CS CAPSTONE PROJECT ARCHIVE

MAY 30, 2020

TRACK TOWN PIZZA WEB APPLICATION

PREPARED FOR

TRACK TOWN PIZZA

TIM MEYERS

PREPARED BY

GROUP 74
TRACK TOWN PIZZA

KAITLIN HILL
AIDEN NELSON
HANNAH VAUGHAN

Abstract

This document contains a culmination of the documentation produced by team CS 74 throughout the Capstone course. Included in this document is the Requirements Document, Design Document, Technology Reviews, weekly blog posts from the fall and winter terms, showcase poster, code listings, code review critiques and responses, and more.

CONTENTS

1 Introduction to Project 8

2 Requirements Document 9

2.1 Change Log . . . 1

2.2 GitHub Repo . . . 1

2.3 Introduction . . . 1

2.3.1 Purpose . . . 1

2.3.2 Scope . . . 1

2.3.3 Product Overview . . . 2

2.4 Specific Requirements . . . 3

2.4.1 Functions (User Stories) . . . 3

2.4.2 Usability Requirements . . . 4

2.4.3 Performance Requirements . . . 4

2.4.4 Logical Database Requirements . . . 4

2.4.5 Design Constraints . . . 4

2.4.6 Software System Attributes . . . 4

2.4.7 Supporting Information . . . 4

3 Design Document 4

3.1 Introduction . . . 1

3.2 Overview . . . 1

3.2.1 Scope . . . 1

3.2.2 Purpose . . . 1

3.2.3 Intended Audience . . . 1

3.2.4 Restrictions . . . 1

3.2.5 Stakeholders . . . 1

3.3 Design . . . 2

3.3.1 Design Component 1: Wireframe/Mockup Technologies . . . 2

3.3.2 Design Component 2: Information Architecture . . . 2

3.3.3 Design Component 3: System Architecture . . . 3

3.3.4 Design Component 4: Hosting Platforms . . . 4

3.3.5 Design Component 5: Frontend Technologies . . . 5

3.3.6 Design Component 6: User Management Technologies . . . 6

3.3.7 Design Component 7: Game . . . 7

3.3.8 Design Component 8: Backend Technologies . . . 7

3.3.9 Design Component 9: Database . . . 8

3.4 Testing . . . 8

3.5 Usability Tests . . . 8

3.6 Timeline . . . 9

3.7 Conclusion 10
3.8 Glossary 10

References 11

4 Tech Review: Kaitlin Hill 11

References 3

5 Tech Review: Aiden Nelson 3
5.1 Introduction 1
5.2 Project Role 1
5.3 Backend 1
5.3.1 Node.js 1
5.3.2 PHP 2
5.3.3 Conclusion 2
5.4 Database 2
5.4.1 MySQL 2
5.4.2 MongoDB 2
5.4.3 Conclusion 3
5.5 Game 3
5.5.1 Paper.js 3
5.5.2 Phaser.js 3

References 3

6 Tech Review: Hannah Vaughan 4
6.1 Introduction 1
6.1.1 Project Role 1
6.1.2 Project Goals 1
6.2 Piece 1: Software Architecture 1
6.2.1 Tech Option 1: Client-Server Architecture with AJAX 1
6.2.2 Tech Option 2: Single-Page Application Architecture 2
6.2.3 Tech Option 3: Node.js Web Application Architecture 2
6.3 Piece 2: Frontend Frameworks/Libraries 2
6.3.1 Tech Option 1: AngularJS 2
6.3.2 Tech Option 2: Vue.js 3
6.3.3 Tech Option 3: React 3
6.4 Piece 3: User Management Technologies 3
6.4.1 Tech Option 1: Cookie-Based Authentication 3
6.4.2 Tech Option 2: Token-Based Authentication 4
6.4.3 Tech Option 3: OAuth 2.0 4
6.5 Conclusion 4

References 4

7 Weekly Blog Posts (Fall and Winter): Kaitlin Hill 5

7.1 Fall, Week 4 . . . 5

7.1.1 Progress, Problems, Plans . . . 5

7.2 Fall, Week 5 . . . 5

7.2.1 Progress, Problems, Plans . . . 5

7.3 Fall, Week 6 . . . 5

7.3.1 Progress, Problems, Plans . . . 5

7.4 Fall, Week 7 . . . 5

7.4.1 Progress, Problems, Plans . . . 5

7.5 Fall, Week 8 . . . 5

7.5.1 Progress, Problems, Plans . . . 5

7.6 Fall, Week 9 . . . 5

7.6.1 Progress, Problems, Plans . . . 5

7.7 Winter, Week 1 . . . 6

7.7.1 Progress, Problems, Plans . . . 6

7.8 Winter, Week 2 . . . 6

7.8.1 Progress, Problems, Plans . . . 6

7.9 Winter, Week 3 . . . 6

7.9.1 Progress, Problems, Plans . . . 6

7.10 Winter, Week 5 . . . 6

7.10.1 Progress, Problems, Plans . . . 6

7.11 Winter, Week 6 . . . 6

7.11.1 Progress, Problems, Plans . . . 6

7.12 Winter, Week 7 . . . 7

7.12.1 Progress, Problems, Plans . . . 7

7.13 Winter, Week 8 . . . 7

7.13.1 Progress, Problems, Plans . . . 7

7.14 Winter, Week 9 . . . 7

7.14.1 Progress, Problems, Plans . . . 7

7.15 Winter, Week 10 . . . 7

7.15.1 Progress, Problems, Plans . . . 7

8 Weekly Blog Posts (Fall and Winter): Aiden Nelson 7

8.1 Fall, Week 3 . . . 7

8.1.1 Progress, Problems, Plans . . . 7

8.2 Fall, Week 4 . . . 7

8.2.1 Progress . . . 7

8.2.2 Problems . . . 8

8.2.3 Plans . . . 8

8.3 Fall, Week 5 8
8.3.1 Progress 8
8.3.2 Problems 8
8.3.3 Plans 8
8.4 Fall, Week 6 8
8.4.1 Progress 8
8.4.2 Problems 8
8.4.3 Plans 8
8.5 Fall, Week 7 8
8.5.1 Progress 8
8.5.2 Plans 9
8.6 Fall, Week 8 9
8.6.1 Progress 9
8.6.2 Problems 9
8.6.3 Plans 9
8.7 Fall, Week 9 9
8.7.1 Progress 9
8.7.2 Problems 9
8.7.3 Plans 9
8.8 Fall, Week 1 9
8.8.1 Progress 9
8.8.2 Problems 9
8.8.3 Plans 10
8.9 Fall, Week 2 10
8.9.1 Progress 10
8.9.2 Problems 10
8.9.3 Plans 10
8.10 Fall, Week 3 10
8.10.1 Progress 10
8.10.2 Problems 10
8.10.3 Plans 10
8.11 Fall, Week 5 10
8.11.1 Progress 10
8.11.2 Problems 11
8.11.3 Plans 11
8.12 Fall, Week 6 11
8.12.1 Progress 11
8.12.2 Problems 11
8.12.3 Plans 11
8.13 Fall, Week 7 11

8.13.1 Progress . . . 11
8.13.2 Problems . . . 11
8.13.3 Plans . . . 11
8.14 Fall, Week 8 . . . 12
8.14.1 Progress . . . 12
8.14.2 Problems . . . 12
8.14.3 Plans . . . 12
8.15 Fall, Week 9 . . . 12
8.15.1 Progress . . . 12
8.15.2 Problems . . . 12
8.15.3 Plans . . . 13
8.16 Fall, Week 10 . . . 13
8.16.1 Progress . . . 13
8.16.2 Problems . . . 13
8.16.3 Plans . . . 13

9 Weekly Blog Posts (Fall and Winter): Hannah Vaughan 13

9.1 Fall, Week 3 . . . 13
9.1.1 Progress . . . 13
9.1.2 Problems . . . 13
9.1.3 Plans . . . 13
9.1.4 Additional Notes . . . 14
9.2 Fall, Week 4 . . . 14
9.2.1 Progress . . . 14
9.2.2 Problems . . . 14
9.2.3 Plans . . . 14
9.3 Fall, Week 5 . . . 14
9.3.1 Progress . . . 14
9.3.2 Problems . . . 14
9.3.3 Plans . . . 14
9.4 Fall, Week 6 . . . 14
9.4.1 Progress . . . 14
9.4.2 Problems . . . 14
9.4.3 Plans . . . 14
9.5 Fall, Week 7 . . . 15
9.5.1 Progress . . . 15
9.5.2 Problems . . . 15
9.5.3 Plans . . . 15
9.6 Fall, Week 8 . . . 15
9.6.1 Progress . . . 15

9.6.2 Problems . . . 15
9.6.3 Plans . . . 15
9.7 Fall, Week 9 . . . 15
9.7.1 Progress . . . 15
9.7.2 Problems . . . 15
9.7.3 Plans . . . 15
9.8 Winter, Week 1 . . . 16
9.8.1 Progress . . . 16
9.8.2 Problems . . . 16
9.8.3 Plans . . . 16
9.9 Winter, Week 2 . . . 16
9.9.1 Progress . . . 16
9.9.2 Problems . . . 16
9.9.3 Plans . . . 16
9.10 Winter, Week 3 . . . 16
9.10.1 Progress . . . 16
9.10.2 Problems . . . 17
9.10.3 Plans . . . 17
9.11 Winter, Week 5 . . . 17
9.11.1 Progress . . . 17
9.11.2 Problems . . . 17
9.11.3 Plans . . . 17
9.12 Winter, Week 6 . . . 17
9.12.1 Progress . . . 17
9.12.2 Problems . . . 17
9.12.3 Plans . . . 17
9.13 Winter, Week 7 . . . 18
9.13.1 Progress . . . 18
9.13.2 Problems . . . 18
9.13.3 Plans . . . 18
9.14 Winter, Week 8 . . . 18
9.14.1 Progress . . . 18
9.14.2 Problems . . . 18
9.14.3 Plans . . . 18
9.15 Winter, Week 9 . . . 18
9.15.1 Progress . . . 18
9.15.2 Problems . . . 18
9.15.3 Plans . . . 19
9.16 Winter, Week 10 . . . 19
9.16.1 Progress . . . 19

9.16.2 Problems 19
9.16.3 Plans 19

10 Final Poster for Showcase 20

11 Project Documentation 20
11.1 Installation 20
11.1.1 Prerequisites 20
11.1.2 Clone 20
11.1.3 Setup 21
11.2 Usage 21
11.3 Additional Information 21

12 Recommended Technical Resources for Learning More 21

13 Conclusions and Reflections: Kaitlin Hill 22

14 Conclusions and Reflections: Aiden Nelson 23

15 Conclusions and Reflections: Hannah Vaughan 24

Appendices 25

Appendix A: Essential Code Listings 25
A.1 The About Page: 25
A.2 The API Call for the About Page: 26

Appendix B: Additional Artifacts 27

Appendix C: Code Review Critiques and Responses 28

Foreword: Release Notes 1.0

For future development, there are more features that the website could include. One feature the owners would like to see is an interactive map that shows their delivery area. This would allow customers to get a better sense of where they deliver and would allow for customers to be able to tell if they live in the delivery zone. This could be added onto the google map in the contact page, or could be a whole new feature.

The management system is disconnected from the website. It must be run locally. It would be more ideal to have the management system embedded into the website. There would need to be login and authentication for this to happen. Once these are added, the two could be merged and act as one entity instead of two different applications.

More features could be added to the management system. As of right now, the owners can update the prices, but not add or delete elements from the menu. They also cannot add or delete events. It would be more ideal for them to be able to do this from the management system instead of through the database.

Finally, it would be great for the owners to upload pictures through the management system instead of manually doing it through google docs. This would allow for ease of use for them and a more streamlined ability for making changes.

1 Introduction to Project

- Who requested it?
  - The owners of Track Town Pizza requested this project.
- Why was it requested?
  - It was requested because their current website is outdated and confusing for users. They wanted a fresh, new website that worked for today's technological world.
- What is its importance?
  - This project is important because many small businesses struggle with effectively using technology as promotion of their business. Most small businesses don't have the resources or knowledge to create and maintain a website. This creates a gap between them and their customers as so many potential customers look on a website before choosing to utilize a business. Track Town Pizza no longer has that problem as their current website allows them to effectively communicate with the customers with their new website.
- Who was/were your client(s)?
  - Our clients were the owners of Track Town Pizza.
- Who are the members of your team?
  - Aiden Nelson, Hannah Vaughan, and Kaitlin Hill made up the Track Town Pizza website team.
- What were their roles?
  - Our team did not assign specific roles. Each team member acted as a technical writer, web designer, developer, and communicator. This setup worked through constant communication with each other. All team members were flexible and were okay with taking on multiple roles.

- What was the role of the client(s)? (I.e., did they supervise only, or did they participate in doing development)
  - The clients only supervised. Our team communicated with the clients and the clients gave the okay. The clients did brainstorm with us possible website features and told us of different things they were hoping to see on the new website.
- How did the changes in spring term affect your deliverables?
  - The changes in spring term delayed our deliverables. We were not able to meet with our clients face to face and our clients don't do online meetings, so we had to resort to only email to communicate. This delayed us as it took longer to receive needed information for moving forward with the project. Also, with the pandemic, we were not sure how their business would be affected. If they were hit hard financially, we weren't sure if they would want to purchase a new domain and the hosting cost. As we are starting to come out of the pandemic, the clients are okay with moving forward and hosting the website. It is happening later than we hoped for though.
- How do you recommend the next team use this final documentation to pick up where you left off? (Refer readers to Appendix C—that's where you'll write up the remaining work to do).
  - If the clients want to add more features to this website, then there would be more work to do for this project. Readers can look at Appendix C where they will see the remaining work for this project. From this documentation, you can see the technologies used and what was implemented for the project.

2 Requirements Document

Oregon State University
College of Engineering

CS Capstone Requirements Document

May 30, 2020

Track Town Pizza Web Application

Prepared for

Track Town Pizza

Tim Meyers

Prepared by

Group 74

Track Town Pizza

Kaitlin Hill
Aiden Nelson
Hannah Vaughan

Abstract

Track Town Pizza is a themed pizzeria in honor of Eugene being Track Town, USA. It is a city favorite and attracts many locals and out of area people to come eat at the historic restaurant. Currently, they are missing out on a huge market due to their lack of technology. Technology plays a big part in attracting customers in our technology driven world. Currently, Track Town Pizza has a website that is hard to use and lacks up-to-date information. We want to change that for Track Town Pizza. We are going to update the business' website, so Track Town will compare to many large restaurants in the technology sector. It will also allow Track Town to engage with a larger audience and attract more customers to the business.

2.1 Change Log

Numbers next to the change are the section where the change was made.

- *2.3.3.1* - Specified web hosting platform and made changes discussing domain.
- *2.3.3.2* - Updated list of web pages.
- *2.3.3.2* - Clarified where prompts for online and phone ordering are.
- *2.3.3.2* - Clarified page that contains the tool to build orders and calculate prices.
- *2.3.3.2* - Removed browser game from list of product functions.
- *2.3.3.4* - Updated limitations
- *2.4.1* - Removed game user story.
- *2.4.4* - Revised list of information that database system stores.
- *2.4.5* - Took away information about current host, since it is not accurate anymore.
- *2.4.6* - Mention what the target browsers are.

2.2 GitHub Repo

The GitHub repo to this project can be found at the following link:

https://github.com/track-town-pizza/capstone

2.3 Introduction

This document will detail the functional requirements of the new web application being developed and why the choices were made.

2.3.1 Purpose

Track Town Pizza is a local pizzeria in the heart of Eugene and is a cultural staple among families and sports fans within the community, but the business's website is notably outdated, difficult to use, and ultimately discouraging to all users who attempt to navigate it. The existing website displays some of the information the owners deem necessary to display online (i.e. Track Town's history, contact information, location, online menu, and a link to the online ordering system), but both the owners and various users of the website find it challenging to update and/or utilize. Updating the user interface of the website by creating a new full-stack, multi-featured web application will encourage users of the website to purchase from Track Town Pizza more frequently and thus improve the business's sales. Easy-to-navigate web pages and interactive menus that people of all technical skill levels can use will entice users to purchase from the company because it will be easier to obtain the information they seek from Track Town Pizza at any time.

2.3.2 Scope

We will be starting from the beginning to create the new website. There is no current code base for us to work with because the original website creator used tools to create the current website. We will be making the new website from scratch and will use a web hosting service to display the site.

The new Track Town Pizza web application will better display the current information about the business on the Internet in a more user-friendly manner. This new web application will display the majority of the information that is displayed on the existing website as well as update the information, create a more interactive menu, have a blog of current events in the area, and have an interactive game for users to play while they are waiting for their food. The functional requirements specified in this document will be completed in order for the project to be accepted by the client.

There will also be a login feature for the owners. The owners will need to be able to login for a couple of purposes. First, they must be able to update menu prices. A user interface will be provided for them so they don't have to directly update the database with queries. They will also login to post stories in the blog section. They are located near Matthew Knight Arena, so there are many popular events that they want to promote to help bring in business. There will also be a user interface for them to post stories to their website.

2.3.3 Product Overview

All subsections in this section will describe the product previously introduced in the Purpose and Scope sections. Here, the product perspective and functions, user characteristics, and limitations will be detailed.

2.3.3.1 Product Perspective

The product will be a web application hosted on a web hosting service, specifically Heroku. It will be under a domain name that promotes Track Town Pizza and will consist entirely of a new code base. The new web application will provide Internet users with a more user-friendly experience.

2.3.3.2 Product Functions

- Display information about Track Town Pizza on the following web pages:
  - Home (for navigation to other pages)
  - Pizza Menu
  - About Track Town Pizza (including history of Track Town Pizza)
  - Order Menu
  - Merchandise
  - Blog
  - Login page (for owners)
  - Update menu feature (for owners)
  - Update blog (for owners)
- Display visuals of the physical restaurant location and the food offered by the company.
- Prompt users to begin an online order or to call the business for ordering on most, if not all, web pages.
- Provide a tool to construct a food order and calculate its price.
- Improve usability of the website so that users can find the information they are seeking faster.
- Increase Track Town Pizza's sales by promoting easy ordering to all online users.
- Create a browser game for customers to play in order to promote the popularity of the website and decrease boredom while waiting for order completion.
- Display information about Track Town Pizza on the following web pages:
  - Home (for navigation to other pages)
  - Menu: pizzas, sides, buffet, drinks, merchandise, sauces
  - Pizza Builder
  - Contact (with Google map)

  - Blog
  - About Track Town Pizza (including history of Track Town Pizza)
  - Login page (for owners)
  - Management Hub (for owners)
  - Content/Info Management Pages (for owners)

- Display visuals of the physical restaurant location and the food offered by the company.
- Prompt users to begin an online order or to call the business for ordering on the Home and Pizza Builder pages.
- Provide a tool (i.e. the Pizza Builder page) to construct a food order and calculate its price.
- Improve usability of the website so that users can find the information they are seeking faster.
- Increase Track Town Pizza's sales by promoting easy ordering to all online users.

2.3.3.3 User Characteristics

The target audience of this product includes people who enjoy Track Town Pizza's food or who are considering purchasing from them. It also includes individuals with varying technical experience that want to obtain information about Track Town Pizza via the Internet.

2.3.3.4 Limitations

The central limitations to this product are the accessibility to Super Menu's point-of-sale system that Track Town Pizza currently uses for both its online ordering system and its cash register interface. Since we cannot change Track Town's point-of-sale system (due to it being tied to their register and ticket hardware), we cannot incorporate any sale or payment processing functionality into the website. To compensate for this, we opted to build a price estimation tool for custom pizza orders that is integrated into the website.

2.4 Specific Requirements

2.4.1 Functions (User Stories)

2.4.1.1 User Interface

- As a website user, I want an easily navigable website that displays information about Track Town Pizza so that I can learn what I want to know about the company quickly.
- As a website user, I want visuals of the restaurant and its pizzas so that I can know what the business and its food looks like.

2.4.1.2 Blog

- As Track Town Pizza's owner, I want to make posts to promote the restaurant so that I can keep users up-to-date on special events and deals.

2.4.1.3 Map

- As a website user, I want to see the restaurant's location in a map view, so that I can figure out how to get there.

2.4.1.4 Menu

- As a customer, I want to view pizza information, so that I can decide what I want before placing an order.
- As a customer, I want an interactive pizza menu that calculates my order total, so that I can know the cost of my order before placing it.

- As Track Town Pizza's owner, I want to be able to edit the interactive pizza menu, so that I can provide users with consistently updated menu information.
- As Track Town Pizza's owner, I want to be the only one who can update the interactive pizza menu so that other people cannot update the menu with inaccurate information.

2.4.1.5 Merchandise

- As a customer, I want to view purchasable merchandise other than food (such as clothing) on the website so that I know it is available.

2.4.2 Usability Requirements

Track Town Pizza's website must be able to serve a customer information about the business. The website must allow for a customer that has never used the website before to intuitively get information about the company, learn about company products, and all other information the owners want potential customers to know. The website must also provide an interactive menu page. This will allow users of all capabilities of creating their own pizza or modifying existing ones to their liking efficiently and without confusion to promote purchases.

2.4.3 Performance Requirements

The website must be up 24 hours per day, 7 days a week. It must be able to support above average amounts of web traffic without slowing down. Our website must be reactive to users. It must give the user feedback based on their actions.

2.4.4 Logical Database Requirements

The database must be able to keep track of the items and their cost. It will also hold information on blog posts, company information, links, and events.

2.4.5 Design Constraints

We are given all the information that must be displayed on the website by the owners. We get to choose how to display this information, but we must display everything the clients want.

2.4.6 Software System Attributes

The website must be compatible with different web browsers. Someone using it in Chrome should have the same experience as someone using the website in Safari. The website should also be fully functional on Firefox and Microsoft Edge. The website will not be developed or configured to work on Internet Explorer. It also must be compatible on different devices. A user on mobile has different needs than someone using a laptop. The website must also be learnable and usable. Users should not struggle very much when using our website. They should be able to quickly learn the new layout and use it to get all the information they need.

2.4.7 Supporting Information

The website is for a local restaurant who currently has an out-of-date website. Ultimately, the website should exist for users to make informed decisions about where to eat. Technology is a driving force in all industries, including food. The website exists to help users learn more about Track Town Pizza and its offerings to customers and potential customers.

3 Design Document

CS Capstone Design Document

May 30, 2020

Track Town Pizza Web Application

Prepared for

Track Town Pizza

Tim Meyers

Prepared by

Group 74

Track Town Pizza

Kaitlin Hill

Aiden Nelson

Hannah Vaughan

Abstract

This document describes the choices in our technology, organization, and architecture that will be used to construct Track Town Pizza's new website. Along with descriptions of our technology choices and designs, this document will also describe the reasoning behind our decisions in choosing those technologies and designs. This document will also outline the context of the project and its purpose.

3.1 Introduction

Track Town Pizza is a pizzeria located in Eugene Oregon. It is a town favorite and attracts many locals and people passing through. The restaurant currently has a website that is hard to use and lacks up-to-date information. We want to change that for Track Town Pizza by building a new and improved website. This document will describe the project and our plan to execute it.

3.2 Overview

3.2.1 Scope

This document addresses the design plans for the Track Town Pizza web application project. The web application will consist of a database, an application server with a backend implementation, and a web server with a frontend interface. Before implementation, however, mockups for the frontend interface must be designed for all web pages.

3.2.2 Purpose

The purpose of this web application is to provide web users with a more intuitive, user-friendly experience on the Track Town Pizza website. This will allow potential Track Town Pizza customers to obtain information about the company more easily and make a more informed decision about purchasing from them. This document details the various aspects of this web application's design and how they will aid in reaching the web application's goals.

3.2.3 Intended Audience

The intended audience for this document includes the designers of the web application (a.k.a. Our Capstone group), the managers and owners of Track Town Pizza, and any future maintainers of the application.

3.2.4 Restrictions

There are two main restrictions for this project. The first is restrictions from our web hosting service. The web hosting service only allows so much power before charging us at a higher cost. We have to take this into consideration as we think about our design. We also have to follow the specifications of our web hosting service. Tasks will need to be performed a certain way to comply to their standards.

There are also design restrictions for this project. The UI design we create needs to promote the company, their product, and location. This includes using company colors, having pizza incorporated, and promoting the business in any way possible. We have been in communication with our clients about their desire for the UI and what they are intending it to include.

3.2.5 Stakeholders

The stakeholders of the Track Town Pizza web application are the owners and managers of Track Town Pizza, the future maintainer of the web application, the application's designers, web users visiting the Track Town Pizza website, and Track Town Pizza customers. Firstly, the company owners and managers have design concerns about the affordability, usability, and security of the website as well as the ease of customizing it via their private customization features (i.e. blog posting and menu updating). Secondly, the future maintainer of the web application has design concerns about the maintainability of the web application. Thirdly, the designers of the application (i.e. the Capstone group) have design concerns about the feasibility of implementing the application, the assurance of creating a quality product, and the assurance of providing the company owners with an application that addresses their concerns. Fourthly, the web users visiting the website have design concerns about the usability of the website and the ease of finding the information they seek. Finally, Track Town Pizza customers have design concerns about how easy it is to use the website and how easy it is to order from the company, whether it be via Mealage, phone, or a visit to the physical pizzeria.

3.3 Design

3.3.1 Design Component 1: Wireframe/Mockup Technologies

Before one can build a website, one must first create mock-ups and wireframes. Mock-ups and wireframes help developers understand what the site will look like and how it will function. It is an essential step in the developmental process of a website. There are many technologies that help designers and developers mock up and wireframe their websites. It can be hard to find which technology will be best for the specific project. The technologies each have their own functionalities. There were six requirements that were considered for picking a wireframe and prototyping technology. It must be easy to learn, allow for collaboration among team members, be able to make basic and complicated mock ups, allow us to do both wireframing and prototyping, let us save our mock ups as a file format that our non-technical client can understand and use, and be low cost.

A technology considered for our mock-ups was Figma. Figma is a web-based service that allows for building mock-ups and wireframes. Figma offers a variety of tutorials to help users learn their product [10]. They also allow multiple designers and developers to work on one project together to collaborate [10]. Figma offers the ability to create simple prototypes, final mock-ups, and wireframes [10]. This would allow us to use one technology throughout our whole design process instead of having to switch technologies in the middle of our design process. Figma also allows users to export their designs as png files [10]. This will allow us to easily share our designs with our client. Figma is also free for up to three projects [10]. Since Capstone is only one project, it is a great option for us.

Our group will use Figma to create our wireframes and mock-ups. First, we will have to learn how to use Figma. There are many tutorials online that will allow us to learn how to use the product. To learn Figma, we will have to watch the videos and try them out. We will need a week or two to learn Figma. It is important to learn Figma because our group needs to understand its offerings and limits. We want to create prototypes of what we actually want, not just what we are able to create from our knowledge of the product. After our group members feel comfortable using Figma, we will start creating mock-ups for our website. Mock-ups will take about two weeks to complete. The mock-ups need to clearly and professionally display our thoughts for what the website will actually look like.

3.3.2 Design Component 2: Information Architecture

User Experience is an important part of web development. A web developer does not want the user to be frustrated when using their product. Information Architecture is an important aspect of UX. Information Architecture is "a science of organizing and structuring content of the websites, web and mobile applications, and social media software [8]". Within Information Architecture, there is Organization Systems. This is how websites structure their information [8]. This is essential to do correctly as it drives how the user looks at the site and finds information they need. A good Organization Systems will effectively use hierarchical, sequential, and matrix style. Dominos website has an excellent example of Information Architecture.

When the menu button is clicked on Dominos website, the user is taken to a page that lays out matrix style all the food options available for ordering at a Dominos store [9]. This makes it easy for the user to examine and understand what their options are. Dominos also used hierarchical style by listing their food in order of most eaten to least eaten [9]. At the top is pizza and at the bottom is extra items (like their extraneous items). When pizza is clicked, Dominos takes the user through a sequential routine of picking size, crust, sauce, toppings, and sides [9]. This allows the user to hone in on one aspect at a time and understand what all their options are, so they don't miss out on anything or become frustrated with the overwhelming amount of options. When a user clicks on other food options, Dominos displays their food options in a matrix. Users get a clear picture of everything they can purchase from Domino's use of the matrix style.

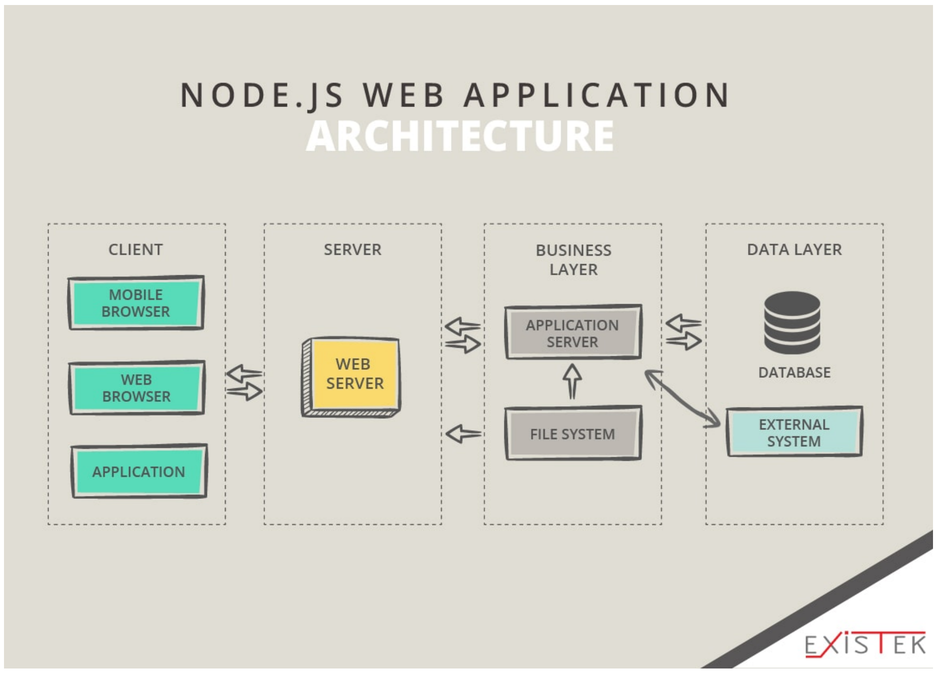
Dominos effectively uses hierarchical, matrix, and sequential Information Architecture throughout their menu page. Our group is going to decide on our Information Architecture through the creation of our prototypes and mocks. When we create our first prototype, we will white board out the necessary information for the website. This will allow us to differentiate between the information and section it out. We will then decide if the information best fits a hierarchical, matrix, sequential, or a mix of the styles. Our basic white boarded prototype will help us roughly plan how to structure the information. Our mock-ups will allow us to decide our more solid plan for Information Architecture. They will be well designed ideas that will allow us to clearly show the plan for organizing the website. Our group will have our mock ups completed by the end of fall term.

3.3.3 Design Component 3: System Architecture

System architecture refers to the design of how all components in the web application are structured and interact with each other. The system architecture functions as an incredibly high-level overview of all pieces in the application, treating each piece as a blackbox and focusing on the input and output of each component. Additionally, the architecture demonstrates how the pieces interact with each other. Designing a software or system architecture prior to implementing a project is crucial because it serves as a roadmap for all smaller pieces in the project. Each piece can then be further broken down in whatever way it needs to so that it may ultimately play the role it is supposed to in the architecture.

As described by Mohamed Aladdin of codeburst.io, "software architecture is the process of converting software characteristics such as flexibility, scalability, feasibility, reusability, and security into a structured solution that meets the technical and the business expectations" [1]. Using this definition of software architecture, the desirable characteristics of a software architecture for this project must be established. Characteristics that are necessary for this project's software architecture include all the characteristics listed in the definition: "flexibility, scalability, feasibility, reusability, and security" [1]. The architecture must be feasible so that this group may reasonably complete a working web application within the allotted amount of time that is the senior software engineering project. Next, it must be secure because the website may contain sensitive data (e.g. username and password credentials of the Track Town Pizza owners). Finally, the architecture must be flexible, scalable, and reusable so that all pieces in the architecture have their own functions, are either loosely coupled or (ideally) decoupled, and code is not unnecessarily reused. These characteristics are all important to this project's software architecture because it allows for code to be much more maintainable in the long run, for functionality to be added and removed without producing glaring errors in other areas of the architecture, and for the solution to expand if the Track Town Pizza owners ever need it to do so.

With the desirable characteristics of software architecture outlined above, the architecture that is used for the Track Town Pizza web application is the Node.js web application architecture, chosen for its ability to provide "code sharing and reusability... [as well as] flexibility and reliability" [2]:



The Node.js web application architecture consists of four layers: the data layer, the business layer, the server, and the client. In the data layer lies the database as well as any external systems that may be used for the project. As this project currently stands, only the database will exist in the data layer. However, this layer allows for the introduction of future external systems if the Track Town Pizza owners wish to add any. The application server and a file system reside in the business layer, consolidating all business logic into this layer. No file system is currently being incorporated into the Track Town Pizza web application, but similar to external systems in the data layer, a file system may be introduced into the business layer in the future. The application server performs all business logic, retrieves data from and sends data to the data layer, and manages the logic behind all requests sent from the server layer. The server layer solely consists of the web server, which has a primary purpose of managing all requests sent from the client layer and all responses sent from the business layer for the client layer. Finally, in the client layer is the client application, web browser, and/or mobile browser. In other words, the client layer is whatever platform the frontend of the web application is on.

Both the application server and the web server use the same backend technologies save for one major difference: any technologies used to handle HTTP requests and responses on the web server is not necessary on the application server. The purpose of having two servers in the application instead of just one is to abstract the business logic away from HTTP request handling. When an entity in the client layer (i.e. the mobile browser, web browser, or even client application) sends a request to the web server, the web server will determine what the request is, what next steps need to be taken to complete the request, then send it to the appropriate endpoint in the application server. From there, the application server processes the request via business logic, updates and/or retrieves the data with REST as necessary, then sends a response back to the web server. Finally, the web server takes the response from the application server and sends it back to the client entity using the HTTP protocol. This whole process may be long, but it allows for all components in the architecture to be loosely coupled if not completely decoupled.

3.3.4 Design Component 4: Hosting Platforms

Any website that users can actually access through the internet needs to be hosted on a server. Some companies have their own servers, but many rely on the cloud. The cloud is a server, but they are owned and run by a company. This takes some pressure off the developer. It allows them to focus on creating beautiful websites, not handling hardware. There are many hosting platforms for developers to use. It is beneficial to use a hosting platform because they handle all the server maintenance. A good hosting platform must be affordable, reliable, secure, have helpful customer service, and allow the company to use their current domain name.

Heroku is another common platform used to host web applications. Heroku uses containers, called dynos, for charging their customers [7]. The more the dyno is used, the more the user is charged. Heroku is more expensive than other services, but their service is unmatched. They protect against DoS attacks among other security hazards [7]. Heroku is both secure and reliable as they are used by many companies such as Dubsmash and Moneytree [7]. Heroku has a support tab at the top of their site to help their customers with any issues they may have [7]. For paying customers, Heroku allows unique domain names [7]. Heroku also has a free option which is perfect for development. We do not want to pay or have our client pay for a service before we are ready. With Heroku, one is able to do development on the free option and then switch to a paid option when ready. Heroku also has integrated development with GitHub [7]. This is useful because any changes pushed to the master branch will automatically be shown on the website. This also allows developers to follow good software engineering practices, such as code reviews, before committing code to the master branch. Heroku is an excellent choice for web hosting and is easy for developers to use.

Heroku has excellent documentation for us to follow when we are ready to host our website. They take developers through the basic steps of how to use their services. This will be beneficial when we are ready to switch to their services. Our group has decided that over winter break each one of us will deploy a small project to Heroku, so we have the opportunity to understand it before starting our project. Having a basic understanding of how a hosting service works is beneficial for winter term when we will want to spend more time understanding our project, not Heroku.

For our project, we will start using Heroku from the beginning. They have a free option, so we will fall under that option as long as possible. We want to do our development on their machines, so we do not have issues later switching from a local project to a project on their server. The GitHub integration is perfect for us because we can continue working on our project through their servers. We will switch our project over from a free option to a priced option in the Spring. For the best website, our client will have to pay for the service, but not while we are developing.

3.3.5 Design Component 5: Frontend Technologies

The design viewpoint that will be addressed in this section is frontend technologies. Frontend technologies are frameworks, libraries, toolkits, etc. that determine how frontend data and requests are processed and sent to other components of the system. This design viewpoint covers the design concern of formatting and rendering the content to web users in a visually appealing, intuitive, and user-friendly manner. Some of the most popular frontend technologies used for websites today include the following: Angular, Vue, React, Bootstrap, AJAX, jQuery, and many more. Despite the various options provided, the frontend technologies that will be used in this project are Bootstrap, React, AJAX, and jQuery. These four frontend technologies and paradigms were selected for this project because they are all easy to integrate into web page views and into each other; together they provide a wide variety of functionality for the frontend, and they integrate unit tests smoothly.

React is a JavaScript library that will allow the designers to develop reusable user interface components. These components actively respond to the form factor of the client application (i.e. a web or mobile browser). According to the React website, React is mainly used "for building user interfaces. . . [that] will efficiently update and render just the right components when your data changes" [3]. Therefore, React will be a powerful frontend technology that will aid the designers in creating an interactive user interface that is easy to use. In addition, React also allows developers to build user interfaces that are declarative, component-based, and almost completely decoupled from other parts of the technology stack being used [3]. The React website also states, "React can also render on the server using Node and power mobile apps using React Native" [3]. This is important to this project because the web server is run with Node, so selecting a frontend library that can also work server-side aids in the interconnectivity between the web server and the user interface application.

Next, React relies on JavaScript functionality, thus providing the power of a full functional programming language on the frontend. This library is also notably fast, has one-way data-binding to minimize unexpected errors, and can incorporate TypeScript. Its optimization for AJAX functionality benefits in the development of the interactive pizza menu and calculator as well as the rendering of mini games that will be added to the web application. AJAX is an acronym that stands for Asynchronous JavaScript and XML and is a development technique that results in parts of a web page updating and sending/receiving data to/from the backend without the whole page reloading. This minimizes how much data is transferred in each data transaction, thus improving response times on the website.

Another frontend technology that will be used for this project alongside React is Bootstrap. Bootstrap is a toolkit for HTML, CSS, and JavaScript that will allow the designers to "[b]uild responsive, mobile-first projects on the web" [4]. Bootstrap is beneficial to this project because it keeps design consistent across pages as well as across browsers. It is also "lightweight and customizable... [and has] [r]esponsive structures and styles" [5]. However, Bootstrap will mainly be used in this project for its CSS libraries and not its JavaScript libraries. Therefore, React is used to provide the reactive JavaScript functionality. Finally, jQuery will be yet another JavaScript library used in this project. The purpose of adding jQuery onto the existing list of frontend technologies is to simplify the JavaScript implementations of DOM (Document Object Model) manipulation and AJAX calls [6].

Overall, the combination of React, Bootstrap, AJAX, and jQuery for the frontend provides the designers with all the necessary tools to develop reusable user interface components; responsive web pages that are mobile friendly; partial page reloading to improve both response time and responsiveness; and lightweight, easy-to-read code implementations for DOM manipulation and AJAX.

3.3.6 Design Component 6: User Management Technologies

User management technologies is an umbrella term that can be used to mean all libraries, frameworks, and paradigms in which an application may manage its users. In the context of this project, the only user roles that will be managed are potential customers and the owners of Track Town Pizza. The owners have access to special pages which allow them to update the interactive menu and site information and to make posts about upcoming events and specials. This means that the number of users with username and password credentials is small, but their security is imperative to the project. For the sake of this project, the main attributes that are crucial for the user management technologies is its security, ease of integration with Node.js, and statefulness.

The user management technology in the Track Town Pizza web application is OAuth 2.0 (Open Authentication). OAuth 2.0 is a third-party authentication and authorization framework and protocol that securely manages users as well as permits them access to specific resources. One benefit of this third-party security provider is that it leaves the web and application servers stateless, unlike cookie-based authentication yet similar to token-based authentication. It is important to leave the servers stateless to ensure the application remains scalable. If the servers were stateful instead of stateless, they would have to contain data on all active sessions on the website, which could lead to bottlenecking in cases of higher traffic. Therefore, OAuth 2.0's statelessness is crucial for maintaining scalability in this project. Additionally, OAuth 2.0 can easily be integrated into Node.js via libraries such as openid-client and PassportJS. The Node.js library that is used for OAuth 2.0 in this project is PassportJS. This option also allows users to login securely via other accounts, such as Google, Facebook, and Twitter, which would be beneficial to this project. If any of the Track Town Pizza owners wish to register their initial accounts with an existing web account elsewhere (for the sake of ease), OAuth 2.0 would be one of the best options to use. Additionally, this option would allow for straightforward social media integration in the future. Therefore, OAuth 2.0 will be the framework/protocol used for the security and user management technologies piece of the project.

3.3.7 Design Component 7: Game

To promote the business and encourage customer sharing of the website, we will be developing and embedding a "Track Town Pizza" themed game inside of the website. This part of our project will probably be built entirely on the front-end. Because the game will only be a small portion of our project, limiting its presence to the front end space will keep us from going overboard and spending too much time focusing on it. The game will have to be relatively simple, and have no state saved between plays. Requirements in choosing a game framework/technology are described in the list below.

1) Lightweight: The game and whichever framework/technology we choose to use for it must not have an affect on the performance of the rest of the site.
2) Learnability: Since this is a non-essential piece of our project, spending a large amount of our time learning how to use it would not be ideal. The technology/framework we choose must be straightforward and intuitive in order to ensure that the game portion of the project can be built in a reasonable amount of time.

The technologies/frameworks that our team has chosen to research as potential options for producing our website's embedded game are Paper.js and Phaser.js. Our current choice is Phaser.js, but since the game is a smaller part of the project, this technology could change with little consequence if better options are discovered.

Phaser.js is a 2d game development framework built for creating HTML5 games [14]. The framework's last stable release was in October of 2019, and it is currently quite popular. It also has quick rendering in browsers because it switches between rendering in an HTML5 canvas and WebGL renderer depending on the browser and it's given support [14].

3.3.8 Design Component 8: Backend Technologies

The backend software of a website is what serves pages and performs logic for tasks included those created out of requests sent by clients. The technology used for this part of the project will be very important because it will provide a large amount of functionality for the website, namely for anything that is a dynamic website feature. It will also be the application that performs queries upon the database, which means that database compatibility and feasibility of integration will be a major factor in deciding the technology we use as well. The most important things our team has considered when looking at technologies for the purpose of the Tracktown Pizza website backend are listed below.

1) Capability: Will the technology be able to perform the tasks we need?
2) Learnability: Will we be able to learn and use the technology in the amount of time we have to complete the project?

3) Support: Is the technology widely used and is it supported by other technologies we may need to connect to it? The technologies our team chose to research for the role of our team's backend are Node.js and PHP. Our team ended up choosing Node.js.

Node.js web framework for building server side networking applications [12]. It is written in the language of Javascript. At this moment in time it is a very popular choice to use as a backend technology in modern web development. Some of the notable companies that use Node.js are eBay, Microsoft, and Uber [12]. Node.js is known to be an efficient, lightweight, and reliable choice in backend technologies [12]. The appeal of this technology is further enhanced by the large amount of support and documentation available for it [11]. It also works well with applications that use JSON information formating [12], such as MongoDB. In addition to these advantages, Node.js is taught at Oregon State University in the web development course, meaning that all of the members of my team including myself have a basic level of familiarity with it. As far as disadvantages are concerned, Node.js is not recommended for CPU intensive tasks [2].

3.3.9 Design Component 9: Database

The database management system software is what will be used to organize, store, and query information needed by our website. This includes but is not limited to menu item information and blog post information. Since our application is not heavily data driven, support and learnability (decision factors mentioned in the "Backend" section of this paper) will be the most important factors in choosing a technology for our team. The technologies that our team has chosen to research for the role of our website's database technology are MySQL and MongoDB. Our team ended up choosing MongoDB for our implementation.

MongoDB is a NoSQL database management system, meaning that it does not follow the traditional row and column model that SQL was designed for. MongoDB defines itself to be a document based language, replacing rows and columns with JSON files [13]. This provides the advantage of data being served as JSON from origin, which minimizes data format processing in the website's backend. MongoDB is touched on briefly in Oregon State's Web Development course, meaning that our entire team is at least familiar with the concept MongoDB provides.

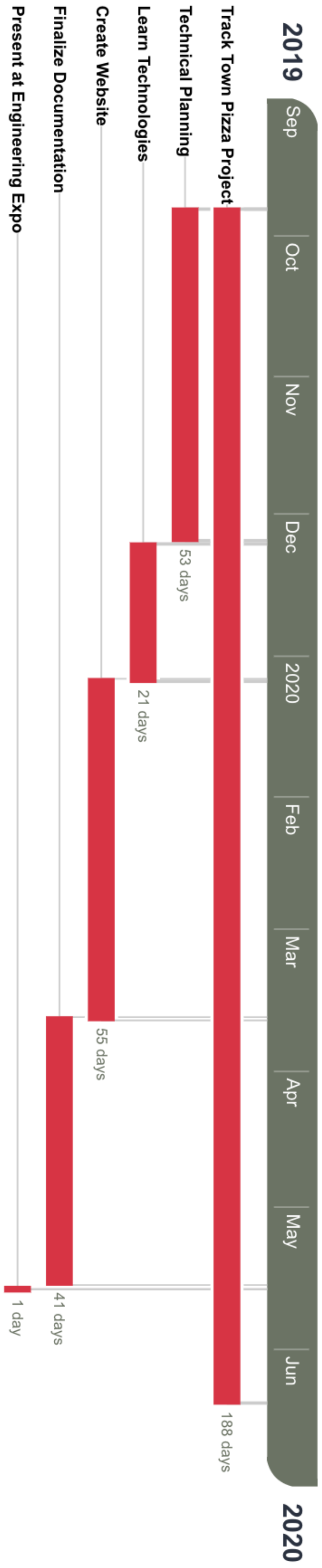
3.4 Testing

3.5 Usability Tests

It is important to run user tests on our mock-ups and prototypes. When we have our mock-ups complete, we will run our mocks past others. We want feedback from individuals to make sure our Information Architecture makes sense. We will run usability tests with people of different ages and tech abilities. We will ask them questions that relate to the layout of the website. If there is confusion about where to find something or how to maneuver through, we will know that we need to make changes to our Information Architecture.

We will also run usability tests to confirm that our design is interesting and portrays the correct information. We will ask the users to complete certain tasks. We will evaluate their performance. If the users do well, then we know our design is good. If users struggle, then we will make changes to our design based on what users had trouble performing. We will run the tests with people of different ages and tech abilities.

3.6 Timeline


2019
Sep
Oct
Nov
Dec
2020
Feb
Mar
Apr
May
Jun
2020
Track Town Pizza Project
188 days
Technical Planning
53 days
Learn Technologies
21 days
Create Website
55 days
Finalize Documentation
41 days
Present at Engineering Expo
1 day

3.7 Conclusion

Track Town pizza has needed a new website for a long time. The new website will be beneficial to all who are interested in learning more about Track Town and the restaurant itself. This website will be built using React, Node.js, and MongoDB. It will be hosted using Heroku. Mock-ups will be built so our clients can approve of the new website look before it is created. The project will also include a game for customers to play while waiting for their food. It will also be used to promote the new website. The site will be thoroughly tested. Usability tests will be run to ensure the design makes sense and is usable. Tests will also be created and run to ensure the site runs flawlessly.

3.8 Glossary

GitHub: A hosting service so multiple people can collaborate on code and run it on different machines.

Heroku: A cloud based web-hosting service.

Cloud: A remote server that someone else maintains.

Server: Hardware that manages and runs an application.

UX: User Experience. How a user feels when using a software application.

UI: User Interface. The interface that a user sees when using a software application.

Information Architecture: The way information is structured in a software application.

Mock-ups: A page that has no functionality, but it looks the same as a page in a web application.

Wireframes: A skeletal frame that show how the website's components will look.

Figma: A web application that allows users create mock-ups and wireframes.

System Architecture: A model of a system that displays its central components and how they interact with each other.

Flexibility: The capacity a system has to efficiently adapt when changes are made.

Scalability: The ability a system has to handle an increase in traffic, usage, or resources.

HTTP: An acronym for HyperText Transfer Protocol; the main protocol that is used by the Internet to handle requests for and responses to data.

Business Logic: The part of the system that implements real-world business rules.

REST: An acronym for REpresentational State Transfer; an architectural style that uses the GET, PUT, POST, and DELETE methods to provide a separation of concerns for data transfers and management.

Decoupled: An adjective that describes system components that are separate from other components in the same system.

Bootstrap: An HTML, CSS, and JavaScript toolkit for developing responsive websites.

React: A JavaScript library for developing websites with reusable components.

AJAX: An acronym for Asynchronous JavaScript And XML; a development technique that allows only portions of pages to be updated without the entire paging having to be updated.

jQuery: A JavaScript library for simplified JavaScript implementations.

REFERENCES

[1] M. Aladdin, *Software Architecture - The Difference Between Architecture and Design,* codeburst, 27-Jul-2018. [Online]. Available: https://codeburst.io/software-architecture-the-difference-between-architecture-and-design-7936abdd5830. [Accessed: 02-Nov-2019].

[2] *Web Application Architecture,* Existek, 19-Dec-2018. [Online]. Available: https://existek.com/blog/web-application-architecture/. [Accessed: 03-Nov-2019].

[3] Facebook Open Source team, *React,* [Online]. Available: https://reactjs.org. [Accessed: 02-Nov-2019].

[4] Bootstrap team, *Bootstrap,* [Online]. Available: https://getbootstrap.com. [Accessed: 02-Nov-2019].

[5] N. Patel, *Pros and Cons of Bootstrap Foundation – Know what you need!,* WebbyMonks, 25-Jul-2017. [Online]. Available: https://webbymonks.com/blog/what-are-the-pros-cons-of-foundation-and-bootstrap/. [Accessed: 11-Nov-2019].

[6] *jQuery Introduction,* w3schools. [Online]. Available: https://www.w3schools.com/jquery/jquery_intro.asp. [Accessed: 19-Nov-2019].

[7] *Heroku,* Cloud Application Platform. [Online]. Available: https://www.heroku.com/. [Accessed: 07-Nov-2019].

[8] A. Arhipova, *Information Architecture. Basics for Designers,* Tubik Studio, 17-Sep-2018. [Online]. Available: https://tubikstudio.com/information-architecture-basics-for-designers/. [Accessed: 07-Nov-2019].

[9] *Order Pizza Pasta Online for Carryout Delivery - Domino's Pizza,* Domino's Pizza, Order Online. [Online]. Available: https://www.dominos.com/en/. [Accessed: 07-Nov-2019].

[10] *the collaborative interface design tool.,* Figma. [Online]. Available: https://www.figma.com/. [Accessed: 07-Nov-2019].

[11] N. Foundation, *About,* Node.js. [Online]. Available: https://nodejs.org/en/about/. [Accessed: 09-Nov-2019].

[12] *Node.js - Introduction,* Tutorialspoint. [Online]. Available: https://www.tutorialspoint.com/nodejs/nodejs_introduction.htm. [Accessed: 09-Nov-2019].

[13] *The most popular database for modern apps,* MongoDB. [Online]. Available: https://www.mongodb.com/. [Accessed: 09-Nov-2019].

[14] *Phaser (game framework),* Wikipedia, 31-Oct-2019. [Online]. Available: https://en.wikipedia.org/wiki/Phaser_(game_framework). [Accessed: 09-Nov-2019].

4 TECH REVIEW: KAITLIN HILL

CS Capstone Tech Review

May 30, 2020

Track Town Pizza Web Application

Prepared for

Track Town Pizza

Tim Meyers

Prepared by

Group 74

Track Town Pizza

Kaitlin Hill

Abstract

Track Town Pizza is a themed pizzeria in honor of Eugene being Track Town USA. It is a town favorite and attracts many locals and out of area people to come eat at the historic restaurant. Currently, Track Town Pizza has a website that is hard to use and lacks up-to-date information. We want to change that. We are going to update the business' website. When building a new website, there are many aspects to consider. In this paper, prototyping and wireframing technology, information architecture, and web hosting services are considered. There are many options available. In this essay, three options for each section are described. The options are compared and the best option for the project is stated.

Before one can build a website, one must first create prototypes and wireframes. Prototypes and wireframes help developers understand what the site will look like and how it will function. It is an essential step in the developmental process of a website. There are many technologies that help designers and developers prototype and wireframe their websites. It can be hard to find which technology will be best for the specific project. The technologies each have their own functionalities. There were six requirements that were considered for picking a wireframe and prototyping technology. It must be easy to learn, allow for collaboration among team members, be able to make basic and complicated mock ups, allow us to do both wireframing and prototyping, let us save our mock ups as a file format that our non-technical client can understand and use, and be low cost.

There were three technologies considered: Adobe Illustrator, InVision, and Figma. Illustrator was a great option. It had many tutorials both on the Adobe website and through YouTube. It had a plethora of features that would allow us to make basic mock-ups all the way to our final prototypes [5]. It would also cover the need of being able to create both prototypes and wireframes. Illustrator also allows creators to save files as jpeg or png [5]. This is a necessary feature to show our clients our ideas for the website. There are two cons to using Illustrator. Illustrator is not built for collaboration. It can only be downloaded on one machine [5]. The mock-ups would have to be sent back and forth in some other environment for collaboration. The second con is Illustrator is not cheap. It is twenty dollars a month [5]. The service would be required for six months, so it would end up costing one hundred and twenty dollars per person. Overall, Illustrator has great potential, but some major drawbacks.

The second technology considered was InVision. InVision is a web-based service for prototyping. When a user starts a new project, there is an option to go through a tutorial [4]. This makes their product easy to learn. As a web-based service, it is built for collaboration. An unlimited number of developers and designers can be working on one prototype at the same time [4]. InVision is also completely free [4]. This makes it a great choice for college students. InVision is a more basic technology and only allows for simplistic prototypes. It does not have the technology for wireframing or creating more intricate mock-ups [4]. This would also make it more difficult to share our vision with the client. Our thoughts need to be communicated clearly with our client because they do not have many technical skills. InVision, has some excellent features, but is missing a key feature, as it does not allow us to wireframe.

The last technology considered was Figma. Figma is another web-based service that allows for building prototypes and wireframes. Figma offers a variety of tutorials to help users learn their product [10]. They also allow multiple designers and developers to work on one project together to collaborate [10]. Figma offers the ability to create simple prototypes, final mock-ups, and wireframes [10]. This would allow us to use one technology throughout our whole design process instead of having to switch technologies in the middle of our design process. Figma also allows users to export their designs as png files [10]. This will allow us to easily share our designs with our client. Figma is also free for up to three projects [10]. Since Capstone is only one project, it is a great option for us.

Each of the technologies have their pros and cons. Illustrator is popular in industry and would allow us to create beautiful mock-ups and wireframes. InVision offers a free web-based service that is built for collaboration. Figma is built for both collaboration and for the whole design process. InVision and Figma are better than Illustrator in the areas of collaboration and cost. Illustrator and Figma are better than InVision because they allow for more intricate mock-ups. Ultimately, Figma is the best option because it meets all our needs for free. It will allow us to fully plan our website before the development process which will help us save time and build a better website.

User Experience is an important part of web development. A web developer does not want the user to be frustrated by using their product. Information Architecture is an important aspect of UX. Information Architecture is "a science of organizing and structuring content of the websites, web and mobile applications, and social media software [1]." Within Information Architecture, there is Organization Systems. This is how websites structure their information [1]. This is essential to do correctly as it drives how the user looks at the site and finds information they need. A good Organization Systems will effectively use hierarchical, sequential, and matrix style. Dominos, Pizza Hut, and Mod Pizza's use of Organization Systems was examined on their menu page on their website.

When the menu button is clicked on Dominos website, the user is taken to a page that lays out matrix style all the food options available for ordering at a Dominos store [8]. This makes it easy for the user to examine and understand what their options are. Dominos also used hierarchical style by listing their food in order of most eaten to least eaten [8]. At the top is pizza and at the bottom is extra items (like their extraneous items). When pizza is clicked, Dominos takes the user through a sequential routine of picking size, crust, sauce, toppings, and sides [8]. This allows the user to hone in on one aspect at a time and understand what all their options are, so they don't miss out on anything or become frustrated with the overwhelming amount of options. When a user clicks on other food options, Dominos displays their food options in a matrix. Users get a clear picture of everything they can purchase from Domino's use of the matrix style. Dominos effectively uses hierarchical, matrix, and sequential Information Architecture throughout their menu page.

Pizza Hut uses both matrix and hierarchical style for their menu page [7]. When a user puts their mouse over the menu tab, it puts all their options in hierarchical list [7]. This is clear to the user, but it's not something that would make the user desire to press it again to find the other food options at Pizza Hut. Once the user selects from the list of food options, it takes them to the specific page for their food. Here the user sees a matrix of their specific options [7]. This clearly lays out the options for the user. Pizza Hut uses matrix and hierarchical style well to show their food options.

Mod Pizza takes an interesting approach to their Information Architecture. They aim to seamlessly combine matrix and hierarchical style. When a user clicks on the menu button, they are taken to a screen that shows them their products and a fun saying [6]. As the user scrolls, they are taken through each of the different options at Mod. It starts with pizza because that is the most popular option and ends with calorie count [6]. Before it lists out the food, it has a one-dimensional array of all their food options. If a user clicks on one of the options, it will jump them down to that section [6]. Mod pizza takes a different approach as they are more concerned with the hierarchical style.

Dominos, Pizza Hut and Mod Pizza each did an excellent job of displaying their information. Users can go through the information and decide what they would like to order. All three effectively use the matrix style of information. They all displayed their food with pictures for users to scroll through and see their options. They also used hierarchical style and took into consideration that a user is more likely to order pizza than other menu options. Dominos also effectively used the sequential style. The other two websites did not use sequential display. Dominos allowed the user to walk through building their pizza and modifying as they desire. This style increases the usability and understanding of the options that a user has. This makes Dominos style of Information Architecture to be more effective.

Any website that users can actually access through the internet needs to be hosted on a server. There are many hosting platforms for developers to use. It is beneficial to use a hosting company because they handle all server maintenance. A good hosting platform must be affordable, reliable, secure, have helpful customer service, and allow the company to use their current domain name. Three different hosting platforms were analyzed: AWS, Heroku, and Ionos. AWS is a popular hosting platform across industry. AWS is affordable as users only pay for what they use [11]. For a small restaurant, this is good as they do not have too many people visiting their site compared to other large entities. AWS is both secure and reliable [11]. Their services are used by many large companies like Nike and Netflix and are nearly a decade old [11]. AWS has customer service, but it is sort of hidden on their website. This is somewhat concerning as it gives the message that they are not concerned about customer satisfaction. A company can have their own domain name on AWS [11]. AWS has documentation on how to change the IP Address to point to the website's new location on the internet [2]. AWS is an industry standard option for hosting the client's web site.

Heroku is another common platform used to host web applications. Heroku uses containers, called dynos, for charging their customers [3]. The more the dyno is used, the more the user is charged. Heroku is both secure and reliable as they are used by many companies such as Dubsmash and Moneytree [3]. Heroku has a support tab at the top of their site to help their customers with any issues they may have [3]. For paying customers, Heroku allows unique domain names [3]. Heroku is an excellent choice for web hosting and is easily for developers to use.

Ionos is a popular web hosting service in Europe but can also be used in the US. Ionos charges based on a flat rate [9]. The customer must have a decent idea of how much their website is used, so they do not get overcharged. Ionos is both secure and reliable [9]. Ionos has great customer service [9]. When a user first gets on the site, they have a pop-up for customer service. They prioritize customer happiness with their product. Ionos also gives the option for a user to transfer their domain name to their servers [9].

AWS, Heroku, and Ionos all have the key features when choosing a hosting service. They are secure, reliable, and let customers keep their previous domain names. Heroku and Ionos have better customer service, but AWS does have customer service. AWS and Heroku charge on a usage basis, while Ionos charges a flat rate. Both AWS and Heroku are solid options, but AWS is industry standard and has proved its reliability through time and customers. AWS is the best option for web hosting.

References

[1] A. Arhipova, ”Information Architecture. Basics for Designers,” *Tubik Studio*, 17-Sep-2018. [Online]. Available: https://tubikstudio.com/information-architecture-basics-for-designers/. [Accessed: 07-Nov-2019].

[2] D. Reagan, ”Twitter application development for dummies,” *Amazon*, 2010. [Online]. Available: https://aws.amazon.com/application-hosting/benefits/. [Accessed: 07-Nov-2019].

[3] ”Heroku,” *Cloud Application Platform*. [Online]. Available: https://www.heroku.com/. [Accessed: 07-Nov-2019].

[4] ”How we use Freehand at InVision,” *Inside Design Blog*. [Online]. Available: https://www.invisionapp.com/inside-design/use-freehand-invision/. [Accessed: 07-Nov-2019].

[5] ”Illustrator workspace basics,” *Illustrator workspace basics*. [Online]. Available: https://helpx.adobe.com/illustrator/using/workspace-basics.html. [Accessed: 07-Nov-2019].

[6] ”Menu,” *Mod Pizza*. [Online]. Available: https://modpizza.com/menu/. [Accessed: 07-Nov-2019].

[7] ”Menu,” *Pizza Hut*. [Online]. Available: https://www.pizzahut.com/index.php?menu=pizza/home. [Accessed: 07-Nov-2019].

[8] ”Order Pizza Pasta Online for Carryout Delivery - Domino's Pizza,” *Domino's Pizza, Order Online*. [Online]. Available: https://www.dominos.com/en/. [Accessed: 07-Nov-2019].

[9] ”Powerful web hosting with dedicated hardware,” *Dedicated hosting — 11 IONOS*. [Online]. Available: https://www.ionos.com/hosting/dedicated-hosting. [Accessed: 07-Nov-2019].

[10] ”the collaborative interface design tool.,” *Figma*. [Online]. Available: https://www.figma.com/. [Accessed: 07-Nov-2019].

5 Tech Review: Aiden Nelson

CS Capstone Tech Review

May 30, 2020

Track Town Pizza Web Application

Prepared for

Track Town Pizza

Tim Meyers

Prepared by

Group 74

Track Town Pizza

Aiden Nelson

Abstract

The purpose of this paper is to articulate the pieces of the Tracktown Pizza website which are my responsibilities. Along with each description of a piece I will be describing various technology options which I have deemed as potentially being viable for use in the website. For each piece I will also compare and contrast the technology options I have chosen and then summarize my final thoughts on which will work best for what my team is trying to accomplish: To create a modern website for Track Town Pizza that is efficient, reliable, and has excellent usability for both customers and administrators.

5.1 Introduction

Track Town Pizza, a local business in the heart of Eugene, is looking to update their website so that users may find it easier to navigate, ideally leading to increased customer traffic and sales. The currently out-of-date website produces the risk of customers losing interest in purchasing goods from the company, especially if they struggle to find the information they need online. Customers having difficulty with the website could then lead to a decrease in traffic and eventually sales.

The issue of lacking up-to-date and frequently maintained technology is prevalent among local businesses, which makes it more challenging for them to compete against bigger companies that have already gained traction with larger customer bases. Considering how out-of-date technology can negatively impact local companies, the goal of this project is to create a new updated and modern web application for Track Town Pizza.

5.2 Project Role

The pieces of the project that I am responsible for are the website's backend, the website's database, and the game that will be embedded in the website's frontend. I will be responsible for setting up a backend to work with a database for the Tracktown Pizza website. Creating the backend for the website will require me to make the decision of which technology/framework will work best for our team. Similarly, it will also require me to choose a database management system that works well for our purposes. It will be my responsibility to design and populate the database, as well as to ensure that the backend I design can make query calls to it. Additionally, I will also be in charge of creating the Tracktown Pizza browser game, which will be embedded in the frontend of the website.

5.3 Backend

The backend software of a website is what serves pages and performs logic for tasks included those created out of requests sent by clients. The technology used for this part of the project will be very important because it will provide a large amount of functionality for the website, namely for anything that is a dynamic website feature. It will also be the application that performs queries upon the database, which means that database compatibility and feasibility of integration will be a major factor in deciding the technology we use as well. The most important things I have considered when looking at technologies for the purpose of the Tracktown Pizza website backend are listed below.

- Capability: Will the technology be able to perform the tasks we need?
- Learnability: Will we be able to learn and use the technology in the amount of time we have to complete the project?
- Support: Is the technology widely used and is it supported by other technologies we may need to connect to it?

The technologies I have chosen to research for the role of our team's backend are Node.js and PHP. In the following paragraphs I will be describing these technologies and comparing their benefits and drawbacks and how those may affect our project.

5.3.1 Node.js

Node.js web framework for building server side networking applications [2]. It is written in the language of Javascript. At this moment in time it is a very popular choice to use as a backend technology in modern web development. Some of the notable companies that use Node.js are eBay, Microsoft, and Uber [2]. Node.js is known to be an efficient, lightweight, and reliable choice in backend technologies [2]. The appeal of this technology is further enhanced by the large amount of support and documentation available for it [1]. It also works well with applications that use JSON information formating [2], such as MongoDB. In addition to these advantages, Node.js is taught at Oregon State University in the web development course, meaning that all of the members of my team including myself have a basic level of familiarity with it. As far as disadvantages are concerned, Node.js is not recommend for CPU intensive tasks [2].

5.3.2 PHP

PHP is a server-side scripting language that was built for creating dynamic web applications [3]. It is still one of the most widely used technologies for developing website software on the backend side of things. It is also great for CPU intensive applications [4]. PHP has been around for a long time (as opposed to Node.js which has come into existence much more recently) and can work well with classic relational database management systems such as MySQL [3].

5.3.3 Conclusion

Node.js will most likely be our team's choice for a backend technology for a few reasons. First is that it is a more modern framework that is likely to continue to last into the future. Second, it keeps the programming languages used in both the frontend and the backend consistent (as opposed to using multiple languages for different parts of the stack) because our frontend will most likely be written in javascript as well. Finally, our application will not be performing any CPU intensive tasks. Responsiveness and reliability are the most desired features our team will want out of a backend technology, so as of now Node.js seems to be the better candidate.

5.4 Database

The database management system software is what will be used to organize, store, and query information needed by our website. This includes but is not limited to menu item information and blog post information. Since our application is not heavily data driven, support and learnability (decision factors mentioned in the "Backend" section of this paper) will be the most important factors in choosing a technology for our team. The technologies that I have chosen to research for the role of our website's database technology are MySQL and MongoDB. In the following paragraphs I will be describing these technologies and comparing their benefits and drawbacks along with how they may have an affect in the context of our project.

5.4.1 MySQL

MySQL is a relational database management system. It responds to SQL (structured query language) which is the older, more classical database querying language. It is commonly used in the LAMP (Linux, Apache, MySQL, PHP) stack and has widespread application support. Some notable companies that use MySQL are Facebook, Google, and Adobe [5]. Additionally, MySQL and SQL are taught extensively in Oregon State University's Introduction to Databases course, which means that our entire team is familiar with using the technology.

5.4.2 MongoDB

MongoDB is a NoSQL database management system, meaning that it does not follow the traditional row and column model that SQL was designed for. MongoDB defines itself to be a document based language, replacing rows and columns with JSON files [6]. This provides the advantage of data being served as JSON from origin, which minimizes data format processing in the website's backend. MongoDB is touched on briefly in Oregon State's Web Development course, meaning that our entire team is at least familiar with the concept MongoDB provides.

5.4.3 Conclusion

Our team is currently leaning towards using Node.js as a backend technology for the Tracktown Pizza Website, which makes MongoDB have a big appeal due to its inherent usage of JSON format data storage. Our team is less familiar with it than MySQL, but since MongoDB is still widely used and popular, we are confident that adequate documentation exists for us to overcome its learning curve.

5.5 Game

To promote the business and encourage customer sharing of the website, we will be developing and embedding a "Track Town Pizza" themed game inside of the website. This part of our project will probably be built entirely on the front-end. Because the game will only be a small portion of our project, limiting its presence to the front end space will keep us from going overboard and spending too much time focusing on it. The game will have to be relatively simple, and have no state saved between plays. Requirements in choosing a game framework/technology are described in the list below.

- Lightweight: The game and whichever framework/technology we choose to use for it must not have an affect on the performance of the rest of the site.
- Learnability: Since this is a non-essential piece of our project, spending a large amount of our time learning how to use it would not be ideal. The technology/framework we choose must be straightforward and intuitive in order to ensure that the game portion of the project can be built in a reasonable amount of time.

The technologies/frameworks that I have chosen to research as potential options for producing our website's embedded game are Paper.js and Phaser.js. In the following paragraphs I will be describing these technologies and comparing their benefits and drawbacks along with how they may have an affect in the context of our project.

5.5.1 Paper.js

Paper.js is a vector graphics scripting framework for frontend web applications [7]. Although it was not created for games in mind, it provides developers an easy way to render graphics and images to the screen with movement. I have personally had experience with using this framework in the past, so the learning curve for me would not be high. The technology has been around for over ten years, meaning that there is a possibility that it is obsolete for modern purposes, but it also means there is more documentation and forum discussion regarding it.

5.5.2 Phaser.js

Phaser.js is a 2d game development framework built for creating HTML5 games [8]. The framework's last stable release was in October of 2019, and it is currently quite popular. It also has quick rendering in browsers because it switches between rendering in an HTML5 canvas and WebGL renderer depending on the browser and it's given support [8].

References

[1] N. Foundation, "About," *Node.js*. [Online]. Available: https://nodejs.org/en/about/. [Accessed: 09-Nov-2019].

[2] "Node.js - Introduction," *Tutorialspoint*. [Online]. Available: https://www.tutorialspoint.com/nodejs/nodejs_introduction.htm. [Accessed: 09-Nov-2019].

[3] "What is PHP?," *php*. [Online]. Available: https://www.php.net/manual/en/intro-whatis.php. [Accessed: 09-Nov-2019].

[4] P. HoodaCheck and P. Hooda, "PHP vs. Node.js," *GeeksforGeeks*, 19-Feb-2019. [Online]. Available: https://www.geeksforgeeks.org/php-vs-node-js/. [Accessed: 09-Nov-2019].

[5] ”Why MySQL?,” *MySQL*. [Online]. Available: https://www.mysql.com/why-mysql/. [Accessed: 09-Nov-2019].

[6] ”The most popular database for modern apps,” MongoDB. [Online]. Available: https://www.mongodb.com/. [Accessed: 09-Nov-2019].

[7] ”Paper.js,” *Paper.js - About*. [Online]. Available: http://paperjs.org/about/. [Accessed: 09-Nov-2019].

[8] ”Phaser (game framework),” *Wikipedia*, 31-Oct-2019. [Online]. Available: https://en.wikipedia.org/wiki/Phaser_(game_framework). [Accessed: 09-Nov-2019].

6 TECH REVIEW: HANNAH VAUGHAN

College of Engineering

CS Capstone Tech Review

May 30, 2020

Track Town Pizza Web Application

Prepared for

Track Town Pizza

Tim Meyers

Prepared by

Group 74

Track Town Pizza

Hannah Vaughan

Abstract

One of the central goals of the Track Town Pizza web application project is to craft an efficient and more user-friendly website for potential customers that will encourage more web users to purchase from the company and thus increase sales. To achieve this goal, three fundamental components and their technology options must be analyzed so that the best technologies for this project's use cases are selected. The three pieces of this project that are addressed in this document include software architecture, frontend frameworks/libraries, and user management technologies. For software architecture, the client-server architecture with AJAX, single-page application architecture, and Node.js web application architecture are compared for scalability, reusability, and more. The frontend frameworks/libraries being compared in this document include AngularJS, Vue.js, and React; desirable attributes include easy integration and a wide variety of functionality. Finally, the user management technologies are being compared for security, easy incorporation, and statefulness. The technology selections for each piece will also be defended.

6.1 Introduction

The purpose of this document is to compare and contrast possible technologies for the following three pieces of the Track Town Pizza web application project:

- Software architecture
- Frontend frameworks/libraries
- User management technologies

Furthermore, this document will reiterate the goals of this project as well as the roles that I, Hannah Vaughan, possess which allow me to facilitate in the development of this product.

6.1.1 Project Role

My role in this project is similar to the other individuals on this team: to collaboratively design and develop a full web application with a well-structured architecture and pipeline as well as with all the features that the client desires. To be more specific, my foci begins with the initial system/software architecture design. The software architecture design is part of the initial application design and will determine how every component used in this application will work together and where specific logic will lie.

Beyond the architecture design, I am also leading in the frontend implementation of the web application by determining which framework will be most beneficial for displaying information to the screen in a user-friendly manner. Finally, my third responsibility is ensuring the application is secure and is able to properly manage user accounts with username and password credentials.

6.1.2 Project Goals

The central goal of this project is to create a scalable, maintainable, and user-friendly web application that replaces the existing website for Track Town Pizza in Eugene. This new website will aid the company in obtaining more customers and thus increasing sales. The web application will help the business achieve their goal of more customers by providing Internet users with an improved experience on the Track Town Pizza website, allowing them to obtain the information they want more quickly, and to make a more informed decision about purchasing from the company. Ultimately, Internet users will have an easier time learning about Track Town Pizza, what it sells, and how they can purchase from the company.

6.2 Piece 1: Software Architecture

The first component of this application that will be addressed in this document is the software architecture. This component is important because it will determine how many separate parts of the system are necessary and how they will work together. For this project, the software architecture will desirably be flexible, scalable, feasible, reusable, and secure [1].

6.2.1 Tech Option 1: Client-Server Architecture with AJAX

The client-server architecture with AJAX consists of the classic client-server architecture, but instead of the server having to send full HTML web pages and data to the client each time a request is made, the client consists of a series of components that can contact the server separately from each other via AJAX. AJAX stands for Asynchronous JavaScript and XML and is what connects all the client components to the server individually. This fundamental difference allows the client and server to send less data to each other at a time because all data transfers are handled in separate requests and thus improve performance.

Compared to the single-page application architecture and the Node.js web application architecture, the client-server architecture with AJAX is a bit easier to work with and is closer to the simple, easy-to-manage client-server architecture except without the notably slower performance. This layout, however, encourages a tightly coupled, monolithic layout that will be difficult to maintain in the future and will contain code that is deeply intertwined.

6.2.2 Tech Option 2: Single-Page Application Architecture

The single-page application (SPA) architecture is another variation of the classic client-server architecture in which the server sends HTML and JavaScript to the client for essentially a singular page. This singular page, however, becomes a full-fledged JavaScript application and implements all of its different functionalities on one page. The client page then transfers data in JSON to and from different web services to provide its full functionality. Similar to a client-server architecture with AJAX, the SPA architecture also utilizes AJAX. The main difference between the two is that the SPA architecture functions off of one web page which "allows the user to continue interaction with the page while new elements are updated, thus [receiving] the fast interaction with the content reloading at the same time" [2]. For this reason, SPA proves to be a viable design option for many use cases. In the use case of this project, though, too many pages require different types of content that would have fundamentally different code, leading to potentially bloated and monolithic software. Therefore, the SPA architecture is not the best system for this project and will not be used.

6.2.3 Tech Option 3: Node.js Web Application Architecture

The Node.js web application architecture is the most complex and unique of the three architectures addressed in this document. Instead of simply building off of the client-server architecture, this one consists of a data layer, a business layer, a web server, and the client side. The data layer holds the database and any external systems that may also need to interact with the business layer. In the business layer resides the application server and any file system that could be used by the application server and the web server. The application server interacts with the database and any external systems that are also kept in the data layer of the application, and the file system sends data to both the application server and the web server. The web server interacts with the application server and receives data from the file system. Finally, on the client side, there is the mobile browser, the web browser, and the application. The browsers and the web server frequently interact with each other.

This architecture is larger and more intricate than the client-server architecture with AJAX and the SPA architecture, which could make it more challenging to build, but the patterns ultimately allow for "code sharing and reusability" that "provide flexibility and reliability" [2] – all traits that are fundamental to the Track Town Pizza web application architecture. The complexity of the Node.js web application architecture may appear convoluted at first, but its levels of abstraction and singular-purpose components result in a well-structured system that seamlessly works together. In summary, the Node.js web application architecture will be used for the Track Town Pizza web application project because it is scalable, decoupled, reusable, flexible, and reliable.

6.3 Piece 2: Frontend Frameworks/Libraries

Frontend frameworks/libraries are another crucial component of this project because it will determine how the user interface (UI) will be implemented and how easily frontend data and requests will be processed and sent to other pieces of the software system. Regardless of which frontend framework is ultimately selected, Bootstrap will be included in the UI portion of the project as well. Bootstrap is a frontend framework for both CSS and JavaScript that allows developers to "[b]uild responsive, mobile-first projects on the web" [3].

Besides that, a desirable frontend framework/library will provide the developers with a well-structured yet flexible base upon which more code may be built. This base should allow for easy additions to web page views, a large variety of features to add, and smooth test integration.

6.3.1 Tech Option 1: AngularJS

AngularJS is a "structural framework" that can be used to develop "dynamic web apps" [4]. The point of this framework is to continue to use HTML like developers do for static web pages, then extending its syntax and functionality "to express [the] application's components clearly and succinctly" [4]. Additionally, the framework incorporates other programming paradigms like data binding and dependency injection, which allows for nonrepetitive code and centralized instantiation. Compared to React and Vue.js, Angular is the oldest of the three, restricts its implementation to the MVVM (Model-View-ViewModel) software design pattern, and harnesses a focus on scalability. Despite its classic status, its performance is notably slower than that of React and Vue.js. Additionally, its amount of different structures can lead to confusing implementations. Therefore, AngularJS will not be the frontend technology used in this project.

6.3.2 Tech Option 2: Vue.js

Vue.js is a “progressive framework” that can be “incrementally adopt[ed],” beginning at the view layer [5]. Its adaptability provides developers with the option to incorporate it into applications of any architecture, whether it be a single-page application or microservices. This framework’s flexibility also permits it to be integrated with other frameworks, libraries, and tools. Since Vue.js is such an adaptable framework, it would be relatively straightforward to incorporate into this project and to add incrementally as its role in the project evolves. If requirements change over the course of the project, Vue.js would be easier to revise and to maintain.

Compared to React, both frameworks are very similar: they “utilize a virtual DOM, . . . provide reactive and composable view components, . . . [and] maintain focus in the core library, with [other] concerns . . . handled by companion libraries” [6]. However, they do still possess fundamental differences from each other. One of the biggest differences between the two is that React is heavily JavaScript-oriented while Vue “builds on top of [classic web technologies]” [6]. React being more JavaScript-oriented can be more beneficial because it provides the power of a complete programming language on the frontend while Vue’s classic web technology base allows for easier integration whenever. Compared to AngularJS, Vue.js is a much lighter and more scalable framework that can be slowly incorporated into any project.

6.3.3 Tech Option 3: React

React is a JavaScript library that allows developers to build user interfaces that are declarative, component-based, and almost completely decoupled from other parts of whatever technology stack is being used [7]. Additionally, “React can also render on the server using Node and power mobile apps using React Native” [7]. This is important to this project because the web server will likely be run with Node, so selecting a frontend library that can also work server-side would allow for this project to not become cluttered with too many frameworks and will aid in any interconnectivity between the web server and the UI application that is introduced to the project in the future. Compared to AngularJS and Vue.js, React is JavaScript-oriented and focuses more on incorporating functional programming paradigms into the frontend than simply building off of classic HTML and CSS.

Of the above three tech options, React will be the frontend framework used for the Track Town Pizza web application project. React was selected over AngularJS and Vue.js because it is JavaScript-oriented, provides the full power of an entire programming language on the frontend, is notably fast (although Vue.js is too), has one-way data-binding to minimize unexpected errors (like AngularJS but without the slow performance), and can incorporate TypeScript. Its optimization for AJAX functionality will benefit in the development of the interactive pizza menu/calculator as well as the rendering of mini games that will be added to the web application. Overall, any of the three frontend frameworks/libraries detailed above could work well on this project, but React’s benefits combine into a mix of AngularJS and Vue.js with a powerful functional programming layer on top that prove its overwhelming benefits on this project.

6.4 Piece 3: User Management Technologies

User management technologies is an umbrella term that can be used to mean all libraries, frameworks, and paradigms in which an application may manage its users. In the context of this project, the only user roles that will be managed are potential customers and the owners of Track Town Pizza. The owners will have access to special pages which allow the owners to update the interactive menu and site information and to make posts about upcoming events and specials. This means that the number of users with username and password credentials will be small, but their security is imperative to the project. The three forms of user management being analyzed in this document include cookie-based authentication, token-based authentication, and a third-party provider that has become a standard authentication protocol called OAuth 2.0. For the sake of this project, the main attributes that are crucial for the user management technologies is its security, ease of integration with Node.js, and statefulness.

6.4.1 Tech Option 1: Cookie-Based Authentication

Cookie-based authentication is a form of authentication which receives a client’s login credentials, verifies them, creates a session ID that is stored server-side, then sends the session ID back to the client in the form of a cookie [8]. The session ID being stored on the server introduces state into the server, but it allows for all further requests from the client to be verified with the session ID. This statefulness requires the server to store as many session IDs as there are sessions, which can worsen performance if traffic is high enough. Therefore, cookie-based authentication directly implemented with Node.js will not be used in this project.

6.4.2 Tech Option 2: Token-Based Authentication

Token-based authentication is another standard form of authentication which has a client also send its credentials to the server side so that the server can validate them, generate a signed token (typically a JSON Web Token, or JWT) that contains the user's information, then send the token back to the client for future validation. Unlike cookie-based authentication, token-based authentication never stores the token on the server side (rendering the server stateless) and instead has the token "maintained at client side in local storage, session storage or even in cookies" [8]. Compared to cookie-based authentication, managing tokens on the client side with cookies replaces the storing of active session IDs server-side and instead translates the data in the cookies into tokens to be sent over. If token-based authentication implemented with Node.js is compared to OAuth 2.0, on the other hand, then token-based authentication lacks the ability to easily authenticate users via other existing web accounts. This proves that token-based authentication does not provide as many authentication and authorization options as OAuth 2.0, so it will not be used for the Track Town Pizza web application project.

6.4.3 Tech Option 3: OAuth 2.0

OAuth 2.0 (Open Authentication) is a third-party authentication and authorization framework and protocol that securely manages users as well as permits them access to specific resources. OAuth 2.0 can easily be integrated into Node.js via libraries such as openid-client and PassportJS. This option also allows users to login securely via other accounts, such as Google, Facebook, and Twitter, which would be beneficial to this project. If any of the Track Town Pizza owners wish to register their initial accounts with an existing web account elsewhere (for the sake of ease), OAuth 2.0 would be the best option among the three in this document to use. Additionally, this option would allow for straightforward social media integration in the future. Therefore, OAuth 2.0 will be the framework/protocol used for the security and user management technologies piece of the project.

6.5 Conclusion

The central goal of the Track Town Pizza web application is to provide the business's potential customers with a more user-friendly website that reliably, efficiently, and securely provides web users with important content regarding the business. Creating this website will provide users with a more encouraging experience that entices them to purchase from Track Town Pizza, either online, over the phone, or at the store. Three pieces that will be fundamental to this web application include the software architecture, the frontend frameworks/libraries, and the user management technologies. The software architecture selected for this project is the more complex Node.js web application architecture (compared to the client-server architecture with AJAX and the SPA architecture) because it is flexible, reliable, reusable, and promotes code sharing. On the frontend side, React is the library that will be used alongside Bootstrap. It is a better fit for this project than AngularJS and Vue.js because it provides the powerful functionality of an entire programming language (JavaScript) at the frontend's disposal. Finally, the user management technology that will be used for this project is OAuth 2.0 via libraries. This was selected over classic cookie-based authentication and token-based authentication because it allows for easy social media integration and streamlined registration processes. Overall, any of the technologies or paradigms analyzed in this document could be justifiable options for a web application, but a Node.js web application architecture, React and Bootstrap frontend libraries/frameworks, and an OAuth 2.0 user management framework are the most suitable options for the Track Town Pizza web application project.

References

[1] M. Aladdin, "Software Architecture - The Difference Between Architecture and Design," *codeburst*, 27-Jul-2018. [Online]. Available: https://codeburst.io/software-architecture-the-difference-between-architecture-and-design-7936abdd5830. [Accessed: 02-Nov-2019].
[2] "Web Application Architecture," *Existek*, 19-Dec-2018. [Online]. Available: https://existek.com/blog/web-application-architecture/. [Accessed: 03-Nov-2019].
[3] Bootstrap team, *Bootstrap*. [Online]. Available: https://getbootstrap.com. [Accessed: 02-Nov-2019].
[4] AngularJS team, "What Is AngularJS?," *AngularJS*. [Online]. Available: https://docs.angularjs.org/guide/introduction. [Accessed: 02-Nov-2019].
[5] Vue.js team, "Introduction," *Vue.js*. [Online]. Available: https://vuejs.org/v2/guide/. [Accessed: 02-Nov-2019].
[6] Vue.js team, "Comparison with Other Frameworks," *Vue.js*. [Online]. Available: https://vuejs.org/v2/guide/comparison.html. [Accessed: 02-Nov-2019].
[7] Facebook Open Source team, *React*. [Online]. Available: https://reactjs.org. [Accessed: 02-Nov-2019].
[8] V. Madurai, "Different ways to Authenticate a Web Application," *Medium*, 04-Feb-2018. [Online].

7 Weekly Blog Posts (Fall and Winter): Kaitlin Hill

7.1 Fall, Week 4

7.1.1 Progress, Problems, Plans

We met with our client. That was helpful for getting more information about our project and their expectations. We finished our project requirements. We are continuing to do research about appropriate technology stacks and hosting services for the website.

7.2 Fall, Week 5

7.2.1 Progress, Problems, Plans

We started research for our writing assignment. It has been helpful for us as a group to find our strength and research in those areas. We are also going to meet with another person who has quite a bit of web development knowledge to help guide us.

7.3 Fall, Week 6

7.3.1 Progress, Problems, Plans

Our project is going well. Splitting up the parts of our assignment helped us understand what questions we have moving forward. We have a plan to get our questions answered. The tech review also helped me think about how to plan out the information of our website.

7.4 Fall, Week 7

7.4.1 Progress, Problems, Plans

We are working on our design document. It is going well. Next week we will finish our design document. We haven't had any major problems. We are planning on meeting with our clients once more at the end of the term, so we can get their approval for our project design.

7.5 Fall, Week 8

7.5.1 Progress, Problems, Plans

Our next design document is almost complete. We will be able to turn it in on time. We have been working on mocks for our site as well. The only problem we had is figuring out what to include in the design document. We are planning on meeting with our client at the end of week 10.

7.6 Fall, Week 9

7.6.1 Progress, Problems, Plans

We are continuing to work on our mock ups for our website. My team had no problems this week, as it was a short week. We are planning on meeting with our client week 10 to discuss our project and our progress.

7.7 Winter, Week 1

7.7.1 Progress, Problems, Plans

This week, my team met to discuss how we are going to move forward with our project. We started by discussing how we want to start our project. We created a GitHub repo and started setting up our web app. We had some problems getting started with our web app. We tried to get too much working at once. We also had to cut our time short due to some miscommunication. We are planning on meeting every week for multiple hours to work on our project. We think that pair programming to start will be beneficial for our group.

7.8 Winter, Week 2

7.8.1 Progress, Problems, Plans

This week our group met twice to do some pair programming. We were able to create both our header and our nav bar. We also learned some bootstrap for responsive styling. We also met with our TA.Our team didn't really have any issues this week. The only issue is the learning curve of bootstrap.Our group made milestones so we know what we want to complete each week. Next week we will be working on our about page and contact page.

7.9 Winter, Week 3

7.9.1 Progress, Problems, Plans

Our group made our first merge into the master branch. We finished the header, footer, about page, and contact page. I had some issues getting google maps to work. I had to read stackoverflow and the google api docs to understand what it wanted me to do. Our group also is not completely pleased with our mobile view of the header, but we put that in our GitHub issues. Our group is planning on continuing progress with our project. We are starting on the blog and menu pages next.

7.10 Winter, Week 5

7.10.1 Progress, Problems, Plans

This week, the menu styling got fixed. Now the elements look centered and the box isn't too big. My team also had a very successful meeting for moving forward the rest of the term. One problem we had was one of our group members had made no progress this last week. It was a sanity check for this group member on how to move forward. Moving forward, our group is aiming to have all the front-end development completed by end of week 6. We are planning on getting the rest of the images ready on the 21st and we have a client meeting planned for the 21st.

7.11 Winter, Week 6

7.11.1 Progress, Problems, Plans

This week our team made our first draft poster. I personally fixed a few problems in the UI and opened some PRs. Our team had no problems this week. Our group plans on finishing our front end work very soon, so we can work on deployment.

7.12 Winter, Week 7

7.12.1 Progress, Problems, Plans

This week there was significant progress made on the pizza builder page. The UI is mostly complete with a couple bugs. The backend is half done. Before this week, this page hadn't been started yet. I had problems understanding the logic of react. I had to learn how to use state and keep and update the information. This week, the pizza builder page will be completed. We are also meeting with the client to discuss progress.

7.13 Winter, Week 8

7.13.1 Progress, Problems, Plans

This week, I made significant progress on one of the pages. It has been challenging to learn everything, but it is close to being finished. There were no big problems this week. Just little problems that I figured out as I continued developing. We plan on getting ready for our design review this weekend.

7.14 Winter, Week 9

7.14.1 Progress, Problems, Plans

This week our team gave our design review. We practiced then presented. We did have a problem this week, but we went to Kirsten to talk through it. We are planning on finishing up the last two assignments for capstone. We will complete them this next week.

7.15 Winter, Week 10

7.15.1 Progress, Problems, Plans

This week my team worked on our final assignments for this course. We also ran usability tests for our site to gather improvements.We had no problems this week. We plan to make our changes and to finish our final assignments for the course.

8 Weekly Blog Posts (Fall and Winter): Aiden Nelson

8.1 Fall, Week 3

8.1.1 Progress, Problems, Plans

The progress of our project is going well. Our group is meeting with our client today to discuss the requirements for the website. We have requested an extension for the requirements document so we will have time to draft it after our initial meeting today.

We will also be able to update the problem statement document after we know more about the specific issues they are having with their current website/system.

8.2 Fall, Week 4

8.2.1 Progress

Progress is going well. Last week we went to see our client at the restaurant we are building the website for. He gave us some good notes about what he would like to see in a new website.

8.2.2 Problems

Problems are minimal. We were having a hard time nailing down the scope of our project, but now that we have had multiple discussions with the Capstone instructors, as well as a meeting with the client, we feel like we know what we want to do.

8.2.3 Plans

Plans are underway! We have set up weekly meetings and are continuing research into avaliable technologies that would help us build the website. We will be meeting with our previous web development instructor to bounce some ideas of of him and get some suggestions for what tech we could use.

8.3 Fall, Week 5

8.3.1 Progress

Progress is going well. We have talked to the capstone instructors and will meet with them soon once we have made the final definitions for what are requirements will be. I feel good about the direction we are heading with this at the moment.

8.3.2 Problems

I see no immediate problems currently.

8.3.3 Plans

We plan on meeting with Rob Hess, the instructor for Web Development at OSU, the week after this next one to get advice.

8.4 Fall, Week 6

8.4.1 Progress

Everything is going well! I just finished the tech review and I feel great about it! It's finally done!

8.4.2 Problems

The only problem I have had is finding the time to get this done.

8.4.3 Plans

My team is meeting with Rob Hess, the instructor for the web development course at OSU, next Tuesday. We are hoping that he will be able to provide some valuable insight about our project and give us advice about what technologies we may want to use along with common pitfalls that developers often encounter in projects such as ours.

8.5 Fall, Week 7

8.5.1 Progress

I just met two hours ago with my capstone group to go over web design schemes and page organization planning. We plan on each of us separately completing a home page mockup by next Tuesday so that we can come together to compare, contrast, and decide which color schemes, page organizations and formats to choose.

8.5.2 Plans

We have organized our next client meeting (entailing a trip to Eugene) for Friday December 6th.

8.6 Fall, Week 8

8.6.1 Progress

This week the members of our group got together to present mockups. The mockups we each made were drafts for what a homepage for the website may look like. Based on ideas we all showed, we were able to make some design decisions about the website and also what color scheme and style we want the site to have.

8.6.2 Problems

This week it seemed like our TA Eytan Brodsky was not avaliable for meeting us at the normally scheduled time. We didn't retrieve any heads up or communication from him about this, but since we didn't have any questions we did not think it was a big deal.

8.6.3 Plans

The majority of the design document is completed. What remains is a little editing, some glossary additions, the introduction, and the abstract. I will be completing these tonight and tomorrow morning.

8.7 Fall, Week 9

8.7.1 Progress

We have been keeping up to date with our capstone course assignments. We have also been working on our mockups as well, and hope to have those entirely done by this next Tuesday.

8.7.2 Problems

We have received some "strange" feedback on a few of our assignment papers. Critical feedback we don't entirely understand. We may be going in to see the capstone instructors this next week to ask about it.

8.7.3 Plans

Our team will be meeting next week (Friday) with our client in Eugene to go over our complete draft set of mockups and receive feedback for them.

8.8 Fall, Week 1

8.8.1 Progress

Our team's first meeting was on Tuesday. During it we made plans for the term, including when our meetings would be, and what our plan was as far as getting started. We are currently at our second meeting right now (as of me writing this). We have set up our GitHub repository and have already begun coding for our project.

8.8.2 Problems

We have not encountered any problems yet. This is probably due to it being the first week and we just are getting into it.

8.8.3 Plans

Our first TA meeting will be next week on Tuesday.

8.9 Fall, Week 2

8.9.1 Progress

Our team has been pair programming for the first few parts of the website so we can stay consistent with our style and design. the parts we started with are the footer and navbar, which have also been good for practice with bootstrap, react, and next.js. Our mockups/framework designs have been altered slightly to accommodate for some changes that the client wanted to the blog section (adding events info).

8.9.2 Problems

Because we wanted to pair program when creating the first parts of the website, we have been spending a lot of time in these first few weeks getting together for meetings. One of my teammates has said that it is a little hard for them because they can't spend as much time with friends because of this (they have a busy schedule with classes and work, and our meetings have blocked out a lot of the time they have used in the past to spend with friends).

8.9.3 Plans

We plan to soon split up the work so we can all work individually and in parallel on our own time rather than at an allocated time for meetings. Once we start doing this (which will be organized during our next meeting), we will reduce the time we meet during the week.

8.10 Fall, Week 3

8.10.1 Progress

Our team has put the base of the website together. We have the navbar and footer working on all the pages. We have a basic version of our blog page up and running. The contact page is going well too. We have integrated a google maps view of the restaurant into the page by utilizing the Google Maps API.

8.10.2 Problems

The time we spend in team meetings is eating up a lot of our free time.

8.10.3 Plans

We are planning to cut down one of our meetings by an hour so we can free all of our schedules a little more. To compensate for this, we will be planning more independent work that we can do on our own (without the need for each other's immediate presence).

8.11 Fall, Week 5

8.11.1 Progress

Our group has continued to work without much hinderance. We have the majority of the front end done and will soon start to work on our back-end, which will include setting up a MongoDB database. As far as work that still needs to occur on the front end, we still have a small part of the menu to do, and setting up edit modules for an admin view.

8.11.2 Problems

We were having a difficult time choosing how we wanted to do authentication for the admin access of the website. We are currently looking into Auth 0 as a technology we can use for this.

8.11.3 Plans

In order to be ready for Alpha release our team will be working more hours than normal this weekend. We also will be preparing the first draft of our poster this following week.

8.12 Fall, Week 6

8.12.1 Progress

The front end is almost complete. This puts us in a good place for the alpha functionality deadline. We have also finished the first draft of the poster, which is due this Friday.

8.12.2 Problems

Other classes have ramped up their burdens as midterm season has begun. This has taken away some of time we have all had available to work on the website.

8.12.3 Plans

Back end development will begin soon. Next Friday we will be meeting with the client to present the work we have done so far. We hope to collect valuable feedback that will tell us if we need to make any changes or edits to the front-end of the site.

8.13 Fall, Week 7

8.13.1 Progress

The navbar has been fixed up so it is now much more clean and also changes it's format to fit the device's screen dimensions (mobile, desktop, ipad, etc...). Account pages have been added.

8.13.2 Problems

We have not yet completed the entire front, which according to our original plan was supposed to be completed by this date. We will have to put in more effort and time this next week to compensate for this. Since the majority of our team member's midterms are over, we should have the extra time.

8.13.3 Plans

We will be meeting with our client today to show off our progress so far. Since the majority of the front end is complete, we are in a good spot to show them what we have made so far. We will make notes of any changes they want (aesthetic and functional), and then evaluate later whether the requested changes are feasible for our team to implement in the remaining time we have.

8.14 Fall, Week 8

8.14.1 Progress

Today we went to see Kirsten in order to ask questions about how we can improve our poster since there were a few things we lost points for. In addition to asking questions about revising our poster, since our design review is next week on Thursday, we also asked questions about how we need to prepare for that. We went to visit our client in Eugene last Friday in order to gather the remaining photos we needed for the site, as well as to show them what progress we had made so far. They were pleased with what we presented them, which was the home, menu, about, contact, and blog pages. The Nav bar and dropdown has been further edited to be functional and look presentable on all of the chrome developer view device screen options. The home page has been created. The pizza builder feature has has a huge amount of progress done on it.

8.14.2 Problems

Due to us being beginners in using react, we have not been able to find an easy way to break up components that have attached functionality that interact together. We still also have not made as much progress as we would have wished on the front end. We know that this is an issue of needing to put in more time, but that has become difficult due to other group projects for other classes.

8.14.3 Plans

Since our reviewer is Scott, our team plans on seeing him next week. His next office hours are on Wednesday (the day before our review), so we plan on requesting an appointment to see him earlier in order to have time to take action in response to his feedback/advice. Things that I personally would like to finish up before the review are the blog edit page and the drinks page. We also hope to make some progress on the database integration research this weekend so that we may be able to speak more intelligently to our plan on implementing that.

8.15 Fall, Week 9

8.15.1 Progress

We just completed our design review yesterday. The blog page has been modified to look better and a few bugs have been removed. The pizza builder page has been completed and merged into the master branch.

8.15.2 Problems

When evaluating the amount of work we would have to do to integrate content management edit modes on all of the components of the website that will have editable data, we determined that it would be a more efficient use of our time to build a web form page in which the administrators will be able to edit site information. This will centralize the information as well, so the admins won't have to edit the same information in multiple places. We had a hard time with the design review due to our grading reviewer. We feel that they are basing our grade on the nature of our project (which is a website) rather than the actual technical aspects of our project and the progress we made. We went to discuss this issue with one of the capstone instructors today.

8.15.3 Plans

The pages left that we have to create (barring the database sides of things) are the information editing pages and the merchandise page. The blog page also needs some logic implemented for switching blog pages. We also need to create our video demo.

8.16 Fall, Week 10

8.16.1 Progress

Our team has been working on the final document and video for capstone requirements. We also had each of our team members perform a usability test on a person they knew. The usability test included questions about the user's thoughts on the current design and functionality of the website. It also included a few instructions, like find the pizza builder page, so that we could see how easily a user could interact with the site. From this we obtained many useful insights.

8.16.2 Problems

Our team has been absolutely swamped this last week due to finals and final projects. As a result of this we have not had much time to work on our project. We also were not able to get into contact with our TA. Both our team and him forgot to check in with each other last week. This week we tried to get into contact during our usual meeting time but we have not received any communications from him.

8.16.3 Plans

We have set up meeting times for next term. We plan to finish the backend over next week and winter break. This weekend we will be finishing our final capstone document and video.

9 Weekly Blog Posts (Fall and Winter): Hannah Vaughan

9.1 Fall, Week 3

9.1.1 Progress

At this point in the term, I have completed and submitted the individual problem statement as well as have met with my group members after each lecture. Additionally, my group and our TA have agreed upon a weekly virtual meeting time, and my group members and I often meet once a week beyond our other meetups to further plan next steps. We have not yet been able to meet with our client and discuss the project, but that will be changing by the end of the week.

9.1.2 Problems

My central problems this week are the vague requirements of our project and my group's and my inability to meet our client until the end of this week. We intend to resolve these issues ideally before week 4 even begins.

9.1.3 Plans

My group and I will be commuting down to Eugene tomorrow (Friday, 10/17) to meet our client for the first time and discuss what they would like to see out of this project. This will then allow us to finalize our group problem statement and to crack down on our first draft of the requirements document. Fortunately, our group receiving a two-day extension on the requirements document will provide us with just enough time to complete our initial draft by the end of the weekend.

9.1.4 Additional Notes

My group's team dynamic is excellent and going strong. We all work very well together and encourage each other to do the best we can for each other, ourselves, and our team.

9.2 Fall, Week 4

9.2.1 Progress

We met with our client last Friday (10/18) and discussed the scope of the project with them. We also completed the first two drafts of the requirements document.

9.2.2 Problems

The only problems my group members and I have is finding enough time to actually work on this project with how busy this term has been.

9.2.3 Plans

We plan to continue communicating with our clients and researching technologies for the new web application.

9.3 Fall, Week 5

9.3.1 Progress

This week my group mates and I determined nine different pieces of our project and divvied them up equally among each of us. We now all have three unique pieces so that we may research various tech options for draft 1 of the tech review.

9.3.2 Problems

We have no problems right now besides deciding which technologies to choose so that they all work together well.

9.3.3 Plans

Next steps include continuing research for the tech review, writing draft 1 of the tech review, and preparing for a meeting with Rob Hess to discuss modern web technology options.

9.4 Fall, Week 6

9.4.1 Progress

This week I revised my final draft of the Tech Review and submitted it, peer reviewing other classmates' drafts as well. Also, I met with my group on Tuesday, planned for our meeting with Rob Hess next week, and made a plan for how we all want to approach draft 1 of the Design document.

9.4.2 Problems

There are currently no problems hindering us in moving forward with our project.

9.4.3 Plans

Next week my group and I will be meeting with Rob Hess to discuss web application technology options. We have prepared a description of our project and guiding questions for him. This will help us finalize our first draft of the Design document.

9.5 Fall, Week 7

9.5.1 Progress

This week, I completed as much of my portions for the Design Document Draft 1 as possible, then met with my group mates and Rob Hess to discuss viable technology options for web applications.

9.5.2 Problems

I have no problems to report this week.

9.5.3 Plans

My group mates and I are working towards finishing a more polished Design Document Draft 2. We are also beginning our mockup design process so that we can receive feedback from our client during our December 6th meeting with them.

9.6 Fall, Week 8

9.6.1 Progress

This week (week 8 of fall term), my group mates and I completed our second draft of the design document and submitted it by Saturday, November 23 (the deadline). Additionally, all of us made mockups of the home page for the web application so that we may establish the theme, color scheme, etc. of our mockups moving forward.

9.6.2 Problems

Our main problem this week was understanding what had to be included in the design document because the outline appeared to be different for every group with which we spoke.

9.6.3 Plans

Moving forward, we are working on creating our initial mockups by Tuesday, November 26, so that we may receive feedback on them and revise them further before our client meeting on Friday, December 6. Additionally, we are sending all of our documents to our client in preparation for the upcoming client approval deadline on the same day as our client meeting (Friday, December 6). Finally, we are still accepting feedback on our second draft of the design document so that we can provide the best possible draft for our clients.

9.7 Fall, Week 9

9.7.1 Progress

This week, my group mates and I shared our mockups for the Home, About, and Contact pages, then we established our remaining timeline for the term regarding deadlines and mockup goals.

9.7.2 Problems

No problems to report this week.

9.7.3 Plans

My group mates and I plan to complete our initial mockup drafts by Wednesday, 12/4, submit our Progress Report by Friday, 12/6, complete our nearly final mockup drafts by Friday, 12/6, and meet our clients to review our mockups and documents from this term on Friday, 12/6. (A lot is happening on Friday, 12/6.)

9.8 Winter, Week 1

9.8.1 Progress

This week, my team and I met three times to establish our code base for the term and to figure out how we want to approach our project to reach the deadlines.

9.8.2 Problems

This week, my team and I mildly struggled with coming to an understanding of how our code base should look to start off and how we want our technology to develop moving forward.

9.8.3 Plans

Since we only established so much of our code base this week and are still working on how we want everything to be set up moving forward, my teammates and I plan to meet on Monday of week 2 to establish term-long milestone goals and tasks to break up work among the 3 of us. To resolve our struggle with our code base, we will also meet one to two times next week specifically to pair program and to create a more substantial code base for the rest of the project.

9.9 Winter, Week 2

9.9.1 Progress

This week, my teammates and I continued to meet and pair program together so that we could establish the project's code base together and ensure that we are all on the same page with coding standards moving forward. We have mostly completed the React components for the navigation bar and the footer (save for a few styling inconsistencies) and have begun developing content for the About and Contact pages. Additionally, we agreed on several milestones/checkpoints for implementation, especially revolving around the due dates for Alpha and Beta functionality.

9.9.2 Problems

The only problems that my teammates and I ran into this week were setting up the project and getting Bootstrap incorporated into source control. We have created a temporary solution that gets Bootstrap working, but it results in repeated code and we hope to eventually replace it with a better solution in future weeks.

9.9.3 Plans

Moving forward, my teammates and I plan on still pair programming each week but with slightly less hours than these first two weeks now that we have gotten started with implementation. We will plan our work around the milestones we created on Monday, January 13th, and split up responsibilities based on our current progress and how that relates to our goals.

9.10 Winter, Week 3

9.10.1 Progress

This week, my teammates and I completed the navigation bar and footer React components (save for a few design inconsistencies that we intend on resolving in future weeks) and combined them into a Layout component that will be reused for all pages. Additionally, we completed the Contact and About pages and managed to incorporate a basic Google Map into the Contact page. We started working on the Blog and Blog Post pages and are currently working on rendering dummy JSON data onto these pages.

9.10.2 Problems

The main problem we struggled with this week was properly integrating Google Maps into our project with a Google Maps API key as well as figuring out how to store the key as a secret on our Next.js web server.

9.10.3 Plans

Moving forward, my teammates and I plan on continuing to research how to keep secrets in our project in order to ensure security, implementing the Blog and Blog Post pages, and researching how to integrate OAuth into a Next.js/Node.js server. I also intend to create the Login and Register pages for the admin accounts and hopefully implement basic credentials management either in a proof of concept application (at the very least) or in the project itself on its own branch.

9.11 Winter, Week 5

9.11.1 Progress

This week, my teammates and I continued to work on completing as much of the front end UI as possible for the Alpha functionality deadline. In particular, I completed the Login and Register pages (neither are hooked up to authentication middleware yet). I also started the Account Management page, which allows admin users to update their username and password as well as delete their account. My other teammates made progress on the menu pages as well as the Blog page.

9.11.2 Problems

My biggest inhibitor for my studies right now is personal issues that are occurring in my life and are detracting from my ability to contribute my full time and effort to this project. I also had to put authentication setup on the back burner because I was having difficulty finding a setup that works well for the project and I needed to dedicate more time to the UI.

9.11.3 Plans

Moving forward, I plan on completing the Account Management and Home pages before Alpha functionality is due, then working further on authentication setup for Beta release.

9.12 Winter, Week 6

9.12.1 Progress

This week, my teammates and I completed the first draft of our poster as well as continued to work on the front end UI of our web application. I completed the Account Management page and started on the Home page.

9.12.2 Problems

The only problems inhibiting progress on my portions of the project are a busy schedule, homework in other classes, and the difficulty of ensuring truly mobile-friendly web pages.

9.12.3 Plans

Moving forward, I plan to continue to test different strategies in implementing mobile-friendly web pages when I have the time. The front end UI (i.e. Alpha functionality) is almost complete, so I will return to researching authentication integration once Alpha functionality is complete.

9.13 Winter, Week 7

9.13.1 Progress

This week, I continued to make the account pages more mobile friendly and responsive as well as further implement the content of the home page. I dynamically added data on upcoming events and then rendered them on the home page in preparation for pulling data from a MongoDB database.

9.13.2 Problems

The only issues my teammates and I are facing are the ability to allocate time to work on this project during such a busy time in the term in which other assignments are much more time restricted as well as the ability to ensure mobile-friendly web pages.

9.13.3 Plans

Moving forward, I plan on completing the home page and making it mobile-friendly and future-proof (for the database and for user sessions). I also plan on designating more time to this project when possible.

9.14 Winter, Week 8

9.14.1 Progress

This week, I finished designing the Home page and customizing the view to be mobile-friendly on various devices. My teammates and I also revised our draft of the Engineering Expo poster to better fit the poster’s requirements as well as began preparation for our Design Review, which occurs on Thursday, March 5th.

9.14.2 Problems

The only problems with which my teammates and I are struggling are having enough time to progress in our project as quickly as we would like and customizing our web pages to be truly mobile-friendly with the resources we have.

9.14.3 Plans

Moving forward, I plan on implementing the front-end design of the Lunch Buffet Menu page, then moving onto back-end technology research to start incorporating MongoDB, account management, and user sessions.

9.15 Winter, Week 9

9.15.1 Progress

This week, my teammates and I cleaned up our code and some of our UI in preparation for our Design Review, then we created slides for the Design Review and practiced in advance. Finally, we completed our Design Review, received feedback from Scott and other groups, and finally provided feedback for the other groups.

9.15.2 Problems

Still, the main problem that my teammates and I are facing is having sufficient time to work on this project alongside other coursework and life responsibilities. I am also facing the challenge of figuring out how to host both a MongoDB database and a Next.js application on Heroku in preparation for the integration of a database and the deployment of our application.

9.15.3 Plans

My current plan is to continue researching how to host both a MongoDB database and a Next.js application on Heroku, then also implementing the Content Management System views in parallel.

9.16 Winter, Week 10

9.16.1 Progress

This week, my teammates and I started working on our Winter End-of-Term Report, planning our demo video, and establishing what our remaining roles will be for the remaining tasks we hope to complete by the end of this term. I will be responsible for beginning to research how to set up MongoDB databases that can be hosted online (ideally Heroku), how to integrate authentication into the existing web application, and how to host the web application and database together.

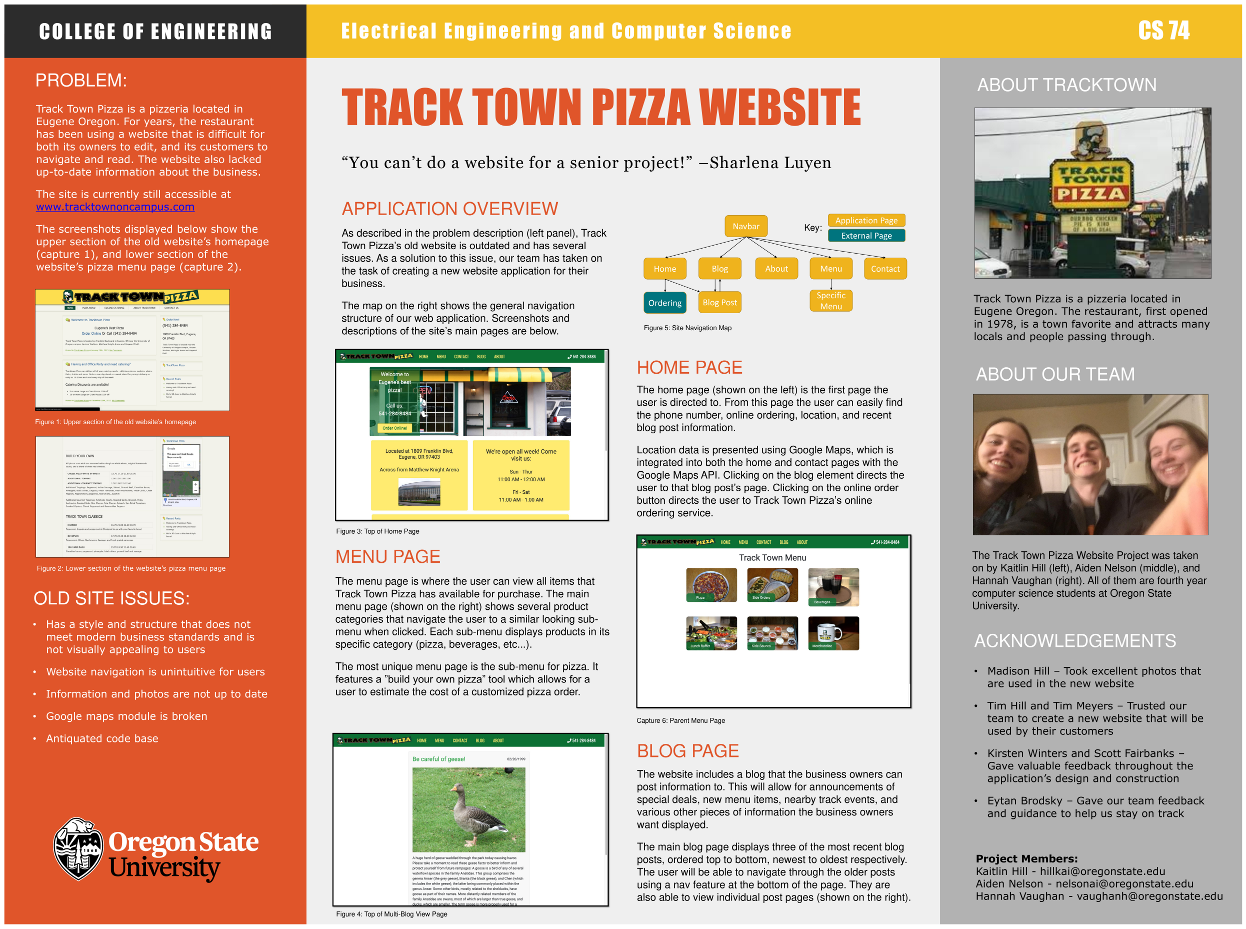
9.16.2 Problems

Once again, the only issue that my teammates and I have run into this week is not having enough time to complete the tasks we had hoped to complete by the end of the week. This is due in part to outside responsibilities increasing as the end of the term nears and also due in part to accessibility restrictions set in place because of the COVID-19 conditions being handled by the university.

9.16.3 Plans

Moving forward, my teammates and I plan on recording our demo video and completing our Winter End-of-Term Report on Sunday, March 15th (two days in advance of the due date), and we intend on emailing (and potentially even having a Zoom meeting with) Rob Hess to ask for professional expertise on our back-end implementation. As for myself, I plan on researching how to create a MongoDB database and how to host it on Heroku. After that, I plan on looking more into authentication with Next.js servers.

10 Final Poster for Showcase


Navbar
Key:
Application Page
External Page
Home
Blog
About
Menu
Contact
Ordering
Blog Post
Specific Menu
Track Town Menu
Pizza
Side Orders
Beverages
Lunch Buffet
Side Sauces
Merchandise

COLLEGE OF ENGINEERING

Electrical Engineering and Computer Science

CS 74

PROBLEM:

Track Town Pizza is a pizzeria located in Eugene Oregon. For years, the restaurant has been using a website that is difficult for both its owners to edit, and its customers to navigate and read. The website also lacked up-to-date information about the business.

The site is currently still accessible at www.tracktownoncampus.com

The screenshots displayed below show the upper section of the old website's homepage (capture 1), and lower section of the website's pizza menu page (capture 2).

Figure 1: Upper section of the old website's homepage

Figure 2: Lower section of the website's pizza menu page

OLD SITE ISSUES:

- Has a style and structure that does not meet modern business standards and is not visually appealing to users
- Website navigation is unintuitive for users
- Information and photos are not up to date
- Google maps module is broken
- Antiquated code base

Oregon State University

TRACK TOWN PIZZA WEBSITE

"You can't do a website for a senior project!" –Sharlena Luyen

APPLICATION OVERVIEW

As described in the problem description (left panel), Track Town Pizza's old website is outdated and has several issues. As a solution to this issue, our team has taken on the task of creating a new website application for their business.

The map on the right shows the general navigation structure of our web application. Screenshots and descriptions of the site's main pages are below.

Figure 5: Site Navigation Map

Figure 3: Top of Home Page

HOME PAGE

The home page (shown on the left) is the first page the user is directed to. From this page the user can easily find the phone number, online ordering, location, and recent blog post information.

Location data is presented using Google Maps, which is integrated into both the home and contact pages with the Google Maps API. Clicking on the blog element directs the user to that blog post's page. Clicking on the online order button directs the user to Track Town Pizza's online ordering service.

MENU PAGE

The menu page is where the user can view all items that Track Town Pizza has available for purchase. The main menu page (shown on the right) shows several product categories that navigate the user to a similar looking sub-menu when clicked. Each sub-menu displays products in its specific category (pizza, beverages, etc...).

The most unique menu page is the sub-menu for pizza. It features a "build your own pizza" tool which allows for a user to estimate the cost of a customized pizza order.

Capture 6: Parent Menu Page

Figure 4: Top of Multi-Blog View Page

BLOG PAGE

The website includes a blog that the business owners can post information to. This will allow for announcements of special deals, new menu items, nearby track events, and various other pieces of information the business owners want displayed.

The main blog page displays three of the most recent blog posts, ordered top to bottom, newest to oldest respectively. The user will be able to navigate through the older posts using a nav feature at the bottom of the page. They are also able to view individual post pages (shown on the right).

ABOUT TRACKTOWN

Track Town Pizza is a pizzeria located in Eugene Oregon. The restaurant, first opened in 1978, is a town favorite and attracts many locals and people passing through.

ABOUT OUR TEAM

The Track Town Pizza Website Project was taken on by Kaitlin Hill (left), Aiden Nelson (middle), and Hannah Vaughan (right). All of them are fourth year computer science students at Oregon State University.

ACKNOWLEDGEMENTS

- Madison Hill – Took excellent photos that are used in the new website
- Tim Hill and Tim Meyers – Trusted our team to create a new website that will be used by their customers
- Kirsten Winters and Scott Fairbanks – Gave valuable feedback throughout the application's design and construction
- Eytan Brodsky – Gave our team feedback and guidance to help us stay on track

Project Members:
Kaitlin Hill - hillkai@oregonstate.edu
Aiden Nelson - nelsonai@oregonstate.edu
Hannah Vaughan - vaughanh@oregonstate.edu

11 Project Documentation

To install and run our program, one can look at the GitHub readme from https://github.com/track-town-pizza/capstone/blob/master/README.md. It can also be seen below:

11.1 Installation

11.1.1 Prerequisites

Before setting up this project, download the most recent version of Node.js. To check if Node.js is installed and/or what version it is, run the following command:

```
node —version
```

11.1.2 Clone

Once the most recent version of Node.js has been installed, clone this repository with https://github.com/track-town-pizza/capstone.git.

11.1.3 Setup

Create a .env file in the root directory of the project and add the MONGODB_URL, URL_ROOT, API_KEY, and PORT environment variables with the values securely delivered to you. If you do not have the proper values for any of the variables, please contact any of the project members to gain access to them. Note: the project will not run properly without some of these values. Please make sure the values you have are correct.

Install the project's npm packages using

```
npm install
```

then specifically install the next npm package to ensure the server commands will run with

```
npm install next
```

11.2 Usage

To run the web server and start the application in a development environment, run

```
npm run dev
```

If you wish to run the application in a production environment, build the application with

```
npm run build
```

then start the application with

```
npm run start
```

To access the application on a web browser, navigate to the following URL:
http://localhost:3000
To access the Content Management System, navigate to the following URL:
http://localhost:3000/admin/managementHub

11.3 Additional Information

To view the code review critiques written for this project and our responses to them, please refer to the tables documented in Appendix C.

12 Recommended Technical Resources for Learning More

- What web sites were helpful? (Listed in order of helpfulness.)
    1) https://scrimba.com/playlist/p7P5Hd
    2) https://www.figma.com/
    3) https://getbootstrap.com/
    4) https://www.w3schools.com/
    5) https://nextjs.org/
    6) https://uxplanet.org/bad-ux-roundup-17-rain-information-architecture-and-ergonomics-5376c6cf7e69
    7) https://blog.tubikstudio.com/ux-design-readable-user-interface/

8) https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Destructuring_assignment

- Were there any people on campus that were really helpful?
  - Rob Hess was helpful. He was able to guide us towards some useful technologies. He could also answer questions we had about how to use our hosting service, Heroku.

13 Conclusions and Reflections: Kaitlin Hill

- What technical information did you learn?
  - From this project, I gained a deeper understanding of web development. I learned how to use react and bootstrap to create a website front-end. I learned next.js and how to use npm packages in code. I also improved my git and GitHub skills as well as my understanding of issues and Kanban boards.
- What non-technical information did you learn?
  - From this project, I learned more about communication and where to look for help. In a project like this, communication is essential. I had to communicate with our clients and my team members. I learned to communicate through email, text, and person-to-person. I also learned how to look for help. There were many resources I used such as people who had more technical knowledge than I did, websites, and my team members. I learned how to ask questions that would get me closer to my goal.
- What have you learned about project work?
  - I have learned that project work can go well only if you communicate and set expectations. As a group, so much more work can be accomplished. You can also check in on each other's work to create an even better product.
- What have you learned about project management?
  - I have learned that coming up with goals is essential for project work. It is important to lay out what you want to do and when you want to do it. This will help keep everyone in the group on task and able to complete everything in a timely manner. If you don't, then everyone will have a different idea of what should be done and it will get confusing and frustrating.
- What have you learned about working in teams?
  - I have learned that communication is key and you can't expect your teammates to function like you do. You must communicate your needs and listen to what they need. You also need to understand that every team member will work at their own pace. You must trust that your team members will do their part.
- If you could do it all over, what would you do differently?
  - If I could do it all over again, I would set more deadlines for finishing work. It was hard because I didn't know how long things would take. I would still set some estimated guesses though. If you don't set any deadlines, it is hard to track progress and make sure that work is getting done in a timely manner. Also, without deadlines, you tend to prioritize other work. Deadlines would help ensure the project is always on track for being finished.

14 Conclusions and Reflections: Aiden Nelson

- What technical information did you learn?
  - I learned how to use many new technologies for the first time. Among those included are React, Next.js, MongoDB, and various JavaScript libraries. I also gained more proficiency in using technologies I was at least slightly familiar with. These included JavaScript, Node.js, Git, Github, Bash, HTML and CSS. I also learned a lot more about website structure, security, and protocols.
- What non-technical information did you learn?
  - I learned how to design a website by starting with a wireframe. I also learned how to create thorough documentation for waterfall projects, some skills of which can be applied to other development methodologies as well. I gained a lot of experience with version control (that might be technical), working and collaborating on a team, and planning out work effectively.
- What have you learned about project work?
  - I have learned that creating documentation can take up a huge amount of time. I learned that it is important to make goals and set deadlines. I have learned that when working through a problem that you have spent hours on while making little progress, sometimes it is best to consult the other members on the team to work through it together.
- What have you learned about project management?
  - I have learned that setting deadlines and goals is key in project management. Without them it is extremely easy to fall behind. Deadline is often used to define something when it really should be a goal or “soft” deadline. Too often than not, you only know how long something is going to take once you have actually done a little bit of it.
- What have you learned about working in teams?
  - I have learned that it is crucial to assign tasks and objective dates to have those tasks completed by. I have learned that it can be very helpful to work through some problems as a team rather than an individual, and consult them too. With more esoteric and specific questions, google sometimes is not sufficient, and a person with more expertise on the topic is.
- If you could do it all over, what would you do differently?
  - If I could do it all over again then I wouldn’t have had us spend the time researching and trying to implement user authentication. For the purposes of the application, we only really needed an interface for modifying the database, so that is what we ended up building. This was separate from the website and only ran on a local machine, which eliminated the need for web authentication and a lot of other security overhead (while still maintaining just as much security). We didn’t know we wouldn’t be implementing web authentication until much later in the project, so we could have saved some time up front if we knew that. I would also have done more self learning in preparation for implementing some technologies, such as MongoDB and Next.js.

15 Conclusions and Reflections: Hannah Vaughan

- What technical information did you learn?
  - I learned how to create a web application with a technology stack that was relatively new to me. On the front end, I learned how to create reusable components with the JavaScript framework React and I honed my web design skills with Bootstrap. On the back end, I practiced creating a database API with Next.js, an isomorphic framework that provides both front-end and back-end functionality. Finally, I improved my source control and database management skills with GitHub and MongoDB/MongoDB Atlas.
- What non-technical information did you learn?
  - I learned what to do and what not to do when planning a web application, from its architecture to the wireframe technology used to design each web page before implementing them. Additionally, I learned more about working freely on a team without a manager guiding the implementation and what it takes to get everything done on time on our own.
- What have you learned about project work?
  - I learned that project work and how well it is executed heavily depends on the team you are working with, the project management style your team has developed, and the communication habits that have been established. Splitting up work is essential, and can be the true key behind successfully completing a project.
- What have you learned about project management?
  - When it comes to project management, I learned that creating explicit tasks and openly communicating who does what is absolutely essential to making a project happen to completion as seamlessly as possible. As long as everyone is on the same page and general deadlines are agreed upon, everyone has the opportunity to complete responsibilities on time.
- What have you learned about working in teams?
  - Similar to what I have mentioned previously, communication and proper team planning over time are two fundamental components to working in teams because ensuring everyone is on the same page can save a lot of time and confusion, and understanding that responsibilities are split up and not all on one person can relieve a lot of stress and misunderstandings.
- If you could do it all over, what would you do differently?
  - If I could do this project all over again, I would try incorporating the Agile methodology more strongly throughout our entire project so that my teammates and I could develop our requirements based on a widely-used standard and so that tasks could be split up into sprints with more solid deadlines. My teammates and I did well with completing what we needed to with the time we had, but I think the sprints and backlogs and such would have given us that extra advantage to get everything done even faster and even more seamlessly.

Appendices

APPENDIX A

ESSENTIAL CODE LISTINGS

A.1 The About Page:

This portion shows how to create and format react elements.

```
import Layout from "../components/Layout"
import Link from "next/link"
import fetch from "isomorphic-unfetch"
import React from "react"

const About = ({ about, info }) => {
    return (
        <Layout info={info}>
            <div className="w-75 mx-auto">
                <h1 className="text-center">About Track Town Pizza</h1>
                <div className="text-center">
                    <img src={about.imgLink} className="w-75 img-fluid rounded"
                        alt="Track Town Pizza" />
                </div>
                <br/>
                <div className="mx-auto">
                    <p className="about-text">{about.p1}</p>
                    <p className="about-text">{about.p2}</p>
                    <p className="about-text">{about.p3}</p>
                    <p className="about-text">{about.p4}</p>
                </div>
                <div className="text-center">
                    <h3>Check out the rest of Track Town USA</h3>
                    <a href={about.pdfLink} target="_blank" className="text-success">
                        <h5>Track Town USA Map</h5>
                    </a>
                    <Link href="/blog">
                        <a className="text-success"><h5>Track Town Pizza Blog</h5></a>
                    </Link>
                </div>
            </div>
        </Layout>
    )
}

About.getInitialProps = async (req, res) => {
    const resJson = await fetch(`${process.env.URL_ROOT}/api/about`).then(_ => _.json())
    const info = await fetch(`${process.env.URL_ROOT}/api/info`).then(_ => _.json())

    return {
        about: resJson,
        info: info
    }
}

export default About
```

A.2 The API Call for the About Page:

This portion shows how to make an API call to the database to get the information that will appear on the web page.

```
import nextConnect from "next-connect"
import middleware from "../../utils/database"

const handler = nextConnect()

handler.use(middleware)

handler.get(async (req, res) => {
        // Set CORS headers in advance
        res.setHeader("Access-Control-Allow-Origin", "*")
        res.setHeader("Access-Control-Allow-Methods", "GET,PUT,POST,DELETE,OPTIONS")
        res.setHeader(
            "Access-Control-Allow-Headers",
            "Content-Type, Authorization, Content-Length, X-Requested-With"
        )

        let doc = await req.db.collection("about").findOne({ "key": "about" })
        res.json(doc)
})

export default handler
```

Appendix B

Additional Artifacts

Hannah Vaughann (left), Aiden Nelson (center), and Kaitlin Hill (right) – the members of Capstone group CS 74 – standing in front of the Track Town Pizza sign.

A fresh and delicious pepperoni pizza handcrafted by the Track Town Pizza in-house workers.

Appendix C

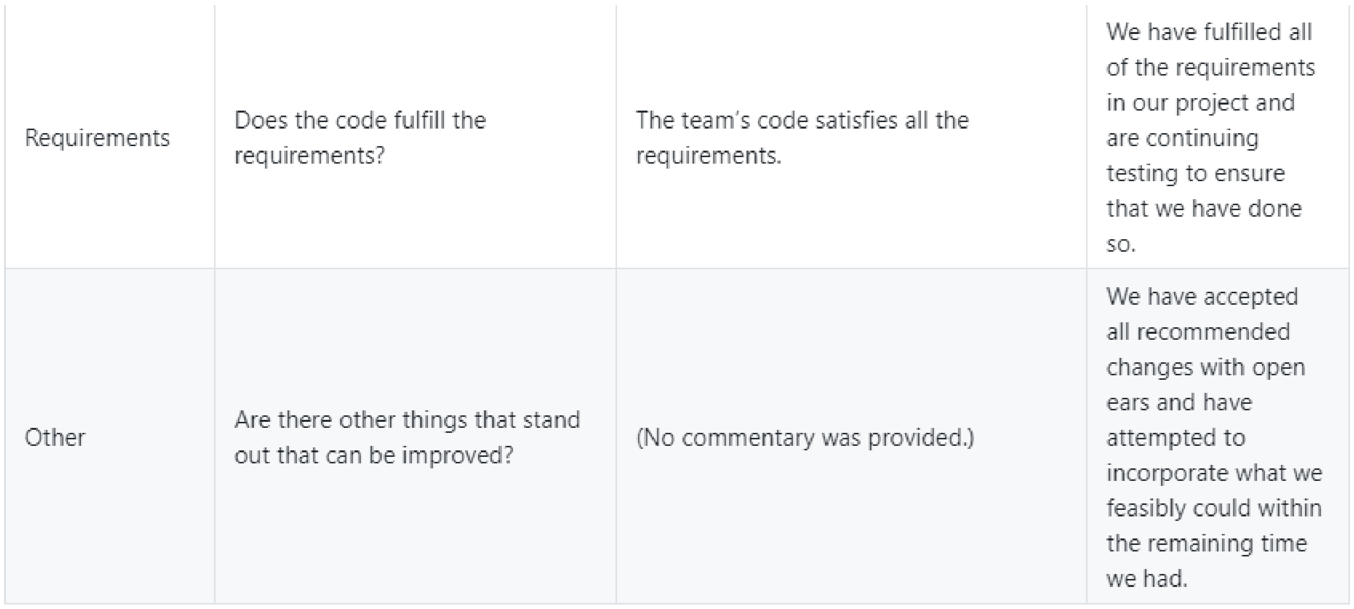
Code Review Critiques and Responses

Category	Description	Reviewer's Comment	Action Taken by Reviewed Group
Build	Could you clone from Git and build using the README file?	was able to build using the Readme file. Readme file was easy to follow and had step by step instructions.	We took measures to ensure the README file remains as clear and as easy-to-follow as possible moving forward.
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	The code adhered to general guidelines of code style and variable names were easy to follow. All the functions in JavaScript files had sane variable names which were easy to follow.	We took steps to make sure the code stays at this quality or better.
Implementation	Is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	The implementation was clean and safe, and it will be easy to write a functionally equivalent code.	We continued to work on consolidating the code so that it is even more efficient than from during the code review.
Maintainability	Are there unit tests? Should there be? Are the tests covering interesting cases? Are they readable?	There were no unit tests done.	We ultimately decided that unit tests were not necessary in this UI-oriented project and that consistent functional testing was a better use of our time and energy for this project.
Requirements	Does the code fulfill the requirements?	The code almost fulfills the requirements. It only lacks the database to actually work with the real inventory.	We completed the database implementation and fully integrated it into the project.
Other	Are there other things that stand out that can be improved?	Other things that can be improved is the user interface of the app.	We have accepted all recommended changes with open ears and have attempted to incorporate what we feasibly could within the remaining time we had.

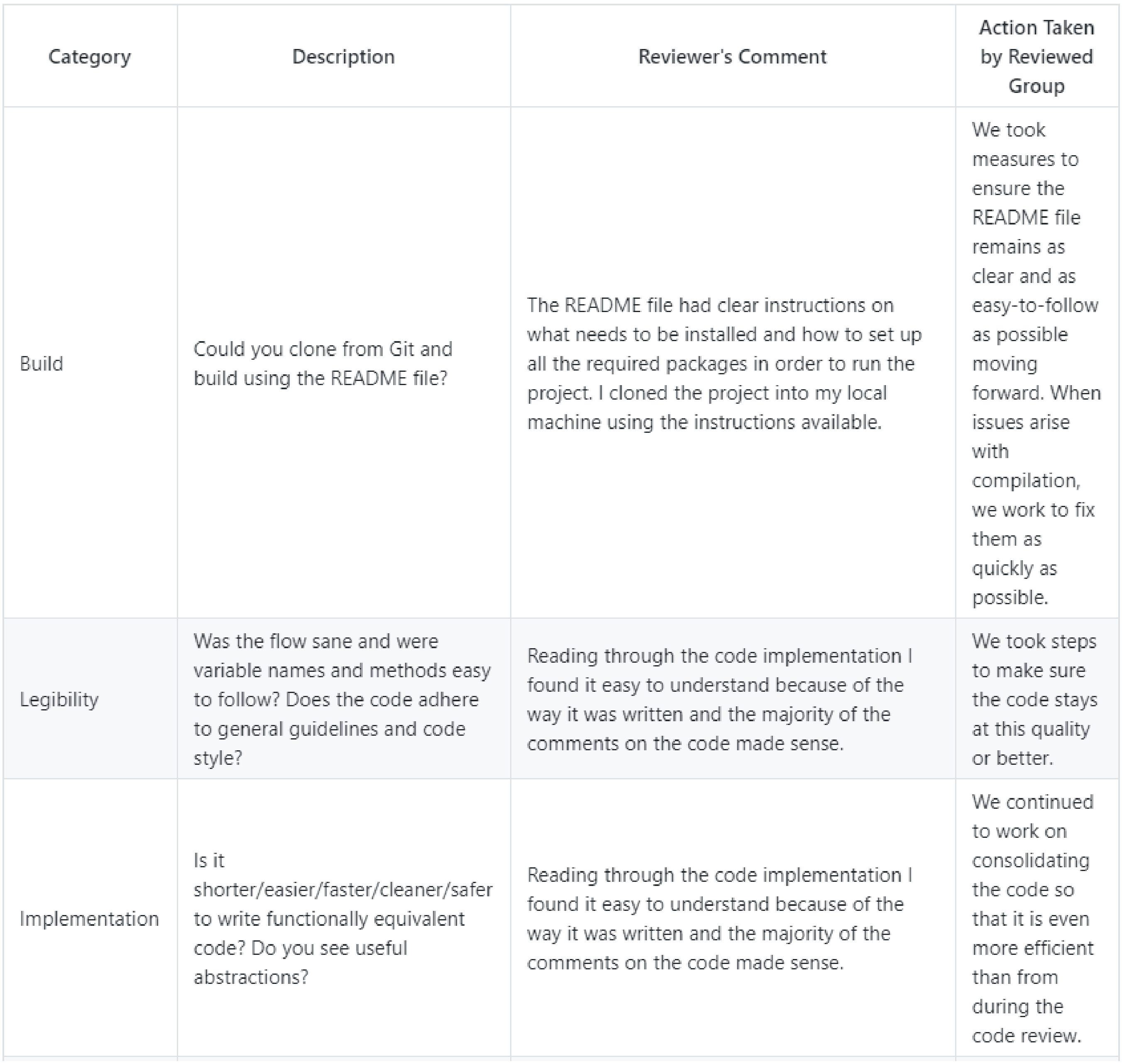
Category	Description	Reviewer's Comment	Action Taken by Reviewed Group
Build	Could you clone from Git and build using the README file?	I was able to clone the project using the README file easily. The instructions were clear and straight to the point. The document has spacing to, so it made it easier to go step by step. The instruction where introduce in only three section which made it so easy to know what to do. Of course, if the user doesn't have a good understanding of computer science will not be able to compile it. As a computer science student, I found it clean how to compile it and run it.	We took measures to ensure the README file remains as clear and as easy-to-follow as possible moving forward.
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	The way the code implemented is clear. Looking to the variable names, I was able to understand how each element constructed. Each function had a clear heading and represented its task and how it contributes to the code. There is minor function that I had to read the function before and the one after to be able to understand the structure of the code.	We took steps to make sure the code stays at this quality or better.
Implementation	Is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	The way the code is written is written well in most of the time. Checking the written file as manifest.json, index.html, ...etc. I could understand how the code works. It was easy since the majority of the code was in line and indented where it supposed to be indented. Plus, there was almost no running the code for long lines, so I didn't have to put the screen in full screen mode or scroll to read all the code.	We continued to work on consolidating the code so that it is even more efficient than from during the code review.

Maintainability	Are there unit tests? Should there be? Are the tests covering interesting cases? Are they readable?	The maintainability of the code is there. The code could be change easily since there are many files and each one has a specific task. Even throughout the testing everything was able to connect with the other file without any issue. Each file had their own correct directory too. It is easy to go through the code and files without any difficulty.	We continued to keep our code base organized and easy to navigate.
Requirements	Does the code fulfill the requirements?	(No critique was provided.)	We believe that we have fulfilled all of the requirements in our project and are continuing testing to ensure that we have done so.
Other	Are there other things that stand out that can be improved?	The design could be improved. It looks good, but as a person who does art, I wish if they had a person where their task to only focus on the colors of the design and some of the shapes proportions. Great job overall!	We agree that having a professional designer for this project would have been lovely so that we could have a UI specialist/team make the most state-of-the-art design choices, but we are working to make our website as aesthetically pleasing as possible while also still being practical, functional, and up to the client's satisfaction.

Category	Description	Reviewer's Comment	Action Taken by Reviewed Group
Build	Could you clone from Git and build using the README file?	Yes, it seems other members were able to just follow along and get the website running. The README provides a straightforward approach detailing very specific instructions. I was not able to compile their code, but that seems to be due to an npm issue on my side. Otherwise, going through the review the team has satisfied their client requirements.	We took measures to ensure the README file remains as clear and as easy-to-follow as possible moving forward. When issues arise with compilation, we work to fix them as quickly as possible.
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	Yes, the code is simple to understand and follows conventions for web design. They have also provided comments to the code and have named their files appropriately. Looking through their code files as well, variables follow naming conventions and are consistent.	We took steps to make sure the code stays at this quality or better.
Implementation	Is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	The code makes use of React libraries and creates abstraction when needed to help reduce the backend clutter.	We continued to work on consolidating the code so that it is even more efficient than from during the code review.
Maintainability	Are there unit tests? Should there be? Are the tests covering interesting cases? Are they readable?	The team also mentioned that during testing they simply do some visual testing and checking the backend database. The team also mentioned that they do manual testing to make sure that all the prices add up and that users can go through the motions of ordering a pizza.	We ultimately decided that unit tests were not necessary in this UI-oriented project and that consistent functional testing was a better use of our time and energy for this project.



Requirements	Does the code fulfill the requirements?	The team's code satisfies all the requirements.	We have fulfilled all of the requirements in our project and are continuing testing to ensure that we have done so.
Other	Are there other things that stand out that can be improved?	(No commentary was provided.)	We have accepted all recommended changes with open ears and have attempted to incorporate what we feasibly could within the remaining time we had.



Category	Description	Reviewer's Comment	Action Taken by Reviewed Group
Build	Could you clone from Git and build using the README file?	The README file had clear instructions on what needs to be installed and how to set up all the required packages in order to run the project. I cloned the project into my local machine using the instructions available.	We took measures to ensure the README file remains as clear and as easy-to-follow as possible moving forward. When issues arise with compilation, we work to fix them as quickly as possible.
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	Reading through the code implementation I found it easy to understand because of the way it was written and the majority of the comments on the code made sense.	We took steps to make sure the code stays at this quality or better.
Implementation	Is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	Reading through the code implementation I found it easy to understand because of the way it was written and the majority of the comments on the code made sense.	We continued to work on consolidating the code so that it is even more efficient than from during the code review.

Maintainability	Are there unit tests? Should there be? Are the tests covering interesting cases? Are they readable?	(No commentary was provided.)	We continued to keep our code base organized and easy to navigate.
Requirements	Does the code fulfill the requirements?	(No commentary was provided.)	We have fulfilled all of the requirements in our project and are continuing testing to ensure that we have done so.
Other	Are there other things that stand out that can be improved?	I believe the overall visualization of the website still needs some improvements. From what I understand that their main task is to upgrade this pizza website application into a more modern, easy to navigate web page. Therefore, making further changes to some of graphic designs and interfaces would improve the web page. I think focusing on the structure first is the way to go here considering how the original pizza wb page was very basic but hard to navigate. Overall great job and I like what you guys accomplished with this.	We have accepted all recommended changes with open ears and have attempted to incorporate what we feasibly could within the remaining time we had.

Category	Description	Reviewer's Comment	Action Taken by Reviewed Group
Build	Could you clone from Git and build using the README file?	Yes, the repository was easily runnable.	We took measures to ensure the README file remains as clear and as easy-to-follow as possible moving forward. When issues arise with compilation, we work to fix them as quickly as possible.
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	The coding convention seems consistent. Nothing seems outlandish beyond what I believe are typoes.	We took steps to make sure the code stays at this quality or better.
Implementation	Is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	It looks like they follow the standard MVC architecture for modern websites. There are strange bits where the same styling is inlined multiple times, but that's easily refactorable.	We continued to work on consolidating the code so that it is even more efficient than from during the code review.
Maintainability	Are there unit tests? Should there be? Are the tests covering interesting cases? Are they readable?	No unit tests, however it is difficult to test UI.	We ultimately decided that unit tests were not necessary in this UI-oriented project and that consistent functional testing was a better use of our time and energy for this project.
Requirements	Does the code fulfill the requirements?	Seems to meet most of the requirements (at least for front end). Code seems good.	We have fulfilled all of the requirements in our project and are continuing testing to ensure that we have done so.
Other	Are there other things that stand out that can be improved?	Since the backend isn't hooked up yet, I don't know how it will be implemented. However, I would highly recommend being careful with how remote site administration is set up. This is definitely one part that cannot be implemented incorrectly.	We are taking as many modern security precautions as we can to ensure our content management system is secure and only accessible by authorized entities.

Category	Description	Reviewer's Comment	Action Taken by Reviewed Group
Build	Could you clone from Git and build using the README file?	Yes. Very clear and straightforward. It was easy to follow through to access the dev site.	We took measures to ensure the README file remains as clear and as easy-to-follow as possible moving forward. When issues arise with compilation, we work to fix them as quickly as possible.
Legibility	Was the flow sane and were variable names and methods easy to follow? Does the code adhere to general guidelines and code style?	Well organized in folders. Really easy to navigate in a text editor. Code is readable, usable. Able to run dev, unable to run build, start.	We took steps to make sure the code stays at this quality or better.
Implementation	Is it shorter/easier/faster/cleaner/safer to write functionally equivalent code? Do you see useful abstractions?	I think their code is really clean, as an React application.	We continued to work on consolidating the code so that it is even more efficient than from during the code review.

Maintainability	Are there unit tests? Should there be? Are the tests covering interesting cases? Are they readable?	I don't think unit tests are needed here. Console log output is good enough.	We ultimately decided that unit tests were not necessary in this UI-oriented project and that consistent functional testing was a better use of our time and energy for this project.
Requirements	Does the code fulfill the requirements?	Yes, very clear and followed great coding style. Well organized.	We have fulfilled all of the requirements in our project and are continuing testing to ensure that we have done so.
Other	Are there other things that stand out that can be improved?	Beverages menu: Unnecessary scrolling. Could minimize the time that the user needs to scroll down to view the rest of Beer, wine list and so on. Description/Price box under menu: The box follows while using scroll. That's fine. When I tried to see other pizza's details, instead of refreshing the box and showing the newer msg, it showed 2 boxes and one on top of the other. (Pepperoni description -> Garlic Des -> Garlic Price)	We have accepted all recommended changes with open ears and have attempted to incorporate what we feasibly could within the remaining time we had.