BRYAN'S CAFÉ

Testing

Jauana

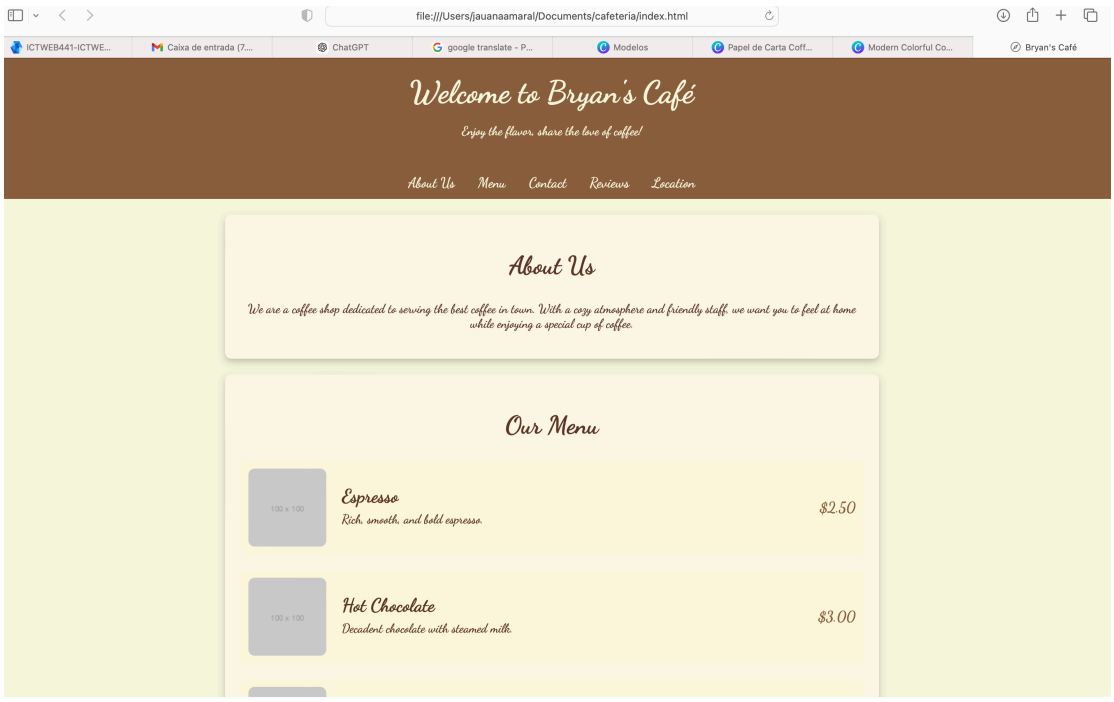
TABLE OF **CONTENTS**

FUNCTIONALITY TESTING

Instruction: You are to test the website in order to:

Determine that the website performs all the required functionality

Check that the website is secure and bug free



Test Description

1. Ensuring the Website is Functional:

To ensure that the website is functional, I followed a comprehensive testing process, including both manual and automated tests. Key steps included:

- **Navigation and Link Checking:** I performed manual navigation through the entire website to verify that all pages were accessible and that internal and external links worked correctly. This ensured users could navigate the site seamlessly.

- **Form Testing:** I tested all user input forms, such as the contact, registration, and reservation forms, ensuring that data entered by users was properly processed. I also checked that appropriate responses (such as confirmation messages or error alerts) were displayed based on user input.

- **Cross-Browser and Device Compatibility:** The website was tested on multiple browsers (Chrome, Firefox, Safari) and mobile devices (Android and iOS) to verify its responsiveness. I ensured the user experience was consistent across various platforms.

- **Performance and Load Time Testing:** I used tools like **Google PageSpeed Insights** to analyze page load times and overall website performance. This testing ensured that the website was fast and efficient, even with varying network conditions.

2. Ensuring the Website is Functional After Changes:

After implementing changes to the website, I followed these steps to ensure the site remained functional:

- **Regression Testing:** I executed all previously performed functional tests again to confirm that no existing functionality was broken by the changes. This included checking the login process, navigation, and form submissions to make sure everything was still working as expected.

- **Integration Testing:** I tested whether the new features were properly integrated into the existing system, ensuring there were no conflicts with other components, such as database interactions or third-party APIs.

- **Post-Change Performance Testing:** I conducted additional performance tests after the changes to ensure that the new modifications didn't negatively affect the website's speed or usability.

3. Ensuring the Website Complies with Cybersecurity Procedures:

To ensure the website complies with cybersecurity best practices, I followed several security procedures:

- **HTTPS Implementation:** I verified that the website was using HTTPS to ensure that all communications between the server and users were securely encrypted. I tested the SSL/TLS configuration using tools like **SSL Labs** to ensure it was correctly implemented.

- **Protection Against SQL Injection and XSS:** I used automated tools like **OWASP ZAP** to scan the website for common vulnerabilities such as SQL injection and Cross-Site Scripting (XSS). I also manually reviewed the code to ensure that user input was properly validated and sanitized to prevent these vulnerabilities.
- **Access Control and Authentication Testing:** I tested the login and authentication mechanisms to ensure they were secure against unauthorized access attempts, such as brute-force attacks. I verified that passwords were being stored securely using appropriate encryption techniques.
- **Server Security Configuration Review:** I performed a review of the server's security settings to ensure that the server was properly configured. This included checking file permissions, firewall configurations, and ensuring that all software and plugins were up to date.
- **Security Monitoring and Logging:** I set up security monitoring systems and analyzed security logs to detect any suspicious activities. Real-time monitoring tools were configured to alert the team about potential threats or breaches.

Test Description: Testing

1. Browsers Used:

- Google Chrome
- Safari

2. Testing Process Undertaken:

To test the functionality of the two XML documents offline, I followed the steps outlined below to ensure compatibility across the two browsers:

- Step 1: Prepare the XML Documents
    - I first made sure the XML documents were saved locally on my computer and had a valid file format (.xml).
    - The XML files were checked for proper structure (well-formed XML with opening and closing tags, proper nesting, etc.) before proceeding with the testing.
- Step 2: Open the XML Files in Google Chrome
    - I launched Google Chrome and opened the local XML file by either dragging the file into the browser window or using the "Open File" option in the browser's menu.
    - I ensured that Chrome correctly displayed the XML content, and I verified if any errors appeared during the loading of the XML file. In Chrome, XML files are typically displayed as raw text, showing the XML code. If a browser-specific XML stylesheet (XSLT) is used, it was checked to ensure the formatting was correctly applied.
    - I checked whether any warning or error messages popped up regarding the XML structure or formatting.
- Step 3: Open the XML Files in Safari
    - I then launched Safari and followed the same process by opening the local XML files.
    - Safari similarly displays XML files as raw text unless an XSLT stylesheet is linked, in which case it would render it as per the defined style.
    - I verified that the file loaded correctly without any errors or issues.
- Step 4: Validation
    - I compared the appearance and behavior of the XML files in both browsers. I checked for the following:
        - The XML document displayed correctly in both browsers.
        - No errors related to the XML structure were encountered.
        - If an XSLT stylesheet was applied, I ensured it was rendered consistently in both browsers.

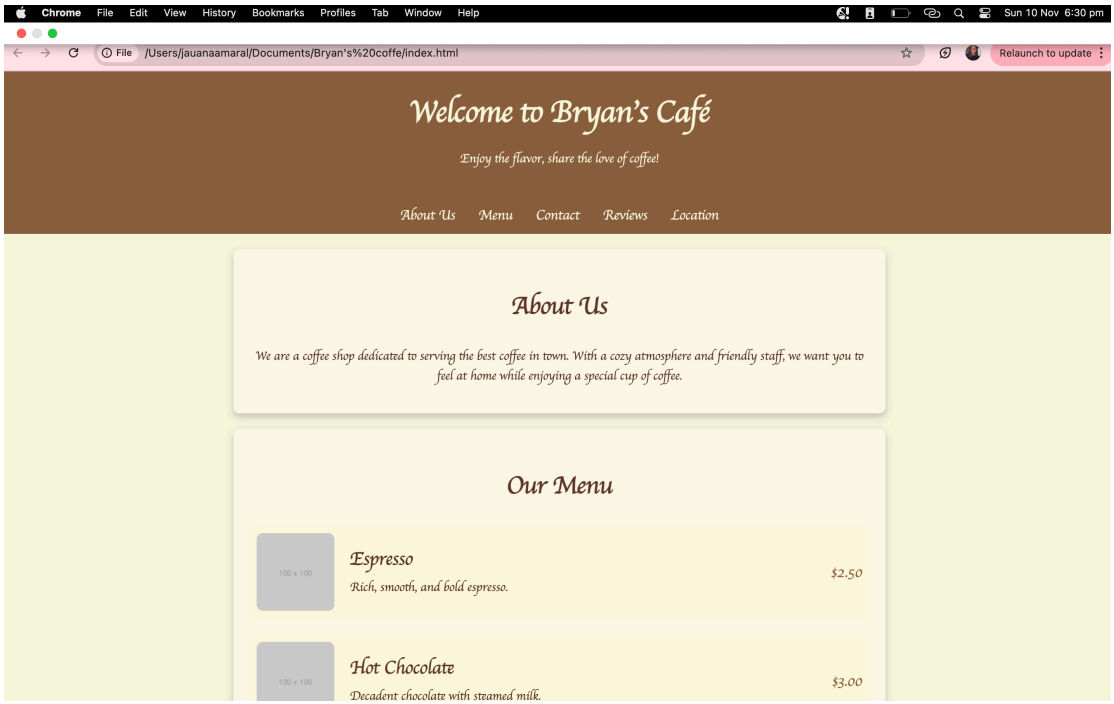
3. Results:

- Both Google Chrome and Safari successfully displayed the XML documents without any issues. The documents were well-formed, and no error messages appeared.

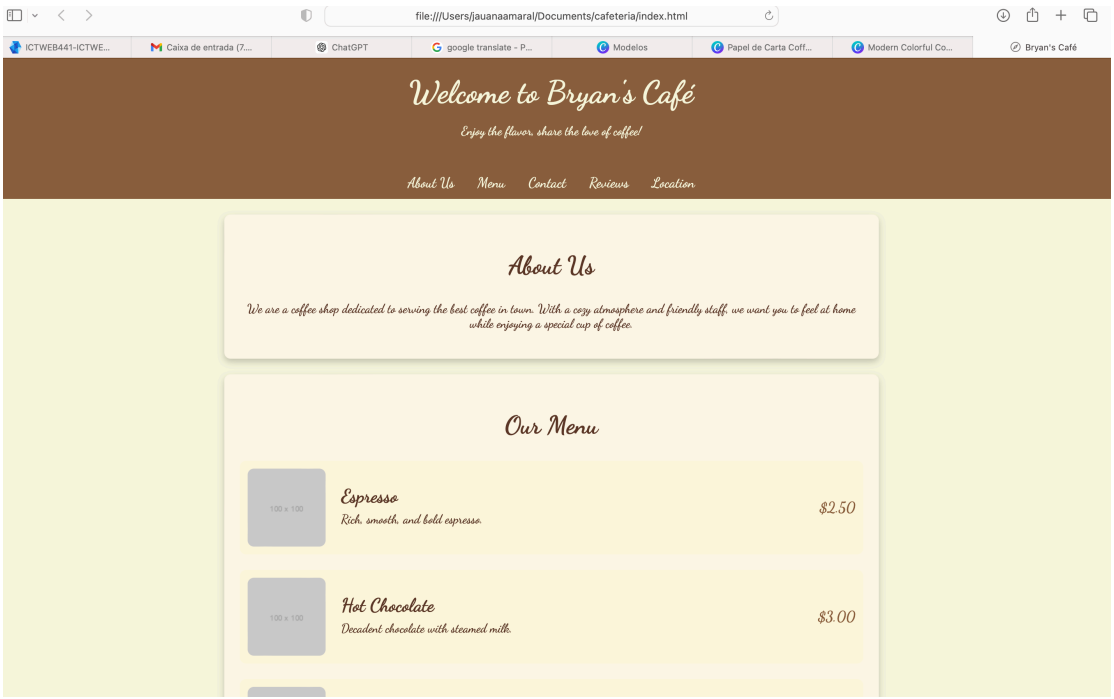
- The raw XML structure was visible in both browsers, and in the case where XSLT was used, the data was styled appropriately in both browsers.

4. Screenshots: (Since I am unable to directly insert screenshots in this environment, please take screenshots of the XML files as displayed in both Google Chrome and Safari and include them below.)

- Screenshot 1: Showing the XML document in Google Chrome.



-
- Screenshot 2: Showing the XML document in Safari.



Conclusion: Both browsers, Google Chrome and Safari, successfully displayed the XML documents without errors, confirming that the documents are well-formed and compatible across these two popular browsers. No issues related to browser-specific XML rendering were encountered during testing.