EMBRACING THE POWER OF THE WEB: TRANSFORMING A LOCAL BUSINESS IN THE DIGITAL AGE

A Research Paper submitted to the Department of Computer Science
In Partial Fulfillment of the Requirements for the Degree
Bachelor of Science in Computer Science

By

William Helmrath

November 23, 2021

ADVISOR
Daniel Graham, Department of Computer Science

EMBRACING THE POWER OF THE WEB: TRANSFORMING A LOCAL BUSINESS IN THE DIGITAL AGE

CS4991 Capstone Report, 2021

William Helmrath
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
wph2tu@virginia.edu

ABSTRACT

Forge, a Charlottesville-based non-profit that provides professional development services to students, needed a system for automating many of the menial tasks that team members of Launch, its internship matching program, have to perform daily. To solve this problem, I built out and hosted a web application to automatically construct student resumes from information submitted through an interactive form. This application leveraged the power of React, a front-end JavaScript library, in order to create fast and responsive dynamic webpages. Since then, the application has evolved into a marketing tool that also helps to generate revenue. During the 2020-2021 school year, effective use of the web portal resulted in increased revenue and client satisfaction for Forge and the Launch program. As the Launch team currently prepares for the summer of 2022, my team and I continue to maintain and update the project.

1 INTRODUCTION

In the summer of 2020 I interned for a Charlottesville-based non-profit, Forge, as a software engineer. The company, which runs an internship matching program of their own called Launch, was in the middle of a rapid expansion, looking to double the number of interns enrolled in the program. With this expansion came several issues that are common with increasing the scale of a program. Many of the tasks typically performed by hand, such as creating Forge-branded intern resumes and reviewing these resumes, were becoming increasingly time-consuming, taking the team away from performing their more crucial duties associated with outreach and finding company partners.

As an intern, I was tasked with designing a web app to streamline these manual processes. In the end, I developed a web portal with three main purposes:

- To allow students the ability to create their own personalized Forge-branded resumes
- To display student resumes to company partners as marketing material, and
- To facilitate the company matching process through an interactive dashboard

This paper will focus on the first purpose: building an interactive resume editor.

There are three main components to a web app: the front-end UI, the backend logic, and the database. I chose to develop the front-end with React, a JavaScript web library designed to help build dynamic webpages and forms. The backend, also written in JavaScript, was deployed using Firebase Cloud Functions.

The database proved to be the most difficult component to work with, as I was constrained to using Airtable – a user-friendly and web-accessible method of storing information – as it was already being used by the Launch team to manage their data. The project, which was completed by the end of the summer, is something that I am still actively maintaining as the needs of the Launch team evolve.

2 RELATED WORK

In order to complete the project as quickly as possible, I took advantage of many resources. These mainly came in the form of online documentation for various libraries and some web tutorials.

The most helpful of these resources was the documentation for React itself [1]. Not only does it provide a step-by-step guide for building a web app with React, but it also contains detailed descriptions and use cases for its tooling. I always made sure to reference this documentation when faced with problems that I could not solve given the knowledge I already had at my disposal.

The way in which I consulted and used the React documentation was fairly consistent with my use of other library documentation, such as those of React Router and Airtable [2, 3].

3 PROJECT DESIGN

The requirements for the project were almost entirely defined by the desired functions of the project, as the Launch team is comprised of mostly non-technical people. From these soft requirements, I was able to work backwards in order to determine and define the technical specifications.

3.1 Front-end UI

The desire for a program that could be accessed by anyone from anywhere naturally lent itself to a webpage, due to the ubiquity of web browsers. The resume builder and the need for an interactive form further narrowed the project down into a “dynamic” or interactive web app, for which there are many different programming libraries and frameworks publicly available.

The decision to use React over other commonly used front-end web libraries was simple: it was the only library I was familiar with. I had previously taken a class on React through Forge, and had gotten additional training on it at the beginning of the summer through their internship program.

3.2 Database

While users may only interact with a webpage while browsing the internet, there are actually several other aspects to traditional web apps that facilitate the handling of data and web requests. This is referred to as the “back-end” of a web app (as opposed to the “front-end,” or the end user interface). Specifically in terms of managing and saving user data, the Launch team had a hard requirement of using an Airtable database. Airtable was already being used by the Launch team to manage much of the data that they had previously been compiling on their interns, and they had grown accustomed to using it as opposed to more traditional databases due to its user-friendly nature. Unlike other databases that typically require the use of a coding language called SQL to interact with data, Airtable provides an online interface (a web app of their own) that customers can use to enter and edit data. This made it accessible to non-technical members of the Launch team.

3.3 Back-end Server

It is possible for this system to work with just a front-end and a database, however I decided against this system design because of two main concerns: responsiveness and security.

Responsiveness is a measure of how quickly a web page responds to user interaction, such as clicking on a button or typing in information. The React library prioritizes responsive by design, however it can get slowed down when running time-intensive operations. In order to reduce this delay in responsiveness as much as possible, I decided to additionally implement a dedicated back-end. This structure allowed me to offload many complicated algorithms onto a server as opposed to running them directly on the users' computers. For instance, Airtable requires data to be submitted in very specific format that differs from the format that the data is managed inside of React. Instead of performing this data conversion on the front-end, I send the data (formatted in the React way) directly to the back-end, which then manages the conversion to the Airtable format.

Security is the other main concern that kept me from allowing the front-end to interact directly with the database. In order to gain access to Airtable, they provide users with a unique "key" which they need to attach to every web request sent to the Airtable application programming interface (API). This key is meant to be kept secret, as if you allow someone to view your key they can then use it to impersonate you and modify your database without consent. If I were to allow the front-end to use this key when sending data to the Airtable API, I would need to expose Forge's secret key to potentially malicious users. Therefore, every web request sent from the front-end was designed to go through the back-end, obfuscating the Airtable key and ensuring its security.

4 RESULTS & OUTCOMES

4.1 Resume Builder

This system, dubbed the "Launch Resumes Portal", was used by the Launch team in the Summer 2021 and is currently being used as the team onboards new students for Summer 2022. During training, students are introduced to the web app and given instructions on how to log in and modify their information.

Utilization of the Launch Resumes Portal reduced the time spent micro-managing student resumes by a significant margin for the Launch team. This is mainly due to the system giving agency over a student's resume directly to the students themselves, as opposed to the Launch team. With students able to edit their information at-will and able to read team feedback from within the app, the number of emails manually sent between Launch team members and students fell to near zero. Instead, emails are automatically sent after a team member indicates whether or not a resume has been approved for publication to Launch company partners.

4.2 Additional features

While the major details are beyond the scope of this paper, I think it is important to also mention the end results of the two additional goals mentioned in the Introduction.

Initially, the Launch team limited the scope of their marketing to mass emails sent out to potential company partners. However, there was no easy way for Forge to generate company interest in actual candidates until they had responded to an email or two. The candidates portal which I developed, however, flipped this dynamic. By displaying live candidate resumes on an easily accessible website, the Launch team could direct potential partners ~~directly~~ to a link, allowing them to gain a better understanding of the interns they could potentially be hiring. This lead to a notable increase in the reply rate of emails that the Launch team sent out during the marketing in their Fall 2020 and Spring 2021 campaigns.

The third and final additional feature of the web app was a "matching portal," a webpage for current company partners to view the resumes of potential candidates and indicate which ones they would like to interview. Previously, this had been an entirely manual process, facilitated through countless emails between Forge and company contacts. The portal, however, reduced the email load per company to only one or two. This resulted in a marked increase in satisfaction from all parties: the Launch team, company partners, *and* students. The portal, in addition to simplifying the interview matching process, also sent out automatic emails and text messages using SendGrid and Twilio, two messaging APIs. The emails connected students with their interviewers immediately after decisions were made, and the texts informed the students to check their inboxes in order to ensure a timely response to the emails.

5 FUTURE WORK

Throughout the 2021-2022 academic year, I took on an active maintenance role with the project, solving problems and patching bugs as users came across them. Currently, there are no known issues with the software that have not already been addressed. However, there are many improvements I would like to make to the code in order to make future development easier.

Currently, the most pressing of these matters is migrating the project from JavaScript to TypeScript. JavaScript, the project's initial coding language, is "weakly typed." This means that any variable can be interpreted as, or "coerced into" any type. For example, the string "1," when added to the number 1 using the plus operator, returns a value of the string "11" ("1" + 1 evaluates to "11"). Because of this weak typing, it is very easy for bugs to crop up without realizing it, as code editors like Visual Studio Code will allow your code to "just work" without checking if the operations you are attempting to perform are even valid. This is where TypeScript comes in.

In essence, TypeScript is JavaScript with strictly enforced typing. If you attempt to evaluate "1" + 1, your IDE will display an error, informing you that you are attempting to evaluate different types. While it may seem minor, this can save a *lot* of time in the development process. Problems that may have taken several minutes of debugging will now be displayed almost immediately. In addition to saving time when writing *future* code, converting current JavaScript files to TypeScript will allow me to retroactively identify errors that I may have never noticed.

It is my hope that with these changes in place, the project will be used by the Launch team for years to come. With my graduation this upcoming Spring, it is crucial that the project is in a manageable enough state to be taken over by other students in future years.

REFERENCES

[1] React. 2021. Getting Started. Retrieved from https://reactjs.org/docs/getting-started.html.

[2] React Router. 2021. Documentation. Retrieved from https://reactrouter.com/docs/en/v6.

[3] Airtable. 2021. REST API. Retrieved from https://airtable.com/api.