**Test Strategy Document**

### **Test Strategy-ID: TS-001**

### **Introduction**

This document defines the test strategy for validating the **Advanced Search functionality** of the eCommerce platform **nopCommerce** ([demo.nopcommerce.com](https://demo.nopcommerce.com/)). The feature is live in production, and testing will be conducted in a real-time environment to ensure correctness, usability, performance, and security.

### **In Scope**

* Functional Testing
* UI/UX Testing
* Performance Testing
* Compatibility Testing
* Security Testing
* Exploratory Testing
* Regression Testing (Manual & Automated)

### **Out of Scope**

* Database migration testing
* Third-party API testing (except those directly involved in search functionality)
* Backend system performance benchmarking
* Load testing beyond anticipated user traffic

### 

### 

### **Standards to be Used**

* **Test Management**: Follow ISTQB standards for test design and execution.
* **Automation Framework**: Selenium WebDriver with TestNG.
* **Security Standards**: OWASP guidelines for security testing.
* **Reporting & Logging**: Extent Reports & Allure Reports for execution tracking.

### **Test Strategy for Automation**

* **Tools & Frameworks**:  
  + Selenium WebDriver for UI automation
  + TestNG for test execution and reporting
  + RestAssured for API testing (if applicable)
  + Jenkins/GitHub Actions for CI/CD
  + Extent Reports & Allure Reports for reporting
* **Automation Scope**:  
  + Regression test suite covering core functionalities
  + Cross-browser compatibility tests
  + Data-driven tests using external data sources (Excel, JSON, DB)
  + Parallel execution using Selenium Grid
* **Non-Automated Areas**:  
  + UI/UX validation (except for layout consistency checks)
  + Exploratory and ad-hoc testing
  + Tests requiring manual intervention (e.g., Captcha handling)

### **Risks and Mitigations**

| **Risk** | **Mitigation** |
| --- | --- |
| Unavailability of live environment | Use a pre-production staging environment as a fallback |
| Inconsistent search results | Conduct multiple test iterations under varied conditions |
| Browser compatibility issues | Execute cross-browser testing on Chrome, Firefox, Edge, and Safari |
| Performance degradation under high load | Perform stress and load testing to assess system resilience |

### **Entrance Requirements**

* Business requirements for Advanced Search are finalized and approved.
* Test environments are set up and accessible.
* Test data is prepared for different search scenarios.
* Automation scripts are reviewed and approved for execution.

### **Exit Criteria**

* All high and critical defects are fixed and verified.
* Functional, UI/UX, security, and performance tests are completed successfully.
* Regression tests confirm stability post-changes.
* Test summary report is approved by stakeholders.

**Testing Rounds**

| **Testing Phase** | **Description** |
| --- | --- |
| **Unit Testing** | Developers verify individual functions & locators. |
| **Functional Testing** | Ensuring the UI behaves as expected. |
| **Integration Testing** | Validating interactions between frontend & backend. |
| **Regression Testing** | Checking if new changes don’t break existing features. |
| **End-to-End Testing** | Ensuring the entire user flow works correctly. |

## **Tools Used for Testing**

| **Category** | **Tool Used** |
| --- | --- |
| **Automation Tool** | Selenium with Java |
| **BDD Framework** | Cucumber |
| **Test Runner** | JUnit |
| **CI/CD Integration** | Jenkins |
| **Browser Testing** | ChromeDriver |
| **Reporting** | Extent Reports |
| **Cloud Execution** | Selenium Grid (Optional) |

## **Bug Tracking Tool**

| **Tool** | **Usage** |
| --- | --- |
| **JIRA** | Logging, tracking, and managing bugs. |
| **TestRail** | Managing test cases and reporting bugs. |
| **Bugzilla** | Alternative bug-tracking tool (if needed). |

**Test Design Techniques**

* **Equivalence Partitioning**: Validate search with valid, invalid, and boundary inputs.
* **Boundary Value Analysis**: Test minimum and maximum character limits in search fields.
* **Error Guessing**: Identify and test potential failure scenarios based on experience.
* **Pairwise Testing**: Optimize test coverage by considering key parameter combinations.

## **Traceability Matrix**

The **Traceability Matrix** maps test cases to the corresponding requirements to ensure complete test coverage.

| **Requirement ID** | **Test Case ID** | **Test Scenario** |
| --- | --- | --- |
| SR-001 | TC-001 | Search with valid input |
| SR-002 | TC-002 | Search with special characters |
| SR-003 | TC-003 | Empty search validation |
| SR-004 | TC-004 | Search with numerical input |
| SR-005 | TC-005 | Validate sorting and filtering in search results |
| SR-006 | TC-006 | Login with Registered details |
| SR-007 | TC-007 | Register new user with mandatory details |
| SR-008 | TC-008 | Verify search results pagination |

## **6. Test Architecture Document (TAD)**

The **Test Architecture Document (TAD)** outlines the automation framework, tools, and reporting mechanisms used in testing.

### **Automation Framework**

* **Framework**: Selenium WebDriver with TestNG
* **Programming Language**: Java/Python
* **Test Execution**: Parallel execution using TestNG
* **Page Object Model (POM)**: Used for maintainability and scalability
* **Data-Driven Testing**: Implemented using Excel/CSV files for parameterized tests
* **Exception Handling**: Implemented in Selenium scripts for robustness

### 

### **CI/CD Integration**

* **Version Control**: GitHub/GitLab
* **CI/CD Tool**: Jenkins/GitHub Actions for scheduled test executions
* **Execution Strategy**:  
  + Automated test execution on every pull request merge
  + Nightly regression suite execution
  + Triggered builds on feature deployment

### **Reporting & Logging**

* **Reporting Tools**:  
  + Extent Reports for detailed test execution reports with screenshots
  + Allure Reports for real-time execution status visualization
* **Logging Mechanism**:  
  + Log4j for capturing logs
  + Console logs and file logs for debugging

### **Test Environment**

* **Operating Systems**: Windows 10/11, macOS (latest version)
* **Browsers**: Chrome, Firefox, Edge, Safari (latest versions)
* **Devices**: Desktop (Windows/macOS), Mobile (Android/iOS)
* **Network Conditions**: High-speed internet & simulated low-bandwidth conditions

### 

### 

### **Test Process Improvement**

* Implement continuous feedback loops to refine test cases and automation scripts.
* Use AI-driven test automation tools to optimize test execution.
* Periodic review and updates to regression test suites based on defect trends.

### **Approvals**

| **Role** | **Name** | **Approval Status** |
| --- | --- | --- |
| Test Lead | [Name] | Pending/Approved |
| QA Manager | [Name] | Pending/Approved |
| Product Owner | [Name] | Pending/Approved |