Tech Assessment - Automation Test Strategy

#### **Note:** A true Test Strategy needs a good amount of research done beforehand to understand the purpose of the application. The plan for the program's development, maintenance and deployment needs to be understood before being able to plan how best to test it. So this test strategy is taking some assumptions which were formulated while testing the website, and comparing to similar websites that involve booking products.

It is also worth mentioning that a Test Strategy will also define the manual testing that needs to be carried out alongside these automated checks, but this document will be focusing on the automation aspects as per the assessment brief.

#### **Objective:** To ensure the functionality, usability, and reliability of the B&B room booking website through well-structured automated UI tests.

### **Scope:**

1. **Core Features**:
   * Room booking process (selecting dates, entering user details, and confirming booking).
   * Contact form functionality.
2. **Browsers**:
   * Chromium (Edge/Chrome), Firefox, Safari.
3. **Devices**:
   * Desktop and mobile resolutions.
4. **Types of Tests**:
   * Functional tests.
   * Accessibility tests.
   * Performance tests.

### **Test Approach:**

1. **Test Automation Framework**:
   * Use Playwright for automated UI testing for its efficient waiting, out-of-the-box browsers, advanced selectors/locators and debugging tools.
   * Use Typescript for its widespread documentation and resources.
2. **Test Design**:
   * Each test to be tagged for easy categorization (e.g., @critical, @regression, @accessibility).
   * Split code into Page Object Model(s) and tests.
     1. Create Base Page object to prepare for potential additional pages later in development
3. **Prioritization**:
   * Focus on critical path scenarios first (searching for rooms, booking a room, and form submission).
   * Include edge cases and boundary conditions (e.g., invalid dates or incomplete forms).

### **Test Scenarios:**

#### **Critical Path Tests:**

* Verify that all displayed rooms have necessary details (image, price, and description).
* Verify booking a room (select dates, fill form, and submit).
* Verify confirmation message after successful booking.

#### **Form Validation Tests:**

* Verify validation errors for invalid/missing user inputs (e.g., blank name, invalid email).
* Verify field-specific constraints (e.g., phone numbers accepting only digits).

#### **Regression Tests:**

* Ensure features from previous releases remain functional (e.g., forms open and close).

#### **Accessibility Tests:**

* Verify keyboard navigation and tabbable elements.
* Ensure meaningful labels and ARIA roles for assistive technologies.

**Performance Tests:**

* Ensure pages load within acceptable timeframes.
* Ensure flows (submit contact form) can be completed in acceptable timeframes.

**API Tests:**

* Ensure API endpoints react with relevant error codes for valid/invalid payloads.

### **Test Data:**

1. **Valid Data**:
   * Realistic user inputs for booking.
2. **Invalid Data**:
   * Missing, incomplete, or incorrectly formatted inputs.

### **Reporting:**

* Generate detailed test reports with test case names, pass/fail statuses, and error messages.
* Use HTML or JSON reports for ease of sharing.

### **Execution Plan:**

1. **Local Runs**:
   * Give developers tools/config files to allow quick test suites to run, to ensure quality is being maintained during their development work.
2. **Cross-Browser Execution**
   * Execute tests across multiple browsers before deployment.
3. **CI/CD Pipeline**:
   * Integrate automated tests into the CI/CD pipeline for continuous feedback.

### **Risks and Mitigation:**

1. **Flaky Tests**:
   * Use retries for unstable tests.
   * Debug flaky tests promptly.
2. **Dynamic Content**:
   * Use robust selectors (e.g., data-test-id) and wait mechanisms.

### **Success Criteria:**

* All critical path tests pass.
* No major accessibility issues identified.
* Regression tests validate key features.
* Performance tests highlight no significant resource increases.