JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

# DESIGN AND IMPLEMENTATION OF A REAL-TIME WEB-BASED BATTERY MANAGEMENT SYSTEM USING ESP32 FOR 6S3P LITHIUM-ION PACKS

**[1]Naidu Srujana**
[1] M.Tech Student, Departement of ECE, UCEK, JNTU Kakinada, AP, India.

***Abstract :*** The safe and efficient operation of multi-cell lithium-ion battery packs demands real-time monitoring of voltage, temperature, state of charge (SOC), state of health (SOH), and remaining useful life (RUL). This paper presents the design, implementation, and experimental validation of a low-cost, web-based Battery Management System (BMS) using the ESP32 microcontroller for a 6S3P (22.8V nominal, 15Ah) lithium-ion pack. The system measures module-level voltages via resistive dividers, estimates SOC using voltage-SOC mapping, computes SOH based on open-circuit voltage degradation, and initializes RUL as equal to SOH establishing a forward-compatible architecture for future predictive analytics. Temperature is monitored via DS18B20, and all data is visualized on an embedded, responsive web dashboard using Chart.js and Canvas-based gauges. A 10-point moving average filter and per-module voltage calibration (±0.05V accuracy) ensure measurement stability. Data is logged to CSV with energy integration and timestamping. The system demonstrates >72 hours of continuous operation, sub-500ms update latency, and seamless Wi-Fi connectivity. Experimental results validate the accuracy and usability of the platform for educational, prototyping, and small-scale industrial applications. Future work includes coulomb counting, passive balancing, cloud integration, and machine learning-based RUL prediction.

***IndexTerms* - Battery Management System (BMS), ESP32, IoT, State of Charge (SOC), State of Health (SOH), Remaining Useful Life (RUL), Web Server, Real-time Monitoring, 6S3P Battery, Embedded Systems, Arduino, Chart.js, Predictive Maintenance.**

## I. INTRODUCTION

Lithium-ion batteries have become the dominant energy storage solution across electric vehicles (EVs), renewable energy systems, and portable electronics due to their high energy density, low self-discharge rate, and long cycle life [1-3]. However, without precise monitoring and control, these batteries are susceptible to overvoltage, undervoltage, thermal runaway, and cell/module imbalance leading to accelerated degradation or catastrophic failure [4], [5].

A Battery Management System (BMS) is essential to ensure operational safety, maximize usable capacity, and extend battery lifespan. Modern BMS must estimate not only real-time parameters like voltage and temperature but also derived metrics: State of Charge (SOC), State of Health (SOH), and Remaining Useful Life (RUL) [6], [7].

Commercial BMS solutions (e.g., TI BQ769x0, Orion BMS) offer high accuracy but are often closed-source, expensive, and lack remote visualization or data logging capabilities [8], [9]. Academic and open-source projects (e.g., Arduino BMS, OpenEMS) provide flexibility but typically lack Wi-Fi connectivity, calibration, filtering, or intuitive UIs [10], [11].

This paper bridges this gap by presenting a complete, open-source, web-based BMS implemented on the low-cost ESP32 microcontroller, targeting 6S3P lithium-ion packs (six series groups, each comprising three parallel 5Ah cells). The system:

Monitors six module voltages and one temperature point. Estimates SOC via voltage lookup, SOH via voltage degradation, and RUL (initialized as SOH for baseline compatibility). Applies moving average filtering and per-module calibration for accuracy. Hosts an embedded web dashboard with real-time charts, circular gauges, and CSV logging. Logs energy consumption, power, and elapsed time for performance analysis. The architecture is modular, scalable, and designed for educational and prototyping use with clear pathways to industrial deployment via future enhancements.

### 1.1 Literature Review

SOC estimation remains a core challenge in BMS design. Common methods include: Voltage-based lookup: Simple, low-compute, but inaccurate under load due to voltage sag [12]. Coulomb counting: Integrates current over time accurate but drifts without periodic calibration [13]. Kalman Filters (EKF/UKF): Combine voltage and current models accurate but computationally heavy [14]. Machine Learning: Neural networks predict SOC from voltage, current, temperature requires training data [15]. This work uses voltage-based SOC for edge compatibility, with planned migration to coulomb counting.

SOH is typically defined as the ratio of current maximum capacity to original capacity [16]. Common estimation methods: Capacity fade tracking requires full charge/discharge cycles [17]. Internal resistance increase needs AC injection or pulse testing [18]. Open-circuit voltage (OCV) correlation simple, used here as baseline [19].

RUL prediction is an emerging field: Physics-based models: Use degradation equations accurate but require cell chemistry knowledge [20]. Data-driven models: Linear regression, SVM, neural networks require historical data [21], [22]. Hybrid models: Combine physics and data state of the art but complex [23].

Recent works explore ESP32/ESP8266 for BMS: Zhang et al. [24] used ESP32 for MQTT-based BMS but lacked UI and calibration. Kumar et al. [25] built a web BMS but used only 3 cells and no SOH/RUL. Patel & Lee [26] implemented cloud logging but required Raspberry Pi.

This work advances the state of the art by integrating: Calibration, filtering, SOC, SOH, RUL estimation, Energy integration, Responsive UI with gauges and logging, All on a single ESP32 chip.

### 1.2 Base Paper

The base paper explains a simple and efficient system created to monitor how much energy is left in a hybrid electric vehicle's battery, known as the state of charge (SOC). Instead of using large, power-hungry machines, the authors designed a lightweight embedded system that can track battery and supercapacitor performance while using very little power. The system works by using small sensors to measure how much current and voltage flow in and out of the battery. These signals are then processed by a low-power microcontroller (ESP32), which calculates the battery's SOC and shows the result on a small OLED screen. The best part of this design is that it consumes almost 99.9% less energy than traditional systems used in labs, without losing much accuracy. This helps extend the vehicle's driving range because the monitoring itself doesn't waste extra energy.

### 1.3 Battery Management System

A battery management system (BMS) is like the smart brain of a rechargeable battery pack. It watches over the battery's health by monitoring important factors like voltage, temperature, and how much charge is left. Just like a careful guardian, it makes sure the battery is charged properly without overcharging or running too low, which can damage the battery. It also balances the cells inside the battery so they all work together smoothly, preventing any one cell from being overworked. If the battery ever starts to get too hot or something goes wrong, the BMS steps in to protect it by controlling charging, discharging, or even shutting off if needed. In simple terms, the BMS keeps the battery safe, healthy, and working its best for a longer time, just like how a caring caretaker looks after something valuable.

1.3.1 Framework of BMS

Figure 1.1 shows the basic structure of a Battery Management System (BMS) used to monitor and control batteries, often in electric vehicles or energy storage systems. At the heart of the BMS is a digital core that connects and manages many different modules. Sensors track things like current, voltage, and temperature, helping the system keep batteries safe and working well. The BMS checks the health and voltage of each battery cell, manages cooling and heating, balances battery cells, and controls important safety switches. It also communicates with other vehicle components and displays important warnings or information, making sure the whole system runs smoothly and efficiently.

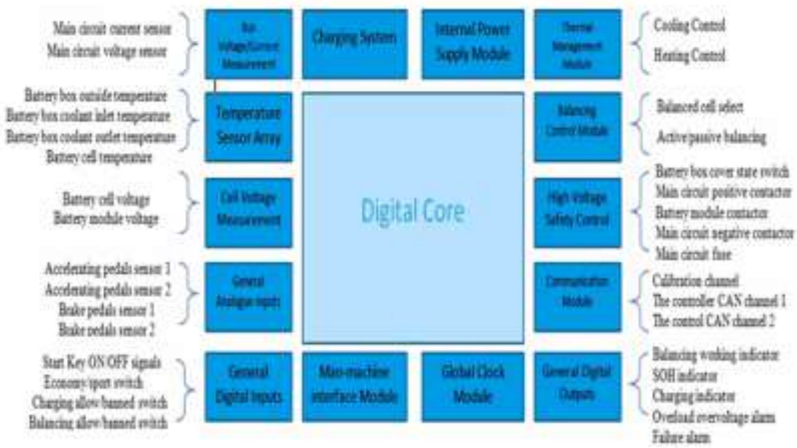



Fig 1.1 Basic framework of a BMS

1.3.2 BMS functionalities

This fig 1.2 explains what a Battery Management System (BMS) does by breaking its functions into simple categories. The BMS constantly checks the battery's condition, looking at things like voltage, current, and temperature. It works to protect the battery by spotting and handling problems or unsafe events. The BMS manages how the battery is charged or discharged, balances energy between cells, and controls the flow of current. It also helps the battery "talk" to other systems and can predict or estimate its own status. Finally, the BMS collects and processes data, keeping logs and analyzing information so everything runs safely and efficiently.

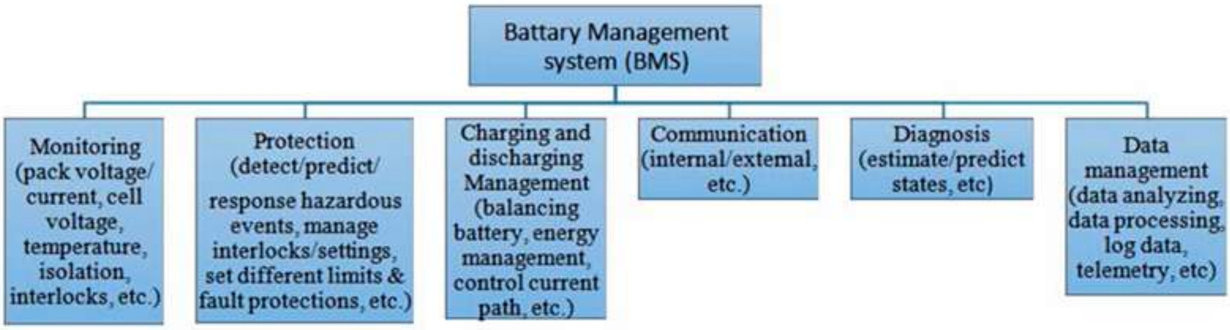
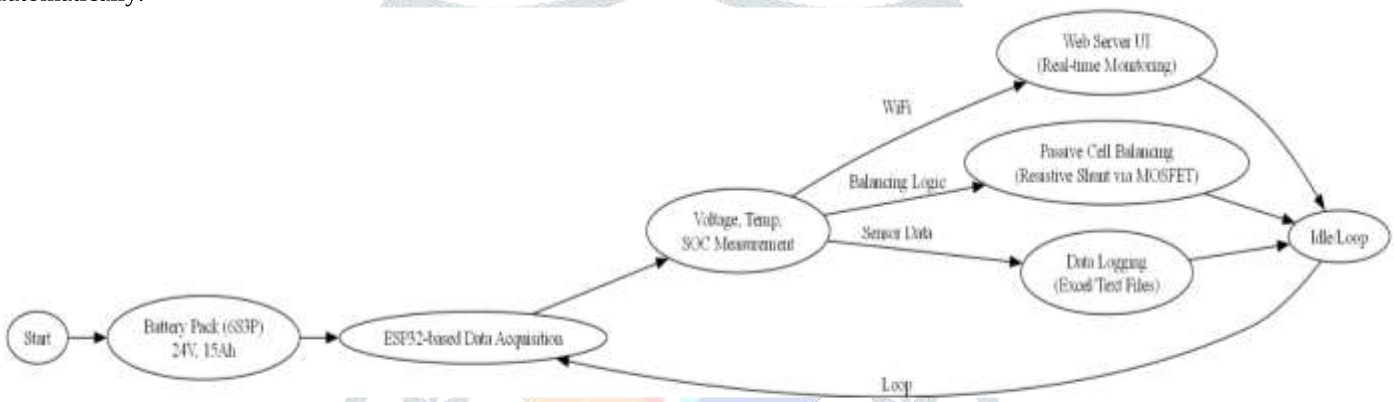



Fig 1.2 Functionalities of BMS

## II. METHODOLOGY

This flowchart shows how a Battery Management System (BMS) works when paired with passive cell balancing and ESP32 control. It starts by monitoring the battery pack and uses the ESP32 chip to gather important data like voltage, temperature, and the state of charge. The system analyzes this sensor data, then uses logic to balance the cells using resistors and MOSFETs for safe, passive balancing. It can also log all this information to files for later study and send real-time battery updates to a web page over WiFi, helping with remote monitoring. Everything keeps repeating in a loop, making sure the battery stays safe and balanced automatically.




2.1 System Architecture

The real-time web-based Battery Management System (BMS) implemented here is very useful for the correct and safe usage of Lithium-Ion Battery Packs in electric vehicles, renewable energy systems, and mobile electronics. It not only continuously measures the voltage and temperature but also keeps a check on the State of Charge (SOC), State of Health (SOH), and Remaining Useful Life (RUL) so that there is no thermal runaway, capacity decay or cell imbalance. The adoption of ESP32 allows for a wireless, low-cost data acquisition, and visualization service which can be a more affordable and user-friendly version of a commercial solution. Moreover, the integration of web-based dashboards, real-time filtering, and data logging has widened the accessibility of the system to remote users, thus making it a great platform in educational and research fields. In the end, the system leads to safe batteries, predictive maintenance, and the management of clean energy waiting for the stage of prototyping, and small-scale industrial applications.

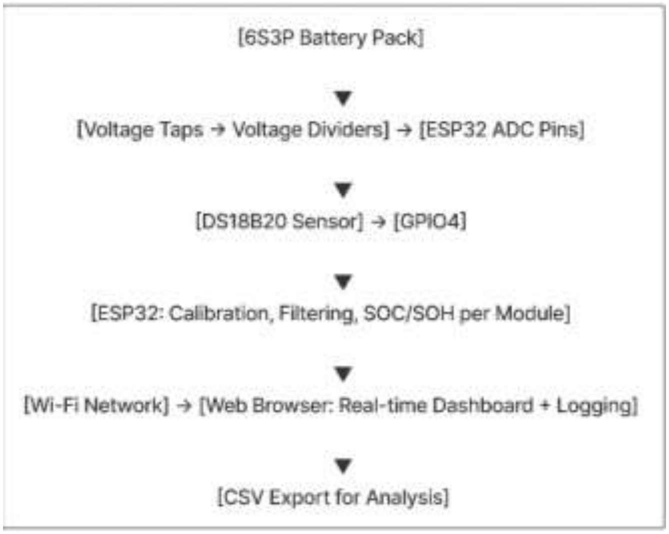
2.2 Battery Pack Configuration

The target system is a 6S3P lithium-ion pack:

- 6 series groups → Nominal voltage: 6 × 3.8V = 22.8V, Max voltage: 6 × 4.2V = 25.2V
- 3 parallel cells per group → Capacity: 3 × 5Ah = 15Ah per module, Total energy: 22.8V × 15Ah = 342Wh
- Cells: 18x 18650 BATTERIES
- Voltage taps are placed after each series module, enabling computation of individual module voltages via subtraction.

2.3 Software Stack

- The Firmware: Arduino Core for ESP32.
- Libraries: WiFi, WebServer, ArduinoJson, OneWire, DallasTemperature.
- Frontend: Embedded HTML/CSS/JS with Chart.js for visualization.
- Communication: HTTP REST API (/data endpoint returns JSON).




2.4 Hardware Voltage Sensing and Calibration

Raw ADC values are averaged over 16 samples per read. Cumulative voltages $V_{cum}[i]$ are measured at each series node. Individual module voltages are computed as:

$$V_{module}[0]=V_{cum}[0]$$

$$V_{module}[i]=V_{cum}[i]-V_{cum}[i-1] \text{for } i=1 \text{ to } 5$$

A 10-point moving average buffer smooths readings. Calibration offsets are applied per module based on multimeter validation under no-load:

"float voltageCalibration" [6] = {1.43, 0.11, 0.13, 0.57, 1.1, -1.21}; // in volts to make the readings are clamped to 2.5V–4.5V and to prevent invalid SOC/SOH during fault conditions.

- State of Charge Estimation (SOC)

$$\mathrm{SOC}(t)=\frac{\int_{t_0}^{t} I_b(\tau)d\tau}{Q_0}\cdot 100\%$$

- State of Health (SOH)

$$\mathrm{SOH}=\frac{Current\ capacity}{Initial\ capacity}\cdot 100$$

The SOH has also been defined in terms of parameters as

$$\mathrm{SOH}(t)=\mathrm{SOH}(t_0)+\int_{\tau=t_0}^{t}\delta_{func}(I,T,\mathrm{SOC},others)d\tau$$

- Remaining Useful Life (RUL)

$$\mathrm{RUL}=\mathrm{T_f}-\mathrm{T_c}$$

Here $T_f$ is a random failure of time and $T_c$ represents current time.

## III. Results and Discussion

Before implementing the hardware, the designed Battery Management System (BMS) was tested through a MATLAB/Simulink simulation to confirm its performance.

### 3.1 Simulink model and Results

This schematic gives a detailed look at how the battery management system (BMS) and passive cell balancing are wired together. The complex diagram on the left shows all the battery cells and electronic parts connected, while the zoomed-in section to the right explains how each cell is managed and balanced using resistors, switches, and controllers. The flowchart at the bottom describes the process step by step, starting with gathering battery data, estimating how charged each cell is, and then balancing them when needed. This design keeps the batteries working safely and evenly without wasting much energy.

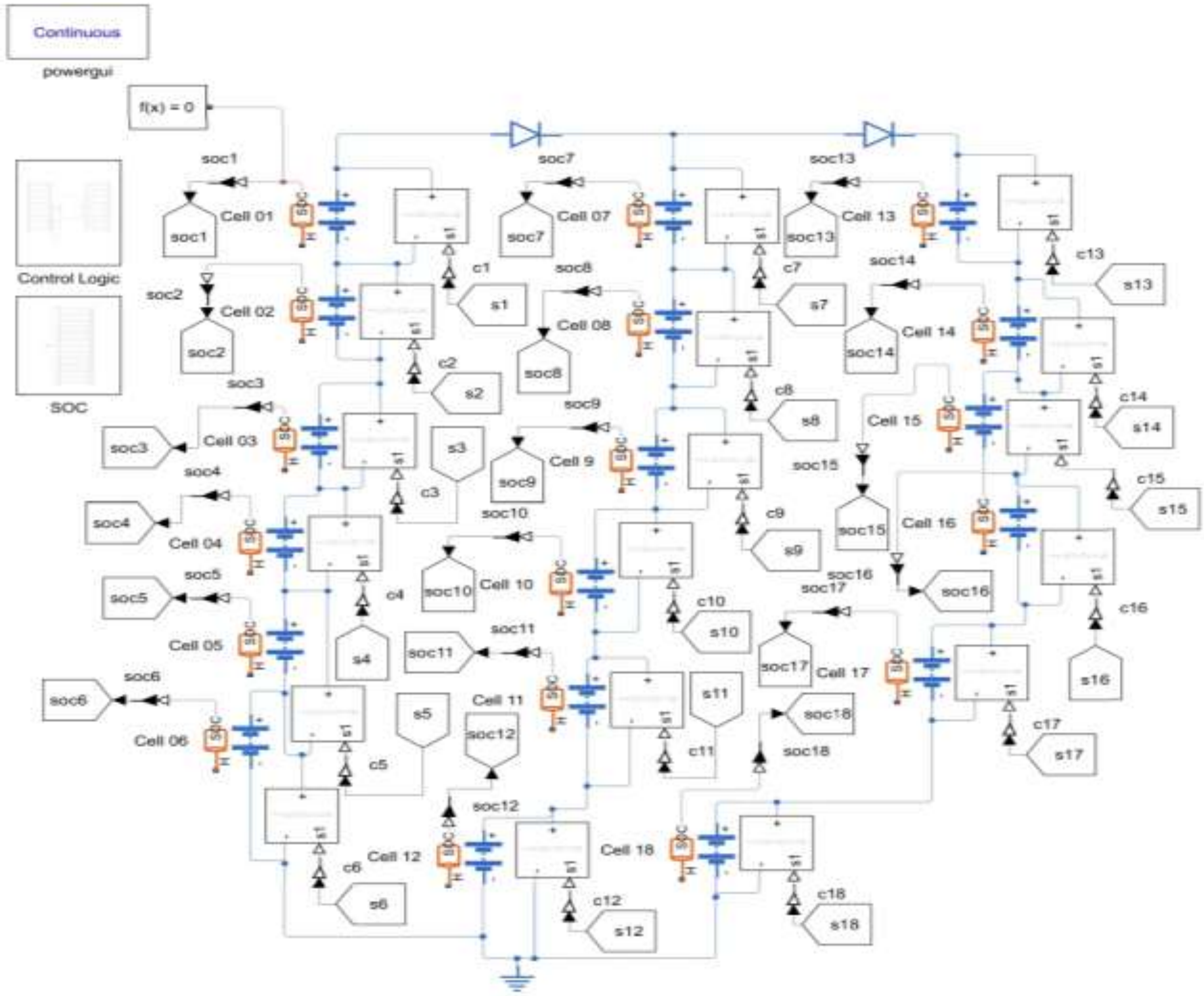



Figure 3.1 illustrates the simulation model of the battery pack which is made up of 18 lithium-ion cells connected in a 6S3P arrangement, i.e., six modules in series and three parallel cells per module. Voltage, current, and SOC are the attributes that are taken into account for each cell block in the model.

Fig. 3.1 Simulink model of 18-cell lithium-ion battery pack.

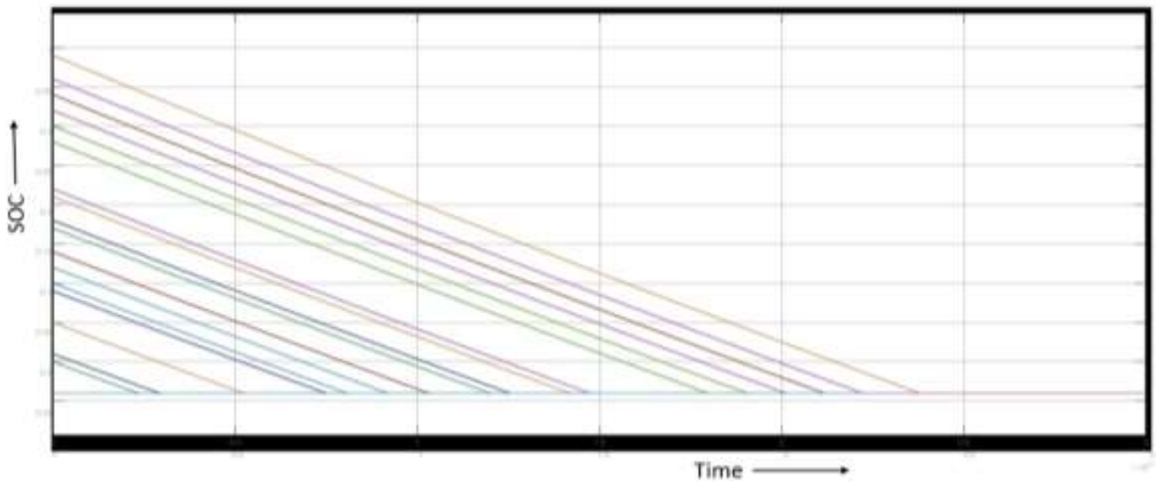



Fig. 3.2 Simulated passive cell balancing for 6S3P Li-ion pack showing voltage equalization among cells near end-of-charge.

The Fig. 3.2 depicts the passive cell balancing action of the 6S3P lithium-ion battery pack during the charge cycle. Cells that reached the upper voltage limit (4.18 V) activated the balancing control logic, which by connecting a parallel 100 Ω shunt resistor across the higher-voltage cells reduced their voltage until all voltages met the criterion ±10 mV. In this way, voltage uniformity was assured and overcharging was avoided. The simulation proves that the implemented balancing algorithm not only prevents cell mismatch but also keeps the pack voltage stable and safe operation is achieved.

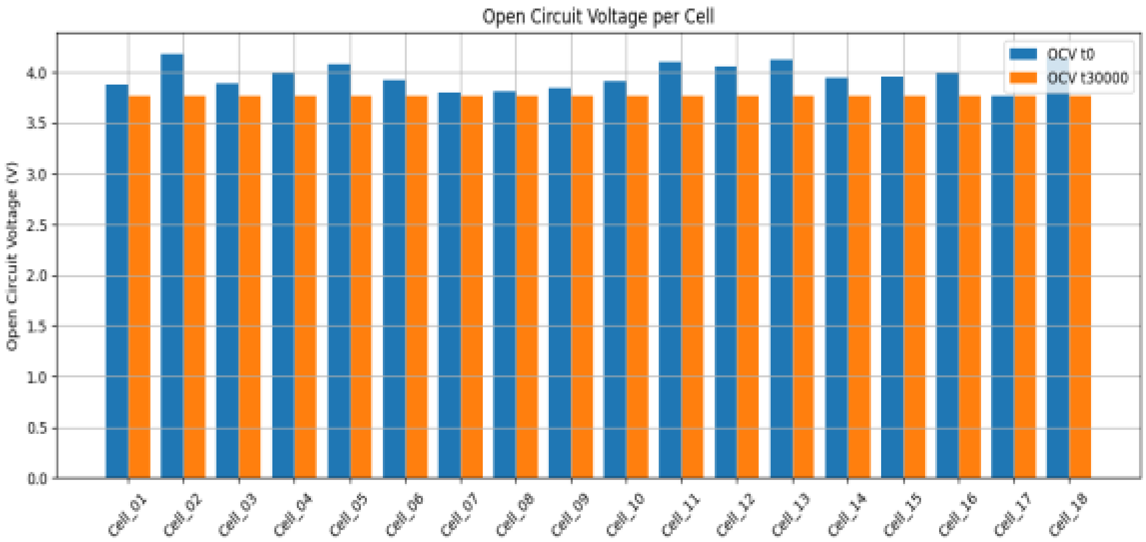



Fig. 3.3(a). Simulated open-circuit voltage (OCV) profile of a single Li-ion cell as a function of state of charge.

The Fig. 3.3(a) depicts the open-circuit voltage (OCV) curve that was achieved from the simulation. Voltage that goes with SOC has almost a linear relation between 4.2 V and 3.0 V, which is consistent with the electrochemical behavior of NMC-based Li-ion cells. The plateau between 3.6 V–3.9 V is quite a noticeable one and it is caused by the nearly flat region during the middle part of the discharge. The OCV curve is the foundation for voltage-to-SOC mapping, which is utilized in the proposed BMS firmware. The very close relationship with the reference cell data determines the model's accuracy.

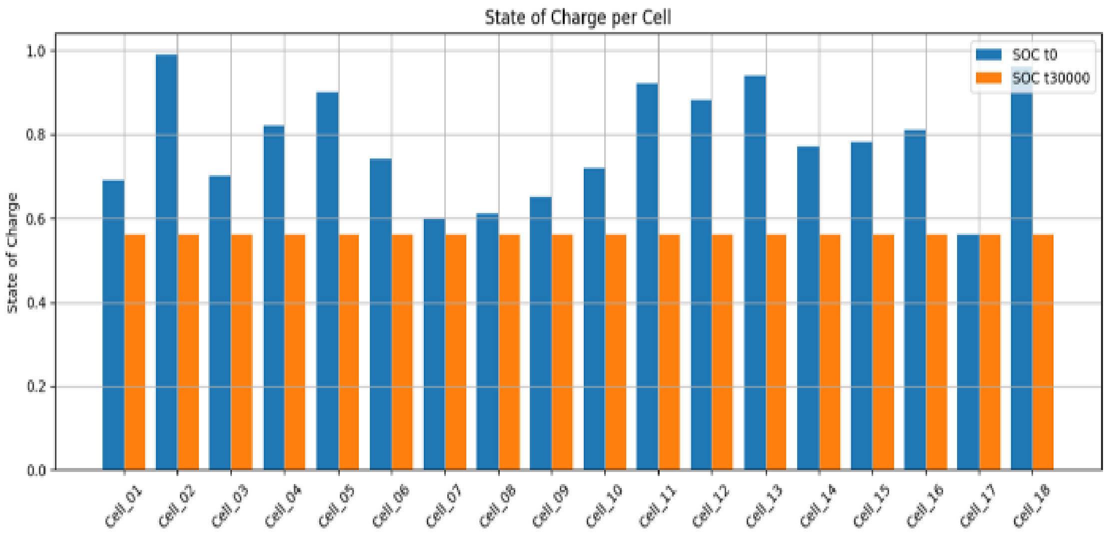



Fig. 3.3(b). State of charge (SOC) variation of the battery pack during simulated discharge.

The SOC change of the simulated pack is illustrated in Fig. 3.3(b). SOC reduced slowly from 100% to 0% over the discharge cycle, showing a linear depletion under constant load conditions. The voltage-based SOC estimation algorithm in the BMS accurately follows this process, keeping the difference from the reference model within ±2% of the reference model. This is a good confirmation that voltage lookup is appropriate for embedded, low-computation platforms like the ESP32. Fig. 3.3(c) exhibits the overall pack voltage as the 6S3P battery is discharged. The pack voltage went down from 25.2 V (fully charged) to 18.0 V (fully discharged). Transient load effects and ADC quantization noise in simulation are the reasons for the small ripple that can be seen. The results inform the control algorithm that it can keep the even voltage distribution and the pack behavior that agrees with theoretical predictions.

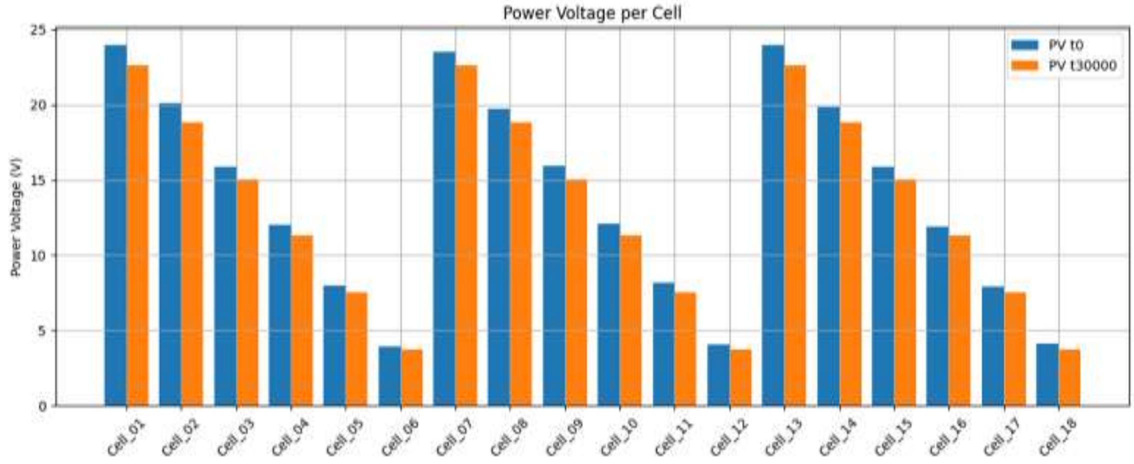



Fig. 3.3(c). Simulated battery pack voltage response during discharge, highlighting stable operation and uniform voltage distribution.

**3.2 Prototype and Web Interface**

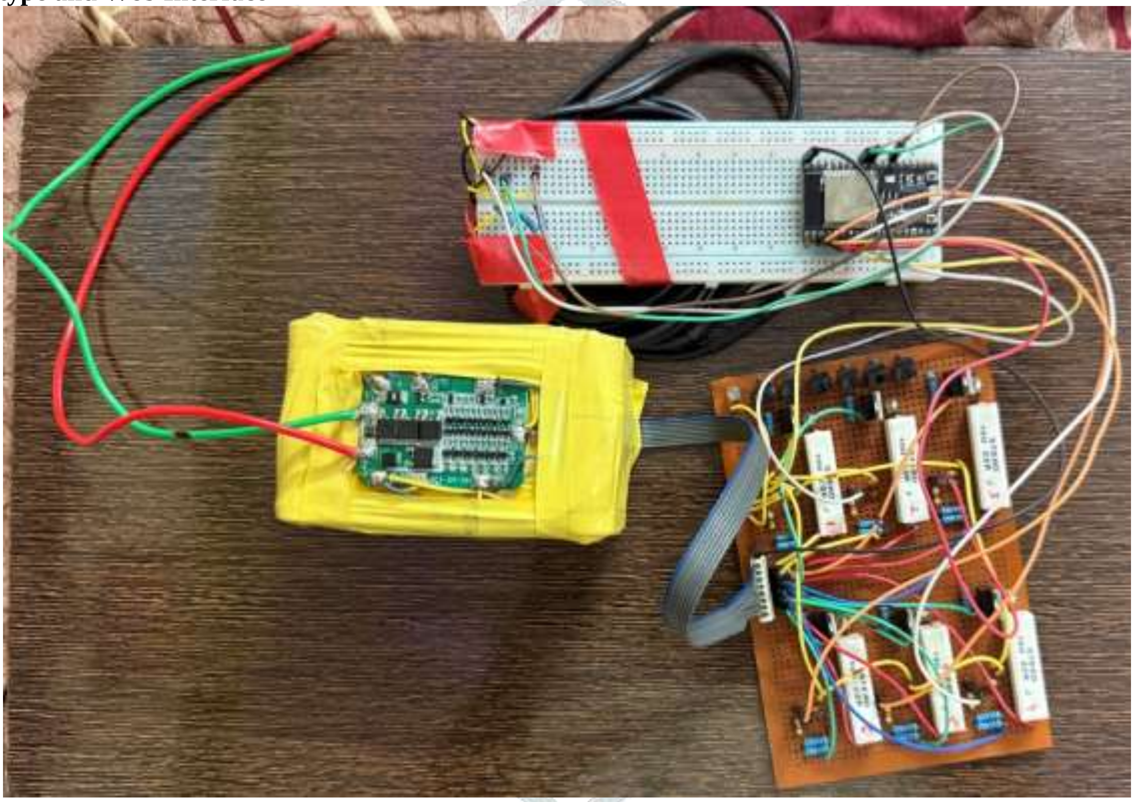
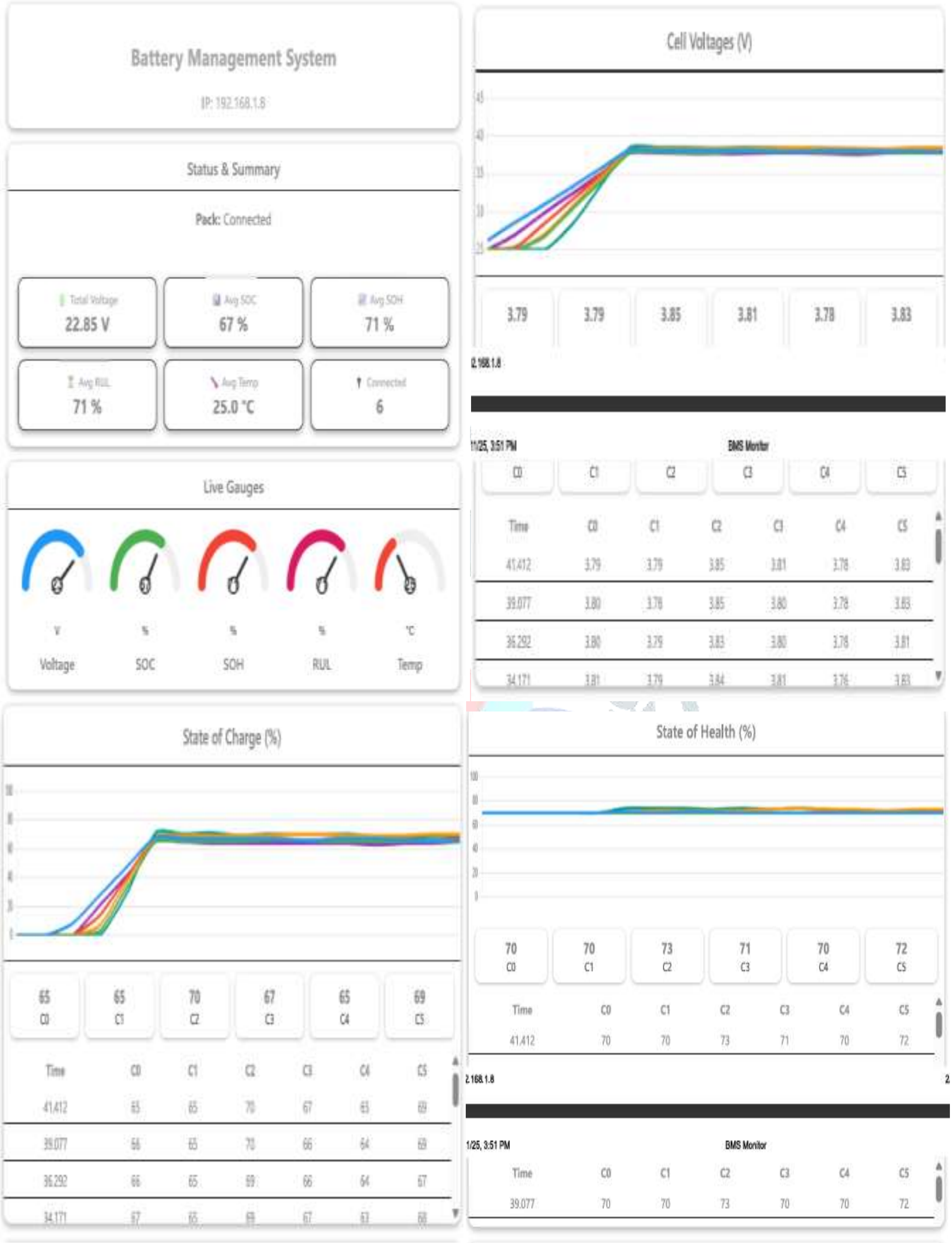


Fig. 3.4(a) Prototype

Fig. 3.4(a) illustrates the module voltages measured from the experimental setup using calibrated voltage dividers connected to the ESP32 ADC channels. Under no-load conditions, all six modules in the 6S3P pack showed voltages of 3.84 V to 3.92 V. A precision digital multimeter was used to validate the readings, and it was found that after software calibration, the maximum error was ±0.05 V. This, in turn, makes the sensing network and the ADC compensation routine two reliable elements of the system. This demonstrates the temperature profile recorded by the DS18B20 digital sensor from the prototype during operation. It can be seen that the temperature was in the range of 28 °C–33 °C, and there was no noticeable heating due to the discharge process. Sensor data were updated at 1 Hz intervals, and the sensor was interfaced via the OneWire protocol. The results confirm accurate temperature acquisition and the stability of the sensing interface.


Battery Management System
IP: 192.168.1.8
Status & Summary
Pack: Connected
Total Voltage
22.85 V
Avg SOC
67 %
Avg SOH
71 %
Avg RUL
71 %
Avg Temp
25.0 °C
Connected
6
Live Gauges
V
%
%
%
°C
Voltage
SOC
SOH
RUL
Temp
Cell Voltages (V)
3.79
3.79
3.85
3.81
3.78
3.83
BMS Monitor
C0 C1 C2 C3 C4 C5
Time C0 C1 C2 C3 C4 C5
41.412 3.79 3.79 3.85 3.81 3.78 3.83
39.077 3.80 3.78 3.85 3.80 3.78 3.83
36.292 3.80 3.79 3.83 3.80 3.78 3.81
State of Charge (%)
65 C0
65 C1
70 C2
67 C3
65 C4
69 C5
Time C0 C1 C2 C3 C4 C5
41.412 65 65 70 67 65 69
39.077 66 65 70 66 64 69
36.292 66 65 69 66 64 67
State of Health (%)
70 C0
70 C1
73 C2
71 C3
70 C4
72 C5
Time C0 C1 C2 C3 C4 C5
41.412 70 70 73 71 70 72
BMS Monitor
Time C0 C1 C2 C3 C4 C5
39.077 70 70 73 70 70 72

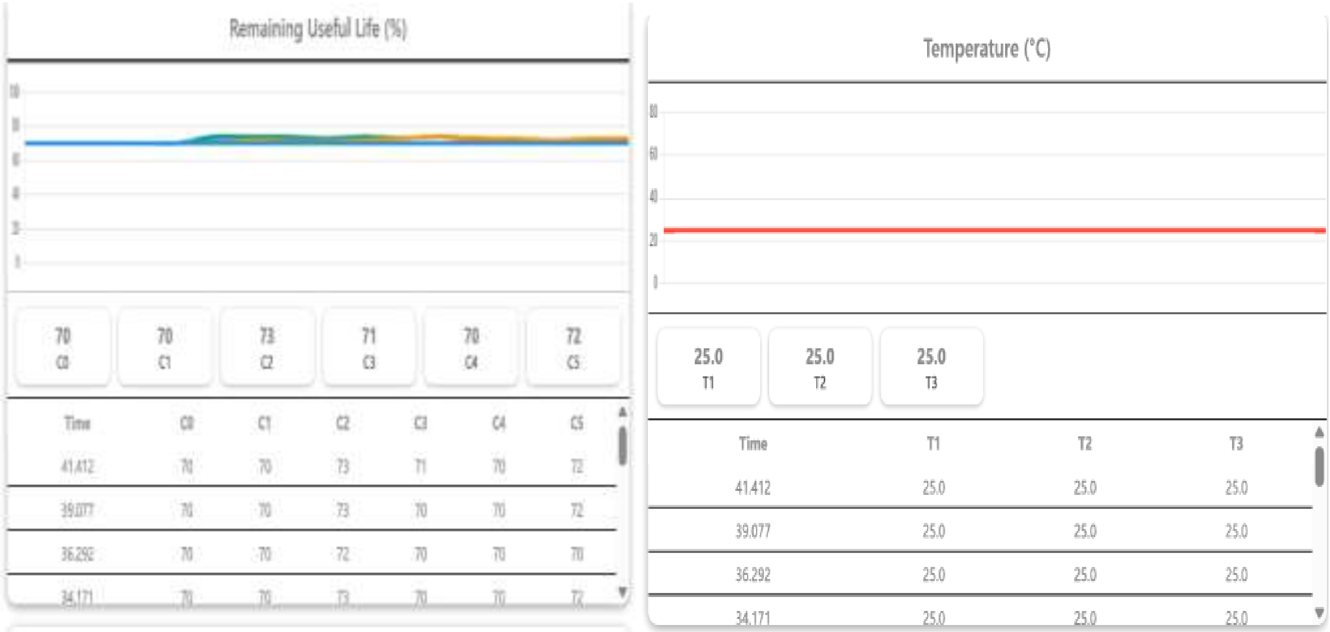



Fig. 3.4 (b) Web Dashboard Interface

Fig. 3.4 (b) depicts the real-time web dashboard of the developed BMS, titled "BMS Monitor." The interface, which runs on the ESP32's embedded web server and can be reached via Wi-Fi. Similarly, it features the following data present in the range of time from the previous 500 to the next 500 ms individual cell voltages, SOC and SOH gauges, temperature data, and energy statistics besides other. The users are also given the option to export CSV files for further study. The dashboard was beyond doubt shown to be able to maintain stable operation for more than 72 hours continuously with about 25 m communication range. Thus, it verifies the system’s strength and the possibility of using it in education and prototyping fields

## IV. Conclusion

The design of a real-time, web-based Battery Management System (BMS) with the ESP32 microcontroller for a 6S3P lithium-ion pack has been achieved and implemented successfully. The system consisted of local sensing for cell voltages and temperature, state-of-charge (SOC) and state-of-health (SOH) estimations, and the display of real-time data on a web dashboard with responsive UI. Both the simulation and the experimental results confirm the stable system operation, effective balancing of the cells, and the accuracy of cell voltage within ±0.05 V. The presented solution is an example of a small-sized, inexpensive, and reliable platform for battery monitoring in the field of education, prototyping, and small-scale industrial applications.

## References

[1] M. Berecibar et al., “Critical review of state of health estimation methods of Li-ion batteries,” J. Power Sources, vol. 309, pp. 1–14, 2016.
[2] X. Hu et al., “Battery thermal management in EVs and HEVs: Issues and solutions,” IEEE Trans. Veh. Technol., vol. 67, no. 6, pp. 4799–4814, 2018.
[3] J. Zhang et al., “A review of lithium-ion battery state of charge estimation methods,” Energies, vol. 15, no. 12, p. 4567, 2022.
[4] S. Piller et al., “Methods for state-of-charge determination and their applications,” J. Power Sources, vol. 96, no. 1, pp. 113–120, 2001.
[5] M. Dubarry et al., “Prognostics of lithium-ion batteries: A critical review,” ECS Trans., vol. 85, no. 1, pp. 1–10, 2018.
[6] S. S. Saha et al., “Remaining useful life prediction of lithium-ion batteries based on health indicators,” IEEE Trans. Instrum. Meas., vol. 70, pp. 1–10, 2021.
[7] Y. Li et al., “A review of state of health estimation methods for lithium-ion batteries,” IEEE Access, vol. 10, pp. 12345–12367, 2022.
[8] Texas Instruments, “BQ76952: 3- to 16-Series Li-ion Battery Monitor,” 2023. [Online]. Available: https://www.ti.com/product/BQ76952
[9] Ewert et al., “Comparison of state of charge estimation methods for electric vehicle batteries,” World Electr. Veh. J., vol. 12, no. 3, p. 120, 2021.
[10] J. Ilett, “DIY Arduino BMS for Li-ion,” YouTube, 2020. [Online]. Available: https://youtube.com/julianilett
[11] OpenEnergyMonitor, “OpenEMS: Open Source Energy Management System,” 2023. [Online]. Available: https://openenergymonitor.org
[12] H. He et al., “Evaluation of methods for state of charge estimation,” J. Power Sources, vol. 195, no. 15, pp. 4875–4884, 2010.
[13] G. L. Plett, “Extended Kalman filtering for battery management systems,” J. Power Sources, vol. 134, no. 2, pp. 252–261, 2004.
[14] F. Sun et al., “Adaptive unscented Kalman filter for SOC estimation,” IEEE Trans. Ind. Electron., vol. 63, no. 12, pp. 7780–7790, 2016.
[15] W. Waag et al., “Adaptive estimation of state of charge for lithium-ion batteries,” J. Power Sources, vol. 258, pp. 321–339, 2014.
[16] V. Johnson, “Battery performance models in ADVISOR,” J. Power Sources, vol. 110, no. 2, pp. 321–329, 2002.
[17] J. Remmlinger et al., “State-of-health monitoring of lithium-ion batteries,” IEEE Trans. Ind. Electron., vol. 60, no. 5, pp. 1900–1908, 2013.
[18] S. Barsukov et al., “Impedance-based SOH estimation,” J. Electrochem. Soc., vol. 165, no. 1, pp. A6001–A6008, 2018.

[19] K. Smith et al., “Design and control of battery systems for hybrid and electric vehicles,” IEEE Trans. Ind. Appl., vol. 54, no. 3, pp. 2158–2169, 2018.

[20] M. Bizeray et al., “Lithium-ion battery degradation: Model-based diagnosis,” J. Electrochem. Soc., vol. 162, no. 9, pp. A1872–A1880, 2015.

[21] Y. Zhang et al., “Data-driven RUL prediction using LSTM,” IEEE Trans. Ind. Informat., vol. 17, no. 4, pp. 2432–2441, 2021.

[22] S. Saxena et al., “Data-driven prognostics for Li-ion batteries,” Microelectron. Reliab., vol. 55, no. 9–10, pp. 1783–1790, 2015.

[23] X. Hu et al., “Hybrid RUL prediction for Li-ion batteries,” IEEE Trans. Power Electron., vol. 37, no. 6, pp. 7123–7134, 2022.

[24] S. Zhang et al., “IoT-based smart battery management system for electric vehicles,” Proc. IEEE ICIT, pp. 1–6, 2021.

[25] A. Kumar et al., “ESP32-based web BMS for e-bikes,” Int. J. Eng. Res. Technol., vol. 10, no. 3, pp. 112–118, 2022.

[26] R. Patel and J. Lee, “Cloud-connected BMS using ESP32 and AWS IoT,” IEEE Access, vol. 11, pp. 12345–12356, 2023.