Static Site Generator

DESIGN DOCUMENT

Team 22
Client: Amanda Dropinski
Advisers: Samik Basu
Team Members:
Andrei Baechle
Joshua Flory
Meet Patel
Bennett Ray
Luke Shay
Ty Trinh
Team Email: sdmay21-22@iastate.edu
Team Website: https://sdmay21-22.sd.ece.iastate.edu/

Revised: 10.22.2020 v2

Executive Summary

Development Standards & Practices Used

Software Practices

- Kanban Development
- Unit Testing
- Branch-Review-Merge Workflow

Summary of Requirements

List all requirements as bullet points in brief.

- Sites generated must be static
- Users will be given base templates to edit
- Sites will have different theme options
- Some content will be integrated with Buildertrend (login, lead generation, contact)
- Admin panel to edit templates
- Basic api to save and load edited templates
- Users will be able to view saved sites by visiting the site domain

Applicable Courses from Iowa State University Curriculum

- S E 319
- COM S 309
- S E 329
- S E 339

New Skills/Knowledge acquired that was not taught in courses

- React
- TinaCMS

- Express
- TypeScript

Table of Contents

List of figures/tables/symbols/definitions

Task Decomposition Tables: For each subtask associated with a task, lists the number of the subtask, name, and which subtasks need to be completed before the task.

Subtasks Lists: Lists the subtasks for each semester. Each subtasks has a start date and expected duration in days indicating how long the subtask will take to complete.

Gantt Charts: The Gantt charts for Fall 2020 and Spring 2021 generated from the subtask lists.

Required Time Table: For each task, lists the task number, duration in days, and the expected number of hours that each task will take to complete.

System Block Diagram: A simple block diagram of our project.

1 Introduction

1.1 Acknowledgement

Amanda Dropiskini: Project Manager

Amanda will be our project manager for this project. She will be assisting with organizing check ins and understanding the project purpose and goals.

Thomas Judge: Application Developer

Thomas is a full-time application developer from Buildertrend who will be providing technical assistance for us during project development.

1.2 Problem and Project Statement

Problem Statement

Buildertrend wants to provide an easier process for clients to build and maintain their company websites.

Solution Approach

The Static Site Content Management system will allow clients to easily set up and maintain a company website via an administrative panel and will increase usage of this feature, as well as provide Buildertrend's clients a custom website building experience so they can stand out in their respective markets.

1.3 Operational Environment

There is no physical component to our project, and all work for our project will be completed on our personal computers. We will not be deploying our project to a server by the end of the year, so any hazards will depend on the individual's own environment.

1.4 Requirements

Functional Requirements

- Admin panel to edit basic JSON file that drives the site
- Generated sites must be static

- Users will be able to use generic templates for their sites including Home, Blank, Gallery, Showcase, Blog, Login, Contact
- Users will be able to choose from at least 2 themes
- Client sites will be integrated with Buildertrend login form
- Basic API to save and retrieve JSON data for generated sites

Technical Requirements

- Frontend application will a created using React with TypeScript
- TinaCMS will be used to build the Content Management System
- All rendering will be handled on the client side
- Node will be used to build the file server

1.5 Intended Users and Uses

The intended users are Buildertrend clients who run businesses and need a website to serve their customers. They will be able to use our site to create their own websites for their clients.

1.6 Assumptions and Limitations

Assumptions

- We only need to support english because all of their clients are in US
- There will be no unexpected costs since we have no budget
- All planned Buildertrend integrations will have no alternative integrations
- Our NodeJS server does not need to be very in depth because it will be replaced

Limitations

- This semester is shorter than previous semesters, so we have fewer weeks to complete our milestones
- A full time team will be taking over the project at the end of May, so we are expected to only use frameworks and libraries that Buildertrend uses
- During this semester, we will not be performing tests for high volume traffic on client websites or the API endpoints
- The generated sites must be static

1.7 Expected End Product and Deliverables

End Product: Static Site Content Management System

Deliverable 1: File Server

Delivery Date: 10.13.2020

This deliverable is for the basic file server that will store and return the edited files for the sites created by the users. This component will be relatively simple since it will be replaced after the project is given to Buildertrend.

Deliverable 2: Static Site CMS Web Application.

Delivery Date: 04.13.2020

This deliverable is for the frontend React application that users will use to create their websites. Buildertrend's customers will be able to use the admin template to edit provided templates to build their sites. Sample templates will include, home, gallery, login, blank, showcase, blog and contact. Each template will have sample content that users can edit to fit their site. Some features will be integrated with the Buildertrend site, like the login and templates with lead generation. Admins will also have the options to choose different themes for their sites. Once the admin has completed the site, their customers will be able to visit the domain for the site to view it.

2 Project Plan

2.1 Task Decomposition

Task 1: File Server

Subtask Number	Task Name	Predecessor
1	Setup file Server	
2	Create sample text file for saving to file system	
3	Post endpoint for file server	1
4	Get endpoint for file server	1
5	Create Basic React with TypeScript Application	
6	Page to save a file on the file server	3, 5
7	Page to load a file from the file server	4, 5
8	Get endpoint to retrieve a list of files	2
9	Page to retrieve a list of files from file	5, 8

	server	
10	Add option to download files	6
11	Implement versioning system	3,4
12	Allow retrieval based on file versions	11

Task 2: Admin Panel

Subtask Number	Task Name	Predecessor
1	Create Admin Page	
2	Add TinaCMS to React Application	
3	Admin: Edit and save template content	1,2
4	User: view modified templates	1,2

Task 3: Template Creation

Subtask Number	Task Name	Predecessor
1	Home Template	
2	Blank Template	
3	Login Template	
4	Integrate Buildertrend login with Login template	3

Task 4: Finish templates and 2 Themes

Subtask Number	Task Name	Predecessor
1	Gallery Template	
2	Showcase Template	
3	Blog Template	
4	Contact Template	

5	Gallery - Document uploads	1
6	Theme 1	
7	Theme 2	

Task 5: 1 Theme, Content, Google Maps Integration

Subtask Number	Task Name	Predecessor
1	Theme 3	
2	About	
3	Floor plan	
4	Testimonials	
5	Google Maps Integration	

Task 6: Finish content, Reviews Integration, SEO

Subtask Number	Task Name	Predecessor
1	Services	
2	Privacy	
3	Policy Process	
4	BT Reviews Integration	
5	Implement versioning for editor	
6	SEO for templates	

2.2 Risks And Risk Management/Mitigation

Task 1: File Server

1. File permissions

   Risk Factor: 0.1

Explanation: As of right now, we are using our own file server which will have the risk of having other users access files they should not have access to. Since our file system server is temporary, this has little risk to our implementation for the frontend.

2.

Task 2: Admin Panel

1. TinaCMS does not fulfill requirements for our content management system.

   Risk Factor: 0.2

   Explanation: This library was chosen by Buildertrend who already tested the library out and determined it was the best CMS for the site. However, it is possible that as the site features become more advanced, TinaCMS will prevent us from implementing features as originally anticipated.

Task 3: Template Creation

1. Templates mockups may change as project progresses

   Risk Factor: 0.1

   Explanation: As templates are completed, Buildertrend may revise what templates need to look like. This is not a big risk, but this may be in any capacity, so we should be prepared to change.

2. Cross-Browser Compatibility

   Risk Factor: 0.1

   Explanation: Different web browsers interpret web pages in different ways, so some browsers may not support certain stylings or functionality. In order to keep these pages consistent, we will have to be aware of these differences.

2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Milestone 1: Basic File Server will be completed by 10.13.2020

Evaluation Criteria: Users will be able to run both the file server and frontend application and will be able to save a JSON body to the file system and download a file.

Milestone 2: Admin Panel will be completed by 10.27.2020

Evaluation Criteria: Users will be able to use the admin panel to edit and save a template file. The user will then be able to view the edited template.

Milestone 3: At least 3 templates and one integration will be completed by 11.10.2020

Evaluation Criteria: Users will be able to build sites by editing the templates. One of the templates will include integration

2.4 Project Timeline/Schedule

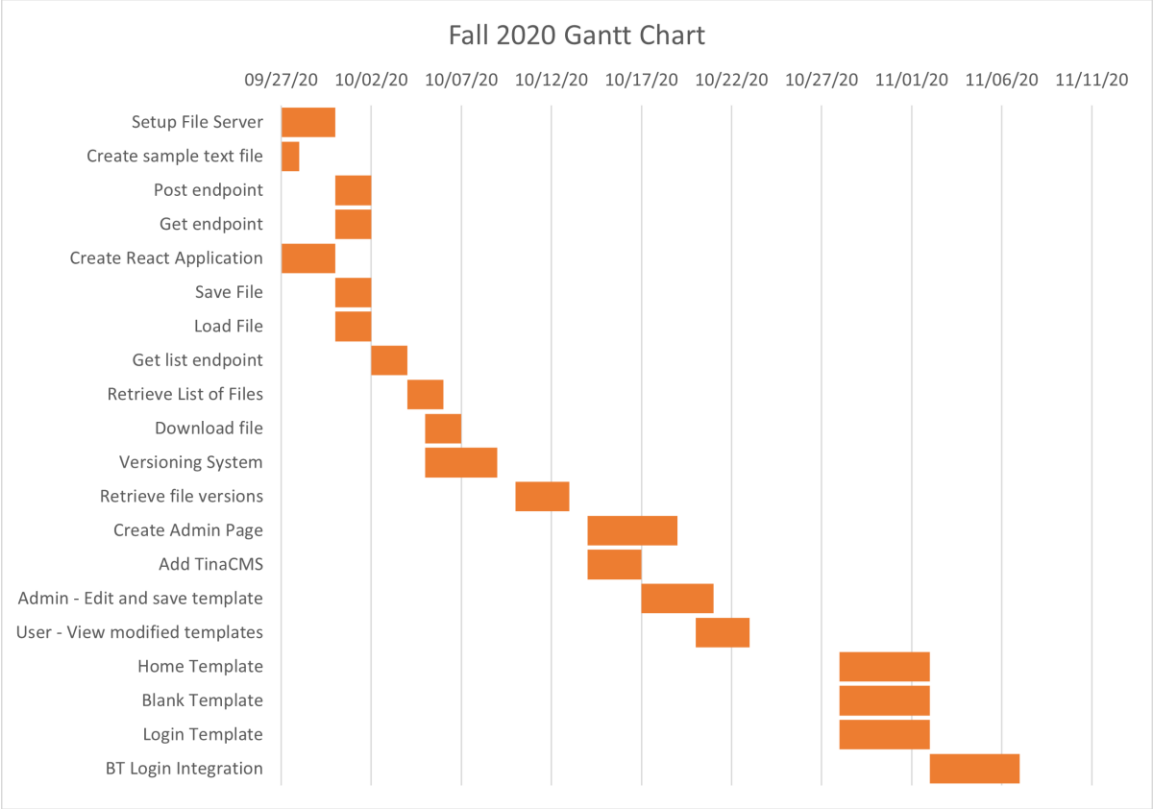
We have already been given the first 3 milestones for the first semester of our project. We also have deadlines for each milestone. Based on these milestones and deadlines, we created tasks and estimated the number of days each task will take. The table below shows each task with their start date and the estimated number of days that each task will take. This table was used to create the Gantt chart.

Subtask Table

Semester 1		
Task	Start Date	Duration
Setup File Server	09/27/20	3
Create sample text file	09/27/20	1
Post endpoint	09/30/20	2
Get endpoint	09/30/20	2
Create React Application	09/27/20	3
Save File	09/30/20	2
Load File	09/30/20	2
Get list endpoint	10/02/20	2
Retrieve List of Files	10/04/20	2
Download file	10/05/20	2
Versioning System	10/05/20	4
Retrieve file versions	10/10/20	3
Create Admin Page	10/14/20	5
Add TinaCMS	10/14/20	3
Admin - Edit and save template	10/17/20	4
User - View modified templates	10/20/20	3
Home Template	10/28/20	5
Blank Template	10/28/20	5
Login Template	10/28/20	5
BT Login Integration	11/02/20	5

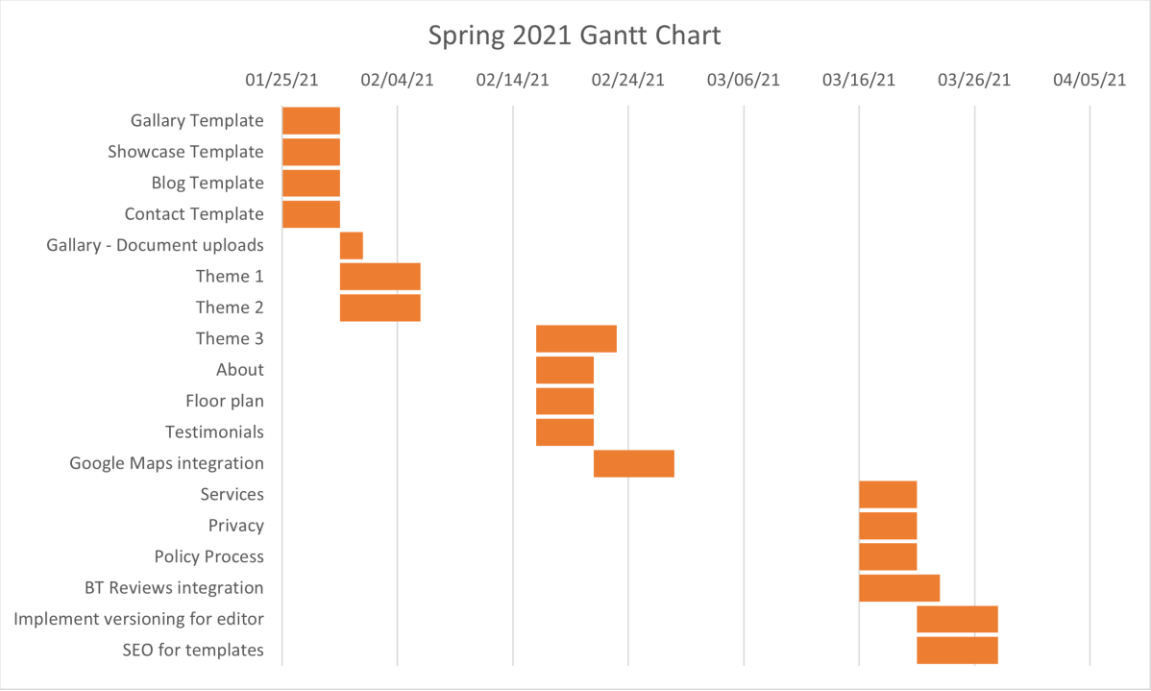
Semester 2		
Task	Start Date	Duration
Gallery Template	01/25/21	5
Showcase Template	01/25/21	5
Blog Template	01/25/21	5
Contact Template	01/25/21	5
Gallery - Document uploads	01/30/21	2
Theme 1	01/30/21	7
Theme 2	01/30/21	7
Theme 3	02/16/21	7
About	02/16/21	5
Floor plan	02/16/21	5
Testimonials	02/16/21	5
Google Maps integration	02/21/21	7
Services	03/16/21	5
Privacy	03/16/21	5
Policy Process	03/16/21	5
BT Reviews integration	03/16/21	7
Implement versioning for editor	03/21/21	7
SEO for templates	03/21/21	7

Semester 1



If we are able to complete all tasks in the expected time, we should be able to complete the first 3 milestones before the end of the semester.

Semester 2



This chart is less complete than the Fall 2020 chart. Since we do not know how much progress we will have made by the end of November, there are gaps between the end of one task and the start of the next to anticipate any new subtasks that might be added for each task.

2.5 Project Tracking Procedures

Github Project

Our team will be using the project feature in Github to track the progress of the project. Each task will be represented as an issue on the project board, and each issue will be assigned to a task member.

Project Board Columns

To-Do: Cards in this column have not been started but are ready to be picked up by a developer

In Progress: Column for cards that have been started but no pull request has been created

Code Review: Column for cards that have pull requests that need to be reviewed

Testing: Cards whose functionality has been merged to master but a developer still needs to verify the functionality still works.

Done: Card works in master and is complete.

2.6 Personnel Effort Requirements

Time Required for Each Task

To determine the number of hours required for each task, multiplied the duration from the table in 2.4 by 2, since we assumed that we would spend, on average, 2 hours a day working on each subtask.

Required Time Table

Task	Duration (Days)	Hours	Explanation
1	28	56	Task 1 involves a lot of setting up components that will be used for the rest of the project. We assumed it would take some time to establish the development cycle during the first task.
2	24	48	Task 2 has fewer subtasks than task 1, but the subtasks are more complex than the ones in task 1. Since the due date for this milestone will most likely need to be moved up, we expect to spend less time on this task.
3	20	40	Task 3 also has fewer subtasks than task 1, and we believe that the subtasks will be relatively easy to implement. Like task 2, we will also have to move up the due date.
4	36	72	Task 4 will require more time due to the number of templates and the 2 themes that will need to be implemented. We expect the themes to take a while to implement across all of the templates.
5	29	58	Task 5 includes the addition of more content to the templates, another theme, and google maps integration. We expect that adding a 3rd theme and implementing the google maps integration will be the most time consuming subtasks for this task.
6	36	72	Task 6 involves implementing the remaining features for this project. This task has some potentially complex subtasks like the reviews integration and SEO, so we anticipate having to spend a significant amount of time on this task.

Total # of Hours: 346

Personal Effort Estimation

Since there are about 23 weeks left in the semester, and we have 6 group members, we expect the personal effort to be about 2.5 hours per week for project implementation.

2.7 Other Resource Requirements

Github will be used for source control for the project.

2.8 Financial Requirements

Financial requirements are not relevant for this project since there are no anticipated costs.

3 Design

3.1 Previous Work And Literature

Squarespace

Squarespace provides software as a service for building and hosting websites. They provide templates for different types of websites including photography, blogs, and online stores. Plans cost $12-$46 a month.

Advantages:

- A lot of templates and customization options
- Allows coding and customization through CSS, HTML, JavaScript
- Drag and drop user interface
- Community support through forums and guides
- Provides Google Analytics [2]
- Fully responsive templates

Shortcomings:

- Squarespace won't help with troubleshooting if CSS, HTML, or JavaScript has been manually changed [3]
- Restricting content through membership on a site can be difficult [3]
- User interface and code changes frequently [3]

Wix

Wix is another website that allows users to build their own websites using templates. The basic plan Wix plan is free, but has limited features. Premium plans cost $14-$39 a month.

Advantages:

- More beginner-friendly than Squarespace [2]
- Drag and drop user interface
- Free basic plan
- Provides Google Analytics [2]

Shortcomings:

- Templates are not responsive [1]
- Cannot use CSS, HTML, or JavaScript
- Templates cannot be changed after the website has been built [1]

3.2 Design Thinking

Define Aspects

- Buildertrend customers need to be able to market themselves and their products to their customers
- Buildertrend customers do not have a way to create their own sites without using sites like Squarespace or Wix or hiring a developer to create it for them.

Design Choices (already provided by Buildertrend)

- Creating a file server to save templates and edited site content
- Creating a frontend React with TypeScript application for building the sites
- Using TinaCMS to implement the content management system

3.3 Proposed Design

File Server

Since the file server will be replaced once we give the project to Buildertrend, we had some more freedom to choose the implementation approach. However, they suggested we use Node since it was lightweight and already had a file system module for creating and modifying files on our local machines. We decided to use Node and have already implemented this component of our project. We created a rest api with endpoints for saving and retrieving files, and have already demoed to our client.

Web Application

The web application will be a React with TypeScript application as specified by our client. Using React with TypeScript is standard for all new web applications created by Buildertrend, so we had to follow that standard. The decision to use TinaCMS to build the content management system was already provided by Buildertrend who determined that it would be the best editing toolkit for our project.

Even though we did not experiment with other approaches, Buildertrend had already created a test application to ensure that TinaCMS had the features needed to fulfill the requirements for our project. It allows users to edit provided templates and save those changes, and then we can save those modified files to our file servers. Most of our work will involve creating the default templates and the content for the templates.

3.4 Technology Considerations

React with TypeScript

Strengths

- Large development community
- There are many React tutorials available
- Already used by Buildertrend, so they can help us with development
- TypeScript is more familiar to people who have experience with Object-oriented programming as opposed to JavaScript which is a scripting language.
- TinaCMS is designed to work with React

Weaknesses

- React is not taught in any course at ISU, so for people who have not used it before, there may be a learning curve
- Most React tutorials use JavaScript and not TypeScript, so it can be harder to find tutorials or help online that use React with TypeScript
- React creates single page applications which are not as performance oriented as other options

Alternatives

- **Angular**: Another web application framework that could be used to create the web application. Angular already uses TypeScript by default, so it would be easier to find

documentation and guides that use TypeScript. However, we will be giving this project to Buildertrend when we are done, and Buildertrend does not use Angular.
- **Vue with TypeScript**: A relatively new web application framework that could be used instead of React with TypeScript. Vue supports using Typescript, but again, Buildertrend uses React for web development.

Node

Strengths

- Lightweight, don't need to create a large application to create a simple file server
- Easy to learn
- Already has modules for working with a device's file system
- Large package ecosystem

Weaknesses

- Relatively simple, if we end up needing to put more work into the file server, we may want to reevaluate using Node
- Insecure compared to other runtimes

Alternatives

- **Creating a backend application with a database:** If we decide that the file server needs to be more complex than initially specified, we could replace the Node file server with a backend application that is connected to a database. However, we do not anticipate having to work on the file server any more than we already have, so this will not be necessary.

TinaCMS

Strengths

- Allows editing of templates
- Easy to add to a preexisting project
- Allows for generation of static websites

Weaknesses

- New technology to the team, may be challenging to work with

Alternatives

- **Magnolia CMS:** Another content management system toolkit that works with React. Allows a lot more customization and features than TinaCMS, but is not free.

3.5 Design Analysis

File Server

We know that using Node for the file server works since we have already completed and demoed the file server to our client. This component was intended to be relatively simple so we most likely will not have to modify it again.

Web Application

Based on the test application that Buildertrend created that was using TinaCMS, we know that the design will work for our project. However, that test application was a relatively simple application, so as we get farther into the project, we may realize that TinaCMS does not have all of the necessary features to fulfill the project requirements. The test application also did not feature saving modified files, so we will have to figure out how to implement that aspect of our project as well.

3.6 Development Process

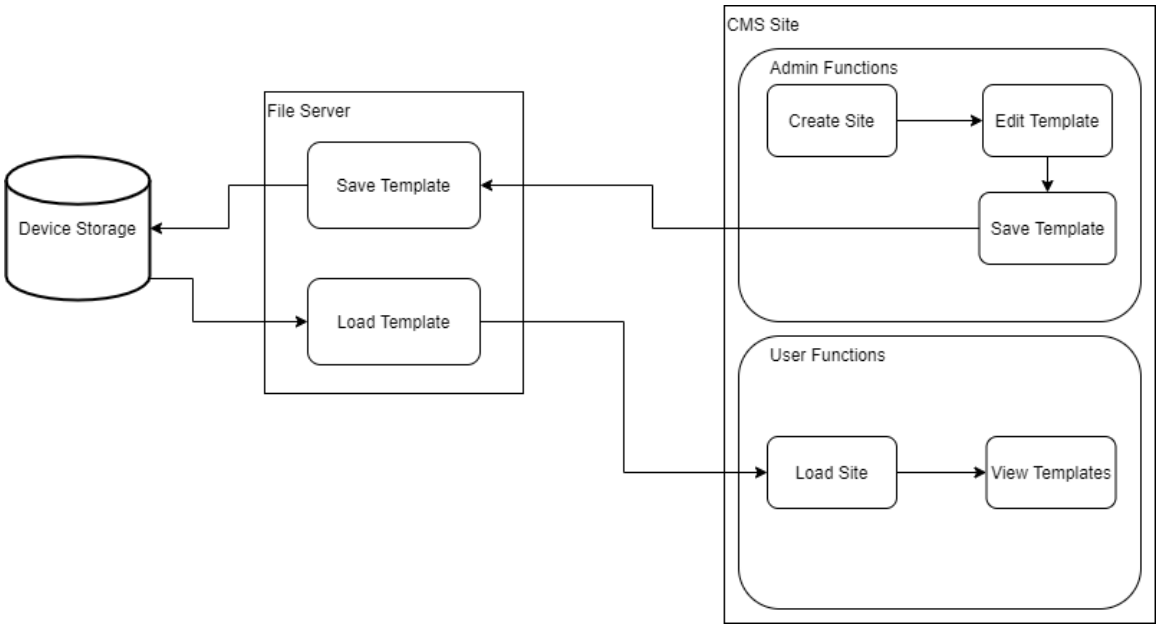
For our development process, we will be following the Kanban system for development. We decided that since we are all working from home and it is difficult to find meeting times where all team members could be present, the best development process would be Kanban. Since we are using Github for source control, we are using the project feature in Github to track our progress and as our Kanban board. Team members will pick up cards that have assigned issues and will move the cards between columns as work on the issue.

3.7 Design Plan

Use Cases

- Admin: Create a site
- Admin: Edit templates
- Admin: Save modified templates
- User: Load site
- User: View Templates

System Block Diagram



4 Testing

4.1 Unit Testing

File Server

The logic behind the endpoints for our file server will need unit testing. We will need to make sure that files are being saved and retrieved correctly from the file system. We will also use tests to ensure that the versioning system is working correctly. These tests also need to be in place for regression testing as changes are made to the file server logic. This will be done with Supertest and Jest.

React Application

The React Application will need unit tests as well. Buildertrend uses Jest and Enzyme for their React unit tests, so we will most likely use Jest and Enzyme for our project. This will ensure our pages render the information correctly and take in user input correctly.

4.2 Interface Testing

Interface between File Server and React App

The only interface testing required will be between our file server and the React Application. To test retrieving from and saving files to our file server, we will be using Postman. Since our file server takes json content and saves it to a file, we just need to send a post request with sample json content in the body to test that the file is created and saved. We can then use get requests to test that the file was saved with the correct format and name.

We are not planning on testing the React Application by itself without already having the file server working properly. This would have required implementing a file system on the front-end side of the project as well, and we decided that the benefits of doing things would not be enough to justify the amount of work required to implement it. As we are connecting to the endpoints in the web application, we will have to make sure that we are getting and saving the files manually.

4.3 Acceptance Testing

Acceptance testing for this project will take the form of 3 milestone demos each semester. During these demos, we will present our work to our project manager, full time developer, and members of the Buildertrend architecture team. If they are satisfied with our implementation of the

milestone, we will be able to begin work on the next milestone. If they find issues with it or think that our implementation has not fulfilled the requirements, we will have to spend time revising our implementation and demo again during the next demo.

4.4 Results

Milestone 1 Acceptance Tests

We have already demoed the file server and basic file server to web application communication to our client. We were successful in meeting the requirements for the first milestone and even meeting some requirements for the next milestone. Our file server was able to save files after sample content was passed from the web application, and we were then able to retrieve that file from the file server on the web application. It was not required for the web application to be a React with TypeScript application, but we had decided to get ahead and create the application using React and TypeScript. The only other thing they mentioned was that they would not have implemented the versioning system the way that we did, but our implementation still worked.

Through this demo, we gained a better understanding of what our client was expecting for this project and demonstrated that we understood the project requirements. Since we most likely will not have to revise the file server, we do not expect significant redesigns of the file server, but we may revise the versioning system if needed. We will continue to build on the React application so that for the next demo, we can demonstrate the basic functionality of the content management system.

Milestone 1 Unit Tests

Unit testing for milestone 1 was more challenging than we initially expected. Although we had experience writing unit tests for Node, we had not written tests related to file management. Some of the unit tests assumed that all machines running the server had the same files, which was true at first, but those files quickly changed as we added more files. We did not have enough time to refactor the tests, but will need to fix them before we pass the project off to Buildertrend.

Unit testing the React application was a challenge as well. Even some of the team members who had worked with React before had limited experience unit testing React applications. It did not help that we ran out of time to work on the tests. Several issues were merged into master without having any testing done since we needed the features for our demo. Going forward, we are going to need to spend more time looking into React unit tests and making sure we have good code cover

5 Implementation

Describe any (preliminary) implementation plan for the next semester for your proposed design in 3.3.

6 Closing Material

6.1 Conclusion

Summarize the work you have done so far. Briefly re-iterate your goals. Then, re-iterate the best plan of action (or solution) to achieving your goals and indicate why this surpasses all other possible solutions tested.

6.2 References

List technical references and related work / market survey references. Do professional citation style (ex. IEEE).

[1] Craig, D., 2015. *5 Reasons Not To Use Wix For Your Website – A Brief Wix Review*. [online] Fully Charged Media. Available at: <https://www.fullychargedmedia.com/online-presence-fundamentals/5-reasons-not-to-use-wix-for-your-website-a-brief-wix-review/> [Accessed 23 October 2020].

[2] Garcia, J., 2020. Wix Vs Squarespace – Squaring Up For A Good Fight. [online] WebsiteToolTester. Available at: <https://www.websitetooltester.com/en/blog/wix-vs-squarespace/> [Accessed 23 October 2020].

[3] Tovey, H., 2018. *6 Reasons Why You Shouldn't Use Squarespace - Heather Tovey*. [online] Heather Tovey. Available at: <https://heathertovey.com/blog/why-you-shouldnt-use-squarespace/> [Accessed 23 October 2020].

6.3 Appendices

Any additional information that would be helpful to the evaluation of your design document.

If you have any large graphs, tables, or similar data that does not directly pertain to the problem but helps support it, include it here. This would also be a good area to include hardware/software manuals used. May include CAD files, circuit schematics, layout etc,. PCB testing issues etc., Software bugs etc.