**SOFTWARE TEST PLAN**



**Otaku House**

**TABLE OF CONTENT**

1. **INTRODUCTION**
   1. Scope
2. Product Display and Listing:

* The website shall display the latest products available for purchase, which includes various categories such as figures, puzzles, and apparel.
* Each product should have an image, name, price, and rating displayed on the initial listing page.

1. Product Search Functionality:

* Users shall be able to search for products using keywords or product names.
* The search function should return relevant products based on the query input by the user.

1. Product Detail Page:

* Upon clicking a product, users shall be taken to a product detail page that includes an enlarged image, detailed description, price, availability status, and user reviews.
* The product detail page will allow users to select the quantity and add the product to their shopping cart.

1. Shopping Cart:

* Users shall be able to view items they have added to their shopping cart.
* The cart will display the product image, name, price, selected quantity, and the subtotal for all items.
* Users can proceed to checkout from the shopping cart page.

1. User Reviews:

* Registered users shall have the option to write and submit reviews for products.
  1. Glossary

1. API : A set of routines, protocols, and tools for building software and applications. It allows different software programs to communicate with each other.
2. Browser Compatibility: The ability of a website to function properly across different web browsers.
3. Cart: A virtual basket that holds items a user intends to purchase.
4. Checkout: The process through which a customer finalizes their purchase, including payment and delivery information.
5. Functional Testing: A type of testing that involves ensuring that the functions of the software application operate in conformance with the requirement specification.
6. Non-Functional Testing: Testing conducted to evaluate the readiness of a system according to nonfunctional parameters (performance, usability, reliability, etc.) that are not covered by functional testing.
7. Test tree
   1. Functional tests:
8. Login page:
   1. Sign in with email

* Login with existing username and password.
* Login with correct username and wrong password.
* Login with not exist username.

1. Search product:
   1. With account

* Search valid product with login account
* Search valid product without login account

1. Product detail page:
   1. Product image and information

* Checking the correctness of products

1. Buy a product:
   1. Logging in to the account
   2. Not Logging in to the account
2. Shopping Cart
   1. Cart persistence
3. Not Logging in to account
   * Check if Items remain in the Cart after exit from the website
4. Logging in to account
   * Check if Items remain in the Cart after exit from the website
5. Update quantity:

* The price is change after update the quantity
* the updating quantity is saved after Logging out and login again

1. Remove item from Cart
2. if there is 1 item

* The Cart is “empty “and the subtotal is 0 and navigate to “start shopping page.”

1. if there is more than 1 item

* Removing some of the items and the subtotal is updating, and the page still Exists. (Cart persistence).

1. Checkout:
2. Logging in to account
3. Without logging to account

Non functional tests:

1. Maintainability Testing:

* Change Management Testing: Evaluating the ease with which new changes can be implemented into the existing system without introducing new issues.
* Documentation Review: Checking that all system documentation is clear, correct, and helpful.

1. Localization Testing:

* Currency Testing: If the website supports multiple currencies, test that all currency-related functionality works as expected.

1. Compatibility Testing:

* Browser Compatibility: Ensure that the website works across different web browsers like Chrome, Firefox, Safari, and Edge.

E2E tests:

1. Complete Purchase Cycle:

* User logs into their account.
* User searches for a specific product.
* User selects a product and adds it to the cart.
* User proceeds to checkout.
* User fills in payment information (mock or sandbox environment).
* User completes the purchase.
* Verify confirmation and summary of the order is displayed.
* User logs out.

1. Product Review by Logged-In User:

* User logs into their account.
* User navigates to a previously purchased product.
* User submits a review and rating for the product.
* Verify the review is posted under the product.

1. Features to be tested

All the features of the Otaku House website which were defined in software requirement specs:

|  |  |  |
| --- | --- | --- |
| Module name | Applicable Roles | Description |
| Login | User | * Confirm that the log in success when login with correct user name and correct password. * Confirm that the login not success when login with user name is correct and password incorrect. * Confirm that the login not success when login with an incorrect user name. * Login using a connected account. |
| payment | User | * Confirm that the payment is processed successfully. * The user receives a payment confirmation by email. * Verify that the booking is a success. |
| Search | User | * Show relevant items to the description in the search. |

* 1. Features will not be tested

These features are not be tested because they are not included in the software requirement specs

-Software Interfaces

-Database logical

-Communications Interfaces

-Website Security

1. Test type

To ensure the strength and reliability of shopping and selling website, a comprehensive testing strategy has been devised, through various types of testing to address both functional and non-functional aspects.

Functional tests:

1. Integration Testing:

* Verify the interaction between different modules.
* Ensure seamless data flow and functionality.

1. API Testing:

* Validate APIs handling data requests and responses.
* Confirm accurate data exchange between the website and external services.

1. Regression Testing:

* Ensure that new updates or changes do not adversely affect existing functionalities.
* Conduct regression testing after each update to validate the overall system stability.

Non-functional tests:

1. Performance Testing:

* Assess responsiveness and load handling capabilities.
* Conduct stress testing to evaluate performance under high user loads.

1. Security Testing:

* Identify and address potential vulnerabilities.
* Test for secure data transmission and storage practices.

1. Compatibility Testing:

* Ensure compatibility across different browsers and devices.
* Validate functionality on various operating systems.

1. Usability Testing:

* Evaluate the overall user experience during the booking process.
* Check for intuitive navigation and user-friendly design.

1. Load Testing:

* Assess the website's ability to handle specific loads.

1. Volume Testing:

* Evaluate system performance with a large volume of data.

1. Stress Testing:

* Test system robustness under extreme conditions.

1. Recovery Testing:

* Verify the system's ability to recover from failures.

1. Localization Testing:

* Test the website for different languages and regions if applicable.
* Ensure that the content and features of the website adjust well to different languages and cultures.

1. Risk and issues

|  |  |
| --- | --- |
| **Risk** | **Mitigation** |
| **Delays in project timelines.** | Implement agile project management practices to allow for flexible planning and adjustments. Holding regular stand-ups to address and mitigate delays promptly. |
| **Poor communication among team members and stakeholders.** | Establish clear communication channels and regular update meetings. Using project management tools to keep everyone informed and engaged. |
| **Changes in project requirements.** | Implement a change management process to evaluate the impact of changes on budget, timeline, and quality. Ensuring stakeholder alignment on changes before implementation. |
| **Inadequate testing coverage leading to quality issues.** | Using risk-based testing to prioritize testing areas. Implement automated testing for repetitive tasks to increase coverage. Regularly reviewing and adjusting test plans as the project evolves. |
| **The project schedule is too tight; it's hard to complete the whole project on time.** | Setting Test Priority for each of the test activities. |

1. Test logistic

5.1. Who will test ?

The project will be tested by Beyond Dev QA Automation Student.

5.2. When will the test occur?

Having the tests is a crucial part of the planning process.

The timing of tests can be broadly categorized into several phases of the software development lifecycle and based on specific criteria.

-During requirements analysis.

-After Design Phase.

-During Development.

-Before and After Deployment.

1. Test Objective

The primary objective of our QA automation testing for the eBay website is to validate the functionality, performance, usability, and security of key features and processes. Through systematic, automated testing, it's aim to :

Ensuring Functional Accuracy Verifying that all features of eBay, including user registration, listing creation, search functionality, bidding processes, and payment transactions, perform as expected. This involves checking the correctness of operations and data integrity across different user scenarios.By achieving these objectives, the QA automation project aims to contribute significantly to the overall quality assurance of the eBay platform, enhancing user satisfaction, operational efficiency, and the platform's reputation in the marketplace.

1. Test criteria

7.1. Suspension Criteria

If the team members report that there are 40% of test cases failed, suspend testing until the development team fixes all the failed cases.

The tests are tests that constitute a better and higher quality user experience.

7.2. Specifies the criteria that denote a successful completion of a test phase

-Run rate is mandatory to be 100% unless a clear reason is given.

-Pass rate is 80%, achieving the pass rate is mandatory.

1. **RESOURCE PLANNING**

8.1 System Resource

|  |  |  |
| --- | --- | --- |
| **No.** | **Resources** | **Descriptions** |
| 1 | **Test Automation Server** | A dedicated server for running automated tests, including scheduling and executing test suites. |
| 2 | **Version Control System** | Tools like Git for source code management, allowing for version tracking and collaboration. |
| 3 | **Continuous Integration (CI) Tools** | CI platform (e.g., Jenkins, CircleCI) for automating the integration of code changes. |
| 4 | **Test Management Tool** | Software to manage test cases, plans, runs, and reporting (e.g., TestRail, Zephyr). |
| 5 | **Issue Tracking System** | Tool for tracking bugs and issues (e.g., JIRA, Bugzilla). |
| 6 | **Automated Testing Frameworks** | Frameworks and libraries for writing and executing tests (e.g., Selenium, Appium). |
| 7 | **Browser and Device Lab** | A collection of physical or virtual devices and browsers for testing website compatibility. |
| 8 | **Performance Testing Tools** | Tools for load and stress testing (e.g., JMeter, LoadRunner). |
| 9 | **Security Testing Tools** | Software for identifying vulnerabilities in the web application (e.g., OWASP ZAP, Burp Suite). |
| 10 | **Database Servers** | Servers hosting databases for test environments, mirroring production setups. |
| 11 | **API Testing Tools** | Tools specifically designed for testing APIs (e.g., Postman, SoapUI). |
| 12 | **Monitoring and Logging Tools** | Tools for monitoring system performance and logging during test execution (e.g., ELK stack). |

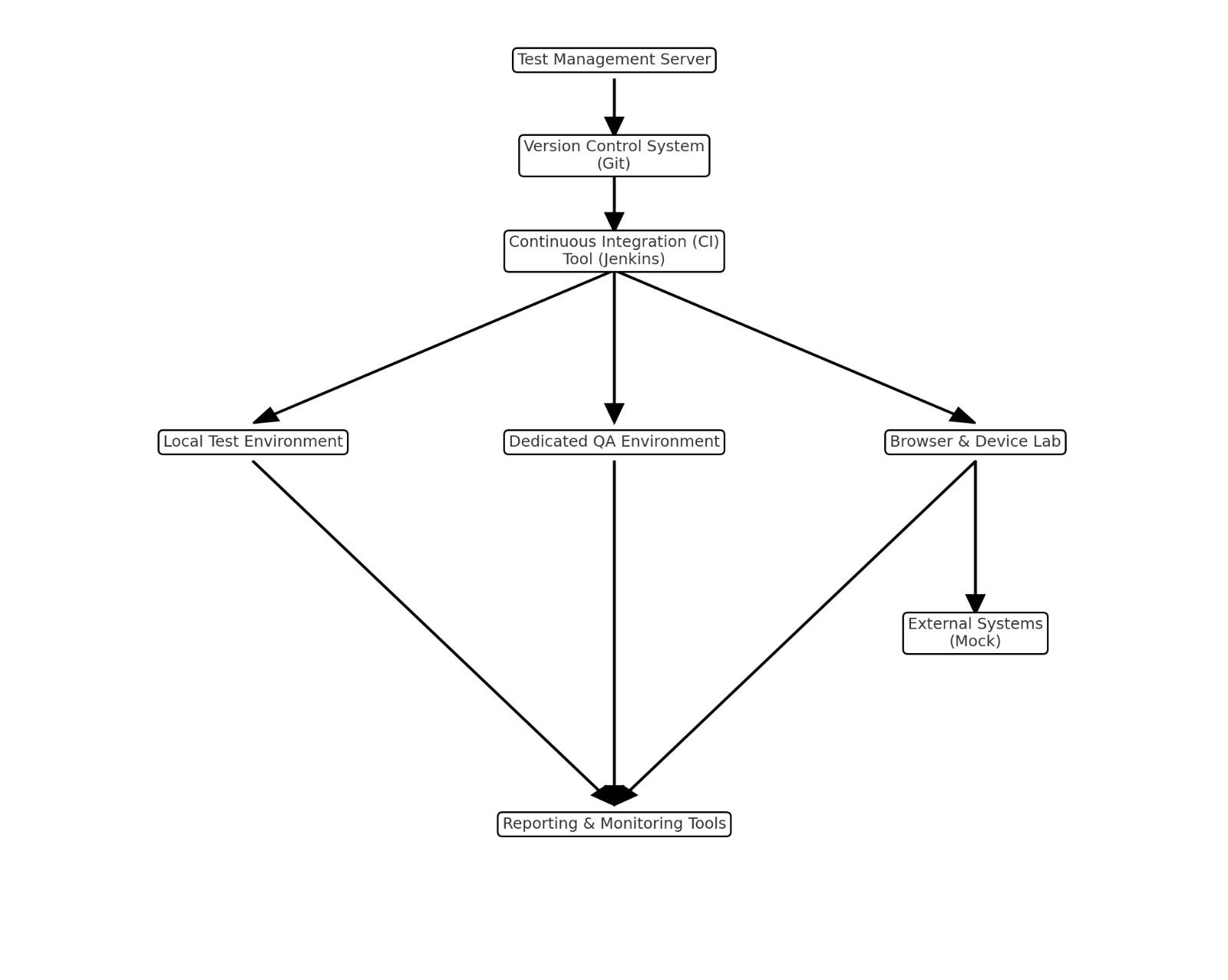
8.2. Human Resource

|  |  |  |
| --- | --- | --- |
| **No.** | **Members** | **Tasks** |
| 1 | Test Lead | Overseeing the test strategy, planning, and ensuring quality standards are met. |
| 2 | Automation Engineer(s) | Designing, developing, and maintaining test automation scripts. |
| 3 | Manual Tester(s) | Executing manual test cases that are not automated and assisting in exploratory testing. |
| 4 | Performance Tester | Specializing in performance testing, analyzing results, and identifying bottlenecks. |
| 5 | Security Tester | Conducting security assessments and vulnerability testing to ensure application security. |
| 6 | DevOps Engineer | Managing CI/CD pipelines, environments, and supporting automation infrastructure. |
| 7 | QA Analyst | Analyzing test results, reporting bugs, and verifying fixes. |
| 8 | Project Manager | Coordinating between teams, managing timelines, and ensuring project goals are met. |
| 9 | Business Analyst | Ensuring the test cases align with business requirements and user stories. |
| 10 | Stakeholder(s) | Providing feedback, prioritizing features, and ensuring the project aligns with business goals. |

1. **TEST ENVIRONMENT**

The Test Environment should be setup as figure below.

test environment for ebay.com that include user, database and web server.



This visual represents the main components and their interactions, including the Test Management Server, Version Control System (Git), Continuous Integration (CI) Tool (Jenkins), various test environments (Local, Dedicated QA, and Browser & Device Lab), External Systems (Mocks), and Reporting & Monitoring Tools. Arrows indicate the flow of information and control between these components.

1. **SCHEDULE & ESTIMATION**

10.1. All project task and estimation

|  |  |  |
| --- | --- | --- |
| **Task** | **Members** | **Estimate effort** |
| **Create the test specification** | Test Administrator | 10 man-hour |
| **Perform Test Execution** | Tester, Test Administrator | 7 man-hour |
| **Test Report** | Tester | 10 man-hour |
| **Test Delivery** | Tester | 8 man-hour |
| **Testing Code** | Tester | 70 man-hour |
| **Total** |  | **105 man-hour** |

1. **TEST DELIVERABLES**

refer to the tangible items or documents produced as a result of the testing process.

Test deliverables are provided as below

1. Test cases: scenarios that we are going to test is a case of a person who is using the product and is going to make an order when he registers for the Guinness program and he will make an order and receive an order confirmation by email

2. Test scripts: manual scripts and automation with unit test cases and run the tests to see the performance against the expected scenarios.

3. Test data: We need user login details and internet connection and the website address.

11.1. Before testing phase

Test plans document.

Test cases documents.

11.2. During the testing

during the test

test tool

simulators.

test data

Email is open to receive notifications

User details for connection and testing

11.3 After the testing cycles is over

Test Results/reports

Defect Report

Installation/ Test procedures guidelines