Overview of initial QA activities for hotel booking client.

**Overview**

Requirement received from a client for independent Quality Analysis of their hotel booking site. No business requirements/use cases were provided so test focus was based off experience with common web site issues that will need to be address should the client wish to enhance the quality of both the web site and user experience.

Area’s in scope for QA are

* Functionality
* Usability
* Security
* Compatibility
* Accessibility

Performance testing is deemed out of scope however recommendations are added to the report.

**Approach**

Perform an initial round of exploratory testing to flush out key functional issues to report back to client as bugs or future tasks to enhance the solution.

Consider usability elements of the website to enhance the end user experience.

Review websites non functional elements i.e. security, compatibility and accessibility.

Design an automated test which exercises a user creating and deleting a booking with the intention it can be added to the client’s continuous integration pipeline.

**Results**

**Functionality**

Although the website functionally works in the context of allowing a user to make a booking there are several areas of functionality that in my opinion need to be enhanced.

Summary of functional issues identified

|  |  |  |
| --- | --- | --- |
| **ID** | **Issue** | **Comment/Recommendation** |
| 1 | User is able to enter non alphabetic characters in First name and Surname fields | Implement field level validation |
| 2 | User is able to enter non numeric characters in price field | Implement field level validation |
| 3 | User is able to enter a check-in date in the past | Validation required on date fields |
| 4 | User is able to enter a check-out date in the past | Validation required on date fields |
| 5 | User is able to enter a check-out date earlier than the check-in date | Validation required on date fields |
| 6 | User is able to enter a check-in and check-out date which are the same | Raise to BA/Product Owner to see if this is a use case that should be supported? |
| 7 | User not completing all mandatory form data and clicking save results in no UI warning prompt | Console 500 error thrown on API call. Error should be handled as a 400 Bad Request and exposed to user in human readable format.  Or only enable “Save” when all mandatory data is input |
| 8 | Simulating connection lost at save - User not informed that connectivity is lost | Console error thrown on API call. Error should be exposed to user in human readable format to inform them booking was not successful. |
| 9 | Column 1 typo - Amend *Firstname* to read *First name* | Cosmetic bug |
| 10 | Inspection of API calls locates a booking ID that is not presented on the UI (ID 4433). Booking can be accessed directly by hitting /booking/{ID} however HTTP status code 418 is returned | Needs investigating as other invalid bookings could find their way into the system. |
| 11 | Jquery console error on page load  *VM222:1 Uncaught SyntaxError: Unexpected token '<'*  *at eval (<anonymous>)*  *at Function.globalEval (jquery-2.2.3.min.js:2)*  *at ua (jquery-2.2.3.min.js:3)*  *at n.fn.init.append (jquery-2.2.3.min.js:3)*  *at Object.success (script.js:19)*  *at i (jquery-2.2.3.min.js:2)*  *at Object.fireWith [as resolveWith] (jquery-2.2.3.min.js:2)*  *at z (jquery-2.2.3.min.js:4)*  *at XMLHttpRequest.<anonymous> (jquery-2.2.3.min.js:4)* | Needs investigating. |
| 12 | Field length validation missing. User can input data that breaks out of the fields | Business rules to be established and implemented to limit the amount of data in each field to confirm with the UI |

**Usability**

To enhance the user experience when making or deleting a booking there are several items that could be improved. In summary a user is left with little guidance as to what the form input requires nor is given feedback when api calls are made (save/delete).

Summary of usability issues identified

|  |  |  |
| --- | --- | --- |
| **ID** | **Issue** | **Comment/Recommendation** |
| 1 | User isn't guided to mandatory/optional field input | To improve user experience the form should include prompt(s) to inform all form fields are currently mandatory |
| 2 | Where a user selects a check-in date beyond the current month the subsequent check-out date picker lands the user on current month. | Make checkout default month dynamic dependant on check-in to improve the user experience  E.g. Select check-in = 01/06/20  Select check-out should default to 02/06/20 |
| 3 | Convert check-in/check-out date format to be dd/mm/yyyyy | Identify user preference to view booking date format |
| 4 | Deposit paid drop down has default option | Risk to user error to be presenting a default of Deposit=True. Good practise would be to enforce the user to select an option. |
| 5 | User clicking save isn't alerted to wait state | User is not guided after clicking save that the process is running. Suggest introducing a load state to inform them the action is progressing. |
| 6 | At scale the website users will have issues locating specific bookings (1) | Suggest implementation of filters on the form columns and/or search functionality to allow users to locate a specific booking |
| 7 | At scale the website users will have issues locating specific bookings (2) | Implement paging to allow users to view configurable number of bookings per page |
| 8 | Booking ID not exposed to user | When making a booking the *bookingID* is not returned to the client. This reference is potentially needed for business processes? |
| 9 | Delete button doesn't trigger warning | Risk to user error of deleting the wrong booking. Suggest "are you sure" validation prompt be replayed to user to confirm deletion. |
| 10 | Website contains no contact/support contact | If user has a query on making or their booking there is no contact information for them to reach out too. |

**Security**

Currently the web site is flawed in terms of standard security protocols and would undoubtably fail a formal penetration test. A site handling user personal data should as a minimum be encrypted using https.

There is also a real concern of having a public API that contains no access control. Suggest implementing user authentication to gain access to the booking page in particular around the ability to delete bookings.

The delete method contains basic authentication however this can be captured and decoded to reveal the user name and password. (i.e. admin:password123)!

Advise the client on the OWASP security standards documentation as good practise going forward.

Summary of security issues identified

|  |  |  |
| --- | --- | --- |
| **ID** | **Issue** | **Comment/Recommendation** |
| 1 | HTTP unsecure page | For a user data driven site HTTPS should be implemented to ensure encryption of data |
| 2 | Site vulnerable to DOM XSS attack | Malicious user can add to css *<iframe src="javascript:alert(`xss`)"></iframe>* for example |
| 3 | No authentication on an API containing personal data | Suggest implementing access control to ensure a user can only view their own bookings (as oppose to all bookings) |
| 4 | Booking ID's are sequential. | Suggest making this a GUID to avoid users being able to iterate booking ID's to view other users bookings (assuming at some point requirement is introduced for a user to only view their own bookings) |
| 5 | Prevent bot attack | As site is currently public facing consider implementing a solution such as Captcha to prevent bot attacks and spam bookings |
| 6 | Cross-Origin Resource Sharing not implemented | Suggest implement CORS to control the origins allowed to use the API’s and also the allowed methods. |

**Compatibility**

Website in its current form is cross compatible between core desktop browsers (Chrome, IE, Edge, Firefox & Safari) but is not responsive (functional on a mobile device).

Should a user attempt to access the site on a mobile the UI would present like this and be rendering unusable.

A screen shot of a smart phone

Description automatically generated

Summary of compatibility issues identified

|  |  |  |
| --- | --- | --- |
| **ID** | **Issue** | **Comment/Recommendation** |
| 1 | Website is not responsive | Website is for desktop view port only. Mobile views don't render css correctly. |
| 2 | Website not rendering at lower view point | For users with smaller monitor sizes the site stops rendering correctly at around 990px x 550px |

**Accessibility**

Analysing the website under a tool such was ‘WAVE’, in its current form would prevent screen reader users from easily navigating around the website however with some simple front end changes this could be improved.

Summary of accessibility issues identified

|  |  |  |
| --- | --- | --- |
| **ID** | **Issue** | **Comment/Recommendation** |
| 1 | Document language missing | Identifying the language of the page allows screen readers to read the content in an appropriate language |
| 2 | Missing page regions | Region landmarks identify significant page areas for screen readers |
| 3 | Skipped heading level | To facilitate keyboard navigation heading levels should be sequential. Currently the site has a <h1> element and then <h3> elements (no <h2>) |
| 4 | Missing form labels | The form labels do not have text labels therefore the purpose of them may not be presented to screen reader users |

**Automated Tests**

Utilised the cypress JavaScript automated test framework to produce automated tests that exercise the core functionality i.e. create a booking and delete a booking. This tool was selected due to its light weight setup and due to it being written in js this could be handed to the current front end developers or QA to build upon.

There are 2 tests spec’s available one that will exercise the UI and one which just calls the API’s directly (written with the intention a FE/BE architecture is implemented in the future). An early observation on the API tests is the ‘DELETE’ method calls return a 201 (creation of resource). Best practise would be to return a 200 or 204.

Please refer to the readme in the repo for further details.

**Performance**

Investigating the API calls, the client makes a request to */booking* which returns a response body with an array of booking IDs. The client then makes individual requests for each booking via */booking/{ID}* to return the booking data. This round trip of API calls will potentially cause performance issues under load. Potentially look to return all booking ID’s and booking data in the response to */booking* instead.

At some point a performance test will need to be provisioned. It was noted during exploratory testing that the GET bookings and form save/delete introduced a lag which under load could manifest itself.

Set up a load test (utilising an appropriate tool such as JMeter) to gauge how the website will handle concurrent users making concurrent booking transactions. Move towards running a stress test to obtain a view of the maximum load the site can take before it reaches its limit (anticipated volumetrics to be provided by the client)

**Next Steps**

The next steps for this process are to feedback the various bugs/issues/concerns to the delivery manager/product owner on the project to both gather further information on the desired user experience both initially and for the roadmap of the project. This will allow triaging of the issues to ensure the correct development priority is assigned to each issue found.

Speak to Tech Lead to understand why a form was used to take bookings as oppose to a microservice solution containing a front and back end architecture to utilise a more secure database. Future architecture enhancements will need QA involvement from the outset to provide guidance on ensuring quality is built into the process from the start as a key learning from the current position is a lot of the issues documented above would’ve been avoided if a QA had been engaged from the start of the project.