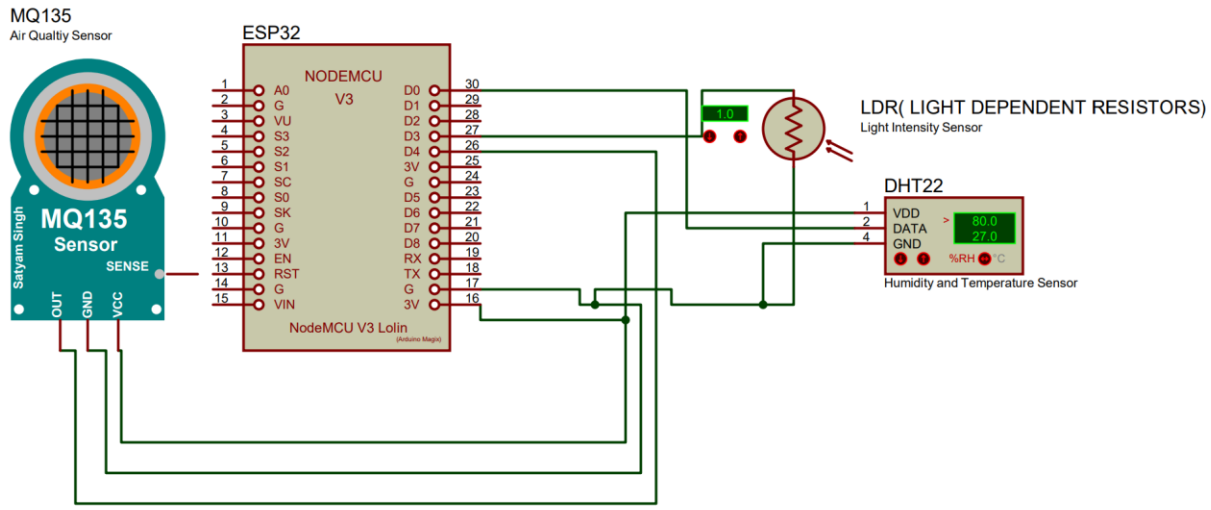
IoT Weather and Environment Monitoring System

Sample Circuit:



This is the sample circuit of our project. In this setup:

- Sensors such as the LDR, DHT22, and MQ135, as mentioned in the previously submitted Phase 2 document, are used to collect data.
- The ESP32 microcontroller, known for its network capabilities and Bluetooth connectivity, is employed as described in the Phase 2 document.
- The MQ135 sensor is connected to the D4 pin of the ESP32 to monitor air quality. The MQ135 is connected to the ground and 3V output of the ESP32.
- The DHT22 sensor is connected to the D0 pin of the ESP32 microcontroller to measure temperature and humidity. The DHT22 is also connected to the ground and 3V output of the ESP32.
- The LDR sensor is connected to the D3 pin and the GND (ground) of the ESP32 sensor to detect light levels.

Please note that this circuit is a sample, and the final project's configuration may vary.

Data Storage:

Data storage is essential as the collected data needs to be processed and displayed. Since the primary objective of the project is Serverless IoT processing, the data should be stored in a suitable medium. We propose using GitHub as a data storage solution because it provides a robust API for storing and retrieving data. Data will be timestamped, enabling visualization, processing, and analysis according to date and time. This timestamped data can also be valuable for research purposes.

Web Application:

The web application is designed as a static site. Given that the primary project goal is Serverless IoT processing, static sites don't require a dedicated server for hosting. This web application can display real-time data as well as historical data. Data visualization is made possible using the JavaScript library called Chart.js, which is an efficient way to process data without relying on a server.

Programming of Microcontroller:

For programming the microcontroller in this project, we plan to use MicroPython. MicroPython is a suitable choice due to its simplicity and ease of handling APIs. It simplifies the task of programming the microcontroller for our specific needs.

Simulation of the Project:

We conduct the simulation of the microcontroller using a platform called Wokwi. Wokwi is chosen because it is user-friendly and offers the advantage of creating custom chip models, allowing us to model sensors like the MQ135 that may not be available in other simulators.

Conclusion:

We have initiated the coding of the microcontroller and will subsequently begin working on the web application interface. We will submit the microcontroller program and the web application UI in the next phase of the project. Thank you for your attention.