Option B: Weather Client - Progress Report

John Maksuta   
Computer Science Department  
CS 370-801  
Colorado State UniversityFort Collins, CO  
jmaksuta@colostate.edu  
  
Ali Fayed  
Computer Science Department  
CS 370-801  
Colorado State UniversityFort Collins, CO  
alifayed@colostate.edu  
  
Matthew Boin  
Computer Science Department  
CS 370-801  
Colorado State UniversityFort Collins, COmboin@colostate.edu

*Abstract*—This is the first progress report for the Term Project for CS370-801 for our group 8B. The project objectives and justifications are introduced. Description of the Hardware, Software Environment, Project Design, and Software development. (*Abstract*)

Keywords—component, formatting, style, styling, insert (key words)

# Introduction

The project objective is to develop and evaluate a system built using a single-board computer. The hardware requirement specifications are a single board computer, a sensor device, and communicating with at least one other computer. The single-board computer must have both Wi-Fi and OS boot capability.

Our project will measure, monitor, and report environmental factors, such as temperature, air pressure, humidity, and air quality over the Internet to a client machine. This progress report describes the current status of the project and our intent to evaluate quantitative information about the project. At the end of the project, we will submit a final report and a demonstration of the application we have produced.

We have chosen the Raspberry Pi 3 Model B+ for our single board computer [9]. It is the final revision of the third-generation single-board computer with Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC 1.4GHz processor, and 1GB LPDDR2 SDRAM. They are running the Raspberry Pi OS, formerly “Raspbian”, which is a “Debian-based operating system optimised for the Raspberry Pi hardware” [10].

For our sensor, we have chosen Adafruit BME688 - Temperature, Humidity, Pressure, and Gas Sensor - STEMMA QT [1]. The sensor features the Bosch BME680 Low power gas, pressure, temperature & humidity sensor [2]. Which is a precision device capable of measuring “humidity with ±3% accuracy, barometric pressure with ±1 ℎ𝑃𝑃𝑃𝑃 absolute accuracy, and temperature with ±1.0°𝐶𝐶 accuracy,” and can be used as “an altimeter with ±1 meter or better accuracy!” [1].

# Progress

## Obtaining the Hardware

Due to our remote team location, our team chose to obtain hardware separately rather than having a shared device to develop with. Although working individually, we have the same hardware board and sensor. We chose the Raspberry Pi 3 B+ model computer board [8] and the Adafruit BME680 sensor board [1].

## Software Environment

Identify applicable funding agency here. If none, delete this text box.

We are developing individually using Visual Studio Code IDE, and our code is maintained in a repository on GitHub (<https://github.com/CS-370-801-Fall24-Term-Project/Fall2024TermProject.git>). We are developing directly on the board, through ssh protocol from our development machines.

## Project Design

Identify applicable funding agency here. If none, delete this text box.

Our design has two components, primary and secondary. The primary component of the design is to have a website using a Node.js React client-side with a Java Spring-boot server-side. The server-side code will have direct communication to the sensor using I2C communication to the Adafruit BME680 sensor board. The Java interface to the sensor will use Pi4J to gain access to the GPIO headers on the Raspberry Pi and use I2C communication pins to communicate directly with the sensor board. A client will use the React UI to make a REST API request to the server side that will communicate directly with the sensor and obtain live data, returning it to the client and displaying it to the user.

The secondary design component is an application or service running in the background, separate from the primary component website client-server environment, that polls the sensor periodically and saves data to a database. This allows the client side to make requests to the server for historical data. For historical data client requests, the server side will directly access the same database that the background service will be inserting records into.

## Software Development

Identify applicable funding agency here. If none, delete this text box.

We created the base code for our website using NPX create-react-app template. The server-side base code was created using the Spring Initializer [3]. We added a basic API endpoint using the Spring Framework to demonstrate the technology [4][5]. We created the project for the Adafruit BME680 interface using Pi4J library for communicating using the I2C communication protocol through the computer’s onboard GPIO header pins [8]. This project will be packaged as a JAR and referenced as a dependency in the server-side backend.

# Development

Sensor Integration - Python works with the BME688 sensor wired to the Raspberry Pi through I2C protocol and successfully printing out temperature, humidity and air pressure readings.

Communication Setup - We verify that the Raspberry Pi will talk to client machine through HTTP and socket programming.

UI Development – But so far, we have not made much progress just yet, and we plan to incorporate live data visualization on the web interface using Java-based Spring Boot.

## Power Consumption Estimates and Measurements

Energy conservation is paramount, particularly for IoT devices that are expected to operate continuously. monitoring power consumption from the raspberry pi circuit and the sensor circuit with time will check the possibility of the system for long term use.

Evaluation Plan

* Measure power consumption using a wattmeter to track the current drawn by:
* Raspberry Pi when it is in idle mode and when it is under load.
* active data collection and the sensor during sleep modes.
* Estimate total power usage - Express the daily and monthly energy consumption in watt-hours (Wh) and estimate the yearly power cost.
* Comparison with benchmarks - Use data on other similar IoT devices to compare the amount of power consumed to assess energy efficiency.

Expected Data

* Idle Power Consumption
* Peak Power Consumption
* Projected Energy Cost - Assuming a power cost of $0.13 per kWh

## Cost and Marketability Analysis

The project's hardware components consist of the Raspberry Pi 3 B+ ($35.00) and Adafruit BME688 sensor ($19.95). Additional required components include power supply ($7.99), SD card ($8.99), case ($9.99), and cables ($5.99), bringing the total prototype cost to $87.91.

Market Analysis

The environmental monitoring sector is experiencing growth driven by increased awareness of indoor air quality and its health impacts. Our analysis of current market offerings identifies two primary competitors:

* AirThings ($299): Features comprehensive monitoring with mobile integration and requires subscription for advanced features.
* Inkbird ($89.99): 6-in-1 indoor air quality monitor.

Our solution targets three primary market segments:

* Educational institutions requiring cost-effective monitoring solutions
* Small businesses needing environmental compliance monitoring
* Home automation enthusiasts seeking customizable platforms

Competitive Advantages

Our system differentiates itself through:

* Open API architecutre enabling custom integration
* Local data processing eliminating subscription requirements
* Multiple sensor integration (temperature, humidity, pressure, air quality)

Financial Viability

Analysis supports a retail price point of $149.99, positioning our solution between basic DIY kits and premium commercial solutions. The open architecture and competitive pricing provide significant market entry advantages, particularly in the educational and maker segments.

## Figures and Tables

A black rectangular object with wires

Description automatically generated

1. John Maksuta’s Adafruit BME680 Sesnor Board and Pi Case.

A black electronic device with wires and wires

Description automatically generated

1. John Maksuta’s Raspberry Pi Board and case, with GPIO header exposed and sensor leads connected.

A green circuit board with red and blue wires

Description automatically generated

1. Ali Fayed’s Raspberry Pi Board, with GPIO header exposed and sensor leads connected.

A green circuit board with many wires

Description automatically generated

1. Adafruit BME680 Sesnor Board and Raspberry Pi Board wired with I2C [12]

A screenshot of a computer

Description automatically generated

1. Ali Fayed’s testing environment using Python.

A screenshot of a weather dashboard

Description automatically generated

1. UI Development progress.

A green circuit board with red and blue wires

Description automatically generated

1. Matt Boin’s Raspberry Pi Board with GPIO expansion header and sensor wires connected.

##### Acknowledgment *(Heading 5)*

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g”. Avoid the stilted expression “one of us (R. B. G.) thanks ...”. Instead, try “R. B. G. thanks...”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

##### References

1. Adafruit, “Adafruit\_BME680,” Github repository, <https://github.com/adafruit/Adafruit_BME680>
2. Bosch. "Bosch BME680 Low power gas, pressure, temperature & humidity sensor datasheet." n.d. September 2024. https://cdn-shop.adafruit.com/product-files/3660/BME680.pdf.
3. Broadcom, “Spring Initializer,” 2005-2024. [Online] Available: <https://start.spring.io/>
4. Broadcom, “Spring Quickstart Guide,” [Online] Available: <https://spring.io/quickstart>
5. Code With Arjun, "Spring Boot Using VSCode," Youtube, youtube video, Aug 7, 2021, [Online] Available: <https://www.youtube.com/watch?v=dq1z9t03mXI>
6. Maven, [Online] Available: https://maven.apache.org/download.cgi
7. Paraschiv, Eugen, "Spring RequestMapping," Baeldung, May 11 2024, [Online] Available: https://www.baeldung.com/spring-requestmapping
8. Pi4J, “Pi4J,” [Online] Available: <https://www.pi4j.com/>
9. Raspberry Pi, “Raspberry Pi 3 Model B+,” [Online] Available: <https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/>
10. Raspberry Pi OS. n.d. September 2024. <https://www.raspberrypi.com/software/>.
11. Raspberry Pi OS. n.d. September 2024. <https://www.raspberrypi.com/documentation/computers/os.html>
12. Adafruit, "Adafruit BME680 Humidity, Temperature, Barometric Pressure & VOC Gas," Adafruit Learning System. [Online]. Available: <https://learn.adafruit.com/adafruit-bme680-humidity-temperature-barometic-pressure-voc-gas/python-circuitpython>
13. ---------------------------------------------------
14. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
15. I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
16. K. Elissa, “Title of paper if known,” unpublished.
17. R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
18. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
19. M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.
20. K. Eves and J. Valasek, “Adaptive control for singularly perturbed systems examples,” Code Ocean, Aug. 2023. [Online]. Available: <https://codeocean.com/capsule/4989235/tree>
21. D. P. Kingma and M. Welling, “Auto-encoding variational Bayes,” 2013, arXiv:1312.6114. [Online]. Available: <https://arxiv.org/abs/1312.6114>
22. S. Liu, “Wi-Fi Energy Detection Testbed (12MTC),” 2023, gitHub repository. [Online]. Available: https://github.com/liustone99/Wi-Fi-Energy-Detection-Testbed-12MTC
23. “Treatment episode data set: discharges (TEDS-D): concatenated, 2006 to 2009.” U.S. Department of Health and Human Services, Substance Abuse and Mental Health Services Administration, Office of Applied Studies, August, 2013, DOI:10.3886/ICPSR30122.v2

**IEEE conference templates contain guidance text for composing and formatting conference papers. Please ensure that all template text is removed from your conference paper prior to submission to the conference. Failure to remove template text from your paper may result in your paper not being published.**