SW Engineering CSC648/848
Spring 2024

Milestone 4 - May 22nd, 2024

“From Here to There”
“The SFSU Exclusive Site for Buying and Selling Items”

Team 03

Gurpreet Natt (gnatt@sfsu.edu)
Omar Dajani
Douglas Cheung
Justin Isidro
Gursimran Singh
Gio Jung

Date Submitted	May 22nd, 2024

1. Product Summary:

- From Here To There.

- Welcome to From Here To There, a SFSU-exclusive site for buying and selling items. From Here To There is a unique marketplace that enables all of the SFSU community, whether it is students, faculty, or staff, to use a thorough search function. Through this functionality, users will be able to get exactly what they need when needed, creating an environment of respect, support, and mutual benefit. From class materials to office supplies, all of it is on From Here To There for your convenience. As long as you are affiliated with SFSU or have been in the past, you can use From Here To There with no issues.

- **List of Major Committed Functions:**

-Unregistered users:

1. Shall be able to create/register an account. (only with SFSU email)
2. Shall be able to view the post details, including the title, description, and images.
3. Shall be able to navigate through the site's public sections without logging in.
4. Shall be able to search for listings using text input, a category, and filters.

-Registered users:

5. Shall inherit all functions of unregistered user.
6. Shall be able to login.
7. Shall be able to store the items they've posted in the listings page in the dashboard.
8. Shall be able to communicate with sellers and buyers.
9. Shall be able to edit or delete their own posts.
10. Shall be able to use the dashboard to view their messages and postings.
11. Shall inherit all the functions of a non-registered user.

-Admin:

12. Shall inherit all functions of registered user.
13. Shall be able to access the server and perform maintenance or updates as needed.
14. Shall be required to remove or edit any posts that do not meet site guidelines before they go live.
15. Shall be able to remove users who repeatedly violate the guidelines.

- Our website allows users to rent a product instead of just buying it. Same with letting users put a product for rent instead of just for selling.

- http://ec2-54-215-110-168.us-west-1.compute.amazonaws.com/Index.html

2. Usability test plan:

- **Test Objectives:**
  - Testing to make sure our Search function works as intended
  - Responsiveness, to see how fast searching is
  - Ease of use
  - Accuracy, this is to ensure that the correct search results are displaying

- **Test background and setup:**

  1. System setup/starting point/what HW user needs to have:

The tester will need access to a computer with an internet connection and a modern web browser. No additional hardware is required. The starting point for each test will be the homepage of the application.

  2. Intended users:

SFSU students, faculty, and staff.

3. URL:

http://ec2-54-215-110-168.us-west-1.compute.amazonaws.com/Index.html

4. Test environment:

- Done at home with no cameras present or monitoring. Individuals would be untrained and not shown how to use the search.

- Plan for evaluation of Effectiveness:
    - Participants will be given specific tasks to complete using the search function, such as finding a specific book or locating furniture items.
    - The percentage of successfully completed tasks will indicate the effectiveness of the search feature. The number and types of errors encountered by participants while using the search function will be documented. Fewer errors indicate a more user-friendly search feature.
    - Participants will be asked to complete a post-test questionnaire to rate their satisfaction with the search function through the use of a Likert Scale on Jotform. Questions will focus on ease of use, speed, search results, and overall satisfaction.

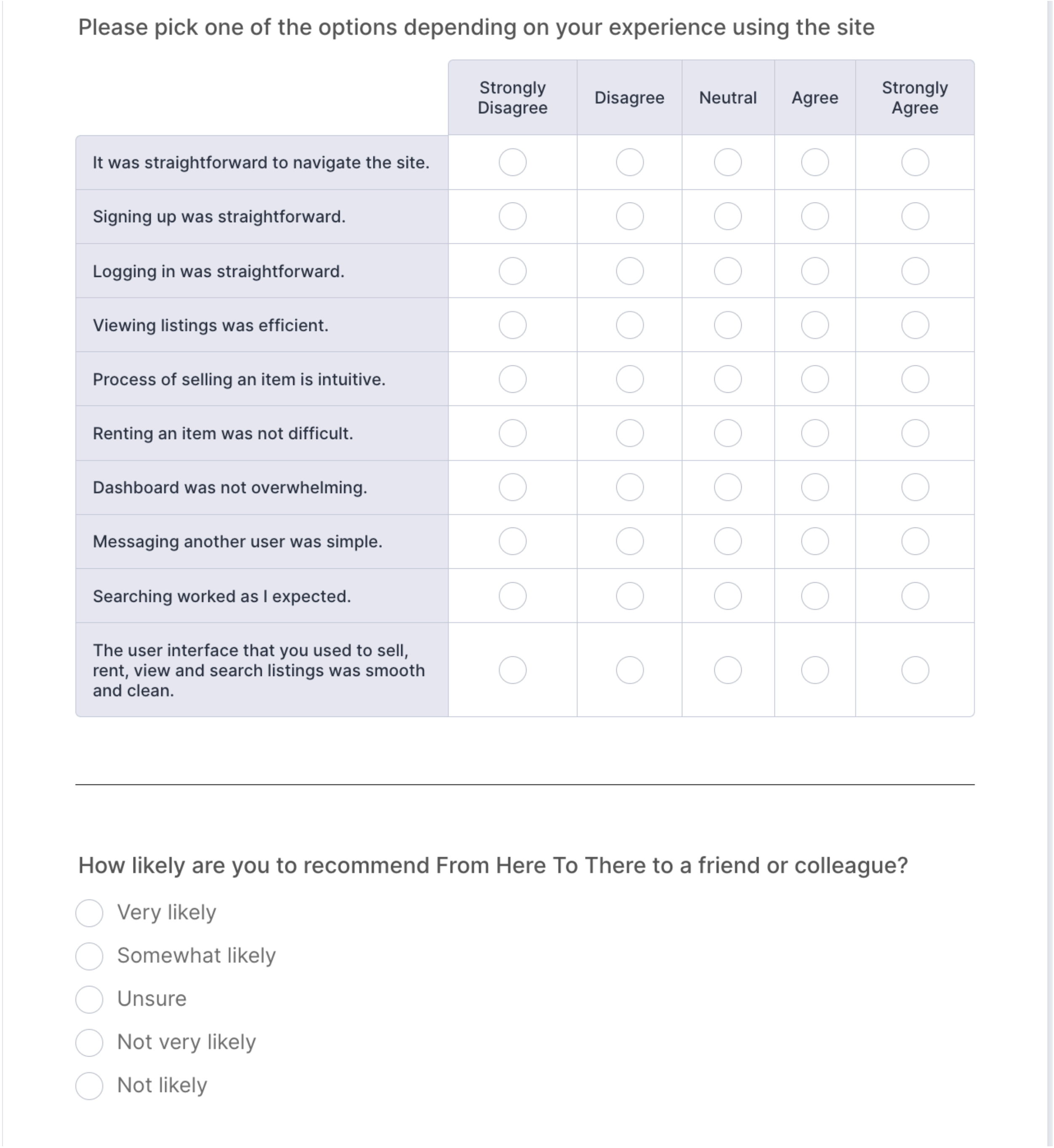
- Plan for evaluation of Efficiency:
    - Record the time taken by participants to complete each search task. Shorter times indicate higher efficiency.
    - Count the number of clicks or interactions needed to complete a task. Fewer clicks suggest a more efficient process.
    - Analyze the navigation paths taken by participants to reach their goals. Direct and straightforward paths indicate higher efficiency.
    - Evaluate the relevance of the first few search results. High relevance reduces the need for extensive searching, indicating greater efficiency.

- Plan for evaluation of user Satisfaction (Likert Scale questionnaire):

A. Usability Task description:

- Users will be given free reign to do whatever they want on From Here To There. We will sit back and watch them do whatever it is that they want to do.
- After all is said done, we will have them fill out the aforementioned Likert Scale made using Jotform (https://www.jotform.com/build/241316062387151?iak=22b3b866700e4f011220cf25c86ee40a-54b84cbfb8eac1e8) so they can record their experiences with the site.
- As mentioned above, there will be a variety of broad questions that cover most if not all of what the users will try and do.

B. Likert Scale evaluation entries:



Please pick one of the options depending on your experience using the site

	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
It was straightforward to navigate the site.	○	○	○	○	○
Signing up was straightforward.	○	○	○	○	○
Logging in was straightforward.	○	○	○	○	○
Viewing listings was efficient.	○	○	○	○	○
Process of selling an item is intuitive.	○	○	○	○	○
Renting an item was not difficult.	○	○	○	○	○
Dashboard was not overwhelming.	○	○	○	○	○
Messaging another user was simple.	○	○	○	○	○
Searching worked as I expected.	○	○	○	○	○
The user interface that you used to sell, rent, view and search listings was smooth and clean.	○	○	○	○	○

How likely are you to recommend From Here To There to a friend or colleague?

○ Very likely

○ Somewhat likely

○ Unsure

○ Not very likely

○ Not likely

3. QA test plan and QA testing:

- Test objectives:
  The primary objective of this QA test plan is to verify the functionality, performance, and compatibility of the search function. We aim to ensure

that the search function accurately returns relevant results, operates efficiently, and is compatible across different web browsers.

- HW and SW setup:
  - Computer or mobile device with internet access
  - Modern web browser (Chrome, Firefox, Safari, Edge)
  - Operating System: Windows, macOS, or Linux
  - http://ec2-54-215-110-168.us-west-1.compute.amazonaws.com/Index.html

- Feature to be tested: needs to be done

- QA test plan in table format:
  Browser used in testing: Chrome

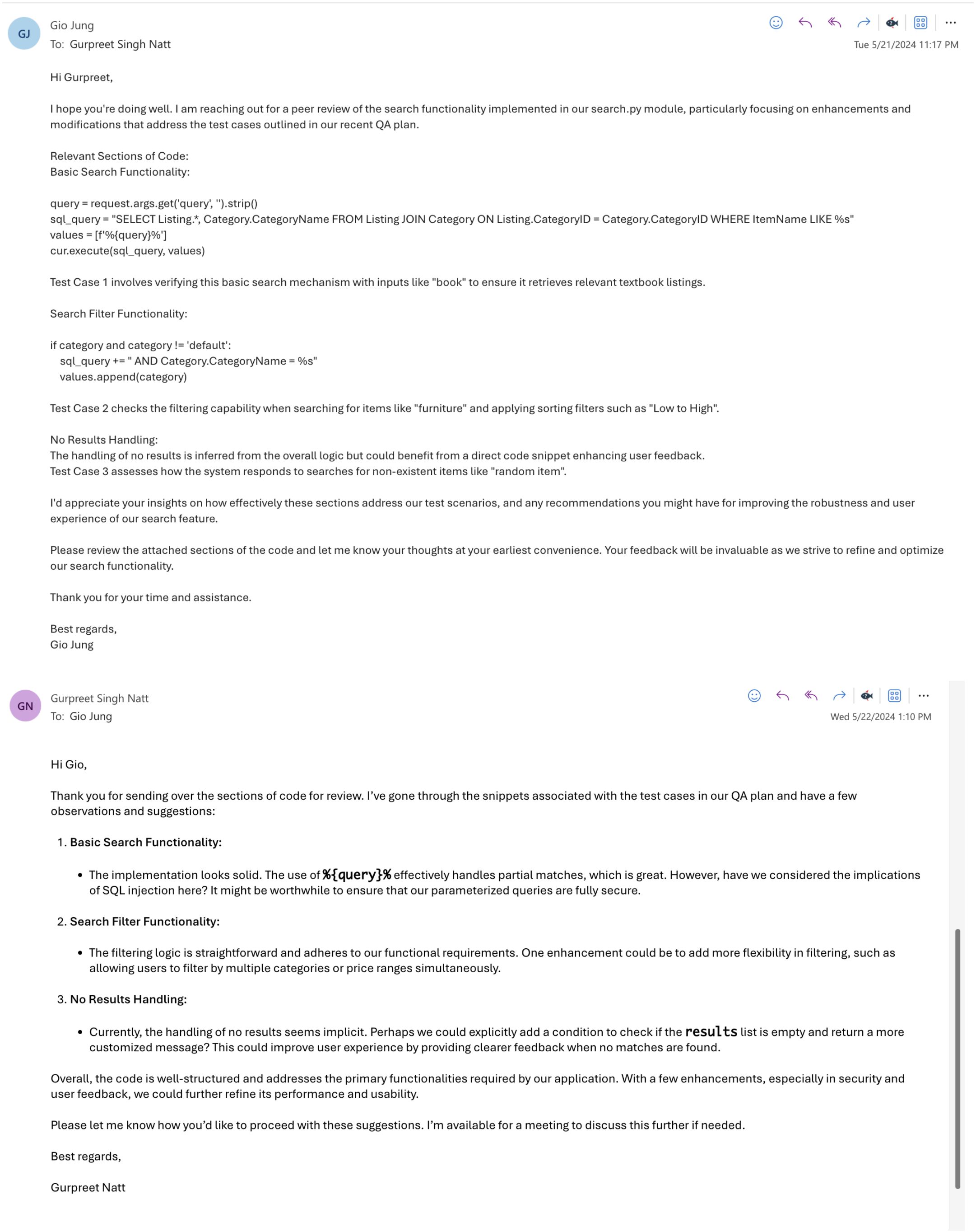
Test #	Test title	Test description	Test input	Expected correct output	Test results
1	Basic Search Function	Verify basic search functionality	Search for "book"	Relevant textbooks listed with correct details	PASS
2	Search Filter	Verify search filter functionality	Search for "furniture", apply price sort filter “Low to High”	Furniture items are displayed from Lowest to Highest price.	PASS
3	No Results Handling	Verify system's response to no search	Search for "random item"	Message indicating no items found	PASS

		results			

Browser Used: Firefox

Test #	Test title	Test description	Test input	Expected correct output	Test results
1	Basic Search Function	Verify basic search functionality	Search for "book"	Relevant textbooks listed with correct details	PASS
2	Search Filter	Verify search filter functionality	Search for "furniture", apply price sort filter “Low to High”	Furniture items are displayed from Lowest to Highest price.	PASS
3	No Results Handling	Verify system's response to no search results	Search for "random item"	Message indicating no items found	PASS

4. Peer code review:



GJ Gio Jung
To: Gurpreet Singh Natt
Tue 5/21/2024 11:17 PM

Hi Gurpreet,

I hope you're doing well. I am reaching out for a peer review of the search functionality implemented in our search.py module, particularly focusing on enhancements and modifications that address the test cases outlined in our recent QA plan.

Relevant Sections of Code:
Basic Search Functionality:

```
query = request.args.get('query', '').strip()
sql_query = "SELECT Listing.*, Category.CategoryName FROM Listing JOIN Category ON Listing.CategoryID = Category.CategoryID WHERE ItemName LIKE %s"
values = [f'%{query}%']
cur.execute(sql_query, values)
```

Test Case 1 involves verifying this basic search mechanism with inputs like "book" to ensure it retrieves relevant textbook listings.

Search Filter Functionality:

```
if category and category != 'default':
   sql_query += " AND Category.CategoryName = %s"
   values.append(category)
```

Test Case 2 checks the filtering capability when searching for items like "furniture" and applying sorting filters such as "Low to High".

No Results Handling:
The handling of no results is inferred from the overall logic but could benefit from a direct code snippet enhancing user feedback.
Test Case 3 assesses how the system responds to searches for non-existent items like "random item".

I'd appreciate your insights on how effectively these sections address our test scenarios, and any recommendations you might have for improving the robustness and user experience of our search feature.

Please review the attached sections of the code and let me know your thoughts at your earliest convenience. Your feedback will be invaluable as we strive to refine and optimize our search functionality.

Thank you for your time and assistance.

Best regards,
Gio Jung

GN Gurpreet Singh Natt
To: Gio Jung
Wed 5/22/2024 1:10 PM

Hi Gio,

Thank you for sending over the sections of code for review. I've gone through the snippets associated with the test cases in our QA plan and have a few observations and suggestions:

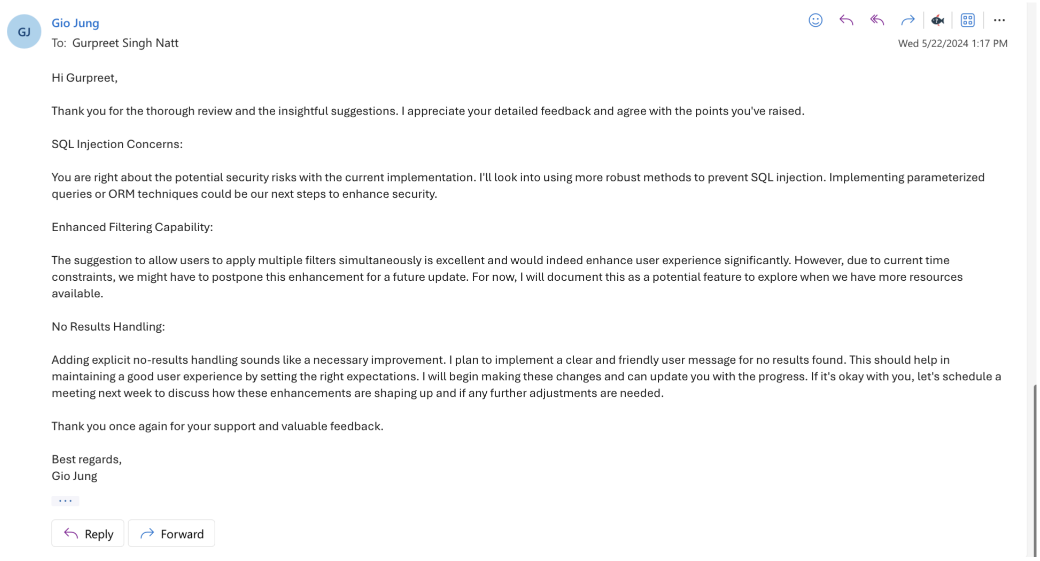
1. **Basic Search Functionality:**
   - The implementation looks solid. The use of `%{query}%` effectively handles partial matches, which is great. However, have we considered the implications of SQL injection here? It might be worthwhile to ensure that our parameterized queries are fully secure.
2. **Search Filter Functionality:**
   - The filtering logic is straightforward and adheres to our functional requirements. One enhancement could be to add more flexibility in filtering, such as allowing users to filter by multiple categories or price ranges simultaneously.
3. **No Results Handling:**
   - Currently, the handling of no results seems implicit. Perhaps we could explicitly add a condition to check if the `results` list is empty and return a more customized message? This could improve user experience by providing clearer feedback when no matches are found.

Overall, the code is well-structured and addresses the primary functionalities required by our application. With a few enhancements, especially in security and user feedback, we could further refine its performance and usability.

Please let me know how you'd like to proceed with these suggestions. I'm available for a meeting to discuss this further if needed.

Best regards,

Gurpreet Natt



Gio Jung
To: Gurpreet Singh Natt

Wed 5/22/2024 1:17 PM

Hi Gurpreet,

Thank you for the thorough review and the insightful suggestions. I appreciate your detailed feedback and agree with the points you've raised.

SQL Injection Concerns:

You are right about the potential security risks with the current implementation. I'll look into using more robust methods to prevent SQL injection. Implementing parameterized queries or ORM techniques could be our next steps to enhance security.

Enhanced Filtering Capability:

The suggestion to allow users to apply multiple filters simultaneously is excellent and would indeed enhance user experience significantly. However, due to current time constraints, we might have to postpone this enhancement for a future update. For now, I will document this as a potential feature to explore when we have more resources available.

No Results Handling:

Adding explicit no-results handling sounds like a necessary improvement. I plan to implement a clear and friendly user message for no results found. This should help in maintaining a good user experience by setting the right expectations. I will begin making these changes and can update you with the progress. If it's okay with you, let's schedule a meeting next week to discuss how these enhancements are shaping up and if any further adjustments are needed.

Thank you once again for your support and valuable feedback.

Best regards,
Gio Jung

5. Self-check on security practices:

Asset to be protected:	Types of possible/expected attacks:	Consequence of security breach:	Our strategy to mitigate/protect the asset:
The user database	Unauthorized user makes system unavailable	Disruption of critical functions	Require authentication, use data encryption
An individual user record	Unauthorized user gains access to other user's confidential data	Leak of personal info into unknown hands	Validate inputs, encrypt data
The information system	System Confidentiality and Integrity	Restoring system, Communication	Grant minimal permissions, use data encryption

- Do we encrypt PW in the DB?

  Yes we do.

- Is there input data validation?

  Yes there is.

6. Self-check in 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 M0: *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 shall render well on mobile devices: *DONE.*
4. Data shall be stored in the database on the team's deployment 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: *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 shall 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. 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.*
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. Media formats shall be standard as used in the market today: *DONE.*
16. Modern SE processes and tools shall be used as specified in the class, including collaborative and continuous SW development and GenAI tools: *DONE.*
17. 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. (Important so as to not confuse this with a real application): *DONE*.

7. Use of GenAI tools:

- *Omar*:

  GenAI and version of it used: ChatGPT 3.5.
  Tasks Used: Used it for Tasks 2 and 3.
  Effectiveness: Really effective.
  Details: While I think I understood the differences between QA Testing and Usability Testing thanks to the slides, I just wanted to be absolutely safe. Thanks to ChatGPT, I got exactly what I needed.
  Examples: Not only did ChatGPT send me paragraphs in response to me asking what the differences between both types of tests are, it also sent me a ton and I mean a ***ton*** of examples to cement my understanding. My team and I already knew what we wanted to be tested so we didn't "borrow" these ideas or whatever, but it was still helpful.
  Comments: Thank you ChatGPT for being this class's third Professor.

- *Gurpreet*:

  GenAI and version of it used: ChatGPT 4o
  Tasks it was used for and its effectiveness: Used for QA Test Plan, effective.
  Details:
  Test Objectives: AI helped articulate the primary goal of the QA test plan, focusing on verifying the functionality, performance, and compatibility of the search function.
  HW and SW Setup: AI provided precise requirements for the hardware and software setup necessary to execute the tests, ensuring that the environment was properly configured.
  Feature to be Tested: AI identified and clearly defined the specific feature (the search function) that required testing.
  Examples:
  Using AI expedited the creation of the test plan, saving time and ensuring that all critical aspects were covered thoroughly.