Faculty Of Engineering

Computer Engineering Department

Helwan university

Driver’s Behaviour and Health Monitoring System

Supervised by

Dr. Hadeer Ahmed

Presented by

Abdelrahman Hassan Mostafa Ahmed

Mostafa Ahmed Zaki Ahmed Ahmed Taha

Nourhan Mohamed Mohamed Saad

Rana Mohamed Hussein Mohamed Bekheet

Reham Mohamed Ali Mohamed

Individual Contributions

Name	Contributions
- Abdelrahman Hassan Mostafa	1- Collecting datasets 2- Data Annotation 3- Build Face Recognition model 4- Build drowsiness detection model 5- Build Seatbelt detection model 6- Build distraction model 7- Evaluating and testing models 8- Worked on hardware by installing operating system and required libraries on Raspberry Pi 9- Test Cases
- Mostafa Ahmed Zaki	1- Collecting datasets 2- Worked on hardware by installing operating systems and required libraries on Raspberry Pi 3- Build Face Recognition model

	4- Build drowsiness detection model 5- Build Seatbelt detection model 6- Build distraction model 7- Evaluating and testing models 8- Use case diagram 9- Test cases
- Nourhan Mohamed Mohamed	1- Implementation of Mobile Application 2- Build Intro, Login &Register Screens using Firebase 3- User Verification Using Firebase Authentication 4- Mobile Application Design 5- Display sensor readings from Realtime Firebase 6- Collecting Dataset 7- Dataset Annotation 8- Sequence Diagram
- Rana Mohamed Hussein	1- Abstract 2- Part of Introduction. 3- Literature Review. 4- Enumerated Functional Requirements.

	5- Block Diagram. 6- MLX90614 Sensor. 7- Firebase Realtime database. 8- Mobile App Design. 9- Location in Mobile Application. 10- SMS Functionality in Mobile Application. 11- Voice Alert functionality in Mobile Application.
- Reham Mohamed Ali	1- Part of introduction 2- Class diagram 3- Programming Languages. 4- Software Tools. 5- Hardware Tools. 6- MAX30102 Sensor 7- Firebase Realtime database. 8- Use Case Diagram updated. 9- Call Functionality in Mobile Application. 10- Voice Alert functionality in Mobile Application. 11- Test Cases.

Abstract

Driver drowsiness, distraction, intoxication, aggravated state of mind, and others, have been a major safety issue worldwide. As a result, systems to monitor driver behaviours are becoming a priority for vehicles built soon. The need for driver monitoring is arising not only in the passenger vehicle segment but also in the commercial vehicle segment, wherein challenges are enormous. In addition, drivers work for long hours, many a time more than what is recommended or stipulated. In these cases, driver fatigue & drowsiness is the primary cause of accidents. There have been rules and logbooks to monitor working hours; however, they have not been the perfect solution to control the number of vehicle accidents caused due to driver fatigue. As a consequence, we have made a simple driver's behaviour and health monitoring system to detect the driver's drowsiness and distraction and take action before accidents happen. Also, monitor the driver's health state and send help to him if he has a health issue.

Table of Contents

Table of Figures

Chapter 1: Introduction

1.1. Problem Statement

We are familiar with the hazards of drinking and driving or even texting and driving, but many people underestimate the dangers of drowsy driving. Each year, drowsy driving accounts for about 100,000 crashes, 71,000 injuries, and 1,550 fatalities, according to the National Safety Council (NSC). Drowsy driving contributes to an estimated 9.5% of all crashes, according to AAA. However, the actual number may be much higher as it is difficult to determine whether a driver was drowsy at the time of a crash. In many cases, drowsy driving is as dangerous as driving while impaired by alcohol. The effects of drowsy driving are more severe than most people realize [1].

Not only drowsy driving that causes accidents but also driving while distracted. Distracted driving causes about 3,000 deaths per year: 3,142 deaths in 2019, 2,628 deaths in 2018 and 3,003 deaths in 2017, according to the NHTSA. Lastly, distracted driving causes about 920,000 total accidents per year (including fatalities and injuries): 938,000 accidents in 2018, 912,000 accidents in 2017, and 905,000 accidents in 2016.

Some estimates place the number of accidents related to a medical condition at about 20% of all accidents. These medical conditions might include health issues like heart attack, seizure disorders, diabetes, or problems with eyesight. Sometimes, these medical conditions are pre-existing, and drivers are well aware of them and the potential driving risk they pose, but in other cases, they happen suddenly and without prior warning or symptoms.

Medical emergencies typically result in:

- Extreme pain.

- Loss of consciousness.
- Changes in mental status.
- Dizziness.
- Choking.
- Loss of muscular control.

1.2. Motivation

Every fraction of seconds drowsiness and distraction can turn into dangerous and life-threatening accidents may lead to death also. To prevent this type of incidents, it is required to monitor driver's alertness continuously and when it detects drowsiness or distracted, the driver should be alerted. Through this we can reduce significant number of accidents and can save lives of people.

1.3. Objectives

- Our main objective is to enhance the driver alert systems used in cars nowadays by taking action to save the driver.
- Improve the security system to save cars from theft.
- Health monitoring and tracking system to help drivers especially who suffer from chronic disease.
- Ensure driver's safety by making sure he/she is wearing the seat belt.
- Allowing the system to contact the Egyptian Ambulance Organization directly if the driver encounters health problems.

1.4. Project Idea

Driver's Behaviour and Health monitoring system seeks to make the driver much safer and therefore passengers safer. This is done by the contribution of various systems that will be implemented together.

Our system aims to reduce the number of accidents by monitoring driver status of drowsiness or awake, wearing seatbelt or not, distracted, or attentive and monitoring data of sensors and depending on these readings the system will decide if it's a must to warn the driver and take an action or not.

The system includes mobile application to interface with the driver by showing his temperature and heartrate readings and voice alert him when readings seem fishy.

First system must check if the driver is the owner of the car by face recognition if not a SMS will be sent to the owner with photo.

In case of system detecting a problem with driver status like he/she is sleepy, isn't wearing seatbelt or distracted the app will alert the driver by voice command. The Application will send SMS to the company he works for or to his relative if he is sleepy with current location.

If the system detects a problem in driver's health the app will call the ambulance and send SMS to the company, he works for or his relative with the location.

1.5. Advantages

- Can reduce the risk of an accident.
- Make reports to transport services providers about the behavior of the driver.
- Ensure driver safety.
- The automatic emergency call system that activates in the event of an illness.

1.6. Disadvantages

There can be times, though, when ADAS can make wrong judgments:

- The sensors may make mistaken measurements.
- Drowsiness model can send misleading signals.
- Raspberry pi is slow.

Chapter 2: Literature Review

Several manufacturers, including Audi, Mercedes, and Volvo, currently offer drowsiness detection systems that monitor a vehicle’s movements, such as steering wheel angle, lane deviation, time driven and road conditions. When drowsiness is detected, drivers are typically warned with a sound and the appearance of a coffee cup icon. Car manufacturers are working on developing systems that can take real actions and not just warn the driver [1].

A Driver Monitoring System was developed by [2] to monitor the driver’s behaviour through pressure sensors placed in the steering wheel, gearshift, and armrest. They transmit data regarding the driver's position if the positioning of both hands is on the steering wheel and if the left hand is on the steering wheel and the right hand is on the gear selector or on the armrest. The light sensor has a very important part in the system, it detects whether the vehicle is driving at night or during the day. Driving at night is much more exhausting and driving time is lower than driving if it is daytime.

Moreover, A Driver Drowsiness Detection System was created by [3] which focused on the detection of the driver’s drowsiness through using machine learning algorithms measuring the distance between the eyelids of each eye. if the aspect ratio value is less than 0.25 an alarm is sounded, and user is warned. Meanwhile the latest paper published on the Driver Monitoring topic included a project developed by [4] which integrated the detection of the driver drowsiness through machine learning algorithms alongside with his/her health monitoring through heartbeat and alcohol sensors. The moment drowsiness is detected the user is being alarmed through a buzzer.

Chapter 3: Analysis and Requirements

3.1. System Requirements.

3.1.1. Enumerated Functional Requirements.

Requirement Number	Requirement brief description
REQ-1	Register users' data to the system • **User's Name.** • **User's Age.** • **User's Photo.** • **User's Relative Phone number.**
REQ-2	Allowing all registered users to view their heartbeat, Oxygen saturation and Temperature values in Mobile Application.
REQ-3	Measure driver's Temperature.
REQ-4	Measure driver's Heartrate and Oxygen Saturation.
REQ-5	Verify Driver.
REQ-6	Detect driver's drowsiness.
REQ-7	Detect driver's distraction.
REQ-8	Detect driver's seatbelt
REQ-9	Detect driver's drowsiness
REQ-10	Upload driver's vital Status to Realtime database.
REQ-11	Upload driver's drowsiness, seatbelt, distraction status to Realtime database.
REQ-12	Detect driver's location

REQ-13	Upload driver's location to Realtime database.
REQ-14	Send SMS to driver's relative if drowsy.
REQ-15	Call driver's relative if drowsy and vital status are severe.
REQ-16	Voice alert driver when drowsy, distracted or not wearing seatbelt.
REQ-17	Upload unverified driver photo to Realtime database.
REQ-18	Notify driver about unverified user and send photo to relative.
REQ-19	Upload user logs to Realtime database.

3.1.2 Enumerated Non-functional Requirements.

Requirement Number	Requirement brief description
REQ-17	Performance: system operates its function in small amount of time
REQ-18	User interface: The user interface of the mobile application is friendly which is easy to use.
REQ-19	Supportability: the application supports all android versions.

3.2. Functional Requirements Specification

3.2.1. Stakeholders.

1. Drivers

2. Riders

3. Investors

4. Mobile operating systems providers

3.2.2. Actors and goals

- **Software Engineers**

Gain experience in building software systems and its development

- **Administrators**

To be able to reach all the data concerning the drivers easily.

- **Drivers**

To be able to track their behaviour and mentoring health during driving

3.2.3. Use Cases

I) Use case Description

Use Case Name	Description	Actors	Conditions
Login	The driver will login with his email and password	Driver's Mobile App	Email and password must be valid and there are stored in firebase
Setup Profile	If the driver doesn't have an account, then he will register with photo, his name, email, password, his number and relative number	Driver's Mobile App	Email and password must be valid
Capture Real Time Video	Camera capture frames	Camera	Frames are detected and stored in firebase

Recognize Face	Model compare between faces in firebase and faces detected by camera	System	Identify the driver to start the system if the driver is not the owner a SMS will be sent to the owner with the photo
Seatbelt Absence Detection	Model detected if there is a Seatbelt or not	System	If there's no Seatbelt. A voice message alerts the driver
Distraction Detection	Model detect if the driver is distracted using Mobile Phone or eating.	System	If the driver is distracted a voice message alerts the driver. And a record will be sent to firebase
Drowsiness Detection	Model detect if the driver is drowsy or yawning	System	If the driver is getting drowsy a voice message alerts the driver and SMS will send to company
Measure Health Parameters	The system will upload sensors readings to firebase to show to the driver	System	If there is issue in sensors readings, a SMS will be sent to the company
Voice Alert	The mobile app makes voice alerts at some conditions	Driver's Mobile App	Voice alert will be sent if system detects drowsiness, seatbelt absence, distraction or any health problem
Send Message	The mobile app sends message at some conditions	Driver's Mobile App	A message will be sent if system detects drowsiness or

			any health problem
Make a Call	The mobile app makes voice call at some conditions	Driver's Mobile App	A message will be sent if system detects drowsiness and any health problem

II) Use case Diagram

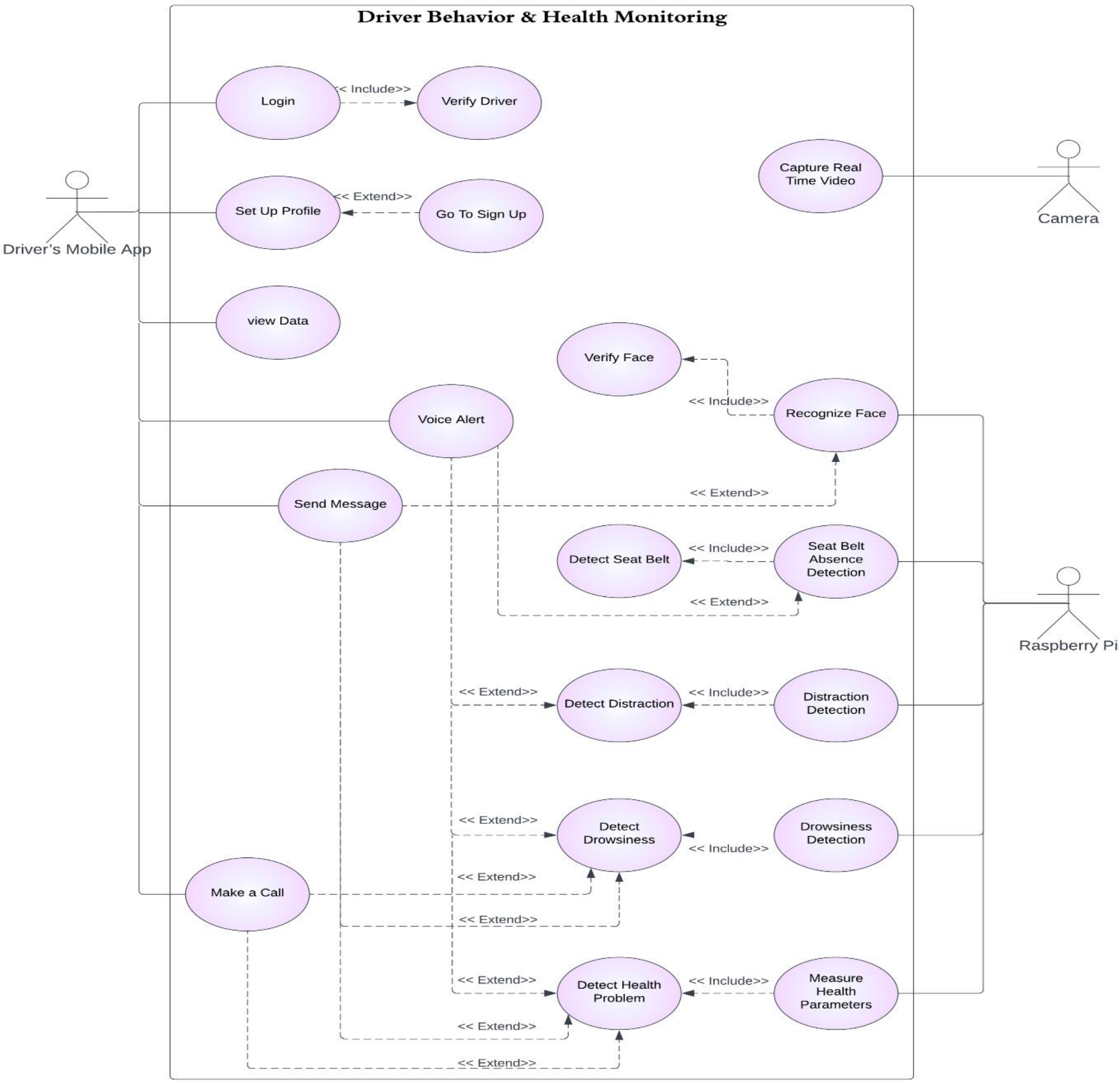


Figure 3. 1: Use Case Diagram

Chapter 4: Design

4.1. Block Diagram

Our System begins with machine learning models running on the raspberry pi for driver's face recognition, distraction detection, drowsiness detection and seatbelt detection running alongside with algorithm created to read sensors data and uploading all changes to Firebase Realtime database. The system starts by recognizing the face of the driver as to when the driver is recognized the system will start else it won't. Moreover, the User interface which is in the form of Mobile Application used by the driver to view his/her vital status which are Heart rate, Oxygen level and Temperature.

When the driver is caught drowsy, the firebase gets updated and a message is sent to the relative with the driver's location, and status, also a voice alert is triggered to alert the driver. In case the readings of the driver's status are severe not only a message is sent but also a call is made to the relative to notify the driver's relative. In other cases where the driver is distracted or not wearing a seatbelt a voice alert is also triggered to warn the driver to focus on the road, as well as the data base being updated. The system is as shown in figure 4.1.

Chapter 4: Design

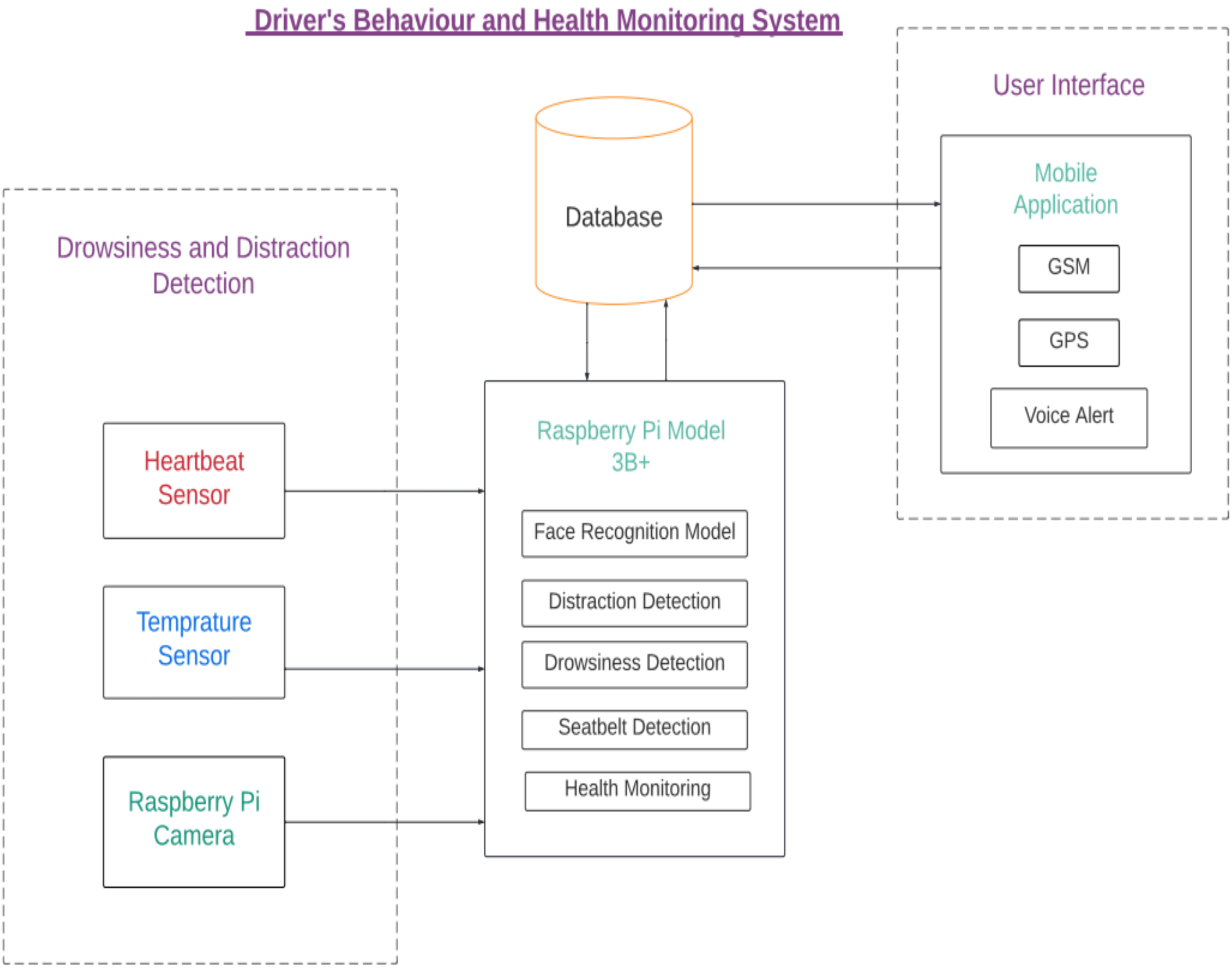


Figure 4. 1: Block Diagram

Chapter 4: Design

4.2. Class Diagram

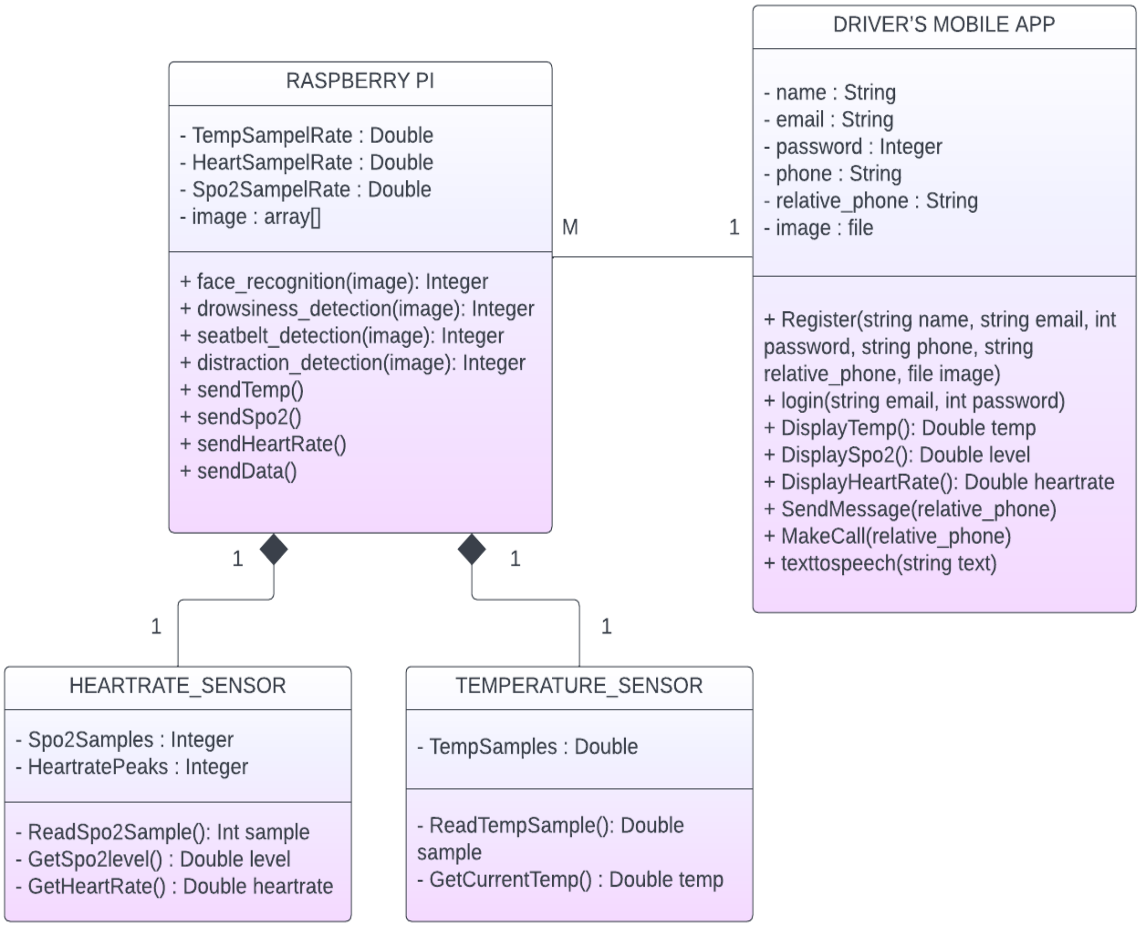


Figure 4. 2: Class Diagram

The Class diagram of our project as presented in Figure 4.2 consists of the following classes and relationships:

Class Raspberry pi: Raspberry pi in our system is responsible for run models of face detection, drowsiness detection, seatbelt detection, distraction detection and calculate data of sensors and send it to firebase to show in mobile app.

Class driver's mobile app: driver's mobile app in our system is the user interface it enables the driver to register with name, email, password, phone, relative phone and his photo and take the sensors readings from firebase and shows it to the driver. It also makes voice alert, sends messages and makes a call.

Class heartrate_sensor: it read heartrate pulses and spo2 level of the driver.

Class temperature_sensor: it read the temperature of the driver.

Association relationship: there is an association relationship between raspberry pi and the mobile application. The raspberry pi send data to firebase and the mobile app take this data from firebase to show it to the driver. One application can used for many cars.

Composition relationship: there is a composition relationship between raspberry pi and the heartrate sensor and temperature sensor. The raspberry pi calculates data of the sensors and send it to firebase. Every car has a one heartrate sensor and one temperature sensor.

4.3. Sequence Diagram

A detailed description of the system’s sequence diagram is presented in figure 4.3.

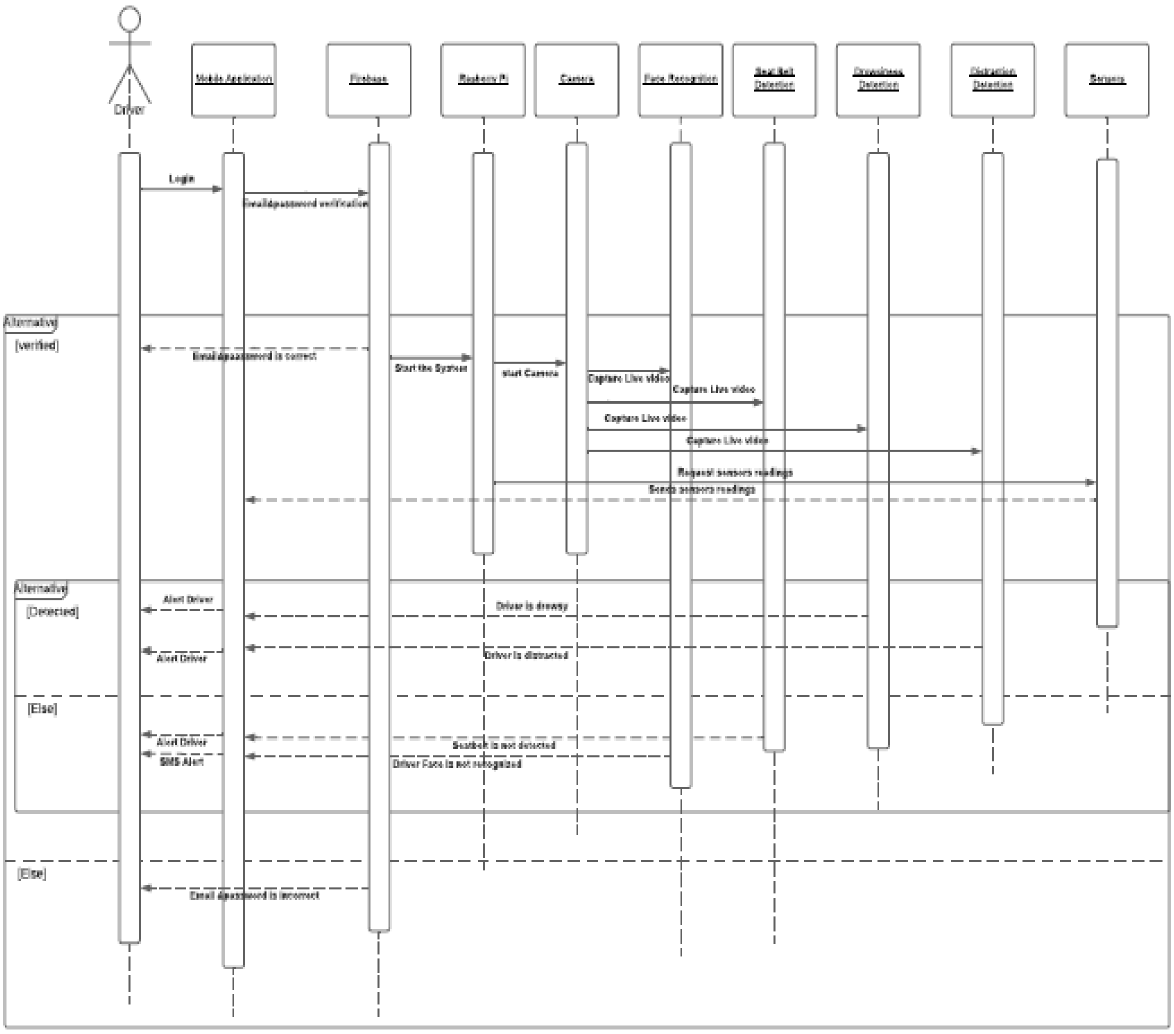


Figure 4. 3: Sequence Diagram

Chapter 5: Tools and Technologies

5.1. Embedded Systems

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

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

An embedded system has three components:

- It has hardware.
- It has application software.
- It has Real Time Operating system (RTOS) that supervises the application software and provide mechanism to let the processor run a process as per scheduling by following a plan to control the latencies. RTOS defines the way the system works. It sets the rules during the execution of application program. A small-scale embedded system may not have RTOS [7].

5.2. Embedded Linux

Embedded Linux is a type of Linux operating system/kernel that was designed to be installed and used in embedded devices or systems. Although it uses the same kernel, embedded Linux is quite different from the standard operating system. First, it gets tailored for embedded systems and, therefore, is much smaller in size, requires less processing power, and has minimal features. The

Linux kernel is modified and optimized as an embedded Linux version. Such a Linux instance can only run applications created specifically for the device.

You may also have heard of bare-metal tasking kernels (without an operating system) for microcontrollers, such as FreeRTOS, Azure RTOS, RT-Thread, µC/OS, but these choices lack memory management and protection which are critical to building modern applications. Because of this, you may be looking for a more comprehensive solution.

Linux has Extensive Compatibility

Over the years, Linux has grown to support a large variety of CPU architectures, including x86, ARM and PowerPC etc. So, no matter what kind of hardware you are working with, there is a good chance that a Linux kernel already exists for it.

Linux supports nearly all the programming languages and utilities that you need for your embedded system development endeavours. With Linux, you are not restricted to any specific software. If you are dissatisfied with anything, there is a good chance that an alternative is available – or you can be the first in the community to develop it!

Linux is Open Source

Linux, as an open-source kernel, allows you to leverage on the work of thousands of developers across the world. By tweaking or directly implementing existing packages, you can get your own applications up and running far more easily and quickly.

For these reasons, and thanks to its versatility, embedded Linux has become very popular among the embedded systems engineers. Many consumer electronics devices such as phones, smart tablets, digital storage devices, personal video recorders, cameras, wearables, and many others typically are developed with Linux. It gets used in cars software, and many other examples such as network equipment, machine control, industrial automation, navigation equipment, spacecraft flight software, and medical instruments in general [8].

5.3. Machine Learning

Machine learning is an application of AI that enables systems to learn and improve from experience without being explicitly programmed. Machine learning focuses on developing computer programs that can access data and use it to learn for themselves.

How does it work?

UC Berkeley (link resides outside IBM) breaks out the learning system of a machine learning algorithm into three main parts.

1. **A Decision Process**: In general, machine learning algorithms are used to make a prediction or classification. Based on some input data, which can be labeled or unlabeled, your algorithm will produce an estimate about a pattern in the data.

2. **An Error Function**: An error function evaluates the prediction of the model. If there are known examples, an error function can make a comparison to assess the accuracy of the model.

3. **A Model Optimization Process**: If the model can fit better to the data points in the training set, then weights are adjusted to reduce the discrepancy between the known example and the model estimate. The algorithm will repeat this "evaluate and optimize" process, updating weights autonomously until a threshold of accuracy has been met.

Classification of Machine Learning:

Chapter 5: Tools and Technology

Machine learning implementations are classified into four major categories, depending on the nature of the learning “signal” or “response” available to a learning system which are as follows:

Supervised learning algorithms are trained using labeled examples, such as an input where the desired output is known. For example, a piece of equipment could have data points labeled either “F” (failed) or “R” (runs). The learning algorithm receives a set of inputs along with the corresponding correct outputs, and the algorithm learns by comparing its actual output with correct outputs to find errors. It then modifies the model accordingly. Through methods like classification, regression, prediction and gradient boosting, supervised learning uses patterns to predict the values of the label on additional unlabeled data. Supervised learning is commonly used in applications where historical data predicts likely future events. For example, it can anticipate when credit card transactions are likely to be fraudulent or which insurance customer is likely to file a claim.

Unsupervised learning is used against data that has no historical labels. The system is not told the "right answer." The algorithm must figure out what is being shown. The goal is to explore the data and find some structure within. Unsupervised learning works well on transactional data. For example, it can identify segments of customers with similar attributes who can then be treated similarly in marketing campaigns. Or it can find the main attributes that separate customer segments from each other. Popular techniques include self-organizing maps, nearest-neighbor mapping, k-means clustering and singular value decomposition. These algorithms are also used to segment text topics, recommend items and identify data outliers.

Semisupervised learning is used for the same applications as supervised learning. But it uses both labeled and unlabeled data for training – typically a small amount of labeled data with a large amount of unlabeled data (because unlabeled data is less expensive and takes less effort to acquire). This type of learning can be used with methods such as classification, regression and prediction. Semisupervised learning is useful when the cost associated with labeling is too high to allow for a fully labeled training process. Early examples of this include identifying a person's face on a web cam.

Reinforcement learning is often used for robotics, gaming and navigation. With reinforcement learning, the algorithm discovers through trial and error which actions yield the greatest rewards. This type of learning has three primary components: the agent (the learner or decision maker), the environment (everything the agent interacts with) and actions (what the agent can do). The objective is for the agent to choose actions that maximize the expected reward over a given amount of time. The agent will reach the goal much faster by following a good policy. So the goal in reinforcement learning is to learn the best policy.

- ## Common machine learning algorithms:

A number of machine learning algorithms are commonly used. These include:

- **Neural networks:** Neural networks simulate the way the human brain works, with a huge number of linked processing nodes. Neural networks are good at recognizing patterns and play an important role in applications including natural language translation, image recognition, speech recognition, and image creation.

- **Linear regression:** This algorithm is used to predict numerical values, based on a linear relationship between different values. For example, the technique could be used to predict house prices based on historical data for the area.

- **Logistic regression:** This supervised learning algorithm makes predictions for categorical response variables, such as “yes/no” answers to questions. It can be used for applications such as classifying spam and quality control on a production line.

- **Clustering:** Using unsupervised learning, clustering algorithms can identify patterns in data so that it can be grouped. Computers can help data scientists by identifying differences between data items that humans have overlooked.

- **Decision trees:** Decision trees can be used for both predicting numerical values (regression) and classifying data into categories. Decision trees use a branching sequence of linked decisions that can be represented with a tree diagram. One of the advantages of decision trees is that they are easy to validate and audit, unlike the black box of the neural network.

- **Random forests:** In a random forest, the machine learning algorithm predicts a value or category by combining the results from a number of decision trees.

- **Real-world machine learning use cases:**

Here is one example of machine learning you might encounter every day:

Computer vision: This AI technology enables computers to derive meaningful information from digital images, videos, and other visual inputs, and then take the appropriate action. Powered by convolutional neural networks, computer vision has applications in photo tagging on social media, radiology imaging in healthcare, and self-driving cars in the automotive industry.

- **Why is machine learning important?**

Machine learning explores the analysis and construction of algorithms that can learn from and make predictions on data. ML has proven valuable because it can solve problems at a speed and scale that cannot be duplicated by the human mind alone. With massive amounts of computational ability behind a single task or multiple specific tasks, machines can be trained to identify patterns in and relationships between input data and automate routine processes.

Resurging interest in machine learning is due to the same factors that have made data mining and Bayesian analysis more popular than ever. Things like growing volumes and varieties of available data, computational processing that is cheaper and more powerful, and affordable data storage.

All of these things mean it's possible to quickly and automatically produce models that can analyze bigger, more complex data and deliver faster, more accurate results – even on a very large scale. And by building precise models, an organization has a better chance of identifying profitable opportunities – or avoiding unknown risks [9].

5.4. Programming Languages

- **Python**

Python is a general-purpose interpreted and high-level programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small- and large-scale projects [10].

- **Dart Programming Language**

Dart is a programming language designed for client development, such as for the web and mobile apps. It is developed by Google and can also be used to build server and desktop applications.

Dart is an object-oriented, class-based, garbage-collected language with C-style syntax.[9] Dart can compile to either native code or JavaScript. It supports interfaces, mixins, abstract classes, reified generics, and type inference [11].

5.5. Software Tools

- **Android Studio**

Android Studio is the official Integrated Development Environment (IDE) for Android app development it provides the fastest tools for building apps on every type of Android device and offers even more features that enhance your productivity when building Android apps [12].

- **Thonny IDE**

Thonny is a free Python Integrated Development Environment (IDE) that was especially designed with the beginner Pythonista in mind. Specifically, it has a built-in debugger that can help when you run into nasty bugs, and it offers the ability to do step through expression evaluation, among other really awesome features [13].

- **Jupyter Lab**

Jupyter Lab is the latest web-based interactive development environment for notebooks, code, and data. Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality [14].

- **VNC viewer**

VNC Viewer gathers your input (mouse, keyboard, or touch) and sends it for VNC Server to inject and achieve remote control. You need a VNC Server for the remote computer you want to control, and a VNC Viewer for the computer or mobile device you want to control from. It is used to view the raspberry pi on the laptop screen as an easier way rather than connecting the raspberry pi to a dedicated screen [15].

5.6. Hardware Tools

- **Raspberry pi 3 model B+**

The Raspberry Pi 3 Model B+ is the latest product in the Raspberry Pi 3 range, boasting a 64-bit quad core processor running at 1.4GHz, faster Ethernet. The Raspberry Pi 3 Model B+ maintains the same mechanical footprint as both the Raspberry Pi 2 Model B and the Raspberry Pi 3 Model B.

Figure 5. 1: Raspberry Pi

Specifications:

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- 1GB LPDDR2 SDRAM
- dual-band 2.4GHz and 5GHz wireless LAN, Bluetooth 4.2, BLE
- Extended 40-pin GPIO header
- 4 USB 2.0 ports
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display

- Micro SD port for loading your operating system and storing data
- Power-over-Ethernet (PoE) support (requires separate PoE HAT) [16].

- **Raspberry pi camera v2**

The v2 Camera Module has a Sony IMX219 8-megapixel sensor (compared to the 5-megapixel Omni Vision OV5647 sensor of the original camera). It can be used to take high-definition video, as well as stills photographs. It's easy to use for beginners but has plenty to offer advanced users if you're looking to expand your knowledge. You can use the libraries we bundle with the camera to create effects.

Figure 5. 2: Raspberry Pi Camera

It supports 1080p30, 720p60 and VGA90 video modes, as well as still capture. It attaches via a 15cm ribbon cable to the CSI port on the Raspberry Pi [17].

- **MAX30102**

The MAX30102 as shown in figure 5.4 can be used as an oximeter by measuring the oxygen concentration in the blood (SpO2 percentage) and also can be used as heart rate sensor (beats per minute). The device has two LEDs, one emitting red light, another emitting infrared light. For pulse rate, only the infrared light is needed. Both the red light and infrared light is used to measure oxygen levels in the blood and also has a photodetector, optimized optics, and low-noise analog signal processing to detect pulse oximetry (SpO2) and heart rate (HR) signals. The MAX30102 features a 1.8V power supply and a separate 5.0V power supply for internal LEDs for heart rate and blood oxygen acquisition in wearable devices, worn on the fingers, earlobe, and wrist.

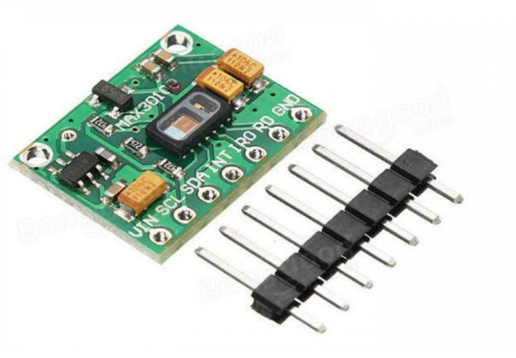


Figure 5. 3: MAX30102

Pin Description: VIN: main power input terminal 1.8-5V 3-bit pad: Select the pull-up level of the bus, depending on the pin master voltage, select 1.8v or 3_3v (this terminal contains 3.3V and above) SCL: the clock connected to the I2C bus; SDA: data connected to the I2C bus; INT : Interrupt pin of the MAX30102 chip The MAX30102 can be programmed to generate an interrupt for each pulse. This line is open drain, so it is pulled HIGH by the onboard resistor. When an interrupt occurs the INT pin goes LOW and stays LOW until the interrupt is cleared; RD: RED LED ground terminal of MAX30102 chip, generally not connected; IRD: The IR LED The MAX30102 integrates an LED driver to drive LED pulses for SpO2 and HR measurements is generally not connected; GND: Ground wire.

The MAX30102 embeds a FIFO buffer for storing data samples. The FIFO has a 32-sample memory bank, which means it can hold up to 32 SpO2 and heart rate samples. The FIFO buffer can offload the microcontroller from reading each new data sample from the sensor, thereby saving system power.

Features and Benefits:

- Working voltage: 1.8-5V
- Interface: I2C Interface
- Fast Data Output Capability (High Sample Rates)
- Ultra-Low-Power Operation Increases Battery Life
- Low-Power Heart-Rate Monitor & Ultra-Low Shutdown Current
- This low power consumption allows implementation in battery powered devices such as handsets, wearables or smart watches.

How does it work?

The MAX30102 works as shown in figure 5.5 by shining both lights onto the finger or earlobe (or essentially anywhere where the skin isn't too thick, so both lights can easily penetrate the

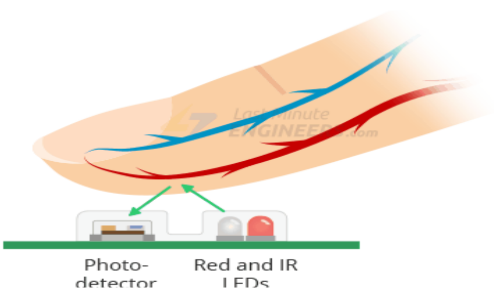


Figure 5. 4: MAX30102 Operation

tissue) and measuring the amount of reflected light using a photodetector. This method of pulse detection through light is called Photoplethysmogram.

Heart Rate Measurement

The oxygenated haemoglobin (HbO2) in the arterial blood has the characteristic of absorbing IR light. The redder the blood (the higher the haemoglobin), the more IR light is absorbed. As the blood is pumped through the finger with each heartbeat, the amount of reflected light changes, creating a changing waveform at the output of the photodetector. As you continue to shine light and take photodetector readings, you quickly start to get a heartbeat (HR) pulse reading.

Pulse Oximetry

Pulse oximetry is based on the principle that the amount of RED and IR light absorbed varies depending on the amount of oxygen in your blood. The following graph is the absorption-spectrum of oxygenated haemoglobin (HbO2) and deoxygenated haemoglobin (Hb).

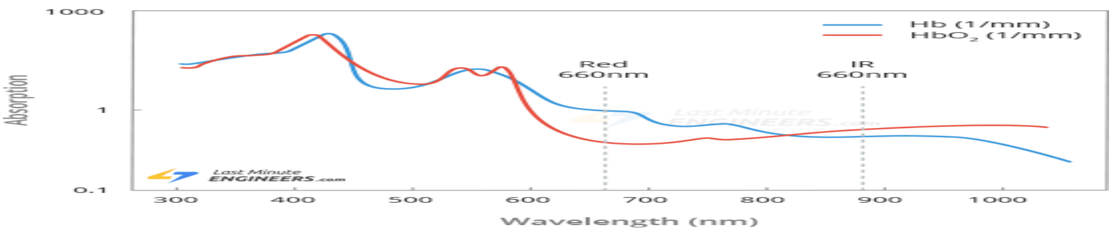


Figure 5. 5 Blood Absorption of IR and Red light.

As you can see from the graph shown in figure 5.6, deoxygenated blood absorbs more RED light (660nm), while oxygenated blood absorbs more IR light (880nm). By measuring the ratio of IR and RED light received by the photodetector, the oxygen level (SpO2) in the blood is calculated. Normal pulse and spo2 rates are shown in figure 5.7.

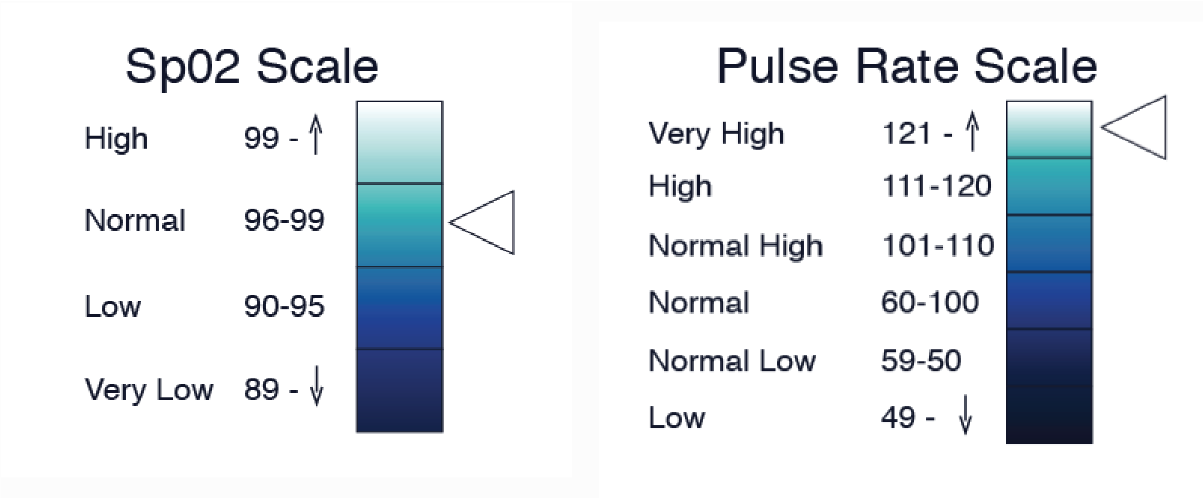
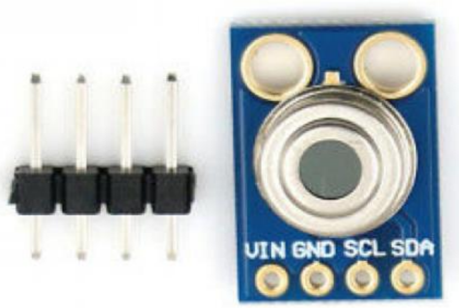


Figure 5. 6: Normal Pulse rate

- **MLX90614**

The MLX90614 shown in figure 5.8 is an Infra-Red thermometer for non-contact temperature measurements. Both the IR sensitive thermopile detector chip and the signal conditioning ASSP are integrated in the same TO-39 can. Thanks to its low noise amplifier, 17-bit ADC and powerful DSP unit, a high accuracy and resolution of the thermometer is achieved. The thermometer comes factory calibrated with a digital PWM and SMBus (System Management Bus) output. As a standard, the 10-bit PWM is configured to continuously transmit the measured temperature in range of -20…120 ˚C, with an output resolution of 0.14 ˚C and the POR default is SMBus.

Figure 5. 7: MLX90614 Sensor



The chip supports a 2 wires serial protocol, build with pins PWM/SDA and SCL.

• SCL – digital input, used as the clock for SMBus compatible communication.

This pin has the auxiliary function for building an external voltage regulator. When the external voltage regulator is used, the 2-wire protocol is available only if the power supply regulator is overdriven. • PWM/SDA – Digital input/output, used for both the PWM output of the measured object temperature(s) or the digital input/output for the SMBus. The pin can be programmed in EEPROM to operate as Push/Pull or open drain NMOS (open drain NMOS is factory default). In SMBus mode SDA is forced to open drain NMOS I/O, push-pull selection bit defines PWM/Thermal relay operation.

Features and Benefits:

- Small size, low cost
- Easy to integrate
- Factory calibrated in wide temperature range: -40…+125 ˚C for sensor temperature and -70…+380 ˚C for object temperature.
- High accuracy of 0.5°C
- Measurement resolution of 0.02°C

How does it work?

Per the Stefan-Boltzmann law, any object that isn't below absolute zero (0°K) emits (non-human-eye-visible) light in the infrared spectrum that is directly proportional to its temperature. The special infrared thermopile inside the MLX90614 senses how much infrared energy is being emitted by materials in its field of view and produces an electrical signal proportional to that. That voltage produced by the thermopile is picked up by the application processor's 17-bit ADC, then conditioned before being passed over to a microcontroller.

The MLX90614 produces two temperature measurements: an object and an ambient reading. The **object temperature** is the non-contact measurement you'd expect from the sensor, while the **ambient temperature** measures the temperature on the die of the sensor.

5.4. Communication Protocols

- ### I2C Communication Protocol

I2C stands for **Inter-Integrated Circuit.** It is a bus interface connection protocol incorporated into devices for serial communication. It was originally designed by Philips Semiconductor in 1982. Recently, it is a widely used protocol for short-distance communication. It is also known as Two Wired Interface (TWI). It uses only 2 bi-directional open-drain lines for data communication called SDA and SCL. Both these lines are pulled high. The I2C message as shown in figure 5.9 consists of:

Serial Data (SDA) – Transfer of data takes place through this pin. **Serial Clock (SCL) –** It carries the clock signal.

I2C is synchronous, so the output of bits is synchronized to the sampling of bits by a clock signal shared between the master and the slave. The clock signal is always controlled by the master. Each data bit transferred on SDA line is synchronized by a high to the low pulse of each clock on the SCL line.

Start Condition: The SDA line switches from a high voltage level to a low voltage level *before* the SCL line switches from high to low.

Stop Condition: The SDA line switches from a low voltage level to a high voltage level *after* the SCL line switches from low to high.

Address Frame: A 7- or 10-bit sequence unique to each slave that identifies the slave when the master wants to talk to it.

Read/Write Bit: A single bit specifying whether the master is sending data to the slave (low voltage level) or requesting data from it (high voltage level).

ACK/NACK Bit: Each frame in a message is followed by an acknowledge/no-acknowledge bit. If an address frame or data frame was successfully received, an ACK bit is returned to the sender from the receiving device.

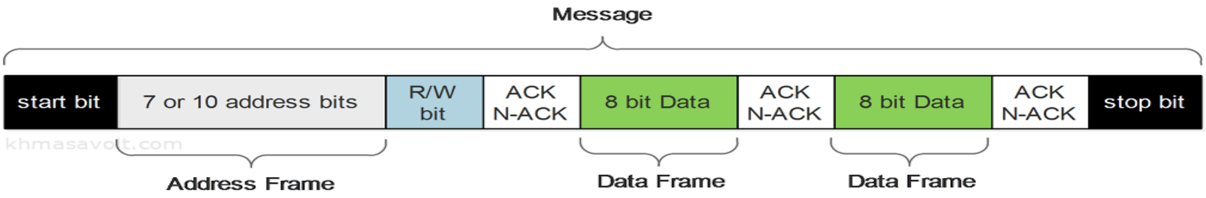


Figure 5. 8: I2C Message

Steps of I2C Data Transmission:

1. The master sends the start condition to every connected slave by switching the SDA line from a high voltage level to a low voltage level *before* switching the SCL line from high to low as shown in figure 5.10.

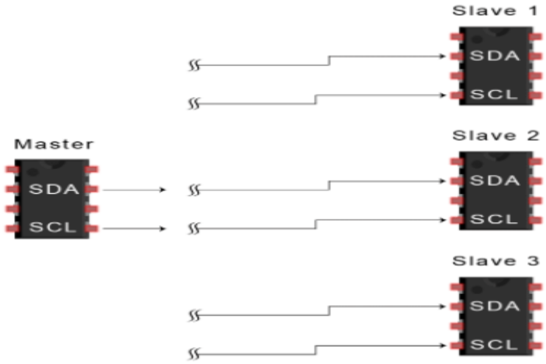


Figure 5. 9: I2C Step 1

2. The master sends each slave the 7- or 10-bit address of the slave it wants to communicate with, along with the read/write bit as shown in figure 5.11.

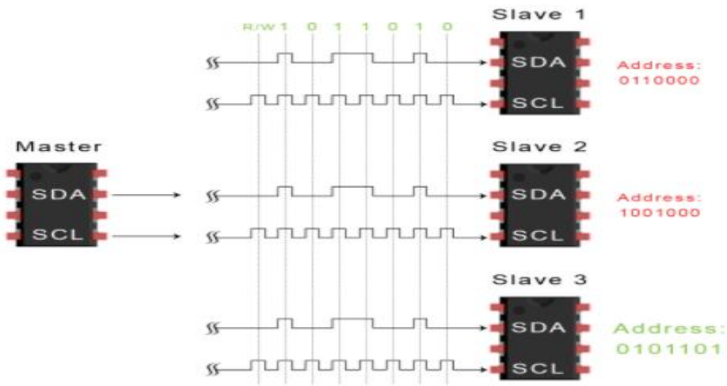


Figure 5. 10: I2C Step 2

3. Each slave compares the address sent from the master to its own address. If the address matches, the slave returns an ACK bit by pulling the SDA line low for one bit. If the address from the master does not match the slave's own address, the slave leaves the SDA line high as shown in figure 5.12.

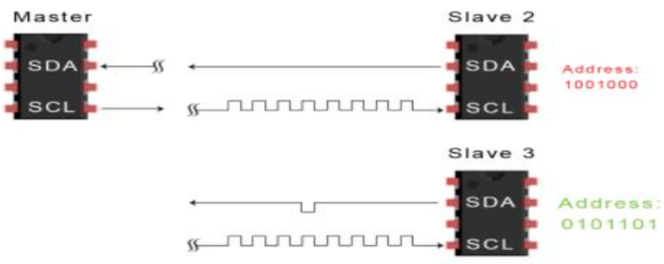


Figure 5. 11: I2C Step 3

4. The master sends or receives the data frame:

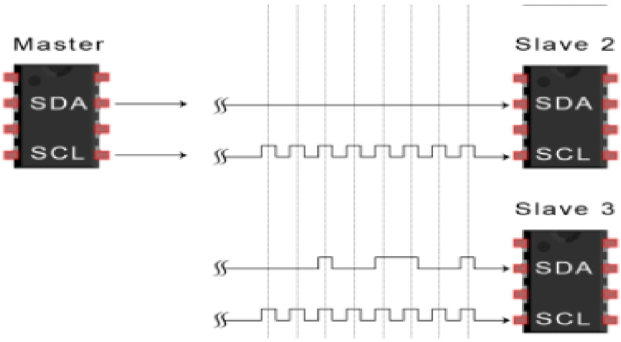


Figure 5. 12: I2C Step 4

5. After each data frame has been transferred, the receiving device returns another ACK bit to the sender to acknowledge successful receipt of the frame:

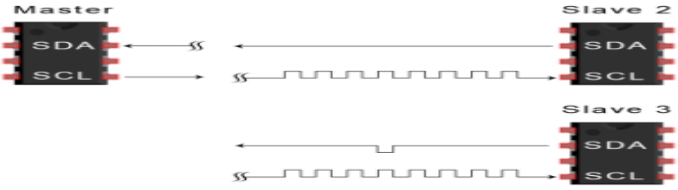


Figure 5. 13: I2C Step 5

6. To stop the data transmission, the master sends a stop condition to the slave by switching SCL high before switching SDA high as shown in figure 5.15.

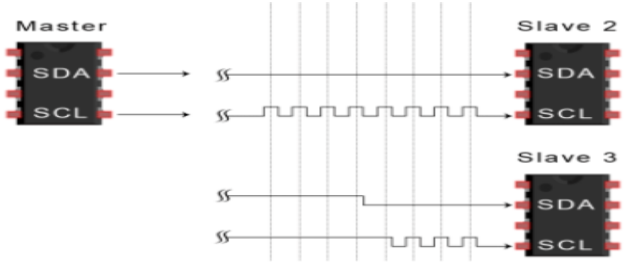


Figure 5. 14: I2C Step 6

5.7. Firebase

Firebase is a Google-backed application development software that enables developers to develop iOS, Android and Web apps. Firebase provides tools for tracking analytics, reporting and fixing app crashes, creating marketing and product experiment. It is categorized as a NoSQL database program, which stores data in JSON-like documents. It offers several services, including Authentication, Realtime database, Test lab, Notifications, Crashlytics.

- ### Realtime Database

Firebase Realtime Database is a cloud-hosted NoSQL database Data is stored as JSON and synchronized in real time to every connected client. When you build cross-platform apps with our Apple platforms, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

Realtime syncing makes it easy for your users to access their data from any device: web or mobile, and it helps your users collaborate with one another.

When your users go offline, the Realtime Database SDKs use local cache on the device to serve and store changes. When the device comes online, the local data is automatically synchronized.

The Realtime Database integrates with Firebase Authentication to provide simple and intuitive authentication for developers.

- **How does it work?**

The Firebase Realtime Database lets you build rich, collaborative applications by allowing secure access to the database directly from client-side code. Data is persisted locally, and even while offline, realtime events continue to fire, giving the end user a responsive experience. When the device regains connection, the Realtime Database synchronizes the local data changes with the remote updates that occurred while the client was offline, merging any conflicts automatically.

The Realtime Database provides a flexible, expression-based rules language, called Firebase Realtime Database Security Rules, to define how your data should be structured and when data can be read from or written to. When integrated with Firebase Authentication, developers can define who has access to what data, and how they can access it.

- **Why we use Firebase Realtime Database?**

The Realtime Database is a NoSQL database and as such has different optimizations and functionality compared to a relational database. The Realtime Database API is designed to only allow operations that can be executed quickly. This enables you to build a great realtime experience that can serve millions of users without compromising on responsiveness. Because of this, it is important to think about how users need to access your data and then structure it accordingly.

- **Firebase Advantages:**

1. Free to start
2. Development speed
3. Powered by Google
4. It’s serverless

5. Security

- **Firebase Limitations:**

1. It's not Open-Source.
2. Firebase does not work in many countries.
3. Only NoSQL databases are available.
4. Not all services are free to start.
5. It's not cheap and pricing is difficult to forecast.

5.11 Flutter

There are lots of benefits of using Flutter for your App development for small and large-scale businesses. It clearly stands out as a popular cross-platform application development framework by many mobile app development service providers. Here we list some key points of the advantages of Flutter.

- **Benefit #1 Open Source**

Flutter is an open-source code software development toolkit from Google. It provides easy posting of issues and access to documentation from open developer forums. It increases the efficiency and productivity of the coder and results in less time and cost for the entire project.

- **Benefit #2 Single Codebase**

Since it is a cross-platform framework, it allows programmers to write code once and they can use it on multiple platforms. This means that a single version of an application runs on both iOS and Android. This saves a lot of time and effort in writing code for different platforms, as with native frameworks. This consequently saves a lot on the overall cost of developing and launching the app.

- **Benefit #3 Dart As Programming Language**

Flutter uses Dart as an object-oriented programming language to create apps. The prominent features of Dart include a rich standard library, garbage collection, strong typing, generics, and async-awaits. Dart is like Java and uses a lot of the popular features of other languages too.

- **Benefit #4 Hot Reload and Development**

This is a unique feature to Flutter, where we can see changes made to code instantly. Any updates are available to both the designers and developers in a matter of seconds. They do not have to wait for updates and can continue using the framework to develop other features with no interruptions.

- **Benefit #5 Use of Custom Widgets**

Flutter offers a myriad of widgets to help developers in their creation process. It makes designing a basic user interface much easier and faster.

Benefit #6 Create Apps for Mobile, Desktop, and Web

You can create applications for six different platforms simultaneously. This includes different operating systems like Android, iOS, macOS, Windows, Linux, and Web.

- **Benefit #7 Requires Less Testing**

Normally testing would require checking on compatibility on different platforms. With Flutter, apps use a single code base with no change to run across different platforms. All one needs to do is to test a Flutter application just once and save a lot of time and money for the developer.

- **Benefit #8 Voice alerts**

For safety driving we used flutter text to speech for alerting the driver by voice so he can know about his status. This situation is done only if one of sensors has respond with an abnormal value, or camera detect any issue.

Chapter 6: Proposed System and Results

6.1. Temperature and Heart rate sensors output

To start calculating the sensors value and view them in the Firebase Realtime Database, we connected the sensors to raspberry pi, ran the code. The sensors value appears in the console and then gets updated in the Realtime database in the same second as shown in figure 6.1 and figure 6.2.

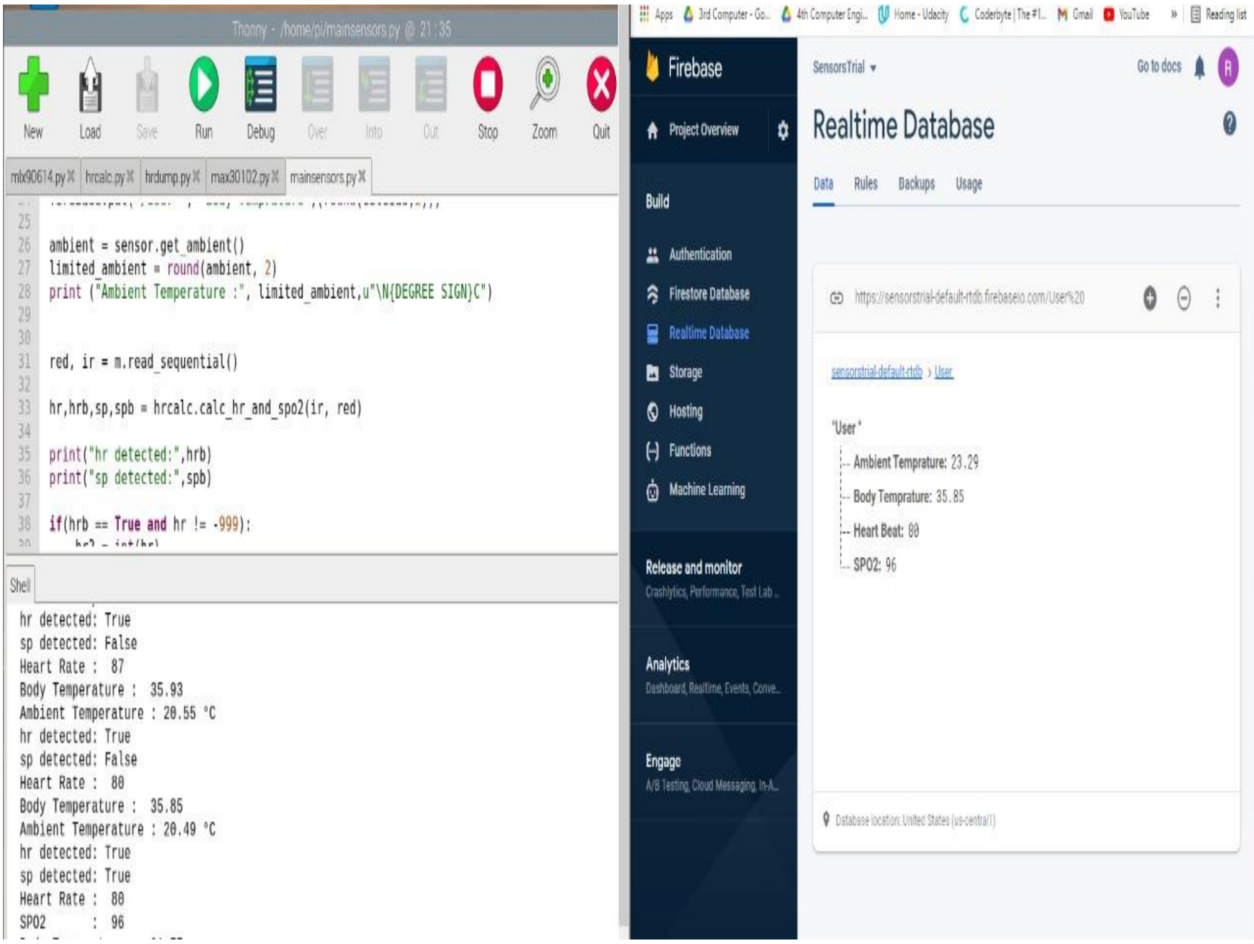


Figure 6. 1: Temperature and Heartrate Output

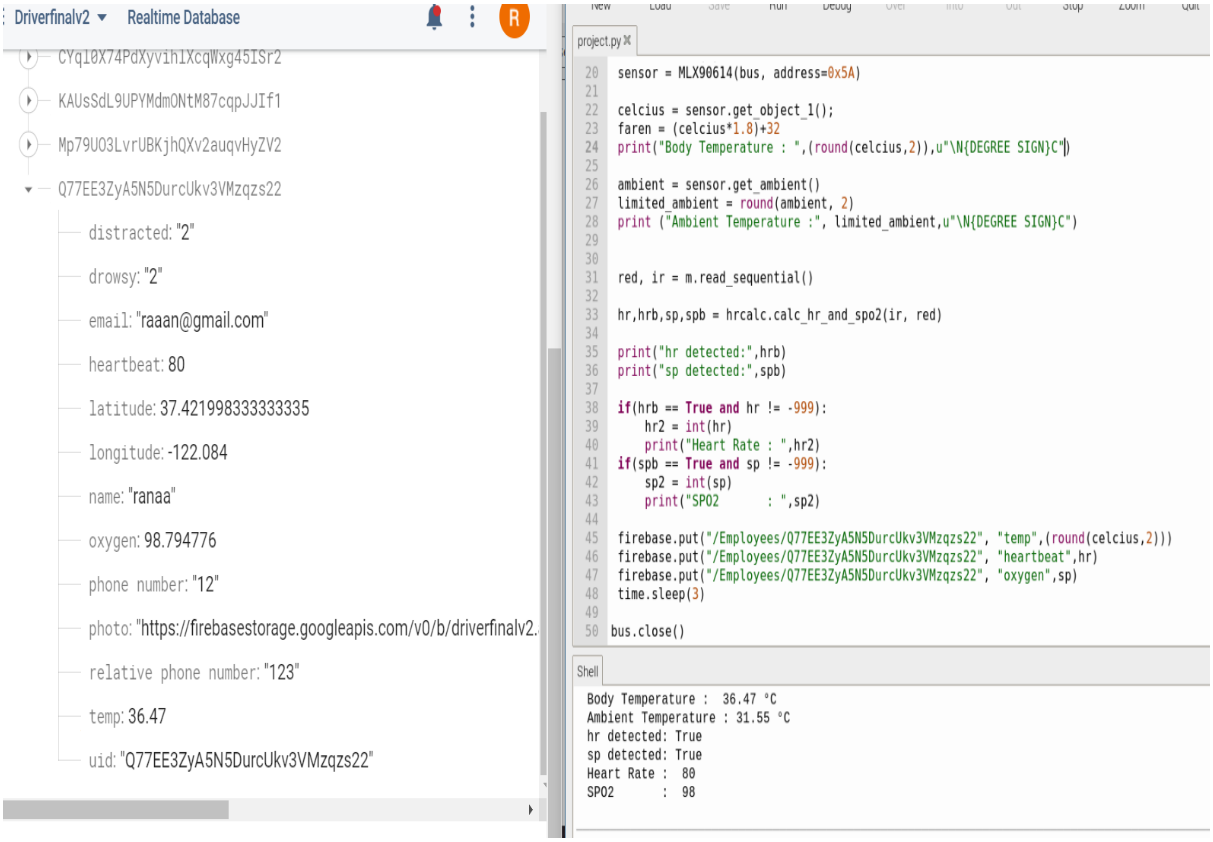


Figure 6. 2: Realtime Database Sensors readings

6.2. Our Dataset

The dataset is classified into 2 classes (Distracted, Seatbelt) and consists of 1200 images with 1080 Train, 60 Validation and 60 Test. The data were collected from different resources with different qualities and there were images that we took by ourselves.

The images in figure 6.3 were taken in the car with different conditions Seatbelt-detected, Seatbelt-not-detected, Driver distracted using Mobile Phone and Driver distracted eating.

Figure 6. 3: Dataset Sample

6.2.1. Image annotation

We have used LabelImg tool to annotate the data so we can use it to finetune Tensorflow Object Detection Model to detect the behavior of the driver. The annotation file has to be XML extension and includes the metadata of each image.

LabelImg is a free, open source tool for graphically labeling images. It's written in Python and uses QT for its graphical interface. It's an easy, free way to label a few hundred images to try out your next object detection project.

6.2.2. Data splitting

We used "train_test_split" function to split data into train, validation and test with 90%,5% and 5% respectively.

This function Splits arrays or matrices into random train and test subsets.

6.3. Face recognition

We used face recognition library which is built with Dlib. And it depends on 3 main functions to detect the landmarks of the face Eye ratio, Ear, and Lip distance.

This library gives us the ability to manipulate facial features in pictures with high accuracy.

Steps:

1. First, look at a picture and find all the faces in it
2. Second, focus on each face and be able to understand that even if a face is turned in a weird direction or in bad lighting, it is still the same person.

3. Third, be able to pick out unique features of the face that you can use to tell it apart from other people— like how big the eyes are, how long the face is, etc.

4. Finally, compare the unique features of that face to all the people you already know to determine the person's name.

Our model compares between key-points of the captured face on the image if the key-points matching the face in stored in our database from the registered user. Then the model can identify the driver and our system begins to start.

The below figure 6.4 shows a person that registered in our application:

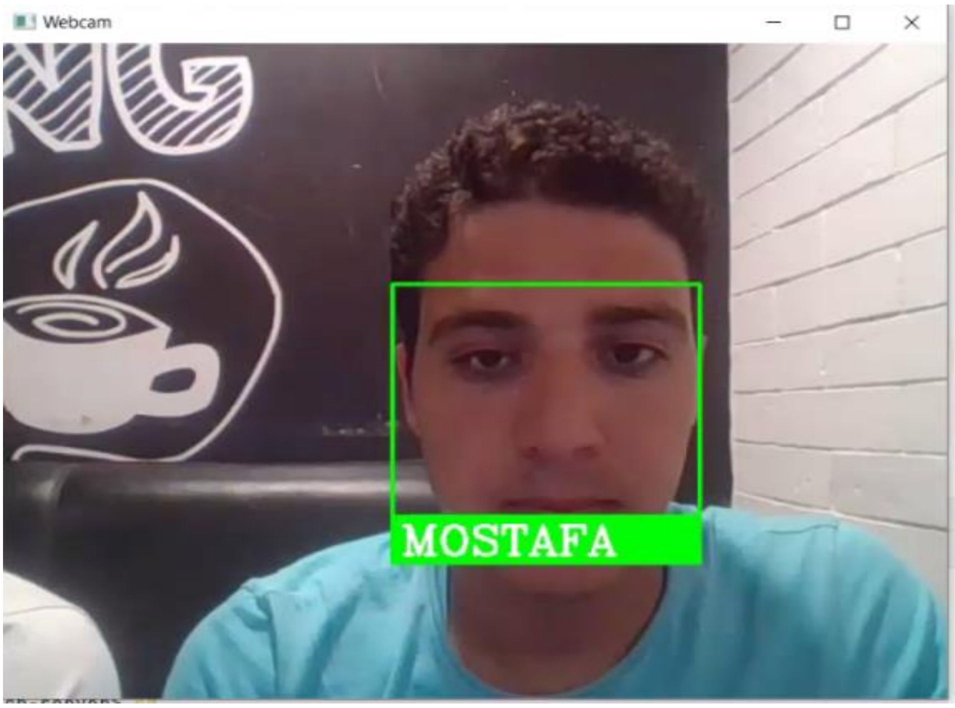


Figure 6. 4: Face Recognition for a stored user in database

On the other hand, if the person not logged in to our system and we don't have information or image of him the system will not recognize him, and the system won't start working. in addition to we will capture and send the unrecognized person to database and the application will send notification to the car owner as shown in figure 6.5.

Figure 6.5: If the user not registered in our database

All users' images either was Authorized or not will be sent to our database, and we will record the time of which they tried to start our system.

6.4. Drowsiness Model

Driver drowsiness detection is a car safety Technology which helps prevent accidents caused by the driver getting drowsy. The following model uses computer vision to observe the driver's face, either using a built-in camera or on mobile devices. When driver fatigue or drowsiness is detected, the system alerts the driver to prevent possible accidents. The following figure 6.5 shows the 68 landmarks that our model can detect in the face.

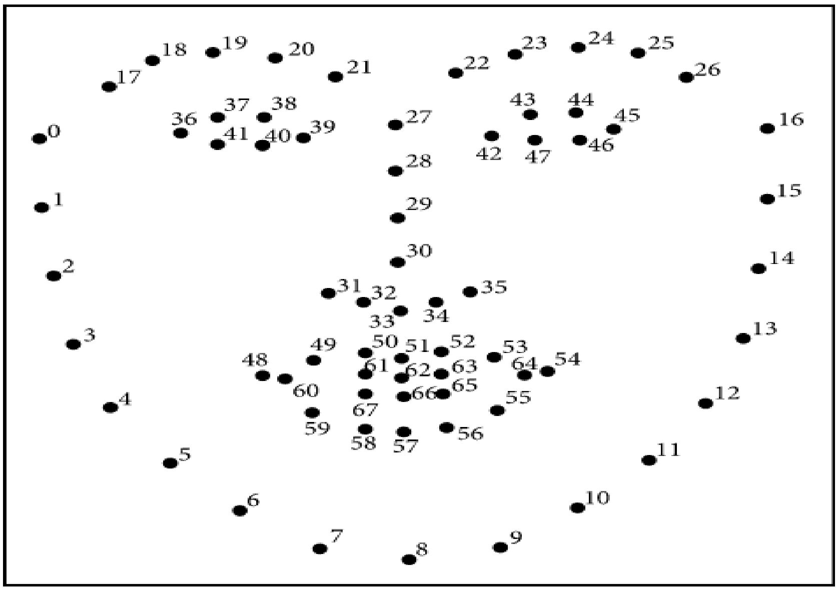


Figure 6.6: 68-Landmarks detection

Python's dlib library uses Kazemi and Sullivan's One Millisecond Face Alignment with an Ensemble of Regression Trees to implement the eye blink feature on the face. The program uses a facial training set to understand where certain points exist on facial structures. The program then plots the same points on region of interests in other images if they exist. The program uses priors to estimate the probable distance between key-points.

The following figure 6.7 shows the workflow of our model:

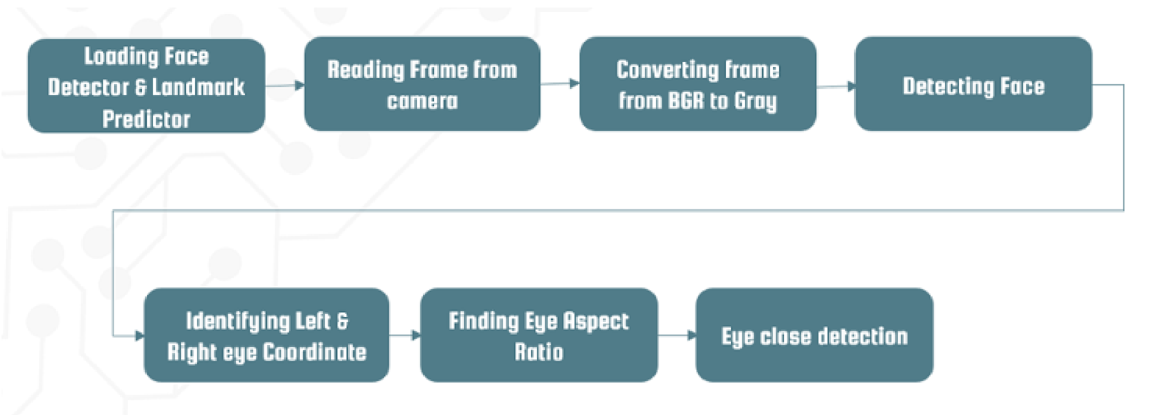


Figure 6.7: Face Recognition workflow

If a face is found, we apply facial landmark detection and extract the eye regions. Now that we have the eye regions, we can compute the eye aspect ratio to determine if the eyes are closed. If the eye aspect ratio indicates that the eyes have been closed for a sufficiently long enough amount of time, we'll sound an alarm to wake up the driver.

The below figure 6.8 shows the ratio in which we detect if the eyes is closed or opened.

Figure 6.8: Eye Aspect ratio

Each eye is represented by 6 (x, y)-coordinates, starting at the left-corner of the eye (as if you were looking at the person), and then working clockwise around the eye.

It checks 20 consecutive frames and if the Eye Aspect ratio is less than 0.25, Alert is generated.

6.4.1. Results

The following figure 6.9 shows the model detects if the driver is yawning.

If the yawning lasts for more than 5 seconds the system will detect drowsiness and will send alert

If the eyes remind closed for more than 5 seconds the system will detect drowsiness

Figure 6.9: Drowsiness detection

6.5. Seatbelt & Distraction Detection Model

this part TensorFlow Object Detection API is used as shown in Figure 6.10. The TensorFlow Object Detection API is an open-source framework built on top of TensorFlow that makes it easy to construct, train and deploy object detection models.

.

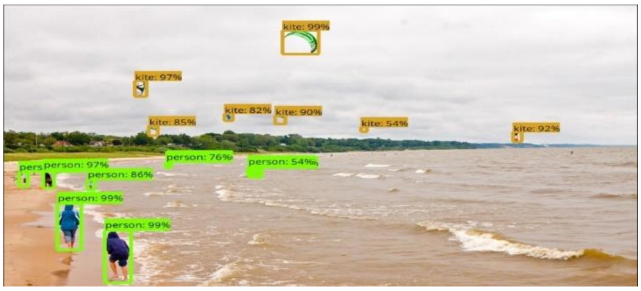


Figure 6.10: TensorFlow Object Detection API

6.5.1. What is TensorFlow object detection?

TensorFlow object detection is a computer vision technique that detects, locates, and traces an object from a still image or video. The method allows us to recognize how the models work and provides a fuller understanding of the image or video by detecting objects. Essentially, TensorFlow object detection enables us to train machines to understand human behaviors better and perform at optimal levels using the knowledge. When presented with an image or video, object detection TensorFlow works by identifying known objects from all instances through the help of computer vision. The history of object detection is as recent as the internet. The first recorded neural network for object detection was Overfit, as developers believed object detection would help improve image identification.

6.5.2. How does object detection work?

Object detection works in two ways:

- The first division permits networks to isolate the tasks of locating objects and classifying them (Faster R-CNN).
- The second division allows networks to predict class scores and bounding boxes (YOLO and SSD networks).

Nevertheless, most object detection tutorials are done by drawing bounding boxes around the input image to define the objects and their locations. However, object detection is different from image identification, although both usually go together. The difference lies in how models detect the objects. In image identification, detectors see and label an entire image, while only objects within an image are detected in object detection. More like zooming into the pixels of the image. The closest example of object TF detection is the Google Lens, an image and object recognition program. Google Lens works by identifying objects for curious users. All users need to begin is to take a picture of the thing. Google Lens then identifies the real-world object and fetches the required information from the internet.

6.5.3. What is the use of TensorFlow in object detection?

There are several reasons why the use of TensorFlow object recognition proves essential. As earlier mentioned, object detection informs machines about a part of human behaviours. By separating objects from humans, machines learn to improve everyday human lives. There are also other uses of object detection in the following:

1 - Automotive Technology: Automotive technology requires object detection models for self-driving cars, also known as driverless cars or robo-cars. Such cars employ little to zero human inputs to operate. However, their operations are primarily influenced by their capability to distinguish objects from humans.

2 - Surveillance: When you monitor activities or gather both human and nonhuman behaviours to influence or manage information, then you are surveillant. However, this form of security observation requires the distinction of humans and objects to arrive at a conclusive intelligence.

3 - Anomaly Detection: Every avid internet user must deal with the prompts of reCAPTCHA. Sometimes, it gets so frustrating that you either want to get rid of the test or bypass the prompt. Yet, the program is needed to protect sites from spam and abuse. ReCAPTCHA uses object detection algorithms to separate humans from robots.

6.5.4. Approaches of object detection

The algorithms deployed for object detection work in two ways: trained before use or used without being trained (unsupervised). In essence, it is either you're using:

- Machine Learning approach to detect objects

- The Deep Learning approaches.

But bear in mind that: The Machine Learning approach scans the surface features of an image to detect objects. Some of these features include the histogram or edges of the picture. After that, ML runs a regression test to predict the thing and identify its location. In the Deep Learning approach, it's a different ball game. Deep Learning uses convolutional neural networks to identify objects and their location. Since these networks are unsupervised, the system identifies objects on their own as well as their areas.

6.5.5. Workings of object detection

Most models detect objects in three ways, as you would see it once you've deployed a model to carry out object detection tasks. The task is done through the following steps:

- **Step one**: The object detection model breaks down the input image into several components. Then it draws boxes bound together around the segments, spanning the entire input image.

- **Step two**: Each broken-down component follows the process of feature extraction as the model begins to intensify. If there are any visual features in any bounding boxes, the model predicts the objects in that bounded box.

- **Step three**: The model closes the overlapping boxes without visual features and weaves them into a single bounded box.

6.5.6. Architecture models in TensorFlow object detection

The object detection framework in TensorFlow works on trained models, so you don't have to build from scratch. But these models have varied architecture and, therefore, differences in predictive accuracy. Most trained models on TensorFlow run through the following architecture:

MobileNet SSD

MobileNet SSD runs on a base layer known as MobileNet with several convolutional layers. Since it's a single convolution base layer with multiple other layers, the architecture detects objects at a single pass by identifying the locations of the bounding boxes and not the box shape itself. Each listed bounding box in MobileNet SSD contains

- The offset information of the box in four corners (cx, cy, w, h).
- The probabilities of C class (c1, c2, …cp).

MobileNet

Unlike MobileNet SSD, the convolution here is standard and helped by factorization into a point-wise convolution, such as the 1*1 convolutions. In computing MobileNet, as the model size reduces due to factorization, the computation likewise reduces.

Inception-SSD

Inception SSD has a similar design with MobileNet SSD, with the main distinction being the base layers. While the MobileNet SSD runs on the MobileNet base layer, Inception-SSD runs on Inception Model.

Faster RCNN

In 2015, Faster RCNN was presented by Ross Girshick, Shaoqing Ren, Kaiming He, and Jian Sun to the public as an advanced architecture of regional CNN. Since then, it has been deployed by several developers for object detection through the help of neural networks like YOLO and SSD. It works this way:

- Feed the neural networks with an input image to generate a convolutional feature map.
- Through the convolutional feature map, the region proposal gets identified and warped into squares.
- Squares are reshaped into a fixed size through the Region of Interest layer (ROL).
- From there, it becomes an input to a connected layer.
- The SoftMax layer predicts the region proposal class and the offset bounding box values through the Region of Interest layer (ROL).

6.5.7. Training TensorFlow Object Detection Model

Step 1. Clone this repository: https://github.com/nicknochnack/TFODCourse

Step 2. Create a new virtual environment

python -m venv tfod

Step 3. Activate your virtual environment

source tfod/bin/activate # Linux

.\tfod\Scripts\activate # Windows

Step 4. Install dependencies and add virtual environment to the Python Kernel

python -m pip install --upgrade pip

pip install ipykernel

python -m ipykernel install --user --name=tfodj

Step 5. Manually divide collected images into two folders train and test. So now all folders and annotations should be split between the following two folders

\TFODCourse\Tensorflow\workspace\images\train

\TFODCourse\Tensorflow\workspace\images\test

Step 6. Follow the steps in our notebook to start training and evaluate the trained model.

6.5.8. Results

The following figures show our system detect the Seatbelt with confidence 100% as shown in figure 6.11.

Figure 6.11: Seatbelt Results

Here’s our system detect that the driver is using his Mobile phone and distracted as shown in figure 6.12.

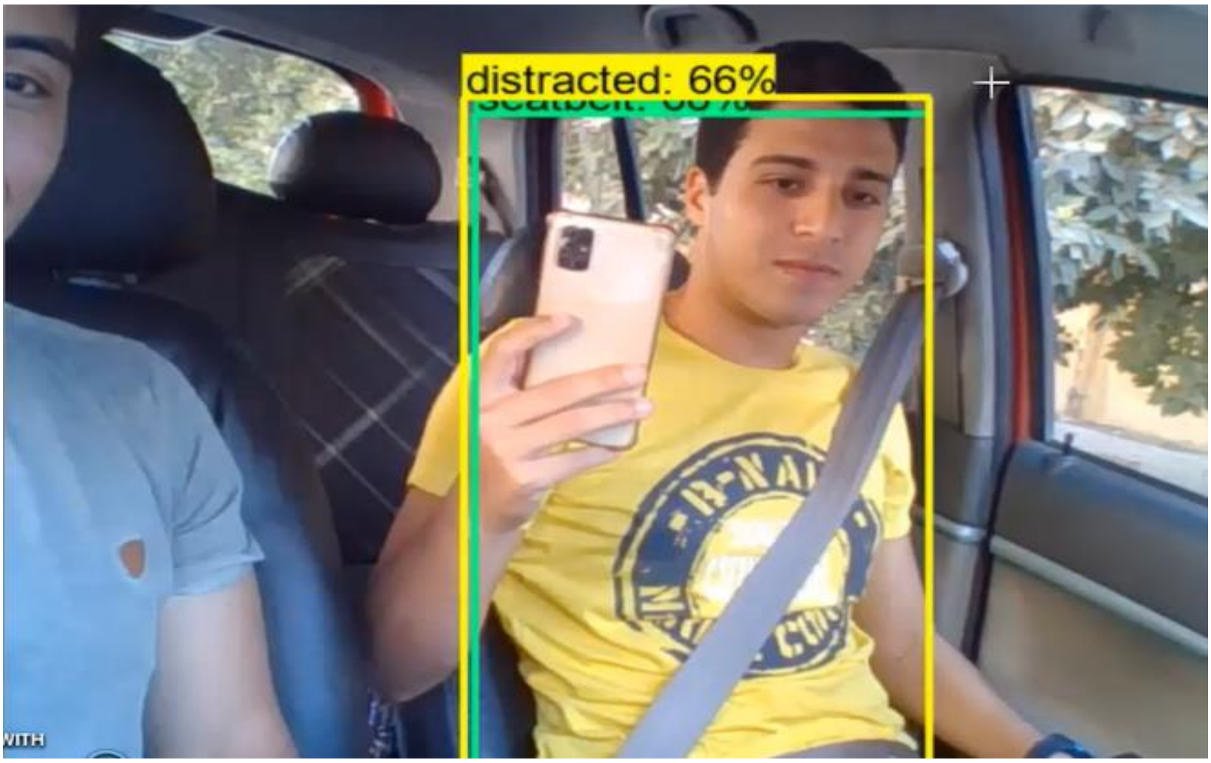


Figure 6. 12: Distraction Results

All records of the driver during the trip will be recorded and send to our database. So we can have a history of the driver behaviour.

6.6. Our Mobile Application

The following figure 6.13 is a screenshot of the welcome page in our Mobile Application.

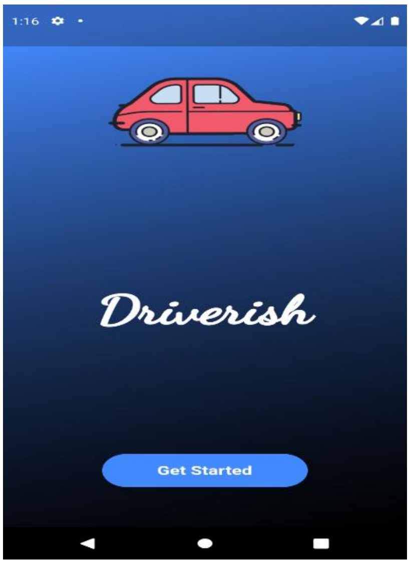


Figure 6. 13: Welcome Page

The following figure 6.14 is a screenshot of the Login in page in our Mobile Application.

‘

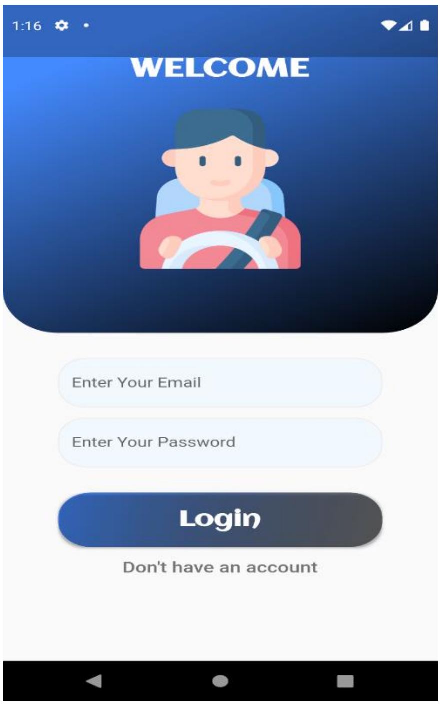


Figure 6. 14: Login Page

The following figure 6.15 is a screenshot of the Sign up page in our Mobile Application.

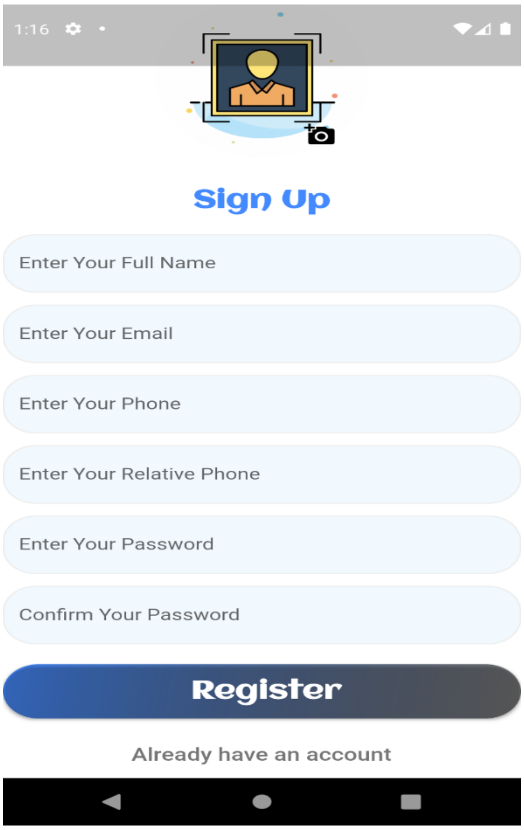


Figure 6. 15: Sign Up Page

The following figure 6.16 is a screenshot of the Home page in our Mobile Application.

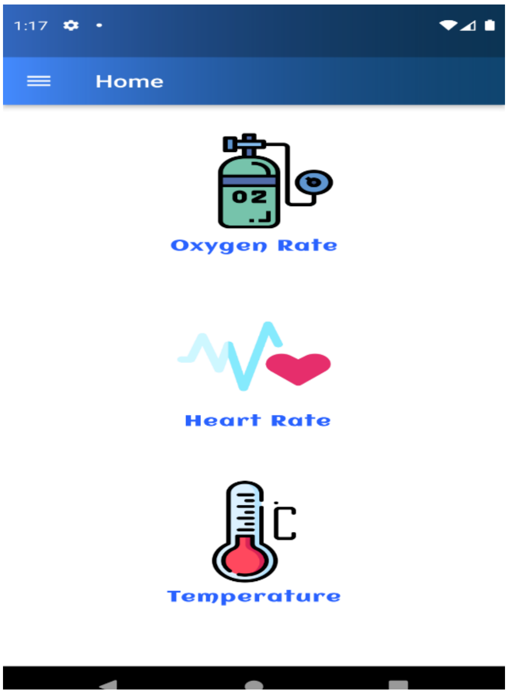


Figure 6. 16: Home Page

Chapter 6: Proposed System and Results

The following figure 6.17 is a screenshot of the Temperature page in our Mobile Application.

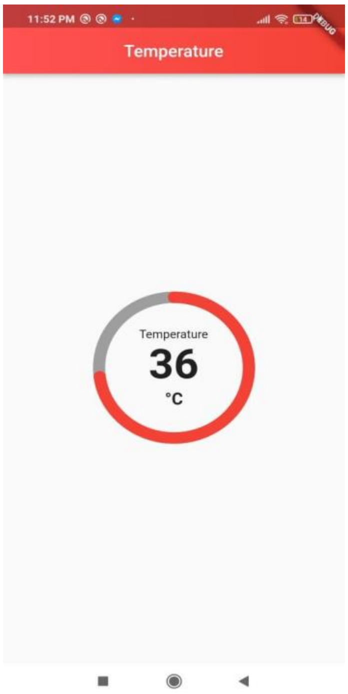


Figure 6.17: Temperature Reading

The following figure 6.18 is a screenshot of the Heart rate page in our Mobile Application.

Figure 6.18: Heart rate Reading

The following figure 6.19 is a screenshot of the Oxygen Saturation page in our Mobile Application.

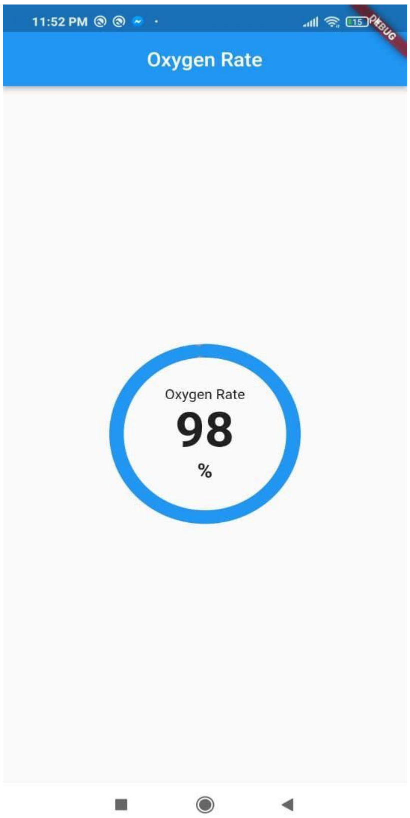


Figure 6.19: Oxygen Saturation Reading

Chapter 7: Testing

- **Register / login**

Testcase Id	Preconditions	Test data	Expected Result	Actual Result
TC_01	Valid Email and Password	Email= Reham11@gmail.com Password=1234567	Login successfully	Login successfully
TC_02	Invalid Email and Password	Email= Reham@demo.com Password=1234	Message will appear to tell you that this email is not valid, there is no such user with this email.	Message appeared
TC_03	Valid Email and Invalid Password	Email= Reham11@gmail.com Password= 1234	Message will appear to tell you that this password is wrong.	Message appeared
TC_04	Invalid Email and valid Password	Email= Reham@demo.com Password= 1234567	Message will appear to tell you that this	Message appeared

			email is not valid, there is no such user with this email.	

- **Sensors & camera**

Testcase Id	Sensors			Camera			
	Temp	**Heartrate**	**Spo2**	**Face detection**	**Drowsy**	**Seatbelt**	**Distraction**
Optimum value	**36-38**	**50-110**	**95-99**	**The user**	**awake**	**Wear seatbelt**	**attentive**
TC_01	37	70	97	The user	awake	Wear seatbelt	attentive
TC_02	37	70	97	Not The user	awake	Wear seatbelt	attentive
TC_03	33	40	90	the user	drowsy	wear seatbelt	Not attentive
TC_04	41	130	110	the user	drowsy	wear seatbelt	Not attentive
TC_05	37	130	110	the user	drowsy	wear seatbelt	Not attentive
TC_06	41	70	97	the user	awake	wear seatbelt	attentive
TC_07	38	110	99	the user	awake	wear seatbelt	attentive

TC_08	36	50	95	the user	awake	wear seatbelt	attentive
TC_09	37	70	97	the user	drowsy	wear seatbelt	Not attentive
TC_10	37	70	97	the user	awake	Don’t wear seatbelt	attentive
TC_11	37	70	97	the user	awake	wear seatbelt	Not attentive

Testcase Id	Testcase for sensors	Testcase for camera	Expected Result	Actual Result
TC_01	Good sensors readings	Camera does not detect any issue	Voice alert (“you have a good health”)	Voice alert (“you have a good health”)
TC_02	Good sensors readings	Camera detects issue in face detection	Send SMS to the user with location and photo	Send SMS to the user with location and photo
TC_03	Abnormal sensors readings	Camera detects issue in drowsiness detection	Send SMS to the company with location and call ambulance	Send SMS to the company with location and call ambulance

TC_04	Abnormal sensors readings	Camera detects issue in drowsiness detection	Send SMS to the company with location and call ambulance	Send SMS to the company with location and call ambulance
TC_05	Abnormal Heartrate sensor readings	Camera detects issue in drowsiness detection	Send SMS to the company with location and call ambulance and active speed limiter circuit	Send SMS to the company with location and call ambulance and active speed limiter circuit
TC_06	Abnormal Temperature sensor reading	Camera does not detect any issue	Send SMS to the company with location and voice alert ("your temperature is high") and active speed limiter circuit	Send SMS to the company with location and voice alert ("your temperature is high") and active speed limiter circuit
TC_07	High sensors readings	Camera does not detect any issue	voice alert ("you have to take a break your health is not good")	voice alert ("you have to take a break your health is not good")

TC_08	low sensors readings	Camera does not detect any issue	voice alert ("you have to take a break your health is not good")	voice alert ("you have to take a break your health is not good")
TC_09	Good sensors readings	Camera detects issue in drowsiness detection	Send SMS to the company with location and voice alert ("you are sleepy please take a break and drink some coffee") and active speed limiter circuit	Send SMS to the company with location and voice alert ("you are sleepy please take a break and drink some coffee") and active speed limiter circuit
TC_10	Good sensors readings	Camera detects issue in seatbelt detection	voice alert ("please wear your seatbelt")	voice alert ("please wear your seatbelt")
TC_11	Good sensors readings	Camera detects issue in distraction detection	voice alert ("please be attentive in your driving")	voice alert ("please be attentive in your driving")

Chapter 8: Conclusions and Suggestion for Future Work

8.1. Conclusion of our System.

We have succeeded in constructing a system that enhance the safety of driver by detecting if he/she is the owner of the car and detect if he is distracted or drowsy and make sure that he is wearing the seatbelt. In case if the driver is drowsy, we send SMS with the location of the driver to the company and alert him. Our system also helps in monitoring the health of driver by measuring his/her heart rate, temperature and spo2 levels if there is any problem, we send SMS to the relative or if an emergency we call ambulance. All of which can be viewed through a user interface in the form of mobile application.

8.2. Suggestion for Future Work

Our goal is to add more features to this project and enhance it by:

- Use more sensors to monitor other health issues.
- Add more feature like Auto parking and v2v communication.
- Add more feature in the mobile application.
- The installation of event recording devices can help to understand how crashes occur and help prevent these.
- Alcohol detection devices that prevent drivers who have consumed alcohol from starting up their vehicles.
- Add more Languages to our system and voice commands.
- Implement our system on a real car and test it.

References

[1] Borrelli, L. (2022, March 2). *Drowsy Driving 2021 Facts & Statistics*. Bankrate. https://www.bankrate.com/insurance/car/drowsy-driving-statistics/

[2] Coleman, S. (2022, March 7). *Distracted Driving Statistics 2022*. Bankrate. https://www.bankrate.com/insurance/car/distracted-driving-statistics/#:%7E:text=Distracted%20driving%20causes%20about%203%2C000,and%20295%2C000%20injuries%20in%202016.

[3] Taub, E. A. (2017, March 16). *Sleepy Behind the Wheel? Some Cars Can Tell*. The New York Times. https://www.nytimes.com/2017/03/16/automobiles/wheels/drowsy-driving-technology.html

[4] Cristina, A., Leba, M.&Nicolae,M. (2018). Driver Monitoring System for Automotive Safety. IOP Sceince, 294 (1), 1-8. DOI:10.1088/1757 899X/294/1/012046. Retrieved from: https://www.researchgate.net/publication/322452541_Driver_monitoring_system_for_automotive_safety

[5] Agarwal,P. & Sharma, R. (2019). *Driver Drowsiness Detection System* [Bachelor's thesis, Kalinga Institute of Industrial Technology University]. ResearchGate. https://www.researchgate.net/publication/336878674_DRIVER_DROWSINESS_DETECTION_SYSTEM

[6]- Sneha, S., Tejaswini, H. & Sangeetha, S. (2021). Automotive Driver Monitoring System. *International Journal of Engineering Research & Technology (IJERT)* , 10 (1), 388- 391. DOI:IJERTV10IS010136. https://www.ijert.org/research/automotive-driver-monitoring-system-IJERTV10IS010136.pdf

[7] *What is an Embedded System? Definition and FAQs | HEAVY.AI*. (2022). Heavy.AI. https://www.heavy.ai/technical-glossary/embedded-systems

[8] Tan, J. (2022, May 11). *Embedded Linux: A Beginner's Guide*. Latest Open Tech From Seeed. https://www.seeedstudio.com/blog/2021/01/20/beginners-guide-to-embedded-linux/

[9] Education, I. C. (2022, July 6). *Machine Learning*. IBM. https://www.ibm.com/cloud/learn/machine-learning

[10] Wikipedia contributors. (2022, July 10). *Python (programming language)*. Wikipedia. https://en.wikipedia.org/wiki/Python_(programming_language)

[11] Wikipedia contributors. (2022a, June 4). *Dart (programming language)*. Wikipedia. https://en.wikipedia.org/wiki/Dart_(programming_language)

[12] *Meet Android Studio* |. (2019). Android Developers. https://developer.android.com/studio/intro

[13] Real Python. (2021, May 15). *Thonny: The Beginner-Friendly Python Editor*. https://realpython.com/python-thonny/

[14] *Project Jupyter*. (2019). Home. https://jupyter.org/

[15] *Security check*. (2019). Real VNC. https://help.realvnc.com/hc/en-us/articles/360002320638-How-does-VNC-technology-work-

[16] Ltd, R. P. (2022). *Buy a 3 Model B+ –*. Raspberry Pi. https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/

[17] Ltd, R. P. (2020). *Buy a Camera Module 2 –*. Raspberry Pi. https://www.raspberrypi.com/products/camera-module-v2/

[18] https://www.amazon.in/xcluma-MAX30102-Upgraded-Ultra-Low-Compatible/dp/B07TZCNG2G

[19] A. (2022, January 26). MLX90614 IR Temperature Sensor: Pinout, Datasheet, Circuit [Video]. Circuitry. https://www.apogeeweb.net/circuitry/MLX90614-temperature-sensor.html

[20] Campbell, S. (2021, November 14). *Basics of the I2C Communication Protocol*. Circuit Basics. https://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/

[21] *Firebase Realtime Database | Firebase Documentation*. (2022). Firebase. https://firebase.google.com/docs/database