DeveloperTown House Locator: A Touchscreen Web Application

An Honors Thesis (CS 498)

By

Ryan Wiesjahn

Thesis Advisor
Dr. Lan Lin

Signed

Ball State University
Muncie, Indiana

April 2015

Expected Date of Graduation
May 2015
Abstract

College classes offer a great opportunity for learning core concepts in a particular field of study. But classes alone are not enough to fully prepare students for a career after college. In the field of Computer Science, having hands-on experience working with a client to produce production software is invaluable experience that will be a major benefit when finding a job after college. This thesis focuses on my experience with my senior capstone project for Computer Science, detailing the full process from finding a client to developing and polishing functional production software. I explain the reasoning behind certain decisions that were made as well as my final takeaway of the experience as a whole.

Acknowledgements

I would like to thank Dr. Lan Lin for instructing the capstone courses and advising me throughout the project. I would also like to thank Michael Kelly for offering my team and I a great capstone project opportunity and giving us a lot of creative freedom on the project. Last but not least, I would like to thank Derek Onay, Luke Martin, and Zach Town for being a great team.
Author's Statement

During my final year at Ball State University, I lead a group of Computer Science students in developing a large touchscreen software application for a company in Indianapolis, Indiana called DeveloperTown. All Computer Science students are required to take two consecutive semesters of a capstone course in their final year of study in which they work in a small group with a professional client to develop production software that the client will regularly use. Groups work together through the full development cycle, from an initial feasibility study to delivering a final and functional product to the client. Each semester consists of two milestones. After each milestone, a progress report and presentation are put together for both the client and the class.

Milestone 1

Finding a Client

The first step for groups in the capstone class is to find a client to work with throughout two semesters. I had multiple leads for potential clients, and after bringing these leads to my group and discussing with them the pros and cons of each client, we settled on working with DeveloperTown in Indianapolis, Indiana. DeveloperTown is a software and media production agency located in a large, warehouse-like building in which employees work in movable offices constructed to look like small houses. These houses are arranged in such a way to give the
feeling of walking through a small town. As more houses are added for new employees, houses move around, or community events are held, the locations of employee offices change. There are many visitors to DeveloperTown, be it clients, interns, house tenants, or guests. With employees’ offices constantly moving around, it is hard for these visitors to find the employee with whom they are visiting.

Requirement Analysis

My group worked with one of the four partners of DeveloperTown throughout the development process. He expressed the need for a large touchscreen application for visitors to use to locate DeveloperTown employees. The touchscreen computer would be stationed at the entrance of the building for visitors to use once they walked in. After multiple discussions with our client, we agreed on a detailed list of requirements.

Our group would develop a main application for public use in which users would be presented with an interactive floor plan of the current arrangement of houses. When a house was touched, the application would show information about the employee such as a name, photo, phone number, and general office hours. They would also be able to touch a button that would automatically send the employee a text message informing the employee they had someone waiting for them at the entrance. Users would be able to search for employees via a search box and see conference room schedules. An admin interface would also be developed to
allow DeveloperTown employees to adjust the floor plan as needed, update employee information, and manage other aspects of the public application. A mobile phone view for clients currently on their way to the office, a backend system to serve the application’s data, and a database to store all of the data were also required.

End of Milestone 1

At the end of Milestone 1, my group presented our client with a Feasibility Study and Plan which detailed my team’s analysis of the project requirements, the final management and technical deliverables, our process to be followed, our project plan and projected milestones (which were different from the class milestones), business considerations, a risk analysis, and the technical feasibility of the project. One group member and I also gave a presentation to the class, outlining many of the aspects of our Feasibility Study and Plan that was delivered to our client.

Milestone 2

Choosing Technologies

After my team had a good grasp of the scope of the project, we started to brainstorm exactly what technologies we would use to create the application. I suggested creating a web application, as it would be the most versatile in deployment and end-user-use options. This is
because web applications are highly accessible from many different devices, operating systems, and web browsers. DeveloperTown would have multiple options on how they would like to deploy the app, and accessing the admin panel would be possible from any computer with Internet access. My team agreed to develop the project as a web application. Since I had the most experience in web development, we decided I should lead the team.

I decided that the frontend application should be built with the AngularJS framework. AngularJS is a framework that aids in easily creating single page web application. Using AngularJS allowed for relatively quick bootstrapping of a Model View Controller (MVC) structured application as well as provided many great features we could utilize for developing the application such as two-way data binding and a robust Application Program Interface (API) for creating resource objects that interact with Representational State Transfer (REST) server-side data sources. Fabric.js, a JavaScript and HTML5 canvas library, would be used to develop the interactive floor plan. This part of the application would consist of different graphic objects that could be selected, dragged around, resized, and more. Fabric.js provided interactive object models that made these features much easier to develop.

For the backend system, which would serve data such as current house locations, employee information, and admin user credentials to the frontend application, I decided on using the Ruby programming language and Ruby on Rails framework. I also decided to use
Prototyping

After agreeing on the technologies of the project, my team went to work on developing early prototypes of different aspects of the application. The first part that I started prototyping was the interactive floor plan, as shown in Figure 1.

Figure 1: An early prototype of the main application. The rounded rectangles could be dragged around and rotated.

The interactive graphic objects, which would soon come to represent employee houses, would need to follow multiple constraints. They should not be able to be pulled off of the canvas, and...
should not be resizable by the user, as this would be automatically handled by the application.

After creating a working prototype that could display a few objects on the canvas with predetermined positions, I wanted to create objects on the canvas with locations determined by data stored in the database as well as show a list of employees, as shown in Figure 2.

![Figure 2: A prototype of the admin panel. Employees on the side panel could be filtered from the input box. The rounded rectangles, which represented houses, could have their positions and angles saved to the database.](image)

I created an early prototype of the database schema as well as a simple backend API endpoint to serve data to my frontend canvas prototype. Once I got this working, the second milestone of the course came to an end.
End of Milestone 2

At the end of Milestone 2, a report detailing my group’s progress was created and given to the client. This Milestone 2 Report detailed the technologies we had decided on using as well as the progress of our prototyping. It also defined the current schema of the database and provided diagrams to show the relationships between tables in the database.

Milestone 3

Technology Stack Changes

After working on the backend system more, I decided that my team should switch from Ruby on Rails to Node.js. Node.js is a platform built on Google Chrome’s JavaScript runtime, which allows server-side applications to be written in JavaScript. It uses an event-driven, non-blocking input/output (I/O) model that makes it perfect for real-time applications, such as our application. I decided that Node.js would be a better fit for our project because Ruby on Rails systems comes with a lot of overhead, which was excessive for our relatively small backend system. Node.js is also quicker to install and deploy than Ruby on Rails and provides a consistent development environment between the frontend and backend as both are written in JavaScript.

After getting the client’s approval to switch technologies, to get my group and the application back up to speed, I recreated the current backend in Node.js, testing to make sure
the prototype worked with the new backend system. I then walked through setting up the Node.js development environment with my team, as it was the first time they had used the technology.

**Fleshing out Application Functionality**

Once the old backend prototype was successfully ported to a new technology, I continued building the backend system. I expanded the current database schema (as shown in Figure 3), business logic, application logic, and API to cover more of the project requirements such as storing more employee data, photo uploading for employees, and serving images to the frontend.

*Figure 3: The Entity-Relationship (ER) diagram of the database.*
I developed the frontend application functionality in tandem with the backend; I added the ability for users to view employee information, as shown in Figure 4, as well as an admin panel to modify employee information and house locations.

Figure 4: Single employee data could be viewed by clicking on their house. Starting to create some of the finalized graphics and layout.

At this point, in the application's current state, there were still a few major problems that needed to be addressed. The interactive floor plan had no sense of scale. The house graphic objects were arbitrary sizes, and were not conscious of the size of the floor plan canvas they were on. The floor plan size was also not to scale with DeveloperTown’s building size. To fix this, we obtained blueprints of the DeveloperTown building from the client. After some measuring of the blueprints, I added constraints to the canvas size to keep a consistent scale.
with the actual size of the building. I then set constraints on the house objects to do the same thing, keeping a consistent scale dependent on the size of the floor plan. These constraints guaranteed that the application could run on any size screen without the dimensions and scale of objects becoming skewed.

The next major task that was yet to be addressed was the problem of authorization and authentication. In its current state, anyone could access the admin panel and change employee information and house locations. The next step was to create an authentication and authorization strategy. For the scope of our project, authentication, the process of verifying that a user is who they say they are, and authorization, the process of verifying that an authenticated user has permission to do what they are trying to do, could be considered the same thing, as all admin users would be able to execute the same tasks. I implemented a session-keeping strategy in the backend and protected certain API functionality that should only be accessible to admin users. I then implemented handling sessions in the frontend to properly send authentication credentials to the backend when required. These additions would serve to keep the admin panel secure from malicious users.

**Application Demo**

At this point in the project, our team had enough of the main required features of the application finished that we felt we could deploy a demo of the application in its current state.
To do this, I set up a server through Amazon Web Services (AWS), a cloud computing service offered by Amazon. Through AWS, I set up a free Amazon Elastic Compute Cloud (EC2) server than ran the Ubuntu operating system. I deployed the source code to the server and ran the necessary installation commands. I then tested the demo application to make sure it was running correctly on the EC2 server. This marked the end of Milestone 3.

End of Milestone 3

A third progress report was due at the end of Milestone 3. This report went over the current progress, changes to the technology stack, the planned testing process that would be implemented, a slightly altered schedule, and instructions on how to access the demo application running on the Amazon EC2 server. It also outlined our plans for deployment at DeveloperTown at the end of project. Another group member and I gave a presentation to the class, going over some of the details of the report, explaining how some of the interesting features of the application worked, and giving a demo of the application.

Milestone 4

Adding Final Features

At this point, the application was working well, but there was a problem we had initially overlooked: once a change was made to employee information or house location by an
employee from the admin panel, this change would not be seen on the main touchscreen application until the web page it was running from was refreshed. Since the main application was meant to be kept running continuously, manually refreshing the application after every change in the admin panel would not be a very efficient solution. I solved this problem with the help of Socket.IO. Socket.IO is a JavaScript library that enables bidirectional event-based communication between web clients and servers. This means that after an initial page load, a web application can be sent data from the server, which may have been sent from another client. I set up the backend server to send a message to all connected clients any time data in the database had changed. Upon receiving one of these messages, the frontend application would then update its cached data accordingly. Now, any time one client made a change to the data, all other clients would see this change in real-time.

Another feature that was added at this time was displaying the conference room schedules in the application. To do this, the application had to pull calendar data from DeveloperTown’s Google Calendars. DeveloperTown keeps their Google Calendars private, which only allows authorized users to see calendar events. Because of this, the application had to perform a few authorization steps before being able to get the calendar data. For this, I used a Node.js library to aid in the OAuth authorization process. I developed a system on the backend server to allow the admins to authorize the application from the frontend admin panel.
to read the calendar data. After this, calendar data could be accessed from the server and sent
to the frontend application upon request.

**Polishing the Application**

Once a good majority of the application was complete, it was time to start working on
many of the visual and user experience aspects of the project. Many parts of the application
still looked like they were a prototype, so finishing touches were needed to make these
features production ready for end users. I created more permanent graphics for the floor plan
background and house objects, as well as designed an improved look for the application overall,
as shown in Figure 5.

![Figure 5: Finalized application design.](image-url)
I added nice user experience enhancing features such as alert messages that pop up whenever the user would successfully complete an action or if an error occurred. Determining the location of an employee's house on the map after selecting them from the side panel was made easier by adding a more prominent border around the house and making it bounce up and down for a moment. All of these information and visual cues were very helpful to make sure the end user was never confused when using the application.

End of Milestone 4

At the end of Milestone 4, my team prepared a final document for the client to serve as an overall reference for all aspects of the project and the application. It was intended for future developers who may expand on the application as well as project managers who would like a detailed overview of the application. The report detailed the project background for anyone who does not know anything about the project. It also covered the feasibility study, the requirement analysis and feature specifications, the system and program design, user-interface designs, our completed milestones, the technology stack that was utilized, the database schema, and more.
Deployment

The application was built on several easy-to-set-up libraries and frameworks that allow for a quick environment set-up on new servers. The source code for the application is hosted on a remote version control server, which allows for quick deployment of the application. There are extensive deployment instructions bundled with the application to allow for DeveloperTown to deploy the application without any hassle. The backend API will be hosted on an Amazon Web Services EC2 server, while the frontend will be ran as a native desktop application with the help of some libraries that package AngularJS projects as native applications.

Takeaway

Working on this project with DeveloperTown has been a great work and learning experience for me. I had previously worked with smaller businesses and entrepreneurs doing freelance web development, but this was the most professional company I have worked with to date. Working with a client through the whole development cycle, from gathering required features to developing a production ready application, is experience you cannot get in the classroom. Keeping consistent contact with a client and constantly reiterating a project schedule by monitoring progress and future milestones are not easy tasks. But this project has definitely provided me with experience in all of these aspects, which will be helpful once I start
working after college. Working closely with one of the partners at DeveloperTown also
provided me an opportunity to get my foot in the door at the company, and helped me to
obtain a job with the company as a software Developer. Considering the outcome of the project
and my job opportunity with DeveloperTown, I would say the project was a great success.
Appendix

DeveloperTown House Locator Application Demo

The following link is to the demo application hosted on an Amazon Web Services EC2 server.

http://ec2-52-0-147-130.compute-1.amazonaws.com/house-locator/frontend/#/