|  |
| --- |
|  |
| Test plan for Automationintesting.online website |
|  |

**1. Introduction**

This test plan provides a comprehensive approach for testing of the website [**https://automationintesting.online**](https://automationintesting.online). The website serves as a booking platform with various features like room booking, contact form submission, and navigation across different sections. The automation suite is designed to ensure the website's critical features function correctly and meet both functional and non-functional requirements. The goal is to validate the website's critical functionalities, ensuring it meets user expectations and performs reliably. The automation will focus on functional testing (e.g., room booking, contact form) and non-functional testing (e.g., performance, cross-browser compatibility). The test suite will showcase automation skills, modularity, and scalability.

**2. Objectives**

* Validate the core functionality of the website, ensuring the end-user experience is seamless. To ensure that website meets user expectations and performs reliably.
* Test both functional and non-functional aspects, such as responsiveness, usability, and performance.
* Identify and mitigate risks through continuous feedback.
* To developed a reusable and maintainable automation framework. Automate regression tests to reduce manual effort.
* To automate critical functional and non-functional tests for the website.
* To generate detailed test reports for analysis and debugging.

**3. Test Scope**

The test plan will cover the following areas:

1. **Functional Testing:**
   * Validate core features such as room booking, contact form submissions, and UI interactions.
   * Test error handling for invalid inputs (e.g., missing fields, invalid dates, incorrect formats).
   * Verify navigation and accessibility of key pages.
   * Checking the links/hyperlinks
   * Checking reporting
   * Checking websites opens on different devices laptop, desktop, mobiles, tables.
2. **Non-Functional Testing:**
   * Assess performance metrics such as page load times and responsiveness.
   * Test cross-browser compatibility (Chromium, Firefox, WebKit).
   * Validate error handling
   * Checking Scalability, Stress and load testing

**The automation suite will cover:**

* + **Functional Testing:** The automation suit will be build covering main functional test cases and main navigation End-to-end workflows, UI elements, and feature validation.
  + **Non-Functional Testing:** The automation test case will also build to cover basic non-functional testing area like Performance, scalability, responsiveness, and combability.

**4. Features to be Tested**

1. **Room Booking:**
   * View available rooms.
   * Submit room booking form with valid inputs.
   * Room booking process (including date selection, guest details, and confirmation).
   * Booking confirmation email or success message.
   * Error handling for invalid inputs (e.g., missing fields, invalid dates).
2. **Contact Form:**

* Submit form with valid inputs.
* Display and behaviour of error messages for invalid inputs.
* Error handling for invalid inputs (e.g., invalid email, empty fields).

1. **UI/UX:**
   * Verify navigation links (e.g., Home, Admin Panel, Contact).
   * Ensure responsive design across devices (desktop, tablet, mobile).
2. **Admin Panel:**
   * Login functionality with valid credentials.
   * Error handling for invalid login attempts.
   * Room creation and deletion.
   * Booking a room.
   * Updating the information related to home pages.
3. **Navigation:**
   * Correct functioning of menus and links.
   * Page redirections and loading states.
4. **UI Responsiveness:**
   * Validation across different devices (mobile, tablet, desktop).
5. **Error Handling:**
   * Appropriate error messages for invalid actions (e.g., invalid dates, input validation).
6. **Performance:**
   * Page load times.
   * Booking process performance under varying load conditions.
   * Performance testing of the login page.
7. **Stress Testing**

* Performing the home page under different load to achieve how much websites is responding under stress conditions.

1. **Scalability Testing**

* Tests how the system handles a small number of concurrent users.
* Tests how the system handles frequent login/logout cycles.
* Tests how the system handles a large number of concurrent users.

1. **Cross browser Testing.**

* Test on multiple browsers (Chromium, Firefox, WebKit, MS Edge, Chrome)

**5. Features Not to be Tested**

1. **Third-Party Integrations:**
   * Integration with external payment gateways or social media platforms (if any).
2. **Non-Critical UI Elements:**
   * Minor UI elements like tooltips or hover effects that do not impact core functionality.
   * Internal server-side logic or APIs not accessible via the UI.
   * Non-critical UI components, such as optional tooltips or animations.
3. Database testing or databased related interactions.

**6. Test Approach**

The testing approach for the https://automationintesting.online/ website combines manual testing and automation testing to ensure comprehensive coverage of both functional and non-functional aspects.

This hybrid approach leverages the strengths of both methods:

**Manual Testing:** For exploratory testing, usability testing, and ad-hoc scenarios.

**Automation Testing:** For repetitive, regression, and performance-critical scenarios.

The test approach will include the following types of testing:

* **Functional Testing**

Validate the core functionalities of the website.

1. Room Booking
2. Verify users can view available rooms.
3. Validate room booking form submission with valid inputs.
4. Test error handling for invalid inputs (e.g., missing fields, invalid dates).
5. Validate form submission with valid inputs.
6. Test error handling for invalid inputs (e.g., invalid email, empty fields).
7. Verify navigation links (e.g., Home, Admin Panel, Contact).
8. Ensure responsive design across devices (desktop, tablet, mobile).
9. Admin Panel:
10. Validate login functionality with valid credentials.
11. Test error handling for invalid login attempts.
12. Verify room creation and deletion.

* **Non-Functional Testing**

Assess the performance, responsiveness, and compatibility of the website.

* **Performance:**

Measure page load times for critical pages (e.g., homepage, room booking page).

* **Cross-Browser Compatibility:**

Test core functionalities on multiple browsers (Chromium, Firefox, WebKit).

* **Error Handling:**

Verify the website handles errors gracefully (e.g., 404 pages, server errors).

1. **Manual Testing Approach**

To validate the website's functionality, usability, and user experience through human observation and interaction. To identify issues that may not be easily detectable through automation (e.g., visual inconsistencies, usability problems).

* **Functional Testing:**
* Validate core features such as room booking, contact form submissions, and UI interactions.
* Test error handling for invalid inputs (e.g., missing fields, invalid dates, incorrect formats).
* Verify navigation and accessibility of key pages.
* **Exploratory Testing:**
* Perform ad-hoc testing to identify unexpected issues.
* Focus on user workflows and edge cases. Explore the website to identify usability issues, UI inconsistencies, and hidden defects. Use session-based exploratory testing to ensure systematic coverage.
* **Usability Testing:**
* Evaluate the user interface for intuitiveness and ease of use.
* Test help and support features (e.g., built-in help, online resources).
* Manually validate the website’s UI for layout, alignment, and font consistency across devices and browsers.
* Check for responsiveness across mobile, tablet, and desktop.
* Validate ease of navigation and clarity of instructions for end users.
* **Accessibility Testing:**
* Verify compliance with accessibility standards.
* Test screen reader support, keyboard navigation, and colour contrast.

**2. Automation Testing Approach**

To automate repetitive and regression test cases to improve efficiency and coverage.

To validate performance, cross-browser compatibility, and error handling.

* **Functional Testing:**
* Automate core features such as room booking, contact form submissions, and UI interactions.
* Test error handling for invalid inputs (e.g., missing fields, invalid dates).
* **Regression Testing:**
* Automate regression test cases to ensure existing functionalities are not broken after changes.
* **Performance Testing:**
* Measure page load times for critical pages (e.g., homepage, room booking page).
* **Cross-Browser Testing:**
* Test core functionalities on multiple browsers (Chromium, Firefox, WebKit).

**Automation Tool Selection**

Need to finalized which automation tool will be used for automation testing.

Playwright will be used for end-to-end automation. Since it supports Cross-browser support (Chromium, Firefox, WebKit) and Reliable and fast execution.

Built-in support for modern web features (e.g., single-page applications, network interception). Easy integration with CI/CD pipelines.

**Framework Design**

The Page Object Model design pattern will be used to create a modular and maintainable framework.

Each page (e.g., Home, Room Booking, Contact) will have its own class containing elements and actions.

**Reusability of code.**

Easy maintenance (changes in UI elements require updates in one place).

Improved readability and scalability.

**Modularity:**

Test scripts will be divided into smaller, reusable modules (e.g., login module, booking module). Common actions (e.g., form submissions, navigation) will be abstracted into utility functions.

**3. Combined Manual and Automation Strategy**

**Manual Testing for New Features:**

Use manual exploratory and functional testing for new features and scenarios requiring human judgment. Eg.: Usability testing of error messages and visual appearance.

**Automation for Repetitive Tasks:**

Automate regression tests and repetitive workflows (e.g., booking flow validation).

**Parallel Execution:**

Execute manual and automated tests in parallel to maximize coverage.

**4. Test Data Management**

Test Data: Use a combination of static and dynamic test data.

Static data: Predefined inputs (e.g., valid email, invalid email).

Dynamic data: Generate data at runtime (e.g., random names, dates).

Data-Driven Testing:

Use external data sources (e.g., CSV, Excel) to drive test cases for scenarios like room booking and contact form submissions.

**5. Test Execution**

* **Smoke Tests**

Quick validation of critical functionalities like homepage loading and room booking form. Ensures the build is stable enough for further testing.

* **Sanity Tests**

Deep testing of specific functionalities after minor changes or bug fixes.

Focuses on ensuring no new issues are introduced in updated areas.

* **Functional Tests**

Validates all core functionalities such as room booking, contact form, and admin panel. Ensures features work as expected and handle invalid inputs correctly.

* **Integration Tests**

Verifies interactions between different modules like room booking and admin panel.

Ensures seamless integration and data flow across the application.

* **API Testing**

Validates API endpoints for room availability and booking confirmation.

Ensures APIs return correct responses and handle errors gracefully.

* **Regression Tests**

Re-tests existing functionalities after changes to ensure no new issues are introduced.

Confirms that previously working features remain unaffected.

* **Performance Tests**

Measures page load times and responsiveness under various conditions.

Ensures the website performs well under expected user loads.

* **Cross-Browser Tests**

Validates core functionalities across multiple browsers like Chrome and Firefox.

Ensures consistent behaviour and compatibility across browsers.

* **Accessibility Tests**

Verifies compliance with accessibility standards.

Ensures the website is usable by all users, including those with disabilities.

* **Usability Tests**

Evaluates the user interface for intuitiveness and ease of use.

Ensures the website provides a positive user experience.

**6. CI/CD Integration**

The automation framework will be integrated with Jenkins for continuous testing.

Tests will be triggered automatically after each build or deployment.

**7. Test Environment**

1. **Browsers:** Chromium (default), Chrome, Firefox, WebKit.
2. **Operating Systems:** Windows, Linux, macOS.
3. **Tools:**
   * Playwright: For browser automation.
   * Java: Programming language for test scripts.
   * Maven: For dependency management and build automation.
   * IntelliJ IDEA/Eclipse: IDE for development.
   * GitHub: For version control.
4. **Test Data:** Dummy user details, booking dates, and room types.
5. **Devices:** Mobile (iPhone, Android), Tablet, Desktop.
6. The testing setup will closely resemble the production environment to accurately identify any environment-specific issues.

**8. Test Deliverables**

1. **Test Plan Document:** A comprehensive document outlining the test plan, detailing objectives, scope, approach, and methodologies.
2. **Test Cases:** Detailed test cases for each identified test scenario, specifying steps, expected results, and criteria for pass/fail.
3. **Test Scripts:** Automated test scripts using Playwright.
4. **Traceability matrix:** Requirement traceability matrix (RTM) will be delivered to map requirements with corresponding test cases, ensuring comprehensive coverage and validation.
5. **Test Execution Reports:** Reports documenting the execution of test cases, including actual results and any deviations from expected outcomes.
6. **Defect Reports:** Detailed reports documenting any defects found during testing, including severity, steps to reproduce, and recommendations for resolution.
7. **Final Test Summary/Signoff Report:** A consolidated summary of all testing activities, including overall findings, test coverage, and conclusions.
8. **Logs:** Detailed logs capturing all testing activities, including test case execution logs, defect logs, and any other relevant logs.
9. **Comprehensive Test Report:** A detailed report summarizing the entire testing process, including objectives achieved, challenges faced, lessons learned, and recommendations for improvement.
10. **Weekly or Biweekly Reporting:** Regular updates provided to management and stakeholders, highlighting progress, issues, and any adjustments made during the testing phase.

**9. Entry and Exit Criteria**

1. **Entry Criteria:**
   * Test environment is set up and ready.
   * Test data is prepared.
   * The plan is prepared and approved.
   * Automation Test Script are ready.
   * Test cases are reviewed and approved.
2. **Exit Criteria:**

* 100% execution of test cases for critical functionalities.
* 95% pass rate for all test cases, with no high-severity defects. All critical/major defects are resolved. If any outstanding defects remaining then approved from Stockholders.
* Test reports are generated and reviewed and sign off.

**10. Pass/Fail Criteria**

* Pass: All test cases execute successfully, and the actual results match the expected results.
* Fail: Any test case fails due to functional or non-functional issues (e.g., broken links, incorrect error messages or any performance related issues ).

**11. Resumption Criteria**

Testing will resume after:

* Fixing critical defects.
* Receiving approval from stakeholders.
* High-severity defects are fixed and retested.
* Test environments are stable and accessible.

**12. Risks and Assumptions**

1. **Risks:**
   * Limited access to the test environment.
   * Changes in requirements during the testing phase.
   * Browser-specific issues affecting cross-browser compatibility.
   * Environment instability leading to false positives/negatives.
   * Potential tool limitations for testing certain aspects (e.g., dynamic elements).
2. **Assumptions:**
   * The test environment will be stable and available throughout the testing phase.
   * All required tools and dependencies will be installed and configured.
   * Test data will be available and representative of real-world usage.
   * Automation tools (e.g., Playwright) are compatible with the website's technology stack.

**13. Responsibilities**

* Test Lead: Oversee the testing process and ensure timely delivery. Review test results, triage issues, and ensure test plan adherence.
* Test Engineers: Develop and execute test cases, report defects.
* Developers: Fix defects and provide support for test environment setup.
* Stakeholders: Review and approve test plans and reports.
* Project Manager: Allocate resources and approve the final deliverables.

**14. References**

* Website: <https://automationintesting.online/>
* Playwright Documentation: <https://playwright.dev/>

**15. Approval**

Reviewers:

* QA Lead
* Project Manager

Approval Status: Pending / Approved