WASHINGTON STATE UNIVERSITY
VANCOUVER

CS 320 Course Project Final Report

for

Collaborator

Prepared by

Group Name: Team Collaborator

Brian Koga 01152937 brian.koga@wsu.edu
Kyle Trimbo 11395503 Kyle.trimbo@wsu.edu

Date: December 12, 2019

Contents

1 Introduction

Collaborator is a web application that helps find team members for software developers and hobbyists. It has features that cultivate this idea such as profile pages for users, project pages to help present ideas, as well as an advertisement section to help find projects or team members. This document details where the project is at after the requirements and design phases have been completed and the implementation phase has begun. This document includes information about the design and implementation of the project, as well as imformation about testing around the requirements.

1.1 Project Overview

Collaborator is a web application with features that helps software hobbyists or developers find team members to complete projects. It can be difficult to find team members with differing skills and values to complete a team, so Collaborator attempts to fulfill this need. It purposefully aims for developers that are not looking for a job but rather projects that would interest them in their free time as not every project and team will be able to compensate them with monetary value.

The goal of the web application is to create a hub for software developers to find others who have similar interests and to be able to collaborate and finish software projects. However, since there are plenty of other websites focused on finding jobs in the industry, this is more focused on personal studies and presenting one's work to find potential collaborators.

1.2 Definitions, Acronyms and Abbreviations

CSS: Cascading Style Sheets
HTML: Hypertext Markup Language
IDE: Integrated Development Environment
JS: Javascript
JSX: Javascript XML – An extension that allows the use of XML-like tags in JS
React: A javascript library designed to make it easier to create responsive web pages
SDD: Software Design Document
SRS: Software Requirements Specification
UI: User Interface

1.3 References and Acknowledgments

The only documents referenced are the SRS created for milestone 1, and the SDD created for milestone 2. Both are available for viewing on the Github project page.

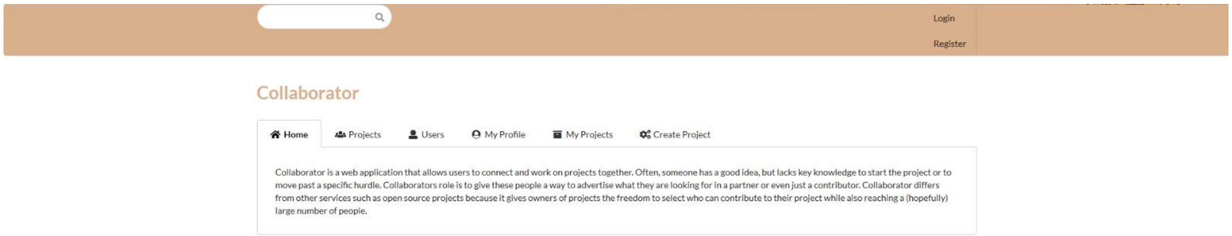
2 Design

2.1 System Modeling

Our implementation strictly follows the design document (SDS) from milestone 2.

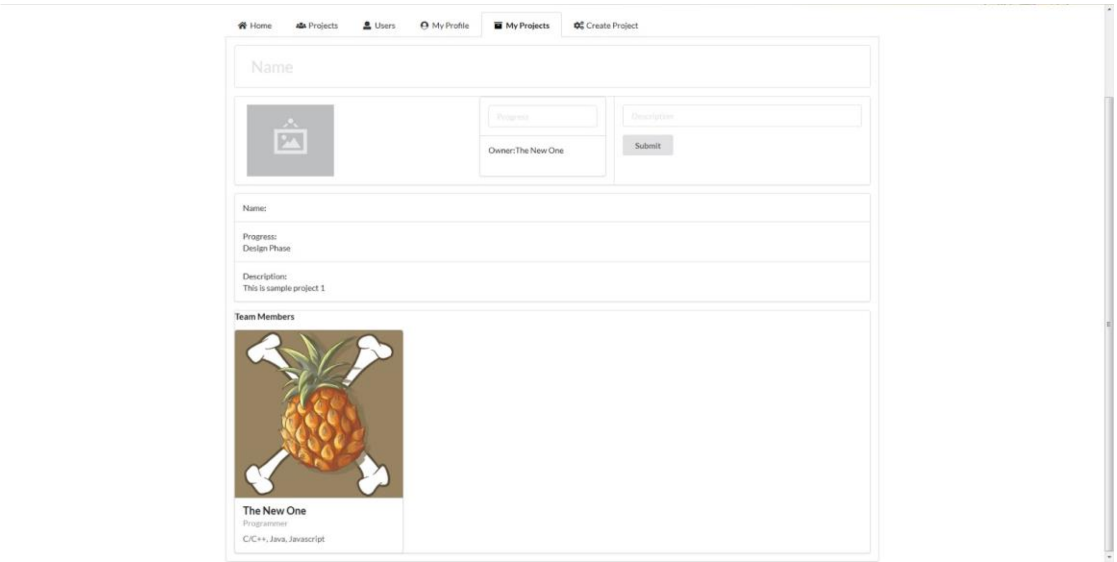
2.2 Interface Design

The picture below is simply the landing page describing the purpose of the web application.



Collaborator is a web application that allows users to connect and work on projects together. Often, someone has a good idea, but lacks key knowledge to start the project or to move past a specific hurdle. Collaborators role is to give these people a way to advertise what they are looking for in a partner or even just a contributor. Collaborator differs from other services such as open source projects because it gives owners of projects the freedom to select who can contribute to their project while also reaching a (hopefully) large number of people.

Below shows the Project edit page with the forms to edit the name, progress, as well as the description. It also displays the team members using the semantic ui card system.



Owner:The New One

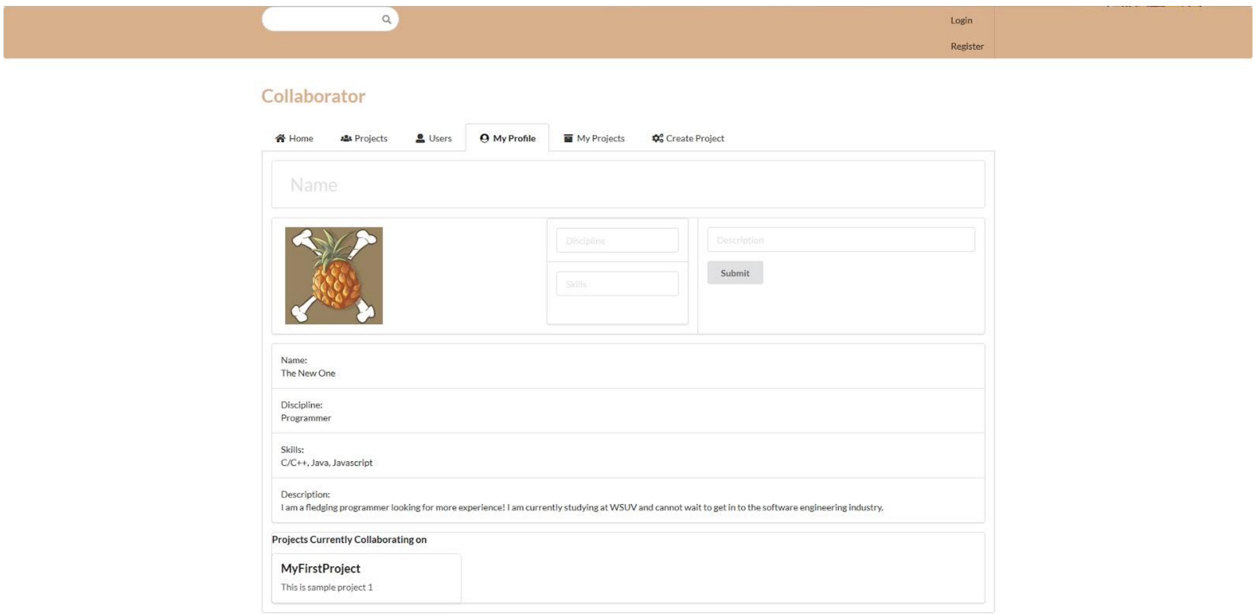
Name:

Progress:
Design Phase

Description:
This is sample project 1

The New One
Programmer
C/C++, Java, Javascript

This is the edit profile page. There are multiple forms to edit name, discipline, skills, as well as description. If you hover over the image it previews the card generated from the details.



Name:
The New One

Discipline:
Programmer

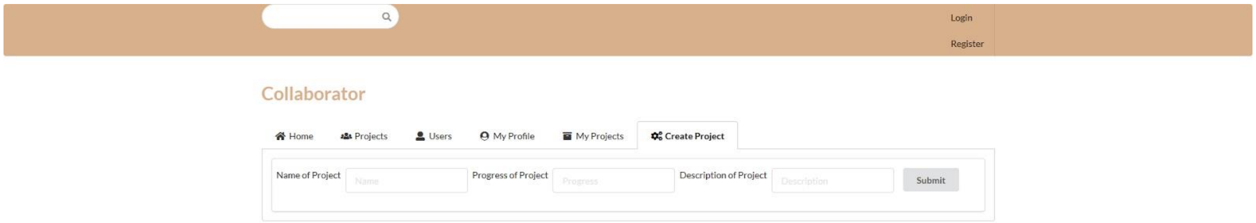
Skills:
C/C++, Java, Javascript

Description:
I am a fledging programmer looking for more experience! I am currently studying at WSUV and cannot wait to get in to the software engineering industry.

Projects Currently Collaborating on

MyFirstProject
This is sample project 1

Lastly this image shows the create a project form. Inputting data in each one will generate a project that will be displayed on the sample profile and under the projects list.



3 Implementation

3.1 Development Environment

The main development environment used in project was the IntelliJ IDE. As it is a web application some HTML and CSS was necessary, but the bulk of the application was built using the React JS library and the Semantic React UI library that works with it. While using React, the JSX syntax was used to simplify some of the React code. Other libraries used include Lodash, which helps with the manipulation of objects and arrays as well as faker which creates fake data to test the search bar.

3.2 Task Distribution

In general the workload was shared among the two group members, but Kyle focused more on the UI and visual elements while Brian focused more on the behavior of the application.

3.3 Challenges

The biggest challenge faced was that neither member of the team had ever worked with the React JS library before. It is a very useful library and allows for the creation of dynamic webpages, but it can be a little confusing to new users and it took some time to become familiar with it.

4 Testing

4.1 Testing Plan

We did not have an explicitly spelled out plan for testing as we set up dummy data and tested the functions while they were being produced for each of the following: users list page, project lists page, profile and project edit pages, create project page and lastly the search bar.

But our general testing plan was the following steps:

1. The first section tested was that the tab menu worked as it was supposed to.
2. The next thing tested was the users page, specifically that the page displays a list of users.
3. The next page to be tested was the project page, with similar functionality to the user page.
4. After the basic functionality of the pages was established, we began testing the parts that changed/added data. The first one being editing projects or profiles.
5. Then the create a project functionality was tested, specifically that creating a project added it to the list of projects on the projects page and added it to the list of user specific projects on the user page.

6. The last thing tested was the search bar on the top menu, simply that it functioned properly.

4.2 Tests for Functional Requirements

Users list: After setting up the tabular menu, applied a sample array of users to see if the semantic ui card system would be applied and listed. After changing the class' variable names to match the card system it worked and displayed as expected.

Project list: This was exactly the same as above, however, to test the dynamic nature of it, the create a project page adds to the array of the sample projects and it was successfully applied.

Profile and Project Edit: The functionality of these two are exactly the same, however, the profile one was created and tested first. Testing the event functions was difficult and required digging in the browser console and logs to find out they weren't being defined at all. Also the variables being presented on the page gave feedback and confirmation it was working as intended.

Create a Project: This follows the same functionality as the edit pages except it delivered a payload of a sample project class. This was tested by adding to the sample projects array as well as another to make sure they were being applied and displayed.

Search Bar: Since we did not have time to create a function to test searches with our own data, we utilized the faker library to create fake data that would display items if it was found.

4.3 Tests for Non-functional Requirements

Many of the non-functional requirements were ones that didn't require a lot of testing, such as how certain pages should be displayed, the prominence of certain features, etc. Additionally, due to time constraints, some of the requirements specified in the SRS weren't able to be tested as there was no functionality to test. These were mostly the ones related to security such as editing permissions and leaving feedback.

Requirements that could be tested included:

1. Portability: Test that the application works on Google Chrome, Firefox and IE. Passed.
2. Search reliability: Test that the search bar returns correct results 99% of the time. This was done using the faker library that provided lots of different data every time the application was started, so that potential edge cases were tested. Passed
3. Update time: Test that the search bar and application navigation does not take too long. There wasn't much to test as this was one of the reasons React was used, as it only updates the parts of the DOM that need to be updated, transition times are nearly instantaneous, certainly less than 1 second. Passed.

4.4 Hardware and Software Requirements

The only hardware requirements was a computer with a browser, internet connection as well as an IDE that would allow editing of the source code. The tests were not hardware intensive at all.

5 Analysis

Kyle approximately dedicated over 30 hours total on this project and the majority of those hours were spent on researching and getting the prototype to work for the last milestone. For the first two milestones approximately 3 to 4 hours were spent on each, creating the UML and helping write the SRS. However, the third milestone for the actual web application over 20 hours was dedicated to get the basic features such as editing projects and profiles working. Working with react was a daunting task as it did not have a familiar syntax to most languages.

Brian spent somewhere between 25 and 30 hours on this project across the three milestones. The first two milestones, creating the SRS and the Design document, took around 4 hours each, but the majority of time spent was on actually creating the web application. The difficulty with this milestone was that React was not very intuitive and took a lot of research, viewing practice projects and even following a tutorial on how to create a practice project

6 Conclusion

The first thing we learned was how confusing react can be. Dealing with page states and trying to figure out why a piece of code wasn’t projecting what it was supposed to as well as apply data to variables was overwhelming. After plenty of hours of research and trial and error, we were able to get the most basic features working including profile and project editing, creating projects, as well as a functional search bar. However, since we did not have the time nor knowledge of setting up databases it is merely a prototype and had very little functionality and utilized dummy data.

Communication is a major asset; if we communicated more efficiently, we would have prevented much refactoring of the code. There were multiple times where we overwrote each other’s code and our intent wasn’t explicitly clear due to the lack of comments and or direct communication. The next time either one of us works on a large project this will be a more imperative task.

Appendix A - Group Log

The group met every Tuesday and Thursday for several weeks, starting October 8th. After the second deliverable had been turned in and most of the details for the project were worked out, the majority of communication was virtual via text message or discord.