

Alcohol Detection System with Alert Notification Application

Shrushti Vishwasrao

Master of Science

Final Year Computer Application Department

Fergusson College (Autonomous),

Pune, India.

Abstract—This project presents the design and implementation of an Alcohol Detection with Alert Notification using the Android Application and Firebase as the CU (Control Unit). The system will continuously monitor level of alcohol concentration in alcohol detection sensor and thus pop notification on the application and provides for option if the alcohol concentration is above threshold level. The model will also show the alcohol level and can alert the driver through the buzzer control with the application. The project provides an efficient solution to control accidents due to drunk driving.

Keywords— ESP8266 NODE MCU, MQ3 Sensor, Buzzer, LED, Bread Board.

1. INTRODUCTION

Vehicle transportation is the most patronized means of transportation in commuting people and products from one location to another. It is therefore expected to witness more road carnages than any other means of transportation such as rail, sea and air transport. Drink driving is one of the major causes of road accidents all over the world, according to the bureau of transportation statistics, every 2 hours three people are killed in alcohol related highway crash. A major method for testing level of alcohol is the blood alcohol content (BAC) test using breath analyzers at check points. A 0.05% blood alcohol is enough to impair the sense of judgment when driving. This method lacks credibility due to unavailability of required manpower, compromise on the part of inspection officers and the cost of maintaining check points over a wide geographical location. This paper presents a better approach by developing a ubiquitous system that prevents drink driving on our roads.

1.1 Existing System Features

Provided the facility of detection and notifying. No alert buzzer for the driver. So there is a need of an effective system to check the drunken driver. In the existing system, alcohol detectors are not proposed in any of the vehicles, hence there is a chance for anyone to drink and drive. Traffic police uses alcohol detectors to avoid drunk and drive system.

1.2 Need of The System

The purpose of this project is to develop vehicle accident prevention by method of alcohol detector in an effort to reduce traffic accident cases based on driving under the influence of alcohol. This system improves the safety of human being. And hence providing the effective development in the automobile industry regarding to reduce the accidents cause due to alcohol. Try to anticipate what the other drivers

might do in the situation, especially on open roads or when there is heavy traffic. It's always better to stay wary. It goes without saying that consumption of alcohol is a serious impediment to making the right choices and it is very dangerous to drive inebriated. If you are going to a party where you are likely to consume alcohol, make sure someone who does not drink is the designated driver or arrange for a taxi. Driving under the influence of alcohol is a punishable offence, including the possibility of cancellation of your license or even jail time.

1.3 Overview of the Project

As increase in the consumption of alcohol by the vehicle drivers, there has been hike in the accidents being taken place. Even with development of the latest technology to stop these, there are still cases of such happening. To avoid this, we are introducing a project called Spotting Alcohol System. This handy, easy to implement project sense the presence the alcohol (consumed by the driver) in the vehicle, and immediately sends the message and starts the blinking LED. The circuit can sense the alcohol after particular level thus minimizing the chances of accident. The alcohol is detected by the sensor; it sends the signal to the Microprocessor which in turn sends the signal to the owner of that vehicle. This takes place until the alcohol is not below the critical level.

2. ANALYSIS

2.1 Feasibility Study

- A feasibility study is an analysis of how successfully a project can be completed, accounting for factors that affect the project.
- The outcome of this study is use to determine the potential positive and negative outcomes before investing time, money and human resource on the project.

The main factors affecting are technical, operational and time factors.

2.1.1 Technical Feasibility

A large part of determining resources has to do with assessing technical feasibility. It considers the technical requirements of the proposed project. The Technical requirements are then compared to the technical capability of the organization.

- The technology required for the System is currently available.

- The System is functional within the given resource constraints.
- Software is already available so there won't be any need to purchase any new software.
- The device is the only product which will be purchased by the user.
- Higher level programming languages will be used for the System.
- The System may include updates which can be easily upgraded to provide additional features.
- The System is technically feasible within the limits of current technology.

2.1.2 Economic Feasibility

Economic analysis could also be referred to as cost/benefit analysis. It is the most frequently used method for evaluating the effectiveness of a new system.

- The cost of maintaining the System is minimal.
- Estimated cost of hardware and software or of the software development and future upgradation is within the given resource constraints.
- No additional Hardware is required other than the conventional hardware used for the device.
- The expenditures for maintaining the device in any given year may not exceed the amount of resources available.
- If possible, resources are carried forward into the subsequent year in order to ensure that adequate funding is available for future expenditures.
- The benefits outweigh costs in the System, hence we can implement it.
- The System is cost effective. Hence, economically feasible.

2.1.3 Operational Feasibility

Operational feasibility is a measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. The proposed System satisfies the following –

- System provides end users with accurate and useful information.
- Provision of effective controls to guarantee accuracy of information and timely delivery of alerts.
- Makes maximum use of available resources, including electrical components & time.
- Provides reliable services.
- Operating the System requires no programming knowledge to the end users and with basic knowledge they can use the System.
- The Users have been involved in the planning.

2.2 Software Requirement Specification

- PROCESSOR: INTEL(R) CORE(TM) i5-7200U
- PROCESSOR SPEED: 2.50GHz
- MONITOR: COLOR MONITOR
- HARD DISK: 1000GB
- RAM: 8GB

- MOUSE: SCROLLING MOUSE
- KEY BOARD: MM KEYBOARD

2.1.1 Hardware Requirements Specification

- PROCESSOR: INTEL(R) CORE(TM) i5-7200U
- PROCESSOR SPEED: 2.50GHz
- MONITOR: COLOR MONITOR
- HARD DISK: 1000GB
- RAM: 8GB
- MOUSE: SCROLLING MOUSE
- KEY BOARD: MM KEYBOARD

IOT Hardware Kit:

- Bread Board
- Wires
- LEDs
- Buzzer
- MQ3 Sensor
- NODE ESP8266

3. DESIGN

3.1 Database Table Design

In Firebase we do not need any table design for our project. It's just dynamic temporary storage.

We have three fields

- Level of Alcohol
- Buzzer ON/OFF
- LED ON/OFF

Above values keep changing according to the IOT kit output.

3.2 Diagrams

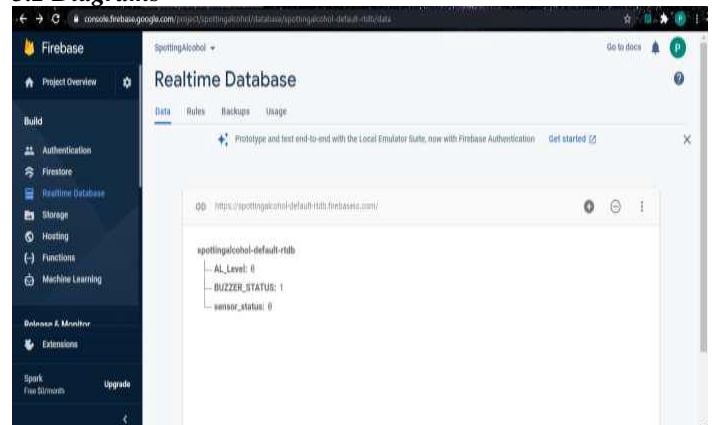


Fig 1: Firebase Data

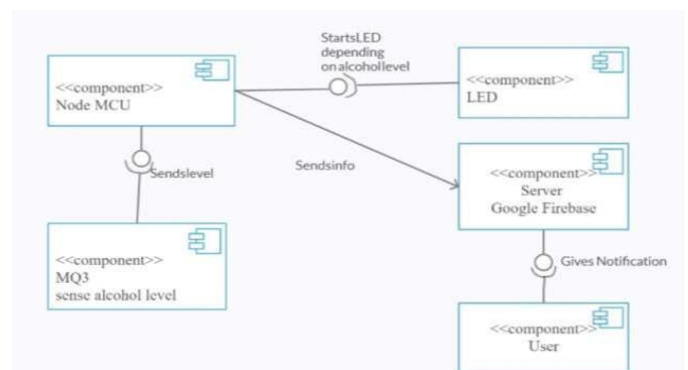


Fig 2: Component Diagram

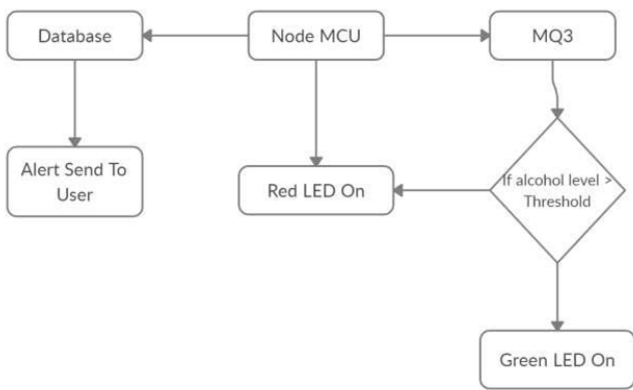


Fig 3: Activity Diagram

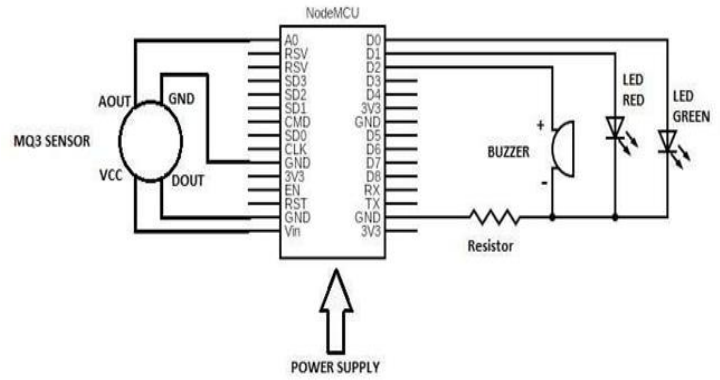


Fig 7: Pin Diagram

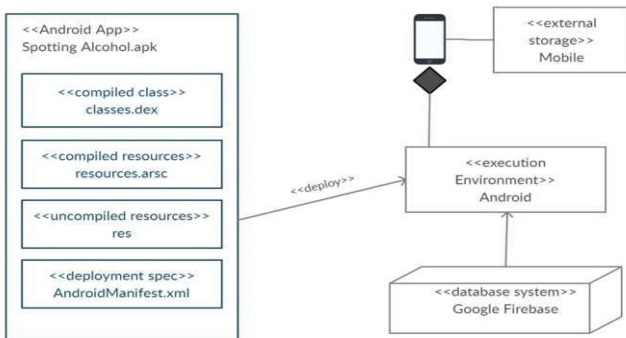


Fig 4: Deployment Diagram

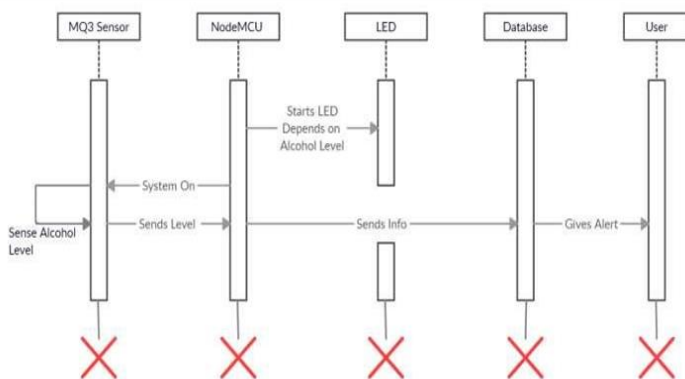


Fig 5: Sequence Diagram



Fig 8: The alert notification

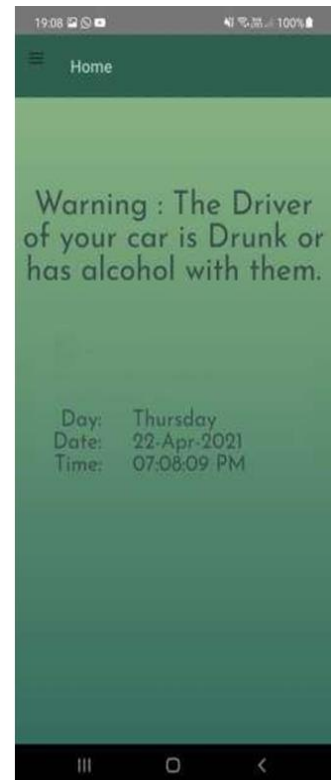


Fig 9: Home Page of app

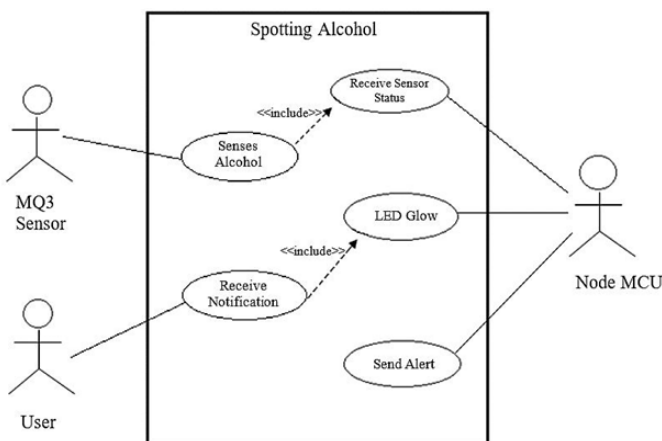


Fig 6: Use case Diagram

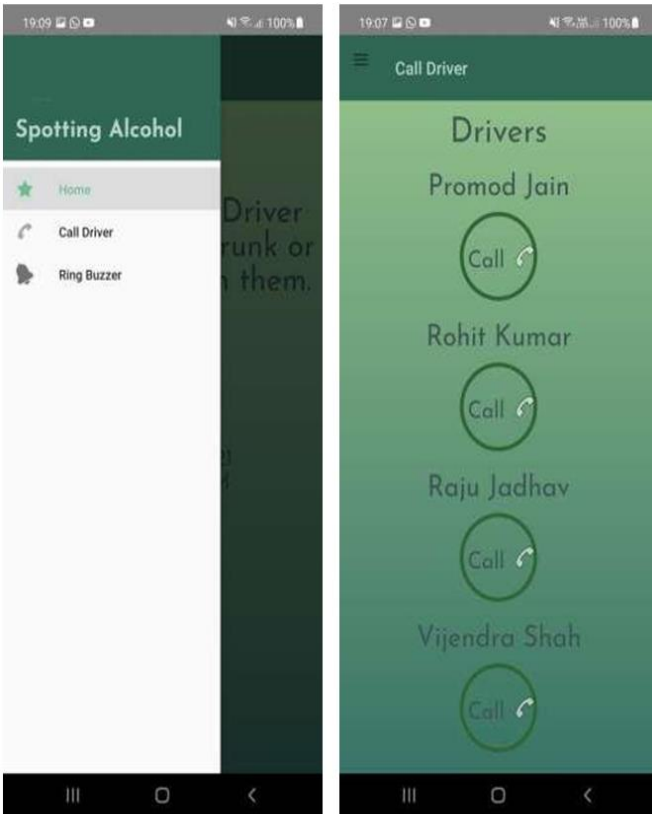


Fig 10: Drawer for more options. Fig 11: Direct call option

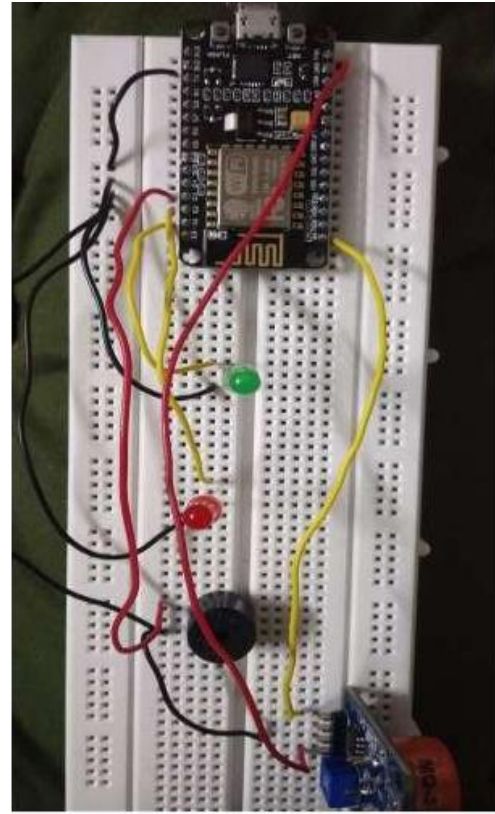


Fig 11: IoT Board



Fig 10: The alcohol level and control of buzzer

4. TESTING

4.1 Importance of Testing

- **Software Testing Saves Money**

Testing has many benefits and one of the most important ones is cost-effectiveness. Having testing in your project can save money in the long run. Software development consists of many stages and if bugs are caught in the earlier stages it costs much less to fix them. That is why it's important to get testing done as soon as possible. Getting testers or QA's who are technically educated and experienced for a software project is just like an investment and your project will benefit budget-wise.

- **Security**

Another important point to add is security. This is probably the most sensitive and yet most vulnerable part. There have been many situations where user information has been stolen or hackers have gotten to it and used it for their benefit. That is the reason people are looking for trusted products that they can rely on. As a user of many products and apps, I am always looking for products that I would give my information to with confidence and know that it will be safe; perhaps so do you. Our personal information and what we do with it should stay as private as possible, especially using services where it is a vulnerability to us, for example, banking information, security details etc. How testing can help your product security:

- The user gets a trustworthy product.
- Keeps user's personal information and data safe.
- Vulnerability free products.
- Problems and risks are eliminated beforehand.
- Saves a lot of troubles later on.

- **Product Quality**

In order to make your product vision come to life, it has to work as planned. Following product requirements is imperative, to an extent, because it helps you get the wanted end results. Products always serve users in some ways, so it's very important that it brings the value it promises, hence it should work properly to ensure great customer experience. Development of an app, for example, has many processes included and testing gets a glimpse of every bit – it checks if the apps graphics are aligned properly, tests the main functionality, checks if menus are intuitive, etc. After developers' fix issues, sometimes another issue may appear unexpectedly somewhere else, that's just how testing goes sometimes, so it's great to find those issues to be resolved and be a part of quality product being delivered to marketplace.

- **Customer Satisfaction**

The ultimate goal for a product owner is to give the best customer satisfaction. Reasons why apps and software should be tested is to bring the best user experience possible. Being the best product in this saturated market will help you gain trustworthy clients which will have great long-term effects. Once users will have amazing customer experience they will, without a doubt, tell their friends and word to mouth will make it advertise itself, but this works both ways. Customer trust is not easy to earn, especially if your product is glitching and functioning only 60% of the time. You are a user of many products and perhaps have had horrible experiences that made you delete the app and tell others not to use it. These days the market is so saturated that first impression is important, otherwise users will find another product that meets his needs.

4.2 Types of Testing

- **Functional Testing:**

Testing is normally achieved by user interface initiated test flows. Not just the flow of a use case is tested, but the various business rules are also tested. Testing is done by certifying the requirements. i.e. whether the application is working based on the requirements.

In addition to the mobile app's specific functionality, there are other scenarios one should test for to limit errors, including but not limited to checking if:

- The application installs and launches correctly
- The users can sign-up and login
- Text boxes and buttons function properly
- Push notifications render correctly.

- **Android UI Testing:**

This is a user-centric testing of the application. In this test phase, items such as visibility of text in various screens of the app, interactive messages, alignment of data, the look and feel of the app for different screens, size of fields etc. are tested under this.

- **Compatibility Testing:**

This testing is done mostly in the form of two matrices of OS Vs app and Device Model Vs App. Usually, a list of

supported OS (and sometimes devices) is provided by the product owner or customer. Specifically, you should know if:

- The app is compatible with different operating systems and their various versions (iOS, Android, Windows, etc.)
- The app performs well with varying networks and their parameters (bandwidth, operating speed, etc.)
- The app is compatible with different browsers (Google, Firefox, Safari, etc.)
- The app is compatible with different devices (screen size, data storage, etc.)
- There are also two types of compatibility testing to consider:
 - Backwards: testing the mobile app behaviour with older software versions
 - Forwards: testing the mobile app behaviour with new — including beta — software versions.

- **Interface Testing:**

In other words, it is also termed as Integration testing. This testing is done after all the modules of the app are completely developed, tested individually and all the bugs are fixed verified.

Interface testing includes tests like a complete end to end testing of the app, interaction with other apps like Maps, social apps etc., usage of Microphone to enter text, usage of Camera to scan a barcode or to take a picture etc.

- **Usability Testing:**

Known as user experience testing, usability testing checks how user-friendly the app is in terms of ease of use and intuitiveness. Ideally, usability testing revolves around the entire app-driven customer experience with insights that include the identification of bugs and recommendations for ways to improve the customer experience, both in and out of the app. Engineers, marketers and product people all want to test whether or not the end-to-end “app-driven” experience is world-class. To that end, it's important for app usability testing to be done with real people, on real devices to quickly identify and fix usability issues prior to app release.

- **Performance and load testing:**

Performance testing checks how well the mobile application performs under a particular workload. These tests are important to ensure your app isn't malfunctioning.

Performance and load tests check for the following:

- Device performance: Start-up time, battery consumption, memory consumption
- Network performance: Delays or errors in receiving information
- API / Server performance: How quickly and in what format data is transferred Additionally, your app should have built-in back-up and recovery functions that save or recover user data that could be lost for any reason. This is where you would test that capability.

• **Automated Testing:**

As we've pointed out before, there are some cases where manual testing is the better option. However, some QA tests are either too tedious or too complex for human testers. That's why smartly executed automated testing, alongside manual tests, can help assure quality and release better products, faster.

A few automated testing best practices and challenges include:

- The thoughtful design, build, and maintenance of accurate test scripts
- The alignment and integration of existing engineering workflows with your automated testing process
- The creation and maintenance of your test automation framework, including infrastructure
- The management of test runs and setups
- Rigorous reviews to validate test results and defects
- Careful monitoring and rapid response to noise and flaky tests.

4.3 Test Cases

| Test ID | Test Case Description | Input | Expected Output | Actual Output | Pass/Fail |
|---------|---|---|---|--|-----------|
| 1 | MQ3 Sensor Working. | Sense the alcohol level. | Glow the appropriate LED. | Glow the appropriate LED successfully. | Pass |
| 2 | ESP8266 Node MCU. | Output of Sensor, buzzer, LED. | Read correct input of sensor, control of LED, & buzzer. | Successfully read's and controls. | Pass |
| 3 | Sensor: If alcohol level<threshold. | Alcohol. | Red LED should glow. | Red LED On successfully glows | Pass |
| 4 | Sensor: If alcohol level>=threshold. | Alcohol . | Green LED should glow. | Green LED On successfully glows | Pass |
| 5 | Buzzer: Should beep when alcohol detected. | Instruction given by Node MCU. | Should beep. | Beeps successfully. | Pass |
| 6 | Notification | Firebase sends the sensor value To app. | A pop up notification should appear. | Notification successfully received. | Pass |
| 7 | Firebase console data: | Node MCU data | Values from Node MCU should display on firebase console accordingly | Correct values displayed on firebase console | Pass |
| 8 | Application: Working of buttons | Click on the buzzer on/off button. | The buzzer on the IOT kit should get turned on/off. | Successfully the buzzer turns on/off. | Pass |
| 9 | Application: Call button | Click on the call button | Phonebook of device should open with phone number. | Successfully open with correct number. | Pass |

The application acts as an interface between the user and the

5. REPORT

As increase in the consumption of alcohol by the vehicle drivers, there has been hike in the accidents being taken place. Even with development of the latest technology to stop these, there are still cases of such happening. To avoid this, we are introducing a project called Spotting Alcohol System. This handy, easy to implement project sense the presence the alcohol (consumed by the driver) in the vehicle, and immediately sends the message and starts the blinking LED. The circuit can sense the alcohol after particular level thus minimizing the chances of accident. The alcohol is detected by the sensor; it sends the signal to the Microprocessor which in turn sends the signal to the owner of that vehicle. This takes place until the alcohol is not below the critical level.

car driver. The system minimizes the risks factor of the driver or anyone being injured.

6. DRAWACKS AND LIMITATIONS

- The limitation for this IOT kit that it can be poisoned.
- Also the IOT kit is not suitable for areas where CO and CO2 produced.
- Also the drawback is that needs 24/7 power supply and works only when +5v power supply.

7. FUTURE ENHANCEMENT

This application already provides the alcohol level, buzzer feature, double alert notification. We enhance this application

by add more features like more strong sensor and GPS, to get the location of the car. Also can add engine locking for safety. It would help the user to know where his/her car is, in which condition, and is it in good hands, and in case important contacts. Also could take necessary precaution needed for it. Also add login module and store data in dynamic database(Firebase). Also user will have more specific information and more clarity about their car and driver safety. Also, we can add a chat module for customer care, as when and if the kit does not work or has a problem. Then they could contact us for help. Finally, there can be one more feature added to this system i.e., token that can be collect on cloud database which can be used to send the information by the car owner to the police office. The token will contain the live location of the driver, name, car number.

8. CONCLUSION

As majority of accidents occurring nowadays are thanks to drunk driving, thus a good methodology to forestall this is often to develop associate Integrated Drunk and Drive Detection. Integrated Drunk Associated Drive Detection system is meant with an economical and increased technology The main unit of this project is associate "Alcohol sensor". If the person within automobile has consumed alcohol, then it's detected by the sensing element. sensing element provides this signal to a comparator IC. The output of comparator is connected to the microcontroller. Integrating options of all the hardware parts used are developed in it. Presence of each module has been reasoned out and placed fastidiously, therefore conducive to the simplest operating of the unit. In this project we have developed a real time model that can automatically push a notification to the owner of the car when a drunken driver tries to drives the car or is carrying huge amount of alcohol. Now-a-days car accidents are mostly seen which can be prevented by this. By using this spotting alcohol model in car, we can save /guard the life of the driver, car and also rest of the passengers. on contrary to the seemingly obvious conclusion, can save you just as much time as money and, more importantly, keep the roads safer for everyone. It will help you in traffic conditions and anticipate the possibility of a mishap.

9. REFERENCES

- [1] <https://testlio.com/blog/10-mobile-testing-types-and-approaches/>
- [2] <https://erictnd.medium.com/the-3-types-of-automated-tests-for-android-apps-ebc444fcf389>
- [3] <https://www.youtube.com/watch?v=cwYZDkflkwc&list=PLhPD b5zFmGR3qxa8JfmUdAfVH9Ou41MUx&index=5>
- [4] <https://www.calljacob.com/top-5-most-important-safety-features-for-cars/>

10. AUTHORS



Shrushti K. Vishwasrao

The Author is currently in the final year of Masters of Science, Computer Application Department from the Fergusson College, Pune, India. Author is passionate about studying topics about IoT, Android and Cloud storage.