SW Engineering CSC648-848

Spring 2024

Campus Buy/Sell Application: SwiftSell
Team 04

Team Members:
Aymane Arfaoui, Team Lead (aarfaoui@sfsu.edu)
Markus Reyer, Github Master
Amandeep Singh, Front End Lead
Alexis Alvarez, Back End
David Daly, Back End Lead

Milestone 4

05/22/2024

1. Executive Summary

Product Name: SwiftSell

Description:

SwiftSell is a dynamic marketplace platform developed by a team of passionate undergraduate, graduate, and postgraduate students, aimed at revolutionizing the way students buy and sell goods within the San Francisco State University (SFSU) community. Our project is motivated by the need to create a safe, convenient, and student-friendly marketplace that offers an alternative to generic platforms like Facebook Marketplace and Craigslist. By providing a platform specifically tailored to the needs of college students, SwiftSell seeks to enhance the overall student experience at SFSU.

Major Committed Functions (P1):

1. An unregistered User shall be able to search for specific items.
2. An unregistered User shall be able to browse the website's items
3. An unregistered User shall be able to see images for items/services with accurate description.
4. An unregistered User shall be able to register.
5. A Registered User shall have a dashboard with information containing order history, messages, notifications, and the items they posted for sale.
6. A Registered User shall be able to post items and/or services(tutoring,delivery, etc) for sale with approval of admin
7. A Registered User who posts items for sale shall upload at least one image, a brief description of said item, and a reasonable price for the item.
8. A Registered User shall be able to message sellers about items for sale.
9. Only Registered Users shall be able to message other Registered Users.
10. A Registered User shall be able to post small jobs such as tutoring, moving furniture, etc.
11. A Registered User shall be informed if another user wants one of their items via in-site messaging.
12. A Registered User shall be able to login.
13. Verified Users shall have profiles via SFSU email authentication.
14. A User shall be able to search items by course number.
15. An Admin shall be required to approve or deny user's posts to go online

Unique Features:

1. Exclusively tailored to the SFSU community with verification through SFSU email addresses.
2. Local listings and **job services** specifically for students, faculty, and staff members.
3. Enhanced search capabilities for finding class-specific textbooks and other items.

Access URL:

2. Usability Test Plan for Search Functionality

Test Objectives

The objective of this usability test is to evaluate the effectiveness, efficiency, and user satisfaction of the search functionality within our web application. We aim to ensure that users can easily and accurately find the information they need.

Test Background and Setup

System Setup:

The web application should be hosted on a live server, accessible via the internet.

The search functionality must be fully implemented and operational.

Required hardware for testers includes a computer or mobile device with internet access.

Starting Point:

Testers should start from the homepage of the web application.

Hardware Needed:

A computer or mobile device with an internet connection.

Screen recording software to capture the tester's interactions, if applicable.

Intended Users:

Users for this functionality are mainly SFSU students, faculty, and staff.

URL of the System to be Tested:

http://ec2-34-228-231-71.compute-1.amazonaws.com/

Test Environment:

The test can be conducted anywhere..

No technical equipment is required.

No formal training is required before the test.

Plan for Evaluation of Effectiveness

Plan for Evaluation of Effectiveness: Effectiveness will be measured by the accuracy and completeness of the search results. This will be evaluated by tracking the number of correct results the user finds, comparing the user's search results with a predefined set of correct results, and measuring the percentage of successful searches where users find the required information within the first three attempts.

Plan for Evaluation of Efficiency

Efficiency will be measured by the time it takes for users to complete search tasks and the number of steps they take. This can be evaluated by recording the time from when the user initiates the search until they find the correct information, counting the number of clicks or interactions required to complete the search task, and evaluating the user's ability to refine their search queries to achieve better results.

Plan for Evaluation of User Satisfaction

We will instruct users to complete the following tasks: Search for specific items or services on our website and contact a seller or inquire about a service. After completing these tasks, users will fill out a survey that includes Likert scale questions to assess their subjective thoughts on the website's effectiveness, efficiency, and overall satisfaction.

Usability Task Description:

Users will be instructed to perform the following tasks before filling out the Likert scale questionnaire:

1. Navigate to the homepage of the web application.
2. Use the search bar to find information on a specific topic (e.g., "search CSC 415 textbook").
3. Note down the first three results and assess their relevance to the search query.
4. Refine the search query to get more accurate results if necessary.
5. Spend no more than 3 minutes on this task.

Likert Scale Evaluation Entries:

Ease of Use:

The search functionality was easy to use (circle one).

Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree

Relevance of Results:

The search results were relevant to my query (circle one).

Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree

Overall Satisfaction:

I am satisfied with the overall performance of the search functionality (circle one).

Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree

3. QA test plan and QA testing

Test Objectives

The objective of this QA test plan is to ensure the reliability, correctness, and overall quality of the search functionality within the SwiftSell platform. This involves verifying that users can successfully search for items, including by course number, ensuring search results are accurate and relevant, the search input persists, and the user interface operates smoothly across different browsers.

HW and SW Setup

Hardware:

- A computer or mobile device with internet access.

Software:

- Three major web browsers: Google Chrome, Mozilla Firefox, and Safari.
- SwiftSell platform URL: http://ec2-34-228-231-71.compute-1.amazonaws.com/

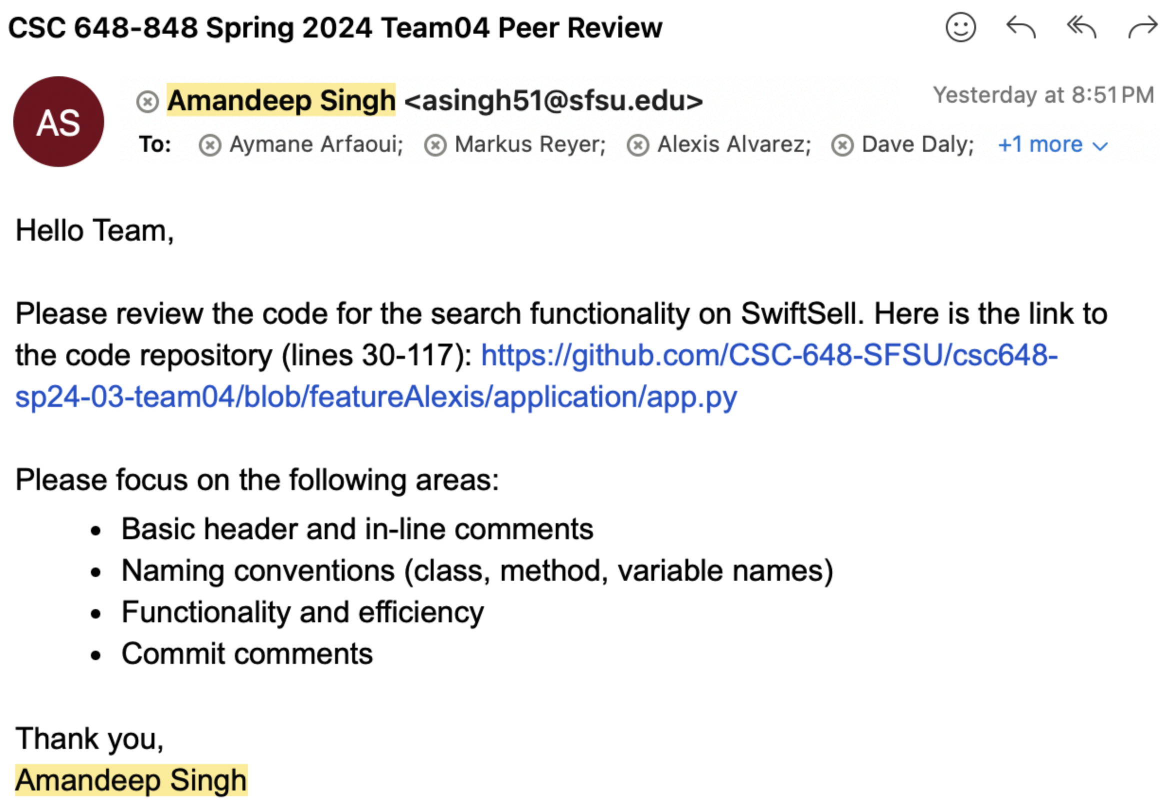
Feature to be Tested

- The search functionality, including searching by item name and course number, with persistent search input and filters.**QA Test Plan**

Test #	Test Title	Test Description	Test Input	Expected Correct Output	Test Results (PASS/FAIL)
1	Basic Search Query	Enter "Book" in the search field and verify that the results are relevant to textbooks	On the home page of SwiftSell, go to the search bar located at the top of the page. Enter "Books" in the search field, and then click the "Search" button to the right of the search bar.	2 items related to books are displayed and the search input persists	Chrome: PASS / Safari: PASS
2	Advanced Search Query	Enter "Book" in the search field and verify that the results are relevant and filtered correctly from lowest to highest price	On the home page of SwiftSell, go to the search bar located at the top of the page. Enter "Book" in the search field, and then click the "Search" button to the right of the search bar. Next, locate the filter under the "Sign Up" page on the right-hand side of the website. In the dropdown menu, select "Sort by price."	Items related to books are sorted from the lowest to highest price and are displayed and the search input persists	Chrome: PASS / Safari: PASS
3	Empty Search	Do not enter anything in the search field and verify that all the results are displayed	On the home page of SwiftSell, go to the search bar located at the top of the page. Do not enter anything in the search field, and then click the "Search" button to the right of the search bar. Next, locate the filter under the "Sign Up" page on the right-hand side of the website. In the	All the items will be returned, up to a maximum of 10 items.	Chrome: PASS / Safari: PASS

			dropdown menu, select "Sort by price."		
4	Search by Course Number	Enter "CSC415" in the search field and verify that the results are relevant to CSC415 course materials	On the home page of SwiftSell, go to the search bar located at the top of the page. Enter "CSC415" in the search field, and then click the "Search" button to the right of the search bar.	2 Items related to CSC415 are displayed and the search input persists	Chrome: PASS / Safari: PASS
5	Persistent Search Functionality	Select the category "furniture" and then type "brown" to see if the category filter stays persistent and only furniture items that contain the keyword "brown" are displayed.	On the home page of SwiftSell, go to the search bar located at the top of the page. Select the category "furniture" and then type "brown" in the search field. Click the "Search" button to see if the category filter stays persistent and only furniture items that contain the keyword "brown" are displayed. Navigate to another page on the website, then return to the search results page to verify that the search input and filter persist.	Items related to the input "furniture" are displayed, showing only furniture that are brown, and the applied filter remains in the search field.	Chrome: PASS / Safari: PASS

4. Peer Code Review



CSC 648-848 Spring 2024 Team04 Peer Review

Amandeep Singh <asingh51@sfsu.edu> Yesterday at 8:51 PM

To: Aymane Arfaoui; Markus Reyer; Alexis Alvarez; Dave Daly; +1 more

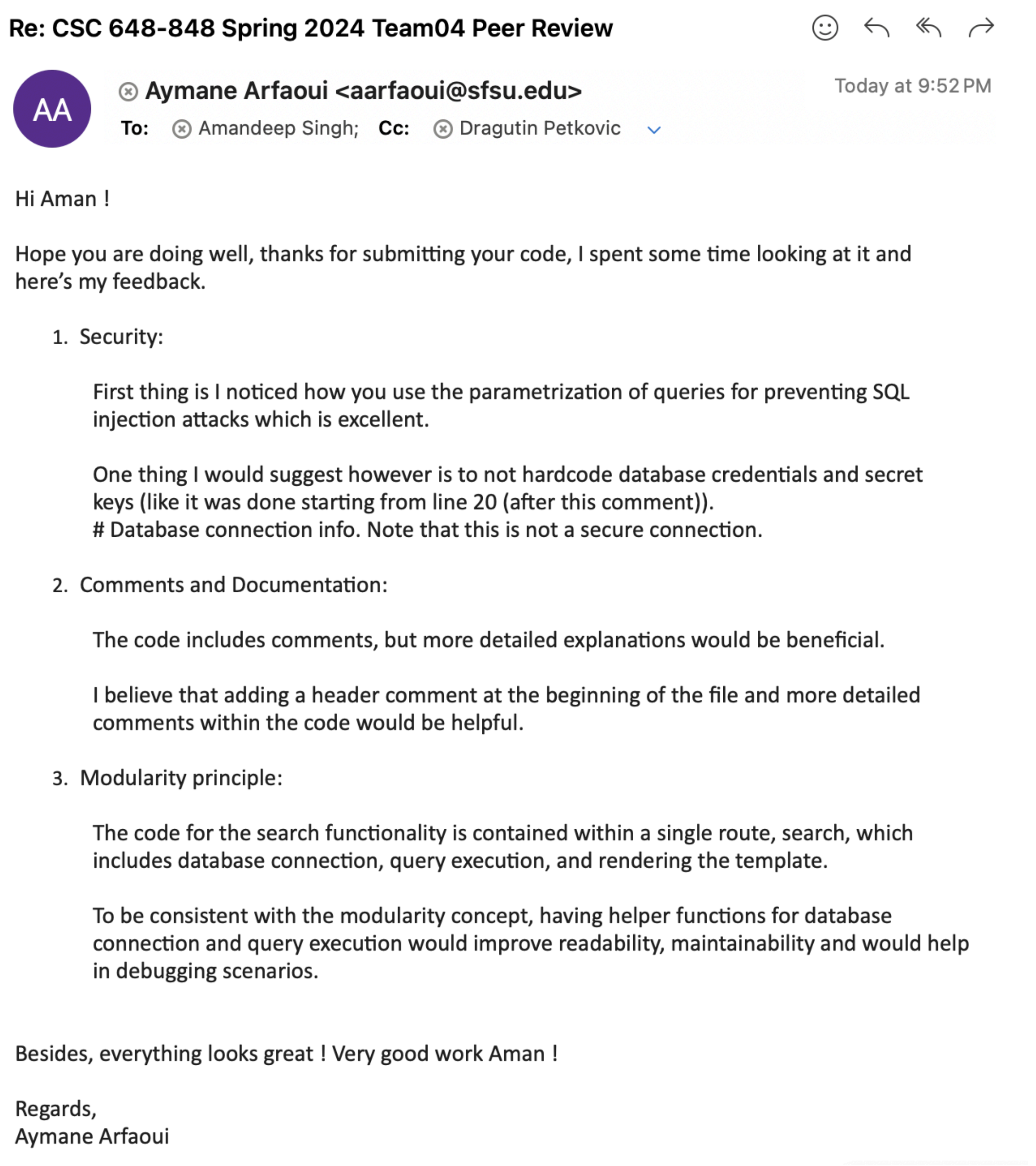
Hello Team,

Please review the code for the search functionality on SwiftSell. Here is the link to the code repository (lines 30-117): https://github.com/CSC-648-SFSU/csc648-sp24-03-team04/blob/featureAlexis/application/app.py

Please focus on the following areas:

- Basic header and in-line comments
- Naming conventions (class, method, variable names)
- Functionality and efficiency
- Commit comments

Thank you,
Amandeep Singh



Re: CSC 648-848 Spring 2024 Team04 Peer Review

Aymane Arfaoui <aarfaoui@sfsu.edu>

Today at 9:52 PM

To: Amandeep Singh; **Cc:** Dragutin Petkovic

Hi Aman !

Hope you are doing well, thanks for submitting your code, I spent some time looking at it and here's my feedback.

1. Security:

   First thing is I noticed how you use the parametrization of queries for preventing SQL injection attacks which is excellent.

   One thing I would suggest however is to not hardcode database credentials and secret keys (like it was done starting from line 20 (after this comment)).
   # Database connection info. Note that this is not a secure connection.

2. Comments and Documentation:

   The code includes comments, but more detailed explanations would be beneficial.

   I believe that adding a header comment at the beginning of the file and more detailed comments within the code would be helpful.

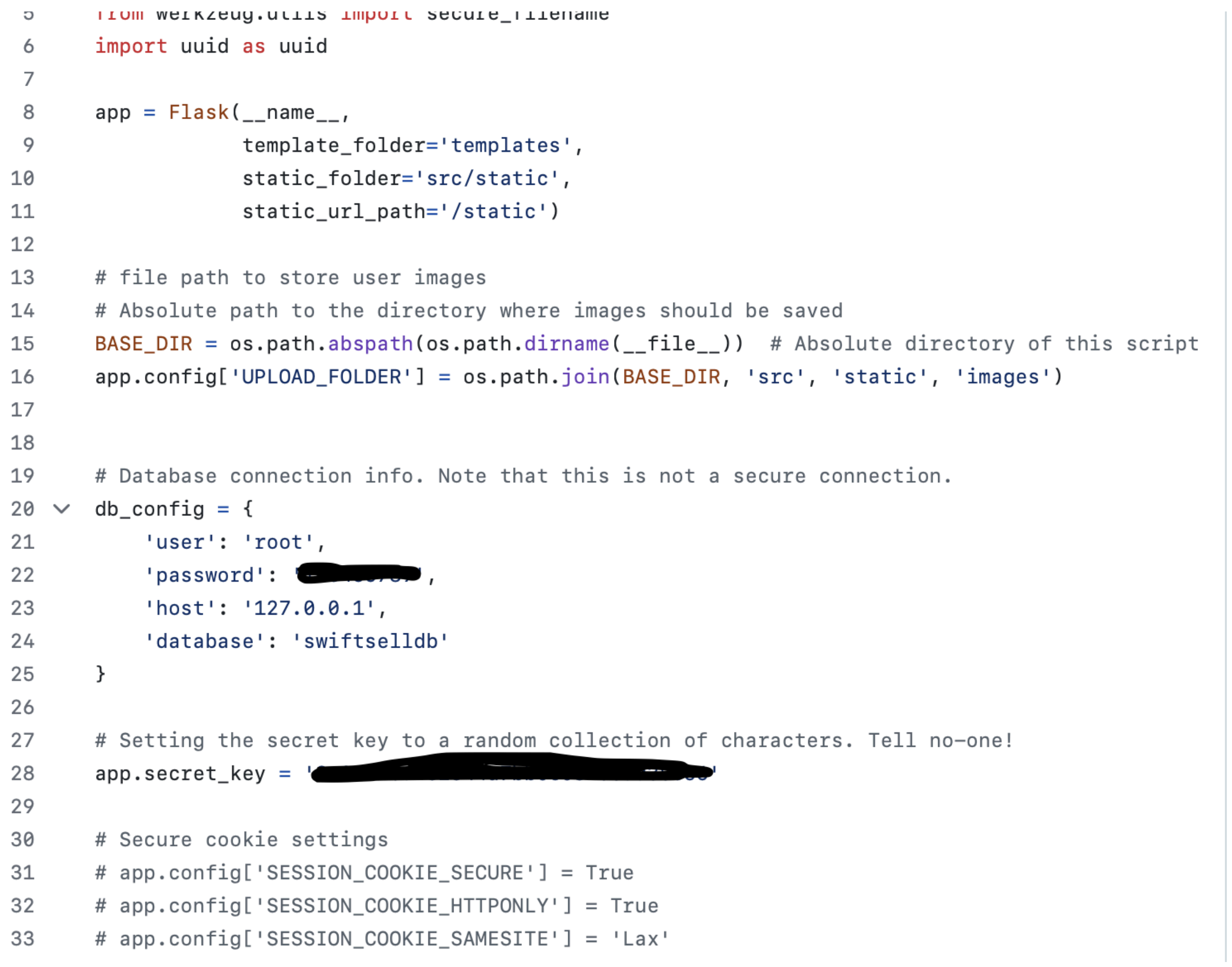
3. Modularity principle:

   The code for the search functionality is contained within a single route, search, which includes database connection, query execution, and rendering the template.

   To be consistent with the modularity concept, having helper functions for database connection and query execution would improve readability, maintainability and would help in debugging scenarios.

Besides, everything looks great ! Very good work Aman !

Regards,
Aymane Arfaoui



```python
from werkzeug.utils import secure_filename
import uuid as uuid

app = Flask(__name__,
            template_folder='templates',
            static_folder='src/static',
            static_url_path='/static')

# file path to store user images
# Absolute path to the directory where images should be saved
BASE_DIR = os.path.abspath(os.path.dirname(__file__))  # Absolute directory of this script
app.config['UPLOAD_FOLDER'] = os.path.join(BASE_DIR, 'src', 'static', 'images')


# Database connection info. Note that this is not a secure connection.
db_config = {
    'user': 'root',
    'password': [redacted],
    'host': '127.0.0.1',
    'database': 'swiftselldb'
}

# Setting the secret key to a random collection of characters. Tell no-one!
app.secret_key = [redacted]

# Secure cookie settings
# app.config['SESSION_COOKIE_SECURE'] = True
# app.config['SESSION_COOKIE_HTTPONLY'] = True
# app.config['SESSION_COOKIE_SAMESITE'] = 'Lax'
```

5. Security Best Practices

Major Assets and Their Protection

Asset	Major Threats	Protection Strategy
User confidential data such as passwords and images	Unauthorized access, data breaches, identity theft	Encrypt data at rest and in transit, implement strong authentication mechanisms, and regularly update security protocols, relative paths for images.
Transaction information	Unauthorized access, data manipulation	Use secure connections (HTTPS), validate and sanitize all inputs, and log and monitor transactions for unusual activity
Application code	SQL injection	Validate all inputs, use parameterized queries, and perform regular code reviews and security testing Search bar also prevent form submission if non-alphanumeric characters are present and only allows up to 40 characters.
User sessions	unauthorized access	Use secure cookies, implement session timeouts.
Trademarked and Copyrighted Content	Stolen content	Add our own copyright and/or trademark to our original materials; apply for trademark protection, if appropriate

User Data Protection

1. **Password Encryption**
    - generate_password_hash and check_password_hash from werkzeug.security for hashing passwords.
2. **Authentication**
    - **Method**: Implement strong password policies, including requirements for complexity and regular updates.

Transaction Information Protection

1. **Secure Transmission**
    - **Method**: Ensure all transactions are conducted over HTTPS to protect data in transit.
2. **Input Validation**
    - **Method**: Validate and sanitize all transaction inputs to prevent injection attacks.

Application Code Protection

1. **Code Reviews**
    - **Method**: Perform regular code reviews and use static analysis tools to detect security vulnerabilities.
2. **Queries with Placeholders**
    - **Method**: parameterized queries with placeholders (e.g., %s) for all database queries

User Session Protection

1. **Secure Cookies**
    - **Method**: Use secure and HttpOnly cookies to protect session data.
2. **Session Management**
    - **Method**: Implement session timeouts and invalidate sessions after a period of inactivity.

6. Adherence to Original Non-functional Specs

1. Copy of All Original 17 Non-functional Requirements

1. Tools and Servers: Application shall be developed, tested, and deployed using tools and servers approved by Class CTO and as agreed in M0.

2. Desktop Optimization: Application shall be optimized for standard desktop/laptop browsers, e.g., must render correctly on the two latest versions of two major browsers.

3. Mobile Optimization: All or selected application functions shall render well on mobile devices.

4. Data Storage: Data shall be stored in the database on the team's deployment server.

5. Concurrent Users: No more than 50 concurrent users shall be accessing the application at any time.

6. Privacy Protection: Privacy of users shall be protected.

7. Language: The language used shall be English (no localization needed).

8. Usability: Application shall be very easy to use and intuitive.

9. Architecture Patterns: Application shall follow established architecture patterns.

10. Code Inspection and Maintenance: Application code and its repository shall be easy to inspect and maintain.

11. Google Analytics: Google Analytics shall be used.

12. Messaging: No e-mail clients shall be allowed. Interested users can only message sellers via in-site messaging. One round of messaging (from user to seller) is enough for this application.

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

14. Site Security: Basic best practices shall be applied (as covered in the class) for main data items.

15. Media Formats: Media formats shall be standard as used in the market today.

16. Modern SE Processes and Tools: Modern SE processes and tools shall be used as specified in the class, including collaborative and continuous SW development and GenAI tools.

17. UI Text: The application UI (WWW and mobile) shall prominently display the following exact text on all pages: "SFSU Software Engineering Project CSC 648-848, Spring 2024. For Demonstration Only" at the top of the WWW page Nav bar.

2. Status of Each Requirement

Tools and Servers: Application shall be developed, tested, and deployed using tools and servers approved by Class CTO and as agreed in M0.
Status: DONE

Desktop Optimization: Application shall be optimized for standard desktop/laptop browsers, e.g., must render correctly on the two latest versions of two major browsers.
Status: DONE

Mobile Optimization: All or selected application functions shall render well on mobile devices.
Status: ON TRACK

Data Storage: Data shall be stored in the database on the team's deployment server.
Status: DONE

Concurrent Users: No more than 50 concurrent users shall be accessing the application at any time.
Status: DONE

Privacy Protection: Privacy of users shall be protected.
Status: DONE

Language: The language used shall be English (no localization needed).
Status: DONE

Usability: Application shall be very easy to use and intuitive.
Status: DONE

Architecture Patterns: Application shall follow established architecture patterns.
Status: DONE

Code Inspection and Maintenance: Application code and its repository shall be easy to inspect and maintain.
Status: ON TRACK

Google Analytics: Google Analytics shall be used.
Status: ON TRACK

Messaging: No e-mail clients shall be allowed. Interested users can only message sellers via in-site messaging. One round of messaging (from user to seller) is enough for this application.
Status: DONE

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

Site Security: Basic best practices shall be applied (as covered in the class) for main data items.
Status: ON TRACK

Media Formats: Media formats shall be standard as used in the market today.
Status: DONE

Modern SE Processes and Tools: Modern SE processes and tools shall be used as specified in the class, including collaborative and continuous SW development and GenAI tools.
Status: DONE

UI Text: The application UI (WWW and mobile) shall prominently display the following exact text on all pages: "SFSU Software Engineering Project CSC 648-848, Spring 2024. For Demonstration Only" at the top of the WWW page Nav bar.
Status: DONE

7. Use of genAI tools like ChatGPT and copilot

In Milestone 4, we utilized GenAI tools such as ChatGPT, focusing on securing our website and enhancing various aspects of our project involved in QA and usability. These tools were invaluable in guiding our code review process and providing key insights. Additionally, they helped us find effective metrics to evaluate user satisfaction, efficiency, and effectiveness. Below is a detailed description of our usage:

Tools Used: ChatGPT (version 4)

Tasks and Usefulness:

Securing the Website: HIGH

Description: We used ChatGPT to identify potential security vulnerabilities and suggest best practices for securing our website. It provided us with detailed explanations and actionable steps to improve our security measures.

Benefit: ChatGPT's insights helped us strengthen our security protocols, ensuring a safer environment for our users.

Guiding Code Review: MEDIUM

Description: ChatGPT and Copilot were used to review our code, highlighting areas that needed improvement and suggesting optimizations.

Benefit: These tools facilitated a more thorough and efficient code review process, though some recommendations required additional verification.

Evaluating Metrics for Satisfaction, Efficiency, and Effectiveness: HIGH

Description: ChatGPT provided us with relevant metrics and methods to measure user satisfaction, efficiency, and effectiveness. It suggested specific survey questions, evaluation techniques, and key performance indicators.

Benefit: This guidance allowed us to create a comprehensive evaluation plan that accurately measures our project's success.

Examples of Key Prompts and Responses:

Prompt for Security: "What are the best practices for securing a web application built with flask?"

Response: ChatGPT provided a detailed list of practices, including implementing HTTPS, using environment variables for sensitive data, validating user inputs, and setting up proper authentication and authorization mechanisms.

Prompt for Metrics: "What metrics should we use to evaluate user satisfaction and efficiency in a web application?"

Response: ChatGPT suggested using Likert scale surveys for user satisfaction, tracking task completion times, counting the number of user interactions, and measuring the success rate of search queries.

Additional Comments:

By integrating GenAI tools into our workflow, we were able to enhance the security, usability, and overall quality of our project, ensuring a better experience for our users.