IOT-BASED BATTERY MANAGEMENT SYSTEM FOR AN ELECTRIC VEHICLE

A Project Submitted by:

THAMIZHAZHAGAN MURUGAN (40255497)

SREEDHAR SUNDARARAJAN KRISHNASAMY (40256680)

SIVAGURUNATHAN GNANASEGARAN (40256664)

To

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

GINA CODY SCHOOL OF ENGINEERING AND COMPUTER SCIENCE

COEN 6841 – INTERNET OF THINGS (FALL 2023)

DR. RODOLFO COUTINHO

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

Numerous breakthrough technologies have transformed several sectors in the modern day, and one prominent advance in the automobile sector is the Electric Vehicle (EV). As a result of growing awareness of the negative environmental effects of conventional internal combustion vehicles—especially the way these pollutants contribute to global warming—efforts are being made to create electric vehicle alternatives. the vehicle sector accounts for more than half of overall CO2 emissions. As a result, numerous automobile firms are actively developing and manufacturing Electric Vehicles as a more ecologically friendly option. Electric vehicles are based on the idea of replacing the traditional internal combustion engine with an electric motor and a rechargeable battery. Induction motors, DC motors, and synchronous motors are some of the most often used electric motors. The change of Permanent Magnet Synchronous Motor, with the Induction Motor being the most common. Because of its low cost, rechargeability, and efficiency, lithium-ion batteries are often used in electric vehicles. Even with continuous efforts to improve capacity and shorten charging periods, one significant obstacle is the growth in heating problems that accompany these breakthroughs in battery technology. The consequences of such heating issues include safety concerns, potential fire mishaps, a reduction in battery cycle life, and decreased efficiency.

Air conditioning and other conventional cooling techniques are insufficient to solve these heating problems. As a result, creative ways to monitor battery characteristics and activate automated cooling systems as needed are needed. This not only protects the battery's safety and durability, but it also reduce the risks connected with high heat. Moreover, owners of electric vehicle have trouble monitoring the battery's remaining charge feature that is frequently accessible in conventional internal combustion vehicles via a gasoline gauge. The lack of such a function frequently leaves electric car owners unsure about when to charge, increasing the danger of running out of battery power on roads. This problem is exacerbated by the restricted availability of electric charging infrastructure, making it critical for customers to have access to it.

To deal with these types of challenges, we would advise you to set up a comprehensive module. This module has two purposes: it operates as an automated cooling mechanism for the battery and also it provides users with current information of the state of their battery. This module aims to improve the safety, efficiency, and user experience of electric vehicles by offering innovative solutions, therefore contributing to the automotive industry's ongoing development towards more sustainable and eco-friendly practises.

2. ANALYSIS

The described battery management system illustrates a complete method for monitoring and protecting the battery of an electric vehicle. The selection of the ARM-based Arduino DUE microprocessor denotes a strong and competent control module. Accurate battery monitoring of voltage and current is ensured by the combination of an ACS712 Hall-Effect DC sensor and a DC voltage sensor. These characteristics are critical in establishing the battery's specific energy and preventing overloading.The system's main feature is the automated cooling mechanism, which is controlled by a DHT22 temperature sensor. When the temperature exceeds a specified threshold, the system uses continuous temperature monitoring to activate a relay-controlled cooling system. This cooling system, which consists of a relay, pump, and cooling tubes wrapped around each battery cell, efficiently distributes heat, hence extending battery life.

An LTC2941 Coulomb Counter is included to provide users with insights into the health of their battery. During charging and discharging, this component properly detects the charge entering and exiting the battery. This data is mathematically converted into ampere-hours, allowing the battery's

State of Charge to be determined. This information is critical for users to determine the remaining battery capacity.

An NODE MCU WIFI module facilitates real-time data transmission between the user and the module. Using the MQTT protocol, data is delivered to the Amazon Web Service (cloud) and ultimately to a user website. This online interface provides a detailed view of battery parameters such as current, voltage, and temperature. Mainly, it gives clarity on the functioning of the cooling system and displays the State of Charge via an easy-to-read gauge. We may also manage the cooling system manually through the Internet using a virtual switch. In the case of a malfuction, the system sends alarm messages and emails to the registered user, which is a useful tool for quick intervention and maintenance. This extensive and user-friendly design provides optimum battery management, hence improving both the longevity and safety of electric vehicle batteries.

3. DESIGN

3.1. Problem Solving Approach:

- ✓ **Sensor Integration:** Extensive battery health monitoring is ensured by the selection of certain sensors (DC voltage, ACS712 current, LM35 temperature, and LTC2941 Coulomb Counter).

- ✓ **Automatic Cooling System:** To avoid overheating, when the battery temperature rises over a certain threshold, the DHT22 temperature sensor activates a relay-controlled cooling system.

- ✓ **User Interface:** By allowing real-time data transmission with AWS, NODEMCU's web interface gives users access to battery metrics. Important data such as voltage, current, temperature, and cooling system is displayed on the interface. status, and State of Charge. It also allows the user to control the cooling system manually.

- ✓ **AWS Integration:** Leveraging AWS ensures secure and scalable cloud storage. The MQTT protocol facilitates efficient data transmission, and AWS hosts a user-accessible website for data visualization

3.2. Architecture

This electric car battery management system's architecture is built thorough data exchange, control, and monitoring to guarantee the best possible battery health. The sensor layer, control layer, communication layer, cloud layer (AWS), and user interface are all included in the system's tiered design.Dedicated sensors are used in the sensor layer to monitor important battery properties. The DC voltage sensor gives information about the voltage of the battery, which is important for particular energy calculations. By checking the output current using the ACS712 Hall-Effect DC sensor keeps the battery from being overloaded. When the temperature is increased, the automated cooling system is activated by the DHT22 temperature sensor, which continually checks the battery's temperature. Furthermore, the precise measurement of charge entering and exiting the battery during charging is provided by the LTC2941 Coulomb Counter, which is situated at the charging and load terminals charging and discharging, enabling precise calculations of the battery's State of Charge.

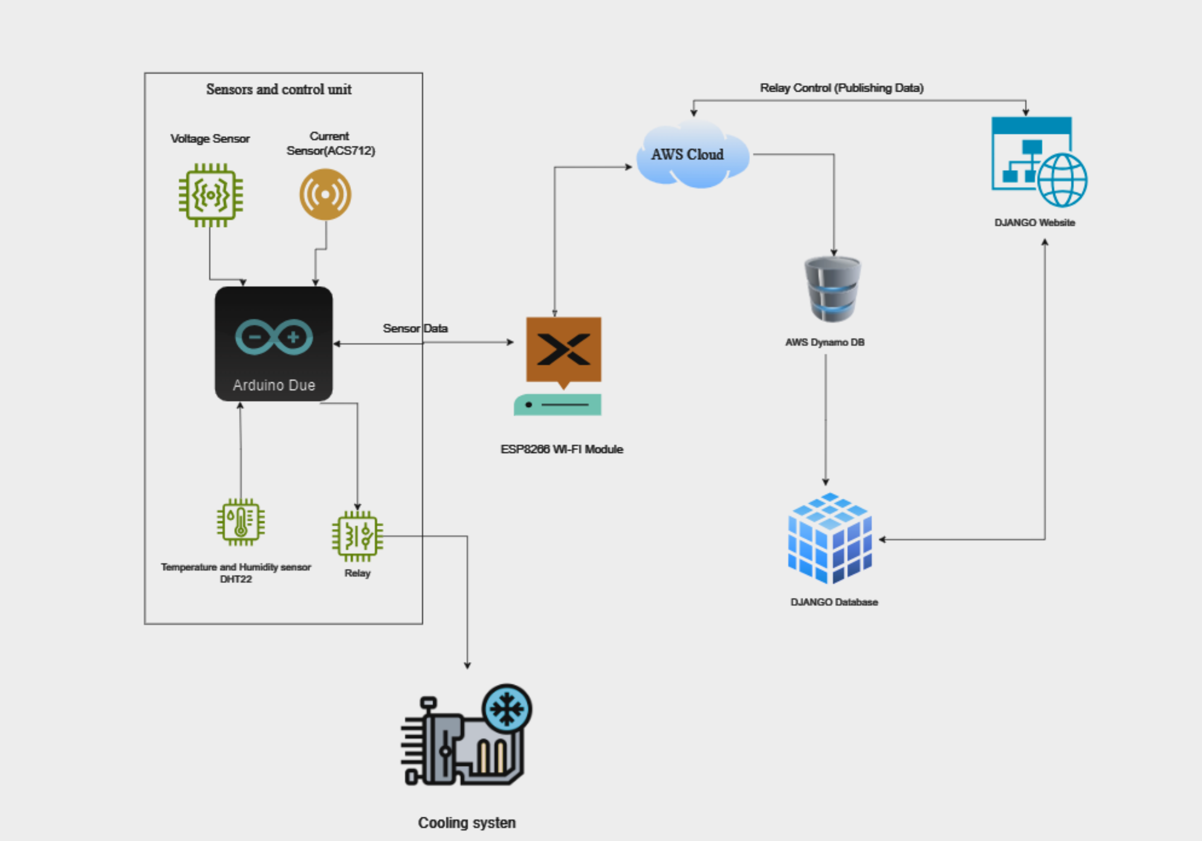

Fig 1 Architecture

The Arduino DUE microcontroller is at the heart of the control layer. After processing the sensor data, it uses a relay to regulate the automated cooling system by using the DHT22 sensor's temperature measurements. This layer keeps the battery operating at its best by minimising overloading and overheating, which prolongs the battery's life. The system and the AWS Cloud may exchange data in real time thanks to the communication layer. Using the MQTT protocol, smooth communication is made possible by the Node MCU WiFi module. In order to ensure that sensor data is available for additional processing and storage, this layer is essential for securely and effectively delivering the data to AWS.

The foundation of data processing and storage is the cloud layer, which is housed on AWS. AWS offers a safe and scalable cloud infrastructure for the system. The IOT system and AWS communicate using the MQTT protocol, which guarantees dependable and lightweight data transfer. Users are able to view real-time battery metrics by accessing a user-accessible webpage hosted on the cloud layer using HTTP queries.

In conclusion, this design combines sensors, a microcontroller, and cloud services to produce a dependable and expandable battery management system for electric vehicles. Because of the layered structure's efficient data flow, control, and user accessibility, batteries are monitored effectively, automatically cooled, and have a longer lifespan.

4. IMPLEMENTATION

4.1. Components

4.1.1. Arduino Due

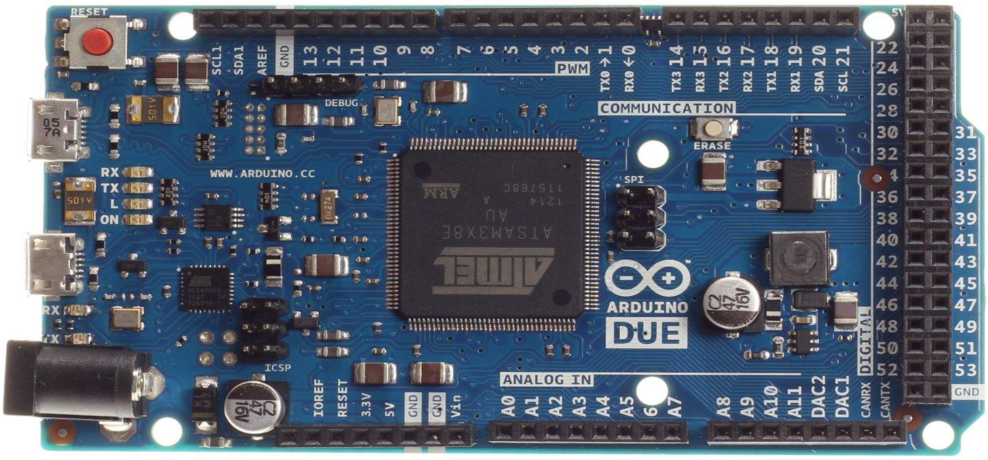

Fig2. Arduino Due

This battery management project's important component is the Arduino Due, as seen in Fig. 2. Precise control and communication between different components are made possible by its ARM Cortex-M3 architecture. The Arduino Due, which serves as the microcontroller, communicates with important sensors such as the LM35 temperature sensor, ACS712 Hall-Effect DC sensor, and DC voltage sensor. It collects real-time data by implementing particular algorithms in the Arduino IDE, automating the cooling system as necessary. Furthermore, the Arduino Due makes it easier for the NODEMCU WIFI module to communicate with one another, which makes it possible to connect to the AWS Cloud via the MQTT protocol. Due to its strong processing capabilities, the system is able to use the LTC2941 Coulomb Counter to determine the State of Charge, giving consumers important information about the condition of their battery on the aws hosted interface

4.1.2. Temperature Sensor

Fig. 3. DHT22 Temperature Sensor

The DHT22 temperature sensor is one of the most main components of the battery management system. By accurately sensing the temperature, it ensures that the battery is operating at optimal temperature. It is connected to the Arduino DUE microcontroller and is a component of the automatic cooling system. When the battery temperature rises above some point, it turns on. Because of this system over heating is avoided and life span of battery is increade. Its real-time temperature data transmission via the NODEMCU to the AWS Cloud guarantees timely monitoring. The DHT22's reliability is crucial to the cooling system's overall effectiveness since it provides a necessary input for maintaining battery health and providing notifications and warnings to the user in the case of temperature fluctuations.

4.1.3. Voltage Sensor

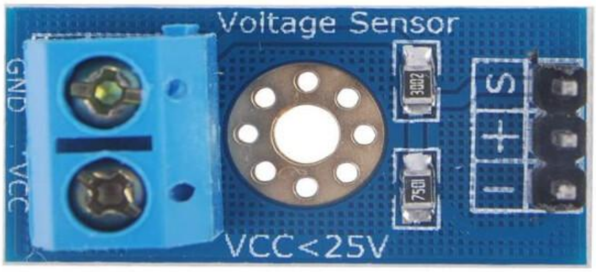

Fig.4 Voltage Sensor

Another component of this battery management system for keeping an eye on the condition of the electric car battery is the voltage sensor, which is shown in Fig. 4. It ensure that the battery runs at the ideal voltage, which has a direct impact on the life span of the battery and the voltage level. The voltage sensor prevents battery from overvoltage or undervoltage scenarios by providing real-time data on the battery's voltage state through its interface with the Arduino DUE microcontroller. Through the NODEMCU WIFI module and MQTT, this vital data is sent to the AWS Cloud, allowing users to remotely check the voltage state of the battery. The voltage sensor's in the system improves the overall efficiency and safety of the battery in the electric vehicle by measuring the voltage.

4.1.4. Current Sensor

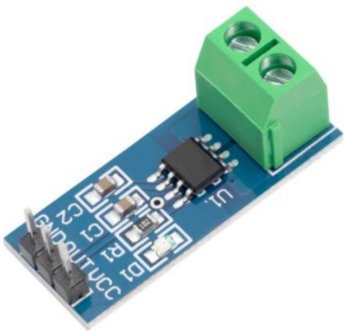

Fig.5 Current Sensor

In our battery management system project, the ACS712 Hall Effect current sensor takes on a crucial role, serving as a linchpin in preventing overloading of the battery. Its primary function lies in precisely gauging the battery's output current, ensuring that it operates within safe parameters. Embedded seamlessly into the system, the ACS712 contributes significantly to bolstering the safety and durability of the electric vehicle's power source.

4.1.5. Node MCU

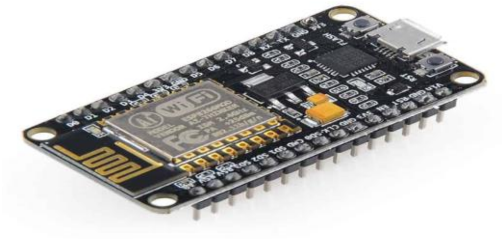

Fig.6 Node MCU

In our battery management system project, the NodeMCU is essential because it links the Arduino DUE microcontroller and the AWS Cloud. The NodeMCU is a data collector and communicator, obtaining data from the Arduino DUE about temperature, voltage, and current drawn from various sensors. After that, it effectively sends this data to the Amazon Web Services (AWS) Cloud using the MQTT protocol.

Real-time communication between the user and the battery management module is made possible by the integration of the NodeMCU, which improves system connection. The NodeMCU enables smooth data sharing using its NODEMCU WiFi module, guaranteeing that customers may remotely monitor important battery statistics. For features like real-time data visualization on the website hosted by AWS to be enabled and provide customers with insights into the performance and health of the battery, connection is essential. The function of the NodeMCU in securely communicating data to the cloud helps to the overall efficacy of the battery management system, increasing user knowledge and control over the battery state of their electric car.

4.1.6. AWS

This battery management system uses Amazon Web Services (AWS), which provides the cloud platform for real-time accessibility and data storage. The NODEMCU module securely transfers battery measurements to AWS through the MQTT protocol. It provides complete real-time data, displayed HTTP queried website, which includes temperature, voltage, and battery current. In order to ensure that users are informed in a timely manner in the case of unusual battery circumstances, the platform also offers email notifications and automatic alarm messages. Through this connection with AWS, customers can monitor and control the status of their electric car batteries through an easily accessible interface.

4.2.Methodology

Our primary objective was to keep an eye on the battery characteristics and notify the user in the event of any irregularities, as was previously said. Furthermore, we intended to incorporate an automated cooling mechanism for the battery. Our module was controlled by an ARM-based microcontroller called Arduino DUE. Using a DC voltage sensor, we first monitored the battery voltage—a critical factor that is directly correlated with the battery's specific energy. This made certain that the cells were functioning optimally to achieve an extended lifespan. Next, we used ACS712, a Hall-Effect DC sensor, to detect the battery's output current in order to avoid overloading. These preliminary actions established the framework for our module's further functionality.

Moving on to the main features, our primary objective was to provide an automatic cooling system for the battery. To achieve this, we employed a DHT22 temperature sensor, which continuously sensed the temperature of the battery. When the temperature value exceeded or equalled the pre-set threshold, the cooling system turned ON automatically. This system comprised a relay, a pump, and cooling tubes spiralled around each cell of the battery. The relay controlled the activation and deactivation of the pump, allowing the liquid to be pumped through the tubes and absorb the heat produced by the battery.

To calculate the State of charge (**SOC**), we have proposed that we are going to use LTC2941 to measure the coulomb count. Since, we are not able to get the sensor we have encountered an alternative mathematical way to calculate the coulomb count. We have calculated the SOC using the below equation:

$$Q(t) = \int_0^t I(\tau)\, d\tau$$

- Q(t) is the accumulated charge in coulombs at time,
- I(τ) is the current in amperes as a function of time,

Note: The sensor LTC2941, also uses the same principle to count coulombs.

Moving forward, we have started integrating our system with NODEMCU WIFI-Module. Firstly, we have successfully established the connection with the AWS Cloud service using "**MQTT**" protocol. In order to send the data to cloud we have published to channel "**bms/pub**" where we successfully received the data from IOT system. The data received are fetched and stored in the default Dynamo DB of AWS using Lambda function.

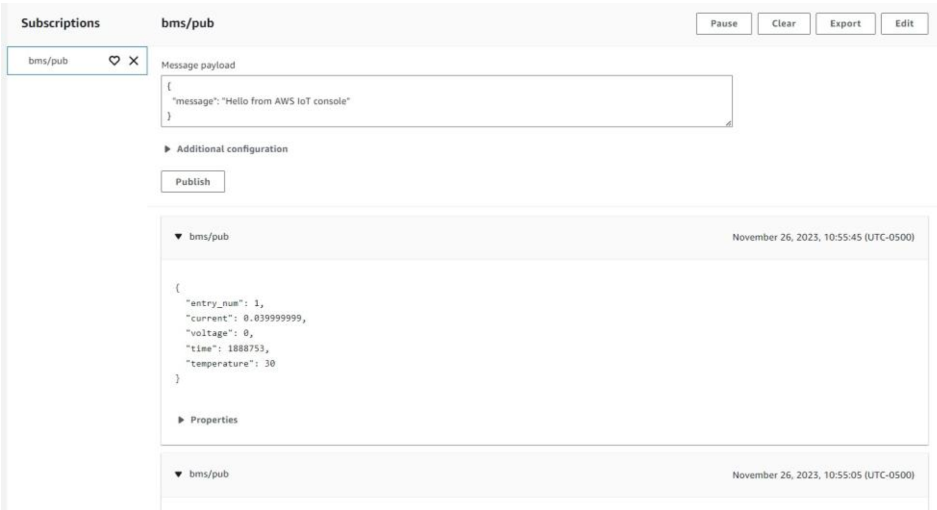

Fig.7 IOT to AWS

In the final stages, we focused on designing a user-friendly website, accessible through HTTP requests, to display the current, voltage, temperature, and state of the battery. Furthermore, we have also included the "**Current Vs Time**", "**Voltage Vs Time**", "**Temperature Vs Time**" graphs to analyze it easily. Additionally, the website also indicates when the cooling system was activated or deactivated, and the State of Charge of the battery was visually represented in the form of a gauge. In case of abnormalities, alert messages were displayed on the website, and automated emails were sent to the registered email address of the consumer.

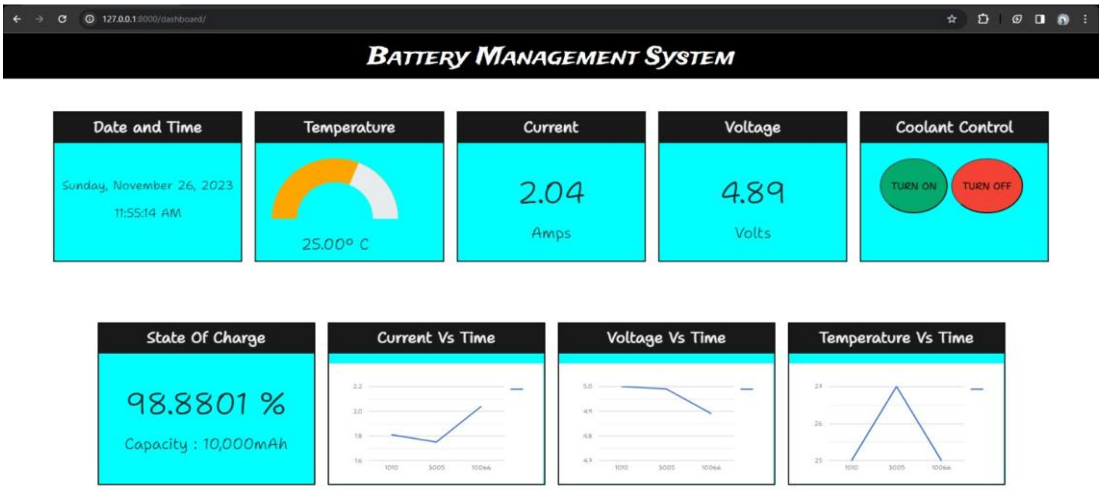

Fig.8 Battery Management System Website

In addition to this, we have also included the virtual relay buttons which can be used to control the relays using internet. To accomplish this, the "ON" and "OFF" commands are sent to AWS Cloud from the website. To send this value to Arduino we have subscribed to a channel "**bms/sub**" from there the data is sent via NODEMCU module.

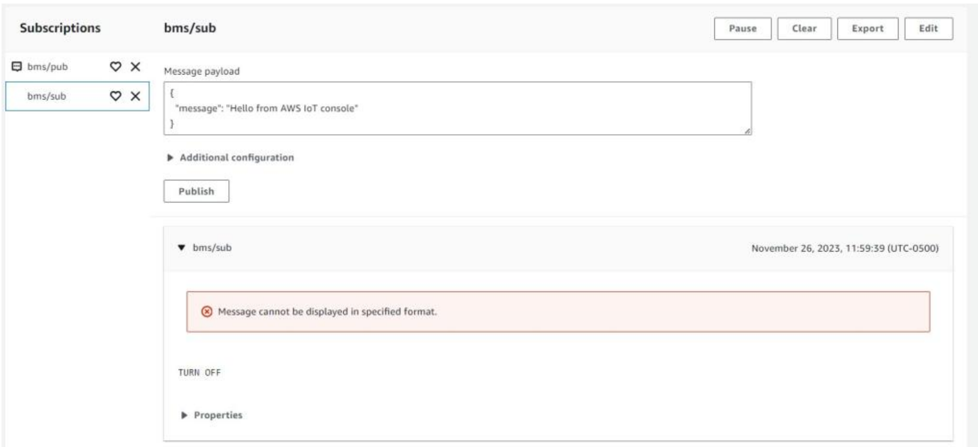

Fig.9 AWS to IOT

This step-by-step implementation ensured a systematic and effective integration of monitoring, control, and communication features in our battery management module.

ARDUINO PIN	DEVICE CONNECTED
Pin "2"	LM35 (DHT22) Temperature sensor
Pin "A0"	ACS712 Current sensor
Pin "A1"	Voltage Sensor
Pin "3"	Relay Module
TX and RX pin of Arduino Due is connected to RX and TX pin of NodeMCU respectively	

Table 1: Pin Configurations

4.3 Software implementation

For simulation, we have used Proteus software. We have implemented the complete hardware as a software model and also tested the functionalities and the results were successful. The software model implementation slightly varies from hardware part, because of availability of few sensors, and due to some compatibility issues. In our case, since Proteus doesn't support Arduino due so we have used "**Arduino Mega**". The NodeMcu available in the Proteus software doesn't connect with the Wifi so we have used a block called "COMPIM" which will help to connect hardware NodeMcu to Proteus Arduino.

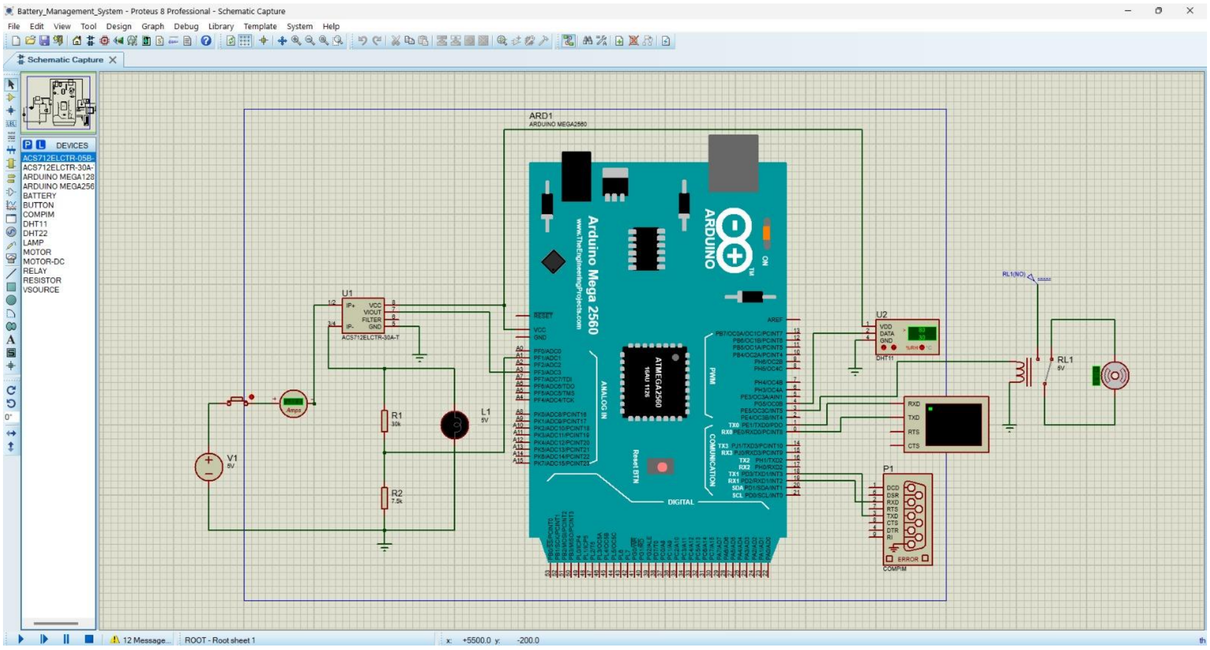

Fig.10 Circuit Diagram

4.4 Partial implementation:

As we mentioned above we are unable to get LTC2941 sensor for calculating SOC, so we have opted a mathematical calculation to count the coulomb from which we can calculate the SOC. Secondly, for the cooling system, we have proposed to implement the complete cooling system with pump, relays, spiraled coolant tubes, coolant, and battery packs. Since we are unable to get pump for coolant we cannot implement the complete cooling system. But, we have implemented till controlling the relay.

5 LESSONS LEARNED

This project presented us with a lot of challenges that deepened on our expertise in battery management systems. One of our initial problems was the complexity of sensor calibration, which we solved by numerous trial and error method for calibrating the sensor to ensure accurate readings. Whether it was integrating Arduino DUE or developing an easy-to-use user interface, each team member was involved. There was an automatic cooling system included, and battery life considerably increased. Our future goals include investigating better cooling strategies, allowing user customization for warning levels, and researching machine learning integration for predictive maintenance. This project provided invaluable experience, showing both opportunities for continued improvement in future efforts and successes.

The project also shown how crucial it is to control power usage and have efficient teamwork. Problems in these domains were resolved by means of efficient coding optimization and frequent team meetings for information exchange and work delegation. In order to increase efficiency, we intend to add tools that let users adjust alarm thresholds and investigate cutting-edge cooling techniques. The project was a great way to gain knowledge about battery management system design and generated ideas for improvements in the future.

5.1 Contribuiton:

1. **SIVAGURUNATHAN GNANASEGARAN**

   Initiated the project with ARDUINO DUE, was responsible for interfacing all sensors with ARDUINO. He calibrated every sensor accurately according to requirement. Designed the user interface website (frontend and backend), designed the database.

2. **THAMIZHAZHAGAN MURUGAN**

   Worked on Proteus simulation and worked on user interface website (backend). Responsible for connection establishment between NodeMcu and Aws cloud. Focused on automated alerts and emails were implemented for abnormal conditions.

3. **SREEDHAR SUNDARARAJAN KRISHNASAMY**

   His focus was on the State of Charge (SoC) calculation and communication aspects. Additionally, he established real-time data transfer via ESP8266 to Amazon Web Service, showcasing battery metrics on a user-friendly website accessible through HTTP requests.

6 CONCLUSION

In short, we successfully implemented and tested our battery management module, showing high reliability for all functions. A system that is stable is the result of the painstaking integration process that included communication, control, and monitoring. The first stages laid the basis by focusing on voltage monitoring and overloading prevention. The LM35-powered automated cooling mechanism proved effective in preserving the ideal temperature of the battery. Accurate real-time State of Charge data can be obtained by the application of the Coulomb counting method. The last phases demonstrated smooth data transport using the MQTT protocol and NodeMCU, providing extensive information on a simple to use webpage. This system efficiently monitors, controls, and transmits critical data to provide customers with a dependable solution for improved battery life and performance.

7 REFERENCES

1. S. Krishnakumar *et al.*, "IoT-based Battery Management System for E-Vehicles," *2022 3rd International Conference on Smart Electronics and Communication (ICOSEC)*, Trichy, India, 2022, pp. 428-434, doi: 10.1109/ICOSEC54921.2022.9952052.
2. S. Çeven, E. Küçükkülahli, A. Albayrak and Y. Biçen, "Development of IoT Based Battery Management System," 2021 2nd International Informatics and Software Engineering Conference (IISEC), Ankara, Turkey, 2021, pp. 1-5, doi: 10.1109/IISEC54230.2021.9672346.
3. L. S. B, A. Sangari, E. K, K. V, J. A. Sheeba and D. Sivamani, "IoT-Based Battery Monitoring System for Electric Vehicle," 2022 IEEE International Conference on Current Development in Engineering and Technology (CCET), Bhopal, India, 2022, pp. 1-5, doi: 10.1109/CCET56606.2022.10080389.
4. V. G. G, A. N, J. S and V. N D, "IoT based Lithium-Ion Battery Monitoring System in Electric Vehicle," 2023 Third International Conference on Artificial Intelligence and Smart Energy (ICAIS), Coimbatore, India, 2023, pp. 1092-1096, doi: 10.1109/ICAIS56108.2023.10073696.
5. S. S, N. S. Manjarekar and S. Barik, "SoC Estimation and IoT based Delayed Charging of Electric Vehicles," 2022 Second International Conference on Power, Control and Computing Technologies (ICPC2T), Raipur, India, 2022, pp. 1-6, doi: 10.1109/ICPC2T53885.2022.9776936.