General Test Plan for JPetStore

Test Objectives:

1. Registration
2. Login
3. Order
   a) Add item to Shopping Cart
   b) Checkout
   c) Viewing order information

Sources:

Design, SRS - Input documents

Tasks:

- Analyse Documentation, Meet with team and discuss functionalities.
- Refinment:
  - Discuss backlog items, and make some communication around Acceptance Criterias,
  - Improve some points,
  - Determine which elements will be automated.
- Creating Test Documentation in TestLink:
  - Requirements,
  - Test Cases.
- Development autotests.

Estimation:

We should make test estimation before each Sprint to understand approximately time to be spend on testing.

Test Design:

Create Test Cases and Checklists using different tests design and test analysis techniques.

Test Case creation using TestLink.
TestLink URI: http://0.0.0.0:8080/index.php

Test Case example:

JPS-2 : Login user with valid data. - Version1

Summary

Test the login functionality of the web application to ensure that a registered user is allowed to login with username and password.

Preconditions

The User must be created in DB.

If needed create User using TC [JPS-1]

#	Step actions	Expected Results	Execution
1	Click "Sign In" button in Header.	Should be redirected to the "signonForm" page.	Manual
2	Fill in "Username" field with **VALID** data.	The "Username" field is filled.	Manual
3	Fill in "Password" field with **VALID** data.	The "Password" field is filled.	Manual
4	Click "Login" button.	Should be redirected to the Main Page. On Main Page appears text "Welcome {First name}!" In Header: • "Sign In" button must be change to "Sign Out". • Should appears "My Account" button.	Manual

Tools:

- Mind Map,
- TestLink,
- Jira,
- ChromeDevtools,
- Chrome plugins (Fake Filler),
- Pycharm,
- GitHub.

Resources:

- QA1 – Manual testing
- QA2 – Automation testing

Test Execution:

1. Execution started when User story moved to “In Testing” column in Jira.
2. Run Test Execution in TestLink using “Execute Tests” button.

Bug Reporting:

1. Create Bug in TestLink.
2. Link Bug to User story in Jira.

Stages of Testing:

1. Testing of all User stories and tasks during Sprint.
2. Smoke Testing.
3. Developing autotests.
4. Regression Testing.

Test Environments:

- Dev
- PreProd
- Prod

Platforms and environments:

- **OS systems:**
  - Windows
  - MacOS
- **Browsers**
  - Chrome
  - Safari
  - Firefox

Test Strategy:

Objectives and actions	Priority	Strategy
Authentication		
Registration	High	Main functionality. This functionality will be automated first. We will provide: • *Positive testing*: Validate that users can successfully register by providing correct and complete information. • *Negative testing*: Validate that appropriate error messages are displayed for incomplete, incorrect, or invalid input data. • *Security testing*: Verify the implementation of security measures such as password encryption and protection against common vulnerabilities (e.g., SQL injection). • *Usability testing*: Evaluate the user-friendliness of the registration process and identify any areas for improvement. For testing this functionality need to use State Transition, Error Guessing and Equivalent Class testing techniques.
Sign In	High	Main functionality. This functionality will be automated first. We will provide: • *Positive testing*: Validate that users can successfully register by providing correct and complete information. • *Negative testing*: Validate that appropriate error messages are displayed for incomplete, incorrect, or invalid input data. • *Security testing*: Verify the implementation of security measures such as password encryption and protection against common vulnerabilities (e.g., SQL injection). • *Usability testing*: Evaluate the user-friendliness of the registration process and identify any areas for improvement. For testing this functionality need to use State Transition, Error Guessing and Equivalent Class testing techniques.
Products		
View a products list	Medium	We will provide: • *Functional Testing*: Conduct end-to-end testing of the view product functionality to validate all product details and navigation features. • *Cross-browser and Cross-device Testing*: Test the functionality on different browsers (e.g., Chrome, Firefox, Safari, Edge) and devices to ensure consistent behavior and layout. • *Usability Testing*: Engage users or usability experts to evaluate the user-friendliness and intuitiveness of the view product functionality. For testing this functionality need to use State Transition and Error Guessing. This functionality will be automated in second time.
Search for items by related attributes	Low	We will provide: *Functional Testing*: • Verify the search box is visible and functional on all relevant pages. • Test basic search functionality using different keywords and verify the accuracy of search results. *Usability Testing*: • Evaluate the search user interface for intuitiveness and ease of use. • Verify that error messages or empty result notifications are clear and helpful. • Test the search functionality with different user personas to ensure it caters to various user needs. For testing this functionality need to use State Transition and Equivalent Class testing techniques. This is low-priority functionality and it will be automated last.
Order		
Adding item to Shopping cart	High	This is main functionality. It will be automated first. We will provide: • *Functional testing*: Ensure that the basic functionality of adding items to the cart is working correctly. • *Usability testing*: Validate that the process of adding items to the cart is intuitive and user-friendly. • *Cross-browser testing*: Verify that the functionality works consistently across different web browsers. • *Cross-device testing*: Ensure that the feature functions properly on various devices with different screen sizes and resolutions. • *Integration testing*: Check the integration between the shopping cart functionality and other components/modules of the web shop. For testing this functionality need to use State Transition and Error Guessingtesting techniques.
Modifing Shopping cart	Medium	Need to ensure thorough testing of the modified shopping cart functionality, covering all relevant aspects and scenarios. The testing will focus on validating the changes made to the shopping cart, ensuring its functionality and usability. This functionality will be automated in second time. The scope of the testing will include the following aspects of the modified shopping cart functionality: • Adding items to the cart • Removing items from the cart • Updating quantities of items in the cart • Applying discounts or promotional codes • Calculating and displaying the subtotal, taxes, and total • Verifying cart persistence across sessions.
Checkout an order from the shopping cart	High	This is main functionality. It will be automated first. The scope of the testing will include: • Testing of the entire checkout process.. • Verifying the accuracy of order details such as item quantity, pricing, and shipping information. • Testing various payment methods and ensuring successful payment processing. • Validation of error handling and messages displayed to users during the checkout process. • Integration testing with external systems such as payment gateways or shipping providers, if applicable. For testing this functionality need to use State Transition and Error Guessingtesting techniques.
View the information about order	Medium	This includes verifying that order details, such as order status, shipping information, payment details, and product information, are displayed correctly to the user. • *Functional Testing*: Verify that order information is displayed accurately and completely. • *Cross-browser Testing*: Test the functionality on different browsers (e.g., Chrome, Firefox, Safari) to ensure compatibility. • *Negative Testing*: Test scenarios such as canceled orders, orders with missing information, or invalid orders to validate error handling. This functionality will be automated in second time.

Test Metrics:

Metrics to consider for reporting purposes include:

- Number of test cases executed and passed/failed.
- Defect density (defects per test case or test hour).
- Test coverage (percentage of requirements covered by tests).
- Defect severity and priority.
- Time taken for each test phase.
- Overall test execution progress.

Test Reporting:

The test results and defects identified during the testing process should be documented and reported effectively. The test report should include:

- Summary of the testing activities conducted.
- Test coverage and traceability matrices.
- Test execution results, including passed and failed test cases.
- Defect reports with detailed information, severity, and priority.
- Recommendations for further improvements or areas of concern.

Risks and Contingencies:

Identify potential risks and associated contingencies, such as:

- ***Risk***: Incomplete or inaccurate requirements.
  - *Contingency*: Collaborate closely with stakeholders to clarify requirements and obtain necessary information.
- ***Risk***: Lack of test coverage for edge cases or specific scenarios.
  - *Contingency*: Perform thorough

Test Sign-off:

Once testing is completed and all critical defects are fixed, obtain the necessary approvals and sign-off from relevant stakeholders, including project managers, business analysts, and product owners.