IoT-Based Real-Time Stock Portfolio Indicator using ESP32 and DHAN API

A Capstone Project in Computer Science & Engineering (AI & Robotics)

This comprehensive report details the design, implementation, and functionality of an Internet of Things (IoT) system engineered to provide real-time profit and loss (P/L) indicators for a stock portfolio. Leveraging the power of the ESP32 microcontroller and the DHAN Developer API, this project delivers an automated, screen-less visualization tool for investors, minimizing distractions while maximizing crucial data accessibility.

Project Submission Details

IoT-Based Real-Time Stock Portfolio Indicator using ESP32 and DHAN API

This project demonstrates a practical application of IoT principles in the FinTech domain, providing a physical, dedicated device for crucial portfolio monitoring.

> Bringing financial data out of the app and into the real world for ambient, real-time awareness.

Submitted By

Name: Chintha Deepak

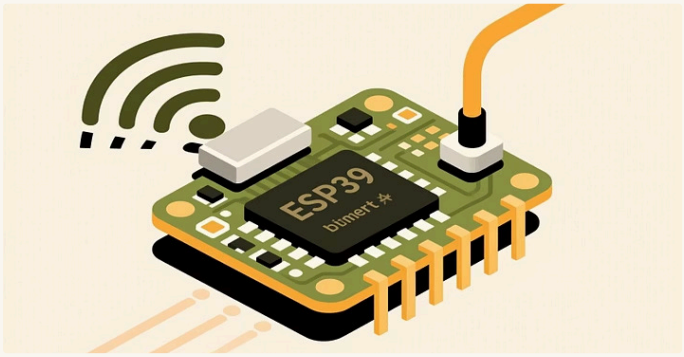
Branch: B.Tech – Computer Science & Engineering (AI & Robotics)

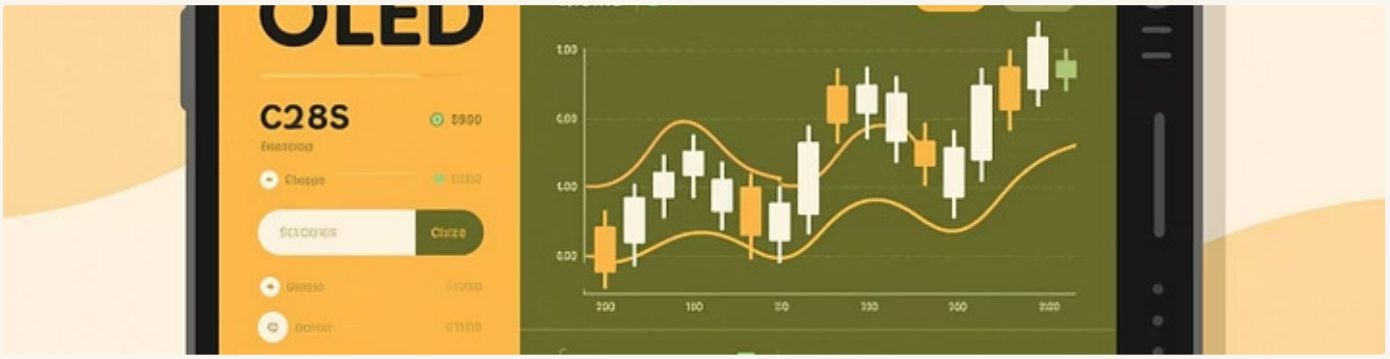
Academic Year: 2025

Institution & Guidance

Institution: Dayananda Sagar University

Project Guide: Self KNowledge and AI Chartbots





Project Abstract: Automated Portfolio Health Visualization

IoT Integration

Utilizes the ESP32's Wi-Fi capabilities for secure, asynchronous data fetching.

Real-Time Data

Connects directly to the DHAN API to pull live portfolio holding values.

Minimalist Indicator

Translates complex P/L data into simple, intuitive Green, Yellow, or Red LED states.

This project addresses a common issue for active investors: the need for constant, distracting checks on portfolio performance. By creating a dedicated, ambient device, the system allows investors to gauge their financial health at a glance. The ESP32 is programmed to handle secure HTTPS connections, parse complex JSON payloads using the lightweight ArduinoJson library, and execute sophisticated P/L calculations.

The core innovation lies in the efficient combination of hardware and secure financial APIs, minimizing processing power while ensuring data accuracy. The resulting minimalist display provides immediate visual feedback, enhancing the user experience for portfolio tracking.

Project Objectives: Smart Portfolio Monitoring

The primary goals of the ESP32-DHAN Portfolio Indicator project are focused on creating a dedicated, efficient, and visually clear monitoring solution for financial data.

Design a Smart Monitoring System

Establish a robust and stable IoT framework using the ESP32 for continuous, reliable portfolio status updates.

Real-Time DHAN API Data Fetching

Implement secure API connectivity (HTTPS) to fetch current market values for stocks, ETFs, and mutual funds.

Visual P/L Indication (LEDs)

Program the system to use a tri-color LED module (Green/Yellow/Red) to provide instant visual feedback on portfolio gain or loss.

Display Live Statistics (OLED)

Utilize the SSD1306 OLED display to present key metrics, including Invested Value, Current Value, and overall Percentage P/L.

Eliminate Mobile Dependency

Provide a standalone tracking solution that reduces the need to check financial apps, improving focus and reducing screen time.

System Architecture: Block Diagram and Data Flow

The architecture integrates key components for data acquisition, processing, and visual output, forming a cohesive IoT system.

Key Components and Roles

- **DHAN Developer API:** Serves as the authoritative source for real-time portfolio holdings and market data.
- **ESP32 Microcontroller:** The central processing unit, managing Wi-Fi connection, secure data requests, and controlling output devices.
- **ArduinoJson Library:** Essential software component for efficiently parsing the large JSON data payloads received from the API.
- **OLED Display:** Used for detailed, low-power display of numerical portfolio statistics.
- **Traffic Light LEDs:** The core indicator module, providing ambient visual P/L status.

Operational Flowchart: The Continuous Monitoring Cycle

The project operates on a fixed, repeating cycle, ensuring the displayed data is refreshed at regular intervals to reflect market changes.

Start / Boot ESP32

Initializes all hardware peripherals and software libraries upon power-on.

Connect to Wi-Fi

Establishes a stable and persistent connection to the local network using stored credentials.

Fetch API Data

Sends an authenticated HTTPS GET request to the `/holdings` endpoint of the DHAN API.

Parse JSON Data

Uses ArduinoJson to deserialize the response, isolating the required average price, LTP, and quantity fields.

Compute P/L Values

Calculates invested value, profit/loss amount, and percentage P/L for the overall portfolio and individual categories.

Display on OLED

Renders detailed numerical statistics on the SSD1306, cycling through different asset categories.

Set LED Indicator

Activates the corresponding LED (Green for Profit, Red for Loss, Yellow for Neutral).

Wait 15 Seconds

Implements a delay to manage API call limits and maintain a consistent refresh rate before repeating the cycle.

Hardware Interfacing and Pin Configuration

The system relies on minimal, robust hardware components carefully mapped to the ESP32's GPIO pins for efficient operation.

ESP32-WROOM-32	Main Wi-Fi Microcontroller for processing and control.	—
OLED Display (I2C)	Monochrome display for numerical stock data.	SDA→GPIO21, SCL→GPIO22
LED Red (Loss)	Indicates a net loss in the portfolio.	GPIO16
LED Yellow (Neutral)	Indicates minimal change or neutral performance.	GPIO17
LED Green (Profit)	Indicates a net profit in the portfolio.	GPIO5
Power Supply	Device power source (USB or external adapter).	VIN / USB

Powering the ESP32 and peripheral devices through a stable 5V source is critical for Wi-Fi module reliability during data fetches.

Software Stack: Libraries and API Integration

The system's intelligence relies on a suite of specialized Arduino libraries for communication, security, and data handling.

Arduino IDE

Used as the primary environment for coding, compilation, and uploading the firmware to the ESP32 module.

ArduinoJson

Crucial for parsing complex, nested JSON data from the DHAN API efficiently on a low-memory microcontroller like the ESP32.

WiFiClientSecure & HTTPClient

Handles the secure establishment of TLS/SSL connections (HTTPS) and manages the HTTP requests and responses required for API interaction.

Adafruit GFX & SSD1306

These display libraries provide the graphical primitives necessary to draw text, lines, and custom indicators on the OLED screen.

DHAN Developer API

The mandatory external service providing authenticated access to the user's real-time portfolio and holdings data.

Working Principle: Data Extraction and Status Logic

The core functionality is governed by a precise sequence of steps, from network connection to the final visual output.

1. Initialization & Connection

The ESP32 connects to Wi-Fi. It then establishes a secure HTTPS connection (using `WiFiClientSecure`) to the DHAN API endpoint for portfolio holdings.

2. Data Acquisition & Parsing

The raw JSON response from the `/v2/holdings` endpoint is received. ArduinoJson parses this payload to extract three key metrics per security: **average price (avgPrice), quantity, and last traded price (ltp).**

3. Financial Calculation Engine

The extracted data feeds into the system's calculation module. These calculations determine the total invested capital, the raw P/L value, and the current market value.

- Invested = avgPrice × quantity
- P/L = (ltp - avgPrice) × quantity
- Current Value = Invested + P/L

4. Visual Output Logic

The calculated P/L value triggers the visual outputs. The system cycles through different asset categories (Stocks, ETFs, MFs) on the OLED, displaying detailed statistics for 15 seconds each.

5. LED Status Indication

The net portfolio P/L determines the state of the three-color LED indicator:

> **P/L > 0** Green LED (Profit)
>
> **P/L ≈ 0** Yellow LED (Neutral)
>
> **P/L < 0** Red LED (Loss)

Calculation Example: Determining Portfolio Health

A concrete example illustrating how raw data points are transformed into actionable financial metrics for display and LED indication.

Invested Value	avgPrice × quantity	₹103.67 × 22,000 = ₹22,80,787
P/L (Absolute)	(ltp - avgPrice) × quantity	(₹101.7 - ₹103.67) × 22,000 = -₹43,127
Current Value	Invested + P/L	₹22,80,787 - ₹43,127 = ₹22,37,660
P/L %	(P/L / Invested) × 100	-1.89%

-43,127

Absolute P/L

Resulting loss requires the Red LED to be activated.

2.23M

Current Value

The current market worth of the portfolio holdings.

-1.89%

Percentage Change

The overall percentage loss reflected on the OLED.