Prepared by:

Mike Dingeldein and Marshall Hobbs

12/28/2021

Objectives

Provide a testing suite for the Lost Chapter bookstore website. The testing suite should provide unit testing, integration testing, and system testing (E2E). The testing suite must allow for regression testing. The development team is developing tests and code in parallel, with the goal of using tests as a gauge of progress. This will allow for a clearer target for the application development.

Tasks

The team will write backend tests using the JUnit5 (Jupiter) framework with Mockito. The front end testing will be automated using Selenium to the maximum practical extent. All other tests will be performed manually. The test suite will have test cases defined in the test case design. The test case design will have the following sections: Use Case ID, Use Case Scenario, Test Case ID, Test, Steps to Test, Expected Result, Actual Result, Status, Defect ID, and notes. Developers will write one test per test case.

SCOPE

General

This test suite will consist of unit testing and integration testing on the backend application, and end to end testing on the frontend.

Tactics

The team is divided into feature groups. Individual feature groups are responsible for unit testing of their respective features. The test group is responsible for end to end testing of the website. The test group will be responsible for the integration testing. Project members that have completed tasks early should assist others members with their tasks.

TESTING STRATEGY

The application team will use agile development and emphasize collaboration. The team will have daily stand-ups to communicate progress and identify any problems as soon as they are encountered. Progress is tracked on the Bach - Lost Chapter - 1191 Trello board originally created by the Revature Center for Excellence. Test cases are derived by the test group from the features.

Tests will follow requirements through use of a requirements traceability matrix. The requirements traceability matrix will include a Business Requirement ID Number, a Requirement Description, a Test Case ID, and a Status.

The team will measure backend test coverage with JaCoCo and SonarCloud. The team will submit a report prepared with JaCoCo and SonarCloud after backend testing is complete. Developers will measure the test percentage coverage of non-trivial methods with corresponding tests. Trivial methods are any method created as part of boilerplate code. Examples of trivial methods are basic accessors, basic modifiers, and the toString method.

Unit Testing

Definition:

Unit tests for the backend will focus on atomic portions of the code. These portions should reflect a single test case defined in the test strategy. Developers will write a series of test cases for each user story. Every test case in every user story must have a unit test. The team will record all tests for their corresponding test case entry. The record for any failed tests must include notes and a defect ID.

Participants:

Unit testing for the features is the responsibility of the feature team.

Methodology:

The feature team will write backend unit tests with JUnit5 (Jupiter) and Mockito.

End to End and Integration Testing

Definition:

Integration testing will include any testing for a single test case. End to End testing will be any black box testing of the website.

Participants:

The testing team will write all integration tests and system tests.

The team will write backend integration tests using JUnit. End-to-end tests will be automated using Selenium.

User Acceptance Testing

Definition:

User acceptance testing will consist of black-box testing conducted by real users.

Participants:

The testing team will find volunteers to conduct user acceptance testing.

Methodology:

The testing team will find volunteers from other departments in the company to act as end-users.

Automated Regression Testing

Definition:

Regression testing is the process of running existing tests to ensure changes do not change the results of any established working code.

Participants:

The testing team will be responsible for regression testing.

Methodology:

The team will run regression tests as part of maintenance and updates. Developers will never commit code that fails regression tests to a production pipeline or source branch.

HARDWARE REQUIREMENTS

For this project a laptop or computer is needed. This device needs a 64 bit processor with 8+ GB of ram. This device can have any kind of operating system.

ENVIRONMENT REQUIREMENTS

With this device there is certain software that is needed to be able to run these tests. The first piece of software is a java JDK 8. This is used to compile and run the java application from the command line if needed. To view the code use Java Spring Tools 4. To actually run these tests, there are libraries needed. The libraries are Junit, and Selenium.

Here are links on how to acquire these additional software

- Java JDK 8 installer: https://www.oracle.com/java/technologies/downloads/#java8
- Java Spring Tools 4 installer: https://spring.io/tools

- JUnit: https://junit.org/junit5/
- Selenium: https://www.selenium.dev/downloads/

TEST SCHEDULE

TBD

CONTROL PROCEDURES

Problem Reporting

Problem reporting will be an extensive procedure that this testing team will implement. Any time a bug or defect is caught during the testing it will be discussed by the entirety of the testing team, even if caught in a smaller subgroup. This will be done to ensure that no one left out in the testing subgroup of this particular bug or defect does not have a different perspective to approach or fix it. Once the testing team has decided they are incapable of fixing the bug or the defect they will send it back through the software defect life cycle to be handled by the development team.

FEATURES TO BE TESTED

MVP Features

- Login
- Session Management
- Register
- Display Products
- User Profile
- Cart
- Search for Products
- Checkout
- Sales/Deals

Feature: Login

1. Scenario: Logging in successfully
2. Scenario: Invalid username
3. Scenario: Invalid password
4. Scenario: Blank username
5. Scenario: Blank password
6. Scenario: Not logged in, welcome page displays guest
7. Scenario: Logging in as admin

Feature: Register

1. Scenario: Signing up successfully
2. Scenario: Blank username
3. Scenario: Blank password
4. Scenario: Blank email
5. Scenario: Blank Firstname
6. Scenario: Blank Lastname
7. Scenario: Blank age
8. Scenario: Blank address
9. Scenario: Username already taken
10. Scenario: email already taken
11. Scenario: Birthday would put age over 125yrs
12. Scenario: Birthday would put age under 0
13. Scenario: Email is invalid format

Feature: User Profile

1. Scenario: Given Im logged in, accessing profile page, viewing information
2. Scenario: Changing username successfully (stretch goal)
3. Scenario: Attempting to change username, but username is already taken (stretch goal)
4. Scenario: Changing password
5. Scenario: Changing email successfully
6. Scenario: Attempting to change email but email is already taken
7. Scenario: Attempting to change email but new email is invalid format
8. Scenario: Changing first name
9. Scenario: Changing last name
10. Scenario: change age
11. Scenario: Attempting to change birthday but age would be over 125 yrs
12. Scenario: Attempting to change birthday but age would be under 0
13. Scenario: Change address

Feature: Cart

1. Scenario: When logged in, add a product to cart
2. Scenario: Remove an item from my cart
3. Scenario: Remove all items from cart at once
4. Scenario: Attempting to add an item to your cart when you are not logged in
5. Scenario: Attempting to add a product that is out of stock
6. Scenario: Adding an item already in your cart
7. Scenario: Adding multiple quantity of single item to cart

Feature: Search for products (might be better for manual testing for the most part)

1. Scenario: Searching by book id
2. Scenario: Searching by book ISBN (Stretch goal)
3. Scenario: Searching by book genre
4. Scenario: Searching for keyword

Feature: Display Products

1. Scenario: Default Page Shows products
2. Scenario: Second Page Shows products
3. Scenario: Product is Out of Stock

Feature: Checkout

1. Scenario: Going to checkout page, displays items in cart
2. Scenario: Successfully checking out (including changing item stock, giving order confirmation)
3. Scenario: Canceling checkout (back to shopping)
4. Scenario: Payment info is correct
5. Scenario: CC# is invalid
6. Scenario: Expiration date is invalid
7. Scenario: Security number is invalid

Feature: Sales/Deals

1. Scenario: Seeing a product on sale
2. Scenario: Adding a product on sale to cart
3. Scenario: Admin can apply sales to products successfully
4. Scenario: Admin removing items from sale
5. Scenario: Admin attempting to put an item on sale for 100%+ (free)
6. Scenario: Admin is attempting to put an item on "sale" for more than original price

FEATURES NOT TO BE TESTED

1. Scenario: Address is invalid? (Not needed, this is an extensive work if we really want to validate an address - Sorry if i missed this earlier.)
2. Scenario: Attempting to change address but address is invalid (Same comment on Scenario 10 in Register Feature)

Stretch goals

- Featured Products
- Dark Mode
- Quantity Select
- Reset Password
- Bidding/Auctions
- Notifications
- Admin Sales Metrics Dashboard
- Tech Support Chat
- Flash Deals

Feature: Featured Products

1. Scenario: Seeing a list of featured products on the main page
2. Scenario: Admin adding featured items to the featured items list

3. Scenario: Admin removing items from featured items list

Feature: Dark Mode
1. Scenario: Clicking on dark mode button

Feature: Quantity Select
1. Scenario: Selecting a quantity of a product
2. Scenario: Adding an item of multiple quantity to cart
3. Scenario: Checking out with an item of multiple quantity
4. Scenario: Removing an item of multiple quantity from cart
5. Scenario: Changing quantity of item in cart
6. Scenario: Selecting 0 or less as quantity

Feature: Resetting password
1. Scenario: Resetting password successfully (I assume we would just ask them username and email or something, maybe some kind of security question)
2. Scenario: Trying to reset password, but don't get security check correct

Feature: Bidding/Auctions
1. Scenario: Listing an item for auction
2. Scenario: Attempting to list an item while not logged in
3. Scenario: Bidding on an auction
4. Scenario: Time runs out on an auction, item should be "sold" to winner

Feature: Notifications
1. Scenario: User has checked out/bought products, should receive some kind of message (either in app or through email) containing order details
2. Scenario: User has won an auction that they've bid on
3. Scenario: User has lost an auction that they've bid on
4. Scenario: Auction for item user has listed has concluded

Feature: Flash Deals
1. Scenario: Items on sale that meet a certain criteria are labeled as "flash sale" items

Feature: Tech Support Chat (leave for now)

Feature: Admin Sales Metrics Dashboard (leave for now)

RESOURCES/ROLES & RESPONSIBILITIES

Everyone on the team is responsible for setting up their own test environment and the test cases have been split up to all members of the team.

SCHEDULES

Deliverables

What we are delivering is the test plan and test case document and traceability matrix

DEPENDENCIES

What are some of the constraints or limitations on testing?
We are limited to the time of the project.

RISKS/ASSUMPTIONS

Possible risk of overlap of tests between features.
Mitigated by proper communication between testing team members.

TOOLS

Eclipse IDE, JUnit, Mockito, Cucumber Selenium, Angular, VS Code, and Postman

APPROVALS

Project Manager:_____