DineQuest

By Zack Wedding

Cybersecurity (B.S.)

Prof. Julie Henderson

# Statement of Purpose

Just about everyone has used the internet to try and find somewhere to eat, whether it be fast food, somewhere fancy, or maybe even a food truck. When doing this, it can often be hard to narrow options down to a single restaurant or two. This is an important issue because it can often lead to wasted time or somebody not happy. Hardly anything is more valuable than time, and nobody wants to see a close friend or family member upset because they don’t want to eat at a certain restaurant. This issue can hopefully be solved by the website in which I create. This website will allow users to find restaurants based on certain criteria and book a reservation at a restaurant of their selection. This will not only allow users to choose a restaurant that everybody wants based on some criteria, but it will also allow them to plan out the meal in advance. The cybersecurity side of a website can often be overlooked. I will ensure that this does not happen and take a deep dive into the cybersecurity aspect. The problem of having a hard time choosing a restaurant or not choosing a restaurant that everybody likes can be solved with this restaurant booking website.

# Research and Background

I chose this topic because of my love for food, my annoyance with people who cannot make decisions, and eagerness to expand my knowledge on web development. The biggest thing that I had to research was by touching up my HTML and JavaScript. Previously, I had been familiar with both languages. However, it had been a while, so I had to go back and touch up on them. I did this via W3Schools. I also had to do some research into which API I should use to fetch restaurant data. The Yelp Fusion API was the only one that I could seem to get for free, which is why I ended up going with it. Aside from these, there were not many things that I had to do some research on.

# Project Language, Hardware, and Software

HTML, Javascript, CSS

Yelp Fusion API, cdnjs, DOMPurify

GitHub Pages, Visual Studio Code

# Project Requirements

|  |  |
| --- | --- |
| ID Number | 01 |
| Type | Functionality |
| Description | The website must be accessible to website users. This will be done by hosting the website on GitHub pages. |
| Rationale | Without being accessible, nobody will be able to use the website and it will be pointless. |
| Fit Criterion | Users can connect to the website. |
| Priority | High |
| Dependencies | N/A |

|  |  |
| --- | --- |
| ID Number | 02 |
| Type | Functionality |
| Description | Website buttons should work as expected |
| Rationale | As with every website, the buttons should be taking customers to where they expect to be taken |
| Fit Criterion | Buttons give customers correct results |
| Priority | High |
| Dependencies | 03, 04, 9, 10, 11 |

|  |  |
| --- | --- |
| ID Number | 03 |
| Type | Functionality |
| Description | The website must be able to get information about nearby restaurants and display that information. |
| Rationale | With any information about restaurants, customers won’t be able to filter them out based on their criteria. |
| Fit Criterion | Information about nearby restaurants is present when filtering them out. |
| Priority | High |
| Dependencies | 10 |

|  |  |
| --- | --- |
| ID Number | 04 |
| Type | Functionality |
| Description | Website users must be able to filter restaurants. |
| Rationale | This is one of the purposes of the website in the first place: pick a restaurant based on a list of criteria. |
| Fit Criterion | Users can filter restaurants based on certain criteria. |
| Priority | High |
| Dependencies | 03, 10 |

|  |  |
| --- | --- |
| ID Number | 05 |
| Type | Usability |
| Description | The website should be user friendly. |
| Rationale | If the website is not user friendly, customers may get frustrated with it and not want to use it. Also, if customers can’t figure out how to use the website then they will not be able to. |
| Fit Criterion | Customers can browse through the website with ease and are able to do certain actions (such as place an order or book a table). |
| Priority | Medium |
| Dependencies | 02, 06, 11, 15 |

|  |  |
| --- | --- |
| ID Number | 06 |
| Type | Look and feel |
| Description | The website should appear professional. |
| Rationale | If the website does not appear professional, it can turn people away. Some customers may not trust the website if it does not look good. |
| Fit Criterion | Customers are not turned away by the appearance of the website. |
| Priority | Low |
| Dependencies | 02, 05, 08 |

|  |  |
| --- | --- |
| ID Number | 07 |
| Type | Security |
| Description | A security plan should be created. |
| Rationale | This security plan gives a good idea of what should be done to meet security needs. |
| Fit Criterion | A security plan document outlining security needs is successfully created. |
| Priority | High |
| Dependencies | 08, 12, 13, 14 |

|  |  |
| --- | --- |
| ID Number | 08 |
| Type | Security |
| Description | An SSL certificate should be obtained. |
| Rationale | By obtaining an SSL certificate, the website can use HTTPS, which offers extra security. |
| Fit Criterion | The web browser uses HTTPS. |
| Priority | High |
| Dependencies | 01 |

|  |  |
| --- | --- |
| ID Number | 09 |
| Type | Functionality |
| Description | The website should be able to track the location of a customer. |
| Rationale | Getting the location of a customer is essential for finding restaurants near them. |
| Fit Criterion | Customers can successfully search for restaurants near them based on their location. |
| Priority | Medium |
| Dependencies | N/A |

|  |  |
| --- | --- |
| ID Number | 10 |
| Type | Functionality |
| Description | Yelp Fusion API should be working properly |
| Rationale | This API is what is giving us the results. Without the API, the website does not work. |
| Fit Criterion | Restaurants are successfully fetched |
| Priority | High |
| Dependencies | N/A |

|  |  |
| --- | --- |
| ID Number | 11 |
| Type | Functionality |
| Description | Website users must be able to book a reservation at a restaurant of their choosing. |
| Rationale | Another key function of the website is that customers are able to reserve a table at a restaurant through the website. |
| Fit Criterion | Customers can successfully reserve a table at a restaurant. |
| Priority | High |
| Dependencies | 03, 10 |

|  |  |
| --- | --- |
| ID Number | 12 |
| Type | Security |
| Description | DOMPurify should sanitize inputs. |
| Rationale | Sanitizing user input adds some security against cross-site scripting attacks. |
| Fit Criterion | Tests to enter malicious scripts do not execute. |
| Priority | High |
| Dependencies | 16 |

|  |  |
| --- | --- |
| ID Number | 13 |
| Type | Security |
| Description | Conduct a security audit |
| Rationale | This will help identify vulnerabilities |
| Fit Criterion | The security audit is successfully conducted |
| Priority | High |
| Dependencies | N/A |

|  |  |
| --- | --- |
| ID Number | 14 |
| Type | Security |
| Description | Bot protection should be implemented |
| Rationale | This can protect against unwanted bots and excessive API calls |
| Fit Criterion | Web crawlers that respect a robots.txt file do not crawl |
| Priority | Medium |
| Dependencies | N/A |

|  |  |
| --- | --- |
| ID Number | 15 |
| Type | Look and feel |
| Description | Website template via HTML5 is used |
| Rationale | This will help make the website look professional |
| Fit Criterion | Website is successfully created from a HTML5 template |
| Priority | Medium |
| Dependencies | N/A |

|  |  |
| --- | --- |
| ID Number | 16 |
| Type | Functionality |
| Description | The content delivery network cdnjs should be used |
| Rationale | This allows us to use DOMPurify |
| Fit Criterion | DOMPurify is successfully loaded |
| Priority | High |
| Dependencies | N/A |

# Project Implementation and Description

Once users navigate to the website, they are greeted by a welcome page (Fig. 1) that explains a little bit about the website and what it does. From here, users have the option to go to the section for either narrowing or booking. They can do this by clicking the buttons at the top of the page or by clicking the buttons after the welcome message (Fig. 2). On the narrowing page, users have five different options to narrow down restaurants from (Fig. 3). These options include cuisine type, location, price range, minimum rating, and restaurants that are open now. Cuisine types include American, Chinese, French, Indian, Italian, Japanese, Mexican, Thai, and Vietnamese (Fig. 4). Price range includes $ (under $10 per person), $$ ($11 - $30 per person), $$$ ($31 - $60 per person), and $$$$ (over $61 per person) (Fig. 5). There are integer options for the minimum rating (Fig. 6) and two options for whether restaurants are open now (Fig. 7). If the user leaves the location field empty, they will be asked if their location can be used (Fig. 8). After entering criteria to narrow down restaurants, a maximum of ten restaurants are displayed, along with their rating, price range, address, and phone number (Fig. 9). These restaurants either match the criteria or almost match the criteria. Once users are on the booking page, only two fields are shown: restaurant name and location (Fig. 10). As with the narrowing page, a maximum of ten restaurants are displayed (Fig. 11). The restaurant that the user entered is displayed at the top along with similar restaurants following. After clicking the “Book Now” button in the results, users are redirected to the restaurant’s Yelp page (Fig. 12). From here, users can book a reservation as they would like. At the bottom of each page is a footer that gives credit to HTML5 for the website template (Fig. 13).

Here is the source code repository link: <https://github.com/zwedding/SeniorProject.github.io>

**A screenshot of a computer

AI-generated content may be incorrect.**

*Fig 1. Welcome page*

*A screenshot of a website

AI-generated content may be incorrect.*

*Fig 2. Narrow and book buttons at the bottom of the welcome page*

*A screenshot of a computer

AI-generated content may be incorrect.*

*Fig 3. Five different options to narrow from*

*A screenshot of a computer

AI-generated content may be incorrect.*

*Fig 4. Different cuisine options*

*A screenshot of a computer

AI-generated content may be incorrect.*

*Fig 5. Different price ranges*

*A screenshot of a computer

AI-generated content may be incorrect.*

*Fig 6. Different minimum ratings*

*A screenshot of a computer

AI-generated content may be incorrect.*

*Fig 7. Open now options*

*A screenshot of a computer

AI-generated content may be incorrect.*

*Fig 8. Browser requests the user’s location*

*A screenshot of a computer

AI-generated content may be incorrect.*

*Fig 9. Results from narrowing*

*A screenshot of a computer

AI-generated content may be incorrect.*

*Fig 10. The booking page*

*A screenshot of a computer

AI-generated content may be incorrect.*

*Fig 11. Results from booking*

*A screenshot of a food website

AI-generated content may be incorrect.*

*Fig 12. User is redirected to the restaurant’s Yelp page*

*A screenshot of a computer

AI-generated content may be incorrect.*

*Fig 13. HTML5 credit*

# Test Plan

Objective: This test plan has been created to ensure that all functionalities within the website are working properly and that the project meets requirements.

Scope: Searching, filtering, buttons on all three pages, displayed results, error handling, location services, security features

Approach: All testing will be done manually. I will use many different restaurants and locations along with many different inputs (price range, rating, open now/not, location). I will allow family and friends to use the website to make sure that it works well across different locations, operating systems, and browsers. This will likely also test different inputs such as capitalization and apostrophes. Users will use the website and provide feedback on what they like and what they think could be improved. This will be done in the form of a survey.

Deliverables: Test plan, test cases, security plan

Test Environment:

* Devices include mobile phone and PC
* Browsers include Safari, Chrome, and Edge
* Operating systems include iOS, MacOS and Windows

Features to be Tested: Narrowing feature gives valid results based on criteria, booking feature allows users to book a reservation from the website, location works well, ensure the website is secure

Testing Data: I will use several different restaurants from several different locations and ensure that each result is displayed as expected. I will also input random characters to ensure error handling works as it should. Family and friends will use whatever input they would like. Everyone will use whatever criteria they would like to narrow restaurants down.

Entry and Exit Criteria:

* Entry Criteria: All features are implemented, and the website is functional for testing.
* Exit Criteria: All critical issues are resolved, and the feedback is complete.

Risks:

* Yelp API goes down
* GitHub pages goes down
* Vulnerabilities within the website code

# Test Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Action | Input | Expected Output | Actual Output | P/F |
| Narrowing restaurants | Cuisine – American Location – user’s  Price Range - $ Minimum Rating – 3 stars  Open Now: Doesn’t Matter | Restaurants that match criteria | Restaurants that were displayed match the desired criteria | P |
| Booking Restaurants | Restaurant: Texas Roadhouse  Location: user’s | Texas Roadhouse’s in the area (and similar restaurants) | Texas Roadhouse and similar restaurants were displayed | P |
| Get user location | Location input is empty | Browser asks for user’s location | Browser asked for the user’s location | P |
| Do not allow user location | Block access to location | Asks user to enter location manually | User is prompted to enter a manual location | P |
| Try to book a reservation | Click the “book” button on the booking page | Users are directed to Yelp | Users are directed to the restaurants Yelp page | P |
| Sanitizing input | <img src=x onerror=alert('XSS')> | Malicious script is removed | Script was removed by DOMPurify | P |
| API key rotation | New key replaces the old key | Restaurants are fetched | Restaurants are still able to be fetched | P |
| Enter multiple responses quickly | Same input, clicking search multiple times quickly | Users should be stopped | The user is told they must wait before submitting another response | P |
| Insert criteria that should not give 10 results | Niche criteria are entered | Less than ten restaurants are displayed | Only four restaurants are displayed | P |
| Content security policy header | Entered a malicious script in the browsers Dev Tools | Script should not run | CSP stopped the script from running | P |
| Used the website on different devices and operating systems | Used devices such as iPhone, MacBook, and Windows laptop | Website should run as expected | Website runs as normal | P |
| Check capitalization and apostrophes | McDonalds, taco bell | Restaurants should still be displayed | The correct restaurants are still being displayed | P |

# Challenges Overcome

The most annoying and prevalent challenge that I faced came from the pricing of the Yelp API. The cheapest Yelp API plan is $229 per month. I did not want to come up with this kind of money every month, so I had to use several different emails to get free trials. These trials lasted a month each, and I believe I used six or seven different emails. There were a few challenges that I faced that dealt with the security of the site. First, the API key is being exposed within my GitHub repository. I figured that this was fine because I am only doing a small-scale project. Next, GitHub Pages does not support custom headers. I tried to implement several different headers but was not able to do so. The only header that I was able to implement was the content security policy header. I faced multiple challenges outside of the security of the website as well. One of these challenges was that the Yelp Fusion API free (base) tier only provides the URL to a restaurant’s Yelp page rather than their actual website. If I wanted to direct users to a restaurant’s actual website, I would need to purchase a better tier that costs at least $299 per month. Another challenge faced was my limited exposure to both JavaScript and HTML. I have worked with both languages before, but I certainly had to touch up on my knowledge of them. I walked through some tutorials from W3Schools, which helped me out a lot. I faced some small challenges during web development, but nothing that halted my progress for too long.

# Future Enhancements

I do not plan on continuing to enhance my website, as I am not interested in the cost that it would take to continue using the Yelp Fusion API. However, if I was to continue enhancing the website, the first thing I would do it implement a backend server so that the API key is securely stored. As mentioned, it is currently in GitHub for anyone to see. User accounts are another thing that could be added into the future. This could allow users to save their criteria and restaurants. With user accounts, users could track their restaurants easier. Another helpful enhancement for the future is reservation integration. This would be helpful so that users do not have to book through Yelp. Doing this would save them a click or two. Lastly, some suggestions I got from my feedback survey included adding menus and restaurant hours to the results displayed. These would both be interesting to add and would enhance the website, and I would be able to do so with the free tier of the Yelp Fusion API.