Style Haven E-commerce Platform - Test Plan

1. Introduction

1.1 Purpose

This test plan outlines the testing approach for the Style Haven e-commerce platform. It defines the testing objectives, scope, strategy, resources, schedule, and deliverables.

1.2 Project Overview

Style Haven is an online fashion marketplace connecting independent designers with customers seeking unique clothing. The platform provides personalized recommendations, secure transactions, and customer support.

1.3 Scope

This test plan covers functional, non-functional, compatibility, and performance testing for all major components of the Style Haven platform.

2. Testing Strategy

2.1 Testing Levels

- Unit Testing
- Integration Testing
- System Testing
- User Acceptance Testing (UAT)

2.2 Testing Types

- Functional Testing
- Usability Testing
- Performance Testing
- Security Testing
- Compatibility Testing
- Regression Testing

3. Test Environment

3.1 Hardware Requirements

- 3 Testing Personnel with dedicated testing machines
- Mobile Devices:

    - iOS: Latest iPhone models (iPhone 13, 14, 15)
    - Android: Samsung Galaxy S21, S22, S23

3.2 Software Requirements

- Operating Systems:
    - Windows 11
    - macOS Monterey, Ventura
    - iOS 16, 17
    - Android 12, 13
- Browsers:
    - Google Chrome (latest)
    - Mozilla Firefox (latest)
    - Safari (latest)
    - Microsoft Edge (latest)
    - Internet Explorer 11

3.3 Network Requirements

- Various connection speeds (WiFi, 4G, 5G)
- Network simulation tools for testing under different conditions

4. Testing Schedule

4.1 Timeline

- Total Duration: 45 days
- Phase 1 (Days 1-10): Test Planning and Environment Setup
- Phase 2 (Days 11-25): Test Execution (Functional and Non-Functional)
- Phase 3 (Days 26-35): Compatibility and Performance Testing
- Phase 4 (Days 36-40): Regression Testing
- Phase 5 (Days 41-45): Final Reporting and Documentation

4.2 Milestones

- Test Plan Approval - Day 5
- Test Environment Setup Complete - Day 10
- Functional Testing Complete - Day 25
- Compatibility Testing Complete - Day 35
- Final Test Report Delivery - Day 45

5. Features to be Tested

5.1 User Accounts

- Registration and Login
- Profile Management
- Order History

5.2 Product Catalog

- Browse and Search
- Product Pages
- Recommendations

5.3 Shopping Cart

- Add/Remove Items
- Quantity Adjustments
- Order Summary

5.4 Checkout

- Shipping Options
- Payment Gateway Integration
- Order Confirmation

5.5 Seller Dashboard

- Product Listing
- Inventory Management
- Order Fulfillment

5.6 Customer Support

- Live Chat
- FAQ Section
- Contact Form

5.7 Additional Features

- Wishlists
- Promotions and Discounts
- Loyalty Program

- Social Media Integration

6. Features Not to be Tested

- Third-party API internals
- Backend database performance (covered by separate DB testing)
- Server hardware stress testing

7. Testing Approach

7.1 Functional Testing

- Test each feature against functional requirements
- Validate all user workflows
- Ensure proper error handling and validation

7.2 Usability Testing

- Evaluate user interface consistency
- Verify responsive design on different screen sizes
- Assess overall user experience

7.3 Performance Testing

- Load testing (simulate multiple concurrent users)
- Response time measurement
- Resource utilization monitoring

7.4 Security Testing

- Authentication and authorization
- Data encryption
- Input validation and sanitization
- Protection against common web vulnerabilities

7.5 Compatibility Testing

- Cross-browser testing
- Cross-platform testing
- Mobile responsiveness testing

7.6 Regression Testing

- Verify that new changes don't break existing functionality

- Run automated test suites after each significant update

8. Test Scenarios and Test Cases

8.1 Test Scenario Template

- **Scenario ID**: [Unique Identifier]
- **Scenario Description**: [Brief Description]
- **Feature**: [Related Feature]
- **Pre-conditions**: [Required Setup]
- **Test Steps**: [Step-by-Step Instructions]
- **Expected Results**: [Expected Outcome]
- **Actual Results**: [Observed Outcome]
- **Status**: [Pass/Fail/Blocked/Not Tested]
- **Remarks**: [Additional Information]

8.2 Sample Test Scenarios

1. User Registration and Login
2. Product Search and Filtering
3. Add to Cart and Checkout Process
4. Payment Processing
5. Order Tracking
6. Profile Management
7. Seller Product Listing
8. Inventory Updates

9. Defect Management

9.1 Defect Reporting Template

- **Defect ID**: [Unique Identifier]
- **Title**: [Brief Description]
- **Description**: [Detailed Description]
- **Steps to Reproduce**: [Step-by-Step Instructions]
- **Expected Result**: [Expected Outcome]
- **Actual Result**: [Observed Outcome]
- **Environment**: [OS, Browser, Device]
- **Severity**: [Critical/High/Medium/Low]

- **Priority**: [High/Medium/Low]
- **Status**: [New/Assigned/Fixed/Verified/Closed]
- **Assigned To**: [Responsible Person]
- **Reported By**: [Tester Name]
- **Reported Date**: [Date]
- **Screenshots/Attachments**: [If Any]

9.2 Defect Severity Classification

- **Critical**: System crash, data loss, security breach
- **High**: Major feature failure, no workaround available
- **Medium**: Feature failure with workaround, UI issues affecting functionality
- **Low**: Minor UI issues, spelling mistakes, non-critical bugs

10. Test Deliverables

10.1 Test Documents

- Test Plan
- Test Cases/Scenarios
- Test Data
- Test Scripts (Automated)
- Test Execution Reports
- Defect Reports
- Final Test Summary Report

10.2 Traceability Matrix

- Mapping requirements to test cases
- Tracking test coverage

11. Testing Tools

11.1 Test Management Tools

- JIRA, TestRail, or similar for test case management
- Excel spreadsheets for simple tracking

11.2 Automation Tools

- Selenium WebDriver with Python for web UI testing
- Appium for mobile testing

- PyTest for test frameworks
- Requests library for API testing

11.3 Performance Testing Tools

- JMeter or Locust for load testing
- Chrome DevTools for frontend performance

11.4 Compatibility Testing Tools

- BrowserStack or Sauce Labs for cross-browser testing
- Device farms for mobile testing

12. Risk Management

12.1 Potential Risks

- Limited testing time (45 days)
- Limited resources (3 testers)
- Multiple platforms and browsers to test
- Integration with third-party payment gateways

12.2 Risk Mitigation Strategies

- Prioritize testing based on critical user journeys
- Use automation for repetitive test cases
- Leverage cloud testing platforms for device coverage
- Focus on high-risk areas first

13. Exit Criteria and Test Completion

13.1 Exit Criteria

- All planned test cases executed
- No critical or high-severity defects open
- 95% of medium-severity defects resolved
- All regression tests passed
- Performance meets specified requirements

13.2 Test Completion Report

- Summary of testing activities
- Test coverage metrics

- Defect summary and statistics
- Recommendations for release

14. Approval

Name	Role	Signature	Date
[Name]	Project Manager		
[Name]	QA Lead		
[Name]	Development Lead		

Appendix A: Sample Test Cases

User Registration

1. Register with valid information
2. Register with existing email
3. Register with invalid email format
4. Register with mismatched passwords
5. Register with insufficient password complexity

Product Search

1. Search by keyword
2. Filter by category
3. Filter by price range
4. Sort results by relevance
5. Sort results by price (low to high)
6. Sort results by price (high to low)
7. Sort results by newest arrivals

Shopping Cart

1. Add item to cart
2. Remove item from cart
3. Update item quantity
4. Apply valid discount code
5. Apply invalid discount code
6. Proceed to checkout with empty cart

Appendix B: Test Automation Suggestions

Python code samples for test automation will be provided separately, focusing on:

- Selenium WebDriver setup for browser testing
- API testing with Requests library
- Mobile testing with Appium
- Performance testing with Locust
- Test reporting with PyTest