SMART CAR MONITORING SYSTEM

1,2,3,4,5 Sona College of Technology, Department of Computer Science and Engineering, Salem, Tamil Nadu
1mohanraj134@gmail.com, 2mari.btech@gmail.com, 3dvbharathi77@gmail.com,
4theetchenya@gmail.com, 5gvidhyacse@gmail.com

ABSTRACT

One such study aims to build a low-cost gadget which allows vintage automobiles to surf the web and store sensor files in the database for subsequent processing and analysis, thereby transforming them into linked automobiles. The existence of equipment in a vehicle that link to connections, software, and activities are outside automobile through the Internet is known as an interconnected automobile. Even though there are many off-the-shelf OBD-II car processors on the market, the bulk of them will be generally horrendously overpriced or just faulty. In addition, nearly every one of these technologies are artificial constructs which can be enlarged or updated. The objectives of this work is to build a Raspberry Processor OBD-II vehicle monitor which might inform users out of an automotive upon the smartphone app. This should enable the participants to instantly check vehicle data from any location around the world. The final result must be inexpensive, dependable, and scalable. The system or any connected equipment must be easy to clean out from car. It ought to be capable to also be adaptable to many other cars with just little modifications, and it will enable for the installation of GPS modules, IMUs, and some other features.

I. INTRODUCTION

A linked car is someone who can interact between other devices beyond the vehicle in both directions (LAN). This enables the automobile to exchange online connection and, as a result, information with some other gadgets even within the car. Automobile manufacturers presently utilize 2 kinds of programs in linked vehicles: embedding and tether technologies. A Chained device now have technology which links to either the vehicle's cellphone, while an Integrated car will have a processor and constructed antenna. The network of hardware could share resources, receive system security patches, link to those other gadgets (Internet of Things), and give customers with Wi-Fi internet connectivity. The connected car telematics can also be accessed through connected network.

Figure.1 Global connected cars vs conventional cars
Even many intermediate automobiles are now considered Linked Vehicles 1.0 since about 2019. In certain instances, individuals offer: news and entertainment devices – a combination of navigation, autonomous driving, as well as a variety of many other technologies; and telematics – telematics such as automobile treatment (notices if tyre pressure is lower, for example) or fleet software are enacted in many other instances. Autonomous Devices 2.0 will be the next phase in the development of automobiles. Electric drivetrains must evolve beyond exclusive OEM platforms toward more accessible, standardised programs that enable users to integrate their preferred cable channels further into vehicle.

Vehicle diagnostics is a handy tool for identifying and fixing problems that may arise during the lifecycle of certain vehicle components. With the help of remote vehicle diagnostics, an expert can obtain an insight into the vehicle’s state and localize the problem without being physically present on-site. This linked automobile is often associated with the internet through an integrated chipset or SIM card, but it could browse the site as long as the wifi service is consistent. Automated cars could provide inside Wi-Fi, receive manufacturer-supplied over-the-air patches, and join various internet web services.

Users of a connected car will be able to connect to a range of pre media apps and services. Users may play music, watch the videos, or hear to online radio. Aside from it, the consumer or passenger may link their cellphone towards the vehicle's infotainment system via applications and manage the audio/video via afar. People will be able to locate the position of a linked automobile in real time. In the Indian market, connected car innovation will be in its infancy. Just some very linked automobiles also are available in the nation. The following are also the linked automobiles that are available at the moment in India: Hector, MG: The MG Hector is an SUV that was introduced in India in 2019. It has i-Smart technology and comes in petrol, diesel, and petrol-hybrid models. Seltos, Kia: In India, Kia released the Seltos SUV in 2019. UVO linked technology is included in the SUV. There are gasoline and fuel oil engines offered for the SUV.

II. LITERATURE SURVEY

Smart Vehicle Monitoring System using IOT [1 & 6] (Arduino based). These concept employs an acceleration technology to measure car smoothness & tremors with in event of a collision. This pulse is sent to the microprocessor as a result of all this. GSM and GPS modems were used to create a car alert system. Texts and alerts are delivered to both the specified mobile phone number. A Gps device, an Arduino, and a GSM modem make up this management system. The geolocation stationary signals satellite are received by the GPS receiver in the notification shade. This data can be processed by the Arduino, and the data collected is delivered towards the consumer via GSM modem. This heat sensor is used to measure the car's heat and the presence of noxious substances.

In [2 & 7] The Raspberry Pi (RPi) is used in the Connected systems installed in automobiles because it is familiar with sensors and can identify disasters quickly. The RPi too is equipped with cameras that may be used to measure the prevalence of an incident. This research employs a neural network picture categorization algorithm to determine the seriousness. Whenever problems occur, the SVMS identifies them right away and determines the gravity including its incident. This machine also will instantly notify the police. This proposal even included a GPS way to identify the cars' whereabouts in real time. Such information would be used to locate the car mostly in case of an injury or crime.

In [3 & 8] IoT Based Smart Car Monitoring proposed system is a car was monitored utilizing IoT, it showed the metrics alcoholic content, smoke threshold, distance between the object for dark spot recognition, rain strength, and brightness via Wi-Fi. Robotics is a term being used technology to describe the process of making a machine automated out of a manned state, allowing the settings to be controlled based on the current data. These platform's mechanization uses IoT to usually observed the car. Authentic vehicles would not have the same amenities as luxury sedan vehicles, therefore the suggested method brings support to owners at an affordable price. This suggested software is implemented at an affordable price and would be adaptable with all legitimate automobiles, equivalent to premium sedans automobiles, enabling pedestrian safety and communication monitoring.

Smart Vehicle Initiation and Surveillance System [4 & 9] is an IoT-based autonomous driving device that also informs the operator about the status about car, and thereby provides extra levels of protection to deter car theft. The document additionally suggests a concept to prohibit the car from commencing if the motorist has drank...
alcohol, as well as an urgent alarm system mostly in event of a crash. This project was created and built utilising a Raspberry Pi and Arduino combo, and it is very economical.

Internet of things technology enabling pretty much any time surveillance of automobile vehicles, [5 & 10], is an Internet of Things (IoT) technology on mobility that lets users to monitored car metrics from anywhere in the world over the Web. These study apply a car inspection and monitoring service that retrieved information from the car’s on-board diagnosing (OBD) systems. This interpersonal and inter wireless sensor network is the central aspect including its suggested Iot network of cars (one per vehicle).

III. PROPOSED SYSTEM

HARDWARE DESCRIPTION:

- Data is read from the OBD II port of the car and sent to the Raspberry pi via Bluetooth for processing.
- The data is processed by Raspberry pi and sent to a real-time cloud database (Google Firebase).
- The stored data is fetched using the react-native mobile app and insights are displayed.

![Figure 2 DATA FLOW DIAGRAM](image)

![Figure 3 BLOCK DIAGRAM & DESCRIPTION](image)
Hardware components:
- OBD II Bluetooth Interface
- Raspberry Pi
- GPS/GSM Module for internet

On-Board Diagnostics:
In some kind of a nutshell, OBD2 refers to a car's identity technology. The OBD2 connection makes it simple to get information via an automobile. The SAE J1962 specification offers two types of female OBD2 16-pin connectors (A & B). On Jan 1, 1996, all automobiles sold in the United States have been outfitted with OBD-II (On-board diagnostics II). Many automakers have adopt a certain standardized 16-pin connection, however they can select among five different OBD-II signal methods.

![On-Board Diagnostic pin configuration](image1)

The OBD2 Connector:
- The OBD2 connection is placed close the steering column, however this might be obscured by coverings or doors.
- It's not like all male connections are compatible with all OBD2 female sockets; verify the kind and OBD port pin-outs before purchasing.
- Pin 16 receives electricity from the automobile charger, which is typically used even when the light is turned down.
- Pins 6 (CAN-H) and 14 (CAN-L) seem to be the most important since CAN (ISO 15765-4) is common throughout many current automobiles (incl. EVs).

![Timeline graph of OBD](image2)

OBD2 is supported by nearly all modern automobiles, and the majority of them use the CAN protocol (ISO 15765). Even though an OBD2 connection with 16 pins is provided, european cars will not enable OBD2.
**OBD's function**

A simple OBD device comprises a power device, a networking of detectors, a central concepts, and displays, which together provide a full tracking system featuring regulated entry & usability. The elements of the OBD process are as follows:

ECU: A Electronic Control Unit, or ECU, is the heart of the OBD system. This ECU receives data from multiple sources of sensing devices across the automobile. This ECU next utilises such information either by regulate or check various aspects of the car, such as spark plugs.

Sensors: Detectors may be found in most every part of a car, from the engine and chassis to the power system core. Many of these technology offer signals towards the ECU that identify the signal strength origin and characteristics. Such message would then be "heard" and "understood" by the ECU.

DFN: When a sensors delivers data towards the ECU which is outside of the acceptable boundaries, the ECU records the data as a Diagnostics Fault Number, or DFN. The DFN code is simply a series of alphanumeric that reflect an issue's origin and type. DFN codes are typically standardised, however they might vary based on the make. The ECU transmits information first to electronic device whenever a DFN is stored, indicating that such a fault has now been discovered. The DFN can also be retrieved by connecting a sensors to the OBD connection.

MIL: Once the ECU receives a DFN code, it transmits information to something like the car dashboard, which causes the relevant adaptive headlights to illuminate. Such lamps, technically called as Defect Adaptive Headlights or DAHs, serve like an alert concerning car problems. Usually, when the light comes onto it and keeps on, the issue is small. When the light flashes, the situation is critical.

DLC: The Diagnostic Link Connector, or DLC, will receive many of the information and DFN codes gathered more by ECU. The DLC is just the car's way to get access for OBD devices, so it's usually placed underneath the console mostly on steering column, however it could also be placed in another models. This standardized OBDII protocol is used in today's automobiles, allowing every diagnostic equipment with just a type 2 cable to link towards the type 2 connection.

**On-board diagnostics Bluetooth Interface**

The Wireless OBD2 Analyzer is more than simply a reader. The analyzer has a developed Wireless network which allows a person to examine, analyse, and delete any OBD2 fault codes mostly in car's software applications. Device works including all OBD2-compliant automobiles as well as features 16-pin downlink ports. Every after diagnosis, the Scanning may display sensory information in real, and that's very beneficial to altering car engines as well as changing certain car parts. The monitor may also be used to gather data regarding performance, inlet viscosity, and gasoline level. The detector also displays the yield point, duration progresses, flow velocity, stress absorption, and spark plug readings, along with many other items.

**Raspberry Pi 4**

The Raspberry Pi is a line of limited, configurable processors which contain a variety of GPIO (General Purpose Input Output) ports for connecting & controlling other electronic items and creating Network of Thing services. The amount and function including its connectors varies by design, although they are typically separated as voltage, grounding, and particular pins. The majority of Raspberry Pi devices already equipped with a built connection. Designed Wireless & Wi-Fi are available on the Raspberry Pi 3, 3B+, Raspberry Pi Zero W, and Raspberry Pi 4.
IV. SOFTWARE DESCRIPTION

PYTHON MODULE PYOBD MODULE

pyOBD (also known as pyOBD-II or pyOBD2) is a Python-based freely available OBD-II (SAE-J1979) compliance diagnostic program. That's made to work using ELM 32x OBD-II testing connections, namely the ELM-USB. This should essentially let you to communicate also with vehicle's ECU, show trouble codes, present measurements, and conduct health checks, among other things. OBD links work on a request-response basis. To get statistics first from automobile, the consumer need to provide instructions to request again for map function (e.g. RPM, Vehicle speed, etc). This would be achieved also with query() method in Python-OBD. These instructions were recorded like classes inside obd.commands, which may have been searched via keyword or values.

Python-firebase module

This package may get all information using Json simply adding a.json extension towards the ending of Web address where information is stored and thereafter sending an HTTPS query via the web page. Firebase, like other REST APIs, allows the customer to change (PATCH, PUT), create (POST), or delete (DELETE) their stored data in addition to merely fetching information. In both synchronous and asynchronous modes, the app contains every one of the effective procedures for such activities. An asynchronous GET query containing a great commission as well as the procedure can indeed be started either by user/driver.

FIREBASE

Google's Firebase technology allows developers to create digital and social services. It started off as a stand-alone business in 2011. Google bought the framework in 2014, so it is now their main option for generation of test.

![Firebase Architecture](image)

Firebase is indeed a toolkit which allows anyone to "create, enhance, as well as expand their application." This contains a lot of functions which programmers will typically just had to create individually but don't want to because they'd instead work just on app selection. Statistics, identification, storage, customization, cloud storage, pushes communication, etc are all part of the above. Such applications generally cloud-based therefore scalable with very little work upon this creator's side. Firebase would be a fantastic method that retain sensitive information gathered just at user end, so this integrates very well Android Interfaces, that Raspberry Pi supports.

Many of the cellular and smartphone engineers have met many difficulty handling program in terms. Firebase could really assist in solving this problem and making it more manageable. It would be fascinating to watch how programmers exploit the disconnected capabilities of the platform. As just a solution, each client must conduct research locally and thereafter transfer that to the database once the internet is accessible. With all its mobile functionality, Firebase might enable this a lot easier for only a lot of programmers. Big Data, Identity, Clouds Messaging, Storing, Host, Test Lab, and Statistics are just a few of Firebase's capabilities, which utilise Authorization and Runtime Repository.
**REACT-NATIVE**

Facebook, Inc. produced Programming Language, an accessible smartphone app platform. It enables designers to leverage React's architecture alongside react native features to create android apps, Android TV, iOS, macOS, tvOS, Web, Windows, and UWP.

**Cross-platform**

React components use React's explicit Interface concept with Html to envelop preexisting test scripts and communicate with react Native. This opens up local content creation to altogether fresh teams of workers, as well as allowing current mobile players to work considerably more quickly.

![React Native Architecture](image)

**Figure.7 React-Native Architecture**

**Fast refresh**

See user changes as soon as user save. With the power of JavaScript, React Native lets users to iterate at lightning speed. No more waiting for native builds to finish. Save, see, repeat.

**Implementation**

Most fundamental concepts behind React Native were essentially equal to those of React, and therefore React Native does not have this Virtual DOM to control the DOM. It operates in the background upon that terminal and connects well with selected device through serialization data through an asynchronously and processed bridging (that understands the Js provided by programmers). It also opens up native mobile apps to totally different organizations of workers, as well as allowing current local staff to collaborate considerably more quickly. Whereas the language of React Native style is identical to that of CSS, it still does not utilise either HTML or CSS. Alternatively, native objects were manipulated via instructions from its JavaScript process. React Native further lets users make test scripts in languages like Java or Kotlin on Android and Objective-C or Swift for iOS, giving this more versatility.
SETUP AND IMPLEMENTATION

FIREBASE PROJECT SETUP
Click on Add project. This will bring up the Create a project, where user will need to provide a project name. Note that Firebase will generate a unique Project ID if user name is not unique across all projects. Select a Country where user would like database to be hosted. Click on Create Project. Click on project settings, and copy the credentials and firebase configuration for the web app. The below picture shows the configuration of my database.

REALTIME DATABASE
Firebase's database is NoSQL, which reduces restrictions whenever working on records and variables. This allows users to more easily construct as well as enhance databases. Database is stored under the format of JSON and to be synchronized with clients in real-time. The cross-platform client is the fundamental platform of this database which all clients share the same resource from Firebase server, and it will automatically update when any data is stored of changed.

VEHICLE INSTALLATION
In the vehicle's OBD port, plug in the OBD-II Wireless dongle. Plug a Raspberry Pi to something like the car's 12v energy source to charge it. BlueZ, the Wifi layer in Unix, will be launched. Attach With the: SPP Dev. and Pair + Trust user ELM327 Wireless Modem warning "Serial port connected to /dev/rfcomm0" should appear to a consumer. Launch the app after trying to ensure the Raspberry Pi gets hooked up to the internet.

SOME OF THE COMMANDS

<table>
<thead>
<tr>
<th>PID</th>
<th>Name</th>
<th>Description</th>
<th>Response Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>PIDS_A</td>
<td>Supported PIDs [01-20]</td>
<td>bitarray</td>
</tr>
<tr>
<td>01</td>
<td>STATUS</td>
<td>Status since DTCs cleared</td>
<td>special</td>
</tr>
<tr>
<td>02</td>
<td>FREEZE_DTC</td>
<td>DTC that triggered the freeze frame</td>
<td>special</td>
</tr>
<tr>
<td>03</td>
<td>FUEL_STATUS</td>
<td>Fuel System Status</td>
<td>(string, string)</td>
</tr>
<tr>
<td>04</td>
<td>ENGINE_LOAD</td>
<td>Calculated Engine Load</td>
<td>Unit.percent</td>
</tr>
<tr>
<td>05</td>
<td>COOLANT_TEMP</td>
<td>Engine Coolant Temperature</td>
<td>Unit.celsius</td>
</tr>
<tr>
<td>06</td>
<td>SHORT_FUEL_TRIM_1</td>
<td>Short Term Fuel Trim - Bank 1</td>
<td>Unit.percent</td>
</tr>
<tr>
<td>0A</td>
<td>FUEL_PRESSURE</td>
<td>Fuel Pressure</td>
<td>Unit.kilopascal</td>
</tr>
<tr>
<td>0B</td>
<td>INTAKE_PRESSURE</td>
<td>Intake Manifold Pressure</td>
<td>Unit.kilopascal</td>
</tr>
</tbody>
</table>
Vehicle’s diagnostics has gained much attention from industry and researchers in recent years. The variety and heterogeneity of vehicle diagnostics implementation has been the major reason which makes it interested. This project presents a technique to analyse diagnostics from vehicle that connected to OBD-II and process the diagnostics data using Raspberry Pi. Although the process occurs only the delivery of vehicle diagnostic data to user’s smartphone, Raspberry Pi is more suitable viewing the ability of Raspberry Pi that can be multitasking. We can include many more functionalities and features including music streaming, navigation, location tracking etc. to utilize the full potential of Raspberry Pi. This project opens a wide range of possibilities when we connect a car to the internet by making it available to all cars only with minor modifications in the program code.

### Table

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0C</td>
<td>RPM</td>
<td>Engine RPM</td>
</tr>
<tr>
<td>0D</td>
<td>SPEED</td>
<td>Vehicle Speed</td>
</tr>
<tr>
<td>0E</td>
<td>TIMING_ADVANCE</td>
<td>Timing Advance</td>
</tr>
<tr>
<td>0F</td>
<td>INTAKE_TEMP</td>
<td>Intake Air Temp</td>
</tr>
<tr>
<td>1F</td>
<td>RUN_TIME</td>
<td>Engine Run Time</td>
</tr>
<tr>
<td>21</td>
<td>DISTANCE_W_MIL</td>
<td>Distance Traveled with MIL on</td>
</tr>
<tr>
<td>23</td>
<td>FUEL_RAIL_PRESSURE_DIRECT</td>
<td>Fuel Rail Pressure (direct inject)</td>
</tr>
<tr>
<td>2D</td>
<td>EGR_ERROR</td>
<td>EGR Error</td>
</tr>
<tr>
<td>2F</td>
<td>FUEL_LEVEL</td>
<td>Fuel Level Input</td>
</tr>
<tr>
<td>33</td>
<td>BAROMETRIC_PRESSURE</td>
<td>Barometric Pressure</td>
</tr>
<tr>
<td>46</td>
<td>AMBIANT_AIR_TEMP</td>
<td>Ambient air temperature</td>
</tr>
<tr>
<td>5C</td>
<td>OIL_TEMP</td>
<td>Engine oil temperature</td>
</tr>
<tr>
<td>5D</td>
<td>FUEL_INJECT_TIMING</td>
<td>Fuel injection timing</td>
</tr>
<tr>
<td>5E</td>
<td>FUEL_RATE</td>
<td>Engine fuel rate</td>
</tr>
</tbody>
</table>

### V. RESULTS AND DISCUSSIONS
This project introduces a unique way to monitor user’s car from anywhere at any time using a cross-platform mobile application (runs both on IOS and Android). This mobile app will provide data about the car which includes Speed, RPM, Fuel Rate, Location, and many more engine specific details. In the present work the OBD has been used to read the diagnostic codes and other data from the car and a mobile app is developed to present the data to the user. Additionally, we plan to include notifications features which will be triggered when the fuel level drops below the reserve level when the car reaches a specific speed and, when there is a fault in the engine.

**VI. CONCLUSION**
REFERENCES