Piqued

Team 5

Jose Avila, Team Lead, javila6@mail.sfsu.edu
Andy Shi, Frontend Lead, ashi2@mail.sfsu.edu
Leo Saeteurn, Backend Lead, lsaeteurn@mail.sfsu.edu
Nishit Pachchigar, Github Master, npachchigar@mail.sfsu.edu
Joshua Hayes, Database Master, jhayes10@sfsu.edu
Gautami Gudla, Document Editor, gkollolusrinivasa@sfsu.edu

Milestone 4
May 18th, 2023

History Table

Version	Submission Date
M4V1	May 18, 2023
M3V2	May 18, 2023
M3V1	April 27, 2023
M2V2	April 19, 2023
M2V1	March 30, 2023
M1V2	April 16, 2023
M1V1	March 5, 2023

Table Of Contents

1.   Product Summary

1.1.   **Name Of The product**:

Piqued

1.2.   **Final P1 Functions:**

1.1.1.   All Users
- 1.1.1.1.   All users shall be able to view the website's information
- 1.1.1.2.   All users shall be able to contact the site moderators with questions
- 1.1.1.3.   All users shall be able to view the privacy policy
- 1.1.1.4.   All users shall be able to view the terms and conditions

1.1.2.   Guest Users
- 1.1.2.1.   Guest users shall be able to create an account
- 1.1.2.2.   Guest users shall be able to use a unique username to create an account
- 1.1.2.3.   Guest users shall be able to only view public posts
- 1.1.2.4.   Guest users shall be able to view the contents of public posts
- 1.1.2.5.   Guest users shall be able to view a profile's public information
- 1.1.2.6.   Guest users shall be able to search for public posts
- 1.1.2.7.   Guest users shall be able to discover other public users
- 1.1.2.8.   Guest users shall be able to view comments on public posts
- 1.1.2.9.   Guest users shall be able to view reactions on public posts

1.1.3.   Registered Users
- 1.1.3.1.   Registered users shall be able to log in
- 1.1.3.2.   Registered users shall be able to log out
- 1.1.3.3.   Registered users shall be able to sign in with either their email or username.
- 1.1.3.4.   Registered users shall be able to add a profile picture
- 1.1.3.5.   Registered users shall be able to edit their profiles
  - 1.1.3.5.1.   Registered users shall be able to change their profile picture
  - 1.1.3.5.2.   Registered users shall be able to change their displayed username
  - 1.1.3.5.3.   Registered users shall be able to change their email address

       1.1.3.6.   Registered users shall be able to search for other users by username
       1.1.3.7.   Registered users shall be able to create blog posts.
       1.1.3.8.   Registered users shall be able to add photos on blog posts.
       1.1.3.9.   Registered users shall be able to search for posts.
       1.1.3.10.   Registered users shall be able to delete their posts.
       1.1.3.11.   Registered users shall be able to add comments on post.
       1.1.3.12.   Registered users shall be able to delete their own comments.
       1.1.3.13.   Registered users shall be able to like/dislike to other people's comments

1.1.4.   Administrator Users

       1.1.4.1.   Administrator users shall be able to create authorized users on account

1.1.5.   Site moderators - creators(us)

       1.1.5.1.   Site moderators shall be able to delete posts/articles
       1.1.5.2.   Site moderators shall be able to ban accounts
       1.1.5.3.   Site moderators shall be able to temporarily block accounts for review
       1.1.5.4.   Site moderators shall be able to edit the layout of the platform
       1.1.5.5.   Site moderators shall be able to add additional features
       1.1.5.6.   Site moderators shall be able to answer users' questions and concerns
       1.1.5.7.   Site moderators shall be able to create an account
       1.1.5.8.   Site moderators shall be able to post on the platform's main page
       1.1.5.9.   Site moderators shall be able to make site announcements
       1.1.5.10.   Site moderators shall be able to update website
       1.1.5.11.   Site moderators shall be able to temporarily suspend the website for maintenance
       1.1.5.12.   Site moderators shall be able to reinstate accounts
       1.1.5.13.   Site moderators shall be able to review accounts

1.1.6.   User's Profile Page

       1.1.6.1.   User's profile page shall have the user's picture
       1.1.6.2.   User's profile page shall have the user's username
       1.1.6.3.   User's profile page shall have published public posts
       1.1.6.4.   User's profile page shall have a biography
       1.1.6.5.   User's profile page shall have contact information that are made public by user

1.1.7.   All pages

       1.1.7.1.   All pages shall have a navigation bar/header
       1.1.7.2.   All pages shall have a menu with links to informational pages

1.1.7.3. All pages shall have our company's logo

1.1.8. User's Logged-In Interface

    1.1.8.1. User's logged-in interface shall have a link to the user's account management page in the navigation bar

1.1.9. Blog Posts

    1.1.9.1. Blog posts shall contain an original author

    1.1.9.2. Blog posts shall contain a date and time of creation

    1.1.9.3. Blog posts shall contain a title

    1.1.9.4. Blog posts shall contain a photo(s)

    1.1.9.5. Blog posts shall contain a category(ies)

    1.1.9.6. Blog posts shall contain a tag(s)/hashtag(s)

    1.1.9.7. Blog posts shall contain a comment sections with comments from other users

1.1.10. Exploration Newsfeed

    1.1.10.1. Exploration newsfeed shall contain random public posts

    1.1.10.2. Exploration newsfeed shall have a search bar

    1.1.10.3. Exploration newsfeed shall have blog posts that can be clicked on to view full posts

1.1.11. Search Bar

    1.1.11.1. Search bar shall have a text box to search up public blog posts by categories, tags, and users

1.1.12. Navigation Bar

    1.1.12.1. Navigation bar shall have a link to the user's profile when logged in

    1.1.12.2. Navigation bar shall have a link to the home page

    1.1.12.3. Navigation bar shall have a link to the user's account management page when logged in

    1.1.12.4. Navigation bar shall have a post button

    1.1.12.5. Navigation bar shall have a log out button when logged in

1.1.13. Menu (Footer)

    1.1.13.1. Menu shall have a link to the company's information page

    1.1.13.2. Menu shall have a link to contact the company by emailing

    1.1.13.3. Menu shall have the company's contact information

    1.1.13.4. Menu shall have a link to "about us" page

    1.1.13.5. Menu shall have a copyright logo

    1.1.13.6. Menu shall have a link to the terms and conditions

    1.1.13.7. Menu shall have a link to the privacy policy

1.1.14. Chat Box

    1.1.14.1. Chat box shall contain user's username for both sender and recipient(s)

1.3. **What is Unique In Our Product:**

Piqued is a unique blogging website that offers a safe and social platform for individuals to express themselves while sharing common interests. Unlike other blogging sites, Piqued allows users to interact with their friends and other like-minded individuals, creating a sense of community around shared interests. The platform is perfect for both extroverted and introverted individuals, as it offers a safe space to share without fear of scrutiny. Piqued also allows users to find new interests and experiences, making it a great platform for exploration and self-discovery. The ability to connect and build a community of like-minded individuals sets Piqued apart from other blogging sites and makes it an excellent choice for those looking to express themselves and connect with others.

What makes Piqued truly special is the ability to connect with individuals who share the same interests. Through engaging and detailed blog posts, you can delve into your favorite topics such as travel experiences, culinary adventures, life hacks, and more. Discover new ideas, gain inspiration, and add exciting activities to your bucket list, all while building a vibrant community of like-minded individuals.

1.4. **URL Of the Product:**

http://ec2-18-224-63-127.us-east-2.compute.amazonaws.com:4000

2.   <u>Usability Test Plan</u>

Functions	Task	Effectiveness	Efficiency	Satisfaction	Errors	Comments
Create	Create a post	90%	95%	95%	No errors	Intuitively accessible and effortlessly discoverable.
Search	Search a post	40%	75%	50%	Nothing popped up when search	The search function provided a user-friendly interface, allowing seamless input, although it failed to generate any relevant results.
Write Comments/ Leave a reaction	Leaving feedback	80%	70%	85%	No errors	Although the comment section was initially challenging to locate, it proved to be user-friendly and intuitive once discovered.
Editing profile	Updating user credentials	95%	95%	90%	Users are able to use invalid data or avatar	Unable to update user avatar
Chat	Chat with chatbox	90%	90%	70%	Messages aren't loaded in real time	Site must be reloaded

Create Function:

Test objectives:
The objective of this test is to ensure that the users can effortlessly locate the "Create a Post" button and seamlessly navigate the process of creating a post without any confusion.
Test description:
- System setup: The user is registered in db and wants to create a post. They are able to click links and create a post through the UI.

- Starting point: Home page
- Intended Users: Users who are looking to create a new article

URL: http://ec2-18-224-63-127.us-east-2.compute.amazonaws.com:4000/#/post

Search Function:

Test objectives:

The objective of this test function is to test the user experience when searching for a variety of items such as searching for another user, category, post, or hashtag. This is being tested to validate that the correct results are being sent as a response to the user. Users can send search requests for various data on the site. The user may send invalid data or data which does not exist on the site's database. In this case, it would be expected for the function to return some useful suggestions and also notify the user that their search did not yield any exact matches. In the case that a user searches for an entity that exists in the database, all relevant results should be returned.

Test description:
- System setup: Entities exist within the database that can potentially match their search. When a search is entered, relevant results should be returned.
- Starting point: Home page
- Intended Users: Users searching for a topic, keyword, or other user

URL: http://ec2-18-224-63-127.us-east-2.compute.amazonaws.com:4000/

Writing comments/reaction Function:

Test objectives:

The objective of this test function is to test the site's usability for when a user would like to leave feedback for a post. Feedback can be done in different ways such as leaving a comment or leaving a reaction. This function is being tested because it is a key feature for the social functionality of the site. This is being tested also to ensure that users can only leave one form of a reaction but also leave many comments. Input validation should also be tested to verify that comments of varying length from invalid input all the way to full responses.

Test description:
- System setup: A user is registered within the db and posts exist. The user is looking to leave a comment on a post. They are able to click a post and leave a reaction or comment.
- Starting point: Home page
- Intended Users: Users that want to leave feedback for an article

URL: http://ec2-18-224-63-127.us-east-2.compute.amazonaws.com:4000/#/article

Editing Profile Function:

Test objectives:

The objective of this function is to measure the usability of the site when a user would like to edit their profile. This is being tested to measure how many steps and how long it takes a registered

user to reach their account details page and edit their personal information. This function is also being tested to make sure that 1) The user is shown the correct personal information when accessing their profile. 2)This is also another area where input validation must be checked. 3) Testing to ensure that non-functional security requirements such as ensuring that password validation are still upheld and are updated on the backend.

Test description:
- System setup: The user is registered and is looking to edit their personal information. The link to edit account details exist and data should be modifiable.
- Starting point: Home page
- Intended Users: Registered users looking to update their user information

URL: http://ec2-18-224-63-127.us-east-2.compute.amazonaws.com:4000/#/account

Chat Function:

Test objectives:
The objective of this test function is to test the user experience when chatting with other registered users. The function is being tested to make sure that messages are sent in real time and are valid input.
Test description:
- System setup: Users exist in the db and are looking to communicate with other users. Chat and send button works as well as sending/receiving messages in realtime
- Starting point: Home page
- Intended Users: Registered users that want to chat with other registered users.

URL: http://ec2-18-224-63-127.us-east-2.compute.amazonaws.com:4000/

3.   QA Test Plan

3.1.   **Test objectives:** The objective of the quality assurance test is to verify that the functionality of the software builds, executes, and runs according to the specifications of the non-functional requirements listed. The test should verify and identify any bugs or defects so that the developers of the software can debug and fix the issues.

3.2.   **HW and SW setup:**
   3.2.1.   The testing requires either a laptop or a computer with internet access and a website browser.
   Website link on AWS server:
   http://ec2-18-224-63-127.us-east-2.compute.amazonaws.com:4000/

3.3.   **Feature to be tested are non-functional requirements:**

   3.3.1.   **QA Test Plan 1:** Password Encryption
      3.3.1.1.   **Case 1:** Enter your information on the signup page. Check that the database does not show the password you entered
      3.3.1.2.   **Case 2:** Enter your username and password on the login page. Check that the database does not show the password you entered.
      3.3.1.3.   **Case 3:** Enter new information with a new password. Check that the database does not show the password you entered.
   3.3.2.   **Test input:**
      3.3.2.1.   **Case 1:** Unique password that contains alphanumeric characters with at least one symbol and minimum length of eight characters.
      3.3.2.2.   **Case 2:** New unique password that contains alphanumeric characters with at least one symbol and minimum length of eight characters.
      3.3.2.3.   **Case 3:** New unique password that is not the same as case 1 and 2.
   3.3.3.   **Expected test results:**
      3.3.3.1.   **Case 1:** pass
      3.3.3.2.   **Case 2:** pass
      3.3.3.3.   **Case 3:** pass

   3.3.4.   **QA Test Plan 2:** Unique Username
      3.3.4.1.   **Case 1:** Enter your information on the signup page. Check that the database does not contain the same username as the one you created.

3.3.4.2. **Case 2:** Enter your information on the signup page. Check that there are no error messages mentioning a duplicate username.

3.3.4.3. **Case 3:** Enter username and password in the login page. Check to make sure there is no error message that mentions a duplicate username.

3.3.5. **Test input:**

3.3.5.1. **Case 1:** Unique username

3.3.5.2. **Case 2:** Unique username

3.3.5.3. **Case 3:** Unique username

3.3.6. **Expected test results:**

3.3.6.1. **Case 1:** pass, no error

3.3.6.2. **Case 2:** pass, no error

3.3.6.3. **Case 3:** pass, no error

3.3.7. **QA Test Plan 3:** Cookie for logging in

3.3.7.1. **Case 1:** Enter your username and password in the login page. Go into the home page. Check to make sure you are still logged in by checking for your username at the top right of the page.

3.3.7.2. **Case 2:** Enter your username and password in the login page. Go into the home page then to the search page. Check to make sure you are still logged in by checking for your username at the top right of the page.

3.3.7.3. **Case 3:** Enter your username and password in the login page. Go into the search page then to the account page. Check to make sure you are still logged in by checking for your username at the top right of the page.

3.3.8. **Test input:**

3.3.8.1. **Case 1:** Unique username and password

3.3.8.2. **Case 2:** Unique username and password

3.3.8.3. **Case 3:** Unique username and password

3.3.9. **Expected test results:**

3.3.9.1. **Case 1:** pass

3.3.9.2. **Case 2:** pass

3.3.9.3. **Case 3:** pass

3.3.10. **QA Test Plan 4:** UI Responsiveness

3.3.10.1. **Case 1:** Open up your preferred browser and login into the website with your username and password. Minimize the page, resize it by hovering over the edge of the browser and adjusting the size in and out. Check to make sure that there are no glitches with the display.

3.3.10.2.   **Case 2:** Go to the website without logging in. Go into the search page and adjust the size of the website window. Check to make sure that there are no glitches with the display.

3.3.10.3.   **Case 3:** Go to the home page without logging in. Adjust the window size of the browser. Check to make sure the display is not distorted as you adjust the window.

3.3.11.   **Test input:**

3.3.11.1.   **Case 1:** Username and password

3.3.11.2.   **Case 2:** No input, adjust browser window.

3.3.11.3.   **Case 3:** No input, adjust browser window.

3.3.12.   **Expected test results:**

3.3.12.1.   **Case 1:** pass, no error

3.3.12.2.   **Case 2:** pass, no error

3.3.12.3.   **Case 3:** pass, no error

3.3.13.   **QA Test Plan 5:** Website navigation through each page

3.3.13.1.   **Case 1:** Log into your account. Click on the account page by clicking on the menu bar then the account management option. Check to make sure you were directed to the account management page.

3.3.13.2.   **Case 2:** Create an account in the signup page by entering the required information. Search a post by entering a keyword in the search bar, such as food. Check to make sure the page lists blog posts based on the word you search.

3.3.13.3.   **Case 3:** Log into your account in the login page with your username and password. Click on one of the blog posts that shows up in the home page. Check to make sure you were directed to the full blog post.

3.3.14.   **Test input:**

3.3.14.1.   **Case 1:** Username and password

3.3.14.2.   **Case 2:** Full name, username, email, and password

3.3.14.3.   **Case 3:** Username and password

3.3.15.   **Expected test results:**

3.3.15.1.   **Case 1:** pass, no error

3.3.15.2.   **Case 2:** pass, no error

3.3.15.3.   **Case 3:** pass, no error

Test #	Test Title	Test Description	Test Input	Expected correct output	Google Chrome Test Results	Safari Test Results
1	Password encryption	Ensure that inputted user passwords are encrypted when stored in the database.	Fullname, username, email, password	Passwords are encrypted with a hash which produces random characters which get stored into the database.	PASS	PASS
2	Unique username	Ensure that the username input in the frontend displays an error if a username is already taken	Fullname, username, email, password	There are no duplicate usernames shown in the database.	PASS	PASS
3	Cookie for logging in	Make sure that session cookies are stored and persist across browser sessions.	Fullname, username, email, password	User is able to navigate between pages and the browser does not log the user out.	PASS	PASS
4	UI Responsiveness	Ensure that that the page reflows and resizes content based on the available window/screen area	Username and password	The web browser window should be able to be adjusted and the website should respond correctly without displaying any distortion.	PASS	PASS
5	Website navigation through each page	Make sure that users can visit each page given the correct authentication conditions and routing.	Fullname, username, email, password	All users shall be able to go from one page to another, no matter the page they're on, and the pages should be redirected correctly based on the page they intended to go into.	PASS	PASS

4. Code Review

4.1. Code Style :

4.1.1. We are using svelte for our frontend coding structure. It is very easy to use as we can create each individual component and attach it with other components.

4.1.2. We created individual pages using svelte which involves HTML and CSS for styling and called individual components like buttons, cards onto the page and added contents inside it. As shown in the below screenshot, we created Searchbar and Card components which have their own styling and structure. We imported them in other pages and those styling will be applied to the contents automatically.

```
        <Searchbar placeholder={"Browse"}/>
</div>
<h1>Highlights</h1>
<div class="card-container">
    <Card {...cardData.card1} />
    <Card {...cardData.card2} />
    <Card {...cardData.card3} />
</div>
```

4.1.3. We are using Node and Express for backend, which renders the data from the database and passes it along to the Frontend.

4.2. Team 5 Code review by Saru of Team 4 - reviewed users.js file

4.2.1. Naming Convention :
   ● The standard naming convention has been followed.

4.2.2. Code Header :
   ● The name of the file is descriptive to reflect the content inside the file.

4.2.3. Commenting :
   ● Comments are missing. Adding comments is recommended as it helps understand code better.

4.2.4.   Limited Use of Global Variables and Functions:
● There is limited use of global variables and functions indicating good coding practice.

4.2.5.   Indentation:
● The indentations are placed properly increasing readability of the code.

4.2.6.   Exception Handling:
● Every function contains an exception handler which is great.

4.2.7.   Statements:
● The statements are clear and comprehensive.

4.3.   **Code Review By Nishit Pachchigar and Gautami Gudla - Reviewed Signup.svelte**

4.3.1.   Naming Convention:
● The standard naming convention is followed well which makes it easy to understand the code.

4.3.2.   Code Header:
● The minimal code header is not adequate. It is important to understand the purpose of the code function, which is why minimal code header is required.

```
        }
    }

    function validateUsername() {
        usernameElem.value = usernameElem.value.trim();
        const regex = /[^A-Za-z0-9]/;
        let test1 = usernameElem.value.trim() != "";
        let test2 = !usernameElem.value.trim().includes(' ');
        let test3 = !regex.test(usernameElem.value);

        if (test1 && test2 && test3) {
            usernameValid = true;
        }
```

4.3.3.   Commenting
● Necessary amount of comments not present in the file. Comment helps other team members to understand the functionality of the code.

```
function validateName() {
    const regex = /^[^!@#$%^&*()_\-\+\={}\[\]\|\\`~<>?]+$/
    fullnameElem.value = fullnameElem.value.trim();
    let test1 = fullnameElem.value != "";
    let test2 = regex.test(fullnameElem.value)
    console.log("Full name", test1, test2)
    if (test1 && test2) {
        fullnameValid = true;
    }
    else if (!test1) {
        fullnameValid = false;
        fullnameError = "Full name must not be blank."
        showFullnameError = true;
    }
    else if (!test2) {
        fullnameValid = false;
        fullnameError = "Full name must not contain special characters."
        showFullnameError = true;
    }
    else {
        fullnameValid = false;
        fullnameError = "Full Name is invalid."
        showFullnameError = true;
    }
}
```

4.3.4.   Limited Use of Global Variables and Functions:
- Limited use of global variables and functions is present which serves the purpose of encapsulation. It is a good practice to make the code better.

4.3.5.   Indentation:
- The functions are guiding the users to put the correct input and incase of invalid prompt by the user, it will help the users to enter valid input by providing the proper indentation.

```
if (passwordElem.value == "") {
    passwordError = "Password cannot be empty";
} else {
    passwordError = "Password must meet the following conditions.";
}
```

4.3.6.   Exception Handling:
- Exception handling is present in the code in the form of if and else statements. It is making sure to handle any unexpected hurdle.

4.3.7.   Statements:
- The statements are following consistent styling and appropriate indication but only comments are missing for easy understanding of the code.
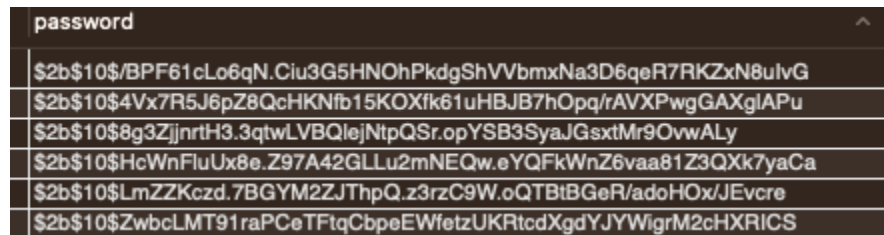
5. <u>Self-Check On Best Practices For Security :</u>

5.1. **List major assets you are protecting :**

5.1.1. We're protecting the user's private information such as logins and passwords.

5.1.2. We are restricting un-registered users to react and respond with comments. In order to do that, they need to create an account.

5.1.3. Users need to have an account in order to follow another user's account.

5.2. **Confirm that you encrypt PW in the DB :**

5.2.1. We have PW encrypted in the db using bcrypt using 10 salt rounds.

5.2.2. Here is a screenshot of PW encryption :


```
password                                                    ^

$2b$10$/BPF61cLo6qN.Ciu3G5HNOhPkdgShVVbmxNa3D6qeR7RKZxN8ulvG
$2b$10$4Vx7R5J6pZ8QcHKNfb15KOXfk61uHBJB7hOpq/rAVXPwgGAXglAPu
$2b$10$8g3ZjjnrtH3.3qtwLVBQlejNtpQSr.opYSB3SyaJGsxtMr9OvwALy
$2b$10$HcWnFluUx8e.Z97A42GLLu2mNEQw.eYQFkWnZ6vaa81Z3QXk7yaCa
$2b$10$LmZZKczd.7BGYM2ZJThpQ.z3rzC9W.oQTBtBGeR/adoHOx/JEvcre
$2b$10$ZwbcLMT91raPCeTFtqCbpeEWfetzUKRtcdXgdYJYWigrM2cHXRICS
```

5.3. **Confirm Input data validation :**

5.3.1. While creating an account, users should meet certain requirements like for the password and username without any space.

5.3.2. Users cannot have special characters for their username and the backend will also have a validation function to check if that username already exists or not. And the same thing will also be checked for the email.

5.3.3. Search box will parse the request into the backend if no results match then it will show suggestions.

6. Self-check: Adherence To Original Non-Functional Specs

6.1. Performance:
- 6.1.1. The website should be available 24/7 with a minimum of downtime for maintenance and updates.
  - 6.1.1.1. ISSUE: the AWS server is a free trial and we have all the instances running on one of the team's accounts. We keep having to restart the instances for the program to work on the site using the URL.
- 6.1.2. Scalability: The website should be able to handle a large number of users and posts without slowing down or crashing.
  - 6.1.2.1. ON TRACK

6.2. Security:
- 6.2.1. Passwords must be encrypted in the database.
  - 6.2.1.1. DONE
- 6.2.2. Each account should have a unique email.
  - 6.2.2.1. DONE
- 6.2.3. Each account should have a unique username.
  - 6.2.3.1. DONE
- 6.2.4. Logged in users must stay logged in while navigating unless they log out.
  - 6.2.4.1. DONE

6.3. Legal:
- 6.3.1. To complete registration, users must accept T&C.
  - 6.3.1.1. DONE
- 6.3.2. Privacy Policy.
  - 6.3.2.1. DONE

6.4. Compatibility:
- 6.4.1. The website should be compatible with a range of devices, browsers, and operating systems.
  - 6.4.1.1. ON TRACK
- 6.4.2. The UI should be responsive to all screen sizes and device form factors
  - 6.4.2.1. ON TRACK

6.5. Usability:
- 6.5.1. The website should be user-friendly, with a clear interface that is easy to navigate.
  - 6.5.1.1. DONE

6.6. Marketing:
- 6.6.1. The website will have a logo to represent the company.
  - 6.6.1.1. DONE

6.6.2.   The website should have a description to explain what the website is about.

      6.6.2.1.   ON TRACK

7.   List Of Contributions

Role	Name	Contributions	Score
Team Lead	Jose Avila	Delegated task, scheduled meetings, set reminders for the team, gave input about U/I, reviewed frontend and backend code, reviewed documents, worked on usability test plan, asked questions to the CTO on behalf of the team, and gave back feedback from CTO to the team.	
Front End Lead	Andy Shi	Coded the frontend UI, set up routes, scheduled meetings with front end members, created wireframes and components, helped with selecting our final P1s and collaborated with the back end.	9
Back End Lead	Leo Saeteurn	Schedule meetings with back end members, coded algorithms, helped with QA Test Plan, helped select final P1 requirements, and Adherence to original non -functional specs.	8
Database Master	Joshua Hayes	Created all database tables, set up routes, collected data and coded back end. Worked on the	9

		usability test plan, and self checked on best security.	
GitHub Master	Nishit Pachchigar	Helped Andy code the frontend, help with routing and creating components, participated in the code review, and helped Josh with the self check on best security. Help select finalP1s and kept Github up to date.	9
Document Editor	Gautami Gudla	Edited & updated the document,wrote the product summary, participated in the code review with Nishit, worked on the frontend, coded components, revised P1 requirements,gave us reminders, and proofread the document.	8