Test Plan for Starschiners.ro

Change Log

Version	Change date	by
1.1	23.05.23	Sabin Marcu

Introduction

The purpose of this test plan is to outline the testing approach and strategy for the login, logout, sign in with Facebook, forgot password, search, add to cart, and checkout functionalities of the system/application (Starschiners.ro) under test.

By following this test plan, we aim to identify and address any issues or defects related to login, logout, sign in with Facebook, forgot password, search, add to cart, and checkout. This will help in delivering a high-quality application that provides a seamless and satisfying user experience.

Test Plan workflow for Starchiners.ro website:

1. Test Plan Overview
2. Test Environment Setup
3. Test Data Preparation
4. Login Test
5. Sign up Test
6. Sign in with Facebook Test
7. Forgot Password Test
8. Search Test
9. Add to Cart Test
10. Checkout Test
11. Error Handling Testing

For this project we use Agile methodology, commonly used for software development. It emphasizes flexibility, collaboration, and iterative development. In the context of login, logout, sign in with Facebook, forgot password, search, add to cart, and checkout functionalities, Agile can be applied to manage the development process more efficiently.

Using Agile, these functionalities can be broken down into smaller user stories, which are then prioritized and developed in short iterations called sprints. The development team works closely with stakeholders, such as product owners or users, to gather feedback and make continuous improvements.

Agile methodologies provide frameworks for organizing and managing these iterative development cycles.

Scope

In scope

In scope features refer to the functionalities or requirements that are included in the current scope of a project. These are typically features that are considered necessary or relevant for the immediate goals of the project. Here are in scope features for the functionalities: login, signup, sign in with Facebook, forgot password, search, add to cart, and checkout

1. Login:
   - Features:

User authentication using username/email and password.

- Remember me functionality.
- Password reset option.
- Functional Requirements:
  - Validate user credentials.
  - Maintain session/authentication state.
  - Provide error messages for invalid credentials.

2. Sign up:
   - Features:
     - Notifications and Communication Preferences:
     - Error Handling and Validation.
     - Registration Form
     - User Authentication

3. Sign in with Facebook:
   - Features:
     - Authenticate users using their Facebook credentials.
     - Import user profile information from Facebook.
   - Functional Requirements:
     - Retrieve user profile information from Facebook API.
     - Link the Facebook account to the user's account in the system.
   - Non-functional Requirements:
     - Security measures to protect user data imported from Facebook.
     - Compliance with Facebook's API guidelines.
4. Forgot Password:

   - Features:
     - Allow users to reset their forgotten passwords.
     - Send password reset instructions via email.
   - Functional Requirements:
     - Provide a password reset form/page.
     - Validate user identity through email
     - Generate and send a unique password reset link to the user's email.
   - Non-functional Requirements:
     - Secure transmission and storage of password reset links.
     - Prompt delivery of password reset emails.
5. Search:
   - Features:
     - Search for products or content within the application.
     - Provide search suggestions and autocomplete.
     - Filter and sort search results.
   - Functional Requirements:
     - Accept user search queries.
     - Query the application's search index/database.
     - Display relevant search results.
   - Non-functional Requirements:
     - Efficient indexing and retrieval of search results.
     - Fast response time for search queries.
6. Add to Cart:
   - Features:
     - Allow users to add products/items to their shopping cart.
     - Display the cart summary and total.
   - Functional Requirements:
     - Provide an "Add to Cart" button or functionality.
     - Update the cart with the selected product/item.
     - Calculate and display the updated cart summary and total.
   - Non-functional Requirements:
     - Persistent storage of cart data for logged-in users.
     - Fast and reliable cart updates.
7. Checkout:
   - Features:
     - Enable users to review and finalize their purchase.
     - Collect shipping and payment information.

    - Place an order and generate an order confirmation.
  - Functional Requirements:
    - Display a summary of the user's cart.
    - Collect shipping address and payment details.
    - Validate and process the payment.
    - Generate an order confirmation with a unique order ID.
  - Non-functional Requirements:
    - Secure transmission and storage of payment information.
    - Confirmation email with order details and tracking information.

Out of scope

1. Login:
    - Two-factor authentication, social media integration (e.g., sign in with Facebook), biometric authentication (e.g., fingerprint or face recognition).
    - Functional requirements: Validate username and password, handle incorrect login attempts, redirect to the appropriate user dashboard or homepage upon successful login.
    - Non-functional requirements: Ensure secure transmission of login credentials (e.g., HTTPS), implement account lockout mechanism after multiple failed attempts, support password complexity requirements.
2. Sign up:
    - Social media integration: twitter
    - Two-factor authentication (2FA):
3. Sign in with Facebook:
    - Sign in with other social media platforms (e.g., Google, Twitter), import user profile information from Facebook.
    - Functional requirements: Authenticate the user with Facebook credentials, create a new account or link with an existing account, retrieve and store relevant user information.
    - Non-functional requirements: Ensure secure communication with the Facebook API, handle errors or exceptions during the authentication process.

4. Forgot password:
    - Password recovery via email or SMS, security questions.
    - Functional requirements: Provide a "Forgot password" link or option, prompt the user to enter their registered email address, send a password reset link or temporary password to the user's email.
    - Non-functional requirements: Ensure secure transmission of password reset information, validate and sanitize user input, handle errors gracefully.

5. Search:

    Advanced search filters, search suggestions, auto-complete.

    - Functional requirements: Accept user input and query the database or search index, retrieve relevant search results, display the results to the user.
    - Non-functional requirements: Optimize search performance, handle large result sets efficiently, support multi-language search if required.

6. Add to cart:
    - Save for later, wish list.
    - Functional requirements: Allow users to add products to their shopping cart, update the cart contents, display the current cart items to the user.
    - Non-functional requirements: Ensure the cart data is persisted across sessions, handle concurrent updates to the cart, support scalability for high traffic.

7. Checkout:
    - Payment gateway integration
    - Order tracking and shipment notifications

Quality objective

Starschiners website aims to achieve the following outcomes:

1. User-Friendly Interface: The website should have an intuitive and easy-to-navigate interface, allowing customers to find products quickly and efficiently.
2. Responsive Design: The website should be optimized for various devices (desktop, mobile, tablet) to provide a consistent and visually appealing experience across platforms.
3. Fast Loading Speed: Pages and product listings should load quickly to minimize waiting times and provide a smooth browsing experience for customers.
4. Accurate Product Information: Product descriptions, images, and pricing should be accurate and up to date, ensuring customers have the right information to make informed purchasing decisions.
5. Secure Payment Process: Implement robust security measures to protect customer information during the payment process, ensuring a secure and trustworthy transaction environment.
6. Efficient Order Processing and Fulfillment: Streamline order management processes to minimize errors, promptly confirm orders, and provide timely updates on shipping and delivery status.
7. Responsive Customer Support: Provide responsive and helpful customer support through various channels (email, live chat, phone) to address inquiries, resolve issues, and enhance customer satisfaction.
8. Continuous Improvement: Regularly collect and analyze customer feedback, monitor website performance metrics, and implement improvements to enhance the overall shopping experience.

Testing project objectives

- Identify defects
- Ensure functionality
- Enhance usability and user experience
- Ensure security and reliability
- Validate compatibility

- Ensure the Application Under Test conforms to functional and nonfunctional requirements
- Ensure the AUT meets the quality specifications defined by the client
- Bugs/issues are identified and fixed before go live

Roles and responsabilities

IT Manager/Team Lead:

- Oversees the IT team's operations and projects.
- Sets goals and objectives for the team and ensures they are met.
- Manages the budget and resources for IT projects.
- Collaborates with other departments to understand their IT needs and align them with the team's capabilities.
- Provides guidance and support to team members, and resolves any issues or conflicts that arise.
- Stays updated on emerging technologies and industry trends to make informed decisions.

Systems Administrator:

- Installs, configures, and maintains computer systems, servers, and networks.
- Monitors system performance and troubleshoots issues.
- Manages user accounts, permissions, and access controls.
- Performs backups and ensures data integrity and disaster recovery.
- Implements security measures to protect systems from unauthorized access or attacks.
- Evaluates and recommends hardware and software upgrades or replacements.

Network Administrator:

- Designs, installs, and maintains the organization's network infrastructure.
- Configures and manages network devices such as routers, switches, and firewalls.
- Monitors network performance and optimizes network traffic.
- Implements security measures to protect the network from threats.
- Manages network protocols, IP addressing, and domain name systems (DNS).
- Troubleshoots network issues and resolves connectivity problems.

Database Administrator:

- Designs, implements, and maintains the organization's databases.
- Ensures data integrity, security, and availability.
- Performs database backups, restores, and recovery procedures.
- Optimizes database performance and resolves performance issues.
- Manages database access and user permissions.
- Monitors database usage and capacity planning.

Software Developer/Engineer:

- Designs, develops, tests, and maintains software applications.
- Collaborates with stakeholders to gather requirements and propose technical solutions.
- Writes clean, efficient, and maintainable code.
- Conducts code reviews and debugging to ensure software quality.
- Implements software version control and deployment strategies.
- Stays updated on programming languages, frameworks, and best practices.

Help Desk/Technical Support:

- Provides technical assistance and support to end-users.
- Responds to and resolves user inquiries, issues, and requests.
- Troubleshoots hardware and software problems.

- Guides users in the setup and use of IT systems and applications.
- Documents and maintains support tickets and knowledge base articles.
- Escalates complex issues to appropriate teams if necessary.

IT Security Specialist:

- Implements and monitors security measures to protect IT systems and data.
- Conducts vulnerability assessments and penetration testing.
- Develops and enforces security policies and procedures.
- Responds to security incidents and conducts investigations.
- Implements security awareness training programs for employees.
- Stays updated on the latest security threats and countermeasures.

IT Project Manager:

- Plans, executes, and monitors IT projects.
- Defines project scope, objectives, deliverables, and timelines.
- Allocates resources and manages project budgets.
- Collaborates with stakeholders to gather project requirements.
- Monitors project progress and manages risks and issues.
- Communicates project status and reports to stakeholders.

QA Manager/Lead:

- Oversees the entire QA process and ensures adherence to quality standards.
- Develops QA strategies, test plans, and policies.
- Assigns tasks to QA team members and monitors their progress.
- Collaborates with other teams to establish quality goals and objectives.
- Manages the QA team's resources, budget, and timelines.
- Reports on QA progress and provides recommendations for improvement.

Test Analyst/Engineer:

- Analyzes project requirements, specifications, and design documents.

- Designs test cases, test scenarios, and test data.
- Executes test cases and documents test results.
- Identifies defects or issues and reports them in a clear and concise manner.
- Collaborates with developers and stakeholders to resolve defects.
- Performs regression testing to ensure the stability of existing functionality.
- Conducts various types of testing, such as functional, integration, performance, and security testing.
- Uses automated testing tools and frameworks to increase testing efficiency.

Test Automation Engineer:

- Develops and maintains automated test scripts and frameworks.
- Identifies test cases suitable for automation.
- Executes automated tests and analyzes test results.
- Integrates automated tests into the continuous integration and deployment pipeline.
- Collaborates with developers to ensure test automation is integrated into the development process.
- Monitors and maintains test automation infrastructure and tools.
- Provides guidance and training to other team members on test automation best practices.

QA Coordinator:

- Assists the QA Manager in coordinating QA activities.
- Schedules and organizes testing activities and resources.
- Tracks and communicates the status of test executions.
- Manages test environments and test data.
- Assists in preparing test reports and metrics.
- Identifies and escalates potential risks or issues to the QA Manager.
- Supports the QA team with administrative tasks and documentation.

QA Specialist:

- Specializes in a specific area of QA, such as performance testing, security testing, or usability testing.
- Conducts in-depth analysis and evaluation of the assigned area.
- Designs and executes specialized tests and methodologies.

- Provides expertise and recommendations related to the specific area of QA.
- Collaborates with other team members to ensure comprehensive test coverage.
- Stays up to date with the latest industry trends and best practices in the specialized area.

QA Documentation Specialist:

- Creates and maintains QA documentation, including test plans, test cases, and test scripts.
- Ensures documentation is accurate, comprehensive, and up to date.
- Organizes and maintains a central repository for all QA documents.
- Collaborates with team members to gather and review documentation requirements.
- Conducts regular reviews and audits of documentation for quality and compliance.
- Provides training and support to team members on documentation standards and processes.

Test methodology

Overview

Those are some reasons why I choose to adopt agile methodologies in my projects:

- Flexibility and Adaptability: Agile methodologies, such as Scrum or Kanban, emphasize adaptability and responding to change. They allow for frequent iterations and adjustments based on customer feedback and evolving requirements. This flexibility enables teams to quickly adapt to shifting priorities, market conditions, and customer needs, ensuring the project stays aligned with the desired outcomes.
- Faster Time to Market: Agile approaches promote incremental and iterative development, enabling teams to deliver working software or product features in short cycles called sprints. By focusing on smaller, manageable increments, teams can deliver value more frequently, reducing time to market compared to traditional waterfall approaches. This rapid feedback loop allows for early validation and course correction if necessary.
- Customer Collaboration and Satisfaction: Agile methodologies emphasize close collaboration with stakeholders, including customers, throughout the project. Frequent customer involvement ensures that the delivered product aligns with their expectations and provides value. By involving customers early and regularly, teams can incorporate their feedback, address concerns, and make necessary adjustments, resulting in higher customer satisfaction.
- Continuous Improvement: Agile methodologies encourage continuous learning and improvement. Through regular retrospectives, teams reflect on their processes, identify areas for improvement, and make necessary adjustments. This iterative feedback loop promotes a culture of learning, experimentation, and innovation, leading to enhanced project performance over time.
- Increased Transparency and Communication: Agile frameworks promote transparency and open communication within the project team and with stakeholders. Daily stand-up meetings, sprint reviews, and other agile ceremonies facilitate frequent communication, ensuring everyone is aware of the project's progress, challenges, and upcoming work. This transparency helps identify and address issues early, mitigating risks and fostering collaboration among team members.

- Risk Mitigation: Agile methodologies provide opportunities for early risk identification and mitigation. By breaking down the project into smaller increments and delivering value iteratively, teams can uncover risks and challenges sooner. This early visibility allows for timely risk mitigation strategies, avoiding potential project delays or failures.
- Empowered and Motivated Teams: Agile methodologies promote self-organizing, cross-functional teams. Team members are empowered to make decisions, collaborate, and take ownership of their work. This autonomy and sense of ownership often lead to increased motivation, creativity, and productivity among team members.

Test levels

When considering test levels for various functionalities in an application, including login, logout, sign in with Facebook, reset password, add to cart, search, and checkout, the following test levels can be considered:

- Unit Testing: Unit testing involves testing individual components or modules in isolation. For login, logout, reset password, and sign in with Facebook functionalities, unit tests can be performed to ensure that these components work correctly on their own. This may involve testing validation logic, authentication mechanisms, and handling of user input.
- Integration Testing: Integration testing verifies the interaction between different components and modules. For sign in with Facebook functionality, integration tests can be conducted to ensure that the application integrates correctly with the Facebook API or any external authentication service being used. Integration tests for other functionalities, such as adding to cart or searching, may involve checking the integration between the frontend and backend systems, database interactions, and third-party service integrations.
- System Testing: System testing involves testing the entire system as a whole, including all interconnected components and modules. For login, sign up, sign in with Facebook, reset password, add to cart, search, and checkout functionalities, system tests can be performed to validate end-to-end scenarios. This includes checking if the user can successfully perform these actions, data integrity is maintained, error handling is appropriate, and the user experience is seamless.

- Regression Testing: Regression testing ensures that existing functionalities continue to work correctly after new changes or updates are introduced. Whenever modifications are made to any of the functionalities mentioned, regression tests should be executed to verify that the changes did not introduce any unintended issues or break existing features.
- Performance Testing: Performance testing focuses on evaluating the performance and responsiveness of the application under different load conditions. For functionalities like search, add to cart, and checkout, performance tests can be conducted to assess how the system handles concurrent user interactions, response times, and resource usage.
- Security Testing: Security testing is crucial for functionalities like login, sign up, sign in with Facebook, reset password, and checkout, as these involve sensitive user data and authentication mechanisms. Security tests can be conducted to identify vulnerabilities, such as authentication bypass, SQL injection, or cross-site scripting, and ensure that proper security measures are in place to protect user information.
- User Acceptance Testing (UAT): UAT involves testing the application from the end-user's perspective. It can be beneficial to involve real users or stakeholders to perform UAT for functionalities like login, sign up, sign in with Facebook, reset password, add to cart, search, and checkout. This helps gather feedback, validate user expectations, and ensure the application meets the desired requirements.

Bug triage

The primary goal of bug triage is to efficiently and effectively manage reported bugs by assigning them to the appropriate teams, determining their severity and priority, and ensuring timely resolution. The key objectives of bug triage, resolution, and prioritization are as follows:

- Efficient Bug Management: Bug triage aims to streamline the bug management process by categorizing and organizing reported bugs. It involves reviewing each bug, understanding its scope and impact, and assigning it to the appropriate team or developer. The goal is to ensure bugs are addressed promptly and by the right individuals or teams, avoiding delays and confusion in bug resolution.
- Bug Severity Determination: During bug triage, the severity of each bug is assessed. Bug severity represents the impact of a bug on the functionality, usability, or performance of the software. The goal is to prioritize bugs based on their severity to ensure critical issues that significantly impact the system's functionality or user experience are addressed with higher urgency.
- Bug Priority Assignment: Prioritization is a crucial aspect of bug triage. Bug priority reflects the order in which bugs should be addressed based on factors like business impact, user impact, risk, and urgency. Prioritization allows teams to allocate resources effectively and focus on resolving critical issues first, ensuring the most important problems are addressed promptly.
- Timely Bug Resolution: The ultimate goal of bug triage is to facilitate timely bug resolution. By assigning bugs to the appropriate teams, determining severity and priority, and ensuring efficient bug management, the aim is to resolve bugs within a reasonable timeframe. Timely bug resolution helps improve software quality, enhances user satisfaction, and minimizes disruptions to project timelines.
- Optimal Resource Utilization: Bug triage involves allocating bugs to the right teams or developers who possess the necessary expertise to address them. By assigning bugs to the appropriate resources, it ensures optimal utilization of available resources and expertise. This leads to more efficient bug resolution, reducing delays and bottlenecks in the overall development process.
- Communication and Collaboration: Bug triage facilitates effective communication and collaboration among stakeholders, including testers, developers, project managers, and product owners. It ensures that all relevant parties have visibility into the reported bugs, their status, and the priority of resolution. Effective communication helps in resolving conflicts, addressing

concerns, and fostering a collaborative environment focused on bug resolution.

- Continuous Improvement: Bug triage provides an opportunity for continuous improvement. By analyzing patterns and trends in reported bugs, teams can identify root causes, recurring issues, or areas of improvement in the development process. This feedback loop helps in implementing preventive measures, enhancing software quality, and reducing the occurrence of similar bugs in the future.

Overall, the goal of bug triage, resolution, and prioritization is to facilitate efficient bug management, timely resolution of critical issues, and continuous improvement of the software development process. It ensures that resources are utilized optimally, stakeholders are well-informed, and the software meets the desired quality standards.

Test completeness

- Requirement Coverage:

  The completion criteria will have at least 95% requirement coverage.

- Code Coverage:

  Test completeness criteria may specify a target percentage of code coverage to be achieved is 90% or higher.

- Test Case Execution:

  Test completeness is 100% of test cases have been executed successfully.

- Defect Closure Rate:

  The criterion could be to close 95% of the reported defects, ensuring that only a minimal number of open issues remain.

- Risk Coverage:

The completeness criteria may include the coverage of high-priority or critical risks.

- Stakeholder Approval:

  In some cases, test completeness can be determined by obtaining formal approval from stakeholders or clients. This involves presenting the test results, demonstrating that the software meets the specified quality criteria, and receiving their sign-off indicating acceptance of the testing effort.

Test deliverables

- Test Plan: It outlines the overall approach and strategy for testing, including objectives, scope, test levels, test types, test environment, and resources.
- Test Cases: These are detailed instructions or scenarios that define the steps to be executed, the expected results, and any necessary test data. Test cases ensure that all aspects of the software or system are adequately tested.
- Test Data: This includes the input data and expected output values used during testing. Test data should cover different scenarios, including normal, boundary, and error conditions.
- Test Environment Setup: Documentation or scripts that describe the required hardware, software, and network configurations needed for testing. It ensures that the test environment is properly prepared and replicates the production environment as closely as possible.
- Test Logs and Defect Reports: These documents capture the details of the test execution, including test case status, actual results, and any defects found. They provide a record of the testing progress and serve as a reference for future analysis or debugging.
- Test Summary Report: It summarizes the overall testing activities, including the number of test cases executed, passed, and failed, along with any critical defects found. The report may also include recommendations for further improvements or actions to be taken.
- Test Completion Certificate: A formal document issued by the testing team to indicate that the testing activities for a specific release or phase have been completed successfully.

- User Acceptance Test (UAT) Results: If applicable, UAT deliverables include test cases, test scripts, test logs, and a summary report that outline the results of testing conducted by end-users or stakeholders.
- Test Metrics and Dashboards: These provide quantitative measurements and visual representations of test progress, test coverage, defect density, and other key performance indicators. They help track the effectiveness and efficiency of the testing process.

• Resource & Environment Needs

• Testing Tools

Requirements tracking tool

- Jira
- Confluence

Bug tracking tool

- Xray for Jira

Test Environment

The minimum hardware requirements that will be used to test the webpage Starschiners.ro:

- Processor: A modern multi-core processor, such as Intel Core i5 or AMD Ryzen 5, would be sufficient for testing webpages.
- Memory (RAM): At least 8GB of RAM is recommended to ensure smooth browsing and avoid performance issues while testing webpages.

- Storage: A standard hard disk drive (HDD) or a solid-state drive (SSD) with sufficient storage capacity should be fine. However, an SSD can offer faster read and write speeds, resulting in quicker page load times.
- Display: A monitor with a resolution of 1920x1080 (Full HD) or higher would be ideal for testing webpages. This allows you to accurately assess the webpage layout and responsiveness across different screen sizes.
- Web Browsers: Ensure you have popular web browsers installed for testing, such as Google Chrome, Mozilla Firefox, Microsoft Edge, and Safari (for Mac users). It's also helpful to have access to different browser versions for compatibility testing.
- Operating System: Use the latest stable version of the operating system that you are comfortable with, such as Windows 10, macOS, or a Linux distribution.