1. Greeting & Introduction

Good morning. My name is Sivanathan Dilukshika. I'm from the University of South Wales. Today, I'm going to present my individual project titled **'Energy Meter with Data Collecting and Monitoring System.'** This project focuses on developing a low-cost smart energy monitoring solution using IoT.

2. Abstract

The main goal of this project is to allow people to monitor their electricity usage in real time. I designed a smart energy meter using an ESP32 microcontroller, along with two sensors—ZMPT101B for voltage and SCT-013 for current. These components collect energy data, which is shown live on an LCD and on a mobile phone using the Blynk IoT app.

A key innovation in my system is that it logs data to Google Sheets using custom HTTP requests, without relying on premium services like IFTTT Pro. This allows users to keep permanent records of their energy usage in the cloud, completely free of cost.

This helps users visualize trends in their electricity usage, estimate bills, and make more informed decisions to save energy. Because it's low-cost and built with open-source tools, this system can be used in areas that don't have access to expensive smart metering infrastructure. It's also a great learning tool for students.

3. System Architecture & Hardware Explanation

Here is the system architecture. At the heart is the **ESP32 microcontroller**, which reads voltage from the **ZMPT101B** sensor and current from the **SCT-013** clamp sensor. This data is processed and sent via Wi-Fi to the **Blynk IoT platform**, which displays it on a smartphone in real time. I also created a connection to **Google Sheets**, where the data is logged every few seconds for analysis. This setup makes the system live, portable, and low-cost.

4. Comparison & Benefits

Compared to commercial smart meters like OpenEnergyMonitor, my system is **cheaper**, more **flexible**, and **open-source**. It costs less than £20 to build and can be replicated easily. It's ideal for **developing countries**, **students**, or **small businesses** who want to understand their energy use.

5. Live Demonstration or Walkthrough of Blynk + Google Sheets

As you can see here, the Blynk app shows the current voltage, current, power, and energy consumed. The interface updates in real time based on the sensor inputs. Additionally, I've connected my system to Google Sheets using Google Apps Script. This allows me to store the data and analyze it later. This is useful for people who want to track daily or weekly energy usage patterns.
Code explanation : At the beginning of the code, I include all the necessary libraries. These libraries handle Wi-Fi connectivity, Blynk communication, LCD display, and energy monitoring through the EmonLib library." I also define the LCD and sensor calibration values for voltage and current. The calibration ensures accurate measurements from the ZMPT101B and SCT-013 sensors.

Blynk and WiFi Credentials - This section includes the Wi-Fi credentials and Blynk authentication token that allow the ESP32 to connect to the internet and the Blynk IoT dashboard.

Data Collection and Upload Function - The `myTimerEvent()` function is the core of the project. It collects the sensor data every 5 seconds, processes it, displays it on the LCD, sends it to the Blynk app, and uploads it to Google Sheets. It calculates voltage, current, and power using the EmonLib library, and also keeps a running total of energy usage in kilowatt-hours (kWh).

Data Display on Serial Monitor and LCD - I show the voltage (Vrms), current (Irms), power (in Watts), and total energy consumed (kWh) on both the serial monitor and an I2C LCD display. This gives the user immediate feedback on their energy consumption locally.

Blynk App Integration - I send the same values to the Blynk IoT platform, where users can view real-time data from anywhere via their phone.

Google Sheets Integration (Cloud Logging) - Here, I create a custom HTTP request and send the data to Google Sheets using a Google Apps Script URL. This stores energy data over time for analysis. This is a cost-free and reliable way to keep a permanent record without using premium services like IFTTT.

Setup and Loop - In the `setup()` function, I initialize the serial monitor, LCD, Blynk connection, and energy monitoring calibration. I also set the timer to run the data collection function every 5 seconds.

6. **Results & Graphs**

These are some of the readings collected during testing. I compared the results with a commercial multimeter. Though not perfect, the accuracy was within a reasonable margin for non-commercial use. You can see energy consumption trends here — which could help users change their habits, like identifying which times their usage spikes.

7. **Limitations & Future Work**

Currently, the system depends on Wi-Fi, and doesn't work offline. It also doesn't yet support mobile alerts or smart switching. In the future, I'd like to add SD card data backup, automatic load control, and an AI-based prediction model to forecast energy use.

8. **Conclusion**

To conclude, this project demonstrates how a low-cost system using ESP32, Blynk, and sensors can help people monitor their energy usage in real time, log data to the cloud, and encourage energy-saving behavior. It's simple, scalable, and socially impactful.

9. **Q&A Time -** Thank you for listening. I'm happy to take any questions.

Technical Questions

1. **Why did you choose ESP32 instead of ESP8266?**
   - *ESP32 offers better performance, more GPIO pins, and built-in Bluetooth. Though I mentioned ESP8266 in some parts, I used ESP32 in the final implementation.*
2. **How accurate are the voltage/current readings?**
   - *Accuracy depends on proper calibration of ZMPT101B and SCT-013. I tested against a multimeter and saw acceptable results for non-commercial usage.*
3. **What happens if Wi-Fi goes offline?**
   - *Currently, the system relies on Wi-Fi, so real-time updates pause. As a future enhancement, I plan to add SD card backup or local storage.*
4. **How did you link Google Sheets without IFTTT?**
   - *I used Google Apps Script and a custom HTTP request to send data directly from Arduino to Sheets—saving costs and offering more flexibility.*
5. **How is your system better than OpenEnergyMonitor or Shelly EM?**
   - *It's more affordable, open-source, and customizable. Unlike proprietary solutions, mine can be tailored for educational or local community use.*

Evaluation-Oriented Questions

1. **What was the biggest challenge you faced?**
   - *Integrating Google Sheets without premium tools and calibrating sensors for accuracy.*
2. **What makes your project innovative?**
   - *Combining real-time monitoring, low-cost components, and cloud logging without paid services, especially suitable for developing regions.*
3. **How can this be expanded commercially or socially?**
   - *It can be scaled to smart communities, integrated with billing alerts, or adapted for educational kits.*