KNOWLEDGE INSTITUTE OF TECHNOLOGY

NAAN MUDHALVAN PROJECT

PROPOSAL

Domain: INTERNET OF THINGS (IOT)

Presented for :

IBM

Presented by :

TEAM : 544

- JEEVABHARATHI R
- SUJITH G
- KEERTHI S
- GOKUL S

📅 01-11-2023

📍 Salem

TRAFFIC MANAGEMENT SYSTEM USING IOT, DATA ANALYTICS AND MACHINE LEARNING

Introduction :

Urban areas are grappling with the pressing problem of traffic congestion, leading to a web of interrelated issues including heightened pollution levels, safety hazards, and economic setbacks. As urban populations swell, discovering effective and sustainable traffic management solutions becomes increasingly urgent. The integration of cutting-edge technology, data analytics, and the promotion of eco-friendly transportation alternatives stands as a vital approach to tackling these challenges and fostering the development of more habitable and environmentally conscious cities.

Project Definition

The Traffic Management project aims to leverage IoT devices and data analytics to monitor traffic flow and congestion in real time. By providing commuters with access to this information through a public platform or mobile apps, the project intends to assist them in making informed decisions about their routes, ultimately alleviating traffic congestion. The project involves defining clear objectives, designing the IoT traffic monitoring system, developing the traffic information platform, and integrating them using IoT technology and Python.

Design Thinking

Project Objectives:

To ensure the success of this project, it is essential to define clear and achievable objectives. These objectives will guide the development and implementation phases:

Real-time Traffic Monitoring: Implement a network of IoT sensors to monitor traffic conditions continuously in real time. Congestion Detection: Utilize data analytics and machine learning techniques to detect traffic congestion and bottlenecks promptly.

IOT Sensor Design:

The effective deployment of IoT devices (sensors) is critical to gathering accurate traffic data. The following steps outline the IoT sensor design:

1. Sensor Selection: Choose appropriate IoT sensors, which may include cameras, GPS devices on vehicles, and other relevant sensor technologies.
2. Strategic Placement: Identify key locations on roadways where sensors should be placed. Locations should be selected based on traffic density and congestion-prone areas.
3. Data Collection: Implement data collection mechanisms for the selected sensors. Cameras will capture visual data, while GPS devices on vehicles will send real-time location information to the cloud.
4. Sensor Crowdsourcing: Develop algorithms to aggregate data from multiple vehicles with GPS sensors to detect congestion based on a predefined threshold of slow-moving vehicles in a specific area.

Real-Time Transit Information Platform:

A user-friendly web-based platform and mobile apps are crucial for providing real-time traffic information to the public. The design of these components involves the following steps:

1. User Interface (UI) Design: Create an intuitive and user-friendly interface for both the web platform and mobile apps. Include features such as traffic maps, route recommendations, and real-time updates.
2. Database Design: Design a robust database system to store and retrieve traffic data efficiently. This database will feed real-time information to the platform and apps.
3. Real-Time Data Integration: Establish a data pipeline that connects the IoT sensor data to the platform and apps, ensuring that users receive the most current information.
4. API Development: Develop APIs to allow third-party developers to access and integrate traffic data into their applications and services, promoting ecosystem growth.

Integration Approach:

To achieve a seamless integration between the IoT traffic monitoring system and the real-time transit information platform, the following approach will be adopted:

1. Data Processing: Implement data processing pipelines that transform raw sensor data into meaningful insights, including congestion detection and route optimization.
2. Machine Learning: Utilize machine learning algorithms for congestion detection and predictive modelling to improve route recommendations.
3. IoT Connectivity: Establish secure and reliable connections between IoT sensors and the cloud infrastructure for real-time data transmission.
4. Scalability: Design the system with scalability in mind to accommodate future expansion and increasing data volumes.
5. User Feedback: Incorporate user feedback mechanisms to continuously improve the accuracy and usability of the system.

OVERVIEW (PLAN)

- ● User 📱
- – – – Alternative route on App
- – – – Road is under congestion
- ● Destination
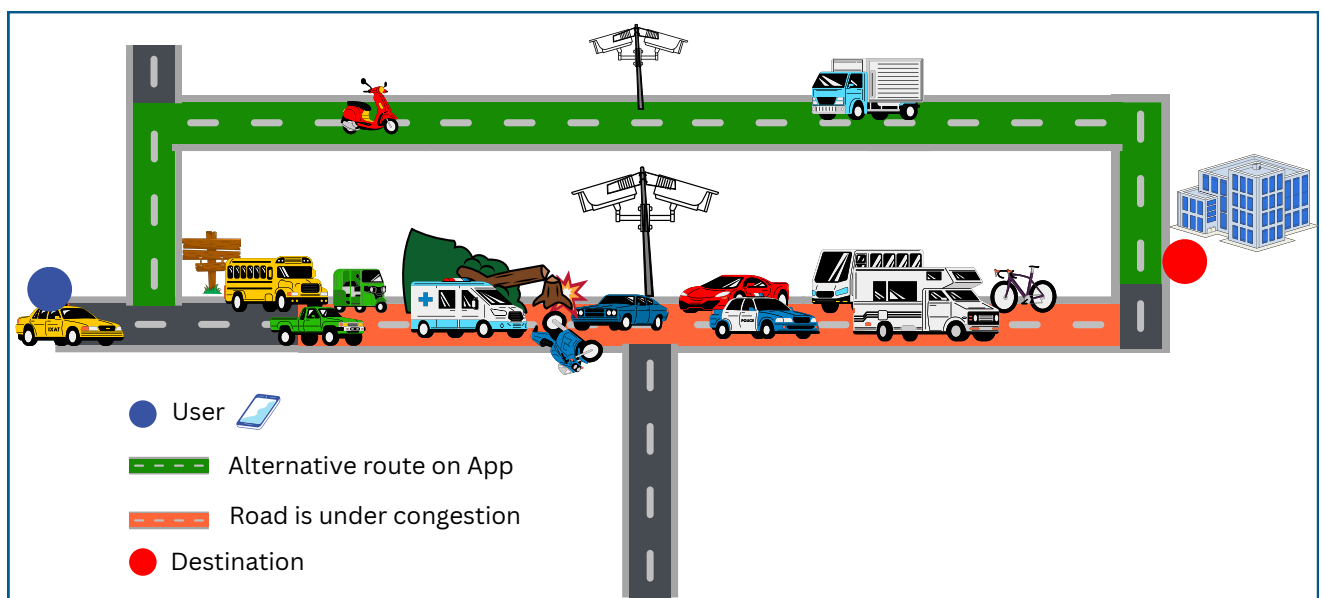
TOOLS AND PLATFORMS :

- **ESP32_CAM** : ESP32 camera module to detect vehicles on the road for the purpose of monitoring traffic.

- **OpenCV :** Using image processing techniques with libraries like OpenCV to perform object deduction to find the congestion area

- **Firebase :** The current and Historical Traffic data are stored in Firebase to carry the traffic information to the central servers like GCP , Heroku etc.

- **Heroku :** The deployed machine learning model in the Heroku server retrieves the traffic data from Firebase to perform machine learning operations and route prediction activity and send required data to the API

- **GMap :** The Heroku API provides the predicted optimal route, which is then visualized on the UI of both the app and the website.

- **UI/UX :** Designing a seamless and user-friendly UI/UX that ensures a smooth and enjoyable experience for app and website users.

CONCEPT FLOW
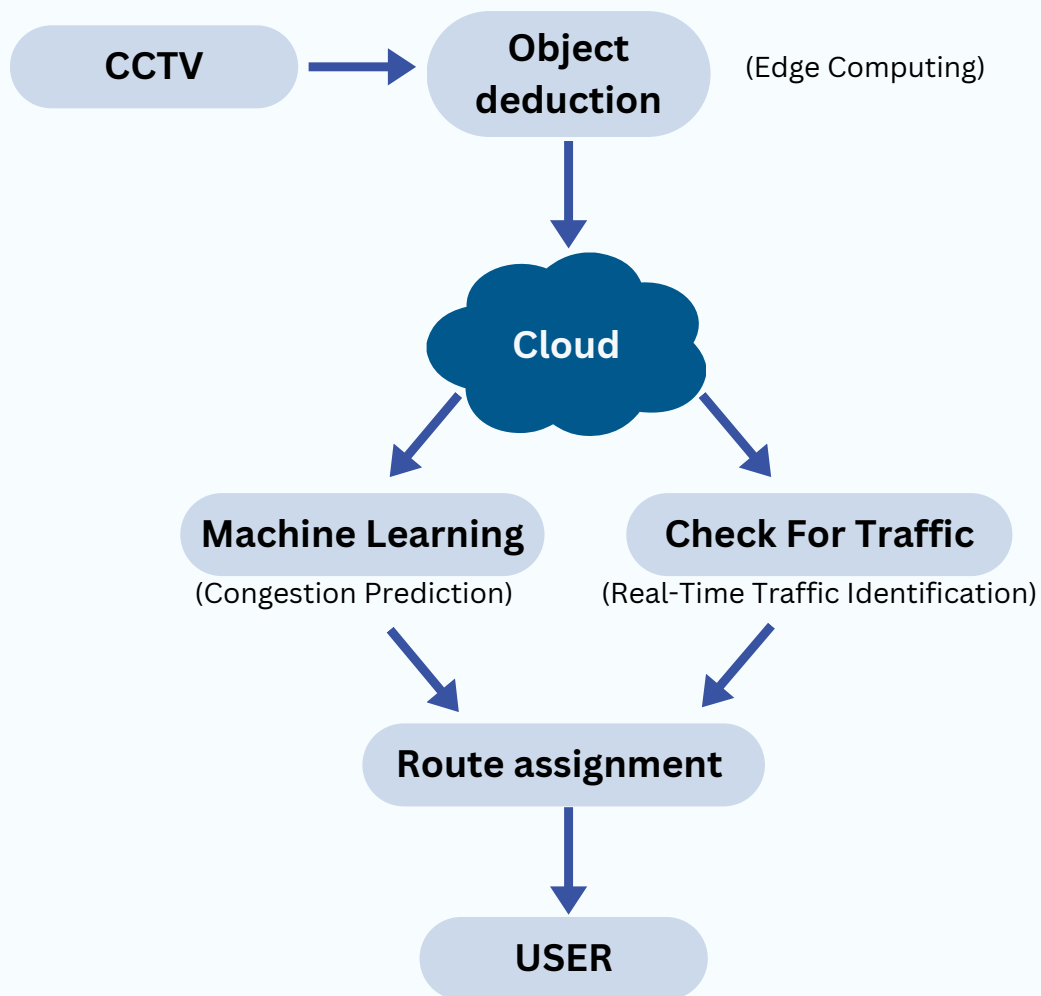
CHALLENGES :

- Variations in image quality due to different lighting conditions can affect the accuracy of vehicle detection
- Developing an efficient and reliable solution using the ESP32 Cam presents challenges related to real-time image processing, data transmission due to the Low Processing power.
- Reliable Wi-Fi or other network connectivity is required for data transmission, which can be challenging in some environments.
- Optimizing power usage for battery-operated ESP32 Cam devices, especially for continuous data collection, is a consideration.

OBJECTIVES:

The objective of the project is to provide real-time traffic data to users, assisting them in selecting the most efficient route to their destination. Additionally, the project aims to utilize machine learning to predict congestion, offering suggestions that incorporate parking availability when it is needed.

COMPONENTS DESCRIPITION:

A traffic management system circuit typically involves the use of electronic components to control and monitor traffic flow.

POWER SUPPLY :

- The bridge rectifier is used to convert alternating current (AC) from the electrical grid or another source into direct current (DC), which is often used to power various components of the system.
- The converted current is pulsated, Using capacitors as a filter to stabilise the DC
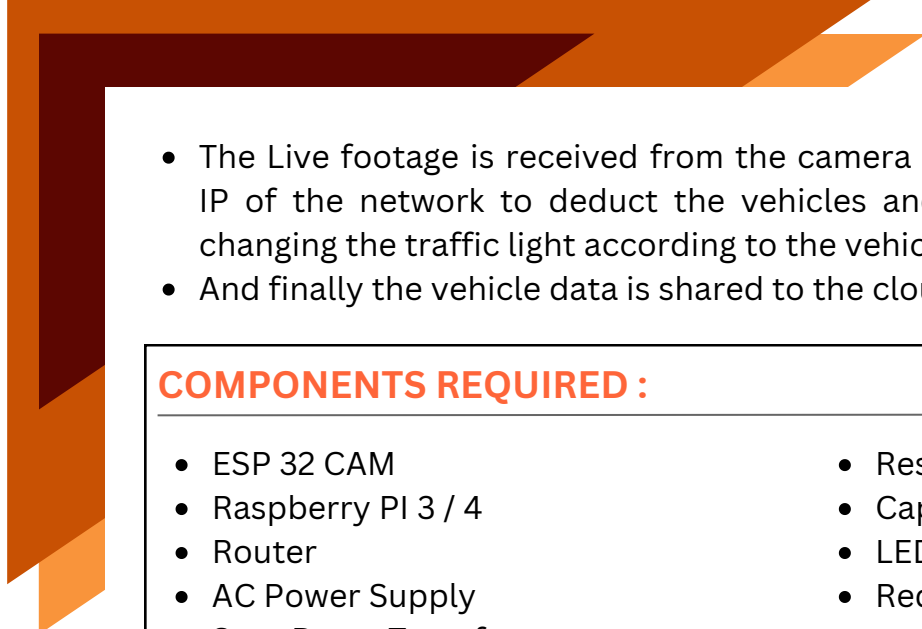
SENSORS :

- In traffic management projects, sensors are vital for data collection and monitoring. Common sensor types include inductive loops, magnetic sensors, video cameras, infrared sensors, and radar sensors.
- In this project Camera sensors are used to Identify vehicle congestion and change traffic lights accordingly.

BOARDS :

- The **ESP32 CAM** is programmed to capture images and videos and It can process the captured data locally by sending the live stream to the local server.
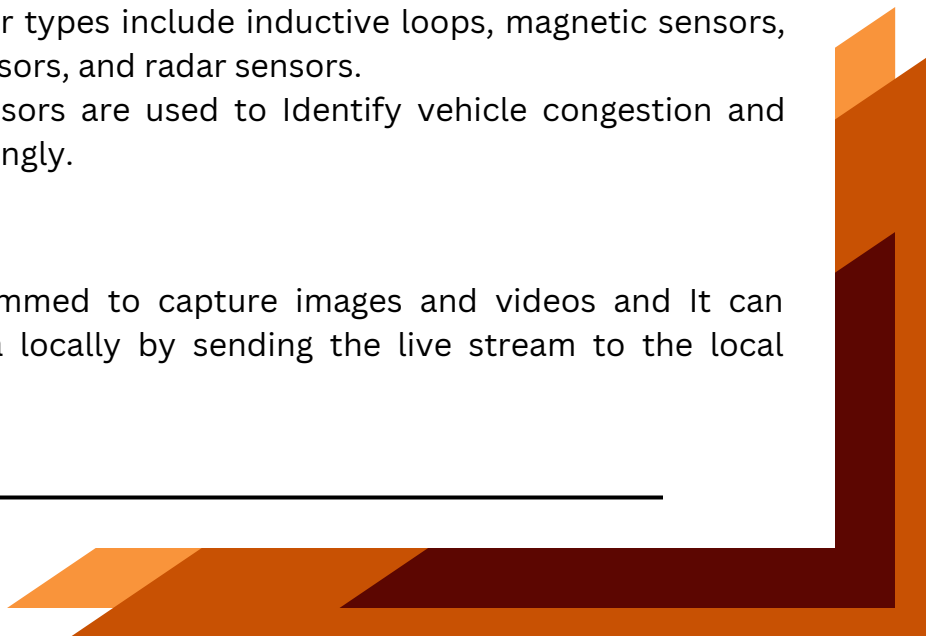
- The Live footage is received from the camera by **Raspberry PI** using the local IP of the network to deduct the vehicles and perform traffic clearance by changing the traffic light according to the vehicle data.
- And finally the vehicle data is shared to the cloud (firebase).

COMPONENTS REQUIRED :

- ESP 32 CAM
- Raspberry PI 3 / 4
- Router
- AC Power Supply
- Step Down Transformer
- Diode

- Resistor
- Capacitor
- LED : Red , Green , Yellow
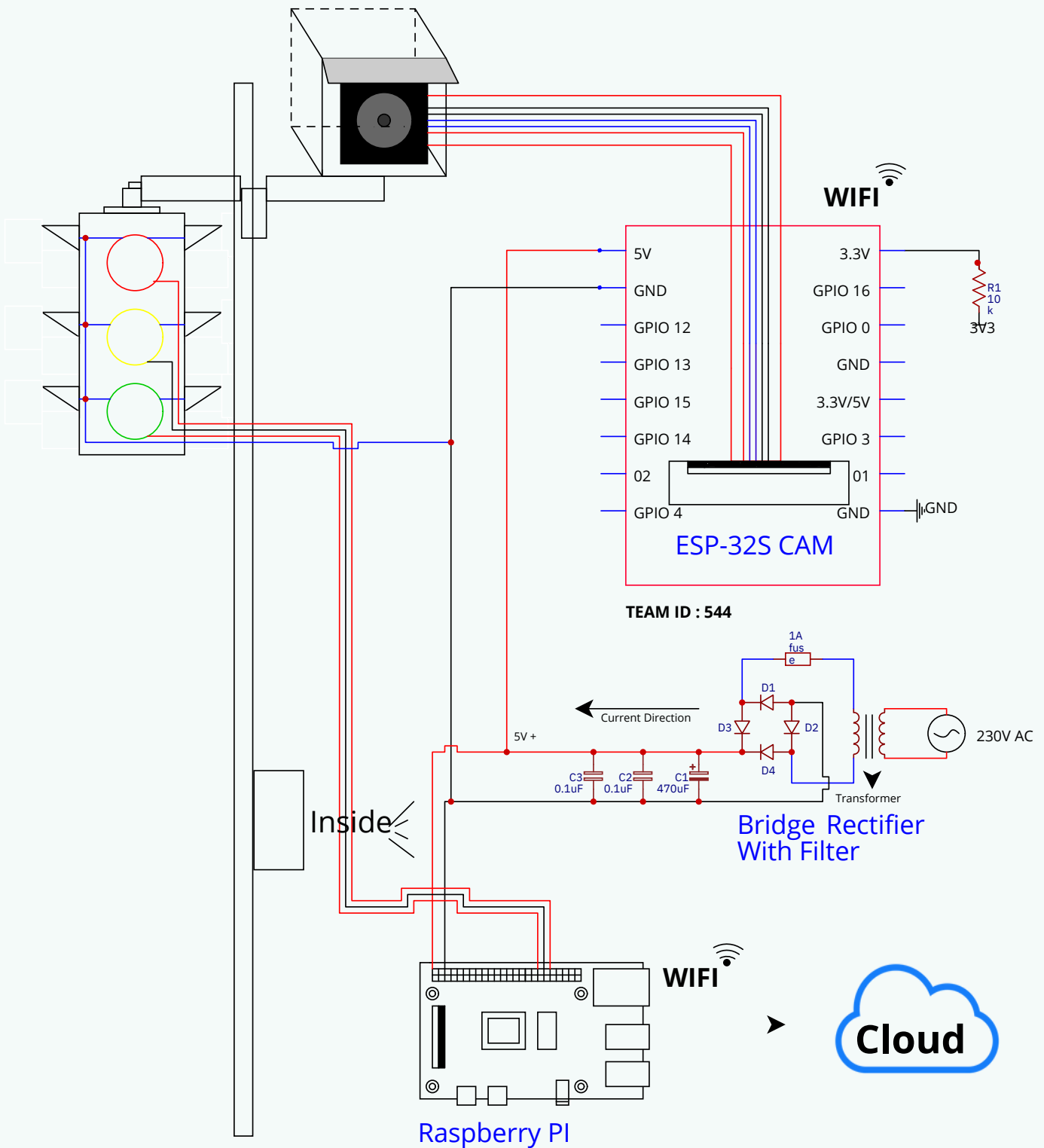- Required amount of wire

---

The Stimulation for ESP 32 Cam is a little complicated due to the unavailability of the Right Software that performs and supports the camera sensors. So thereby we developed a schematic approach to view and understand the circuit design.

CIRCUIT

The program written in ESP32 board for streaming video through IP is shared in the Github repository.

WIFI

ESP-32S CAM	
5V	3.3V
GND	GPIO 16
GPIO 12	GPIO 0
GPIO 13	GND
GPIO 15	3.3V/5V
GPIO 14	GPIO 3
02	01
GPIO 4	GND

R1
10 k
3V3

GND

ESP-32S CAM

TEAM ID : 544

1A fuse

D1

Current Direction

D3   D2

5V +   D4

C3   C2   C1
0.1uF   0.1uF   470uF

Transformer

230V AC

Bridge Rectifier With Filter
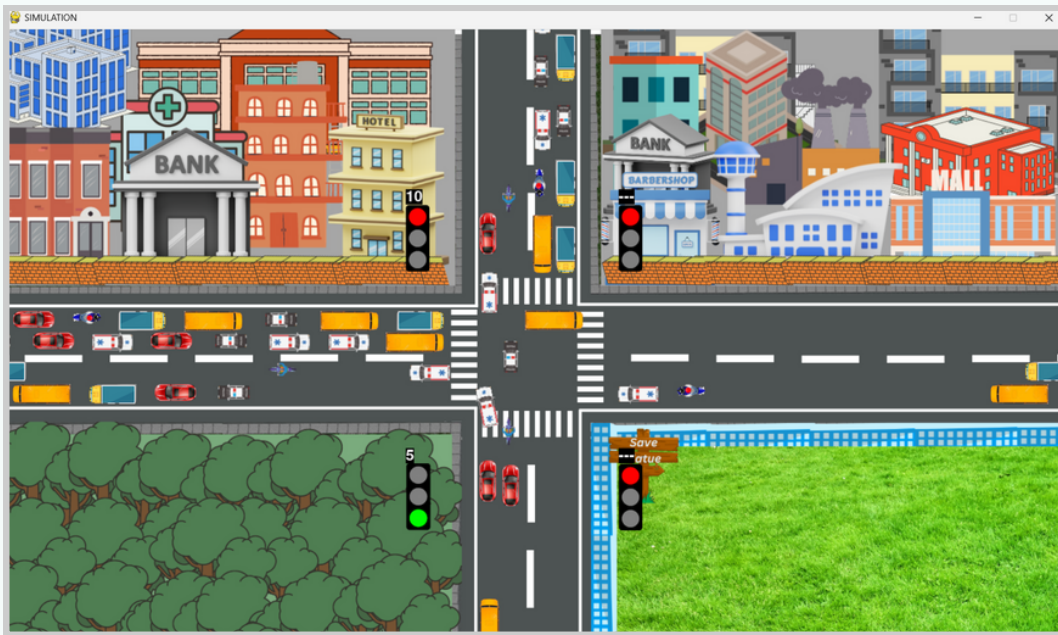
Inside

WIFI

Raspberry PI

Cloud

SIMULATION :

- The Simulation for this project is made by using Python. From this simulation we will understand the actual working of this project under the single area.
- Pygame Library is used to make the simulation.

THE PROGRAM AND ASSETS FOR THIS SIMULATION IS ATTACHED ON PHASE 5 ON REPOSITORY. (SIMULATION.PY / IMAGES)

OUTPUT :

VEHICLE DEDUCTION :

- Image processing is used to deduct the vehicle on the road to assign the timing of red and green lights and also for the App's congestion data.
- **Open Cv**: The "Open Source Computer Vision Library," is a crucial tool designed specifically for computer vision applications tailored to traffic management.
- In **Python**, with a comprehensive set of functions and algorithms for visual data processing, OpenCV facilitates real-time monitoring of traffic conditions, offering valuable insights for data analysis and decision-making.
- Firstly the original frame is taken from the **IP** of the **ESP 32 CAM** and processed by **Raspberry Pi.**
- The code starts by importing the necessary libraries for Image Processing and its dependencies.
- Pillow, opencv-python, requests.
- **Pillow** is a well-known which is used in image processing, Image creation, Image enhancement, Image analysis and image conversion.
- The **requests** library helps Raspberry PI to interact with the cloud and receive frame shared in a localhost by the camera.
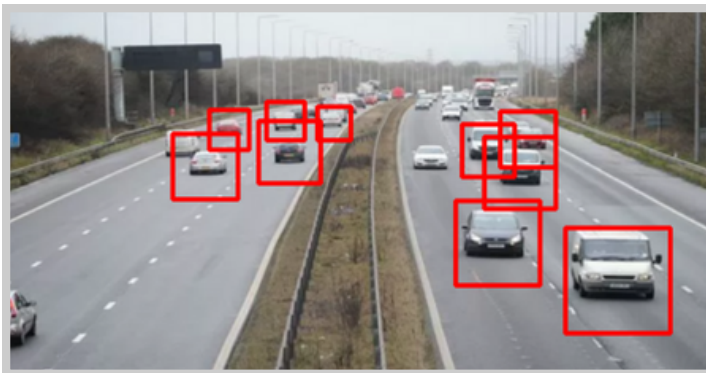
INPUT:

- Link of the Image or video frame callback with the vehicles. It works in both Local and internet streams.

OUTPUT:

- After Deductiong the Vehicles.

USES AND ADVANTAGES:

- **License Plate Recognition (LPR):** It can capture license plate images for automated license plate recognition.
- **Incident Detection:** Detect and capture images of accidents, road obstructions, or traffic violations.
- **Remote Accessibility:** It allows authorized users to access traffic data from anywhere, enabling real-time monitoring, analysis, and decision-making.
- Data can be accessed by traffic management authorities, **law enforcement,** and **urban police officers,** enabling informed decision-making.

- **Scalability and Redundancy:** Cloud storage can scale to accommodate changing data volumes and provide redundancy, ensuring data reliability even in the event of hardware failures.
- It provides enhancement in **road safety** and traffic efficiency by reducing the waiting time to **save fuel**

MAIN OBJECTIVES:

- Using automated technology, the traffic light adjusts automatically to prioritize the passage of the **emergency vehicle**, swiftly clearing the traffic and ensuring a seamless route. This system effectively optimizes the flow of traffic, allowing the emergency vehicle to **proceed rapidly** without any obstructions to save lives.
- In addition, the system aims to offer users a seamless experience through an app that **predicts congestion**. This is achieved by leveraging recorded data and employing a **machine-learning** algorithm. Furthermore, the system provides a low-latency live stream of the traffic in the area, enhancing the user experience with an excellent interface.

CONGESTION PREDICTION:

Congestion prediction in traffic management involves using data and technology to anticipate and manage traffic congestion effectively. Here are some key methods and techniques used for congestion prediction:

- **Data Collection:** Gathering real-time data from traffic cameras, Google map API for Traffic data and from the location of mobile apps, to monitor traffic conditions and collect relevant information.
- **Machine Learning:** Utilizing machine learning algorithms to analyze historical and real-time traffic data, weather conditions, and special events to predict congestion patterns.
- **Predictive Analytics:** Using statistical techniques and historical data to forecast congestion during peak hours or specific events.
- **Dynamic Routing:** Providing drivers with real-time route recommendations to avoid congestion-prone areas.

There are several Machine learning algorithm that makes prediction much easier such as
- Linear regression
- Logic regression
- Support vector machines
- Decision trees etc.

LINEAR REGRESSION ALGORITHM:

Linear regression is a supervised machine learning algorithm used for predicting a continuous target variable based on one or more input features. It assumes a linear relationship between the inputs and the target.

KEY COMPONENTS OF LINEAR REGRESSION:

- **Linear Equation:** The linear regression model is represented as: **$y = mx + c$**, where **y** is the target variable, **x** is the input feature, **m** is the slope (weight), and `b` is the intercept (bias).
- **Least Squares Method:** Linear regression typically uses the least squares method to find the values of **m** and **b** that minimize the sum of squared errors between the predicted and actual values.
- **Simple Vs Multiple Linear Regression:** In simple linear regression, there's one input variable, while in multiple linear regression, there are also some multiple input variables.
- **Assumptions:** Linear regression assumes that the relationship between the input features and the target variable is linear, that the errors are normally distributed and have constant variance, and that the input features are not highly correlated.
- **Evaluation:** Common metrics for evaluating linear regression models include Mean Squared Error (MSE), R-squared (R^2), and others to assess the model's accuracy.

DATASET :

The data set is automatically generated with the help of Raspberry Pi which deducts the vehicle and changes the traffic lights accordingly. The following is just a sample of data for one month January with recorded data on vehicle count and Festivals of that month to make prediction easier.
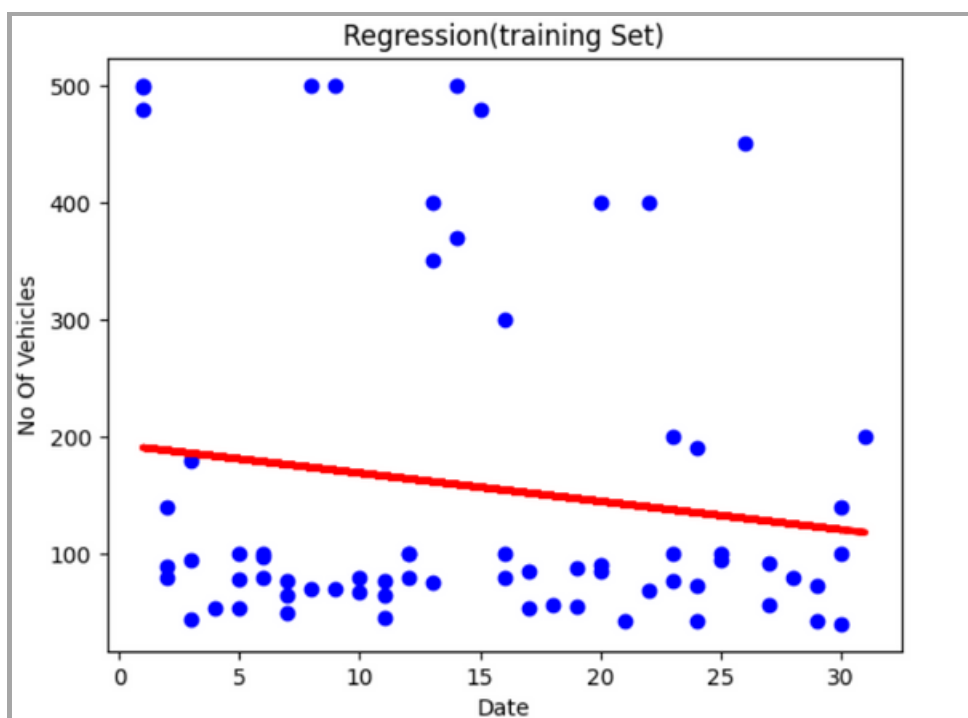
THE PROGRAM AND DOCUMENT FOR THIS VEHICLE PREDICTION ARE ATTACHED ON THE PHASE 5 REPOSITORY.

(CONGESTION_PREDICTION.IPYNB / CONGESTION_PREDICTION.PDF)

Linear regression :- Data Virtualization. (original data)

DATE vs VEHICLE_COUNT
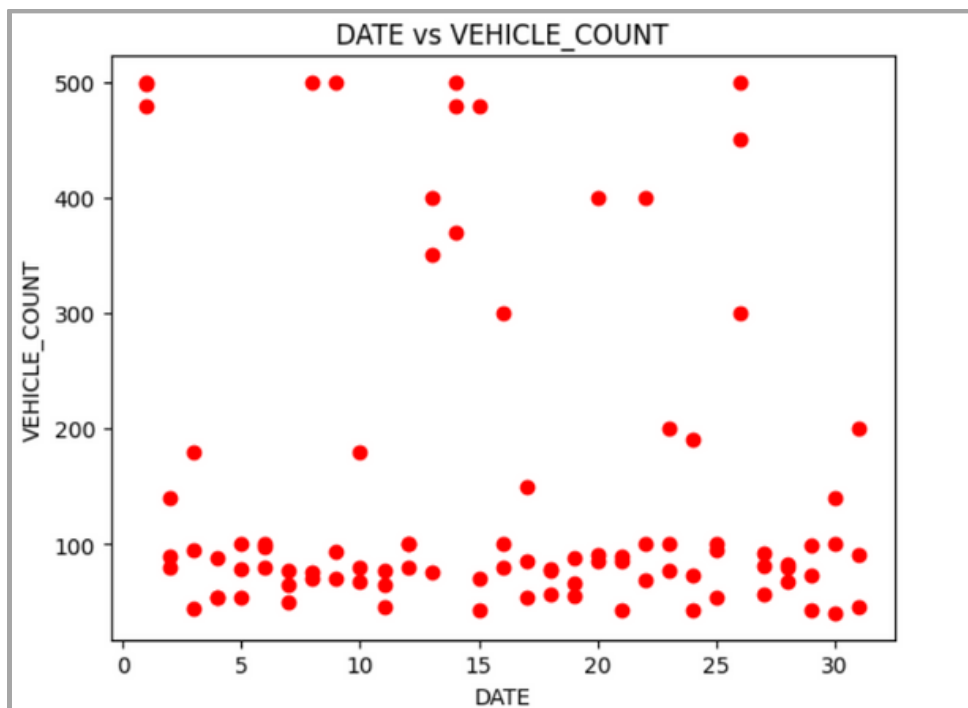
Linear regression :- Data Virtualization. (Trained Data)

Regression(training Set)

Linear regression :- Full CSV Data. (Un sorted Data)

	YEAR	MONTH	DATE	FESTIVAL	VEHICLE_COUNT
0	2021	January	1	new_year_day	500
1	2021	January	2	_	80
2	2021	January	3	sunday	180
3	2021	January	4	_	88
4	2021	January	5	_	100
...	...	...	...	...	...
88	2023	January	27	_	57
89	2023	January	28	_	67
90	2023	January	29	sunday	99
91	2023	January	30	_	100
92	2023	January	31	_	45

93 rows × 5 columns

Moreover, enhancing the training data would yield more precise outcomes. During festive seasons, the likelihood of congestion increases. To mitigate this issue, consider training the model with specific dates, festival names, and the total count of occurrences

Consider pongal as the festival and the trained data set is as follows.

	YEAR	MONTH	DATE	FESTIVAL	VEHICLE_COUNT
13	2021	January	14	pongal	480
44	2022	January	14	pongal	500
76	2023	January	15	pongal	480

To predict the congestion on January 2024.

```
def vehicle_cnt(a):
    result = regressor.predict(np.array(a).reshape(1, -1))
    return(result[0,0])
print("It is possible to travel",int(vehicle_cnt(4)), end="")
print(" Vehicles During Pongal 2024")

It is possible to travel 490 Vehicles During Pongal 2024
```

Please refer above content in the PDF (Congestion_Prediction.pdf) for more details.

The predicted data is send through the API from the **HERO KU** platform and received on the user app.

APPLICATION DEVELOPMENT:

The App is developed by using Kotlin as a primary language and interfacing heroku and firebase to get real time data in traffic deduction and prediction.

Kotlin: a statically typed language, is extensively utilized in application development, especially for Android, offering enhanced null safety and versatility for various application types.

Firebase : The current and Historical Traffic data are stored in Firebase to carry the traffic information to the central servers like GCP , Heroku etc.

Heroku : The deployed machine learning model in the Heroku server retrieves the traffic data from Firebase to perform machine learning operations and route prediction activity and send required data to the API

The Following Screenshot represents the User interface of our app Home screen and Traffic map screen

THE APP AND RELEVANT INFORMATION ABOUT THE RUNNING PROCESSES ARE SHARES IN THE PHSAE_5 FOLDER ON GITHUB REPOSITORY.

APP NAME : **Map My Route** Fig : Home Screen and Traffic Map Screen as follows.
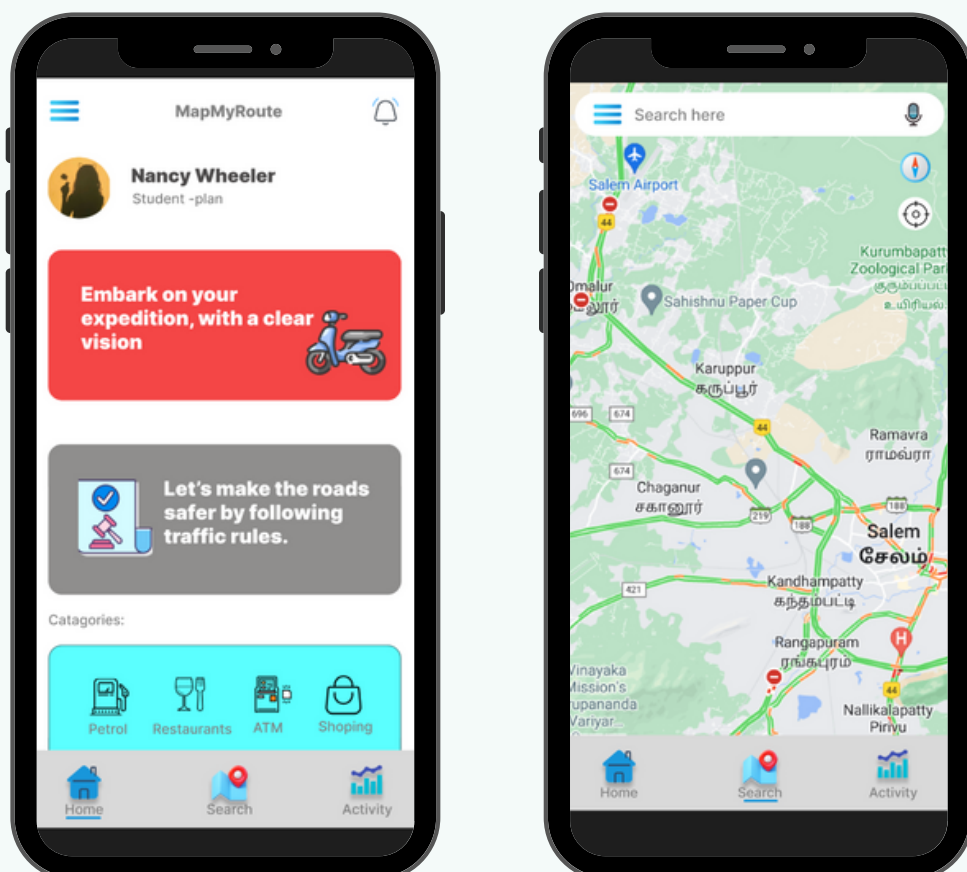
- The app is made by using Android Studio with Kotlin as its primary functioning language And used XML Layouting for its user interface.
- Using Google Map API, and Prediction algorithm the app can provide traffic congestion on the map.
- In addition to that user can able learn traffic rules and guide lines.

Conclusion

The success of the Traffic Management project is dependent on defined objectives, strategic IoT sensor deployment, user-friendly platform development, and seamless IoT and Python analytics integration. This initiative intends to improve commuting and minimize congestion for everyone. And save lives by preventing accidents and catagorizing the vehicle to perform traffic signal changes for Emergency Vehicles.