SW Engineering CSC648/848 Fall 2024

BRAIN BUFFS (Team 1)

Milestone 4: Product description, QA and Usability Testing, Code review, Security self-audit

Team Members:

Shum Usami (susami@sfsu.edu)	Team Lead
Adharsh Thiagarajan	Frontend Lead
Devon Huang	Backend Lead
Kim Nguyen	GitHub Master
Thiha Aung	Software Developer

Change History:

Date Submitted:	Dec 17, 2024
Date Revised:	

1. Product summary

Name of the Product

Brain Buff

Product Description

Brain Buffs is a tutoring platform designed exclusively for SFSU students, making it easier to find experienced tutors for specific SFSU courses. Only students with SFSU email addresses can register. Users can browse tutors by SFSU-specific courses and subjects, send messages directly to tutors, and even sign up as tutors to support their peers.

Major Functions

Guests

1. **Search Tutor Postings:** Guests shall be able to search tutor postings based on subjects, class names, professor names, tutor names, and ratings.
2. **Browse Tutor Postings:** Guests shall be able to browse and view the search results of tutor postings.
3. **Register:** Guests shall be able to register an account using their SFSU email and password.
4. **Login:** Guests shall be able to login using their SFSU email and password.

Users (Students and Tutors)

5. **Inherit Guests Capabilities:** Users shall be able to do what guests can do except Register and Login.
6. **Send Booking Request:** Users shall be able to send booking requests to tutor postings.
7. **Create Tutor Postings:** Users shall be able to create new tutor postings with subjects, availability, rates, custom descriptions, and a PDF document.
8. **Delete Tutor Postings:** Users shall be able to browse their own tutor postings and delete them.
9. **Browse booking requests:** Users shall be able to browse the booking requests sent to their tutor postings.

Admin

10. **Approve Tutor Postings:** Admins shall be required to review and approve created or edited tutor postings before they go live.

11. **Delete Inappropriate Users/Tutor Postings/Booking Requests:** Admins shall be able to delete users, tutor postings, and booking requests that violate platform policies.
12. **Ban Inappropriate Users:** Admins shall be able to ban users that violate platform policies so that banned users shall not be able to login.

URL

http://ec2-13-57-185-95.us-west-1.compute.amazonaws.com or 13.57.185.95

2. Usability test plan for selected function

Test objectives

The primary objective of this test is to evaluate the **effectiveness**, **efficiency**, and user **satisfaction** of **Search Function** in the Brain Buffs application. As the core feature enabling students to find tutors, its usability is critical for the overall user experience. The evaluation will focus on ease of use, relevance of search results, and smooth navigation through the system. Additionally, testers will identify any usability issues or challenges that could hinder performance. Insights gathered will guide future improvements to enhance the overall search functionality.

Test background and setup

The usability test for the Brain Buffs web application will require participants to use a desktop, laptop or mobile with a stable internet connection and an updated browser (e.g., Google Chrome, Mozilla Firefox, or Safari). Testers will begin on Brain Buffs homepage.

The test will involve SFSU students seeking tutoring or interested in becoming tutors. Also, participants will be selected to represent a range of technical proficiency, helping identify usability issues for both experienced and less experienced users.

The test will take place in a home environment. Testers will verbalize their thoughts and challenges, with no cameras or screen recording. No training is required, as tasks will be straightforward with clear instructions provided at the start.

The system URL to be tested:
http://ec2-13-57-185-95.us-west-1.compute.amazonaws.com/.

Usability Task description

Find a tutor for CSC 415 Operating System Concepts class.

Plan for evaluation of Effectiveness

Effectiveness will be measured by the percentage of tasks completed successfully and the frequency of errors encountered during the task. Task success is defined as a participant successfully locating a relevant tutor for “CSC 415 Operating System Concepts” without external assistance. The success rate will be calculated as the ratio of participants who complete the task to the total number of participants. Errors, including navigation mistakes, incorrect actions, or irrelevant search inputs, will be tracked and categorized to identify specific usability issues. This evaluation will help pinpoint areas where users struggle and provide insights into improving the search function’s overall effectiveness.

Plan for evaluation of Efficiency:

Efficiency will be assessed by measuring the time taken to complete the task and the number of interactions or clicks required to achieve the desired outcome. Task time will be recorded from the moment the participant begins until they successfully locate a tutor, with average task time calculated across all participants. Additionally, the number of clicks or interactions will be tracked to identify unnecessary steps or actions. Fewer interactions and shorter task times will indicate higher efficiency, while longer durations or excessive clicks will reveal areas for streamlining the search process. These insights will guide efforts to improve the search function’s speed and ease of use.

Plan for Evaluation of user satisfaction

User satisfaction will be assessed using a **Likert scale questionnaire** with the following statements:

1. **Ease of Use:**
   "Searching for tutors using course numbers, subjects, or keywords was easy."
   Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree
2. **Accuracy of Results:**
   "The search results were relevant to my query."
   Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree
3. **Overall Satisfaction:**
   "I am satisfied with the Search Tutors feature."
   Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree

3. QA test plan and QA testing

Test objectives

The primary objective of this QA test plan is to verify the correctness and reliability of the search functionality. Specifically, these tests will ensure the search results align with user inputs and selections (including subject and course filters), and that no unintended behaviors occur.

HW and SW setup (including URL)

Hardware and Software Setup:

- Platforms:
  - Desktop (PC)
  - Mobile browsers (on smartphone or tablet)
- Browsers:
  - Google Chrome (latest stable release)
  - Mozilla Firefox (latest stable release)

Test URL:

- URL: http://ec2-13-57-185-95.us-west-1.compute.amazonaws.com or 13.57.185.95

Feature to be tested

- **Search Functionality**: Verification includes ensuring correct retrieval of all tutors, filtering by subject, and filtering by course number.

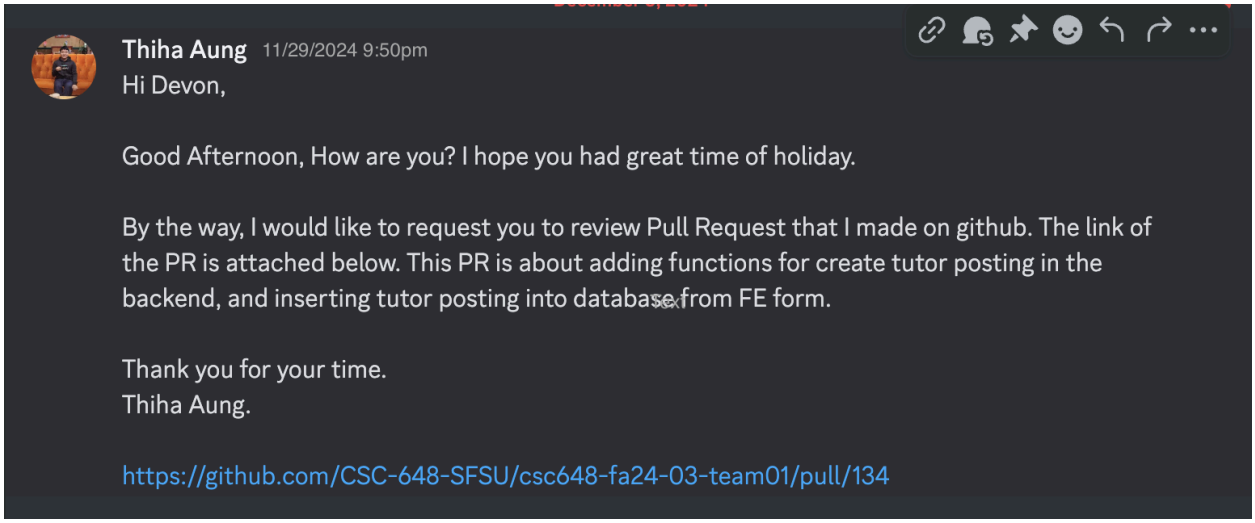
QA Test plan

Test #	Title	Description	Input	Expected correct output	Results (PASS or FAIL for each tested browser)
1	Search All	Validate that an empty search returns all tutor postings.	1. Click on the search field. 2. Do not enter any text. 3. Click the search button (magnifier icon).	The system should return all available tutor postings (18 total).	PASS (Chrome) / PASS (Firefox)
2	Search by Subject	Ensure filtering by subject returns the correct subset.	1. Open the subject dropdown in the search bar. 2. Select “(CSC) Computer	The system should return only tutor postings for Computer Science (8 results).	PASS (Chrome) / PASS (Firefox)

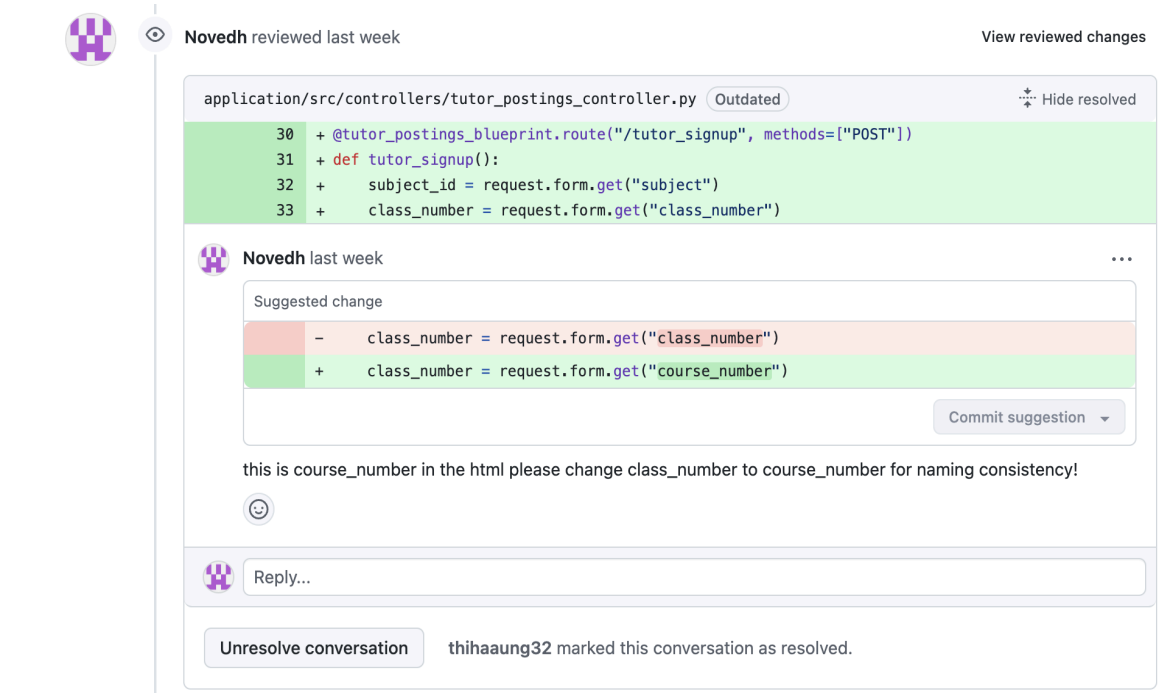
			Science”. 3. Click the search button (magnifier icon).		
3	Search by course	Ensure filtering by course number yields correct results.	1. Click on the search field. 2. Type “648”. 3. Click the search button (magnifier icon).	The system should return the single posting that matches course “648”.	PASS (Chrome) / PASS (Firefox)

4. Peer Code Review

4.1 Requesting a Review

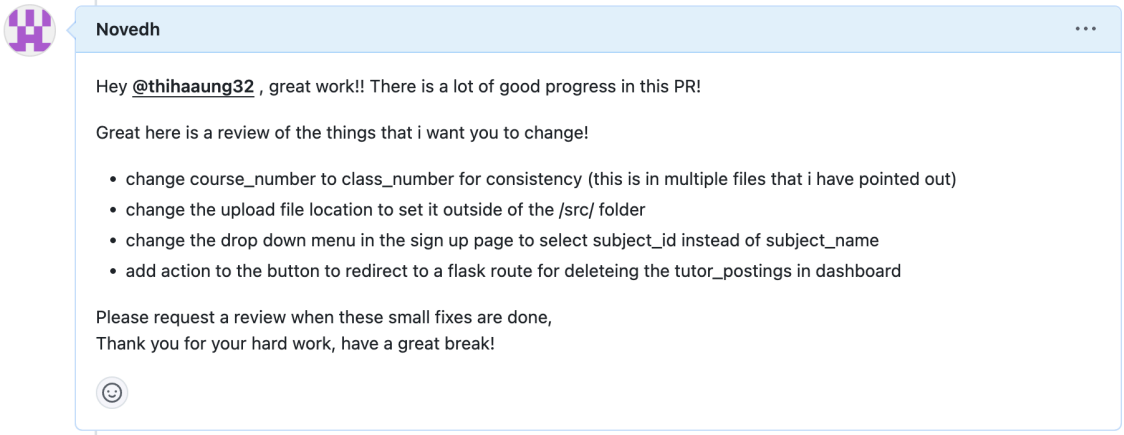


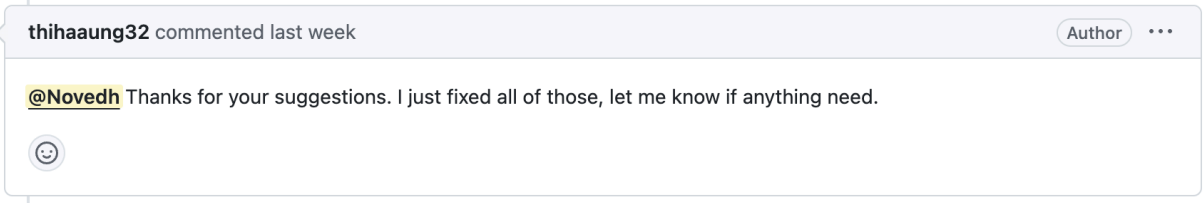
4.2 Performing the Review



-

4.3 Providing Feedback





5. Self-check on best practices for security

Major assets, threats, and protecting strategies

Asset to be protected	**Types of possible/expected attacks**	**Consequence of security breach**	**Your strategy to mitigate/protect the asset**
User passwords in database	Brute force, SQL injection, credential stuffing, data leaks	Unauthorized access, identity theft, loss of user trust	We use bcrypt for encryption by hashing user passwords during creation and verifying provided passwords against stored hashes.
user table in database	SQL injection, database leaks	Privacy violations, financial and legal repercussions	We use parameterized queries, restrict database privileges and regular audits for vulnerabilities.
booking_request table in database	SQL Injection, Data Leaks	Privacy violations, Loss of User Trust, Spam/Phishing Attacks	We use parameterized queries, restrict database privileges and regular audits for vulnerabilities

Password encryption in the Database

Yes, we encrypt passwords before being stored in the database. We use the bcrypt.hashpw method to hash the plaintext password using a salt generated by bcrypt.gensalt. Then, the hashed password is inserted into the database. This confirms the hashing and ensures the plaintext password is not stored in the database.

Input data validation

Input Field	**Validation Code**
Search bar input for up to 40	<input type="text" class="form-control"

alphanumeric characters	name="search_text" placeholder="Search" aria-label="Search field" **maxlength="40"** pattern="[A-Za-z0-9]*" value="{{ search_text }}" />
SFSU student registration e-mail (must include “sfsu.edu” at the end)	<input type="email" class="form-control" placeholder="Enter SFSU Email" name="email" required **pattern=".+@sfsu\.edu"** maxlength="40" title="Please enter a valid SFSU email address" value="{{ email }}" />
Checkbox for acceptance of terms in registration form	<input **type="checkbox"** class="form-check-input me-2" **id="terms"** required /> <label class="form-check-label me-2" for="terms" >Accept the Terms and Conditions</label >

6. Self-check of the adherence to original Non-functional specs

Specs	**Status**
Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0	DONE
Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers	DONE
All or selected application functions shall render well on mobile devices (no native app to be developed)	DONE
Posting of tutor information and messaging to tutors shall be limited only to SFSU students	DONE
Critical data shall be stored in the database on the team’s deployment server.	DONE
No more than 50 concurrent users shall be accessing the application at any time	DONE
Privacy of users shall be protected	DONE
The language used shall be English (no localization needed)	DONE

Application shall be very easy to use and intuitive	DONE
Application shall follow established architecture patterns	DONE
Application code and its repository shall be easy to inspect and maintain	DONE
Google analytics shall be used	DONE
No e-mail clients shall be allowed. Interested users can only message to sellers via in-site messaging. One round of messaging (from user to seller) is enough for this application	DONE
Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.	DONE
Site security: basic best practices shall be applied (as covered in the class) for main data items	DONE
Media formats shall be standard as used in the market today	DONE
Modern SE processes and tools shall be used as specified in the class, including collaborative and continuous SW development and GenAI tools	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 2024. For Demonstration Only" at the top of the WWW page Nav bar. (Important so as to not confuse this with a real application).	DONE

7. Use of genAI tools like ChatGPT and copilot

Tools being used:

- ChatGPT-4o (Used for test plan structuring, clarity improvements, and naming suggestions)

Tasks and Usefulness

1. Test Plan Structuring: (HIGH)
    - Helped organize the QA test plan into a professional and readable format
    - Suggested section headings and table structures for clarity
    - Improved document presentation for easier comprehension
2. Content Rephrasing for Clarity: (HIGH)
    - Refined descriptions of test cases to ensure conciseness and remove ambiguity

- Made test steps and expected outputs clearer and more straightforward

3. Test Case Naming Suggestions: (MEDIUM)
    - Suggested standardized and clear names for test cases
    - Ensured naming consistency across all test scenarios
4. Expected Output Refinement: (HIGH)
    - Rephrased expected results to ensure precision and eliminate Ambiguity
    - Made expected outputs easier to verify during testing

Key Examples and Prompts:

1. Test Plan Structuring:
   "Can you help me structure and improve this QA test plan with test steps, outputs, and better clarity?"
   Result: Created a refined table format with organized test scenarios, steps, and results sections.

2. Test Case Naming Suggestions:
   "Provide concise test case names for a search function test involving empty search, subject filtering, and course filtering."
   Result (Suggested names):
    - Search All
    - Search by Subject
    - Search by Course
3. Expected Output Refinement:
   "Make the expected outputs for these test cases clearer and less ambiguous."
   Result (Updated expected output phrasing to):
    - “The system should return all available tutor postings (e.g., 12 total results).”

Additional Insights:

1. Process Improvements:
    - AI tools streamlined the structuring and writing of the QA test plan
    - Reduced time spent on manual organization and phrasing
2. Best Practices:
    - Maintained clear and concise test case descriptions
    - Ensured consistent naming conventions and documentation
3. Learning Outcomes:
    - Improved ability to write structured test plans
    - Gained insights into professional QA documentation standards