Trent Rivers (trri1737)
CSPB 3112: Professional Development
4 December 2025

Internet of Things Data Collector
Final Report

Introduction

The vision that guided the development of the Internet of Things Data Collector can best be summarized in a user story: "As an owner of a device that measures temperature, humidity, and pressure data, I want a system to receive and store that data, with a dashboard to visualize the data, so that I can make informed decisions about how I should mitigate these conditions." From that user story, we can define a few components that ought to exist: a REST API to serve as the intermediary for data, a database to hold all the data, and a frontend dashboard to show the user the data. In the absence of a real IoT device, a client application can be used to simulate sensor readings and interact with the broader system.

The description of the project so far only resembles a standard full-stack web application. To bring the project in-line with more modern software technology stacks and implementations, a number of other technologies are to be employed. These include containerization of key components using Docker, cloud integration with Amazon Web Services, and Continuous Integration / Continuous Deployment (CI/CD) using Jenkins.

This project was conceived to accomplish two goals. The first was acquiring an increased breadth and depth of knowledge about programming pipelines and technology stacks. Most coursework undertaken in the Computer Science Post-Baccalaureate program has focused on programming specifically, leaving me with little exposure to concepts like DevOps, the nuances of Frontend and Backend development, cloud computing, among others. This project provides me an opportunity to explore and develop skills in these areas.

The second goal of this project was to gain exposure to "Internet of Things" applications. IoT devices are an area of particular interest of mine because I think smarter, interconnected devices can lead to better lives and more efficient businesses. While a device was not used here, developing the infrastructure to receive data from one would serve me well in the future.

Background

In September, just before this project began, I attended an engineering career expo. Many conversations at that event exposed areas of software engineering where recruiters are looking for experience, but my current skillset is lacking. That motivated the design of the project described above. This project is about skills alignment, upskilling myself, and better preparing myself to start a job as a software engineer.

Software engineering is a very skills-based profession, and I know that I will always be learning to stay current on developments within the field. Fundamentally, that's what this project is about: staying current with market trends. As I transition out of the Post-Bacc program, I want my skills to match the needs of my prospective employers. Working on this project allows me to gain experience and learn about the aforementioned tools in a structured way, where I'm accountable for my contributions and my progress.

Methodology, Materials and Methods

To gain the necessary knowledge and skills for this project, a small set of resources were consistently used across the project's lifespan. The first (and most used) resource was structured courses from platforms like LinkedIn Learning. LinkedIn is ubiquitous, and it's a platform that nearly all working professionals are familiar. As an organization, LinkedIn advocates for a "skills-based" approach to hiring, and LinkedIn Learning exists to foster the development of the professionals on their platform. Over the course of this project, a variety of courses were used, including: Learning REST APIs, Learning Docker, HTTP Essential Training, PostgreSQL Essential Training, and Next.js: Creating and Hosting a Full-Stack Site. These courses varied in their usefulness due to the high-level, conceptual nature of some of the material. This was particularly true for Learning REST APIs and HTTP Essential Training. Others, such as Learning Docker and PostgreSQL Essential Training proved to be quite helpful because they dealt with specific software (Docker and Postgres) as opposed to protocols (HTTP and REST).

I did have to venture beyond LinkedIn Learning for one of the structured courses I used. Creating the C++ Client application required learning how to submit HTTP requests and parse JSON in C++, two tasks I was entirely unfamiliar with. To learn how to accomplish them, I used the Basics of HTTP Requests with C++ course from CodeSignal. That was extremely helpful because I would not have known what libraries to use had it not been for this course. Additionally, this sort of course was more specific and focused in scope than a typical LinkedIn Learning course, giving me material to implement almost immediately.

The second most frequently used educational resource was official tutorials and documentation from software maintainers. These included the React Foundations and App Router tutorials from Next.js and React's own Tic-Tac-Toe introductory tutorial. These were useful because they were exclusively focused on walking you through the code and explaining key concepts behind each framework along the way.

The final resource that was used to acquire the necessary knowledge and skills for this project were one-off YouTube tutorials from channels like Fireship. Despite their length, short videos tutorials were helpful in solidifying my understanding of Docker and API programming in JavaScript. The to-the-point nature of these videos and their inclusion of short, usable code snippets helped move the project forward where LinkedIn courses may have been too high-level to implement much.

Results

I would say I learned a great deal during this project. I learned about the standard HTTP web protocol (including sending and receiving requests), the stipulations of REST styling, how to make API requests over HTTP using C++, how to set up a development environment for C++ programming, how to create and query Postgres relational databases, and how to create web applications using frameworks like React and Next.js. That's an admirable assortment of topics, but, admittedly, it does fall short of those laid out in the project vision.

I can say that I learned these through my own ability to describe them now (ask me a question about HTTP requests, I'll answer), and through the implementation in the software.

When this project was first proposed, a few different assessment criteria were outlined. These included evaluating code quality, structure, and functionality, as well as if the components of the application met all functionality requirements. Disappointingly, I can say that, according to the criteria I myself laid out, this project did not meet the assessment criteria. Code quality was

an afterthought in favor of getting something minimal working. Functionality was sacrificed in favor of reducing scope.

Discussion & Reflection

I can say from the outset that my goals for this project were not achieved. I think a couple of factors contributed to this. The first is that attempting to use a single project to familiarize oneself with an entire panoply of technologies and languages, across every level of a modern tech stack is overwhelming. Some people, when attempting to learn a new language, attempt something they can understand, like a Markdown to HTML converter script. The vision for this project was overambitious, and a single project alone is not enough to become competent in every area of software development.

The second factor that hindered this project was the influence of artificial intelligence in the project planning process. The idea for this project was not my first; I had pivoted away from a different idea when I concluded the skills I would acquire would be too specific to help me much in my professional development. Having discarded my first idea, I turned to prompting for an alternative that would expose me to the skills I desired. This ultimately proved to be a poor decision because LLMs have never *learned* anything in the human sense, so estimates for the time necessary to learn each new skill proved wholly inaccurate. My motivation for this project *was* to learn as many new skills as possible, but AI can not tell you how long it will take you to learn what you need.

Another factor that, while not a major hinderance certainly affected the project, was maintaining the motivation to continue investing time into the project beyond what was required. That proved to be a challenge because of the vision of the project itself. The goal of this project was to align my skills with market demands, and while it is, prudent, pragmatic, and undeniably beneficial to me as a professional, it was not something I was excited to work on or learn more about. One thing that I didn't heed enough in my conversations with recruiters was the value of "projects that show that you're curious". This project did not pique my curiosity, and I think, when given the freedom to define my own objectives like in this course, I ought to have selected a project that motivated me and pulled me forward in curiosity rather than dragged me down in obligation.

I think the project results could be better (and they will be better in the future with some more work), and there are some ways I would change my approach to projects in the future. Firstly, I would limit the scope of my project to only a couple of areas and disciplines where I want to grow in order to better situate myself in the zone of proximal development. Throwing myself beyond the bounds of my ability and skills hindered me because I did not have the skills necessary to function independently, away from tutorials, in this area. During the project planning and ideation phase, I will avoid AI because project estimates (especially around time to learn) were dramatically inaccurate. I will also attempt to find a project vision that both makes me curious to learn more but also demonstrates prudent decision making about my professional development. Future projects will doubtlessly be improved through these process changes.

Conclusion

Considering my contributions and incremental updates across the last couple of months, I think my outlook has changed insofar as I acknowledge the need for limitations in scope and the necessity of curiosity in project planning. I also have realized a couple of things about how I

learn new skills. I value depth over expedience, so the most effective method for me to learn is to invest the time necessary to fully understand material, instead of rushing to implementation.

I still intend to continue building the project, to achieve the vision as described above. As I previously mentioned, the motivation behind this project was prudent, and the demand for these skills has not gone away. Once I've delivered on the initial vision, then I can move on to another project.

There are a few subjects I would like to investigate in futures projects. For example, I'm interested in how software is deployed to different devices, and while I have experience developing some desktop and web applications, I have yet to make something for a mobile device. A future project could be something similar like making a calculator or shopping list app for mobile devices. These are by no means innovative, but the goal would be to learn without going overboard with a project I do not fully understand the scope of. My initial idea for this project involved deep reinforcement learning and AI agents, which I was going to learn about using HuggingFace courses. I still think reinforcement learning is the part of AI that far too few people talk about, and it is a topic that excites me and makes me curious.

References & Resources

Learning REST APIs: https://www.linkedin.com/learning/learning-rest-apis/

HTTP Essential Training: https://www.linkedin.com/learning/http-essential-training/

Learning Docker: https://www.linkedin.com/learning/learning-docker-17236240/

PostgreSQL Essential Training: https://www.linkedin.com/learning/postgresql-essential-training-22611610/

Next.js: Creating and Hosting a Full-Stack Site: https://www.linkedin.com/learning/next-js-creating-and-hosting-a-full-stack-site/

Next.js tutorial – React Foundations: https://nextjs.org/learn/react-foundations

Next.js tutorial – App Router: https://nextjs.org/learn/dashboard-app

CodeSignal Basics of HTTP Requests in C++ Course: https://codesignal.com/learn/courses/basics-of-http-requests-with-cpp

httplib – a header-only utility for requests in C++: https://github.com/yhirose/cpp-httplib

nlohmann/json – JSON for modern C++: https://github.com/nlohmann/json