ECEN430: Selected Topics in Computer Engineering.

Cloud connected driver notification system for real time road conditions

Ahmed Ayman	2020000689
Mohamed Shawky	202000218
Mohab Hassan	202001315
Mohab Mohamed	202000083
Mohamed Attia	202002014
Zeyad Hesham	202002027

Supervised by:
Dr. Bishoy Wasfey
Eng. Salma Zakzouk

Table Of Contents

Table Of Figures

Introduction

This project integrates socket programming with a client-server communication system and an AI model. The server uses sockets to receive GPS data from the client, process it, and send periodic heartbeats to the AI for status monitoring. The AI leverages inter-process communication via message queues to monitor the server's status and perform predictions when the server is down. This system showcases the seamless combination of real-time data handling, socket programming, and AI-driven decision-making, making it an efficient and dynamic solution for advanced applications.

Problem statement

Driving on the roads with unpredicted and dynamic changes causes big challenges for drivers. Problems and issues such as weather changes, accidents, traffic and roadblocks can put the driver's safety in danger and efficiency. The main challenge is the lack of time and the exact location of such issues and conditions that may occur causing many problems such as higher risk of accidents, traffic delay and higher fuel consumption.

Proposed solution

To address such a problem, we proposed and implemented a cloud-connected system that provides the driver with real-time road information and updates. The system uses a Raspberry Pi as the main microcontroller, along with a GPS module and cloud-based analytics, to deliver accurate and timely notifications to drivers via a custom dashboard. The dashboard will display the speed limit and road conditions. Additionally, the system includes a chatbot to assist drivers with any queries. By using the data and sources implemented into the system, the system provides the driver with all necessary information such as traffic road signs, speed signs, and upcoming road constructions. All of this is achieved through the cloud. To enhance the system's reliability, we aim to Have an AI-powered traffic sign recognition model that can function autonomously in situations where the cloud connection is unavailable or disrupted. This model, deployed directly on the Raspberry Pi, will process live camera feeds to detect and classify traffic signs in real-time. This ensures that critical road information, such as speed limits and hazard warnings, remains accessible to the driver even in the absence of cloud connectivity. By integrating AI-based local recognition, the system achieves greater robustness and enhances driver safety under various conditions.

Methodology

For our hardware setup, the Raspberry Pi serves as the central microcontroller, interfaced with a GPS module and a camera. The GPS module continuously gathers critical data, including the real-time location of the vehicle, such as latitude and longitude. This data is transmitted to Raspberry Pi, which establishes a reliable internet connection to exchange information with a designated cloud service.

Cloud service plays an important role in analyzing and processing the data received from the GPS module. Once processed, the refined data is sent back to the Raspberry Pi, which then communicates with the custom dashboard. The dashboard provides the driver with clear and timely visual notifications about upcoming changes on the road, such as speed limits, construction zones, or other traffic-related updates.

To enhance system functionality and provide a failsafe mechanism, an AI-powered traffic sign recognition model, developed using the advanced YOLOv8 framework, is integrated into the Raspberry Pi. Supported by a connected camera, this model autonomously detects and classifies road signs in real-time. Utilizing a pre-trained dataset, the YOLOv8 model processes the camera feed with high accuracy, ensuring reliable identification of traffic signs even in scenarios where cloud connectivity is lost. This guarantees uninterrupted delivery of critical road information to the driver, safeguarding against potential network or system failures.

By combining the robust performance of cloud-based analytics with the efficient, on- device capabilities of the YOLOv8 AI model, our system offers a comprehensive solution for real-time road navigation and traffic management. This dual approach enhances both safety and reliability, ensuring drivers are well-informed and supported under all conditions

Workflow

Microcontroller Configuration

We began by setting up and configuring the Raspberry Pi microcontrollers, ensuring proper communication and connectivity for seamless data processing.

Cloud Service Integration

We utilized a Neo-6M GPS module connected to a Raspberry Pi to retrieve real-time latitude and longitude data. This data is then sent to ThingSpeak using an HTTP POST request, enabling cloud-based data storage and visualization. The GPS readings are periodically updated and transmitted, allowing continuous tracking of location. The cloud platform integrates with Google Maps for visualizing the GPS data, providing a real-time location display.

AI Model Implementation

The system integrates an advanced YOLOv8 AI model with a Raspberry Pi for real-time traffic sign recognition. The YOLOv8 model, trained on a pre-annotated traffic sign dataset, is optimized for edge devices and deployed locally on the Raspberry Pi. Equipped with a Raspberry Pi Camera Module v1.3, the system captures and processes road imagery in real-time.

The YOLOv8 model ensures accurate detection and classification of traffic signs, with performance optimizations enabling smooth operation on the Raspberry Pi's hardware.

essential libraries PyTorch, OpenCV, and the Ultralytics YOLOv8 package were installed on the Raspberry Pi, enabling seamless integration and operation of the AI model.

Data Processing in the Cloud

The cloud service processes incoming data from the GPS module. The processed results are then transmitted back to the microcontroller for local use and display.

Connections:

VCC → 3.3V or 5V on RPi

GND → GND on RPi

TX → GPIO15 (UART RX)

Driver Notifications

We configured data streams to handle real-time updates of GPS data, specifically latitude and longitude, by setting up virtual channels for efficient processing. These data streams were configured as Float data types to ensure high decimal precision. A Python script on the Raspberry Pi continuously transmitted the location data to the cloud, enabling seamless real-time communication. A custom dashboard was utilized to display these values clearly and accurately. Additionally, notifications were integrated to alert the driver about road conditions, traffic signs, and other relevant updates, enhancing the overall driving experience.

System Design

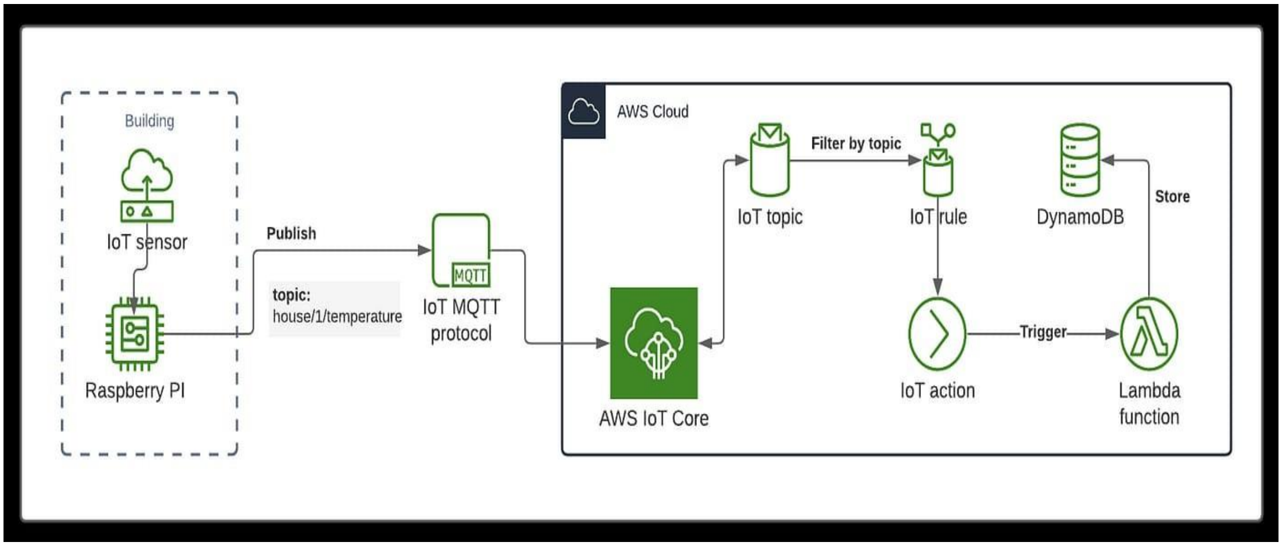

Figure 1

Results

Training and Testing of AI model :

This is the Precision confidence and recall curve.

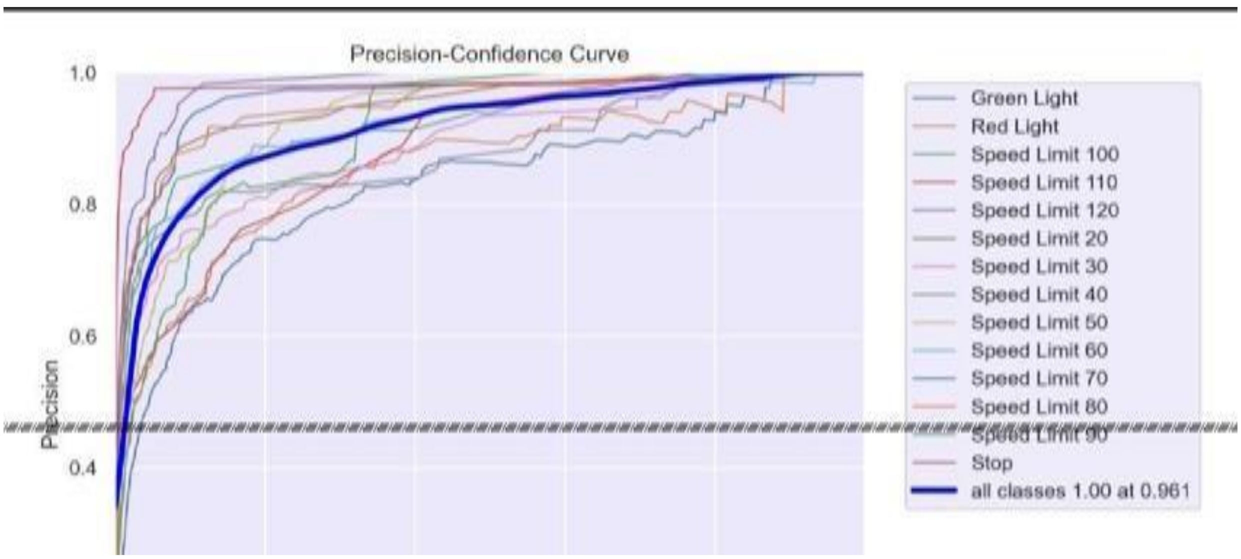

Figure 2

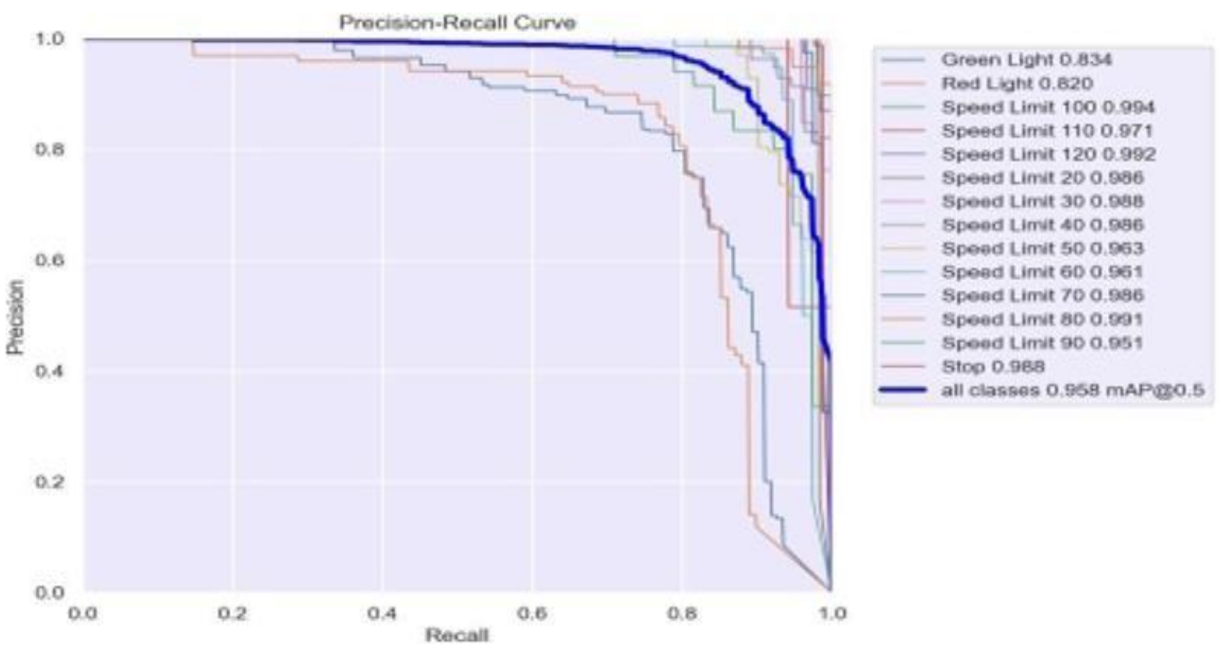

Figure 3

The Confusion matrix

Figure 4

Testing Trained model on camera

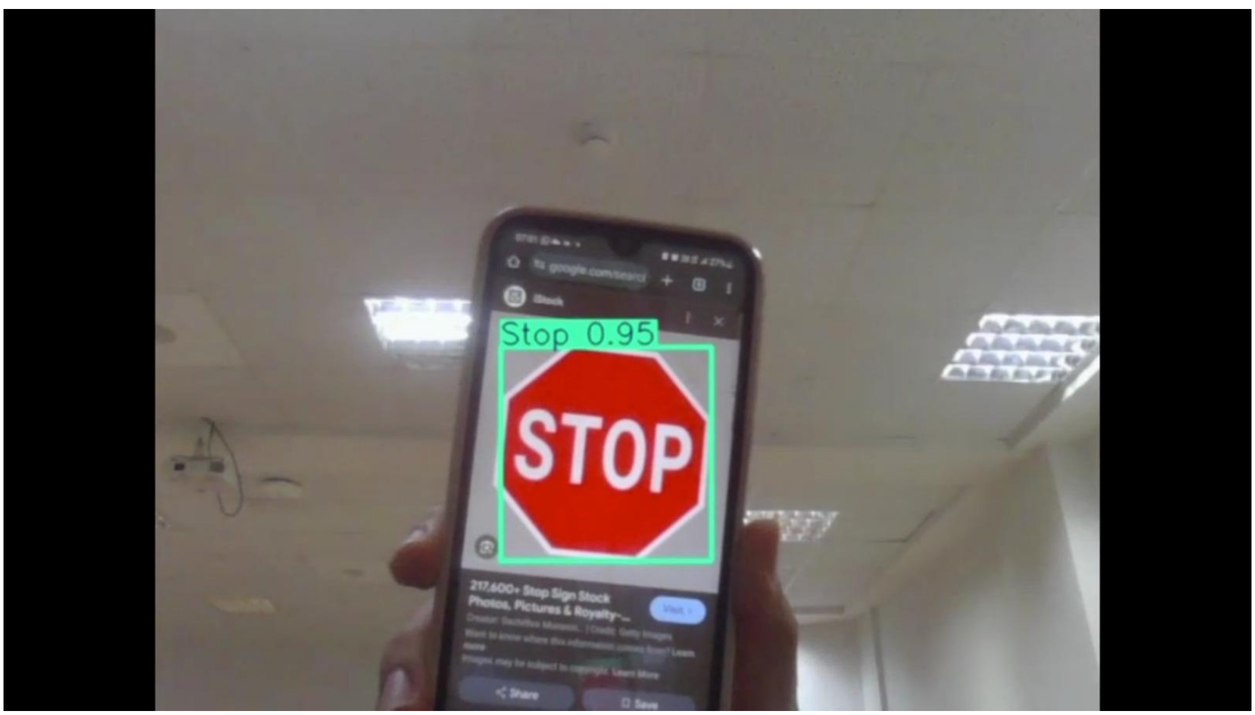

Figure 5

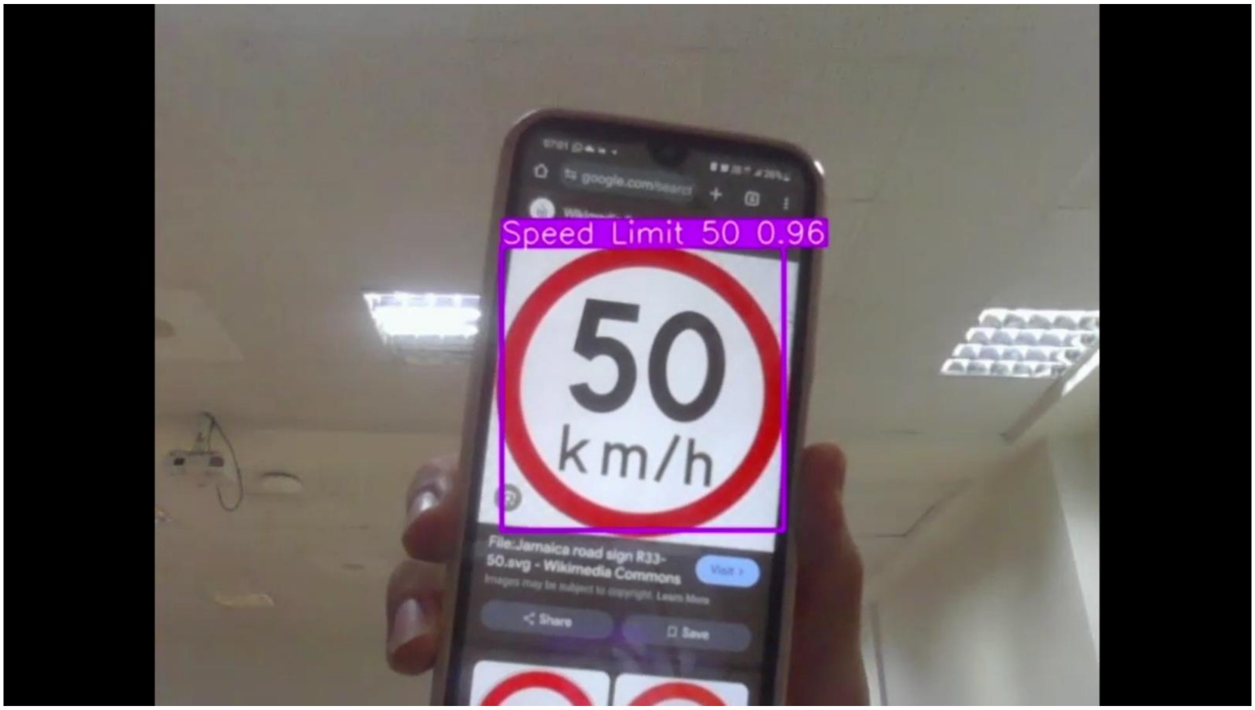

Figure 6

Cloud and Dashboard connections:

Readings appearing on Dashboard and AWS:

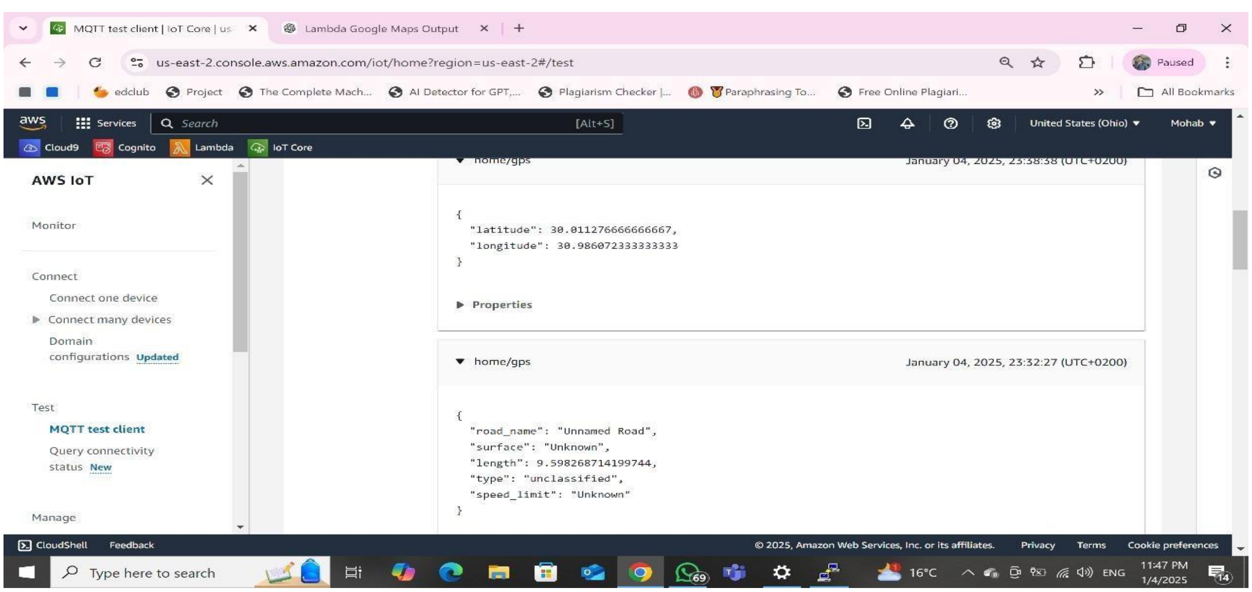

Figure 7

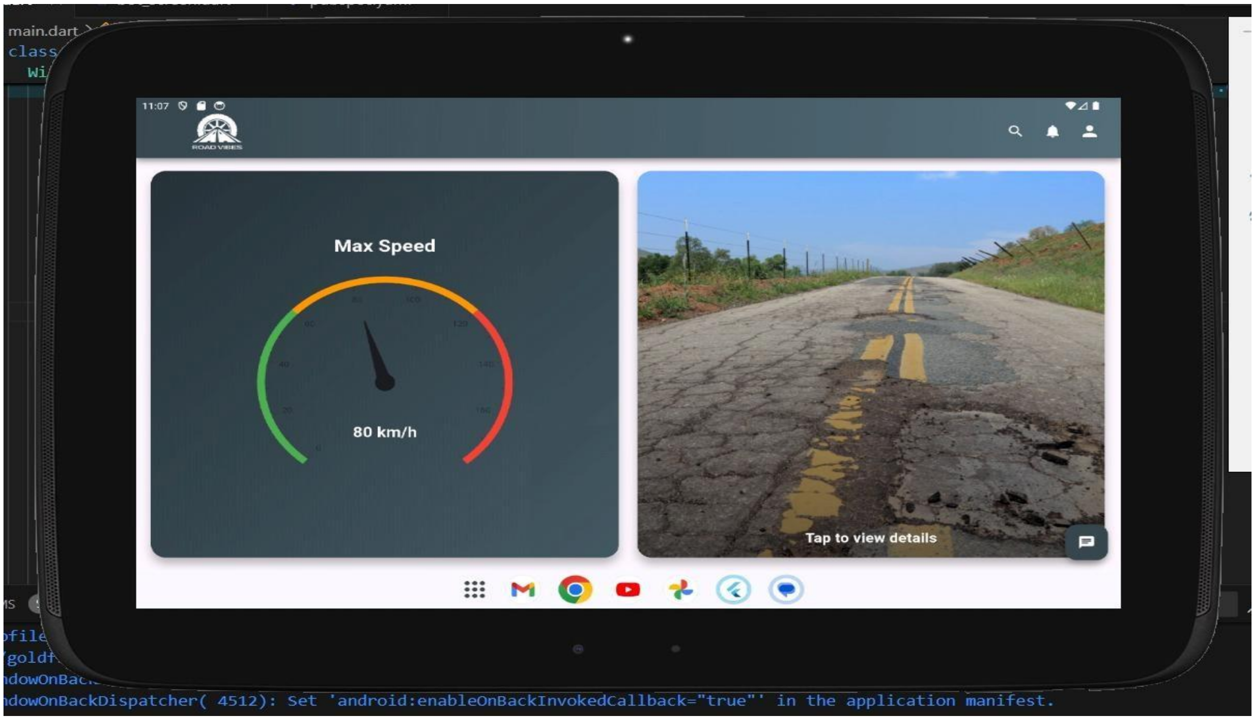

Figure 8

Inter process communication

IPC mechanisms, **sockets** and **message queues**, to establish efficient communication between the client, server, and AI model. Sockets were used to enable real-time data transfer between the client and server. The client simulated GPS data and transmitted it to the server via a TCP socket connection. The server processed this data and sent periodic heartbeat messages back to indicate its operational status. This socket-based approach ensured continuous and reliable communication, replicating real-world GPS functionality.

On the other hand, we employed **POSIX message queues** for asynchronous communication between the server and the AI model. The server sent heartbeat messages to the queue to notify the AI of its status. The AI monitored these messages to determine whether the server was operational or down. If heartbeats were received, the AI remained idle, acknowledging that the server was active. If no heartbeat was detected, the AI initiated predictive functionalities, such as detecting traffic signs, to handle the server's absence. This combination of sockets for network communication and message queues for inter-process communication ensured the system was both modular and robust.

Prototype first impression

We started to implement a functional prototype that supports our system. This is only the first phase of the prototype and still to be fully functional by the end of final project

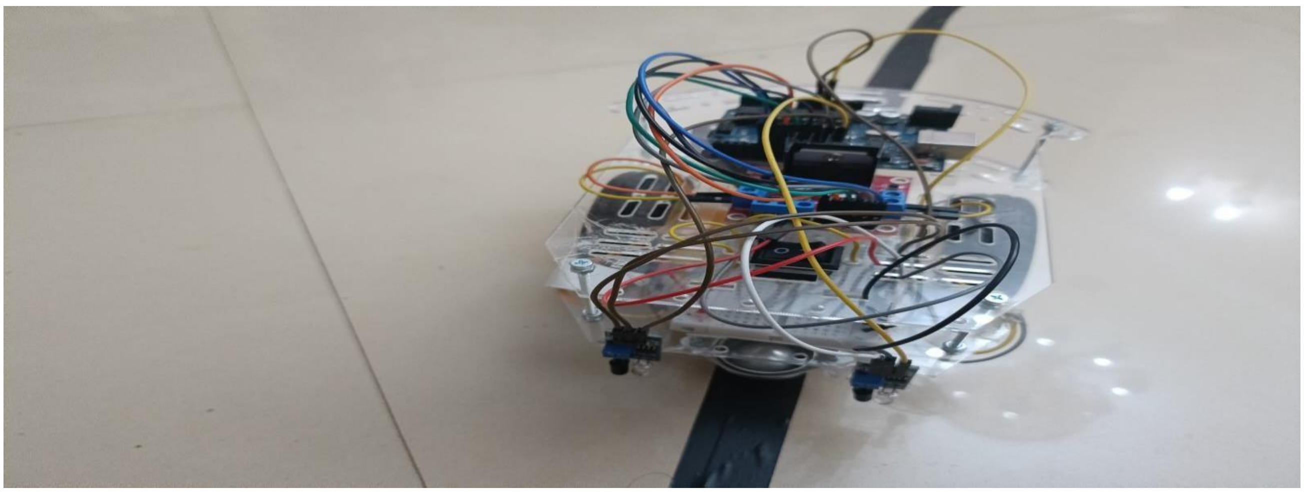

Figure 9

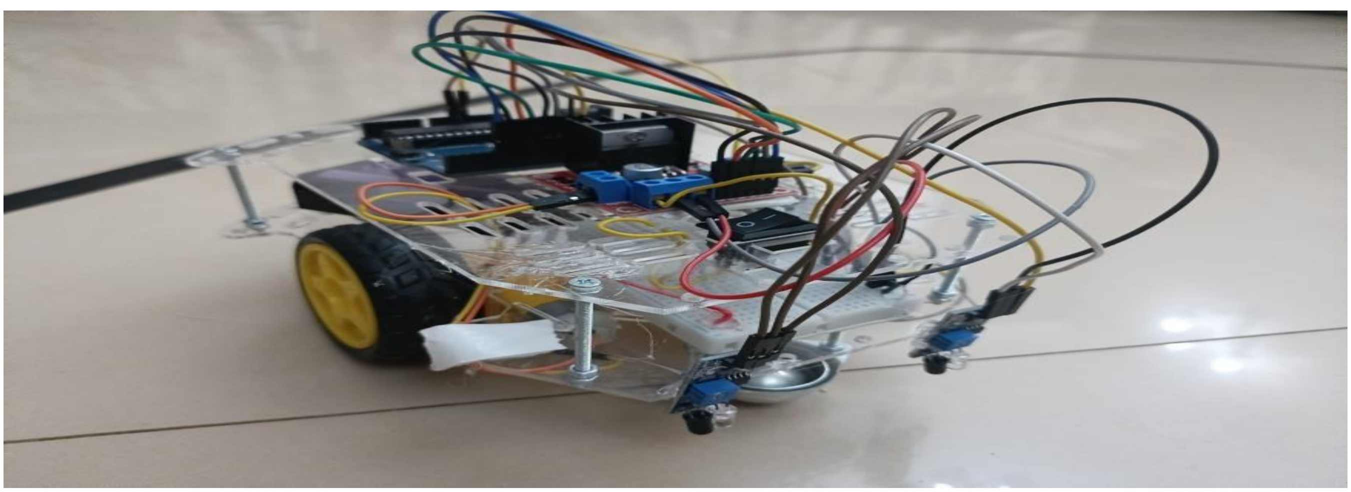

Figure 10

Business Model

Key Partners

- **Technology Suppliers**: GPS module (e.g., Neo-6M), Raspberry Pi hardware vendors
- **Cloud Providers**: AWS IoT Core,
- **AI Framework Developers**:
- **Automotive Industry Collaborators**: Partnerships with vehicle manufacturers and fleet managers

Key Activities

• AI **Model Training & Deployment**:
• **System Integration**
• **Field Testing**: Validating the system in various driving scenarios
• **System Development**: Building a user-friendly interface with critical alerts

Key Resources

• Hardware: Raspberry Pi, GPS module, camera for data capture and processing
• Software: AI model (YOLOv8), cloud services, and dashboard development tools
• Team: Skilled engineers and developers for system design and maintenance

Value Propositions

• **Enhanced Safety**: Providing drivers with timely road condition updates
• **Operational Efficiency**: Minimizing delays, reducing fuel consumption, and optimizing routes
• Reliability: Offline AI-based recognition ensures continuous functionality
• User-**Focused Design**: Easy-to-use dashboard and chatbot for seamless interaction

Customer Relationships

• **Direct Engagement**: Chatbot for on-the-go assistance
• **Community Building**: Online forums and user support for feedback and updates
• **Continuous Improvement**: Regular updates for feature enhancements and security

Channels

• **Retail**: Automotive shops and online marketplaces
• **B2B Partnerships**: Collaborations with fleet management and car manufacturing firms
• **Digital Platforms**: App stores for system software and updates

Customer Segments

• **Safety-Conscious Drivers**: Individuals looking for enhanced driving experience
• **Fleet Operators**: Companies seeking real-time insights for vehicle management
• **Insurance Providers**: Firms aiming to lower risks and optimize premium calculations

Cost Structure

• **Development Costs**: AI model training, software development, and system prototyping
• **Operational Costs**: Cloud service subscriptions, hardware and maintenance
• **Marketing and Distribution**: Promotional activities and distribution network management

Revenue Streams

- **Product Sales**: One-time fees for system hardware and software
- **Subscription Models**: Ongoing fees for advanced cloud services and premium features
- **Licensing**: Offering system design and technology to automotive manufacturers

Figure 11

Conclusion

Cloud-Connected Driver Notification System represents a significant leap forward in driver assistance technology. By integrating real-time data collection, advanced cloud-based processing, and intuitive user notifications, this system effectively addresses the key challenges faced by modern drivers. using Raspberry Pi as the central controller ensures a cost-efficient and flexible platform for system operations. The seamless integration with the Blynk application delivers timely and user-friendly updates directly to the driver, enhancing the overall experience. This system offers multiple benefits, including improved road safety, reduced traffic congestion, and optimized fuel efficiency, making it a practical and scalable solution for modern transportation needs. Furthermore, its modular and adaptable design allows for future enhancements, such as incorporating additional AI-driven features, more comprehensive datasets, and evolving technologies. This forward-thinking approach positions the system as a robust foundation for the ongoing evolution of intelligent driver support systems.

References

ElectronicWings. (n.d.). GPS module interfacing with Raspberry Pi. from https://www.electronicwings.com/raspberry-pi/gps-module-interfacing-with-raspberry-pi

Suryawanshi, R. (2023). Personal cloud storage using Raspberry Pi. Retrieved from https://www.researchgate.net/publication/370314210_Personal_Cloud_Storage_using_Raspberry_PI