Smart OEE System

status active

Smart OEE System

Table of Contents

About

This repo contains

- Firmware
- Circuit Diagram
- Detailed instructions

for Smart OEE System.

Getting Started

These instructions will get you a copy of the project up and running on your system.

Prerequisites

Things you need to install the FW.

```
- Arduino IDE
```

Installing

A step by step series that tell you how to get the Firmware and Backend running

ESP32 Configuration

You should have Arduino IDE Installed

1. Add ESP32 Board to your Arduino IDE 1. In your Arduino IDE, go to File> Preferences Installing ESP32 Add-on in Arduino IDE Windows, Mac OS X, Linux open preferences 2. Enter `https://dl.espressif.com/dl/package_esp32_index.json` into the “Additional Board Manager URLs” field then, click the “OK” button: Note: if you already have the ESP32 boards URL, you can separate the URLs with a comma(each board will go to neaw line) as follows: `https://dl.espressif.com/dl/package_esp32_index.json,\n http://arduino.esp8266.com/stable/package_esp8266com_index.json`
2. Open the Boards Manager. Go to Tools > Board > Boards Manager…
3. Search for ESP32 and press install button for the ESP32 by Espressif Systems“:
4. That’s it. It should be installed after a few seconds.
5. In your Arduino sketchbook directory, create tools directory if it doesn't exist yet.
6. Unpack the tool into tools directory(present in libs/ESP32FS-1.0.zip) (the path will look like <home_dir>/Arduino/tools/ESP32FS/tool/esp32fs.jar).
7. Close and re-open the Arduino IDE.
8. Now copy the contents of the libs folder to the libraries directory of your Arduino
    1. If you are using windows, the libraries directory will be Documents/Arduino/libraries

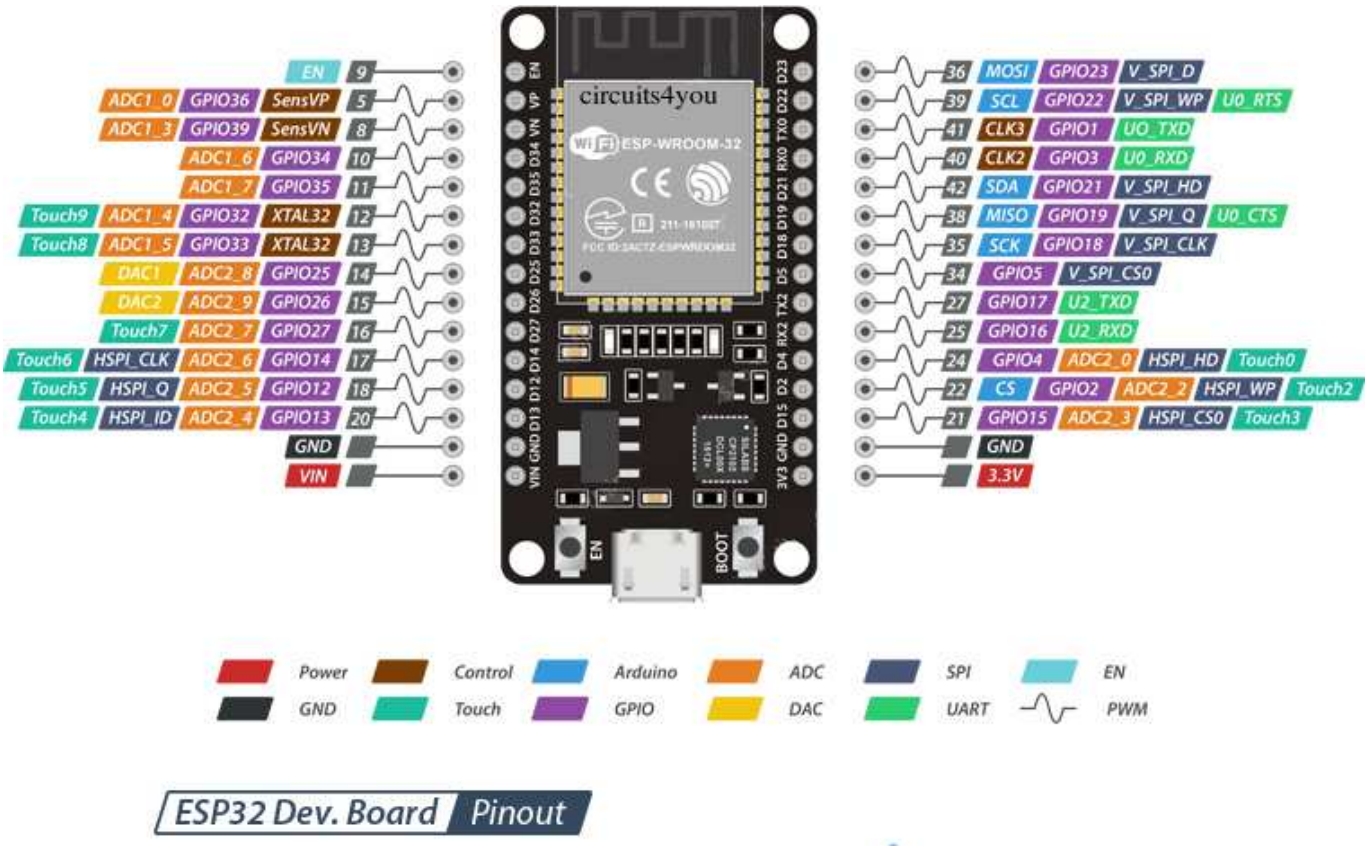
ESP32 Node FW Uploading

1. Select ESP32 Dev Module from Tools->Board->ESP32
2. Select the correct port from Tools->Port
3. Then open Firmware.ino file,
4. Select Tools > ESP32 Sketch Data Upload menu item. This should start uploading the files into ESP32 flash file system.
5. Now Upload the Code to your ESP32 Dev Module.
6. Your ESP32 is now ready to be used.

Circuit

ESP32 Dev Module Pinout

Follow the pinout diagram given below to connect different components to your TTGO LORA32 board.



Other Components

```
Other components pin connection details
```

Temperature Sensor DHT22

DHT22 Connections

DHT22 Pins	ESP32 Dev Module Pins
DATA OUT	23
VCC	5V
GND	GND

Photoelectric Sensor

Photoelectric Sensor Connections

From 2x 10K Resistors' center to Pin 18 of ESP32

Status LED

LED Connections

LED Pins	ESP32 Dev Module

LED Pins	ESP32 Dev Module
Anode	D2 via 220Ω resistor
Cathode	GND

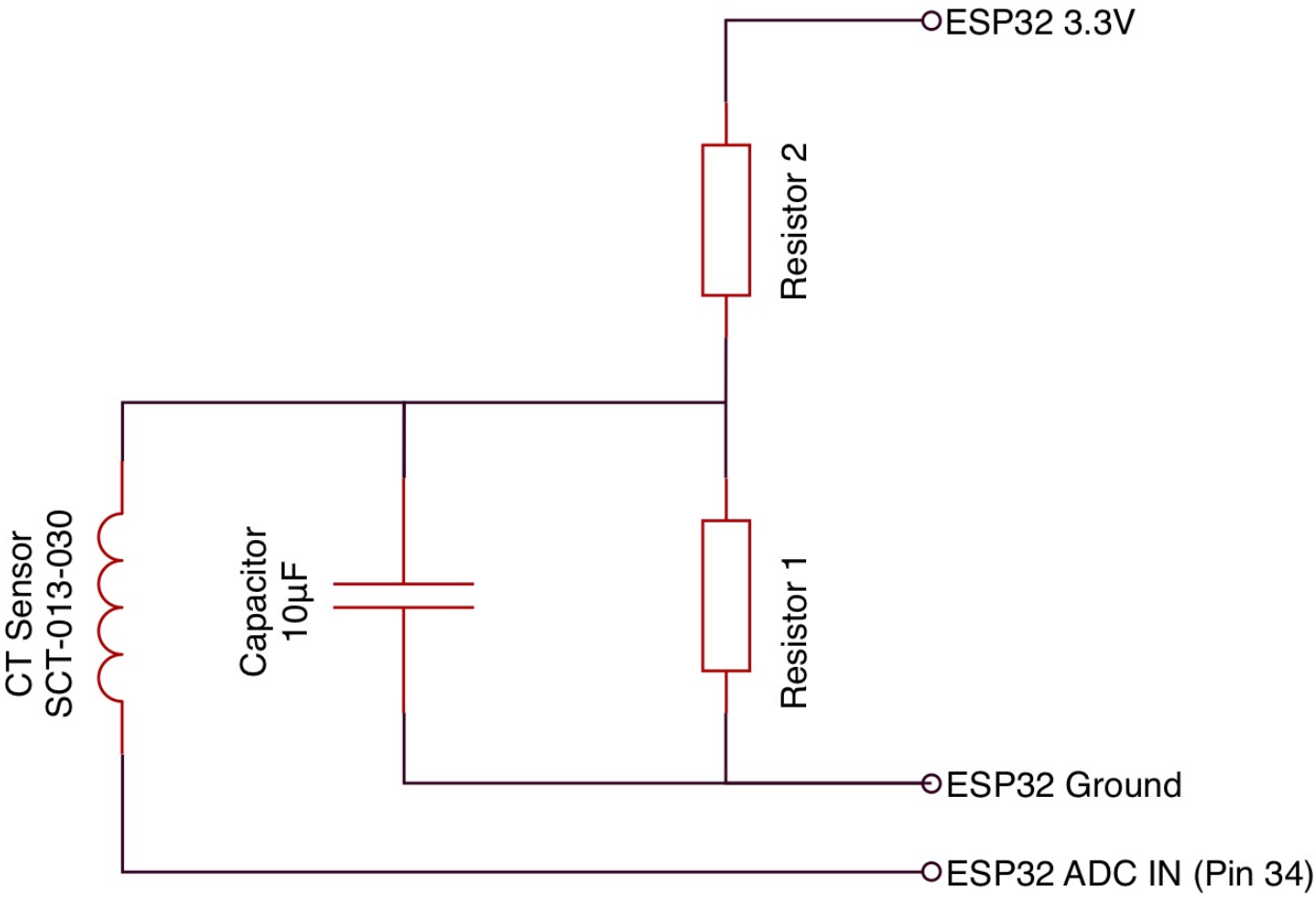
D2 is also connected to the internal LED of ESP32 Dev Module

SCT-013

SCT-013 Connections

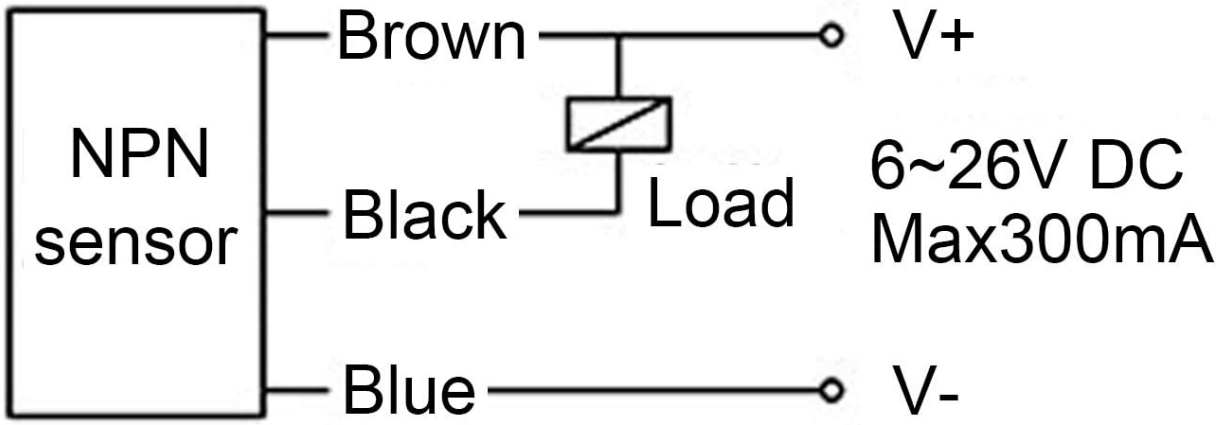
- Voltage Divider with 2x 100KΩ resistors.
- 10uF capacitor connected between Voltage Divider Circuit Output and GND
- 3.5mm Audio Jack connected between Voltage Divider Circuit Output and ESP32 Pin 34.

The overall SCT-013 connection assembly will look something like shown in the diagram below.



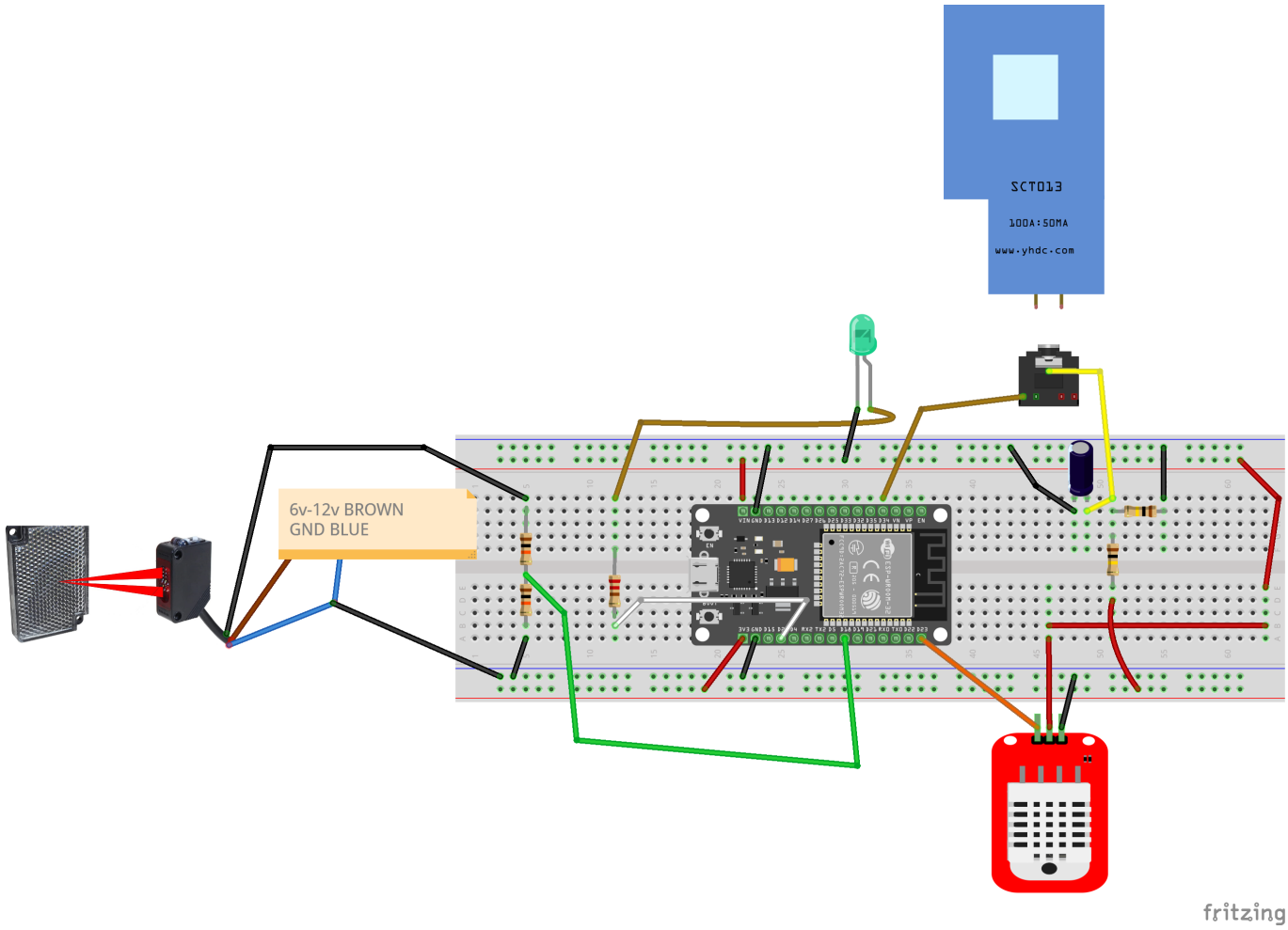
The photoelectric sensor has three wires as shown below

Circuit Wiring Diagram



Complete Circuit Diagram

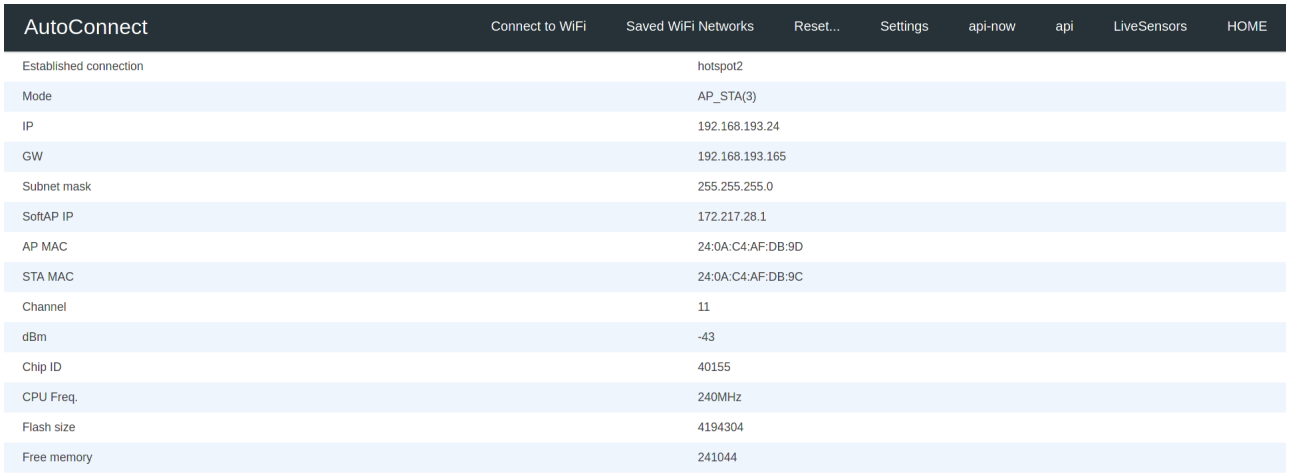
Here's the complete circuit diagram of the system.



Usage

1. Power on your ESP32, it will present you with an AP named `OEE` (while `OEE` can be changed in the portal)
2. Default captive portal password `12345678AP` which can be changed in captive portal.
3. Connect to the ESP32 access point and open the web-browser and navigate to the link `http://esp32.local/_ac`. This link will work on most of the operating systems but if your operating system is not allowing to open it, you may want to check the captive portal IP Address from the serial monitor and can use that IP address inplace of the above mentioned URL.
4. The default access IP Address is `http://192.168.4.1/_ac`

5. You will be presented with a main dashboard as shown below(based on your device)



6. Once connected to a WiFi network, you can again access the captive portal using same URL or the IP Address from the Serial monitor.

7. The data is published to the MQTT Topic `OEE/{hostname}` while the hostname is the one which you can define in Settings page of the captive portal.

Changing Timezone

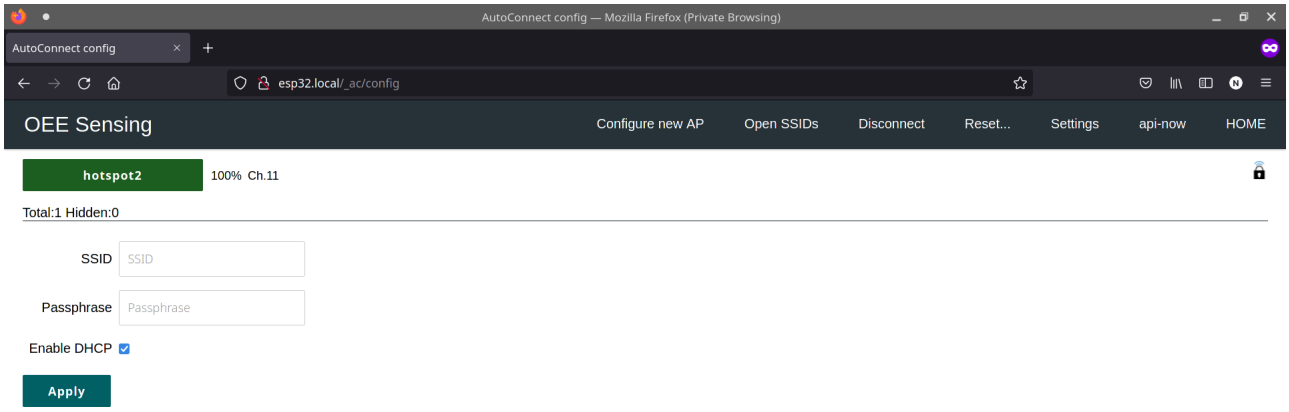
1. Open Settings tab
2. Enter timezone string from https://en.wikipedia.org/wiki/List_of_tz_database_time_zones 'TZ database name' column.
3. Click Save&Start

API Endpoints and HTML URLS

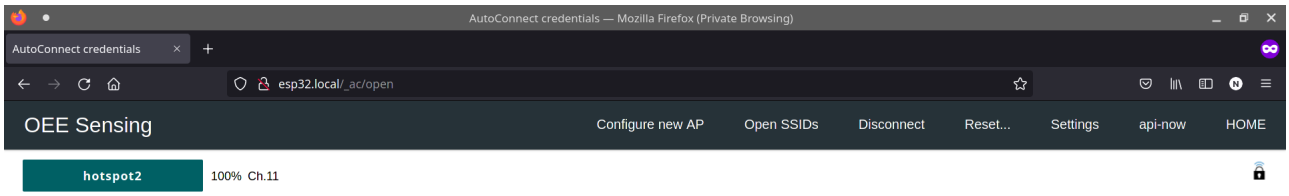
`API Endpoints`

Endpoint	Description
`/api-now`	`API: live sensor readings in JSON format`
`/api`	`API: historical sensors data in JSON format`
`/LiveSensors`	`HTML PAGE: Live Sensor Data`
`/data`	`HTML PAGE: Historical Sensor Data`
`/mqtt_settings`	`HTML PAGE: Settings. Default username: AP Name, Default Password: admin`
`/_ac`	`HTML PAGE: Main Captive portal page`
`/`	`HTML PAGE: Historical Sensor Data`

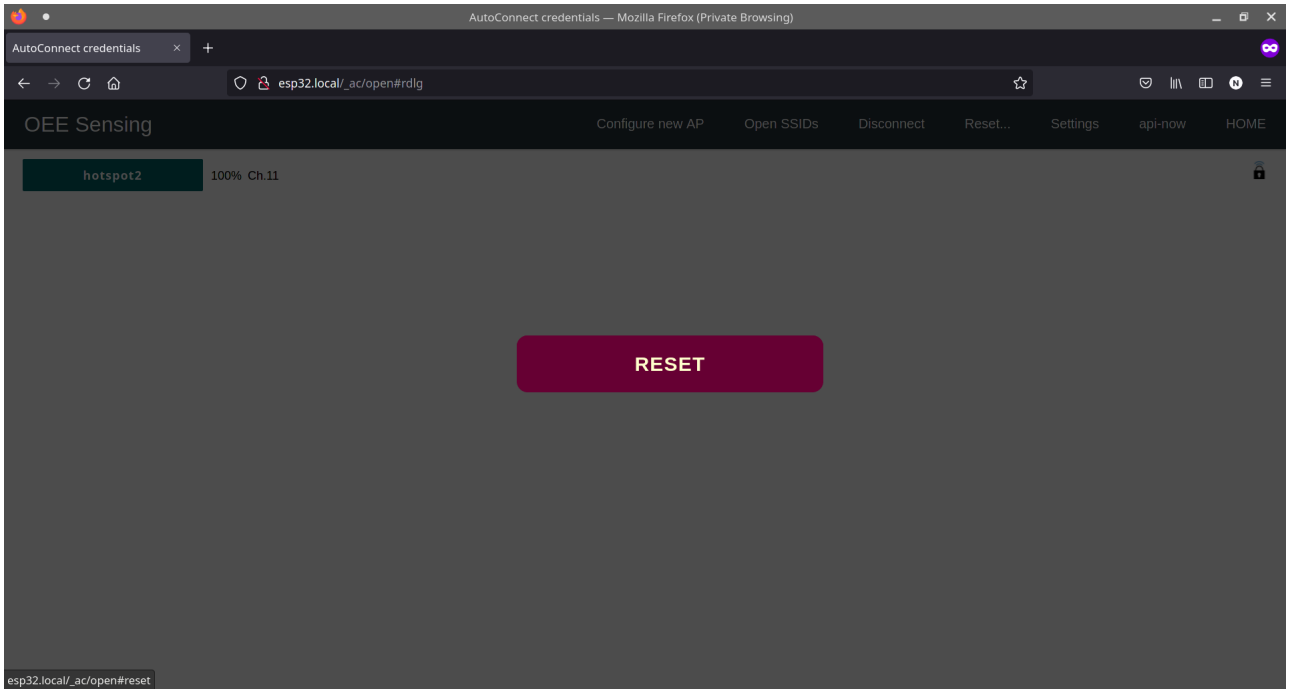
1. **Connect to WiFi** tab allows searching of nearby WiFi APs and adding them to the ESP32.



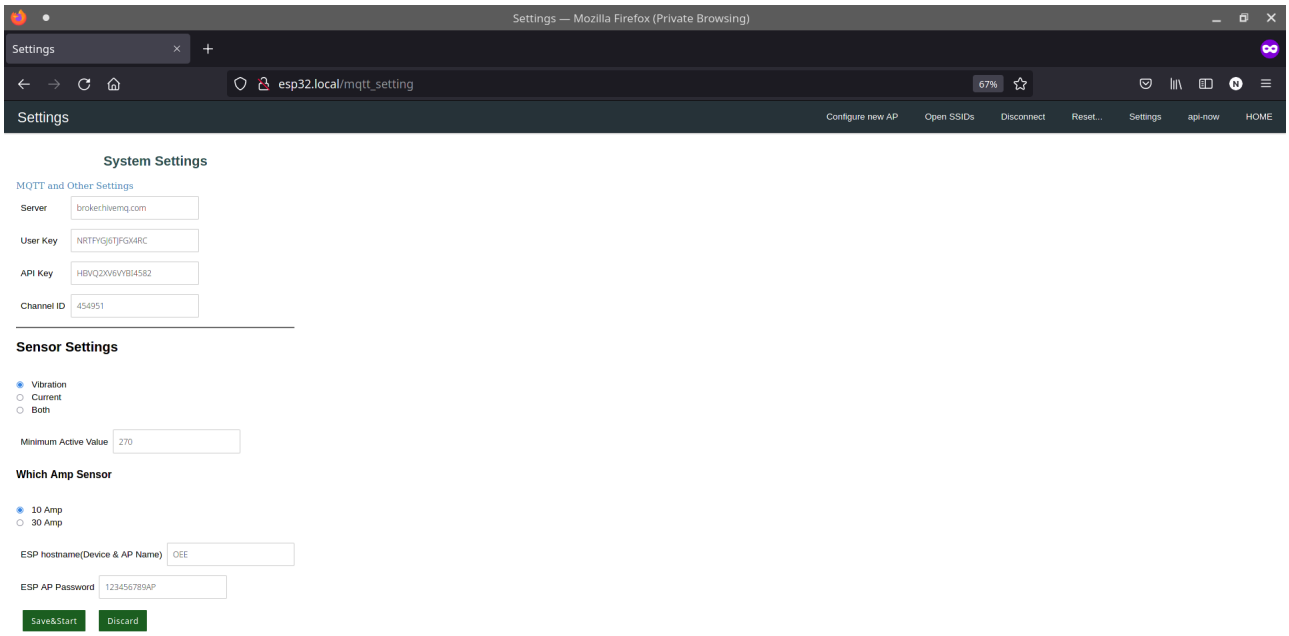
2. **Saved WiFi Networks** tab allows connecting to the saved access points.



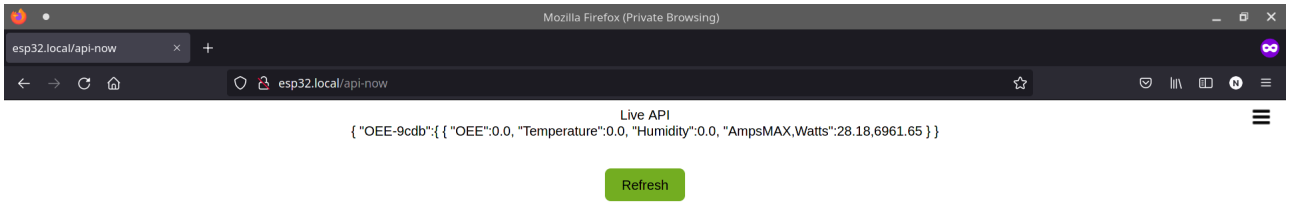
3. **Reset...** tab allows reseting of the device to factory settings.



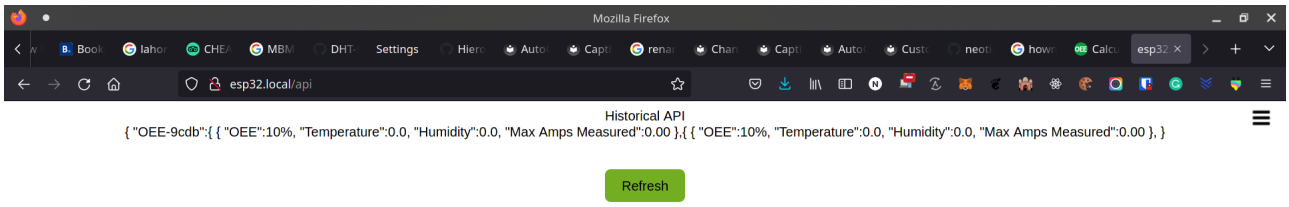
4. **Settings** tab contains settings related to MQTT and sensors.



5. **api-now** tab gives the live-sensor data in JSON format.

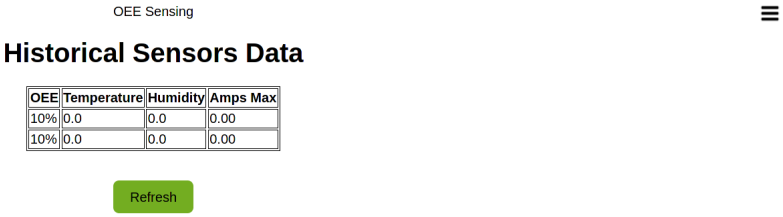


6. **api** tab gives the historical sensor data acquired after every 5 mintues.



7. **LiveSensors** tab shows live values of the sensors.

8. **HOME** tab shows historical sensor data in a HTML table form acquired after every 5 minutes.



List of Components

Following components are used to make this project

1. ESP32 Dev Kit Module
2. Current Sensor (SCT-013)
3. DHT22 Temperature and Humidity Sensor
4. Vibration Sensor (IMU) (MPU6050)
5. Photoelectric Sensor
6. Generic 3.5mm Green/Red/Blue LEDs
7. Generic 3.5mm Audio Jack with Screw Terminals
8. Generic 10uF 25v Capacitor
9. Generic 220Ω Resistors

10. Generic 10KΩ Resistors
11. Generic 100KΩ Resistors
12. Generic 5V USB Micro B cable and adapter
13. Micro USB Cable
14. Breadboard, Jumper Wires and Power Supply Module
15. Jumper Wires
16. 12V 3A Adapter

🔨 Built Using

- Python - For Cloud Gateway Pogramming
- Arduino - Embedded Framework and IDE - For Sensor Node Design

📼 Demo Videos

✍️ Authors

- @Nauman3S - Development and Deployment