

SW Engineering CSC648/848 Fall 2021

Project Title: TutorPal

Team Number: 03

Names of Students:

Snehal Patil(spatil2@mail.sfsu.edu) – Team Lead

Cameron Robinson - Back-End Lead

Daniel Elnaggar - Front-End Lead

James Pratt - GitHub Master

Rollin Kung

Saptarshi Roy

Milestone: 4

Date: 12/17/2021

History Table:

Date	Revision
12/17/2021	1 st Version

Table of Content

Sr. No	Content	Page No
1	Product Summary	3
2	Usability Test Plan	4
3	QA Test Plan	6
4	Code Review	9
5	Self-check on best practices for security	12
6	Self-check: Adherence to original Non-functional specs	13

1. Product Summary

Name of the Product: Tutor Pal

Final P1 List:

1. A user can search for available courses, tutors, or majors using the search bar.
2. A user can browse tutor posts using the “Start Browsing!” button on the front page.
3. Recent registered tutors and their posts are displayed on the front page.
4. Users are able to view each tutor’s post description and picture.
5. A user is able to contact a tutor by sending a message, which will then utilize lazy registration and ask the user to register.
6. A user is able to sign up as a tutor by clicking on the “Become a tutor!” button on the front page.
7. A user is able to offer tutoring by creating a post.
8. Users are able to visit the “About” page from the navigation bar.
9. Users can view their dashboard, which will display recent messages.

Product URL: <https://csc648team3.ddns.net/>

2. Usability Test Plan

Test Objectives:

We will be testing if users can search for a tutor and send them a message. The main selling point of our website is to let SFSU students find tutors for their classes and the search function as well as the contact function are essential for this purpose. Finding and messaging tutors should be intuitive otherwise our site will not be sustainable because it will not be able to attract customers.

Test background and setup:

- System Setup
We will have each user enter separate rooms. Each room will have a computer with multiple types of browsers installed so users can use whichever one they are most comfortable with. A stopwatch will also be provided so users can measure how long they take to complete each task.
- Starting Point
Each user will be provided the URL of the tutoring website and will already be logged in. Software that measures clicks will also be installed on the provided computers and will start counting when the test starts. The decision of which browser they wish to use as well as how to accomplish each task will be left for each user to decide. The timer will start once the home page has finished loading.
- Intended Users
There are multiple intended groups of users. The first group are users who are new to the website and have never interacted with a tutoring website before. The second group are users who have some experience providing tutoring before.
- URL of the system to be tested
The URL is <https://csc648team3.ddns.net/>
- What is to be measured
We will be focusing on measuring user satisfaction by using Likert tests.

Usability Task Description:

The first task for testers is to search for a tutor. The test starts after the landing page has finished loading and ends when the search result page has finished loading. The task will be considered completed successfully if it is completed within 1 minute.

The second task for testers is to contact a tutor. The test starts after the search page has finished loading and ends when the message to the tutor is sent. The task will be considered completed successfully if it is completed within 1 minute.

Evaluation of Effectiveness:

We will perform the usability test with multiple new users and record how many of them are able to complete the task within two minutes.

Evaluation of Efficiency:

We will measure the number of clicks and the total amount of time it takes each user to complete the task.

Evaluation of user satisfaction:

- The search bar was easy to locate.

Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree

- The contact tutor button was easy to locate.

Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree

- The sending a message to the tutor was intuitive.

Strongly disagree	Disagree	Neither agree nor disagree	Agree	Strongly agree

3. QA Test Plan

Test Objectives:

To make sure all listed features are functioning properly.

HW and SW Setup:

Website URL: <https://dev-csc648team3.ddns.net/>

Server Host: Amazon AWS

Operating System: Ubuntu 20.04 Server

Database: MySQL

Web Server: Apache 2.4

Server-Side Language: JavaScript

Additional Technologies: Web Framework: Node.JS/Express

IDE: JetBrains WebStorm

Web Analytics: Google Analytics

SSL Cert: Encrypt (Cert Bot)

Features to be Tested:

- Search Bar
- Contact Button
- Register Button
- Log In Button
- Log Out Button


QA Test Plan:

Test#	1	2	3	4	5
Title	Log In	Log Out	Register	Search	Message Tutor
Description	This will test whether or not a user will be able to log into their account	This will test whether or not a user will be able to log out of their account	This will test whether or a user is able to create a new account	This will evaluate whether a user is able to search for a tutor or a class by typing in search bar	Registered student would attempt to contact the tutor via message
Test Input	From the landing page, user will click on Login and enter credentials (email: testuser@mail.sfsu.edu and password: test)	After confirming that the user is logged in, from the landing page, user will click on the small arrow at the top right corner and select logout	From the landing page, user will click on "SignUp" and then enter first and last name, email id, password, and select a major	User will type in the name of a course "CSC 648," and click on Search	From a tutor's page, the user will type in a message and click on Contact Me!
Expected Correct Output	The user will be redirected into home page showing that the user has logged in by displaying their name on the right-hand corner of the web page	Will redirect to home page showing that the user has the option to log into their account if they choose to do so	Once a user registers, a pop up would appear saying that they have successfully registered, and they would be redirected to the login page.	When a user searches up e.g., csc 648, the tutor post with the above class would appear.	If the user is not logged in, a message will pop up asking whether or not the user wishes to log in. If the user is logged in they will be directed to a messaging page where they can send a message. If the message goes through the user is notified that

					the message was successfully sent.
Test Result	Pass (Chrome) Pass (Firefox)	Pass (Chrome) Pass (Firefox)	Pass (Chrome) Pass (Firefox)	Pass (Chrome) Pass (Firefox)	Pass (Chrome) Pass (Firefox)

4. Code Review


- Review Email



ckRobinson <notifications@github.com>
Sat 12/11/2021 6:49 AM
To: CSC-648-SFSU/csc648-03-fa21-team03 <csc648-03-fa21-team03@noreply.github.com>
Cc: Snehal Shrinivas Patil; Push <push@noreply.github.com>


[@ckRobinson](#) pushed 3 commits.

- [b25242f](#) Added reference to mysql in database export.
- [0080211](#) Properly connected registration controller to app.
- [1e39778](#) Updated register user function to access database.



ckRobinson <notifications@github.com>
Sat 12/11/2021 6:58 AM
To: CSC-648-SFSU/csc648-03-fa21-team03 <csc648-03-fa21-team03@noreply.github.com>
Cc: Snehal Shrinivas Patil; Author <author@noreply.github.com>


We need some documentation on the register.js file. Everything else looks alright, approved pending documentation updates.



rkung117 <notifications@github.com>
Sat 12/11/2021 7:58 AM
To: CSC-648-SFSU/csc648-03-fa21-team03 <csc648-03-fa21-team03@noreply.github.com>
Cc: Snehal Shrinivas Patil; Push <push@noreply.github.com>


[@rkung117](#) pushed 2 commits.

- [d6b2b21](#) removed registerbackend code
- [3018982](#) added comments about register file and registeruser function



rkung117 <notifications@github.com>
Sat 12/11/2021 8:02 AM
To: CSC-648-SFSU/csc648-03-fa21-team03 <csc648-03-fa21-team03@noreply.github.com>
Cc: Snehal Shrinivas Patil; Author <author@noreply.github.com>


removed extra register file. added comments for register.js file. please check when you have time.



ckRobinson <notifications@github.com>
Sat 12/11/2021 8:26 AM
To: CSC-648-SFSU/csc648-03-fa21-team03 <csc648-03-fa21-team03@noreply.github.com>
Cc: Snehal Shrinivas Patil; Author <author@noreply.github.com>

[@ckRobinson](#) approved this pull request.


Looks good. Approved.







ckRobinson <notifications@github.com>
Sat 12/11/2021 8:27 AM
To: CSC-648-SFSU/csc648-03-fa21-team03 <csc648-03-fa21-team03@noreply.github.com>
Cc: Snehal Shrinivas Patil; Author <author@noreply.github.com>


Merged [#27](#) into develop.

- GitHub Commit and reviews





Merged ckRobinson merged 28 commits into `develop` from `landingpage` 6 days ago

 Conversation 3  Commits 28  Checks 0  Files changed 19





snehalp396 commented 6 days ago



please check and update



 ckRobinson commented 6 days ago  ...

We need some documentation on the register.js file. Everything else looks alright, approved pending documentation updates.



 ckRobinson assigned rkung117 6 days ago



 rkung117 added 2 commits 6 days ago

-  removed registerbackend code d6b2b21
-  added comments about register file and registeruser function 3018982



 rkung117 commented 6 days ago  ...


removed extra register file. added comments for register.js file. please check when you have time.

  ckRobinson approved these changes 6 days ago [View changes](#)

 ckRobinson left a comment  ...

Looks good. Approved.

  ckRobinson merged commit 7cc8669 into develop 6 days ago [Revert](#)

 **Pull request successfully merged and closed** [Delete branch](#)

You're all set—the landingpage branch can be safely deleted.

- Code review:

```

22 + * This function is called when user submits data on the registration page. The data
23 + * that the user has given is inserted into the database.
24 + * @param request the request from the user that will be inserted into the database
25 + * @param callback
26 + */
27 + function registerUser(request, response, callback){
28 +
29 +     const hashedPassword = loginHashing.hashPasswordForRegistration(request.body.password);
30 +
31 +     const email = request.body.email;
32 +     const password_hashed = hashedPassword.hash;
33 +     const password_salt = hashedPassword.salt;
34 +     const first_name = request.body.first_name;
35 +     const last_name = request.body.last_name;
36 +     const major = request.body.major_dropdown;
37 +
38 +     const sqlSearch = "SELECT * FROM users WHERE email = ?";
39 +     const search_query = mysql.format(sqlSearch,[email]);
40 +
41 +     request.registered = false;
42 +
43 +     database.query (search_query, async (err, result) => {

```

```
110 application/node/controller/register.js
...
1 + /**
2 + *
3 + * This file is used when the user attempt to register. It takes the data user inputs and
4 + * inserts it into the database. If user is already logged in and attempts to register, it
5 + * redirects them to the home page. Otherwise, it continues to the registration page. If user
6 + * is already registered and they attempt to register again, they are redirected to the login
7 + * page. Otherwise, it continues to the registration page.
8 + *
9 + * @author Rollin Kung and Cameron Robinson.
10 + * @date 12/10/2021
11 + * @since 0.0.1
12 + */
13 +
14 + const express = require('express');
15 + const router = express.Router();
16 +
17 + const loginHashing = require("../model/loginHashing");
18 +
19 + const {database, mysql} = require("../model/mysqlConnection");
20 +
49 application/node/views/studentRegister.ejs
...
1 +
2 +
3 + <!-- This file is used when the user attempt to register, it enters the user details and clicks submit button to register.
4 + If the validation is correct, the data is inserted in the database and new user is created.
5 +
6 + @author Snehal Patil
7 + @date 11/19/2021 -->
8 +
9 +
10 <!doctype html>
11 <html lang="en">
12 <head>
13
22 <br>
23
24 <section class="container">
25 <div class="row content d-flex justify-content-center mt-5">
26 <div class="row content d-flex justify-content-center mt-3">
27 <div class="col-md-5 mt-0">
28 <h3 class="h3 mb-4 text-center">Register Here</h3>
29 <div class="signup-form text-center box shadow bg-white p-4">
30 <form action="/" class="mb-3">
31 <div class="row">
32 <div class="form-floating mb-3 col-md-6">
33 <input type="text" name="fname" class="form-control rounded-0" id="floatingInput" placeholder="Enter First Name" required autofocus>
34 <label for="floatingInput">First Name<span class="text-danger">*</span></label>
35 <input type="text" name="first_name" class="form-control rounded-0" id="first_name" placeholder="First Name" required autofocus>
36 <label for="first_name">First Name<span class="text-danger">*</span></label>
37 </div>
38 <div class="form-floating mb-3 col-md-6">
39 <input type="text" name="lname" class="form-control rounded-0" id="floatingInput" placeholder="Enter Last Name" required autofocus>
40 <label for="floatingInput">Last Name<span class="text-danger">*</span></label>
41 <input type="text" name="last_name" class="form-control rounded-0" id="last_name" placeholder="Last Name" required autofocus>
42 <label for="last_name">Last Name<span class="text-danger">*</span></label>
43 </div>
44 <div class="form-floating mb-3">
45 <input type="email" class="form-control rounded-0" id="floatingInput" placeholder="Email Address" required autofocus>
46 <label for="floatingInput">SFSU Email Address<span class="text-danger">*</span></label>
47 <input type="email" name="email" class="form-control rounded-0" id="email" placeholder="Email Address" required autofocus>
48 <label for="email">SFSU Email Address<span class="text-danger">*</span></label>
49 </div>
50 <div class="form-floating mb-3 col-md-12">
51 <input type="password" name="password" class="form-control rounded-0" id="floatingPassword" placeholder="password" required autofocus>
52 <label for="floatingPassword">Password<span class="text-danger">*</span></label>
53 <input type="password" name="password" class="form-control rounded-0" id="password" placeholder="password" required autofocus>
54 <label for="password">Password<span class="text-danger">*</span></label>
55 </div>
56 </div>
57 </div>
58 </div>
```

5. Self-check on best practices for security

Asset to be protected	Types of possible/expected attacks	Strategy to mitigate/protect the asset
Passwords	Admin can view password in database, database breach, man in the middle attack capturing password	Password is salted and hashed when stored in the database to prevent viewing or sending the passwords in plain text
Tutor data	Tutor data database breach	Protecting the database only allowing access with sshKey.
Entire Database	SQL Injection of malicious SQL code	Use data sanitation on user entered data before running query on database.
Input validation	User inputs invalid or malicious data	Use bootstrap field validation to make sure the data is of a valid type before sending to the back end.
Search bar	User inputs invalid or malicious data	Use bootstrap field validation to make sure the user can only input 40 characters into the search bar to prevent sending too much data to the back end.
Registration email	User attempts to register with a non-SFSU email	Use bootstrap field validation to make sure the email entered contains sfsu.edu

6. Self-check: Adherence to original Non-functional specs

- 1) 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. – **DONE**
- 2) Application shall be optimized for standard desktop/laptop browsers e.g., must render correctly on the two latest versions of two major browsers. – **DONE**
- 3) All or selected application functions must render well on mobile devices. – **DONE**
- 4) Data shall be stored in the database on the team's deployment cloud server. – **DONE**
- 5) No more than 50 concurrent users shall be accessing the application at any time. – **DONE**
- 6) Privacy of users shall be protected, and all privacy policies will be appropriately communicated to the users. – **DONE**
- 7) The language used shall be English (no localization needed) – **DONE**
- 8) Application shall be very easy to use and intuitive – **DONE**
- 9) Application should follow established architecture patterns – **DONE**
- 10) Application code and its repository shall be easy to inspect and maintain – **DONE**
- 11) Google analytics shall be used - **DONE**
- 12) No e-mail clients shall be allowed. – **DONE**
- 13) Pay functionality, if any (e.g., paying for goods and services) shall not be implemented nor simulated in UI. – **DONE**
- 14) Site security: basic best practices shall be applied (as covered in the class) for main data items – **DONE**
- 15) Application shall be media rich (images, video etc.). Media formats shall be standard as used in the market today – **DONE**
- 16) Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development – **DONE**
- 17) 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 – **DONE**
- 18) 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). – **DONE**