**TEST PLAN**

**1. Test Plan Overview**

* **Project Name**: Royal Cars - Car Rental Website
* **Test Plan ID**: TP001
* **Version**: 0.1
* **Objective**: To ensure the functionality, security, performance, and usability of the Royal Cars - Car Rental Website.

**2. Test Items (Modules)**

* Home
* Signup
* Login
* Cars
* Add a car
* My cars
* Contact Us
* About Us
* Compatibility

**3. Features to be Tested**

**3.1 HOME**

* Header
* Navigation Menus
* Image Slider
* Website Introduction
* Images
* Links
* Footer

**3.2 Sign-Up**

* Link visibility
* Signup form layout
* Input Field Validation
* Sign-Up Confirmation
* Duplicate Account Handling
* Browser Autofill Handling
* Session Timeout Handling

**3.3 Login**

* Link visibility
* Login form layout
* Successful Login
* Empty Fields
* Invalid Credentials
* Case Sensitivity
* Button Visibility & Functionality
* Session Expiry
* Multi-Tab Session
* Multi-Device Login
* Brute Force Protection
* Log in via Browser Autofill
* Logout Redirection, Persistence, and Back Button Behaviour

**3.4 Cars**

* Navigation menus
* Social media links
* Page UI
* Cards UI
* Cars image
* Car information
* Chat with seller button

**3.5 Add Car**

* Page UI
* Add a car form layout
* Valid credential
* Invalid credential
* Links
* Button

**3.6 My Cars**

* Header
* Navigation Menus
* Page UI
* Car Image
* Car information
* Edit Button
* Delete Button

**3.7 Contact Us**

* Page UI
* Contact details
* Keyboard Navigation
* Footer

**3.8 About Us**

* Page UI
* Navigation Menus
* Keyboard Navigation
* Images
* Website Introduction
* Links
* Footer

**3.9 Compatibility**

* Browser Compatibility
* Device Compatibility
* Screen Resolution
* Network Compatibility

**4. Types of testing**

* Functional Testing
* Usability Testing
* Performance Testing
* Security Testing
* Compatibility Testing
* Integration Testing

**5. Test Strategy**

**Approach:**  Manual testing and Automation testing

**6. Test Environment Setup**

* Browsers: Chrome, Edge
* Devices: Desktop, Mobile (responsive testing)
* Operating Systems: Windows
* Network: WiFi connectivity
* Tools: Selenium WebDriver, TestNG, Eclipse IDE

#### **7. Test Deliverables**

* **Test Plan Document**: This document.
* **Test Case Document**: A detailed list of test cases, including expected results.
* **Defects Log**: A record of any defects found during testing.
* **Test Summary Report**: A report summarising the testing process, including any issues found and the final status.

**8. Risks and mitigations**

| **Risk** | **Mitigation Strategy** |
| --- | --- |
| **Unstable deployment or hosting downtime** | Use uptime monitoring tools (e.g., UptimeRobot). Maintain backups and test on a staging server. |
| **Incomplete test coverage** | Prepare a detailed test matrix with mapped test cases for all modules and submodules. |
| **Missing or outdated browser compatibility** | Perform cross-browser testing on all major browsers (Chrome, Firefox, Safari, Edge) and devices using tools like BrowserStack. |
| **UI breaks on smaller or larger screens** | Ensure responsive testing is part of UI test cases. Use developer tools to simulate different screen sizes. |
| **Security vulnerabilities (e.g., XSS, SQL injection)** | Include security testing in scope. Perform input validation and use tools like OWASP ZAP or Burp Suite. |
| **Critical functionality failing post-deployment** | Include smoke/sanity test suites after each deployment. Use CI/CD to automate basic checks. |
| **Inconsistent data between the frontend and the backend** | Perform integration testing to validate API responses and frontend rendering. |
| **Session timeout or user data leak** | Test session expiry and cookie security. Enable HttpOnly, Secure, and SameSite flags. |
| **Poor page load performance** | Conduct performance testing. Optimise images, defer JS loading, and use caching strategies. |
| **Broken links or missing resources** | Include automated checks for broken links (e.g., using Screaming Frog SEO or custom scripts). |
| **Regression issues in repeated builds** | Maintain the regression test suite and run it during each release cycle. |
| **Delays in bug fixing or unclear communication** | Use JIRA or similar tools for bug tracking. Conduct regular stand-ups for progress alignment. |

**9. Test Case Templates**

* **Test Case ID**: TC\_LOG\_001
* **Test Case Description**: Verify login functionality with valid credentials
* **Preconditions**: The User must be registered.
* **Test Steps**:
  1. Go to the login page.
  2. Enter a valid username and password.
  3. Click on "Login."
* **Expected Result**: User is logged in and redirected to the homepage.
* **Actual Result**: [To be filled during execution]
* **Status**: Pass/Fail

**Defects**: [Link to defects, if any]

Links

1. **Test Case Reports, RTM, Test Execution Report, Bug Report**

<https://docs.google.com/spreadsheets/d/1VgdpuiWWZUE9z3CBs8Pr4mOrfPaO790KrlvI1wn2dl8/edit?usp=sharing>

1. **screenshot of the automation script document & screen recording file**

<https://docs.google.com/document/d/1ZUXgsvToI9fCdnZFt9Zi-61pN-i0nphzhlF05kTDWQ0/edit?usp=sharing>