**Test Plan – Companies House Technical Assessment**

# 1. Introduction

1.1 Purpose  
This test plan outlines the testing strategy for the hotel booking website (<https://automationintesting.online/>) as part of the Companies House Lead Test Engineer technical assessment. It covers functional and non-functional testing using Cucumber BDD with Playwright and TypeScript, targeting key user-facing features.

## 1.2 Scope

* BDD Testing using Cucumber and Gherkin syntax
* Functional Testing: booking, contact form, navigation, admin access
* Non-Functional Testing: accessibility (WCAG 2.1), performance, security
* API Testing: backend for booking, contact, authentication
* Integration Testing: user journey validation
* Error Handling: edge cases and validation scenarios
* Security Testing: admin access restriction

## 1.3 Objectives

* Demonstrate test automation using BDD principles
* Validate critical functionality across the system
* Ensure accessibility compliance with WCAG 2.1
* Report actionable defects and recommendations
* Deliver maintainable and scalable automation framework
* Validate backend API integrations
* Confirm secure access controls on admin functionality

# 2. Test Strategy

## 2.1 Testing Approach

*Behaviour-Driven Development (BDD)*

* Cucumber with Gherkin feature files
* Stakeholder-readable Given-When-Then structure
* Tag-based test execution using @smoke, @regression, etc.

*Risk-Based Testing*

* High priority: Booking, Contact Form, API, Accessibility
* Medium priority: Admin access, Navigation, Performance

*User Journey Testing*

* Booking flow: search → book → confirm
* Contact form: open → submit message
* Security: direct admin access blocked
* API: backend validation and error handling

*Quality Gates*

* Functional: all critical journeys must pass
* Accessibility: WCAG 2.1 AA compliance
* Performance: acceptable load time and response
* Security: admin functionality properly protected
* API: endpoints handle success and failure scenarios

## 2.2 Test Environment

* Application: <https://automationintesting.online/>
* Technology: web application with RESTful APIs
* Test stack: Cucumber BDD, Playwright, TypeScript
* Browsers: Chromium, Firefox, WebKit
* Devices: desktop and mobile
* Tools: axe-playwright (accessibility), playwright-lighthouse (performance) Playwright API testing
* Reports: Cucumber HTML and JSON outputs

# 3. Test Categories

## 3.1 BDD Features and Scenarios

### Booking Feature

* **Scenarios**: 6 total

### Accessibility Feature

* Scenarios: 15+ covering WCAG checks, labels, ARIA roles, keyboard navigation

### Admin Access Feature

* **Scenarios**: 12 validating login, access control, error handling, SQL injection

### Contact Form Feature

* **Scenarios**: 8 for validation, submission, reset, success messages

### Navigation Feature

* **Scenarios**: 6 for menu, links, responsive layout

### Performance Feature

* **Scenarios**: 8 including load time, resource usage, mobile performance

### API Testing Feature

* **Scenarios**: 11 covering booking, contact, authentication endpoints

## 3.2 Step Definitions

* **Booking**: Calendar selection, URL verification, date validation
* **Accessibility**: WCAG scans, form labels, keyboard usage, ARIA roles
* **Admin Access**: Login checks, security validation, error handling
* **API**: Booking/contact/auth endpoint testing with proper assertions
* **Common**: Shared setup, navigation, teardown utilities
* **Performance**: Load time monitoring, interaction responsiveness

## 3.3 Support Files

* **world.ts**: Custom World with shared state, page objects, performance monitoring
* **hooks.ts**: Setup/teardown, screenshot on failure, test data cleanup, error handling

# 4. Test Execution Strategy

## 4.1 Configuration

* cucumber.js: TypeScript setup, parallel execution, HTML/JSON reporting
* **Execution Environment**: Windows 64-bit, Node.js 20.18.0, Cucumber-js 10.9.0

## 4.2 Execution Phases

**Phase 1**: Booking, Contact -- core flows (83% and 100% pass rates)

**Phase 2**: API, Admin Access -- integration, security (91% and 92% pass rates)

**Phase 3**: Navigation, Accessibility -- usability and standards (83% and 80% pass rates)

**Phase 4**: Performance -- page load, efficiency (75% pass rate)

## 4.3 Exclusions

* Admin room management tests: credentials not available for full admin testing
* Room setup: testing limited to existing data scenarios

# 5. Test Results Summary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Feature | Scenarios | Passed | Failed | Pass Rate | Critical Issues |
| Booking | 6 |  |  |  |  |
| Contact Form | 8 |  |  |  |  |
| Admin Access | 12 |  |  |  |  |
| Accessibility | 15 |  |  |  |  |
| Navigation | 6 |  |  |  |  |
| Performance | 8 |  |  |  |  |
| API Testing | 11 |  |  |  |  |
| Total | **76** |  |  |  |  |

## 5.1 Performance Metrics

* **Expected Execution Time**: Not more than 4 minutes for 76 scenarios
* **Platform Stability**: Consistent results across runs
* **Resource Usage**: Acceptable for test automation
* **Failure Patterns**: Timeouts indicate performance bottlenecks

# 6. BDD Benefits Demonstrated

## 6.1 Stakeholder Communication

* Clear, readable scenarios in natural language
* Business-focused test descriptions
* Tag-based organization for targeted testing
* Comprehensive reporting with screenshots

## 6.2 Maintainability

* Modular step definitions allowing reuse
* Page Object Model implementation
* Separation of concerns between features
* Robust error handling and cleanup

## 6.3 Quality Assurance

* Automated accessibility testing with axe-playwright
* Cross-browser compatibility validation
* API integration testing
* Performance monitoring integration with playwright-lighthouse

# 7. Test Data Strategy

## 7.1 Data Management

* **Dynamic Generation**: Unique customer/contact/admin inputs per test run
* **Edge Case Testing**: Invalid data to trigger validation scenarios
* **Cleanup Process**: Test data removed post-execution when required
* **Realistic Data**: Fake but valid email formats and user information

## 7.2 Data Validation

* Form input validation testing
* API payload validation
* Date range and boundary testing
* Cross-field validation scenarios

# 8. Success Criteria

## 8.1 Functional Requirements

* **Booking Flow**: 83% pass rate (acceptable with known issue)
* **Contact Form**: 100% pass rate (excellent)
* **Admin Access**: 92% pass rate (good with minor security issue)
* **Navigation**: 83% pass rate (acceptable with known scroll issue)

## 8.2 Non-Functional Requirements

* **Accessibility**: 80% pass rate (requires improvement for WCAG compliance)
* **Performance**: 75% pass rate (needs optimization)
* **API Integration**: 91% pass rate (good with server-side fixes needed)

## 8.3 Quality Metrics

* **Overall Pass Rate**: 74% (above minimum threshold but improvement needed)
* **Critical Issues**: 5 high-severity bugs requiring immediate attention
* **Medium Issues**: 10 bugs requiring scheduled remediation
* **Low Issues**: 8 bugs for future optimization

# 9. Risk Assessment & Mitigation

## 9.1 High Risk Areas (Immediate Action Required)

* **Booking API Failures**: Server errors blocking core functionality
* **Date Validation**: Past date booking allowed (business logic failure)
* **API Security**: Inadequate validation and error handling
* **Performance Timeouts**: Multiple scenarios failing due to performance

### Mitigation Strategy:

* Immediate backend debugging for API issues
* Frontend date validation implementation
* Performance profiling and optimization
* Security audit of API endpoints

**9**.2 Medium Risk Areas (Scheduled Remediation)

* **Accessibility Compliance**: WCAG violations affecting user experience
* **Mobile Performance**: Timeout issues on mobile devices
* **Navigation Functionality**: Amenities scroll behavior
* **Admin Security**: Logout button visibility logic

### Mitigation Strategy:

* Systematic accessibility review and fixes
* Mobile-specific performance optimization
* Navigation component debugging
* Authentication state management review

# 10. Reporting and Documentation

## 10.1 Test Artifacts

* **HTML Reports**: Detailed scenario results with screenshots
* **JSON Reports**: Machine-readable results for CI/CD integration
* **Bug Reports**: Documented issues with reproduction steps
* **Performance Reports**: Load time and interaction metrics

## 10.2 Evidence Collection

* **Screenshots**: Automatically captured on test failures
* **Accessibility Reports**: Detailed axe-playwright violation summaries
* **API Response Logs**: Request/response data for debugging
* **Performance Metrics**: Load times and interaction measurements

# 11. Technical Implementation

* Tools: Cucumber, Playwright, TypeScript, axe-playwright
* Page Object Model applied
* Well-structured, maintainable code
* Clean error handling and helper abstractions

# 12. Business Impact Analysis

## 12.1 Revenue Protection

* **Critical Booking Issues**: If there are critical booking and contact us issues, this will affect core business functionality
* **API Reliability**: Server errors impacting booking creation and customer communication
* **User Experience**: Accessibility and performance issues affecting customer satisfaction

## 12.2 Compliance Requirements

* **WCAG 2.1 AA**: Multiple violations requiring systematic remediation
* **Security Standards**: Admin access controls need strengthening
* **Performance Standards**: Mobile optimization required for accessibility compliance

## 12.3 Risk Mitigation Value

* **Early Detection**: Bugs identified before production deployment
* **Systematic Coverage**: Comprehensive testing across all major functional areas
* **Quality Metrics**: Clear pass/fail criteria with actionable remediation steps