GE107 Tinkering

End - Sem Project Report

Title: Indoor air quality monitor with a web dashboard

Group 33

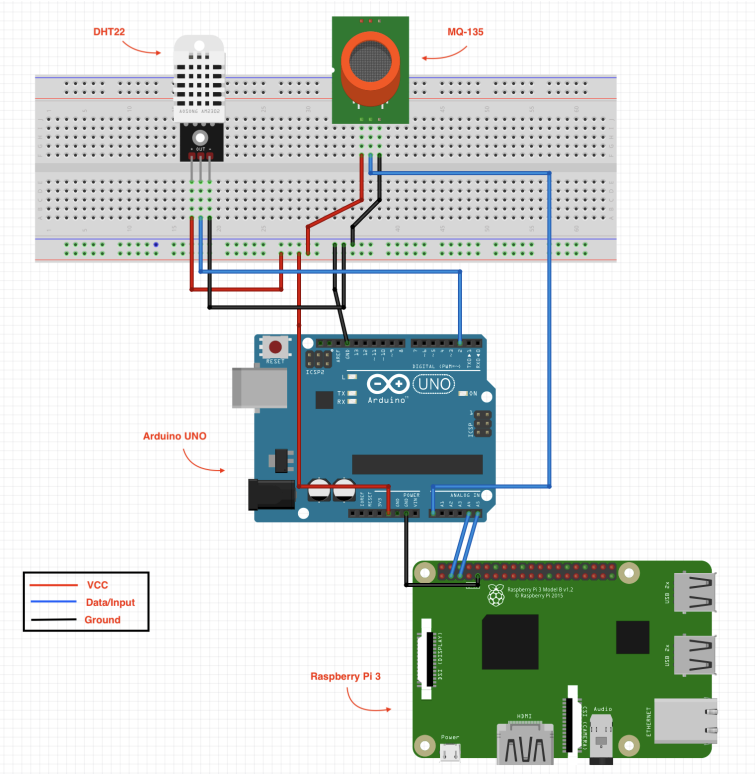
Problem statement

Design and develop an indoor air quality monitoring system using an Arduino Uno and environmental sensors. The system should periodically collect data from the sensors, evaluate the air quality, and transmit the information via a Raspberry Pi to a web dashboard accessible to the user.

Component Used

1. Arduino UNO
2. Raspberry Pi Model 3
3. DHT22 Sensor
4. MQ-135 Sensor
5. Bread Board
6. Jumper Wires

Circuit Diagram and workings



1. Data Flow Architecture

1. **Sensor Layer**: DHT22 and MQ-135 collect temperature, humidity, and gas concentration data.
2. **Microcontroller Layer**: Arduino reads values and sends them via I2C.
   Edge Device Layer: Raspberry Pi receives data and transmits it via HTTP POST requests.
3. **Backend Layer** :
   - Stores data in a local database (e.g., SQLite/PostgreSQL).
   - Updates the web dashboard in real-time.
   - Feeds data to the ML model for prediction.
4. **Frontend Layer**:
   - A web dashboard that displays live temperature, humidity, and PPM values.
   - Visualizes predicted vs. actual pollutant levels using graphs.

2. Machine Learning Integration

- **Model Used**: Random Forest Regressor
  Input Features:
  - Current and past PPM readings
- **Target**: Next-minute PPM value
- **Training**: Model is trained on historical data stored in the database
- **Tools Used**: Python (scikit-learn, pandas), Jupyter Notebook
- **Purpose**: Enables proactive insights into worsening air quality.

3. Web Dashboard

- **Technologies Used**:
  - Backend: Flask/Django (Python-based API)
  - Frontend: HTML, CSS, JavaScript
  - Communication: HTTP POST from Raspberry Pi to backend
- **Features**:
  - Real-time sensor readings display
  - Historical data graphs
  - Predicted PPM values visualization
  - System status updates

Problem Faced during the project

We encountered two major challenges during the development of our indoor air quality monitoring system:

1. **Inability of Raspberry Pi to Read Analog Values Directly**
   The MQ-135 gas sensor provides analog output, but the Raspberry Pi lacks an onboard analog-to-digital converter (ADC), making it incapable of directly reading analog values. To resolve this, we integrated an Arduino Uno into the system. The Arduino collects data from the MQ-135 sensor, processes it, and transmits the readings to the Raspberry Pi using the I²C communication protocol.

2. **Performance Limitations While Hosting the Web Dashboard on the Raspberry Pi**
   Initially, we planned to host the entire web dashboard and backend on the Raspberry Pi itself. However, due to its limited resources—specifically 1 GB of RAM—the system became noticeably sluggish and unresponsive under load. To address this, we shifted the backend and web dashboard hosting to a separate laptop (MacBook). The Raspberry Pi now acts as a data forwarder, sending sensor readings to the laptop via lightweight HTTP POST requests, ensuring smooth performance and real-time data visualization.

Individual Contribution

1. Machine Learning and Backend Development

- **Ram Nivash(2023MEB1375)**:
  - Developed and implemented the Machine Learning model using a Random Forest Regressor for predicting future air quality values.
  - Designed and worked on the backend of the web dashboard, handling the data processing, storage, and communication between the Raspberry Pi and the dashboard.

2. Hardware Setup and Data Transmission

- **Nikhil Mee(2023MEB1362)I**:
  - Managed the hardware setup(setting raspberry pi), including connecting and configuring the sensors
  - Responsible for the collection and transmission of data from the hardware to the web dashboard using Arduino and Raspberry Pi via I2C and HTTP protocols.

- **Mohit Kumar(2023MEB1359)**:
  - Assisted in the hardware setup, ensuring the proper wiring and functioning of the sensor components

3. Frontend Development

- **Palak(2023MEB1364)**:
  - Developed the frontend of the web dashboard, creating an intuitive user interface that displays real-time sensor data and predictions.
  - Implemented dynamic charts and graphs to visualize the air quality data effectively.

4. Backend Development

- **Pooja Pachar(2023MEB1367)**:
  - Contributed to the backend development, setting up the server for handling HTTP requests and managing the data flow from the Raspberry Pi to the dashboard.
  - Worked on integrating the database with the backend and ensuring smooth communication between the sensors, database, and frontend.

Conclusion

This project successfully combines hardware, networking, and machine learning to build a functional indoor air quality monitoring system. By using Arduino Uno to handle analog sensors and communicating with the Raspberry Pi via I²C, we overcame hardware limitations. Due to the Pi's limited RAM, we hosted the backend and web dashboard on a separate laptop and transferred data via HTTP.

The system not only displays real-time data through a clean web dashboard but also predicts future air quality using a Random Forest Regressor. Overall, this project gave us hands-on experience in embedded systems, data handling, and machine learning, while also teaching us how to solve real-world engineering challenges as a team.

Code:
https://github.com/Ram-Nivash/dashboard.git