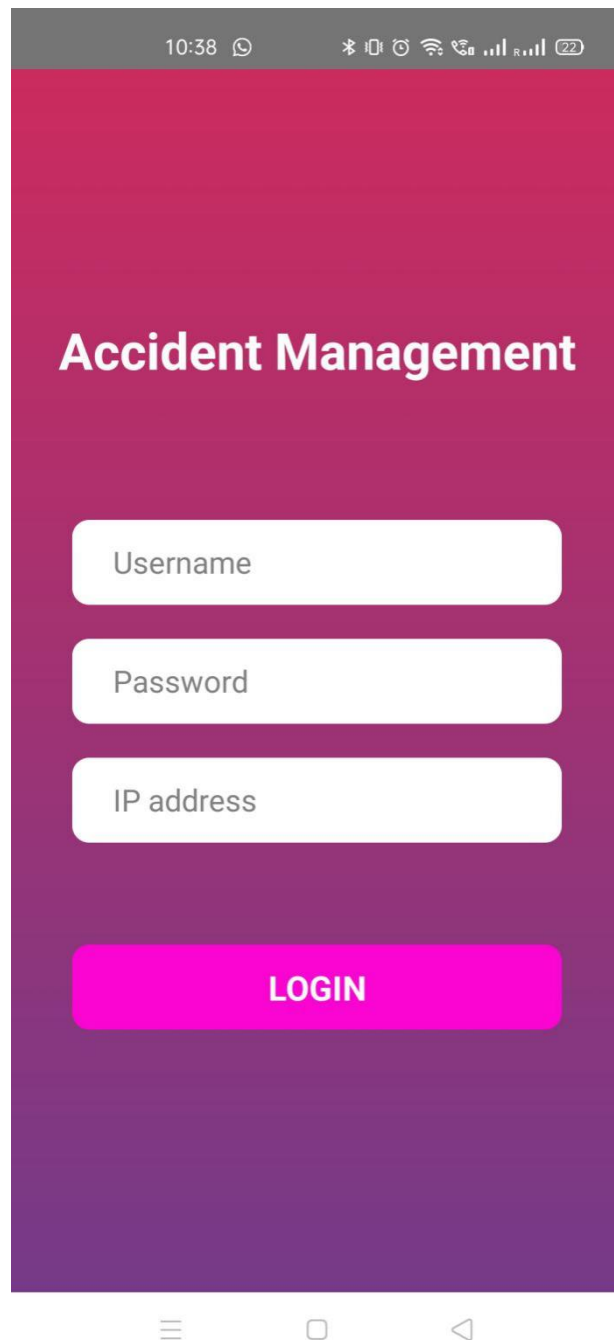
**Fig. 6.4 Hospital login page**

The below image shows the hospital registration page of the accident admin web application:



**Fig. 6.5 Hospital home page**

A mobile application using react native a JavaScript framework is developed for the user. The below image shows the login page of the mobile application:



**Fig. 6.6 Login page of mobile application**

The below image shows the home page of the mobile application on successful login:

The screenshot shows a mobile application interface for 'Patient Creation'. The form is set against a dark purple background and contains the following fields: Name, Age, Gender, Father Name, Mother Name, Device Id, Heart Attack, Bp, Weight, Blood Group, Mobile Number 1, Mobile Number 2, Mobile Number 3, and Address. A bright pink 'SUBMIT' button is located at the bottom of the form. The top of the screen displays the time as 10:38 and various status icons. The bottom of the screen shows the standard Android navigation bar.

**Fig. 6.7 Home page of mobile application**

Thus, from the above results and discussions it is clear that the project has been successfully developed and implemented.

## **CHAPTER – 7**

### **CONCLUSION & FUTURE WORK**

#### **7.1 CONCLUSION**

To summarize, an accident detection and rescue system is an essential technology that can save lives during emergencies. The system utilizes various sensors to detect accidents and immediately notifies emergency services or rescue teams. It also provides critical information, such as the accident's location and the extent of the injuries suffered by those involved.

The system can be integrated into vehicles, wearable devices, or other smart gadgets, making it readily available to everyone. Furthermore, advancements in artificial intelligence and machine learning have made it possible to develop more advanced accident detection and rescue systems that can recognize different types of accidents and react accordingly.

In general, the implementation of an accident detection and rescue system can decrease response times, enhance the likelihood of survival for accident victims, and ultimately create safer roads and communities.

## REFERENCES

1. Canzian, L., Demiryurek, U. and der Schaar, M. van (2016). Collision Detection by Networked Sensors. *IEEE Transactions on Signal and Information Processing over Networks*, 2(1), pp.1–15. doi:<https://doi.org/10.1109/tsipn.2015.2504721>.
2. Chung, Y. and Recker, W.W. (2015). Frailty Models for the Estimation of Spatiotemporally Maximum Congested Impact Information on Freeway Accidents. *IEEE Transactions on Intelligent Transportation Systems*, 16(4), pp.2104–2112. doi:<https://doi.org/10.1109/tits.2015.2394798>.
3. Gillespie, T. and Hailes, S. (2020). Assignment of Legal Responsibilities for Decisions by Autonomous Cars Using System Architectures. *IEEE Transactions on Technology and Society*, 1(3), pp.148–160. doi:<https://doi.org/10.1109/tts.2020.3014395>.
4. Girbes, V., Armesto, L., Dols, J. and Tornero, J. (2016). Haptic Feedback to Assist Bus Drivers for Pedestrian Safety at Low Speed. *IEEE Transactions on Haptics*, 9(3), pp.345–357. doi:<https://doi.org/10.1109/toh.2016.2531686>.
5. Hadjidimitriou, N.S., Lippi, M., Dell’Amico, M. and Skiera, A. (2020). Machine Learning for Severity Classification of Accidents Involving Powered Two Wheelers. *IEEE Transactions on Intelligent Transportation Systems*, 21(10), pp.4308–4317. doi:<https://doi.org/10.1109/tits.2019.2939624>.
6. Kaplan, S., Guvensan, M.A., Yavuz, A.G. and Karalurt, Y. (2015). Driver Behavior Analysis for Safe Driving: A Survey. *IEEE Transactions on Intelligent Transportation Systems*, [online] 16(6), pp.3017–3032. doi:<https://doi.org/10.1109/TITS.2015.2462084>.
7. Liyanage, Y.W., Zois, D.-S. and Chelmis, C. (2020). Near Real-Time Freeway Accident Detection. *IEEE Transactions on Intelligent Transportation Systems*, pp.1–12. doi:<https://doi.org/10.1109/tits.2020.3027494>.
8. Nemcova, A., Svozilova, V., Bucsuhaży, K., Smisek, R., Mezl, M., Hesko, B., Belak, M., Bilik, M., Maxera, P., Seitzl, M., Dominik, T., Semela, M., Sucha, M. and Kolar, R. (2021). Multimodal Features for Detection of Driver Stress and

- Fatigue: Review. *IEEE Transactions on Intelligent Transportation Systems*, 22(6), pp.3214–3233. doi:<https://doi.org/10.1109/tits.2020.2977762>.
9. Savino, G., Giovannini, F., Fitzharris, M. and Pierini, M. (2016). Inevitable Collision States for Motorcycle-to-Car Collision Scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 17(9), pp.2563–2573. doi:<https://doi.org/10.1109/tits.2016.2520084>.
  10. Singh, D. and Mohan, C.K. (2019). Deep Spatio-Temporal Representation for Detection of Road Accidents Using Stacked Autoencoder. *IEEE Transactions on Intelligent Transportation Systems*, 20(3), pp.879–887. doi:<https://doi.org/10.1109/tits.2018.2835308>.
  11. Vangala, A., Bera, B., Saha, S., Das, A.K., Kumar, N. and Park, Y. (2021). Blockchain-Enabled Certificate-Based Authentication for Vehicle Accident Detection and Notification in Intelligent Transportation Systems. *IEEE Sensors Journal*, 21(14), pp.15824–15838. doi:<https://doi.org/10.1109/jsen.2020.3009382>.
  12. Xiang, Y., Huang, S., Li, M., Li, J. and Wang, W. (2020). Rear-End Collision Avoidance-Based on Multi-Channel Detection. *IEEE Transactions on Intelligent Transportation Systems*, 21(8), pp.3525–3535. doi:<https://doi.org/10.1109/tits.2019.2930731>.
  13. Zhao, D., Huang, X., Peng, H., Lam, H. and LeBlanc, D.J. (2018). Accelerated Evaluation of Automated Vehicles in Car-Following Maneuvers. *IEEE Transactions on Intelligent Transportation Systems*, [online] 19(3), pp.733–744. doi:<https://doi.org/10.1109/tits.2017.2701846>.
  14. Zheng, Y.-J., Chen, S.-Y., Xue, Y. and Xue, J.-Y. (2017). A Pythagorean-Type Fuzzy Deep Denoising Autoencoder for Industrial Accident Early Warning. *IEEE Transactions on Fuzzy Systems*, 25(6), pp.1561–1575. doi:<https://doi.org/10.1109/tfuzz.2017.2738605>.
  15. Payalan, Y.F. and Guvensan, M.A. (2020). Towards Next-Generation Vehicles Featuring the Vehicle Intelligence. *IEEE Transactions on Intelligent Transportation Systems*, 21(1), pp.30–47. doi:<https://doi.org/10.1109/tits.2019.2917866>.

