SW Engineering CSC648/848 Fall 2021 SFSU Tutors

Team No.7

Alekhya Gandu (Team Lead & Back End Lead): agandu@mail.sfsu.edu

William Lushbough: Github Master

Justin Diones: Front End Lead

Rui Qi Huang: Front End Developer

Rupak Khatri: Front End Developer

Mai Ra: Back End Developer

Milestone 4

Dec 18, 2021

	Date Submitted	Date Revised
Milestone 1	October 9, 2021	October 13, 2021
Milestone 2	October 27, 2021	November 4, 2021
Milestone 3	December 18, 2021	
Milestone 4	December 18, 2021	

Section 1: Product Summary

Our web app is a tutoring app designed for San Francisco State University students. It connects students with tutors across the world, allowing them to find the best tutors suited for their subject and timezone. The easy to use website is designed for all users even those unfamiliar with web apps to navigate it through easily. The product is in its late stages and is ready to be deployed after some finishing touches are made. Students and tutors are allowed to create their own personalized accounts to make it more user friendly. Students are able to send messages to the tutor and the tutor can receive it and make adjustments for the student. Tutors are also able to post documents and anything necessary for the students' success. We believe in the success of our students and that this app will be able to help students achieve the grade they want.

Name of Product: SFSU Tutors

Product URL: http://54.177.172.73:3000/

Itemized List: Priority 1

1) Unregistered User:
1.1) An unregistered user shall be able to search for a tutor according to the class subject or tutor name.
1.2) An unregistered user shall be able to create an account.

2) Registered User:
2.1) A registered user shall inherit all functions as Unregistered User plus the following below.
2.2) A registered user shall be able to be a tutor or user of tutoring service.
2.3) A registered user shall be able to log in/log out.
2.4) A registered user shall be able to delete his/her posts.
2.5) A registered user shall be able to post tutoring info.
2.6) A registered user shall be able access their profile information
2.7) A registered user shall be able to edit their profile information

3) Administrator:
3.1) An administrator shall be able to approve a new tutor before it goes live.

3.2) An administrator shall be able to view a list of tutors on the website.
3.3) An administrator shall be able to view comments posted by other registered users.
3.4) An administrator shall be able to delete an account of all registered users.
3.5) An administrator shall be able to remove a tutor review.
3.6) An administrator shall be able to validate the appropriate review and post it.

Section 2: Usability Test Plan

Objectives:

We are testing our search function to ensure complete usability for any user that uses our website. This is being tested to make sure that the test results comply with all functional requirements given such as displaying consistent search results, displaying the correct information of tutors that are searched for, displaying the correct number of total search results found, and that it is effective and easy for users to understand and use. We need to make sure that the search function works because it is the main function of our website. Without the search working there is no way for a user to look for and contact tutors since contacting tutors will be done on the search results page. To ensure that it is easy for users to use, we must make the search easily findable on the page as well as convey to the user how to use the search as this was something pointed out to us in our Milestone 3 check in. If there is no guidance given to the user on the search, they will not know what they are searching for. The dropdown menu along with placeholder text in the search bar should guide the user on how to effectively use the search function.

Test Background and Setup:

System Setup: To first set up the search function the server holding the database information of the tutors must first be set up and running. This database information is posted once a user applies to be a tutor and is accepted to become a tutor by an admin of the website. The search function gets this information from the database to display back to the user. The data to be posted and displayed back in the search are the tutor's name, profile picture, and the class they are tutoring. Both sides of the front end and back end need to work together in order to achieve this.

Starting Point: Since the Search bar is a part of our Navigation bar component that is visible and ready to use on any tab on the website, the user's starting point could be anywhere on the site. The Navigation bar is placed at the top of every tab and the Search bar can be found in the middle.

Intended Users: The users intended to use the search function are students who want to search for a tutor. They can type in the course they need tutoring in or the name of the tutor themself in order to find the tutor that they need. A user that wants to become a tutor can also use the search to see how the site functions before they apply to become a tutor. The search is the main function to find a tutor on the site so it is primarily for the students looking to become a tutor.

URL/What is being measured: http://54.177.172.73:3000/

The search function is being measured on a Likert scale in order to measure the usability of the search. We need to make sure that the system is fast and efficient, easy to comprehend, and not feel hard for any user to understand how it works. This is done in order to ensure our customers with ease of usability when using our website.

Usability Task Description:

Visit the site on any browser, the Navigation bar containing the Search bar should be visible at the top of every page. The user can click on the dropdown menu to filter search results by choosing either Tutors or Courses. By picking Tutors, the user can search for the tutor based on their name and by picking Courses the user can search based on the course they need a tutor for. If the user does not choose an option from the dropdown and continues with the search, the displayed results will pull from both Tutors and Courses. For example if a user searches for “c”, the information displayed will show all tutors that have a name that starts with a “C” and all courses containing “c”. To complete a search request the user can choose an option from the dropdown menu, type in a keyword into the search bar for what they are searching for, and click the Search button. After a search request is made the information will be displayed showing the tutor’s profile picture, first and last name, their email, the course they are tutoring, and a short description of the course. All information displayed on the search results should be consistent between the different dropdown filters, ie. searching by Tutors should not display more or less information than if the user searched by Courses.

Evaluation of Effectiveness: Is the information being shown relevant and consistent to what keywords I am searching.

Evaluation of Efficiency: Is the information being displayed in a timely manner so that the user does not have to wait a noticeable amount of time before search result data is displayed.

Evaluation of User Satisfaction: Is the information not only relevant but is displayed neatly and easy to understand for the user.

Section 3: QA test plan:

Test Objective: Checking that the search bar is able to return requests from the user and in a timely manner. The results should also be accurate and what the user intended.

HW and SW setup: http://54.177.172.73:3000/
Setup of HW: Deploy live version of the website to the cloud, access the website on the cloud from any device of the user's choosing that has internet browser capabilities.

Setup of SW: Have a working version of Google Chrome and Mozilla Firefox to access the live application. Two browsers are needed to test for discrepancies if any between the two browsers.

Feature to be tested: Search bar

QA test plan:

3 Test plans:

- **Input:** enter "science"
- **Output:** Check that you get a list of all tutors that are teaching science, all string containing "science" in the subject field

- **Input:** enter "1"
- **Output:** Check that the ID of the returned list contains the tutor with ID #1

- **Input:** enter a tutor name
- **Output:** Check that the results consist of these strings
- The tutors name should be exactly as the user defined it

Tabular Format:

Browser **Chrome**

Test #	Title	Description	Input	Expected correct Output	Test results PASS/FAIL
1	Test “science” is being returned after inputting	Type “science” in search field	science	Get all results that contain “science”	PASS
2	Test “1” is being returned after inputting	Type “1” in the search field	1	Get all results that contain ID 1	PASS
3	Test tutor name is being returned after submission	Type tutor name in search field	Bill	Get all tutors with first name Bill	

Browser **Firefox**

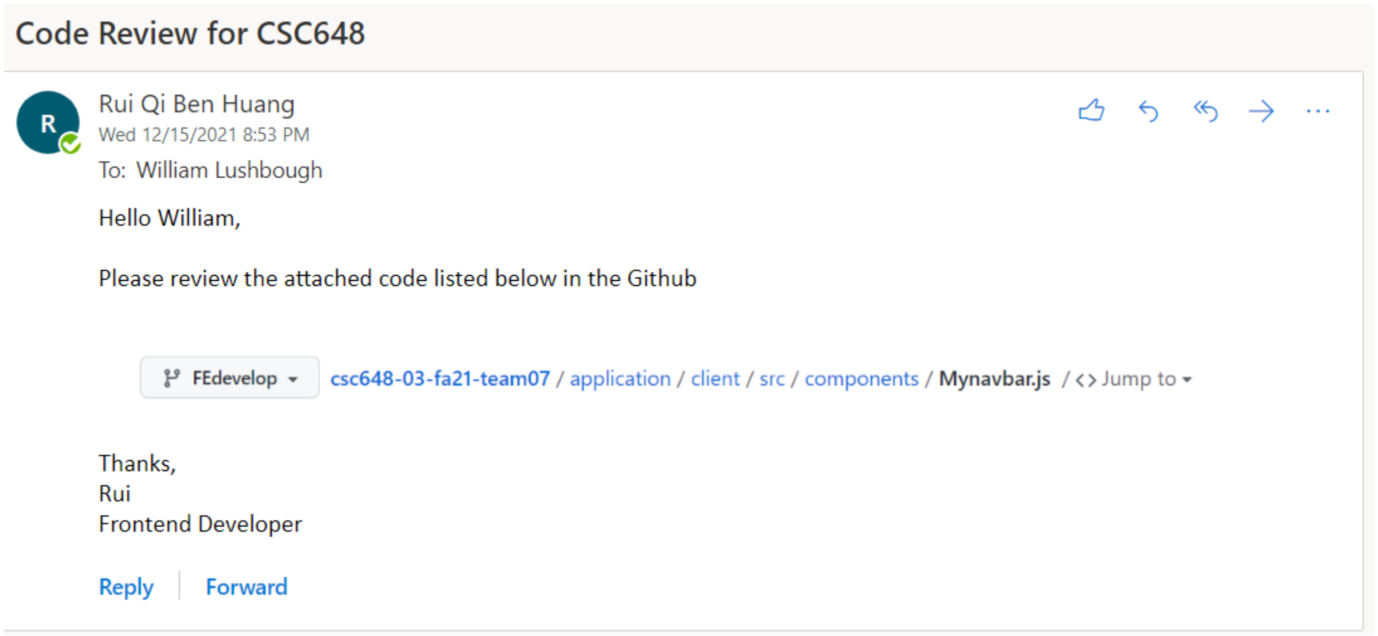
Test #	Title	Description	Input	Expected correct Output	Test results PASS/FAIL
1	Test “science” is being returned after inputting	Type “science” in search field	science	Get all results that contain “science”	PASS
2	Test “1” is being	Type “1” in the search	1	Get all results that	PASS

	returned after inputting	field		contain ID 1	
3	Test tutor name is being returned after submission	Type tutor name in search field	Bill	Get all tutors with first name Bill	PASS

Section 4: Code Review

Coder: Rui Qi Huang

Code Reviewer: William Lushbough



Code Review for CSC648

Rui Qi Ben Huang
Wed 12/15/2021 8:53 PM
To: William Lushbough

Hello William,

Please review the attached code listed below in the Github

FEdevelop csc648-03-fa21-team07 / application / client / src / components / Mynavbar.js / <> Jump to

Thanks,
Rui
Frontend Developer

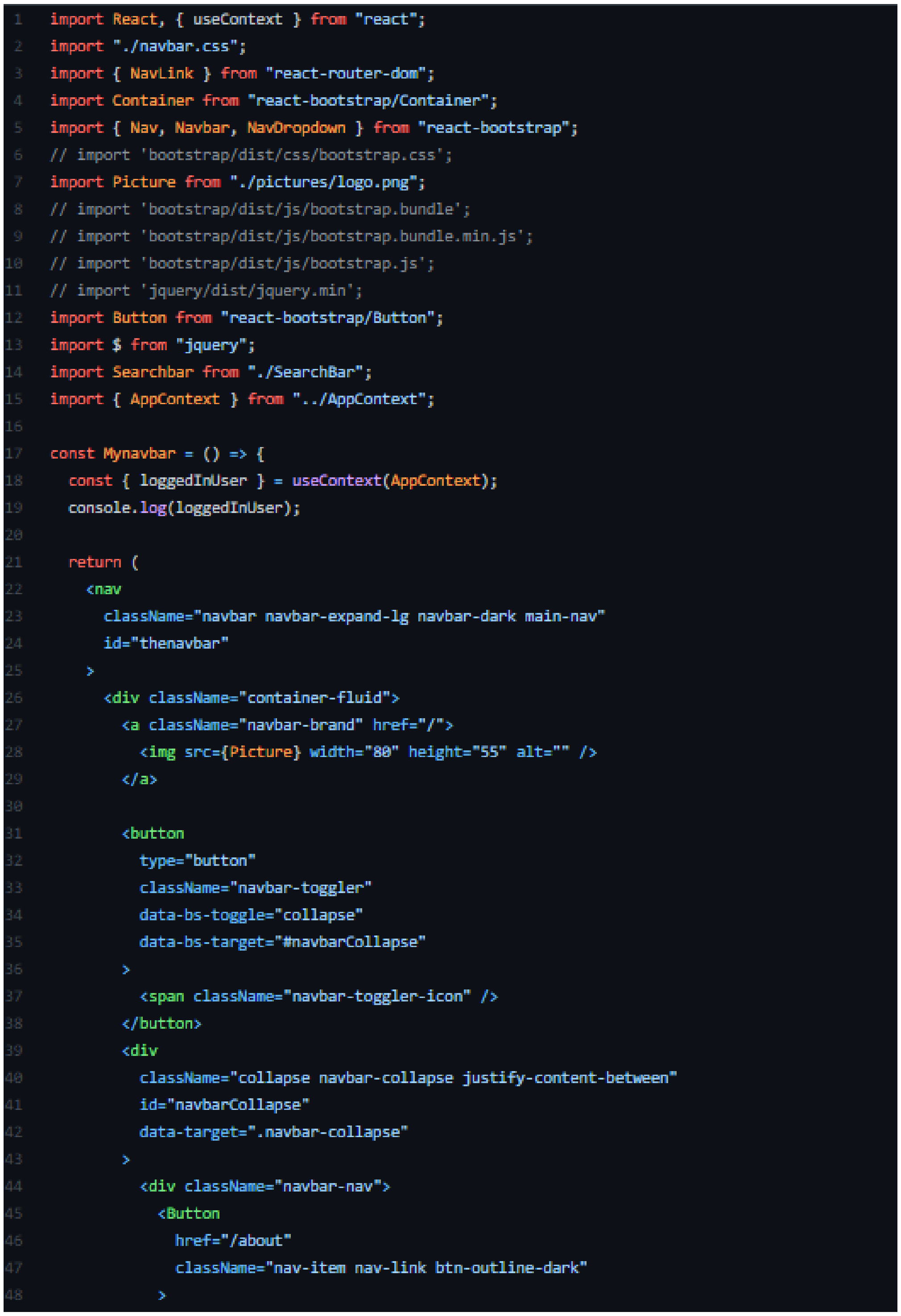
Reply | Forward

Thanks Rui,

I have reviewed your code regarding the navigation bar feature, titled Mynavbar.js. The following are some suggestions for the code.

In general, ought to be implementing basic headers in every code file in our repository. Therefore, we should add a simple header that includes the name of the title of the file, filename, and a list of contributors to the code. We should also remove the comments of unused code unless there is a listed specification as to why we are keeping the code commented out, for example, keeping commented code in the file for a future feature. The naming practices of each of your properties are in line with our Data Section of Milestone 2. The Github commits express great detail. My final suggestion is to include a few in-line comments that list what the following code is doing. Doing this in a brief manner will help future contributors understand the code more quickly.

Best,
William Lushbough
Github Master

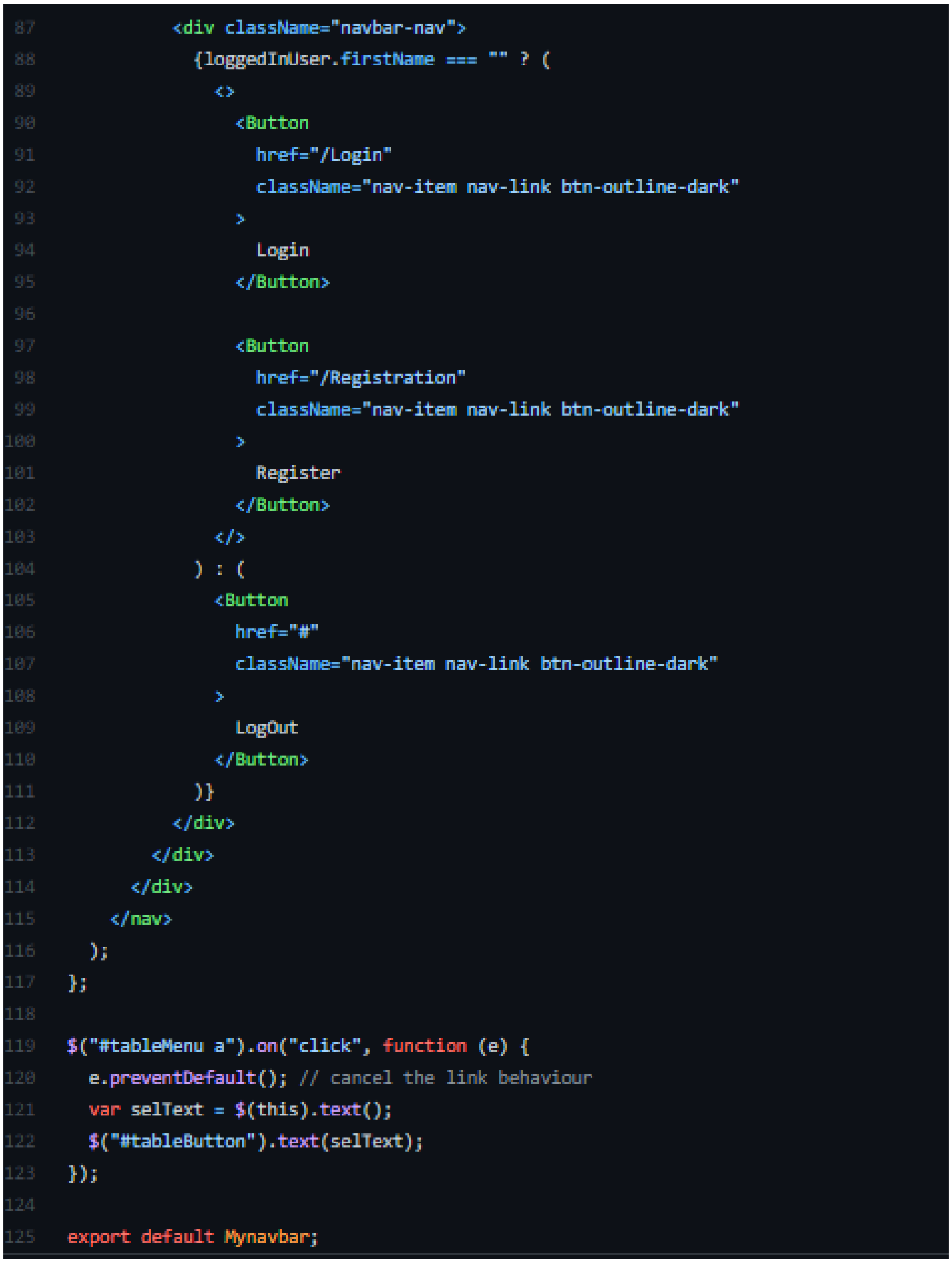


```
import React, { useContext } from "react";
import "./navbar.css";
import { NavLink } from "react-router-dom";
import Container from "react-bootstrap/Container";
import { Nav, Navbar, NavDropdown } from "react-bootstrap";
// import 'bootstrap/dist/css/bootstrap.css';
import Picture from "./pictures/logo.png";
// import 'bootstrap/dist/js/bootstrap.bundle';
// import 'bootstrap/dist/js/bootstrap.bundle.min.js';
// import 'bootstrap/dist/js/bootstrap.js';
// import 'jquery/dist/jquery.min';
import Button from "react-bootstrap/Button";
import $ from "jquery";
import Searchbar from "./SearchBar";
import { AppContext } from "../AppContext";

const Mynavbar = () => {
  const { loggedInUser } = useContext(AppContext);
  console.log(loggedInUser);

  return (
    <nav
      className="navbar navbar-expand-lg navbar-dark main-nav"
      id="thenavbar"
    >
      <div className="container-fluid">
        <a className="navbar-brand" href="/">
          <img src={Picture} width="80" height="55" alt="" />
        </a>

        <button
          type="button"
          className="navbar-toggler"
          data-bs-toggle="collapse"
          data-bs-target="#navbarCollapse"
        >
          <span className="navbar-toggler-icon" />
        </button>
        <div
          className="collapse navbar-collapse justify-content-between"
          id="navbarCollapse"
          data-target=".navbar-collapse"
        >
          <div className="navbar-nav">
            <Button
              href="/about"
              className="nav-item nav-link btn-outline-dark"
            >
```



```
          <div className="navbar-nav">
            {loggedInUser.firstName === "" ? (
              <>
                <Button
                  href="/Login"
                  className="nav-item nav-link btn-outline-dark"
                >
                  Login
                </Button>

                <Button
                  href="/Registration"
                  className="nav-item nav-link btn-outline-dark"
                >
                  Register
                </Button>
              </>
            ) : (
              <Button
                href="#"
                className="nav-item nav-link btn-outline-dark"
              >
                LogOut
              </Button>
            )}
          </div>
        </div>
      </div>
    </nav>
  );
};

$("#tableMenu a").on("click", function (e) {
  e.preventDefault(); // cancel the link behaviour
  var selText = $(this).text();
  $("#tableButton").text(selText);
});

export default Mynavbar;
```

Section 5: Security Self-Check

Asset to be Protected	**Types of possible/expected attacks**	**Strategy to mitigate/protect the asset**
User data that is being transferred over the network.	The network request can be intercepted.	Send sensitive data via POST requests.
User Passwords	If passwords are not encrypted hacker can obtain passwords	Passwords must be encrypted in before they are stored in the DB.
User Passwords	SQL injection attack, unnecessary characters to cause heavy overhead to the server	Put character limits on inputs, make sure email has @mail.sfsu.edu at end
Protecting Database Connection	If hackers get access to database server, they can access information in the database	This can be prevented by not hardcoding database information but putting it in a .env file

Section 6: Self-Check

1.**(DONE)** Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in Milestone 0. Application delivery shall be from chosen cloud server

2.**(DONE)** Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers

3.**(DONE)** All or selected application functions must render well on mobile devices

4.**(DONE)** Data shall be stored in the database on the team's deployment cloud server.

5.**(DONE)** No more than 50 concurrent users shall be accessing the application at any time

6.**(DONE)** Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.

7.**(DONE)** The language used shall be English (no localization needed)

8**(DONE)**. Application shall be very easy to use and intuitive

9.**(DONE)** Application should follow established architecture patterns

10.**(DONE)** Application code and its repository shall be easy to inspect and maintain

11.**(DONE)** Google analytics shall be used

12.**(DONE)** No e-mail clients shall be allowed.

13.**(DONE)** Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.

14.**(DONE)** Site security: basic best practices shall be applied (as covered in the class) for main data items

15.**(DONE)** Application shall be media rich (images, video etc.). Media formats shall be standard as used in the market today

16.**(DONE)** Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development

17.**(DONE)** For code development and management, as well as documentation like formal milestones required in the class, each team shall use their own github to be set-up by class instructors and started by each team during Milestone 0

18.**(DONE)** The application UI (WWW and mobile) shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Fall 2021 For Demonstration Only" at the top of the WWW page. (Important so as to not confuse this with a real application).