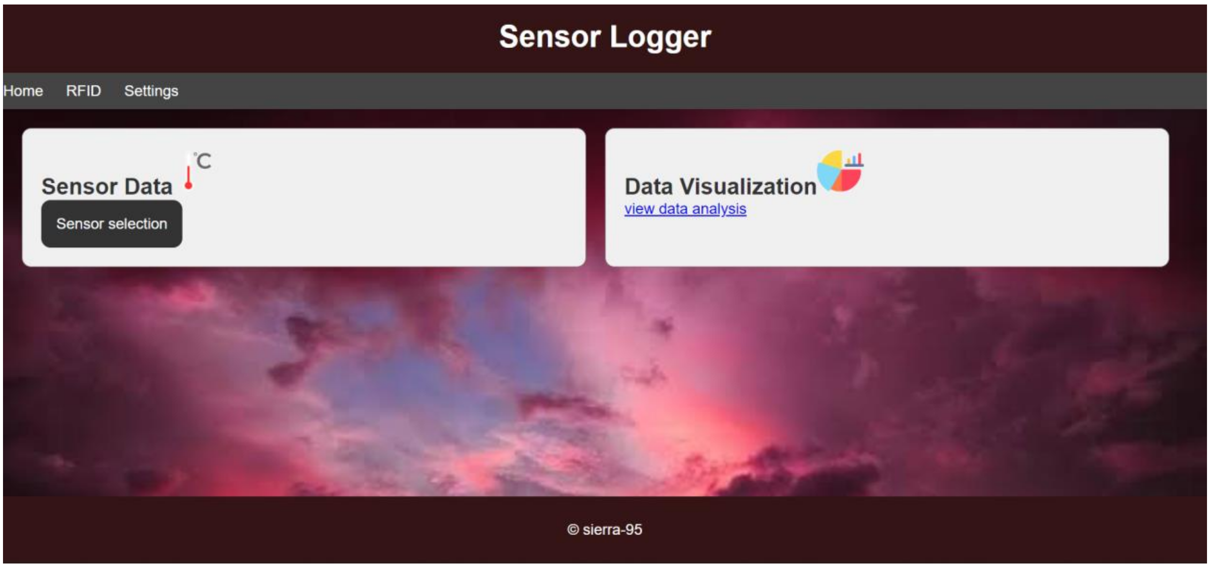
THE SENSOR LOGGER PROJECT



For a while, data collection has been carried out manually at many stations, leading to a gradual increase in paperwork compared to previous years. Even today, despite the shift to computerized data analysis in most stations, data collection still relies on traditional instruments. While this approach is not inherently wrong, it has limited the potential for automation in these fields, resulting in a lack of improvement.

Over time, sensors have been meticulously designed to provide accurate and reliable readings. With their seamless integration into automation systems, they have become the preferred choice. This project aims to streamline data collection by utilizing the Internet of Things; specifically, integrating Internet-connected sensors to automate data collection in the field of statistics and data analysis. Although currently limited to two sensors (DHT11 and RFID), the project demonstrates the feasibility of automating data collection.

I took on the responsibility of overall project management, which included system design, sensor integration, and web development. The goal was to create a user-friendly dashboard enabling users to view raw or analyzed data in graphical formats. Additionally, users would have the capability to export collected data for further analysis

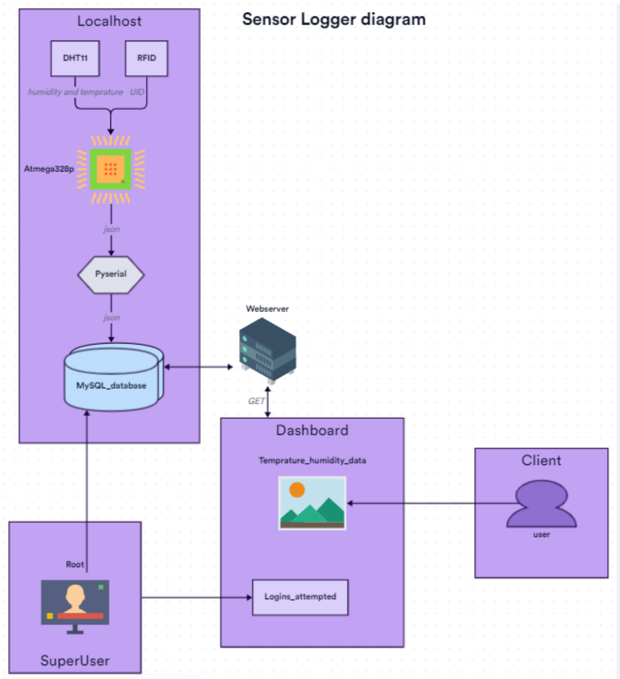
Why this project?

A few years ago, I visited my aunt who resides in Central Kenya, situated in a tea estate where daily weather data collection is crucial. Despite her office responsibilities, she had to manually collect data at 9 a.m. and 3 p.m. At that time, I was still in high school, and while I lacked experience, I couldn't help but consider a solution when I saw an IoT project listed on Holberton's portfolio list. This is when the idea for the sensor data logger began to take shape. I have worked day and night to develop a solution hoping that it would be of help to her, and other dedicated data extractors like her.

Architecture

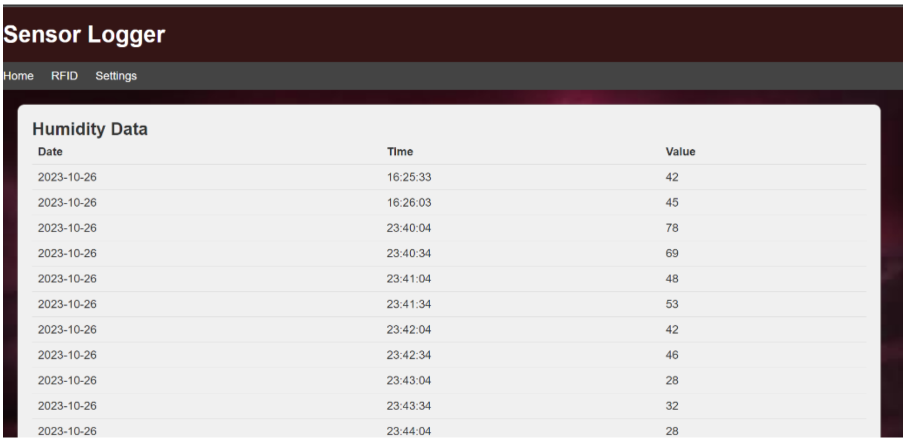
The project is outlined in three main phases: data population, data storage, and data projection.

1. **Data Population:**
    - The Atmega32p serves as the central hub for data collection, interfacing with various sensors like the DHT11 (for temperature and humidity) and RFID sensors (for Unique Identifier - UID).
    - The DHT11 sensor provides real-time environmental information, including temperature and humidity levels.
    - Simultaneously, the RFID sensor captures Unique Identifiers (UIDs) associated with RFID cards.
2. **Data Storage:**
    - Data collected from the sensors is mapped using PySerial to a MySQL database, where it awaits retrieval and posting for users to access.
3. **Data Projection:**
    - The project includes a user-friendly web dashboard created using HTML, CSS, and JavaScript for the front end.
    - Flask was employed for back-end development due to its simplicity.
    - Data is visualized and monitored through this interface, offering users a comprehensive view of sensor-based systems.



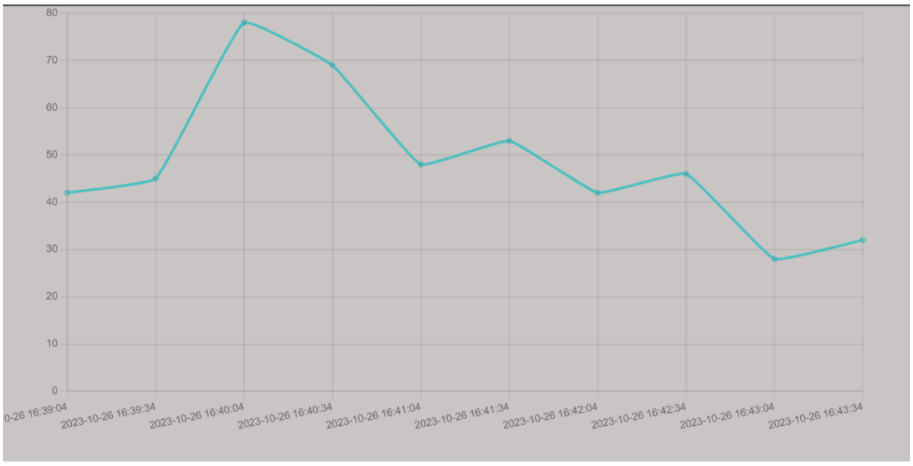
Features:

1. Users have the abilit instantly access real-time raw data from both humidity and temperature sensors, providing a quick and direct insight into environmental conditions.



Date	Time	Value
2023-10-26	16:25:33	42
2023-10-26	16:26:03	45
2023-10-26	23:40:04	78
2023-10-26	23:40:34	69
2023-10-26	23:41:04	48
2023-10-26	23:41:34	53
2023-10-26	23:42:04	42
2023-10-26	23:42:34	46
2023-10-26	23:43:04	28
2023-10-26	23:43:34	32
2023-10-26	23:44:04	28

2. Graphical data can be viewed and analyzed by just clicking a button, no more Excel or manual plotting, the project does it all for you.



3. Superusers have the privilege to access RFID data, enabling them to track access control cards and obtain insights into individuals' clock-in times, fostering enhanced security and accountability.

Technical challenges:

One significant hurdle I faced involved master-slave replication. While I had basic networking knowledge, handling IP addresses for laptops and servers became unexpectedly challenging due to a laptop's ability to change IP addresses dynamically with the help of DCHP.

In my frustration, a friend suggested using Mac addresses, which at the time seemed to be a solution. However, a good night's sleep led me to a simpler solution. The next morning, I found clarity by using SCP to send data to the server and then mapping it to the database, effectively resolving the challenge.

What I have learnt:

1. Technical takeaways:
    - I have acquired proficiency in web frameworks, specifically Flask, enhancing my skills in building dynamic and responsive web applications.
    - I also developed expertise in implementing secure login functionalities and successfully configured a reverse proxy server, showcasing a comprehensive understanding of server-side operations.
2. What I might do differently:
    - I aim to enhance the dashboard functionality by incorporating a feature allowing users to export data for further analysis. This addition aligns with the project's goal of facilitating comprehensive data utilization.
3. What I learned about myself as an engineer:
    - I discovered a persistent and determined aspect of myself throughout the project, highlighting my commitment to overcoming challenges and ensuring project success. This resilience underscores my dedication to achieving engineering goals.
4. How this project influenced my engineering path in the future:
    - This project marks the initial step in combining my knowledge in mechatronics and software engineering, showcasing the potential for interdisciplinary projects. This experience will guide my future endeavors, influencing the direction of my engineering pursuits.