

# Parental Driving Safety System

by

Bishoy Kamel, Daniel Nagy, Omar Mohamed, Mina Magdy

A dissertation submitted in partial fulfillment of the  
requirements for the degree of  
Bachelor of computer science

in

Department of Computer Science

in the

Faculty of Computer Science  
of the

Misr International University, EGYPT

Thesis advisor:

Dr. Mohamed El-Gazzar and Eng. Radwa Samy

(July 2020)

## Abstract

Road accidents are a worldwide disaster with an ever-rising pattern. Absence of consideration, foolish driving, disregarding driving rules take numerous lives every day. In this project, we propose a solution to diminish the number of accidents, to follow the driving conduct through the use of F- DTW, drowsiness detection and a concentration level detector for sibling drivers to create a detailed report which will be sent to the concerned person. This goal will be achieved through the use of a mobile application that uses the built-in phone accelerometer, GPS and GSM, a Drowsiness detection system, built-in speaker. To sum it all up, the goal is to notify concerned person and emergency services as quickly as possible, to reduce the number of fatalities as much as possible using all the mentioned components.

## Acknowledgments

I am heartily appreciative to Dr. Mohamed El-Gazzar and Eng. Radwa Samy in the staff of Computer Science at Misr International University for their endeavors as our guides and being our fifth individual from the group their experience and work impact have truly influenced our presentation overall pushing us forward through creation useful remarks and unequivocal activities that helped push the task to what it is presently, their office was consistently open to us, addressing our every inquiry, and all through the whole year each stage conveyance has been gainful on the two closures. - We would also like to thank Dr. Ashraf AbdelRaouf in the faculty of Computer Science at Misr International University for all the effort he has put into each and every graduation project at Misr International University and for his participation and acknowledgement with feedback and constructive criticism and for his frequent advisory.

We will consistently be in appreciation of Dr. Ayman Nabil for helping the group in various ways, for example, assembling the apparatuses important for applying in rivalries and supporting the group monetarily for distributing our paper in renowned diaries.

Also, Dr. Ayman Bahaa Dean of the staff of Computer Science at Misr International College for giving us a long time of understanding and significant information that just includes more evaluation in our work and for his on going work in further improving the workforce's worth and embodiment.

# Contents

<b>Abstract</b>	<b>ii</b>
<b>Acknowledgments</b>	<b>iii</b>
<b>List of Tables</b>	<b>4</b>
<b>List of Figures</b>	<b>5</b>
<b>1 Introduction</b>	<b>7</b>
1.0.1 Background . . . . .	7
1.0.2 Motivation . . . . .	7
1.0.3 Problem Definitions . . . . .	8
1.1 Project Description . . . . .	8
1.1.1 Objective . . . . .	9
1.1.2 Scope . . . . .	9
1.1.3 Project Overview . . . . .	10
1.2 Project Management and Deliverables . . . . .	12
1.2.1 Tasks and Time Plan . . . . .	12
1.2.2 Budget and Resource Costs . . . . .	12
1.2.3 Supportive Documents . . . . .	12
<b>2 Literature Work</b>	<b>14</b>
2.1 Similar System Information . . . . .	14
2.1.1 Similar System Description . . . . .	18
2.1.2 Comparison with Proposed Project . . . . .	18
<b>3 System Requirements Specifications</b>	<b>19</b>
3.1 Introduction . . . . .	19
3.1.1 Purpose of this document . . . . .	19
3.1.2 Scope of this document . . . . .	19
3.1.3 Overview . . . . .	19
3.1.4 Business Context . . . . .	23
3.2 General Description . . . . .	24
3.2.1 Product Functions . . . . .	24



---

3.2.2	Similar System Information . . . . .	25
3.2.3	User Characteristics . . . . .	26
3.2.4	User Problem Statement . . . . .	27
3.2.5	User Objectives . . . . .	27
3.2.6	General Constraints . . . . .	27
3.3	Functional Requirements . . . . .	28
3.3.1	User . . . . .	28
3.3.2	Accelerometer Readings . . . . .	29
3.3.3	Driver . . . . .	29
3.3.4	Analysis . . . . .	31
3.3.5	Report . . . . .	32
3.3.6	Parent . . . . .	33
3.3.7	Administrator . . . . .	33
3.4	Interface Requirements . . . . .	35
3.4.1	User Interfaces . . . . .	35
3.4.2	Communications Interfaces . . . . .	43
3.5	Performance Requirements . . . . .	43
3.6	Design Constraints . . . . .	43
3.6.1	Hardware Limitations . . . . .	43
3.7	Other non-functional attributes . . . . .	44
3.7.1	Performance And Speed . . . . .	44
3.7.2	Reliability . . . . .	44
3.7.3	Usability . . . . .	44
3.8	Preliminary Object-Oriented Domain . . . . .	45
3.8.1	Class descriptions . . . . .	45
3.9	Database Diagram . . . . .	55
3.10	Scenarios . . . . .	55
3.11	Preliminary Schedule Adjusted . . . . .	57
3.12	Preliminary Budget Adjusted . . . . .	57
<b>4</b>	<b>System Design Document</b>	<b>58</b>
4.1	Introduction . . . . .	58
4.1.1	Purpose . . . . .	58
4.1.2	Scope . . . . .	58
4.1.3	Overview . . . . .	59
4.1.4	Definitions and Acronyms . . . . .	59
4.2	System Overview . . . . .	59
4.3	System Architecture . . . . .	61
4.3.1	Architectural Design . . . . .	61
4.3.2	Decomposition Description . . . . .	65
4.3.3	Design Rationale . . . . .	79
4.4	Data Design . . . . .	80
4.4.1	Data Description . . . . .	80
4.4.2	Data Dictionary . . . . .	80
4.5	Component Design . . . . .	81

---

4.5.1	Pre-Processing . . . . .	81
4.5.2	Classification . . . . .	82
4.6	Human Interface Design . . . . .	84
4.6.1	Screen Images . . . . .	84
4.6.2	Screen Objects and Actions . . . . .	93
4.7	Requirements Matrix . . . . .	95
<b>5</b>	<b>Evaluation</b>	<b>96</b>
5.1	Introduction . . . . .	96
5.2	Experiment 1 Driving Behaviour detection classification . . . . .	96
5.2.1	Goal . . . . .	96
5.2.2	Classifiers Tested . . . . .	96
5.2.3	Task . . . . .	97
5.2.4	Results . . . . .	97
5.3	Experiment 2 Comparing proposed system with related work . . . . .	97
5.3.1	Goal . . . . .	97
5.3.2	Task . . . . .	97
5.3.3	Results . . . . .	98
<b>6</b>	<b>Conclusion</b>	<b>99</b>
6.1	Future directions . . . . .	99
	<b>Bibliography</b>	<b>100</b>

# List of Tables

Tasks and Time Plan . . . . .	12
Similar Systems . . . . .	18
Login . . . . .	28
Logout . . . . .	28
edit Profile . . . . .	28
register . . . . .	29
Read Accelerometer . . . . .	29
Start Trip . . . . .	29
Send Emergency . . . . .	30
Alert Driver . . . . .	30
View Driver Report . . . . .	30
Filter . . . . .	31
Detect Behaviour . . . . .	31
Detect Accident . . . . .	31
Face Detection . . . . .	32
Check Drowsiness . . . . .	32
Get Behaviour . . . . .	32
Find Rating . . . . .	33
View Report . . . . .	33
View User . . . . .	33
Remove User . . . . .	34
Preliminary Schedule Adjusted . . . . .	57
Definitions . . . . .	59
Requirements Matrix . . . . .	95
Results . . . . .	97

# List of Figures

1.1	General Overview . . . . .	10
1.2	Blinking Detection . . . . .	10
1.3	Driving Behaviour Analysis . . . . .	11
1.4	Accident Detection . . . . .	11
1.5	Supportive Documents . . . . .	13
3.1	Overviews . . . . .	20
3.2	Blinking Detection . . . . .	21
3.3	Driving Behavior supervision . . . . .	22
3.4	Accident Detection Medical Support . . . . .	23
3.5	Block Diagram . . . . .	24
3.6	Similar System . . . . .	26
3.7	Log in GUI . . . . .	35
3.8	SignUpAS GUI . . . . .	36
3.9	Driver SignUp GUI . . . . .	37
3.10	Parent Signup GUI . . . . .	38
3.11	Parent Verification GUI . . . . .	39
3.12	Parent Verification GUI . . . . .	40
3.13	Driver Verification GUI . . . . .	41
3.14	Start Driving GUI . . . . .	42
3.15	Driver <sub>Report</sub> GUI . . . . .	42
3.16	Driver <sub>Report</sub> GUI . . . . .	43
3.17	Preliminary Object-Oriented Domain . . . . .	45
3.18	Database Diagram . . . . .	55
3.19	Use Case Diagram . . . . .	56
4.1	System Overview . . . . .	60
4.2	Block diagram . . . . .	61
4.3	Model view controller . . . . .	63
4.4	Class Diagram . . . . .	65
4.5	Process Diagram . . . . .	76
4.6	Activity Diagram . . . . .	77
4.7	Signup . . . . .	77

---

4.8	Driving Behavior . . . . .	78
4.9	Drowsiness Detection . . . . .	78
4.10	Accident Detection . . . . .	79
4.11	Data Description . . . . .	80
4.12	Low Pass Filter . . . . .	81
4.13	DTW . . . . .	82
4.14	DTW 2 . . . . .	83
4.15	Welcome GUI . . . . .	84
4.16	Signin GUI . . . . .	85
4.17	SignUpas GUI . . . . .	86
4.18	Signup Parent GUI . . . . .	87
4.19	SignupDriver GUI . . . . .	88
4.20	Start Service GUI . . . . .	89
4.21	Drowsiness GUI . . . . .	90
4.22	Emergency GUI . . . . .	91
4.23	Driver Report GUI . . . . .	92
4.24	Parent Report GUI . . . . .	93

# Chapter 1

## Introduction

### 1.0.1 Background

Road accidents are a more common phenomenon than most people realise and based on the World Health organization (WHO), in Egypt more than 12 thousand deaths occur every year due to road accidents and thus road accidents are the leading death causality for people between the ages of 5 – 29 [1], this age is the specific region teenagers and young adults fall under. This implies that all families that own a vehicle are hesitant to allow their siblings to drive and in this modern era this solution is not smart. The alternative solution is to ensure the safety of the driver by reducing the risk and the injuries that drivers face as well as, monitoring their driving habits through the usage of mobile applications. Thus, In the proposed project, a mobile application will be developed to meet these requirements through the usage of the front phone camera that is used to track the blinking rate to deduce the physical state of the driver if they were tired or weren't concentrating. Also, the usage of the built in phone accelerometer to help clarify their driving skills and methods. Also, detect whether an accident took place or not and with the help of an alarm, GPS and GSM to alert the authorities and the people around if an accident took place as well as a obstacle notification to all nearby cars feature which will help prevent other drivers to avoid them on the road.

### 1.0.2 Motivation

For a family there is nothing more important than the well being of all its members thus, all rational and irrational measures are taken to ensure their well being. Nowadays teenage

family members are frequently utilizing the family vehicle in their daily life , this opens a risk field upon them whether from external factors or from their own recklessness. Thus all the motivation needed to start developing this project is to track the driving habits of the driver to conclude the level of their recklessness , ultimately specify the level of danger they are in.

### 1.0.3 Problem Definitions

The occurrence of road accidents happen due to the lack of concentration by the driver and as a result of this lack of concentration sudden moves and changes in the car speed or direction take place which can have fatal consequences as well as the presence of road obstacles that forces drivers to make sudden movements in an attempt to avoid them causing accidents. Also, another problem that causes a huge number of accidents is drowsiness, more importantly, when an accident happens the driver is in no condition to send any sort of notification to the emergence response teams and thus is under the mercy of the people around them if any to call the authorities as soon as possible.

## 1.1 Project Description

1. Detect any sudden or abrupt movements in the car speed and/or direction which will act as an indication to drivers ignorance .
2. Detect if an accident occurred notify the closest response team or the emergency contacts of the location of the accident.
3. Track blinking rate of the driver.
4. In case of presence of any obstacle on the road like pits or bumps, mobile phone notifies all the nearby cars about the location of this obstacle.
5. Make a buzzing sound to notify the surrounding people of the occurrence of an accident.

### 1.1.1 Objective

The main objective of this project is to help parents and car owners to make sure that their vehicles as well as the person driving it whether family or stranger are safe also, determine their driving behavior, habits and the dangers produced by it to deduce the level of recklessness thus taking the appropriate measures in addition to, the tracking of the drivers concentration state to try and reduce or prevent these dangers from falling upon the driver.

### 1.1.2 Scope

1. The system will detect minor or abrupt changes in the car velocity.
2. The system will detect if an accident had happened or not.
3. Mobile phone must be stationed and steady such that the eyes of the driver are detectable to its sensor.
4. The phone makes a buzzing alarm in case of happening of an accident.
5. The system will notify the authorities as soon as the accident happens automatically through GPS and GSM.
6. The system will notify nearby cars the presence of any obstacles on the road.
7. The system alerts drivers that close their eyes for more than 3 seconds.



### 1.1.3 Project Overview

Figure 1.1: General Overview

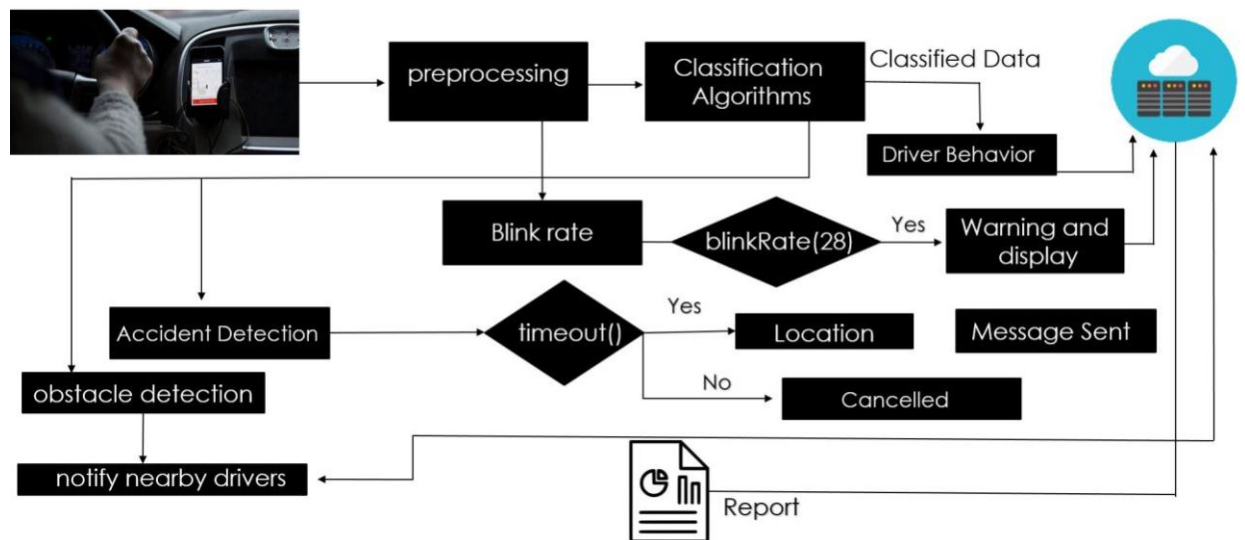


Figure 1.2: Blinking Detection

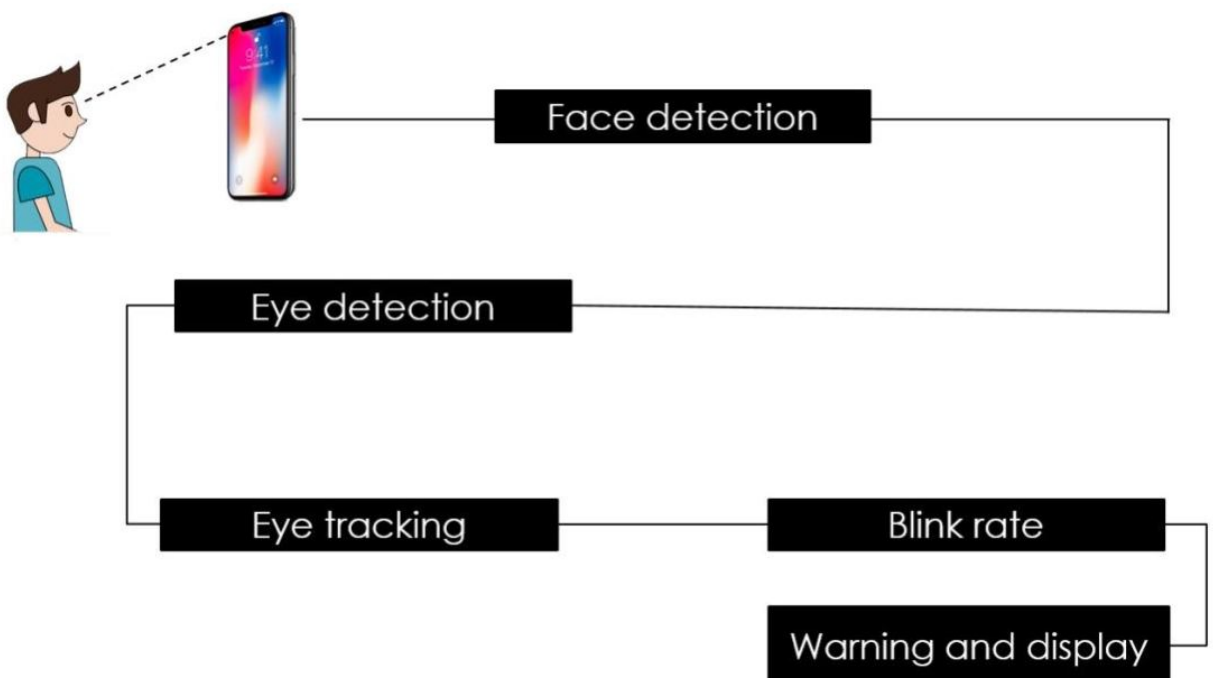


Figure 1.3: Driving Behaviour Analysis

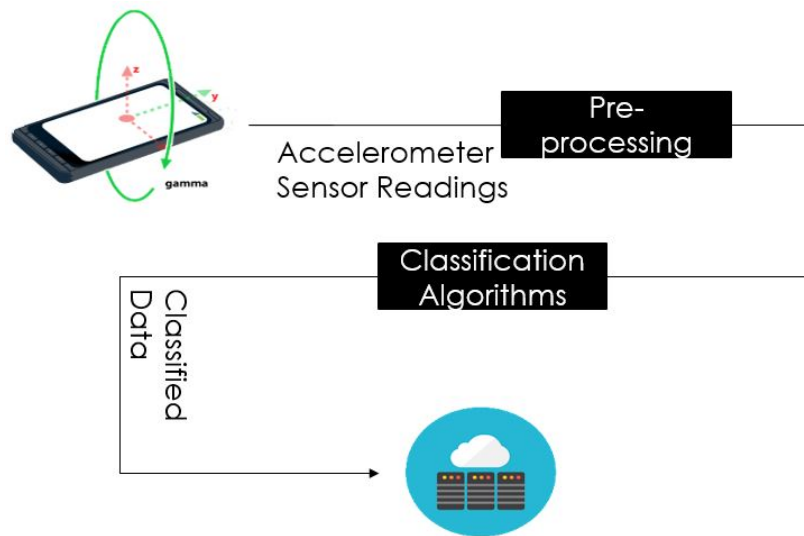
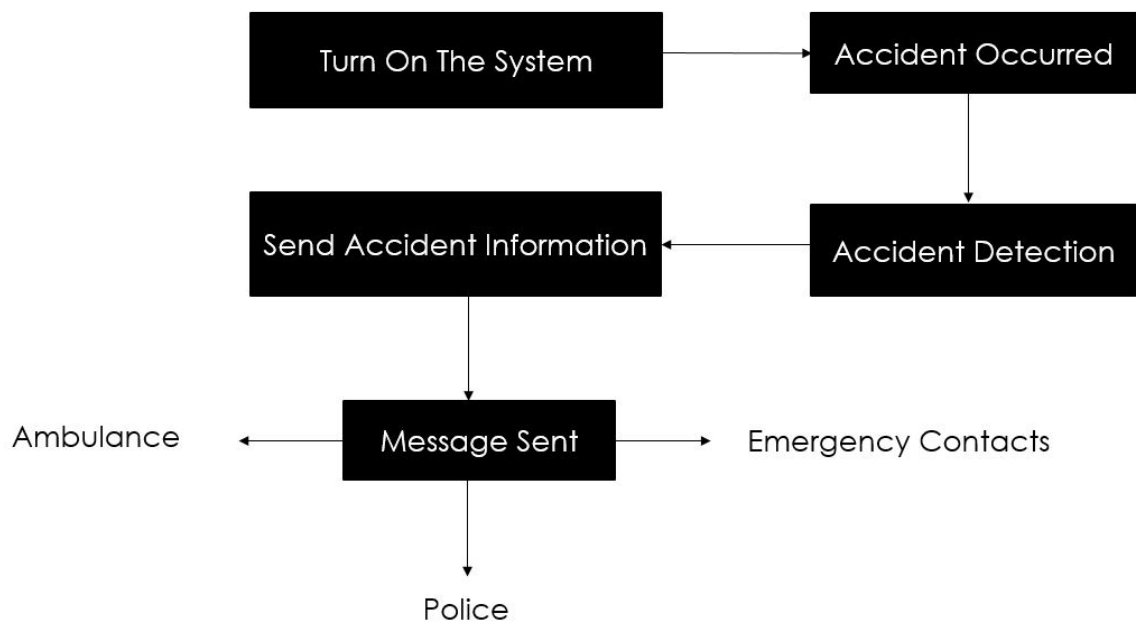


Figure 1.4: Accident Detection



## 1.2 Project Management and Deliverables

### 1.2.1 Tasks and Time Plan

Task	Deadline
PROPOSAL EVALUATION	1st Week in October 2019
SUBMITTING CONTRIBUTION OR SURVEY PAPER	First Semester
SRS EVALUATION	Second week of December 2019
SDD EVALUATION	Third week of February 2020
PROTOTYPE EVALUATION	3 days after Midterm exam
LECTURE WRITING FINAL THESIS	After Spring Break
DELIVERING 8 PAGES CONTRIBUTION PAPER	Second Semester
5- TECHNICAL EVALUATION	1st week of May 2020
LECTURE WRITING CV	Beginning of May 2020
FINAL THESIS	Last 10 days in June 2020
CERMONEY (OOA)	24 June 2020

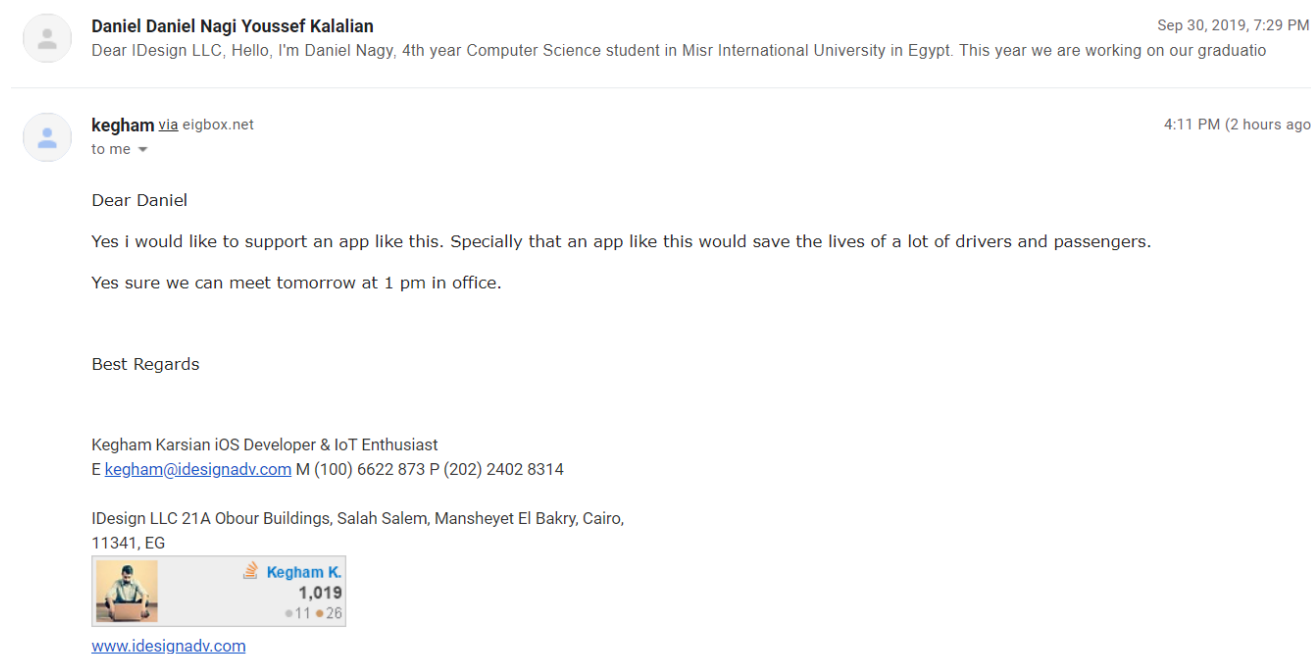
### 1.2.2 Budget and Resource Costs

1. Mobile Device has a built in :

- Accelerometer
- GPS
- GSM
- Front Camera

### 1.2.3 Supportive Documents

Figure 1.5: Supportive Documents



## Chapter 2

# Literature Work

### 2.1 Similar System Information

This paper [1], shows that a great deal of accidents happen in view of the terrible physical condition of the driver, innovation could defeat this mind boggling issue or at least reduce it. The proposed system is an eye blinking detector which can cautiously screen the physical express, the centralization of the driver and caution if necessary. Truly concerning how the system functions; above all else, a camcorder to set edges, do picture handling and neural systems to break down the driver conduct. Also, a productive calculation to experimentally contemplate the eye highlights. At long last, neural system calculation to see a pictures of the eyes in various states.

This paper [2], explains that bicycle accidents are more dangerous than the other sort of vehicles. Since it has less security systems than the four-wheel vehicles. The proposed system is an IOT smart system that could do recognition and counteraction of a potential accident. The system comprises of a Raspberry-Pi containing a few sensors and a camera. Firstly, vibration sensor and accelerometer to watch the driving behaviour. Secondly, a GPS module to locate the exact place of the accident if it occurs and the closest medical clinic. Thirdly, a GSM module to advise the clinic that an accident happened. Fourthly, an eye blinking detector to prevent drowsy of the driver. Likewise, check if the driver is wearing a head protector or not. At long last, liquor sensor to quantify the liquor level of the driver's body.

This paper [3], exhibits that the time between an accident and the clinical help is basic and could spare lives. The proposed system is an application that contains accelerometer and heartbeat sensor. The application can choose the most noticeably terrible physical express that could happen to the driver by the accelerometer (perceive the inclining of the vehicle). The heartbeat sensor checks the heartbeat of the driver, and tell the application if there is something incorrectly. At the point when the application discovers that the driver is in a perilous circumstance, instant message is sent to the driver's companions and the closest emergency clinic also.

This paper [4], shows that vehicle vulnerable sides causes bunches of accidents. The vulnerable sides is the spots that the side mirrors couldn't show. The proposed venture is a vulnerable side accident anticipation framework (BSAPS) that can detect any vehicle in these spots. The most well-known vulnerable sides are situated in the back quarter and the territories confronting the two sides in the back of the vehicle. The framework comprises of front sensor, left and right sensors, LED meter and perceptible alert. The front sensor is useful if there is mist, recognizing close by vehicles or deterrents. Concerning left and right side sensors, to identify vehicles in the vulnerable side. Driven meter to caution the driver that there is a vehicle in the vulnerable side whether left or right. At long last, a discernible caution to alarm the driver there is a vehicle close.

This paper [5] , Shows that traffic dangers is one of the serious issues confronting the world. One of the significant causes is expanding in vehicles and increment of death in light of street mishaps. Thus there is a need of a framework to give better transportation and diminish the proportion of street mishaps and spare life's kin. One of the arrangements that is proposed by this paper is utilizing IR sensors and Arduino uno innovation by two stages accident recognition and avoidance. Recognition completed utilizing IR sensors that could recognize and caution individuals by sending sms by GSM and accident area utilizing GPS.

Second stage, Accident counteraction is done utilizing IR sensors by notice the driver about the neighboring vehicles when separation between them is beyond the threshold value.

The contents of this paper [6], proposed an examination that number of vehicles is expanding ordinarily because of changes in human life and everybody needs his own private vehicle as opposed to utilizing open transportation, so that may bring about numerous issues with respect to substantial traffic and hence the occurrence of numerous mishaps which are incredible misfortunes to the earth. Then again, there is no security for vehicles so they can be taken without any problem. The disclosure of mishaps and their distinguishing proof are the fundamental thoughts of the paper. The paper embraced two unique innovations, specifically installed and android, inserted is utilized to identify the mishap utilizing accelerometer sensor and android innovation is utilized to get the vehicle area utilizing the GPS. The outcomes are that clients would have the option to know their vehicle areas and identify their movement. The accelerometer is utilized to identify mishaps and its examples, in the interim, the GPS is utilized to decide area of the vehicle.

In this paper [7], the framework proposed identifies accidents and alarms. Vehicle mishaps are the foremost string to individuals' lives which causes passing, numerous organizations gained a ton of ground to take care of this issue however couldn't diminish mishaps rates. The principle reason of expanding in mishap numbers are speed, outer weight and change in tilt point. The proposed android application tackles this issue by investigating the mishap occasions and sends crisis alarms to closest police headquarters and human services place, this application is coordinated with an outer compel sensor to remove the outward power on the vehicle body. It estimates speed and change of tilt edge with GPS and accelerometer sensors separately on android telephone.

The contents if this paper [8], demonstrate that an enormous level of individuals bite the

dust from mishap wounds. A successful methodology for lessening traffic fatalities is: first structure programmed car crash location frameworks, second, decreasing the time between when a mishap happens and first crisis reaction vehicle is dispatched to the area of the mishap. These days, the capacity to identify mishaps is simple by cell phones, the majority of the cell phones put together mishap location frameworks depend with respect to the speed readings of the vehicle (separated from the cell phone GPS recipient) and the G-force value (removed from cell phone accelerometer sensor) to distinguish mishaps, numerous references guarantee that 90 percent of the street auto collisions happen at low speed accordingly, this paper focuses on low speed mishaps. The framework was for all intents and purposes tried in genuine recreated condition and accomplished great execution results by utilizing the GPS collector and the accelerometer sensor.

This paper [9], Finds that mishaps rate is expanded by 54 percent contrasted with earlier decade. This framework limits activity time after a mishap and location of it. The paper present an effective security framework for the moving vehicles utilizing SMS ready framework. The framework utilizes the microcontroller which makes it interesting near to different frameworks. This parts used to distinguish the mishaps , sparing telephone numbers , sending SMS. The significant segment is the microcontroller which play out all the activities identified with control the implanted framework circuit, The discovery of the mishap done by vibration sensor and this identification is sent by SMS caution to versatile utilizing GSM and the installed framework is fit inside the vehicle.

Driving behaviour plays huge role in accident occurrence, most of the mishaps are brought about by the driver not the nature. This report[10] says that driving style is a variable in two significant things, well being and mileage. This paper put each driving style in a group; by utilizing GPS sensors and a few distinctive investigation roads (Hidden Markov Models, Behavioral theme extraction and DP-implies). Each bunch (driving conduct) is contrasted with an all around characterized ground truth driving conduct (Normal driving conduct).



### 2.1.1 Similar System Description

### 2.1.2 Comparison with Proposed Project

	Intelligent accident detection and alert system for emergency medical assistance	Design and implementation of an eye blinking detector system for automobile accident prevention	Comparison of Different Driving Style Analysis Approaches based on Trip Segmentation over GPS Information	An IOT based smart system for accident prevention and detection	Our Project
<b>Main Idea</b>	Minimize emergency response time	Prevent drowsiness while driving	Analyze driving behaviour	Prevent accidents using blinking detection, alcohol sensor, detect accidents using accelerometer and notify nearby authorities using GPS and GSM.	Track the driving behaviour and the concentration level of the driver and thus take measures to counter any deficiencies presented by the driver.
<b>Algorithm</b>	Flowchart mentioned only	Haar cascade classifier algorithm	-DP means -Hidden Markov Models -Behavioural topic extraction	Not Mentioned	-
<b>Method</b>	Mobile Application	Microcontroller	Sensor Readings	Sensor Readings	Mobile Application
<b>Readings</b>	-GPS -GSM -Accelerometer -Heartbeat Sensor	-Infrared Camera -Alarm	-GPS	-Accelerometer -Vibration Sensor -GPS -GSM -Alcohol Sensor -Camera	-Accelerometer -GPS -GSM -Front Camera
<b>Features</b>	-Monitoring the tilt of the car and heartbeat sensor -Sends a message to registered people if an accident occurred	-Eye blinking detection	-Clustering driving behaviour	-Eye blinking detector -Locate accident site -Sends a message to registered people if an accident occurred -Measuring driver's alcohol percentage	-Driver behaviour analysis -Clustering between nearby cars -Eye blinking detector -Locate accident site -Sends a message to registered people if an accident occurred

## Chapter 3

# System Requirements Specifications

### 3.1 Introduction

#### 3.1.1 Purpose of this document

The Purpose of this Software Requirement Specification is to identify the requirements needed to run the Parental Driving Safety System which is a system that monitors the driving behavior of siblings and alert parents if any intervention is needed. This is done with the aid of built-in smart phone sensors such as accelerometer , gyroscope and camera.

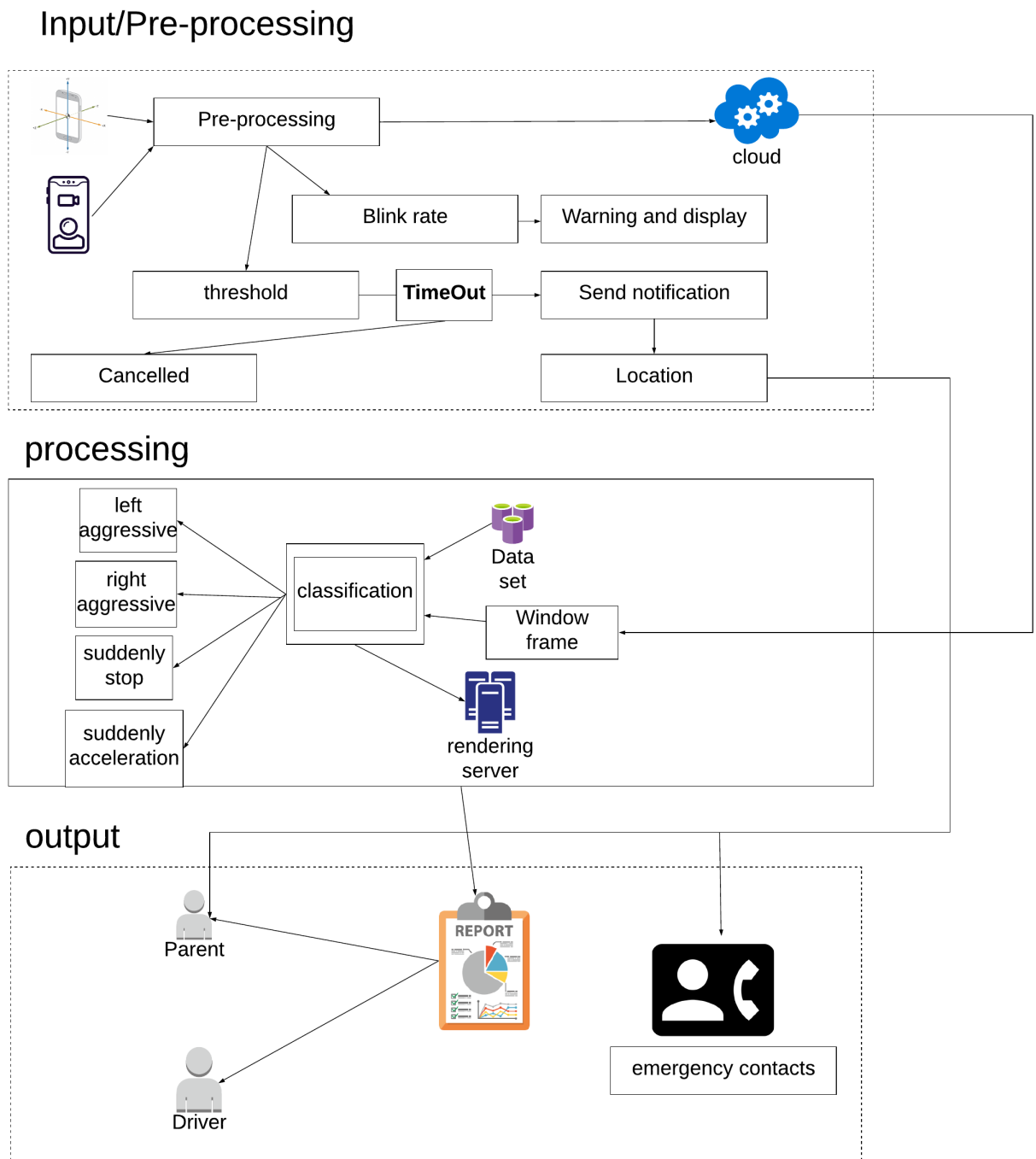
#### 3.1.2 Scope of this document

This Software Requirements Specification (SRS) is the requirements work product that formally specifies Parental Driving Safety System , also illustrating the targeted user such as concerned parents that have siblings that drive. This system also works for people who have private drivers or car rental companies and with the addition of more features it can be used for other numerous applications .

#### 3.1.3 Overview

Real time Monitoring and analyzing of incorrect behaviors of the driver to rate the driver so parents can monitor their children and can track their children if an accident occurred. The system should always aim to provide drivers with a trustworthy feedback by

Figure 3.1: Overviews



drowsiness alert, which also helps the parents to keep track of their children. This document proposes Parental Driving Safety System which is a system that performs detection of an accident, identification of a drivers incorrect behaviour and drowsiness alert using sensors found in smartphones. Parental Driving Safety will consist of a mobile application that will handle the real time data that will be collected from sensors(accelerometer), this data will be analyzed to create automatically generated analyzes that would be helpful to parents to monitor each their children.

Figure 3.2: Blinking Detection

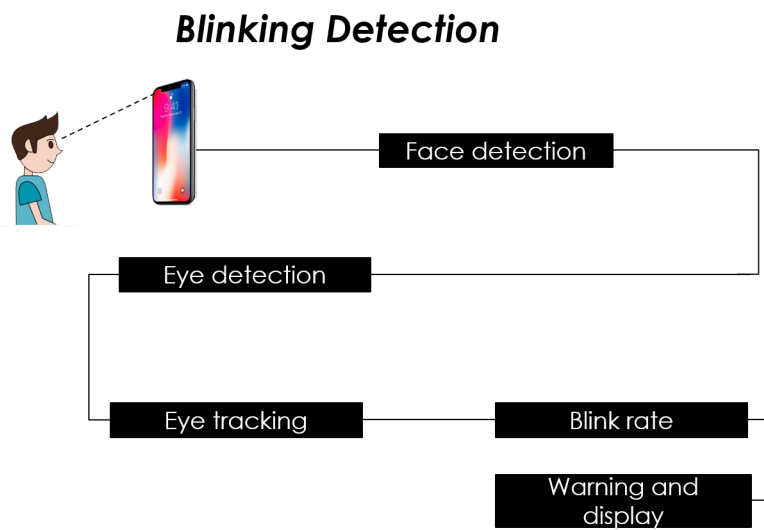
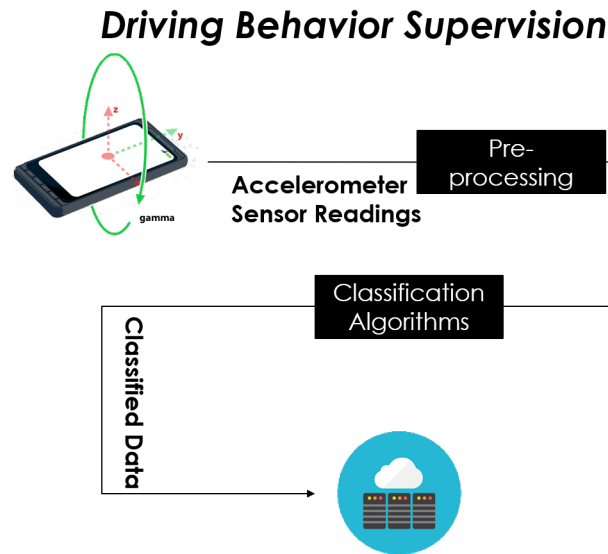
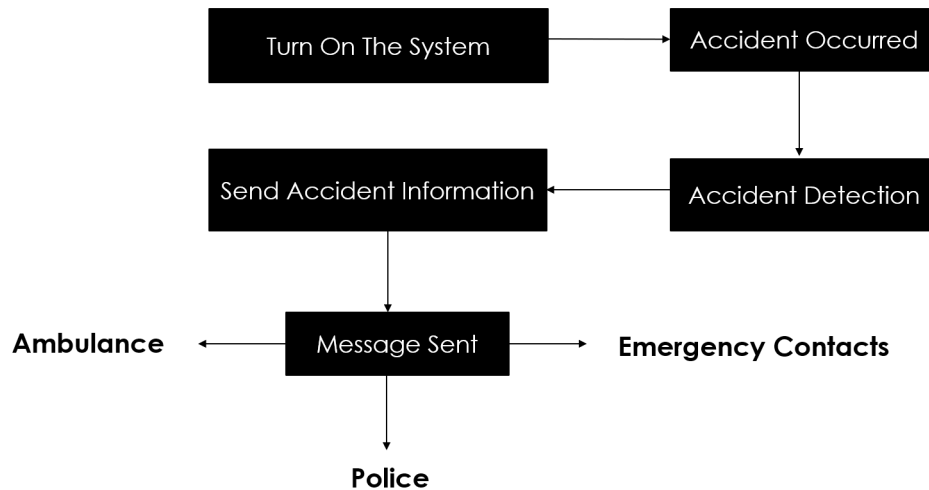


Figure 3.3: Driving Behavior supervision



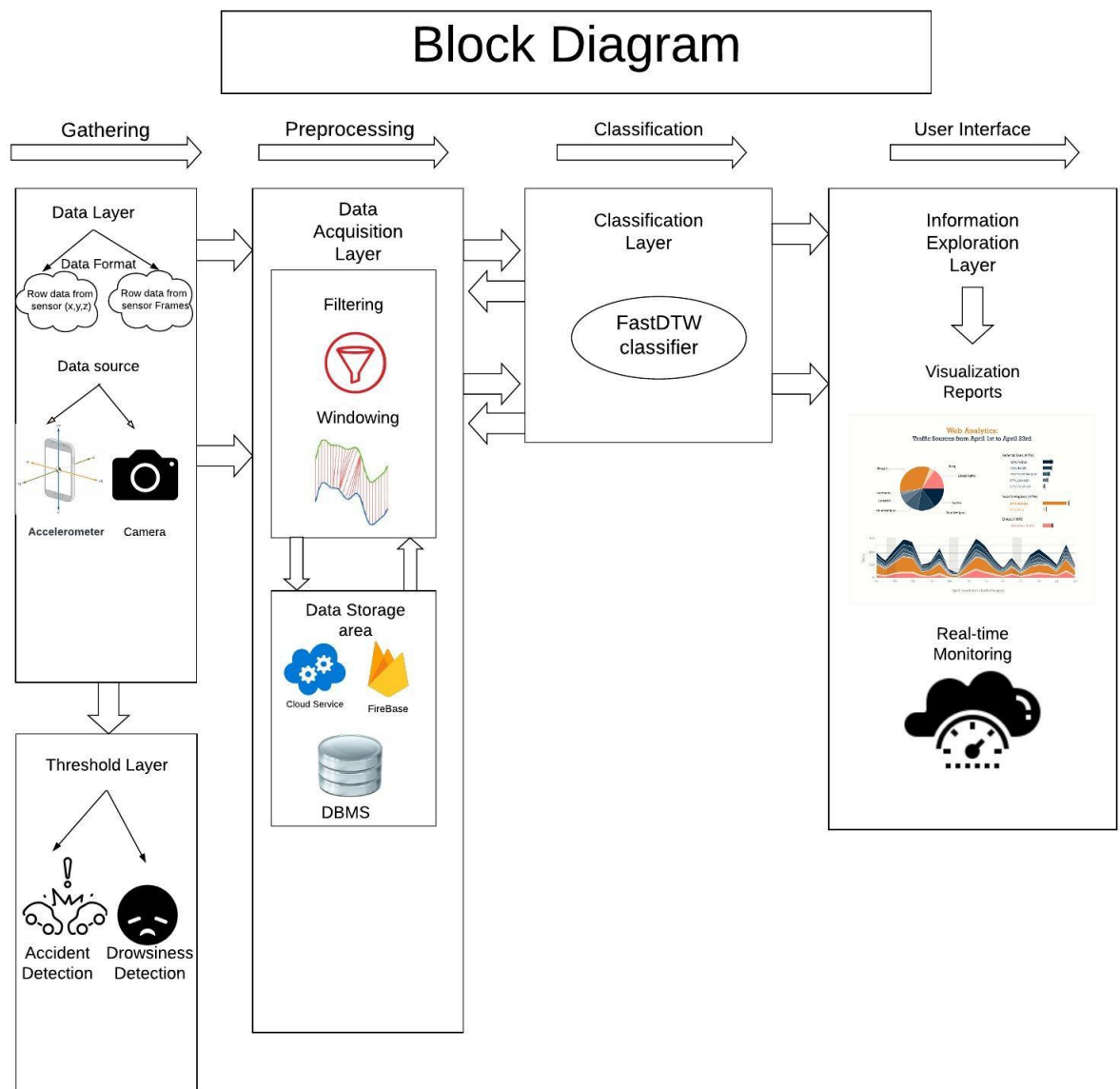
The proposed system a Parental Driving Safety System that uses sensors accelerometer to collect readings of the driving behavior. Therefore, the collected readings is passed through a pre-processing phase that supposedly to get the data formatted. Then, the data is passed to the cloud data storage, which takes the required data to the server side to get the required analysis using classifier Fast DTW algorithm. After analyzing the data the incorrect behavior is obtained. The rating data is always sent to the parent, alert the driver when the driver's eye closed for more than 3 seconds and notify the parent if an accident occurred with the location of the accident and nearby hospitals. Also, the parent can monitor the ratings of the driver by monitor the reports.

Figure 3.4: Accident Detection Medical Support

***Accident Detection For A Quick Medical Support*****3.1.4 Business Context**

The aim of this application is to better the safety measures taken by parents and emergency services to save teenage and young adult lives through regular monitoring, rating and finally communication in case of emergencies. Companies in general would make use of such an application due to its variety of functionalities specifically its design which would benefit by increasing the application scale through uploading the application on various app stores such that it would be available for consumers to use.

Figure 3.5: Block Diagram



## 3.2 General Description

### 3.2.1 Product Functions

#### 3.2.1.1 Module 1: Driving Behavior

1. Reading accelerometer from a smart phone.
2. Detection of the driver behavior (Left/Right Aggressive, Sudden Stop/Acceleration).

3. Rate the driver according to his/her behaviors.
4. Make report and send it to the driver parents on cloud (driving behaviors and rating).

#### **3.2.1.2 Module 2: Accident Detection**

1. Reading accelerometer from a smart phone.
2. Accident Detection.
3. Send SMS and location of the accident and nearby hospitals to the emergency contacts.
4. There is a cancel option.

#### **3.2.1.3 Module 3: Drowsiness Detection**

1. Blinking detection.
2. Blink more than 3 seconds the system will alert the driver.

### **3.2.2 Similar System Information**

Paper (1): This paper considers the issue of describing the manner in which individuals drive and utilize driver help frameworks and coordinated well-being frameworks without utilizing direct driver signals. Account signs can be procured by a GPS information logging framework: position, velocity, accelerations and guiding point. The classifier proposed, presents the structure of an intelligent driver practices model dependent on neural systems and utilizes as data sources statistical transformations of the driving determination time signals: directing profiles, pedals utilized, speeding and escaping the path and road. Driver ID for security frameworks and to classify driver into one of two classifications, forceful and moderate. The proposed approach has been actualized in genuine condition and its presentation tried in reproduction runs.



Figure 3.6: Similar System

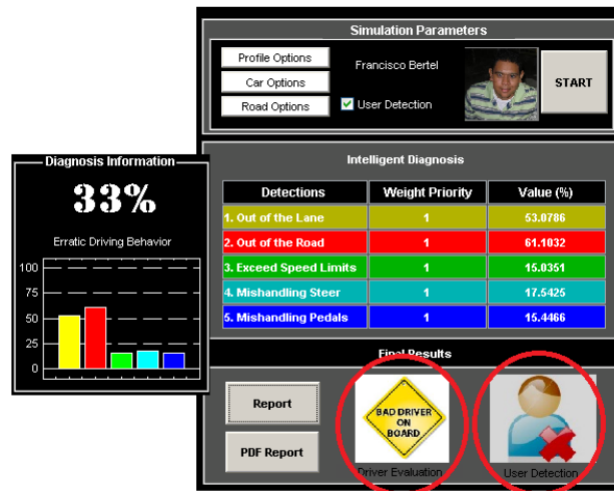


Fig. 12. Negative user identification.

Paper(2): Driving investigation is an ongoing subject of enthusiasm because of the developing security concerns in vehicles. In any case, the absence of freely accessible driving information right now constrains the progress on this field. AI procedures could exceptionally upgrade research, yet they rely on a lot of information which are troublesome and exorbitant to get through Naturalistic Driving Studies, bringing about restricted availability to the general exploration community. Additionally, the expansion of cell phones has given modest and simple to-convey plat-structure for driver conduct detecting, yet existing applications don't give open access to their information. Consequently, this paper presents the UAH-DriveSet, an open dataset that allows profound driving examination by giving a lot of information caught by our driving observing application DriveSafe. The application is controlled by 6 unique drivers and vehicles, performing 3 distinct practices (ordinary, lazy and forceful) on two kinds of roads (motorway and optional street), bringing about over 500 minutes of naturalistic driving with its related crude information and prepared semantic data.

### 3.2.3 User Characteristics

Driver: Must have basic knowledge in using android mobile devices and mobile internet.

### 3.2.4 User Problem Statement

Accurate detection of the driver sudden moves and changes in the car speed or direction that take places which can have fatal consequences as well as the presence of road obstacles that forces drivers to make sudden movements in an attempt to avoid them causing accidents. Another problem that causes a huge number of accidents is drowsiness, more importantly, when an accident happens the driver is in no condition to send any sort of notification to the emergence response teams and thus is under the mercy of the people around them if any to call the authorities as soon as possible.

### 3.2.5 User Objectives

The main objectives of this project is the accurate detection and identify driving behaviour, quick detection of drowsiness and alert the driver, notify emergency contacts in case of accident occurrence and notify near drivers in case of road obstacles to prevent accidents.

### 3.2.6 General Constraints

The application generally doesn't have constraints but existing once are crucial . One of the constraints that faces this application is internet connection, since the application depends on the server the faster the internet connection the less loss of data in case of an accident. Another constraint the phone must be charged and functional in order to perform its functions. Furthermore, the mobile phone position is crucial as it defines the efficiency of the application as it must be positioned such that the eyes of the driver are clearly visible.

## 3.3 Functional Requirements

### 3.3.1 User

#### 3.3.1.1 FR1

<b>FID</b>	FR1.
<b>Name</b>	Login.
<b>Input</b>	Username,password
<b>Output</b>	Login successful / Login failed
<b>Action</b>	Takes Username and Password, checks if account exists
<b>Pre-condition</b>	Registered.
<b>Post-condition</b>	Welcome Page.
<b>Dependencies</b>	FR4.

#### 3.3.1.2 FR2

<b>FID</b>	FR2.
<b>Name</b>	Logout.
<b>Input</b>	None.
<b>Output</b>	Logout successful.
<b>Action</b>	Logout from the application.
<b>Pre-condition</b>	The account is logged in and running.
<b>Post-condition</b>	Logout successful.
<b>Dependencies</b>	FR1.

#### 3.3.1.3 FR3

<b>FID</b>	FR3.
<b>Name</b>	edit_Profile.
<b>Input</b>	Edit(name,Email,phone).
<b>Output</b>	Edit successful.
<b>Action</b>	The user edits all his attributes.
<b>Pre-condition</b>	Account existence.
<b>Post-condition</b>	Data modified into database.
<b>Dependencies</b>	FR4.

#### 3.3.1.4 FR4

<b>FID</b>	FR4.
<b>Name</b>	register.
<b>Input</b>	Id,name,Email,phone.
<b>Output</b>	Registered / Something went wrong.
<b>Action</b>	Fill all attributes of the user.
<b>Pre-condition</b>	None.
<b>Post-condition</b>	Welcome page.
<b>Dependencies</b>	None.

### 3.3.2 Accelerometer Readings

#### 3.3.2.1 FR5

<b>FID</b>	FR5.
<b>Name</b>	Read_Accelerometer.
<b>Input</b>	Accelerometer readings.
<b>Output</b>	Readings fetched.
<b>Action</b>	Get accelerometer readings.
<b>Pre-condition</b>	Accelerometer existence.
<b>Post-condition</b>	Data read successfully.
<b>Dependencies</b>	FR1.

### 3.3.3 Driver

#### 3.3.3.1 FR6

<b>FID</b>	FR6.
<b>Name</b>	startTrip.
<b>Input</b>	Button.
<b>Output</b>	Trip started.
<b>Action</b>	The driver clicks on a button to start the trip.
<b>Pre-condition</b>	Trip hadn't been started.
<b>Post-condition</b>	Trip attributes sent to database.
<b>Dependencies</b>	FR5.

**3.3.3.2 FR7**

<b>FID</b>	FR7.
<b>Name</b>	send_Emergency.
<b>Input</b>	File.
<b>Output</b>	Sending location and nearist hospitals to emergency contacts OR "Don't send" pop-up message.
<b>Action</b>	Send accident location and nearist hospitals to emergency contacts OR the driver cancel the process.
<b>Pre-condition</b>	Accident must be detected.
<b>Post-condition</b>	Send accident counter to the database.
<b>Dependencies</b>	FR12.

**3.3.3.3 FR8**

<b>FID</b>	FR8.
<b>Name</b>	alert_Driver.
<b>Input</b>	File.
<b>Output</b>	Alarm.
<b>Action</b>	Alarm is fired when the driver is sleepy.
<b>Pre-condition</b>	Driver slept.
<b>Post-condition</b>	None.
<b>Dependencies</b>	FR14.

**3.3.3.4 FR9**

<b>FID</b>	FR9.
<b>Name</b>	view_Driver_Report.
<b>Input</b>	Driver id.
<b>Output</b>	Report (Driver's accumilative behaviour rating + Driver's info).
<b>Action</b>	Get the Rating of the driver's driving behaviour.
<b>Pre-condition</b>	Rating must be calculated.
<b>Post-condition</b>	None.
<b>Dependencies</b>	FR16.

### 3.3.4 Analysis

#### 3.3.4.1 FR10

<b>FID</b>	FR10.
<b>Name</b>	Filter.
<b>Input</b>	Accelerometer and Gyroscope (x,y,z) and Timestamp.
<b>Output</b>	Readings filtered.
<b>Action</b>	Removing noise from accelerometer data.
<b>Pre-condition</b>	Check for duplicate readings with the same timestamp.
<b>Post-condition</b>	Send readings to the database.
<b>Dependencies</b>	FR6.

#### 3.3.4.2 FR11

<b>FID</b>	FR11.
<b>Name</b>	detect_Behaviour.
<b>Input</b>	Array of filtered Accelerometer , Gyroscope (x,y,z) and Timestamp.
<b>Output</b>	Daily report.
<b>Action</b>	Detect abnormal driving behaviour.
<b>Pre-condition</b>	Sensor readings are filtered.
<b>Post-condition</b>	Send behaviour details to the database.
<b>Dependencies</b>	FR10.

#### 3.3.4.3 FR12

<b>FID</b>	FR12.
<b>Name</b>	detect_Accident.
<b>Input</b>	Array of filtered Accelerometer , Gyroscope (x,y,z) ,Timestamp and Threshold.
<b>Output</b>	Sending accident location and nearest hospitals to emergency contacts countdown OR "don't send"pop-up message.
<b>Action</b>	Detect accidents using Threshold.
<b>Pre-condition</b>	Sensor readings are filtered.
<b>Post-condition</b>	Send the location and nearest hospitals to emergency contacts.
<b>Dependencies</b>	FR10.

## 3.3.4.4 FR13

<b>FID</b>	FR13.
<b>Name</b>	face_Detection..
<b>Input</b>	Camera.
<b>Output</b>	None.
<b>Action</b>	Detects face.
<b>Pre-condition</b>	Camera existence.
<b>Post-condition</b>	Check for drowsiness.
<b>Dependencies</b>	FR1.

## 3.3.4.5 FR14

<b>FID</b>	FR14.
<b>Name</b>	check_Drowsiness.
<b>Input</b>	Camera.
<b>Output</b>	Alarm if the driver was sleepy.
<b>Action</b>	Detect drowsiness.
<b>Pre-condition</b>	Camera existence.
<b>Post-condition</b>	Send drowsiness counter to database.
<b>Dependencies</b>	FR13.

## 3.3.5 Report

## 3.3.5.1 FR15

<b>FID</b>	FR15.
<b>Name</b>	get_Behaviour.
<b>Input</b>	Abnormal behaviour counters.
<b>Output</b>	Report.
<b>Action</b>	Get the abnormal behaviour counters, the driver accumulative driving rating and outputs the report.
<b>Pre-condition</b>	Counters must be set.
<b>Post-condition</b>	Send the data to database.
<b>Dependencies</b>	FR11,16.

### 3.3.5.2 FR16

<b>FID</b>	FR16.
<b>Name</b>	find_Rating.
<b>Input</b>	Abnormal behaviour counters.
<b>Output</b>	Accumilative rating of the driver.
<b>Action</b>	Get all the counters daily and calculate the accumilative rating.
<b>Pre-condition</b>	Counters must be set.
<b>Post-condition</b>	Put the rating in the report and send it to database.
<b>Dependencies</b>	FR11.

### 3.3.6 Parent

#### 3.3.6.1 FR17

<b>FID</b>	FR17.
<b>Name</b>	view_Report.
<b>Input</b>	Driver id.
<b>Output</b>	Report.
<b>Action</b>	Show a daily report of the parent's son/daughter driving behaviour.
<b>Pre-condition</b>	All report attributes must be set.
<b>Post-condition</b>	None.
<b>Dependencies</b>	FR15.

### 3.3.7 Administrator

#### 3.3.7.1 FR18

<b>FID</b>	FR18.
<b>Name</b>	view_user.
<b>Input</b>	User id.
<b>Output</b>	Selected records from database based on ID.
<b>Action</b>	View user from database.
<b>Pre-condition</b>	User must be existed.
<b>Post-condition</b>	None.
<b>Dependencies</b>	None.



**3.3.7.2 FR19**

<b>FID</b>	FR19.
<b>Name</b>	remove_User.
<b>Input</b>	User id.
<b>Output</b>	Selected records from database based on ID.
<b>Action</b>	Delete the selected records from the database.
<b>Pre-condition</b>	User must be existed.
<b>Post-condition</b>	None.
<b>Dependencies</b>	None.

## 3.4 Interface Requirements

### 3.4.1 User Interfaces

#### 3.4.1.1 GUI

Figure 3.7: Log in GUI

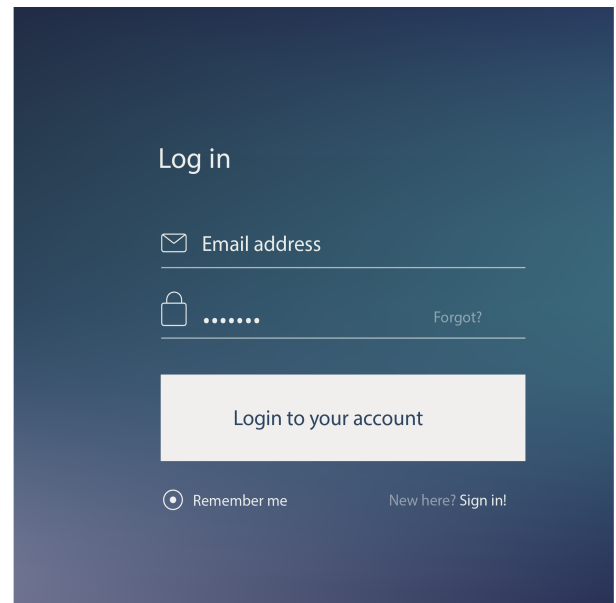


Figure 3.8: SignUpAS GUI

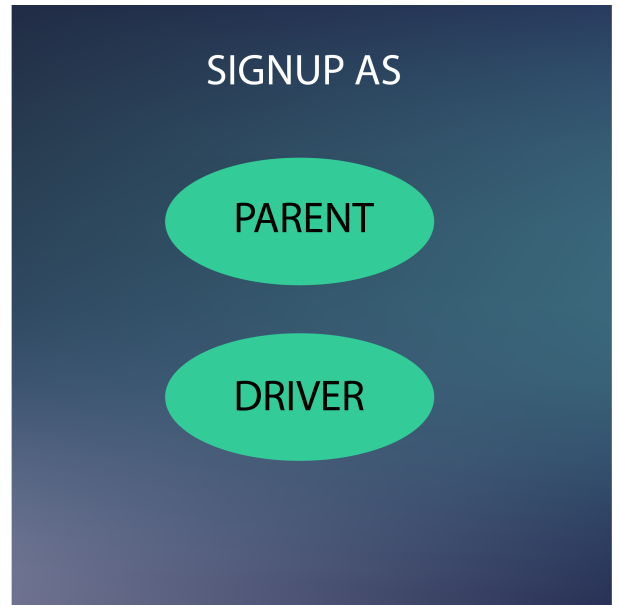
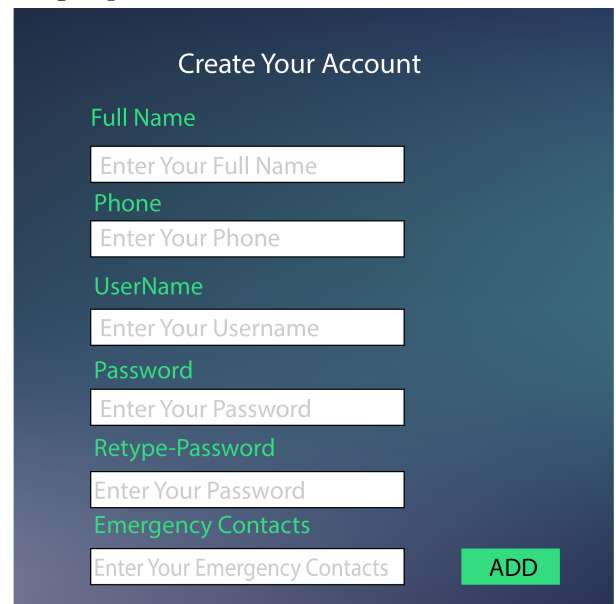


Figure 3.9: Driver SignUp GUI



The image shows a 'Create Your Account' form with a dark blue background. The form contains several input fields with placeholder text, each preceded by a green label. The labels are 'Full Name', 'Phone', 'UserName', 'Password', 'Retype-Password', and 'Emergency Contacts'. The input fields are white with rounded corners. At the bottom right of the form is a green button with the text 'ADD' in white.

Create Your Account

Full Name  
Enter Your Full Name

Phone  
Enter Your Phone

UserName  
Enter Your Username

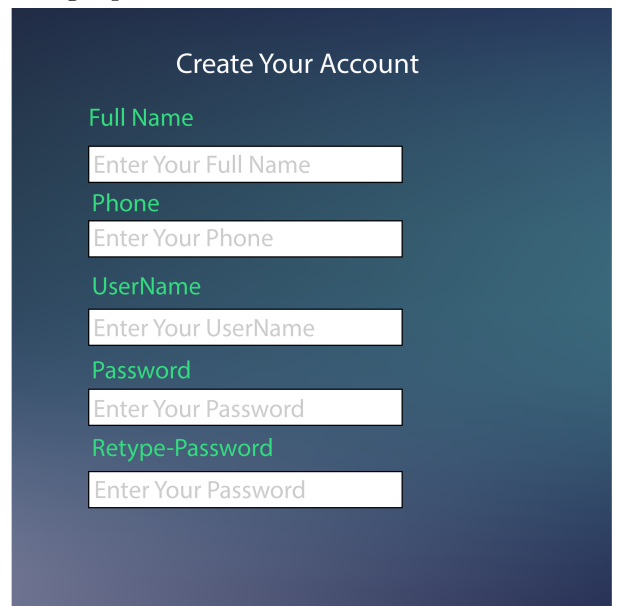
Password  
Enter Your Password

Retype-Password  
Enter Your Password

Emergency Contacts  
Enter Your Emergency Contacts

ADD

Figure 3.10: Parent Signup GUI



The image shows a 'Create Your Account' form with a dark blue gradient background. The form contains six input fields, each with a green label above it. The labels are 'Full Name', 'Phone', 'UserName', 'Password', and 'Retype-Password'. The input fields are white with black text. The first four fields have placeholder text: 'Enter Your Full Name', 'Enter Your Phone', 'Enter Your UserName', and 'Enter Your Password'. The fifth field, labeled 'Retype-Password', also has the placeholder text 'Enter Your Password'.

Create Your Account

Full Name  
Enter Your Full Name

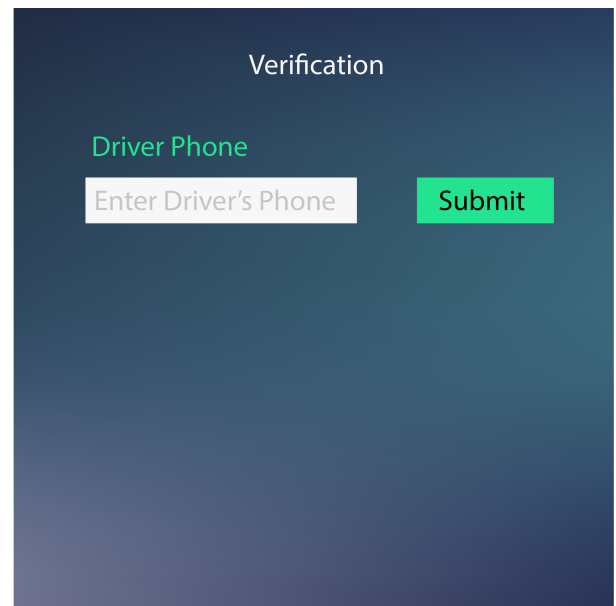
Phone  
Enter Your Phone

UserName  
Enter Your UserName

Password  
Enter Your Password

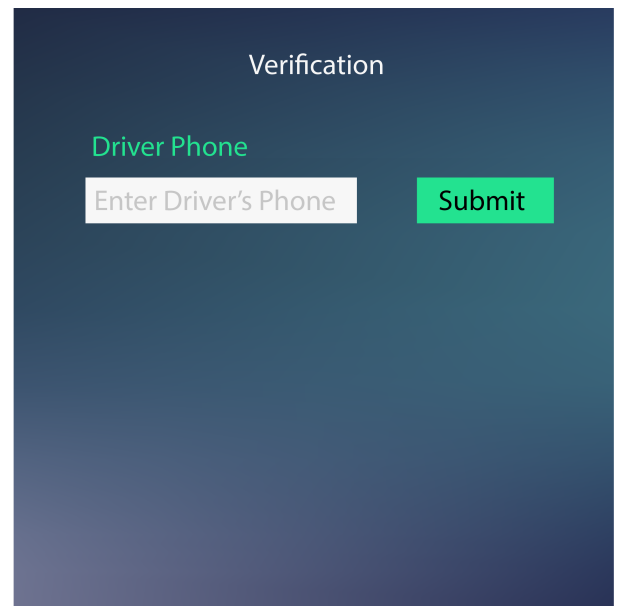
Retype-Password  
Enter Your Password

Figure 3.11: Parent Verification GUI



The image shows a mobile application interface for parent verification. It has a dark blue gradient background. At the top, the word "Verification" is written in white. Below it, the text "Driver Phone" is displayed in a green color. Underneath "Driver Phone" is a white rectangular input field containing the placeholder text "Enter Driver's Phone". To the right of this input field is a green rectangular button with the word "Submit" in white text.

Figure 3.12: Parent Verification GUI



The image shows a mobile application interface for parent verification. It has a dark blue gradient background. At the top, the word "Verification" is written in white. Below it, the text "Driver Phone" is displayed in a light green color. Underneath "Driver Phone" is a white rectangular input field containing the placeholder text "Enter Driver's Phone". To the right of this input field is a bright green rectangular button with the word "Submit" in white text.

Figure 3.13: Driver Verification GUI

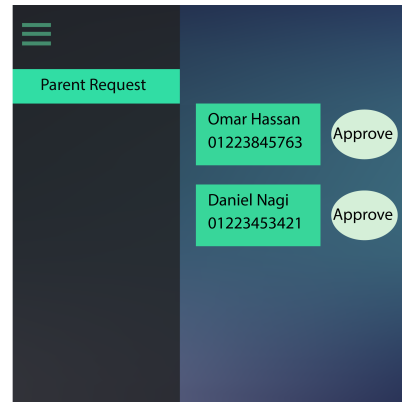
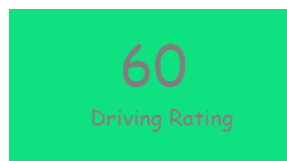
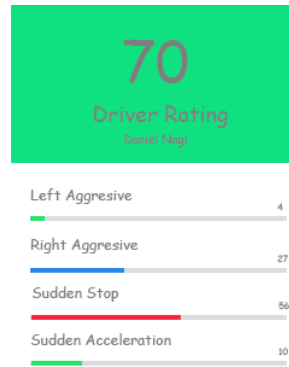




Figure 3.14: Start Driving GUI



**Name**  
Daniel Nagi  
**Mobile No**  
01117295455  
**Parent Name**  
Nagi Youssef  
**Mobile No**  
01227296421



### 3.4.2 Communications Interfaces

The communication interface is one of the most important requirements of our software as it will need a connection to the internet.

## 3.5 Performance Requirements

The application will be fast as we know that the application will send the driving behavior analysis and rating to the cloud to make a report that is sent to the concerned person daily.

## 3.6 Design Constraints

Any smart mobile device that include the android operating system and must have the connection with the internet to deal with the real-time data transfer.

### 3.6.1 Hardware Limitations

This system works on smart phones only and the minimum requirements is 1GB ram ,5 mega pixel.

## **3.7 Other non-functional attributes**

### **3.7.1 Performance And Speed**

Our application deals with critical situations that could save lives. Classify readings in real time to well-known readings requires much speed. Consuming less power as much as possible is the main challenge to let the application efficient.

### **3.7.2 Reliability**

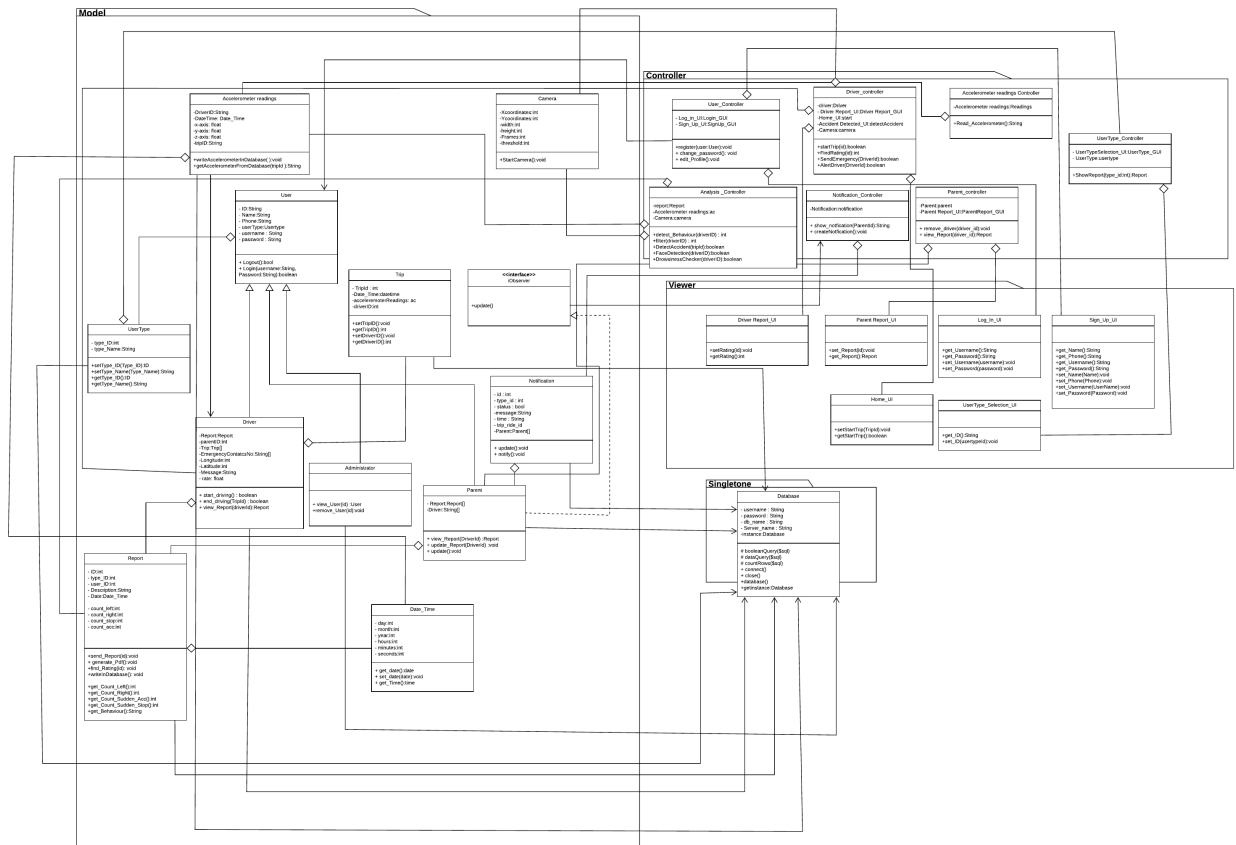
This application aims to be reliable, by checking the accelerometer readings is accurate or not. Also, avoiding false alarm(detection).

### **3.7.3 Usability**

The application will not be complex for the user, because the application mainly does calculations and readings in the backend. So, a simplified GUI that will be friendly with the user.

## 3.8 Preliminary Object-Oriented Domain

Figure 3.17: Preliminary Object-Oriented Domain



### 3.8.1 Class descriptions

#### 3.8.1.1 User

Abstract class

List of Super classes: None.

List of Sub classes: Administrator, Driver and parent

Purpose: Main class has user's information.

Collaborations: This class aggregate with UserType and inherited by Parent , Driver , Administrator and associates with User Controller

Attributes:

- ID:String
- Name:String

- Username:String
- Phone:String
- userType:UserType

Operations:

- + Logout():bool
- + Login(username:String, Password:String):boolean

### 3.8.1.2 UserType

List of Super classes: None.

List of Sub classes: none

Purpose: To differentiate between usertypes.

Collaborations: This class aggregate with User, User Type Controller and associates with Database.

Attributes:

- typeId:int
- typeName:String

Operations:

- +setTypeID(TypeID):ID
- +setTypeName(TypeName):String
- +getTypeID():ID
- +getTypeName():String

### 3.8.1.3 Parent

List of Super classes: User . iobserver

List of Sub classes: None.

Purpose: Every Parent wants to know the driving behavior of his children.

Collaborations: This class aggregate with Notification , Report and parent controller.Inherits from User and iobserver.Association with Parent Controller. Attributes:

- Report:Report[]

-Driver:String[]

Operations:

+ viewReport(DriverId) :Report

+ updateReport(DriverId) :void

+ update():void

#### 3.8.1.4 Administrator

List of Super classes: User.

List of Sub classes: None

Purpose: To edit anything in any user type.

Collaborations: This class association with database and inherits from User

Attributes:

None

Operations:

+ viewUser(id) :User

+removeUser(id):void

#### 3.8.1.5 Driver

List of Super classes: User.

List of Sub classes: None.

Purpose: Start trip and has his own report.

Collaborations: This class aggregates with Report,Driver Controller and Trip.Inherits from User. Association with accelerometer readings and database.

Attributes:

-Report:Report

- rate: float

-parentID:int

-Trip:Trip[]

-EmergencyContatcsNo:String[]

-Longitude:int

-Latitude:int

-Message:String

Operations:

+ startdriving() : boolean

+ enddriving(TripId) : boolean

+ viewReport(driverId):Report

### 3.8.1.6 Report

List of Super classes: Result,Date.

List of Sub classes: None

Purpose: to classify the driver behavior.

Collaborations: Associates with database.Aggregation with Parent, Parent Controller, Date<sub>TimeandDr</sub>

*Attributes :*

– *ID : int*

– *typeID : int*

– *userID : int*

– *Description : String*

– *Date : DateTime*

– *countleft : int*

– *countright : int*

– *countstop : int*

– *countacc : int*

Operations:

+sendReport(id):void

+ generatePdf():void

+findRating(id): void

+writeInDatabase(): void

+getCountLeft():int

+getCountRight():int

+getCountSuddenAcc():int

+getCountSuddenStop():int  
 +getBehaviour():String

### 3.8.1.7 Accelerometer Readings

List of super classes : None

List of Sub classes : Preprocessing

Purpose : Get readings from accelerometer sensor

Collaborations :Aggregates with DateTime, Analysis Controller and Accelerometer Readings Controller. Associates with Driver and Database.

Attributes :

-DriverID:String

-DateTime: *DateTime*

- *x - axis : float*

- *y - axis : float*

- *z - axis : float*

- *tripID : String*

Operations : None

+ *writeAccelerometerInDatabase() : void*

+ *getAccelerometerFromDatabase(tripId) : String*

### 3.8.1.8 Notification

List of super classes : iSubject

List of Sub classes : None

Purpose : pre-processing the data and send to the result class

Collaborations : This class Aggregates with Notification Controller and Parent.Associates with Database.

Attributes :

+ id : int

+ user :user

+ *type<sub>i</sub>d : int*



+ *status* : *bool*  
+ *time* : *String*  
+ *triprideid*  
*Operations* :  
– *update()*  
– *notify()*  
– *createNotification()*

### 3.8.1.9 Trip

List of super classes : None.

List of Sub classes : None

Purpose : Trip details.

Collaborations : Aggregates with driver.Associates with Database.

Attributes :

- TripId : int  
-DateTime:datetime  
-acceleremoterReadings: ac  
-driverID:int

Operations:

+setTripID():void  
+getTripID():int  
+setDriverID():void  
+getDriverID():int

### 3.8.1.10 Camera

List of super classes : None.

List of Sub classes : None

Purpose : Camera Initialization.

Collaborations : Aggregates with Driver Controller,Analysis Controller.

Attributes :

-Xcoordinates:int  
 -Ycoordinates:int  
 -width:int  
 -height:int  
 -Frames:int  
 -threshold:int  
 Operations:  
 +StartCamera():void

#### 3.8.1.11 Database

List of super classes : None

List of Sub classes : none

Purpose : system database

Collaborations : Associates with Trip,Notification,Parent,Usertype,Driver,Report,Administrator,Accelerometer, readings,

Attributes :

- username : String
- password : String
- dbname : String
- Servername : String

Operations:

booleanQuery(*sql*)dataQuery(*sql*)

countRows(*sql*)

– connect() – close()

#### 3.8.1.12 iObserver

List of Super classes: None.

List of Sub classes: Parent

Purpose: Update Notification.

Collaborations: Associates with Notification Controller. Parent Inherits from iobserver.

Attributes: None

Operations:

+update()

### 3.8.1.13 User Controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates User.

Collaborations: Associates with User and aggregates with SignUpUI,LoginUI.

Attributes:

- LoginUI:LoginGUI

- SignUpUI : SignUpGUI

Operations :

+ register(*user : User*) : void

+ changepassword() : void

+ editprofile() : void

### 3.8.1.14 Driver Controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates Driver.

Collaborations: Aggregates with Driver,DriverReportUI,CameraHomeUI and Accident DetectedUI.

Attributes:

-driver:Driver

- Driver ReportUI:Driver ReportGUI

-HomeUI:start

-Accident DetectedUI:detectAccident

-Camera:camera

Operations:

+startTrip(id):boolean  
+FindRating(id):int  
+SendEmergency(DriverId):boolean  
+AlertDriver(DriverId):boolean

#### 3.8.1.15 Accelerometer readings Controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates Accelerometer readings.

Collaborations: Aggregates with Accelerometer readings.

Attributes:

-Accelerometer readings:Readings

Operations:

+Read<sub>Accelerometer</sub>() : *String*

#### 3.8.1.16 UserType Controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates UserType.

Collaborations: Aggregates with UserTypeSelectionUI and UserType

Attributes:

- UserTypeSelectionUI:UserTypeGUI

- UserType:usertype

Operations:

+ShowReport(typeid:int):Report

#### 3.8.1.17 Analysis Controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates UserType.

Collaborations: Analysis *controller aggregates with Report, Camera and accelerometer Readings.*

*Attributes :*

- *report : Report*
- *Accelerometer readings : ac*
- *Camera : camera*

*Operations :*

- + *detectBehaviour(driverID) : int*
- + *filter(driverID) : int*
- + *DetectAccident(tripId) : boolean*
- + *FaceDetection(driverID) : boolean*
- + *DrowsinessChecker(driverID) : boolean*

### 3.8.1.18 Notification Controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates User.

Collaborations: Associates with iObserver and aggregates with Notification.

Attributes:

-Notification:notification

Operations:

- + *show\_notification(ParentId) : String*
- + *createNotification() : void*

### 3.8.1.19 Parent controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates Parent.

Collaborations: Aggregates with Parent ReportGUI,Parent.

Attributes:

-Parent ReportGUI:ParentReportGUI

```
+register(user:User):void
+ removedriver(driverid):void
+ viewReport(driverid):Report
```

### 3.9 Database Diagram

The diagram illustrates the relationships between various GP (Geopoint) entities. The entities are represented as boxes with their attributes. Relationships are shown as colored lines connecting attributes of different entities.

**Entities and their attributes:**

- gp\_notification**
  - notification\_ID : int(15)
  - type\_ID : int(15)
  - status : tinyint(1)
  - message : text
  - date\_ID : int(15)
  - trip\_ID : int(15)
  - parent\_ID : int(15)
- gp\_driver**
  - id : int(15)
  - report\_ID : int(15)
  - parent\_ID : int(15)
  - trip\_ID : int(15)
  - emergency\_Contacts : int(12)
  - longitude : int(10)
  - latitude : int(10)
  - Message : text
  - user\_ID : int(15)
- gp\_parent**
  - id : int(15)
  - report\_ID : int(15)
  - rate : float
  - driver\_ID : int(15)
  - user\_ID : int(15)
- gp\_report**
  - Report\_id : int(15)
  - type\_id : int(15)
  - user\_id : int(15)
  - description : text
  - date\_id : int(15)
  - count\_Left : int(10)
  - count\_Right : int(10)
  - count\_Sudden\_Acc : int(10)
  - count\_Sudden\_Stop : int(10)
- gp\_date**
  - id : int(15)
  - day : int(10)
  - month : int(10)
  - year : int(10)
  - hours : int(10)
  - minutes : int(10)
  - seconds : int(10)
- gp\_user**
  - id : int(15)
  - Name : varchar(30)
  - Phone : varchar(12)
  - Username : varchar(30)
  - Password : varchar(15)
  - user\_Type : int(11)
- gp\_administrator**
  - id : int(15)
  - user\_id : int(15)
- gp\_trip**
  - trip\_ID : int(15)
  - date\_ID : int(15)
  - accelerometer\_id : int(10)
  - driver\_id : int(15)
- gp\_usertype**
  - userType\_id : int(15)
  - type Name : varchar(30)
- gp\_accelerometer\_readings**
  - id : int(15)
  - driver\_ID : int(15)
  - date\_id : int(15)
  - x-axis : int(10)
  - y-axis : int(10)
  - z-axis : int(10)
  - trip\_ID : int(15)

**Relationships (Connections):**

- gp\_notification** to **gp\_driver**: notification\_ID to id, type\_ID to report\_ID, status to emergency\_Contacts, message to Message, date\_ID to date\_ID, trip\_ID to trip\_ID, parent\_ID to parent\_ID.
- gp\_driver** to **gp\_parent**: id to id, report\_ID to report\_ID, parent\_ID to parent\_ID, trip\_ID to trip\_ID, emergency\_Contacts to rate, longitude to driver\_ID, latitude to user\_ID, Message to user\_ID.
- gp\_parent** to **gp\_report**: id to Report\_id, report\_ID to type\_id, rate to user\_id, driver\_ID to date\_id, user\_ID to count\_Left, count\_Right, count\_Sudden\_Acc, count\_Sudden\_Stop.
- gp\_report** to **gp\_date**: date\_id to id.
- gp\_date** to **gp\_trip**: id to date\_ID.
- gp\_trip** to **gp\_accelerometer\_readings**: date\_ID to date\_id, accelerometer\_id to x-axis, driver\_id to driver\_ID, trip\_ID to trip\_ID.
- gp\_user** to **gp\_report**: user\_Type to user\_id.
- gp\_administrator** to **gp\_report**: user\_id to user\_id.
- gp\_usertype** to **gp\_report**: userType\_id to user\_id.
- gp\_usertype** to **gp\_trip**: userType\_id to user\_Type.
- gp\_usertype** to **gp\_accelerometer\_readings**: userType\_id to user\_Type.

### 3.10 Scenarios

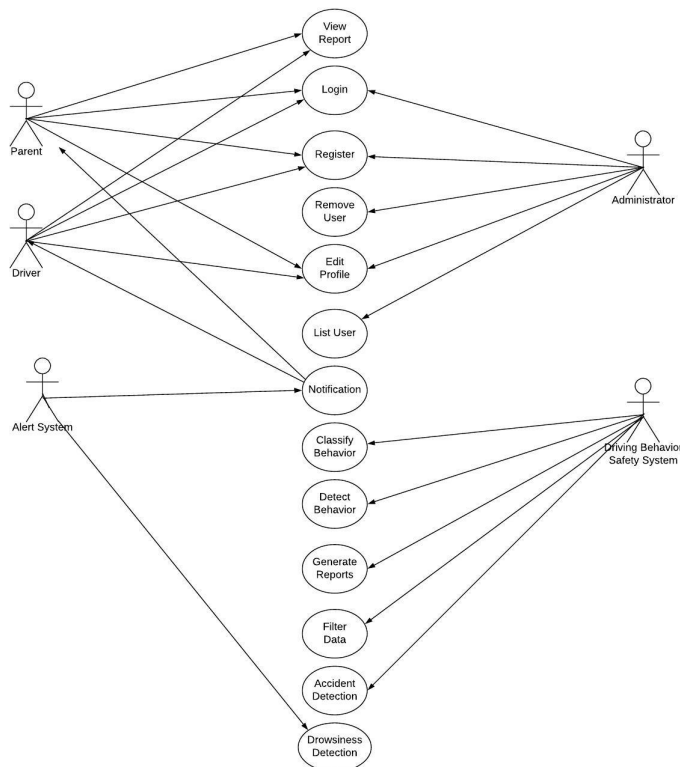
The driver is working for this process and starts with driving and the application starts to send the data to the system and analyzing the data into some generated rating reports.

The administrator is the user that control this process. The administrator is manipulating the driver and the parent in the system and includes: 1. Adding a driver/parent. 2. Editing a driver/parent's information. 3. Removing a driver/parent.

### 3.10.0.3 Scenario 3: Parental Driving Safety System Detecting and Classifying

These actions are done by the Parental Driving Safety System .The system will obtain if there's an aggressive behaviors was made by the driver by comparing the sensor's readings to the normal readings stored in predefined data set. Those readings that have been detected, and then compared against another predefined data set that consists of readings to classify the detecting readings.

Figure 3.19: Use Case Diagram



### 3.10.0.4 Scenario 4: Accident Detection For A Quick Medical Support

People die from late response of medical support so our system will detect if an accident occurred then it will send SMS to emergency contacts contains location of the accident and nearby hospitals .

### 3.10.0.5 Scenario 4: Drowsiness Detection

People sleep while driving .The system will detect if the driver drowsy or not if the driver are sleepy and closes his/her eyes for a few seconds then our system will alert them.

## 3.11 Preliminary Schedule Adjusted

Task	Start Date	End Date
Idea Discussion	1/7/2019	15/8/2019
Proposal	1/9/2019	7/10/2019
Dataset Collection	20/9/2019	20/9/2019
Prototype Implementation	21/9/2019	1/10/2019
Proposal Presentation	7/10/2019	7/10/2019
Designing Class Diagram	8/10/2019	25/10/2019
SRS	25/10/2019	25/11/2019
Implementation	1/11/2019	12/11/2019
SDD	1/2/2020	15/2/2020
Validation And Testing	25/2/2020	25/4/2020
Delivering Papers	30/3/2020	20/4/2020
Delivering Thesis	1/5/2020	20/5/2020
Final Presentation	24/6/2020	24/6/2020

## 3.12 Preliminary Budget Adjusted

Budget will be the cost of a mobile device, 2000 Pounds average. All the users probably will have a mobile device, so no cost needed.



## Chapter 4

# System Design Document

### 4.1 Introduction

#### 4.1.1 Purpose

This software design document purpose is to fully describe the architecture of our Parental Driving Safety System that monitors the driving behavior of siblings and alert parents if any intervention is needed. It will describe the functionality of each sub-system in details, it will also show the communication between the sub-systems and how the data is handled with the aid of diagrams. Also, it explains many algorithms that are used in our system.

#### 4.1.2 Scope

Parental Driving Safety System is a mobile application that monitors the driving behaviour and level of drowsiness of the driver. Once the driver starts driving, the application reads the accelerometer readings throughout the trip and classification is done by the end of the day to show the report to the parent. On the other hand, a real time warning when the driver gets sleepy. Also, the system detects accidents if occurred, send the location of the accident and the nearby hospitals to the emergency contacts. This system also works for people who have private drivers or car rental companies and with the addition of more features it can be used for other numerous applications.

### 4.1.3 Overview

Real time Monitoring and analyzing of incorrect behaviors of the driver to rate the driver so parents can monitor their children and can track their children if an accident occurred. The system should always aim to provide drivers with a trustworthy feedback by drowsiness alert, which also helps the parents to keep track of their children. This document proposes Parental Driving Safety System which is a system that performs detection of an accident, identification of a drivers incorrect behaviour and drowsiness alert using sensors found in smart phones. Parental Driving Safety will consist of a mobile application that will handle the real time data that will be collected from sensors(accelerometer), this data will be analyzed to create automatically generated analyzes that would be helpful to parents to monitor each their children.

### 4.1.4 Definitions and Acronyms

Term	Definition
SDD	Software Design Document.
GSM	The Global System for Mobile Communications
GPS	The Global Positioning System.
MVC	Model View Controller.
Fast-DTW	Fast Dynamic Time Warping.

## 4.2 System Overview

The proposed system a Parental Driving Safety System that uses sensors accelerometer to collect readings of the driving behavior. Therefore, the collected readings is passed through a pre-processing phase that supposedly to get the data formatted. Then, the data is passed to the cloud data storage, which takes the required data to the server side to get the required analysis using classifier Fast DTW algorithm. After analyzing the data the incorrect behavior is obtained. The rating data is always sent to the parent, alert the driver when the driver's eye closed for more than 3 seconds and notify the parent if an

accident occurred with the location of the accident and nearby hospitals. Also, the parent can monitor the ratings of the driver by monitor the reports.

Figure 4.1: System Overview

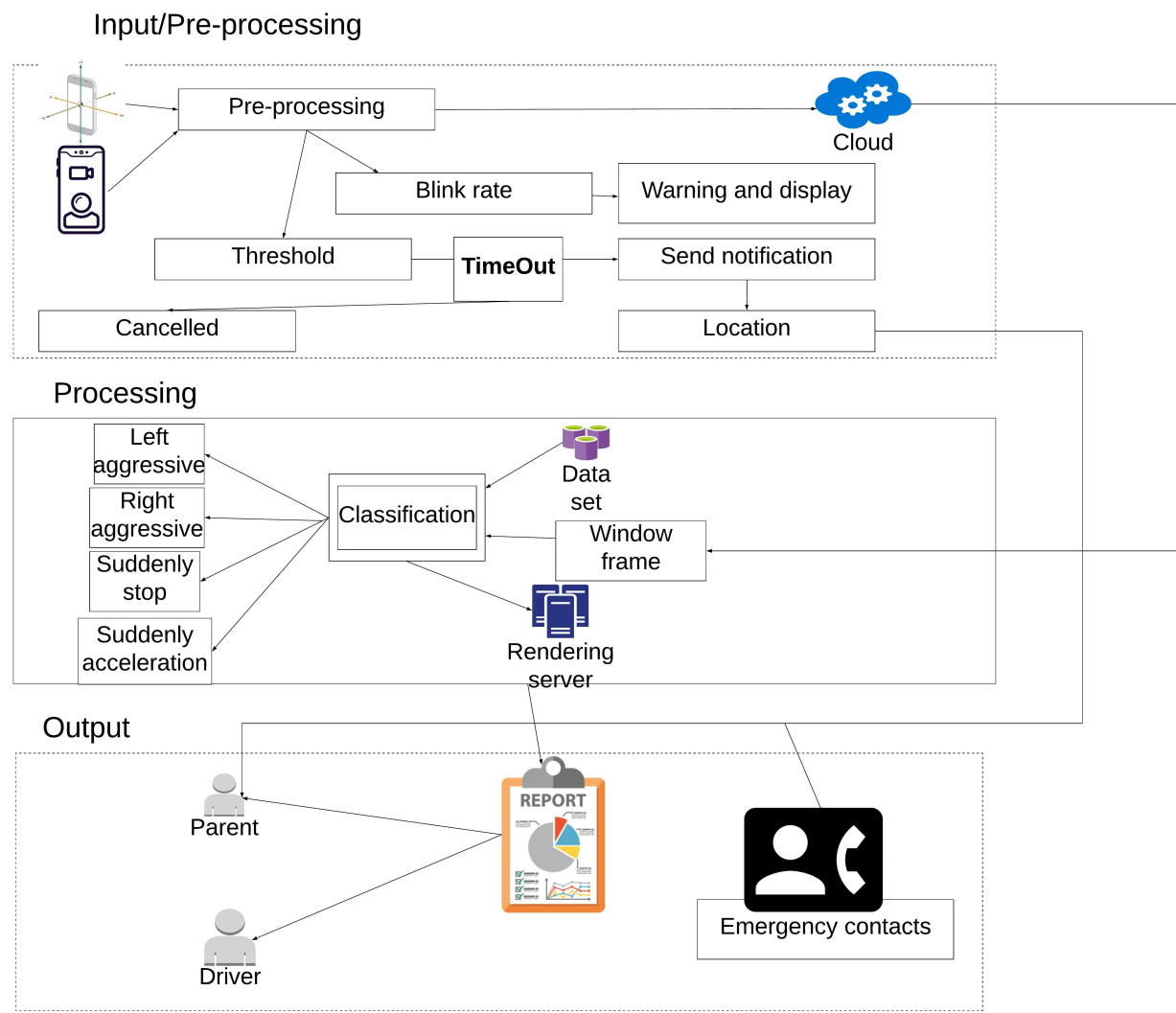
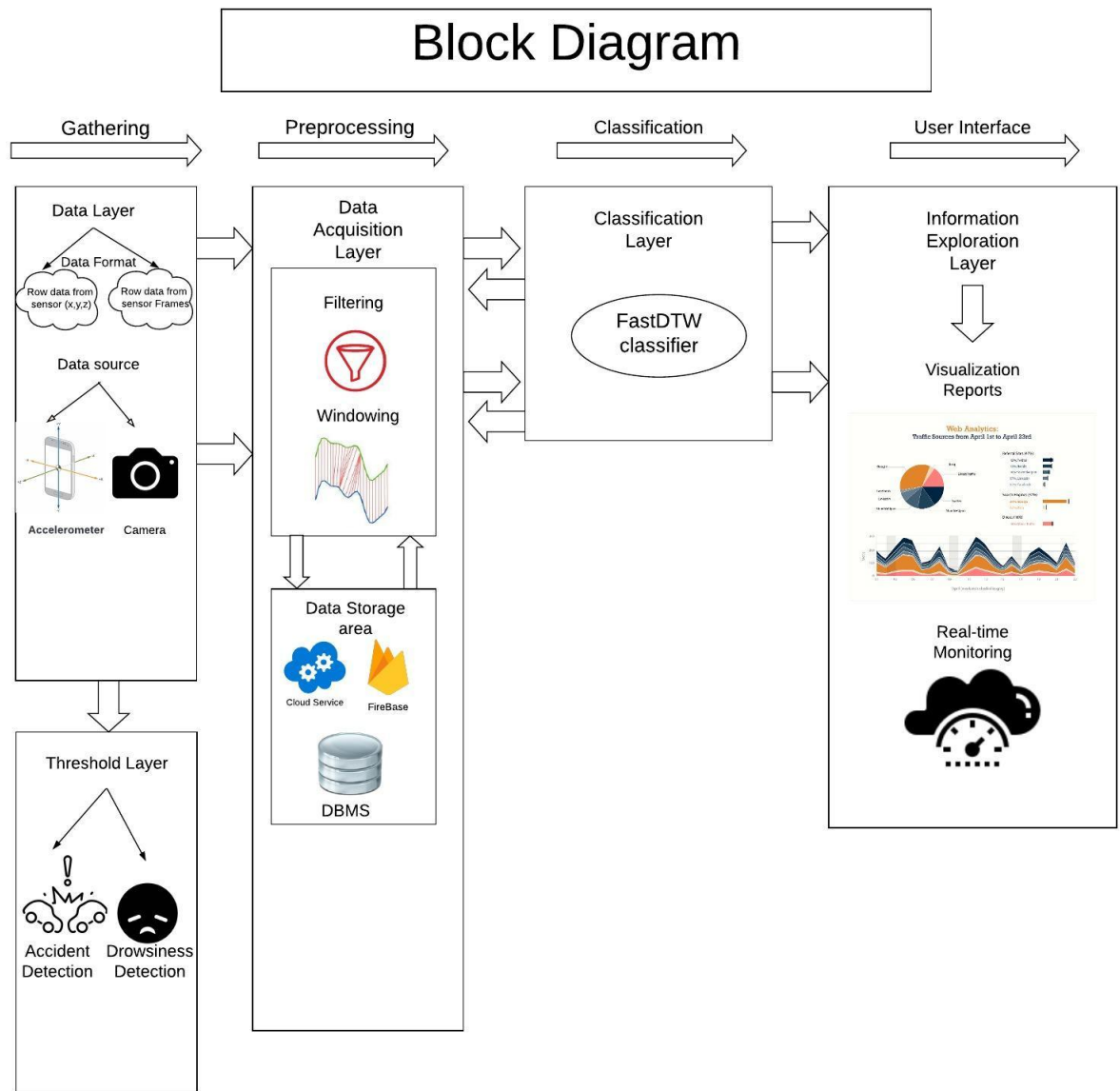


Figure 4.2: Block diagram



## 4.3 System Architecture

### 4.3.1 Architectural Design

In the architectural design, MVC design pattern is used for fast developing process and providing multiple views.

#### 4.3.1.1 Model

Parent, Driver and Administrator models handle the main functionalities of the users of the system.

User type model handles the type of each user.

Trip model manages the details of each trip.

Report model handles the generated number of behaviors on each behavior and the rating of the driver.

Notification model notify the parent with a detailed report about the driver.

Camera model responsible of starting the camera.

DateTime model responsible of having each report date and time.

Accelerometer readings model responsible to get accelerometer readings and write it in database.

Algorithm:

Fast-DTW (Fast Dynamic Time Warping) was used to manage the different behaviors of the driver and to provide the parent with a criticism as exact as could reasonably be expected.

Fast-DTW is a calculation for estimating similarities between two readings.

Libraries:

Google Face Detection: library used for detecting faces.

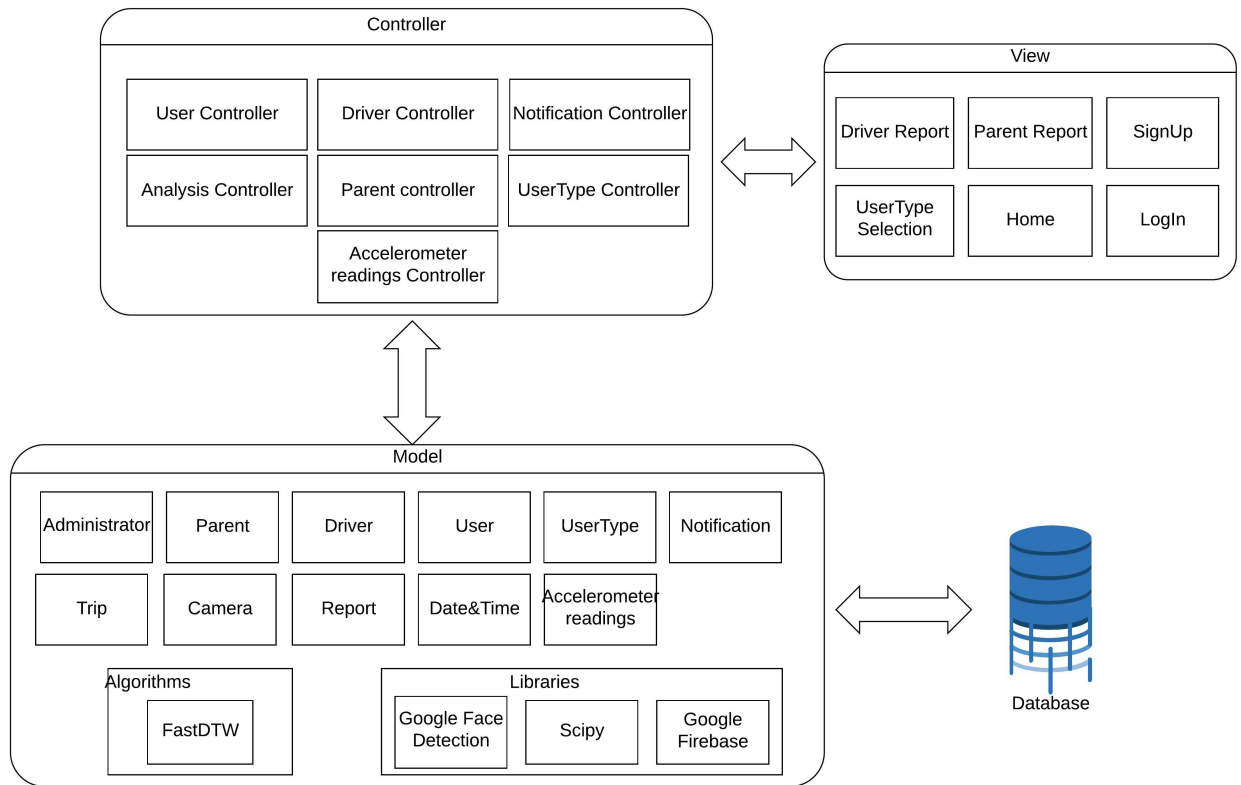
Scipy: library which contains the classifier we are using in our system.

Google Firebase: library used for saving the data in the firebase.

#### 4.3.1.2 View

View is a user interface. View displays data from the model to the user and also enables them to modify the data. The system has three interfaces, one for the driver functionalities, one for the parent functionalities and one for the administrator functionalities.

Figure 4.3: Model view controller



#### 4.3.1.3 Controller

Controller functionality is to connect and bind the model with view. The user interacts with the user interface, the model fetches the data from the database and updates the view. The main controllers we have are : User Controller , login and signup. Driver controller, to start the trip and outputs the accumulative driving behaviour rating of the driver. Parent controller, to show the daily driver driving behaviour report, Accelerometer readings Controller, to read the accelerometer data. User Type Controller, to differentiate between user types. Analysis Controller to Detect the behavior of the driver ,filter data before classification , check if the driver sleepy and finally detect if an accident occurred. Notification Controller , to notify the parent with the driver behaviors.

#### **4.3.1.4 Single Tone**

Singleton pattern is one of the simplest design patterns. This pattern provides one of the best ways to create an object. It involves a single class which is responsible to create an object while making sure that only single object gets created. This class provides a way to access its only object which can be accessed directly without need to instantiate the object of the class. We used it to have only one instance of our DB connection that are shared by multiple objects.

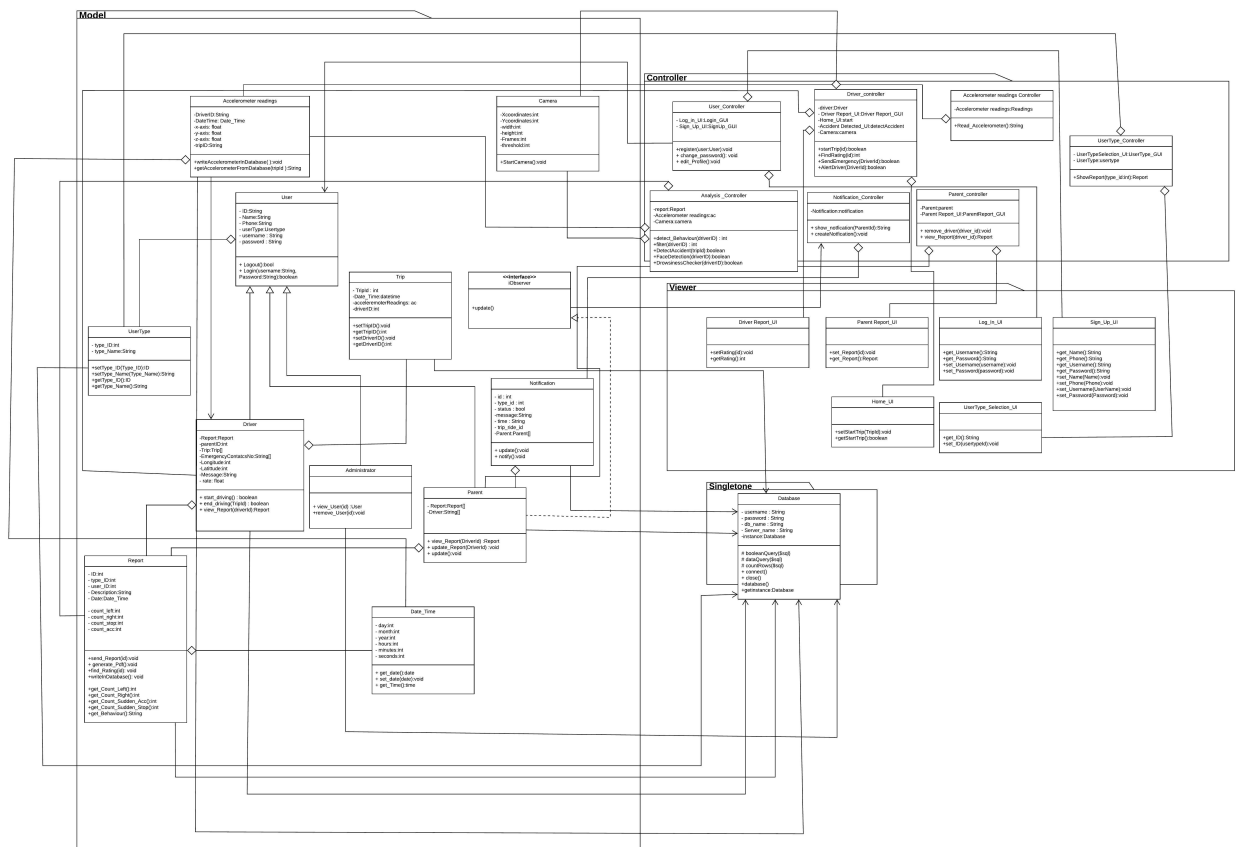
#### **4.3.1.5 Observer**

Observer pattern is used when there is one-to-many relationship between objects such as if one object is modified, its dependent objects are to be notified automatically. Observer pattern falls under behavioral pattern category. We used it to notify many parents in the same time with a report about the behaviors of the driver.

## 4.3.2 Decomposition Description

### 4.3.2.1 Class Diagram

Figure 4.4: Class Diagram



#### 4.3.2.2 Class descriptions

User Abstract class

List of Super classes: None.

List of Sub classes: Administrator, Driver and parent

Purpose: Main class has user's information.

Collaborations: This class aggregate with UserType and inherited by Parent , Driver , Administrator and associates with User Controller

Attributes:

- ID:String
- Name:String



- Username:String
- Phone:String
- userType:UserType

Operations:

- + Logout():bool
- + Login(username:String, Password:String):boolean

#### 4.3.2.3 UserType

List of Super classes: None.

List of Sub classes: none

Purpose: To differentiate between usertypes.

Collaborations: This class aggregate with User, User Type Controller and associates with Database.

Attributes:

- typeId:int
- typeName:String

Operations:

- +setTypeID(TypeID):ID
- +setTypeName(TypeName):String
- +getTypeID():ID
- +getTypeName():String

#### 4.3.2.4 Parent

List of Super classes: User . iobserver

List of Sub classes: None.

Purpose: Every Parent wants to know the driving behavior of his children.

Collaborations: This class aggregate with Notification , Report and parent controller.Inherits from User and iobserver.Association with Parent Controller. Attributes:

- Report:Report[]

-Driver:String[]

Operations:

+ viewReport(DriverId) :Report

+ updateReport(DriverId) :void

+ update():void

#### 4.3.2.5 Administrator

List of Super classes: User.

List of Sub classes: None

Purpose: To edit anything in any user type.

Collaborations: This class association with database and inherits from User

Attributes:

None

Operations:

+ viewUser(id) :User

+removeUser(id):void

#### 4.3.2.6 Driver

List of Super classes: User.

List of Sub classes: None.

Purpose: Start trip and has his own report.

Collaborations: This class aggregates with Report,Driver Controller and Trip.Inherits from User. Association with accelerometer readings and database.

Attributes:

-Report:Report

- rate: float

-parentID:int

-Trip:Trip[]

-EmergencyContatcsNo:String[]

-Longitude:int

-Latitude:int

-Message:String

Operations:

+ startdriving() : boolean

+ enddriving(TripId) : boolean

+ viewReport(driverId):Report

#### 4.3.2.7 Report

List of Super classes: Result,Date.

List of Sub classes: None

Purpose: to classify the driver behavior.

Collaborations: Associates with database.Aggregation with Parent, Parent Controller, Date<sub>TimeandDr</sub>

*Attributes :*

– *ID : int*

– *typeID : int*

– *userID : int*

– *Description : String*

– *Date : DateTime*

– *countleft : int*

– *countright : int*

– *countstop : int*

– *countacc : int*

Operations:

+sendReport(id):void

+ generatePdf():void

+findRating(id): void

+writeInDatabase(): void

+getCountLeft():int

+getCountRight():int

+getCountSuddenAcc():int

+getCountSuddenStop():int  
 +getBehaviour():String

#### 4.3.2.8 Accelerometer Readings

List of super classes : None

List of Sub classes : Preprocessing

Purpose : Get readings from accelerometer sensor

Collaborations :Aggregates with DateTime, Analysis Controller and Accelerometer Readings Controller. Associates with Driver and Database.

Attributes :

-DriverID:String

-DateTime: *DateTime*

- *x - axis : float*

- *y - axis : float*

- *z - axis : float*

- *tripID : String*

Operations : None

+ *writeAccelerometerInDatabase() : void*

+ *getAccelerometerFromDatabase(tripId) : String*

#### 4.3.2.9 Notification

List of super classes : iSubject

List of Sub classes : None

Purpose : pre-processing the data and send to the result class

Collaborations : This class Aggregates with Notification Controller and Parent.Associates with Database.

Attributes :

+ id : int

+ user :user

+ *type<sub>i</sub>d : int*

+ *status* : *bool*  
+ *time* : *String*  
+ *triprideid*  
*Operations* :  
– *update()*  
– *notify()*  
– *createNotification()*

#### 4.3.2.10 Trip

List of super classes : None.

List of Sub classes : None

Purpose : Trip details.

Collaborations : Aggregates with driver.Associates with Database.

Attributes :

- TripId : int  
-DateTime:datetime  
-acceleremoterReadings: ac  
-driverID:int

Operations:

+setTripID():void  
+getTripID():int  
+setDriverID():void  
+getDriverID():int

#### 4.3.2.11 Camera

List of super classes : None.

List of Sub classes : None

Purpose : Camera Initialization.

Collaborations : Aggregates with Driver Controller,Analysis Controller.

Attributes :

-Xcoordinates:int  
 -Ycoordinates:int  
 -width:int  
 -height:int  
 -Frames:int  
 -threshold:int  
 Operations:  
 +StartCamera():void

#### 4.3.2.12 Database

List of super classes : None

List of Sub classes : none

Purpose : system database

Collaborations : Associates with Trip,Notification,Parent,Usertype,Driver,Report,Administrator,Accelerometer, Accelerometer readings,

Attributes :

- username : String
- password : String
- dbname : String
- Servername : String

Operations:

booleanQuery(*sql*)dataQuery(*sql*)

countRows(*sql*)

– connect() – close()

#### 4.3.2.13 iObserver

List of Super classes: None.

List of Sub classes: Parent

Purpose: Update Notification.

Collaborations: Associates with Notification Controller. Parent Inherits from iobserver.

Attributes: None

Operations:

+update()

#### 4.3.2.14 User Controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates User.

Collaborations: Associates with User and aggregates with SignUpUI,LoginUI.

Attributes:

- LoginUI:LoginGUI

- SignUpUI : SignUpGUI

Operations :

+ register(*user : User*) : void

+ changepassword() : void

+ editprofile() : void

#### 4.3.2.15 Driver Controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates Driver.

Collaborations: Aggregates with Driver,DriverReportUI,CameraHomeUI and Accident DetectedUI.

Attributes:

-driver:Driver

- Driver ReportUI:Driver ReportGUI

-HomeUI:start

-Accident DetectedUI:detectAccident

-Camera:camera

Operations:

+startTrip(id):boolean  
+FindRating(id):int  
+SendEmergency(DriverId):boolean  
+AlertDriver(DriverId):boolean

#### 4.3.2.16 Accelerometer readings Controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates Accelerometer readings.

Collaborations: Aggregates with Accelerometer readings.

Attributes:

-Accelerometer readings:Readings

Operations:

+Read<sub>Accelerometer</sub>() : *String*

#### 4.3.2.17 UserType Controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates UserType.

Collaborations: Aggregates with UserTypeSelectionUI and UserType

Attributes:

- UserTypeSelectionUI:UserTypeGUI

- UserType:usertype

Operations:

+ShowReport(typeid:int):Report

#### 4.3.2.18 Analysis Controller

List of Super classes:None

List of Sub classes: None



Purpose: Manipulates UserType.

Collaborations: Analysis *controller aggregates with Report, Camera and accelerometer Readings.*

*Attributes :*

- *report : Report*
- *Accelerometer readings : ac*
- *Camera : camera*

*Operations :*

- + *detectBehaviour(driverID) : int*
- + *filter(driverID) : int*
- + *DetectAccident(tripId) : boolean*
- + *FaceDetection(driverID) : boolean*
- + *DrowsinessChecker(driverID) : boolean*

#### 4.3.2.19 Notification Controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates User.

Collaborations: Associates with iObserver and aggregates with Notification.

Attributes:

-Notification:notification

Operations:

- + *show\_notification(ParentId) : String*
- + *createNotification() : void*

#### 4.3.2.20 Parent controller

List of Super classes:None

List of Sub classes: None

Purpose: Manipulates Parent.

Collaborations: Aggregates with Parent ReportGUI,Parent.

Attributes:

-Parent:parent

-Parent ReportGUI:ParentReportGUI

Operations:

+register(user:User):void

+ removedriver(driverid):void

+ viewReport(driverid):Report

Figure 4.5: Process Diagram

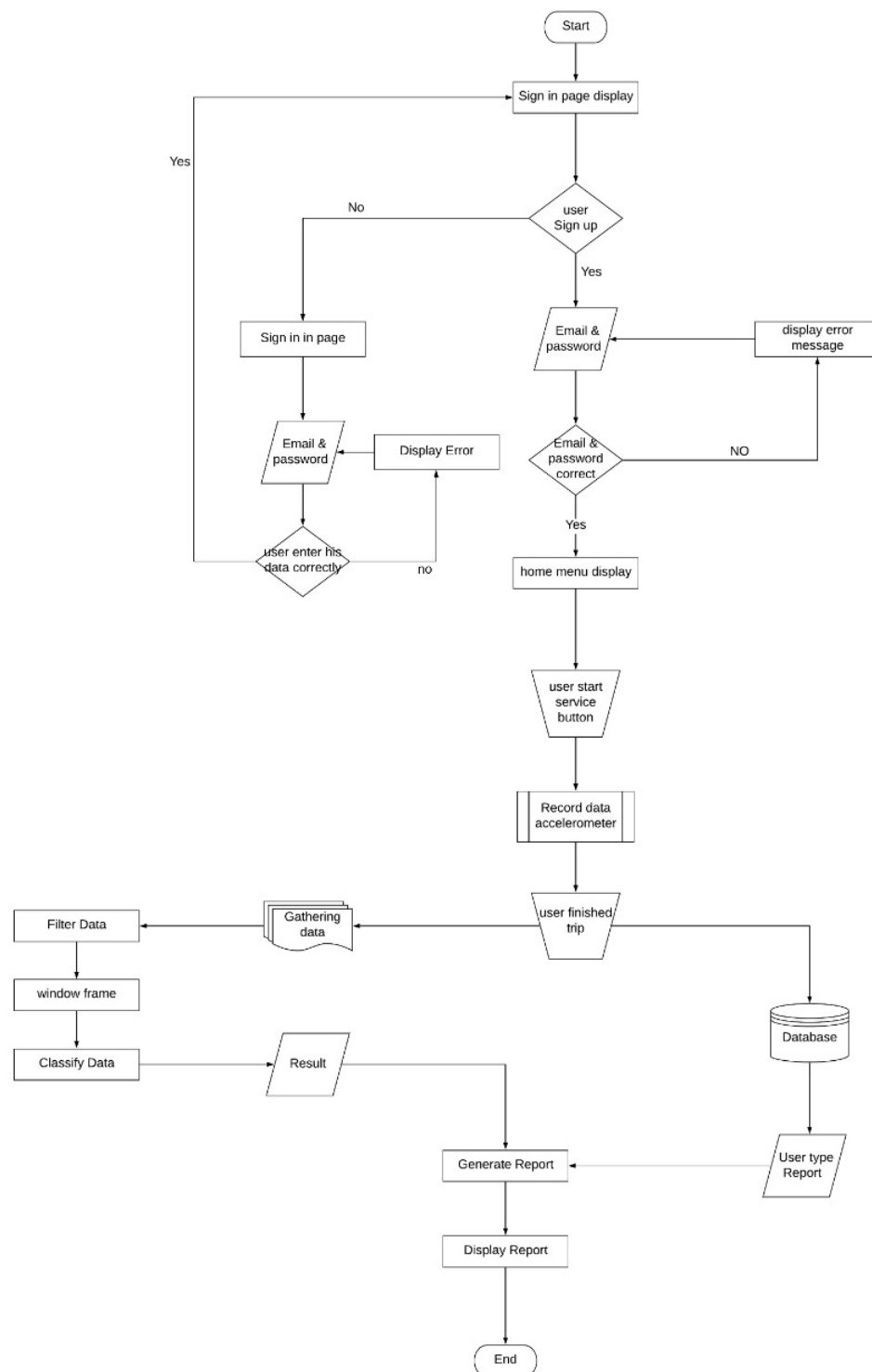


Figure 4.6: Activity Diagram

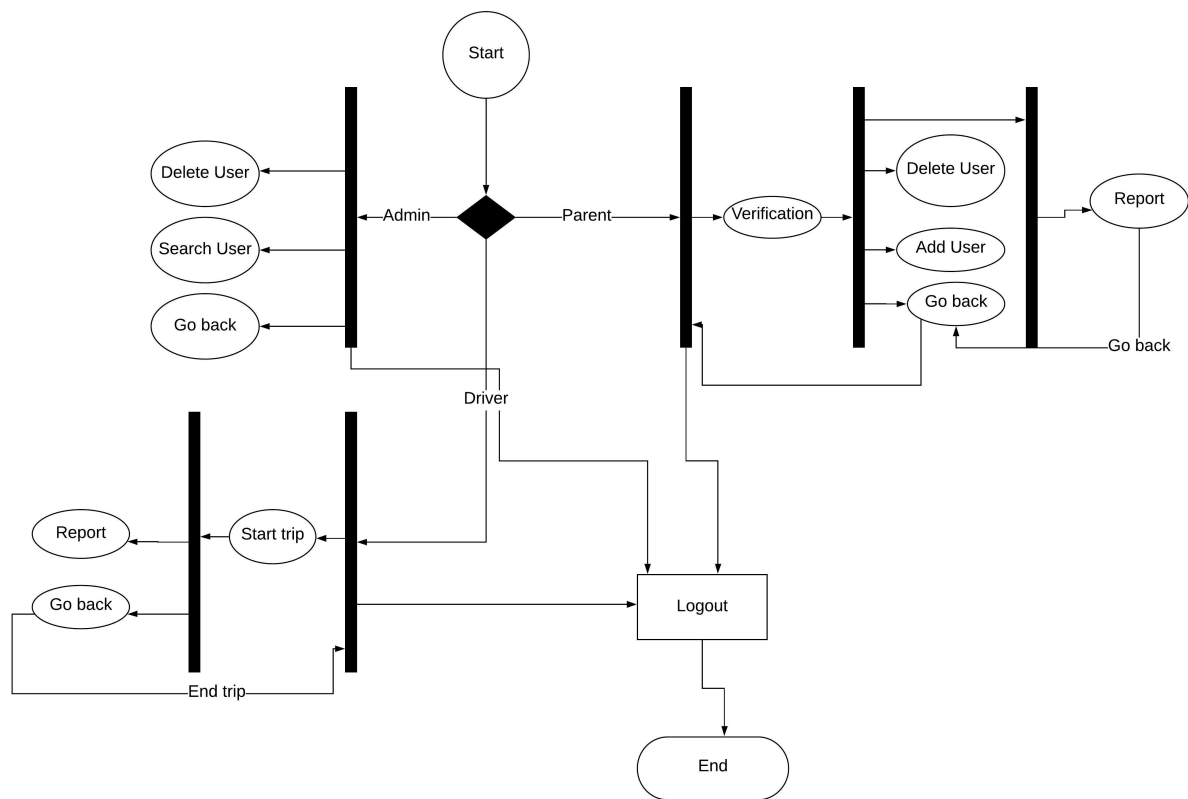


Figure 4.7: Signup

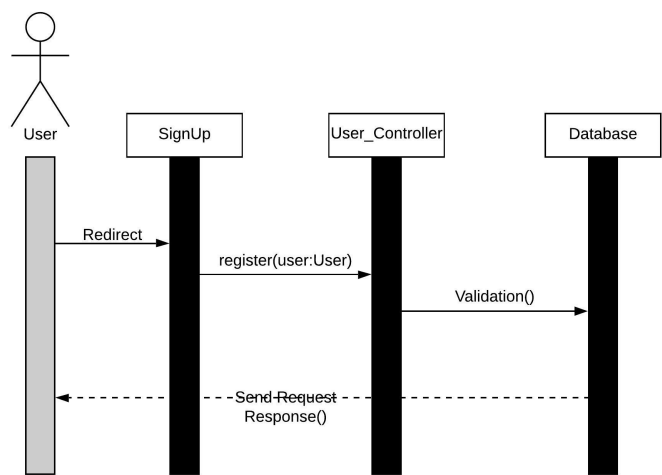


Figure 4.8: Driving Behavior

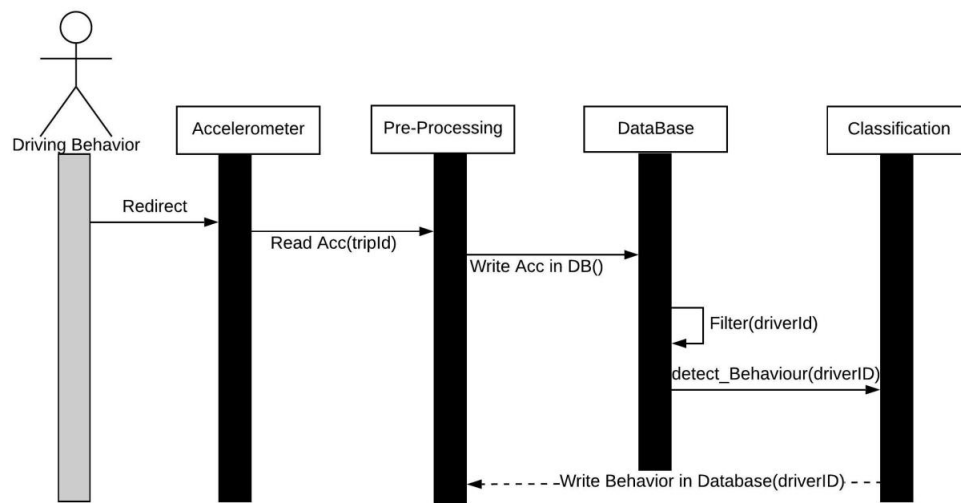


Figure 4.9: Drowsiness Detection

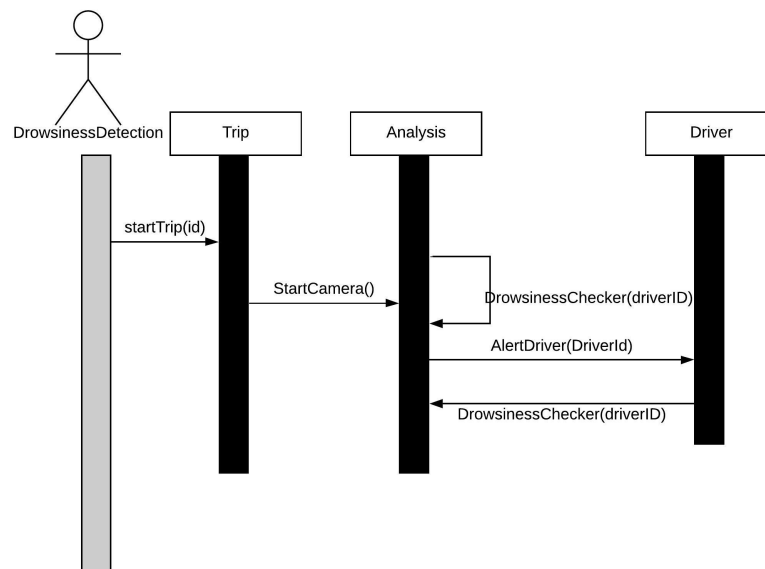
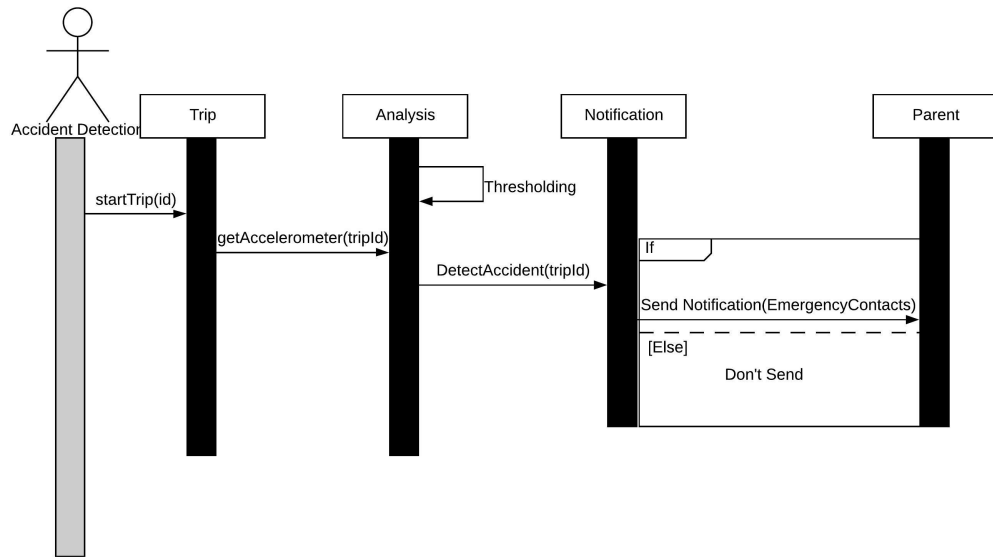


Figure 4.10: Accident Detection



### 4.3.3 Design Rationale

As mentioned previously, we have used Model-View-Controller (MVC) as our architecture as it helped us to separate the application and the data of our system from the presentation. So, we can easily make modifications, re-use and optimize functionality. Also the software we are developing efficiency and accuracy is a very important aspect of it so it will be very sensitive with data so this should be developed in a very accurate and efficient way.

#### 4.3.3.1 Possible Algorithms

**Naive Bayes:** It works on conditional probability. It calculates the probability of the input relative to some decisions that was previously taken. It is well suited when the input has large number of dimensions.

**Random Forest:** It starts with a decision tree as a machine learning. It models the data and draws multiple curves and then chooses the strongest curve which contains the greater number of curves assembled together. The data starts to move through different sub branches of the tree until it reaches a decision.

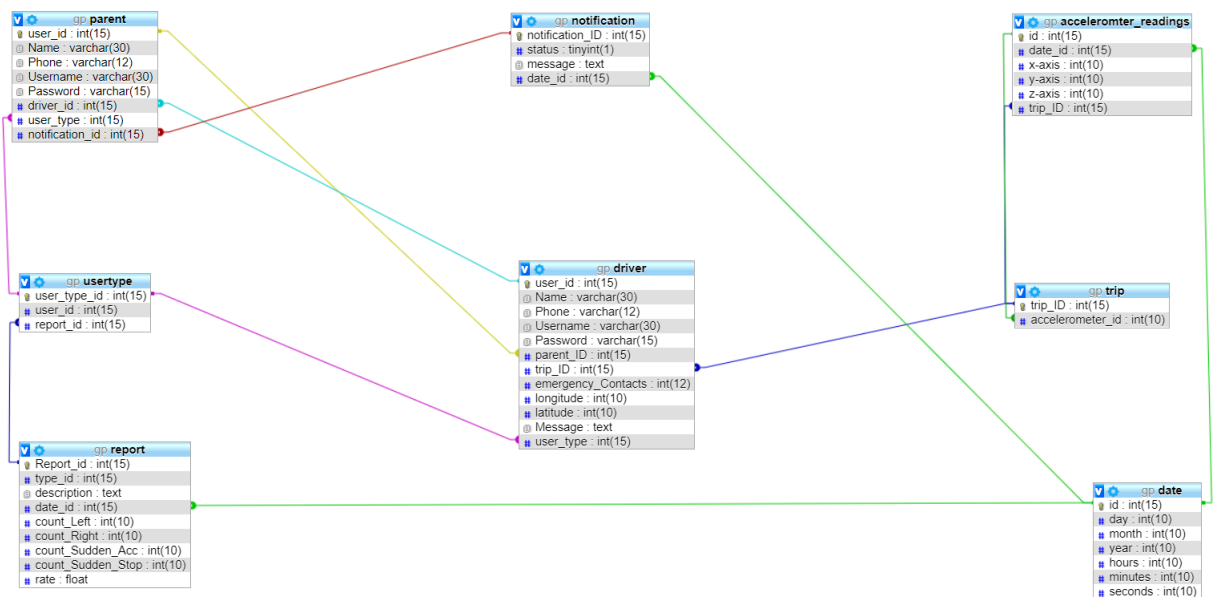
KNN :is a basic algorithm that compares between corresponding vertices (Euclidean Matching) rather than measuring the similarities between two different signals in different time intervals.

We have chosen RF as our classifier because it was the best one that was able to differentiate between different types and the one which has given us the highest accuracy. Also, the architecture that this classifier was based on matches our problem as we are comparing a window with another window that contains many points not point with point .

## 4.4 Data Design

### 4.4.1 Data Description

Figure 4.11: Data Description



### 4.4.2 Data Dictionary

1-Driver :This entity holds the information of each driver like tripId ,reportId and parentId.

2-Parent :This entity holds the information of driver's parent .

3-Report :This entity will hold the information of classification output ,according to the user type.

4-Trip :This entity holds the information of each trip.

5-Notification :This entity holds the message to deliver it to the parent.

6- Date : This entity holds the date of each operation.

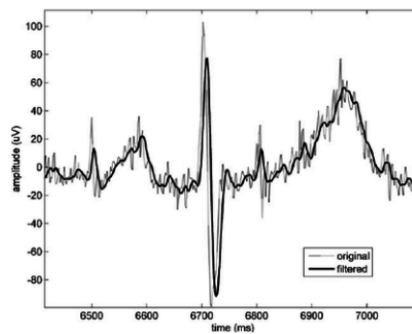
7-Accelerometer Readings : This entity holds the data of the mobile accelerometer of each driver.

## 4.5 Component Design

### 4.5.1 Pre-Processing

In this section, we get the accelerometer readings from the mobile device and pass the readings to low pass filter algorithm to reduce noise for getting accurate readings. For the drowsiness detection feature, the mobile camera first must detect the driver's face to begin processing.

Figure 4.12: Low Pass Filter



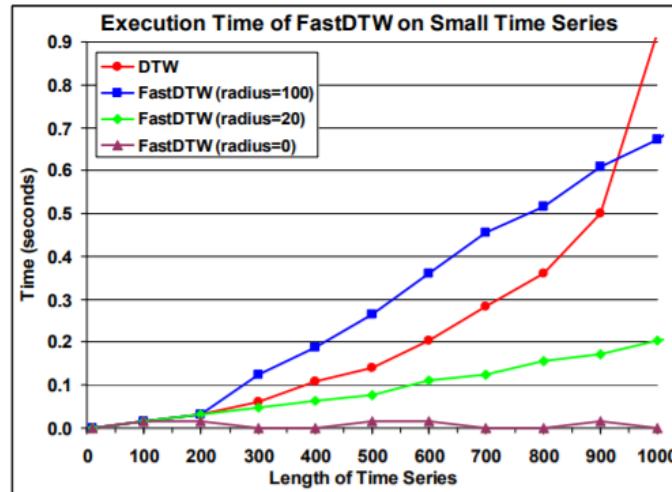


## 4.5.2 Classification

### 4.5.2.1 Fast-DTW

In this section, we use the fast dtw algorithm in the classification section. Basically, the concept of this algorithm is find the optimal alignment between two time series. It is often used to determine time series similarity, classification and to find corresponding regions between two time series. Fast-DTW is an enhancement to the normal dtw algorithm, as the normal dtw's complexity is  $O(N^2)$  and works for few thousands of points. While the Fast-DTW's complexity is  $O(N)$  and works for more points. This algorithm doesn't guarantee the optimal solution, but always very close to the optimal. We used the euclidean distance to calculate the distance between the classes and then passed to the fast dtw to be classified. The algorithm depends on three main steps; coarsening, projection and refinement. In the coarsening part, it shrinks a time series into a smaller time series that represents the same curve as accurately as possible with fewer data points. For the projection part, finds a minimum-distance warp path at a lower resolution, and use that warp path as an initial decision for a higher resolution's minimum-distance warp path. Finally, the refinement part, refine the warp path projected from a lower resolution through local adjustments of the warp path.

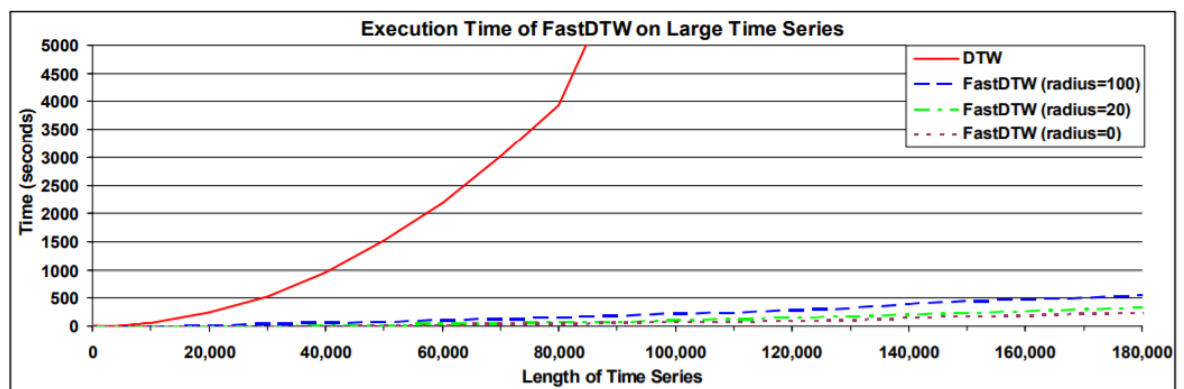
Figure 4.13: DTW



The FastDTW algorithm, with a radius of 100, takes longer to run than DTW until the size of the time series exceeds approximately 900 points. However, with a radius of 0 or

20, the DTW algorithm is never faster than the FastDTW algorithm for small time series, and once the length of the time series exceed 200-300 points, FastDTW becomes the more efficient algorithm.

Figure 4.14: DTW 2



## 4.6 Human Interface Design

### 4.6.1 Screen Images

Figure 4.15: Welcome GUI



Figure 4.16: Signin GUI

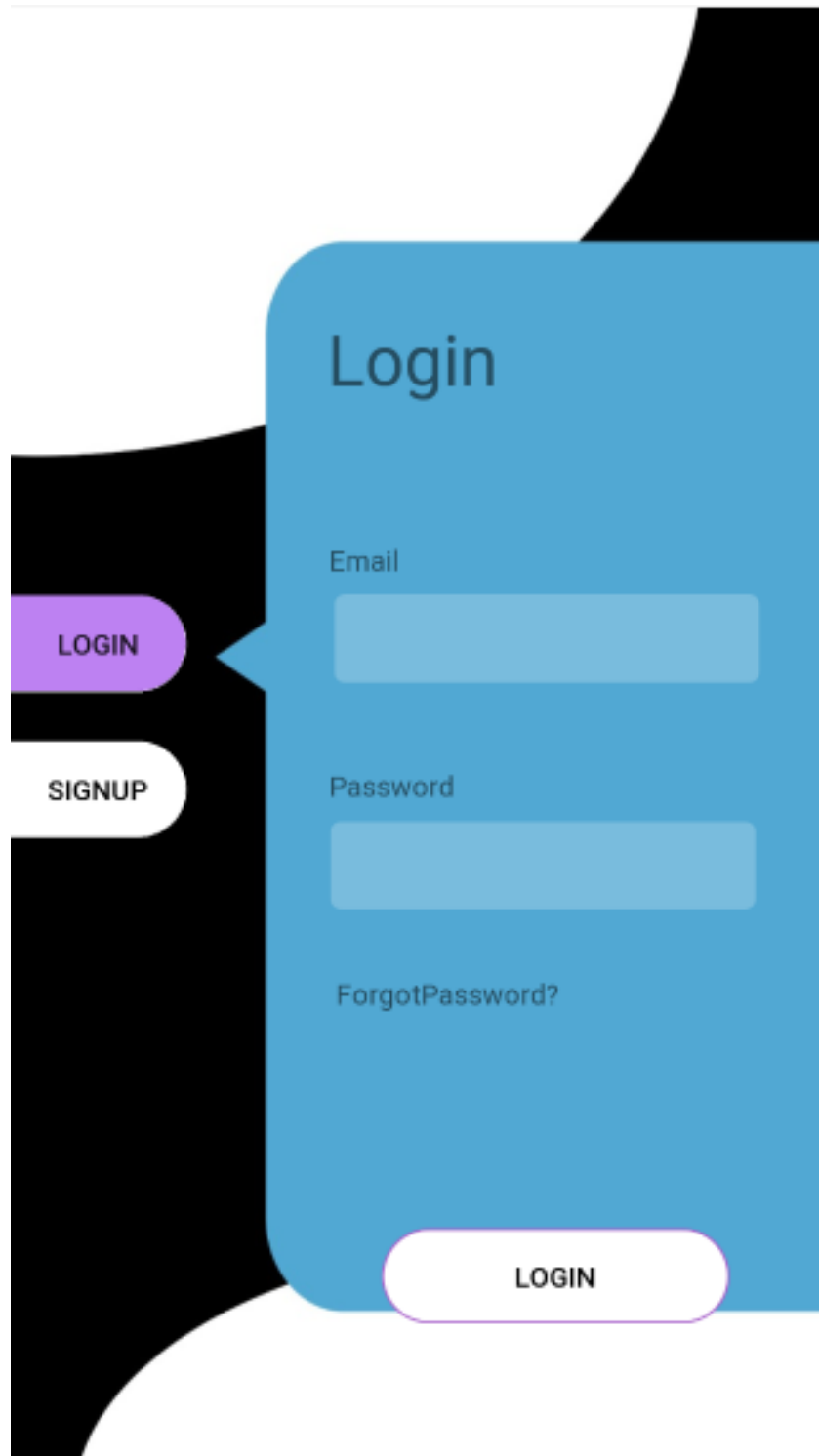


Figure 4.17: SignUpas GUI

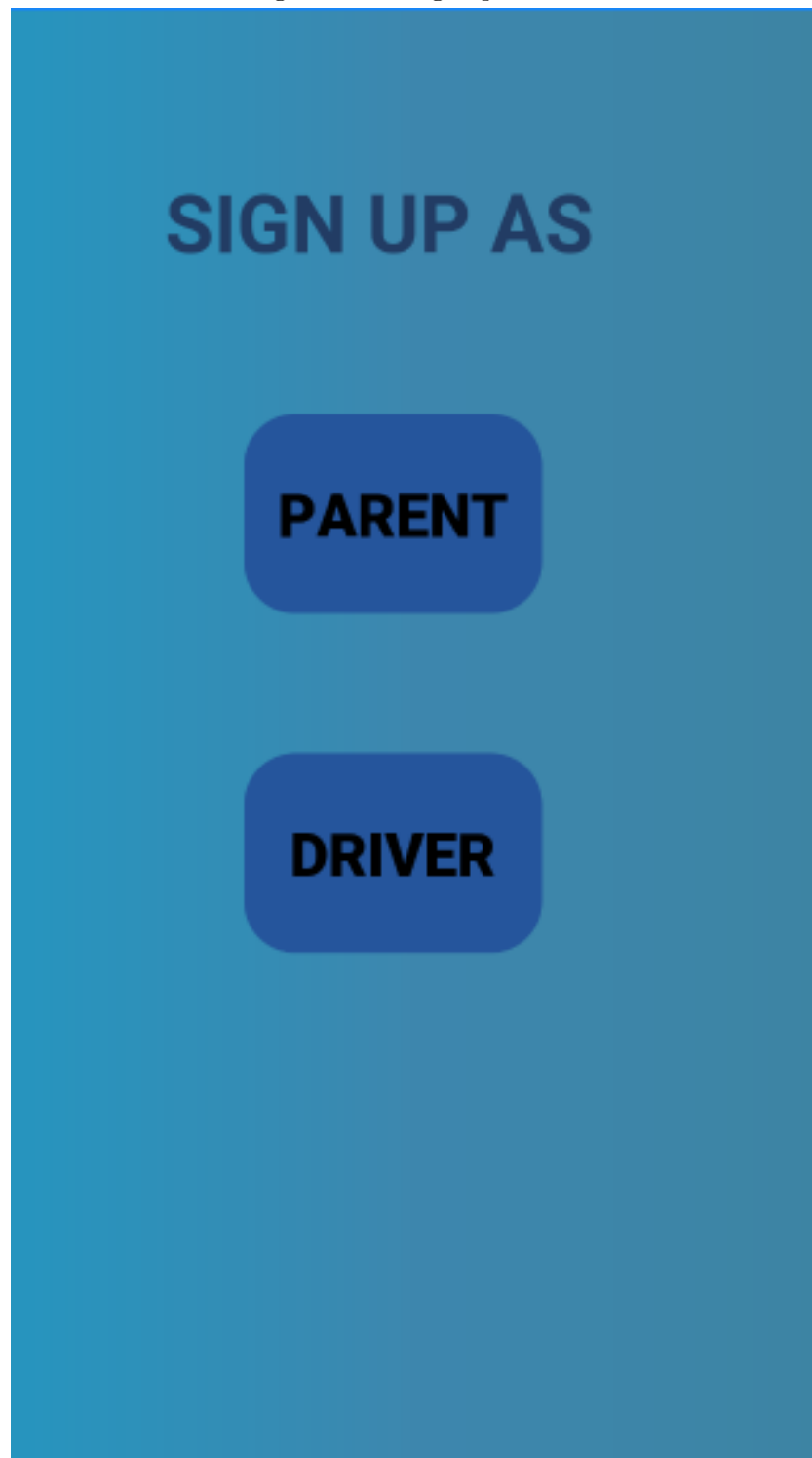



Figure 4.18: Signup Parent GUI



The logo for 'Parental Driving Safety' features a white steering wheel with a yellow sun in the center. A black road with white dashed lines leads from the bottom center of the steering wheel towards the sun. Below the steering wheel, the words 'PARENTAL' and 'DRIVING SAFETY' are written in a stylized, italicized font.

First Name

Last Name

Enter Email

Enter Mobile Number

Enter Password


Confirm Password

Age

Driver Phone

**SUBMIT**

Figure 4.19: SignupDriver GUI



The logo features a white steering wheel with a yellow sun in the center, and a black road with white dashed lines leading towards the sun. Below the graphic, the text "PARENTAL" is in a large, bold, sans-serif font, and "DRIVING SAFETY" is in a smaller, italicized, sans-serif font.

First Name

Last Name

Enter Email

Enter Mobile Number

Enter Password

Confirm Password

Age

Emergency Contact

**SUBMIT**

Figure 4.20: Start Service GUI





Figure 4.21: Drowsiness GUI



Figure 4.22: Emergency GUI

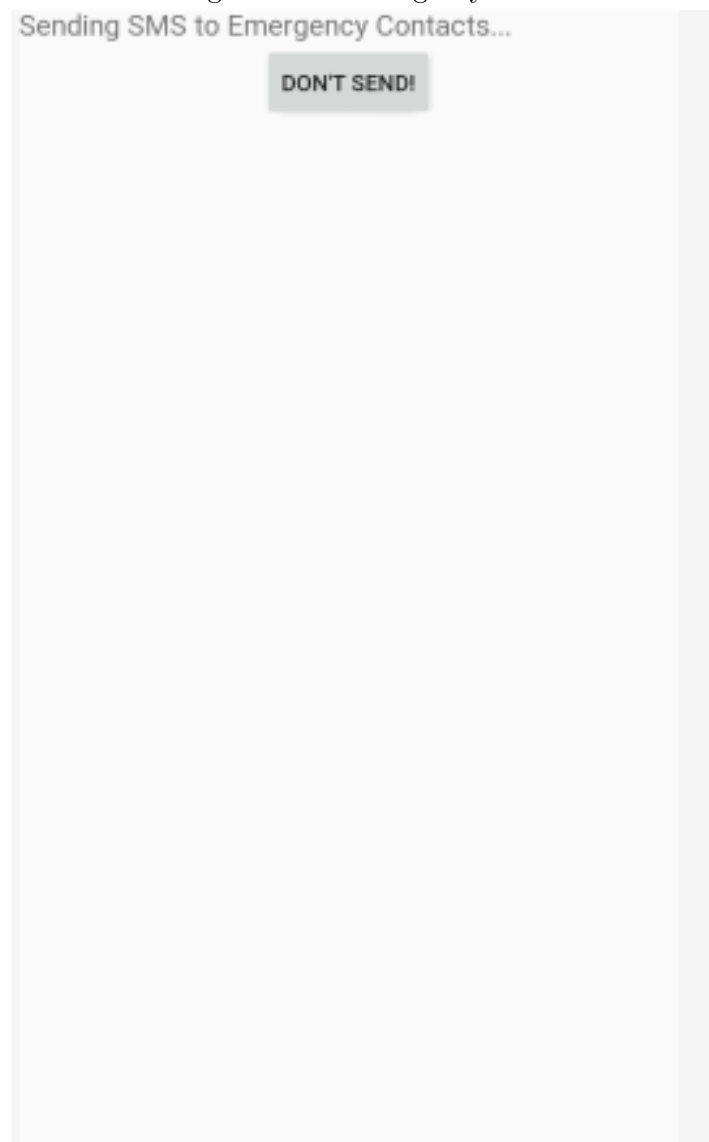


Figure 4.23: Driver Report GUI

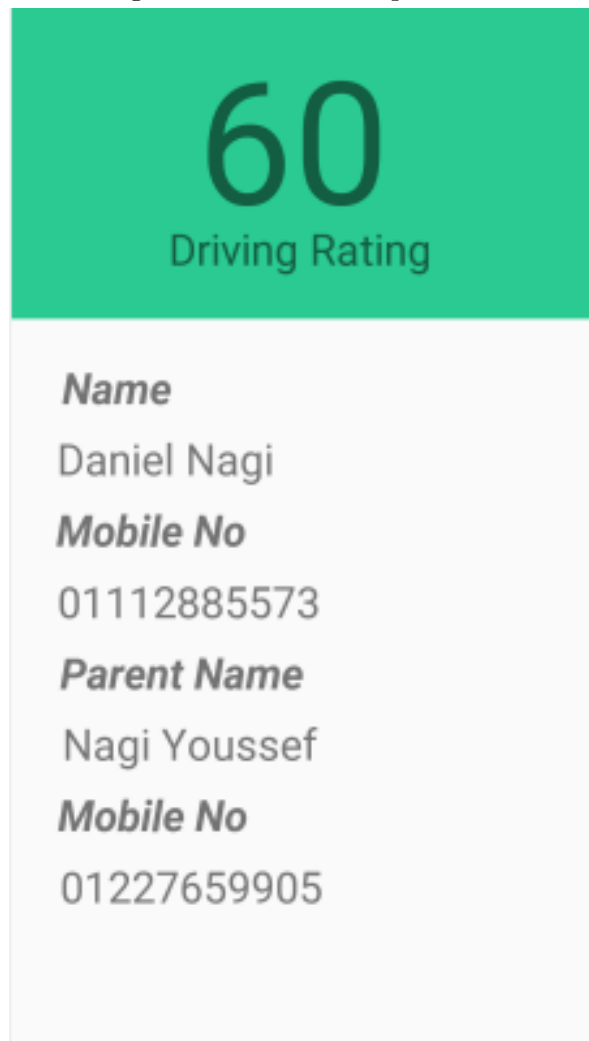
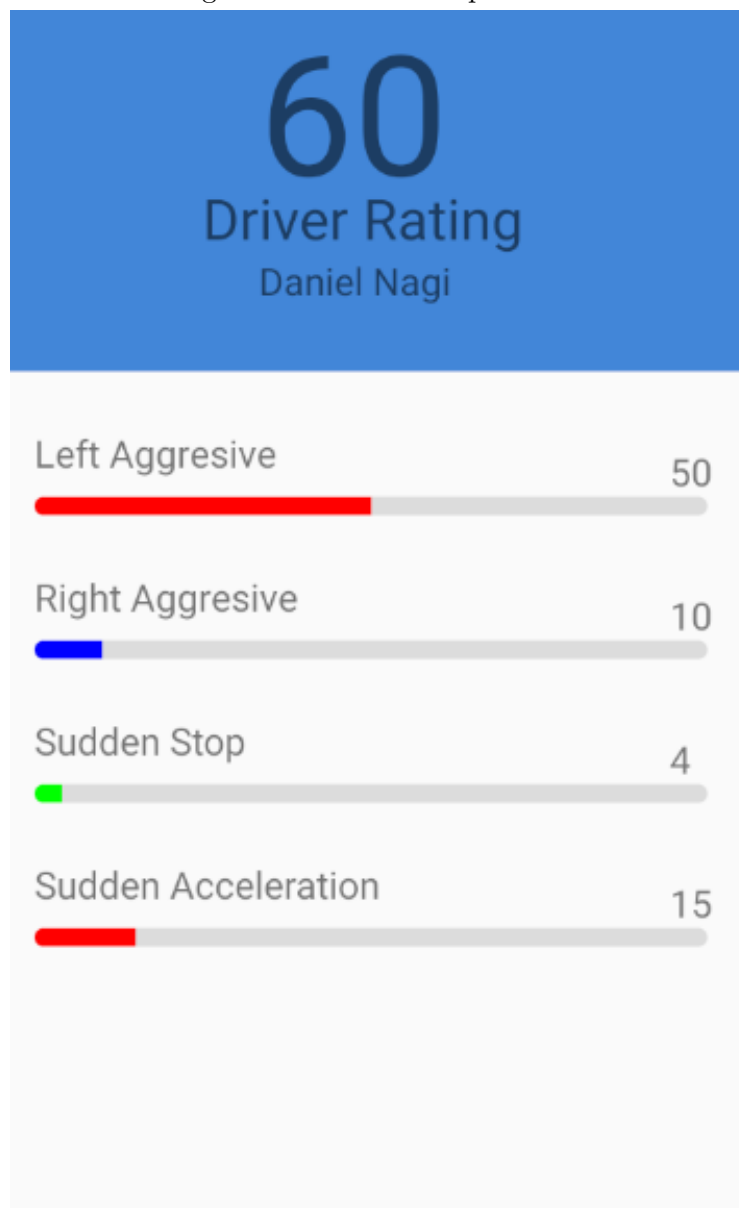


Figure 4.24: Parent Report GUI



#### 4.6.2 Screen Objects and Actions

1. Welcome Screen: This is the main screen when the user open the application.
2. Sign in Screen :In this screen the user log in in his account within our service.

3. SignUp As Screen :In this screen the user chooses the user type to sign up with.
4. Parent SignUp Screen :In this screen the parent register within our service.
5. Driver SignUp Screen :In this screen the driver register within our service.
6. Start Service Screen :In this screen the driver start the service of our application.
7. Drowsiness Screen :In this screen the application checks if the the driver sleepy to alert him/her.
8. Emergency Screen :In this screen the application detected an accident and sends SMS and the location of the accident to the emergency contacts and give you the opportunity to cancel sending it.
9. Driver Report Screen :In this screen a detailed report about the driver .
10. Parent Report Screen :In this screen a detailed report about the driver and the driving behaviors.

## 4.7 Requirements Matrix

<i>Requirement ID</i>	<i>Requirement Type</i>	<i>Requirement Name</i>	<i>Requirement Description</i>	<i>Status</i>	<i>In SDD</i>
3.1.1	Required	Login	The user input his/her username and password to check them in the database.	Completed	Sequence Diagram ,Class Diagram
3.1.2	Required	Logout	User clicks logout so the system is no longer available with its login features..	Pending	Class Diagram
3.1.3	Required	Edit Profile	Admin Edits User.	Pending	Class Diagram
3.1.4	Required	Register	The user input his/her First Name,Last Name,Email,Mobile Number,Password,Age to write them in the database.	Completed	Sequence Diagram ,Class Diagram
3.2.1	Required	Read Accelerometer	Get Accelerometer readings.	Completed	Class Diagram
3.3.1	Required	Start Trip	The driver clicks on a button to start trip.	Completed	Class Diagram
3.3.2	Required	Send Emergency	Send accident location and nearest hospitals to emergency contacts or driver cancel the process.	Completed	Sequence Diagram ,Class Diagram
3.3.3	Required	Alert Driver	Alarm is fired when the driver is sleepy.	Completed	Sequence Diagram ,Class Diagram
3.3.4	Required	View Driver Report	Get the Rating of the driver's driving behavior.	Pending	Class Diagram
3.4.1	Required	Filter	Removing noise from accelerometer data	Pending	Class Diagram
3.4.2	Required	Detect Behavior	Detect Abnormal driving behavior.	Completed	Sequence Diagram ,Class Diagram
3.4.3	Required	Detect Accident	Detecting Accidents using threshold.	Completed	Sequence Diagram ,Class Diagram
3.4.4	Required	Face Detection	Detects Face.	Completed	Class Diagram
3.4.5	Required	Check Drowsiness	Alarm if the driver was sleepy.	Completed	Sequence Diagram ,Class Diagram
3.5.1	Required	Get Behavior	get the abnormal behaviors counters, the driver accumulative driving rating and outputs the report.	Completed	Class Diagram
3.5.2	Required	Find Rating	Get all the counters daily and calculate the accumulative rating.	Pending	Class Diagram
3.6.1	Required	Parent Report	Show a daily Report of the parent's son/daughter driving behavior.	Pending	Class Diagram
3.7.1	Required	View User	View user from database	Pending	Class Diagram
3.7.2	Required	Remove User	Delete the selected records from the database.	Pending	Class Diagram

## Chapter 5

# Evaluation

### 5.1 Introduction

the application was gone through different investigations to test and recognize the purposes of solidarity of the application proposed. The main analysis was essentially to perceive which classifiers are better with our dataset and think about there precision with the remainder of the classifiers on the equivalent dataset. The subsequent investigation expects to think about the aftereffects of our purposed application with the other related work of driving behaviour discovery.

### 5.2 Experiment 1 Driving Behaviour detection classification

#### 5.2.1 Goal

Choose which classifier gives the application the highest accuracy with driving behaviour detection.

#### 5.2.2 Classifiers Tested

1. Fast-DTW
2. KNN
3. SVM
4. Naive Bayes

### 5.2.3 Task

We get the accelerometer readings in the co-ordinates(X, Y, Z) from the mobile device and pass the readings to a low pass filter algorithm to reduce noise and remove gravity for getting accurate readings to classify the behaviour of the driver. Also, we used Window Sliding Technique to perform the required classification on the explicit window size of given enormous array. Window begins from the first component and continues moving right by 6 component. The goal is to divide the large array into windows to classify each window individually. Furthermore, at the same time detect if an accident occurred. For the drowsiness detection feature, the mobile camera first must detect the driver's face to begin processing.

### 5.2.4 Results

<i>Classifier</i>	<i>Fast-DTW</i>	<i>KNN</i>	<i>SVM</i>	<i>Naive Bayes</i>
Accuracy	92.5%	83.47%	89.67%	78.02%
Precision	92.9%	79.3%	89.4%	83.66%
Recall	92.4%	81%	90.2%	70.2%

As should be obvious the distinction in accuracy, precision and recall between Fast-DTW with the other classifiers are excessively enormous so we applied Fast-DTW in our application.

## 5.3 Experiment 2 Comparing proposed system with related work

### 5.3.1 Goal

Our goal is to compare the results of our system that used Fast-DTW for classification with this similar work [23]. They used DTW classifier.

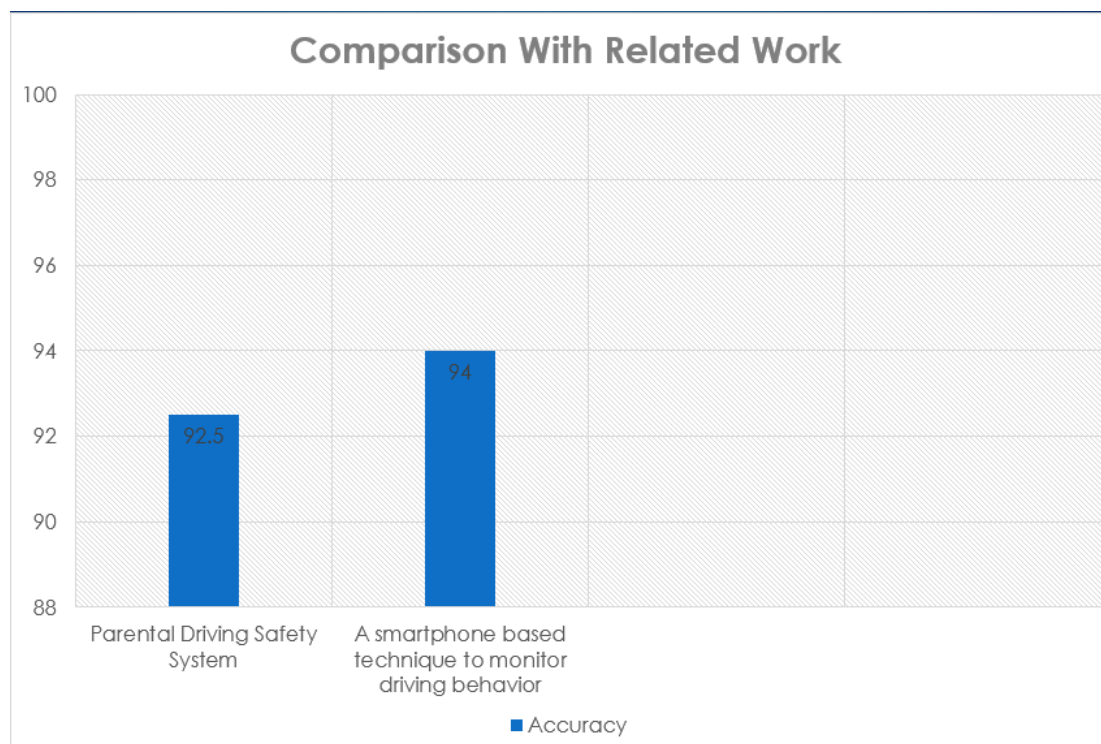
### 5.3.2 Task

we have calculated the average accuracy, recall and precision of our four classifiers. To compare the work, our classifier outputs close to [23]. Their classifier detects in a speed



rate 20 km/h but our classifier detects within range from 20km/h to 80/km/h and they classify three behaviours(Left Aggressive,Right Aggressive,Suddenly Stop) but we classify four behaviours (Left Aggressive,Right Aggressive,Suddenly Stop,Suddenly Acceleration). In terms of average accuracy achieving accuracy 92.5 percent, their output 94 accuracy but in terms of recall and precision this paper didn't mentioned.

### 5.3.3 Results



## Chapter 6

# Conclusion

We concluded from the sections above that our system has an average accuracy of 92.5 percent which is an acceptable number but more accuracy is needed through conducting more trials in controlled environments and with different circumstances. Our principle challenges we confronted while building up our purposed application that distinguishes driving behaviour is false positive classification and finally grouping the three features each other to build up the application.

### 6.1 Future directions

Our future work is add more detection features and increase the accuracy of our application. Moreover, this will help the parents and the companies to detect the driving behaviour of the driver , at the same time detect the drowsiness and detect if an accident occurred, then the application will give him the accurate rate.

# Bibliography

- [1] M., C. G. Q., Lopez, J. O., Pinilla, A. C. C. (2012). Driver behavior classification model based on an intelligent driving diagnosis system. 2012 15th International IEEE Conference on Intelligent Transportation Systems. doi: 10.1109/itsc.2012.6338727
- [2] Bouhoute, A., Oucheikh, R., Boubouh, K., Berrada, I. (2019). Advanced Driving Behavior Analytics for an Improved Safety Assessment and Driver Fingerprinting. IEEE Transactions on Intelligent Transportation Systems, 20(6), 2171–2184. doi: 10.1109/tits.2018.2864637
- [3] Burger, C., Orzechowski, P. F., Tas, O. S., Stiller, C. (2017). Rating cooperative driving: A scheme for behavior assessment. 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). doi: 10.1109/itsc.2017.8317794
- [4] Chakraborty, B., Nakano, K. (2016). Automatic detection of drivers awareness with cognitive task from driving behavior. 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC). doi: 10.1109/smc.2016.7844797
- [5] Chen, J., Wu, Y., Huang, H., Wu, B., Hou, G. (2018). Driving-Data-Driven Platform of Driving Behavior Spectrum for Vehicle Networks. 2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS). doi: 10.1109/hpcc/smartcity/dss.2018.00099
- [6] Imamura, T., Yamashita, H., Zhang, Z., Othman, M. R. B., Miyake, T. (2008). A study of classification for driver conditions using driving behaviors. 2008 IEEE International Conference on Systems, Man and Cybernetics. doi: 10.1109/icsmc.2008.4811499

- 
- [7] Kim, H., Yoon, D., Lee, S.-J., Kim, W., Park, C. H. (2018). A study on the cognitive workload characteristics according to the driving behavior in the urban road. 2018 International Conference on Electronics, Information, and Communication (ICEIC). doi: 10.23919/elinfocom.2018.8330624
- [8] Vaiana, R., Iuele, T., Astarita, V., Caruso, M. V., Tassitani, A., Zaffino, C., Giofrè, V. P. (2014). Driving Behavior and Traffic Safety: An Acceleration-Based Safety Evaluation Procedure for Smartphones. *Modern Applied Science*, 8(1). doi: 10.5539/mas.v8n1p88
- [9] Thajchayapong, S., Saiprasert, C., Charoensiriwath, C., Tanprasert, C. (2015). Severity assessment of anomalies using driving behaviour signals. 2015 International Conference on Connected Vehicles and Expo (ICCVE). doi: 10.1109/iccve.2015.83
- [10] Sunehra, D., Jhansi, K. (2015). Implementation of microcontroller based driver assistance and vehicle safety monitoring system. 2015 International Conference on Information Processing (ICIP). doi: 10.1109/infop.2015.7489420
- [11] Picot, A., Charbonnier, S., Caplier, A. (2012). On-Line Detection of Drowsiness Using Brain and Visual Information. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 42(3), 764–775. doi: 10.1109/tsmca.2011.2164242
- [12] Shinoda, T., Kato, M. (2006). A Pupil Diameter Measurement System for Accident Prevention. 2006 IEEE International Conference on Systems, Man and Cybernetics. doi: 10.1109/icsmc.2006.384964
- [13] Mahapatra, R. P., Kumar, K. V., Khurana, G., Mahajan, R. (2008). Ultra Sonic Sensor Based Blind Spot Accident Prevention System. 2008 International Conference on Advanced Computer Theory and Engineering. doi: 10.1109/icacte.2008.165
- [14] Patel, D. H., Sadatiya, P., Patel, D. K., Barot, P. (2019). IoT based Obligatory usage of Safety Equipment for Alcohol and Accident Detection. 2019 3rd International Conference on Electronics, Communication and Aerospace Technology (ICECA). doi: 10.1109/iceca.2019.8822104

- [15] Amin, M. S., Jalil, J., Reaz, M. B. I. (2012). Accident detection and reporting system using GPS, GPRS and GSM technology. 2012 International Conference on Informatics, Electronics Vision (ICIEV). doi: 10.1109/iciev.2012.6317382
- [16] Tripathi, S., Shetty, U., Hasnain, A., Hallikar, R. (2019). Cloud Based Intelligent Traffic System to Implement Traffic Rules Violation Detection and Accident Detection Units. 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI). doi: 10.1109/icoei.2019.8862694
- [17] Salvador, Stan, and Philip Chan. "Toward Accurate Dynamic Time Warping in Linear Time and Space." *Intelligent Data Analysis*, vol. 11, no. 5, 2007, pp. 561–580., doi:10.3233/ida-2007-11508.
- [18] Hui, Z., Yaohua, X., Lu, M., Jiansheng, F. (2014). Vision-based real-time traffic accident detection. *Proceeding of the 11th World Congress on Intelligent Control and Automation*. doi: 10.1109/wcica.2014.7052859
- [19] Chigan, D., Ma, X., Liu, G., Liu, J., Zhao, C. (2019). Car-Following Behavior of Coach Bus Based on Naturalistic Driving Experiments in Urban Roads. 2019 IEEE International Symposium on Circuits and Systems (ISCAS). doi: 10.1109/iscas.2019.8702201
- [20] Tsai, Y.-C., Lee, W.-H., Chou, C.-M. (2017). A safety driving assistance system by integrating in-vehicle dynamics and real-time traffic information. 2017 IEEE 8th International Conference on Awareness Science and Technology (ICAST). doi: 10.1109/icawst.2017.8256491
- [21] Martin, S., Tawari, A., Trivedi, M. M. (2014). Toward Privacy-Protecting Safety Systems for Naturalistic Driving Videos. *IEEE Transactions on Intelligent Transportation Systems*, 15(4), 1811–1822. doi: 10.1109/tits.2014.2308543
- [22] Vaiana, R., Iuele, T., Astarita, V., Caruso, M. V., Tassitani, A., Zaffino, C., Giofrè, V. P. (2014). Driving Behavior and Traffic Safety: An Acceleration-Based Safety Evaluation Procedure for Smartphones. *Modern Applied Science*, 8(1). doi: 10.5539/mas.v8n1p88
- [23] Singh, G., Bansal, D. and Sofat, S., 2017. A smartphone based technique to monitor driving behavior using DTW and crowdsensing. *Pervasive and Mobile Computing*.