Software Requirements Specification for BargainBay Online Shopping App

1 Introduction

The BargainBay Online Shopping App is designed to provide a seamless and convenient shopping experience for users, offering a wide range of products at discounted prices. This Software Requirements Specification (SRS) outlines the testing approach and strategies for ensuring the software meets its functional requirements effectively.

2 Testing Types and Strategies

2.1 Type of Testing

- Unit Testing: Testing individual components (such as functions or modules) of the software to ensure they work correctly in isolation.
- Component Testing: Testing integrated components to verify their interactions and interfaces.
- System Testing: Testing the entire system as a whole to validate its functionality and performance.

2.2 Testing Strategies

The testing strategy will encompass the following approaches:

- Black-Box Testing: Focuses on testing the functionality of the software without knowledge of its internal structure. This approach mimics how end-users would interact with the system.
- White-Box Testing: Examines the internal structure and implementation details of the software. It tests specific paths, conditions, and statements within the code.
- Static Testing: Analyzes the code or software without executing it. This can include code reviews and inspections.
- Dynamic Testing: Involves executing the software with test data to observe its behavior and verify its functionality.

3 Reasons for Selection

- Unit Testing: Ensures that each part of the software functions correctly on its own before integration, helping to detect and fix defects early in the development process.
- Component Testing: Verifies that integrated components work together seamlessly, reducing the risk of issues during system integration.
- System Testing: Validates the overall system to ensure it meets the specified requirements and behaves as expected in a real-world environment.
- Black-Box Testing: Aligns with the perspective of end-users, focusing on validating the software's external behavior and adherence to functional requirements.
- White-Box Testing: Provides insight into the internal workings of the software, allowing for thorough coverage of code paths and logical conditions.
- Static Testing: Helps identify defects by examining code and software artifacts without executing them, which can be effective for early bug detection.
- Dynamic Testing: Provides insight into how the software performs with different inputs and under varying conditions, aiding in identifying defects related to functionality and performance.

4 Test Cases

4.1 Test Case 1: Product Search and Purchase

Objective: Verify that customers can successfully search for products and complete purchases.
Steps:

1. Enter search keywords or browse product categories.
2. Select desired products and add them to the shopping cart.
3. Proceed to checkout and complete the purchase.

4.2 Test Case 2: Discount Application

Objective: Ensure that discounts are applied correctly to eligible products.
Steps:

1. Add products eligible for discounts to the shopping cart.
2. Apply discount codes or activate promotions.
3. Verify that discounted prices are reflected in the checkout process.

4.3 Test Case 3: User Account Management

Objective: Test the functionality of user account management features.
Steps:

1. Register a new user account or log in with existing credentials.
2. Update account information such as profile details or payment methods.
3. Verify that changes
   are successfully saved and reflected in the user account.

5 Conclusion

The outlined testing approach and strategies aim to ensure the BargainBay Online Shopping App meets the specified requirements, functions reliably, and provides a seamless shopping experience for users. By incorporating rigorous testing practices, we aim to deliver a robust and high-quality software solution that enhances the efficiency and effectiveness of online shopping.