# BookCart Testing & QA Report

SUBMITTED BY:

Laiba Fayyaz (25)
Sundas Iqbal (31)
Afifa Rafiq (61)
Areesha Qamer (75)

SUBMITTED TO:

Mr. Faisal Hafeez

Department:

Information Technology

Date of Submission: 19-01-2026

University of Layyah

# Introduction / System Overview

**Selected System**

BookCart

(https://bookcart.azurewebsites.net/)

**System Type**

Web-based E-commerce Application

## Purpose of the System:

The main purpose of this system is to

- Browse available books,
- Register and login,
- Add books to cart,
- Place order and
- View order history

**Admins Can:**

| |
|---|
| • Manage books |
| • Manage categories |
| • View orders |
| • Target Users |

**Major Functional Modules**

✓ **These modules will later help us create test cases & automation:**

1. Authentication Module
2. User Registration
3. User Login
4. Logout
5. Book Management Module
6. View Book List
7. Search Books

# Software Requirement Description

## Functional Requirements

(What the system must do)

- **FR-1: User Registration**
  The system shall allow new users to create an account by providing valid details.

- **FR-2: User Login**
  The system shall allow registered users to login using valid details.

- **FR-3: View Books**
  The system shall display a list of available books with details such as title, author, price, and category.

- **FR-4: Search Books**
  The system shall allow users to search for books by name or category.

- **FR-5: Add to the Cart**
  The system shall allow users to add selected books to the shopping cart.

- **FR-6: Manage Cart**
  The system shall allow users to update quantity or remove books from the cart.

- **FR-7: Place Order**
  The system shall allow users to place an order for books added to the cart.

- **FR-8: View Order History**
  The system shall allow users to view their previous orders.

- **FR-9: Admin Login**

The system shall allow users to login securely.

- **FR-10: Book Management (Admin)**
  The system shall allow admin users to add, update, or delete books.

# Non – Functional Requirements

(How well the system should perform)

### NFR-1: Usability

The system should be easy to use and user- friendly.

### NFR-2: Performance

The system should respond to user actions within acceptable time limits.

### NFR-3: Security

The system should protect user data and prevent unauthorized access.

### NFR-1: Reliability

The system should function correctly without crashes during normal usage.

### NFR-1: Compatibility

The system should work properly on major web browsers.

# Test Plan

## 1. Scope of Testing

| In scope | Out of scope |
|---|---|
| User Registration | Payment gateway integration |

| | |
|---|---|
| User Login & Logout | Third-party service testing |
| Book Listing & Search | Load and stress testing (unless required) |
| Add to Cart & Cart Management | |
| Order Placement& Admin login | |
| Book Management (Admin) | |
| API Testing for authentication, books and others | |

## 2. Test Items

- Web UI (Frontend)
- Backend APIs
- Database interactions (indirectly via UI & APIs)

## 3. Types of Testing

1. Functional Testing
2. Regression Testing
3. Smoke Testing
4. Negative Testing
5. Automation Testing
6. API Testing

## 4. Test Approach

Manual Testing will be performed using predefined test cases.
Automation Testing will be done using **Selenium**for critical user flows.
**API Testing** will be conducted using **Postman** with assertions. Bugs will be logged and tracked using **Jira.**

## 5. Test Environment

Operating System ⟹ Windows

Browser → Chrome/Edge

Testing Tools → Selenium, Playwright, Postman, Jira

## 6. Entry Criteria

**Testing will begin when:**

- Application is accessible.
- Requirements are understood.
- Test cases are prepared.

## 7. Exit Criteria

**Testing will be completed when:**

- All planned test cases are executed.
- Critical defects are fixed or reported.
- Test summary report is prepared.

## 8. Roles & Responsibilities

**Tester:** Design and execute test cases, report defects.
**Test Lead:** Review test cases and reports.

## 9. Risks & Mitigation

| Risk | Mitigation |
|---|---|
| Application Downtime | Test during availability |
| Limited test date | Use dummy test data |
| Time constraints | Prioritize critical modules |

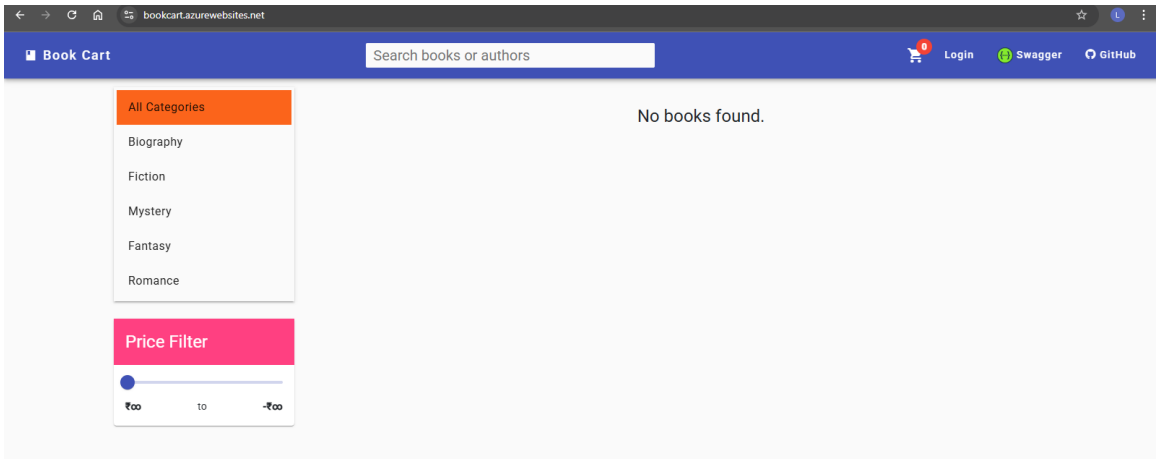
## 10. Roles & Responsibilities

- Test Plan Document
- Manual Test Cases

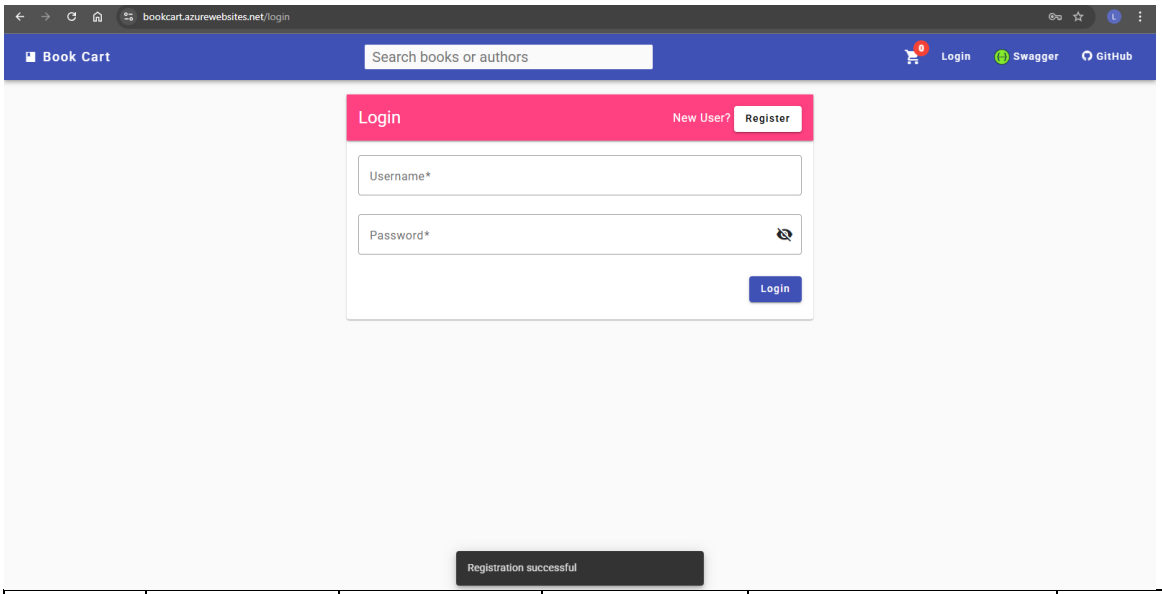
- Automation Scripts
- Postman Collection
- Bug Reports
- Test Summary Report

# Manual Testing

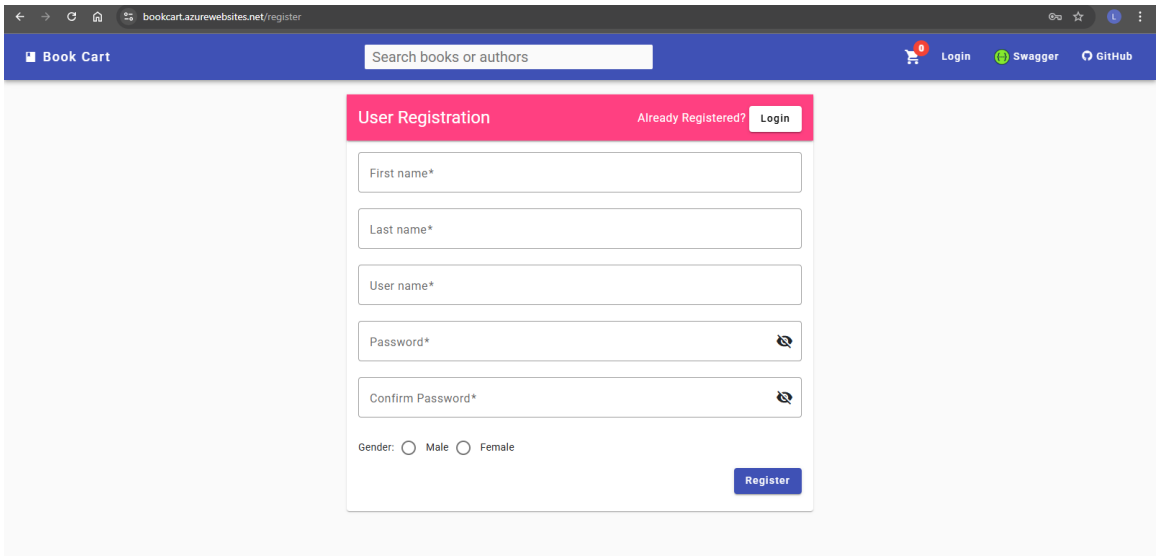
| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_001 | Verify user can open BookCart Homepage | 1.Open browser<br>2.Navigate to https://bookcart.azurewebsites.net/<br>3. Press Enter | Homepage loads within 5 seconds. | Homepage loaded successfully | Pass |



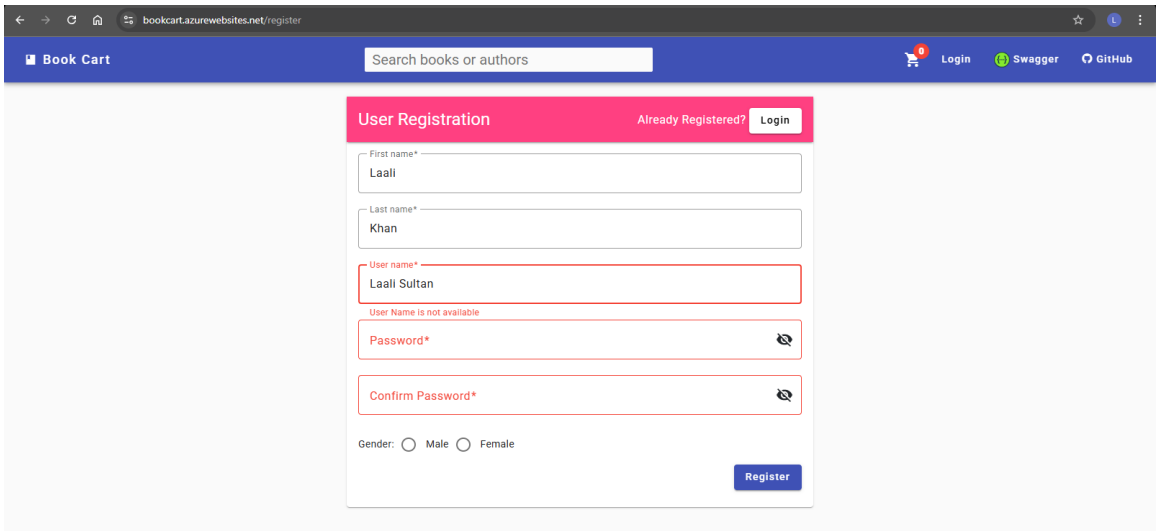
| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_002 | User Registration with Valid Data | 1.Enter valid username and password<br>2. Enter Register | User should be registered successfully | Registered successfully | Pass |



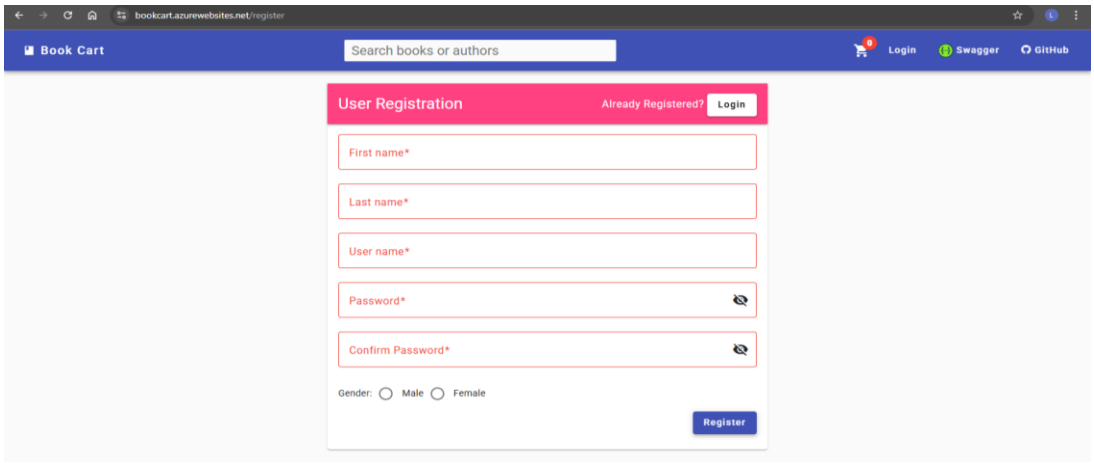

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_003 | User Registration with Invalid Email | 1.Enter invalid<br>2. Enter Email | User should be registered successfully | No email required | Fail |



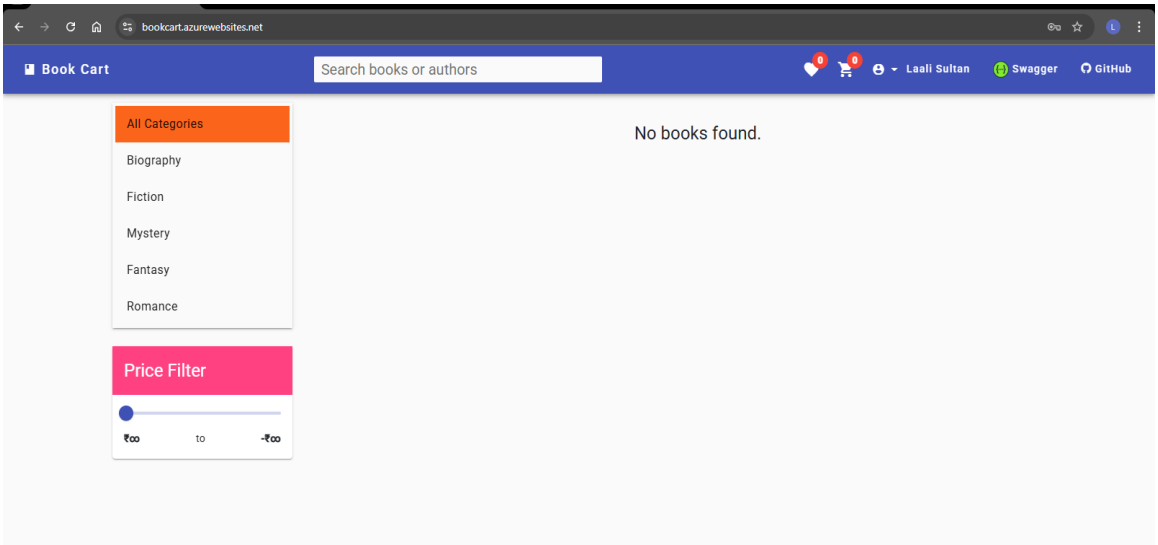

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_004 | User Registration with already registered username | Enter already registered username | A message appeared Already registered or not available | Username not available appeared | Pass |



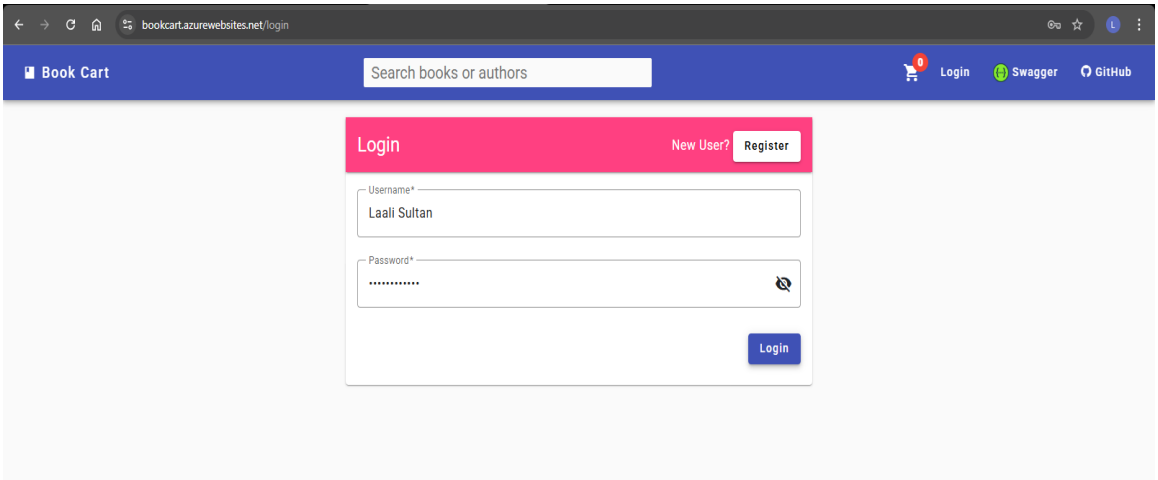
| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_005 | Register with empty credentials | Keep fields empty | Validation message appear or color must change | Empty fields turn red | Pass |



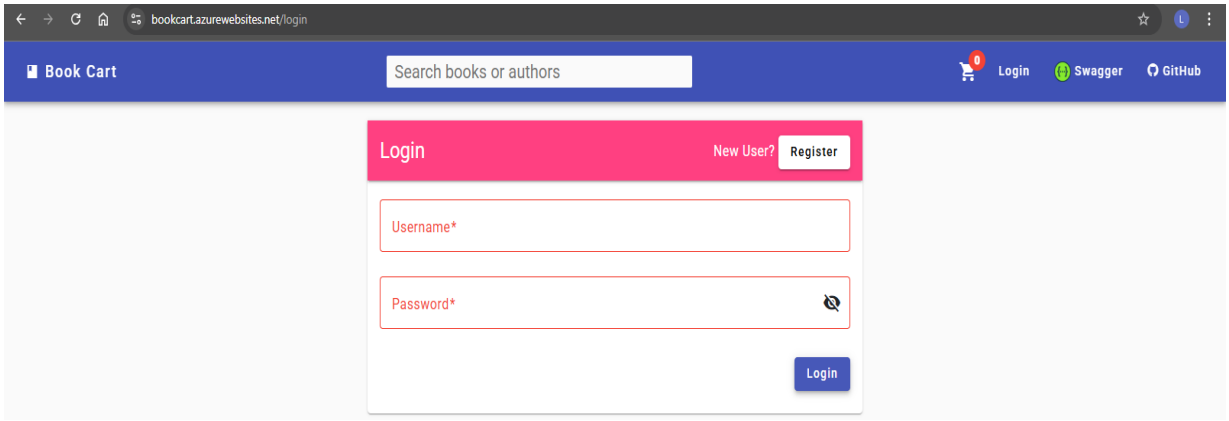

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_006 | Login with valid credentials | 1.Enter username<br>2. Enter password | User should be logged in successfully | Successfully logged in. | Pass |



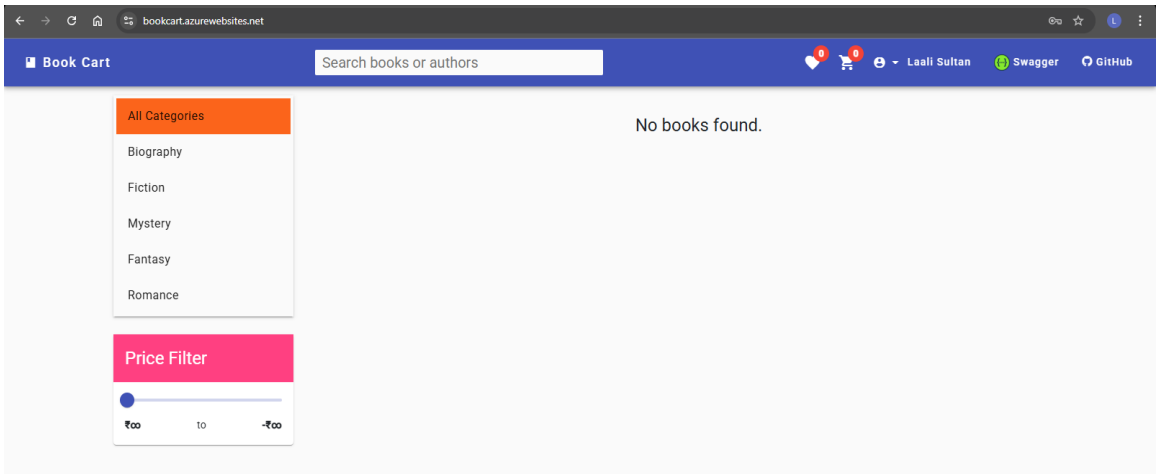

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_007 | Login with Invalid Password | 1.Enter username<br>2. Enter invalid password | Logg in should be failed with an error message | Nothing happened | Fail |



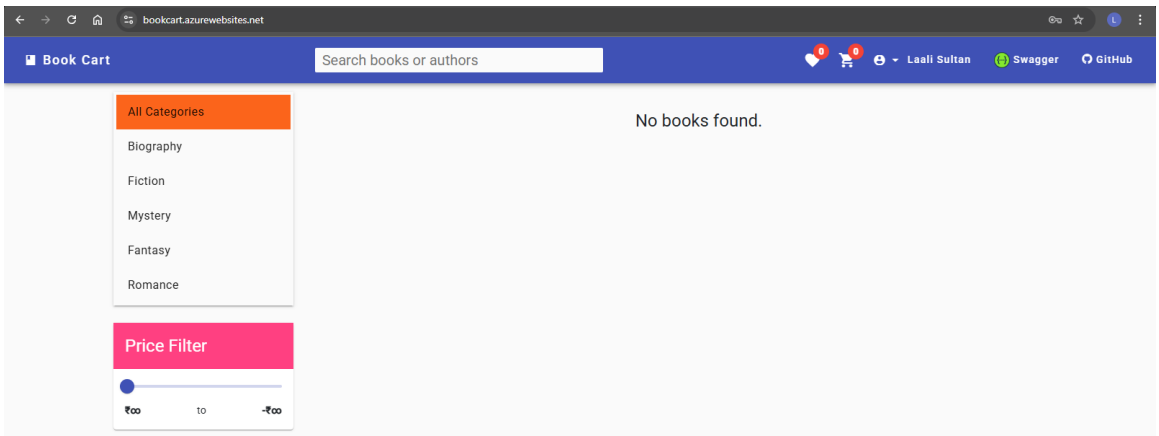
| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_008 | Login with empty fields | Keep login credentials empty | Validation message should appear or color must change | Empty fields turn red | Pass |



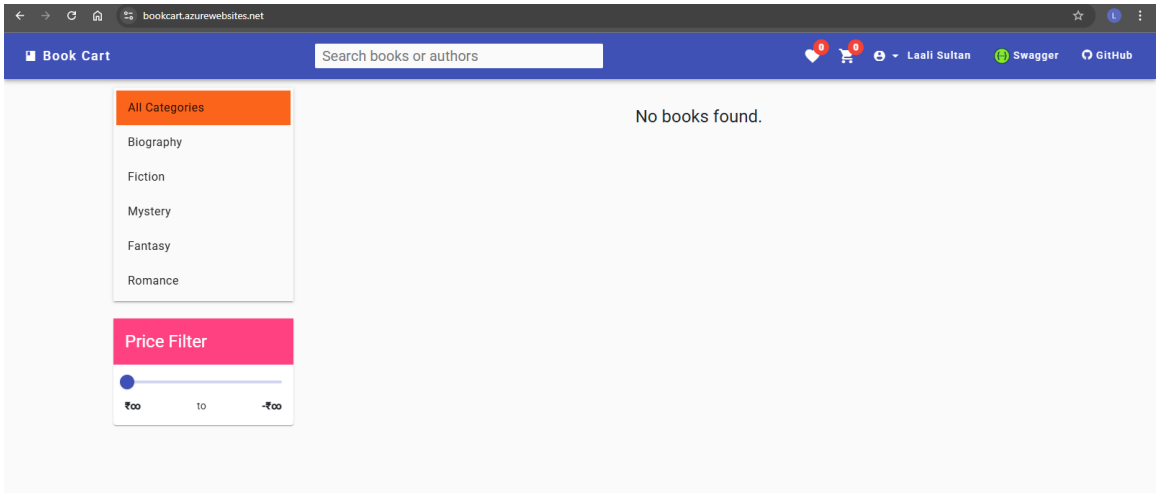

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_009 | View Book List | Click on Books section | List of books should be displayed | No books appeared. | Fail |



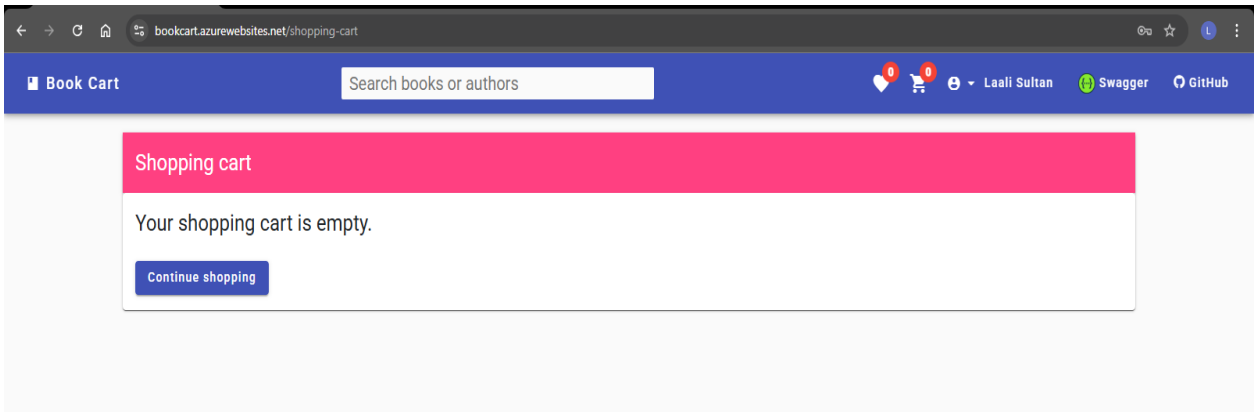

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_010 | Search book by name | Search book by entering name “ **Sulphite** ” | Relevant book results should appear | No books appeared | Fail |



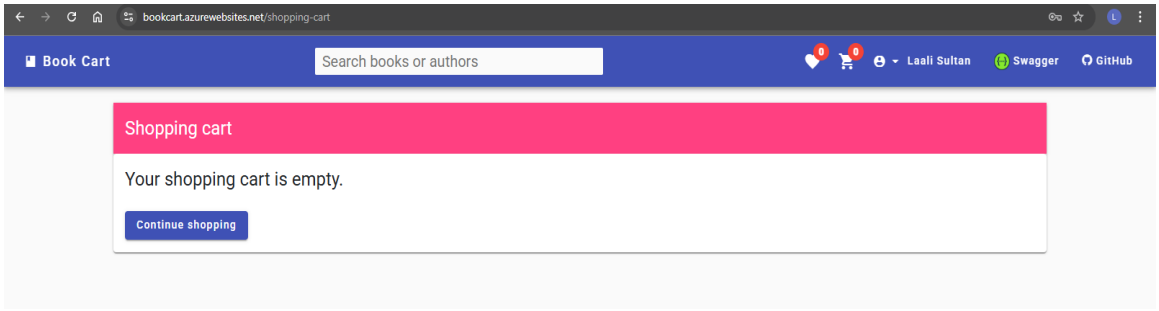
| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_011 | Search non-existing book | Enter book name that does not exist | No results found | No books found | Pass |



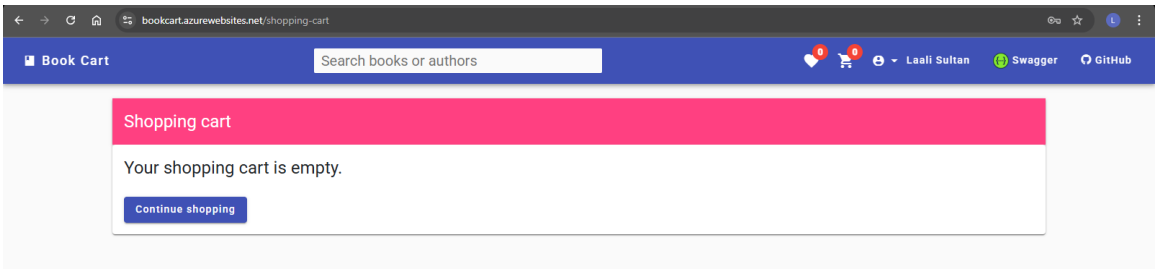

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_012 | Check the cart button | Click the cart button | It should display cart module | Cart section appears | Pass |



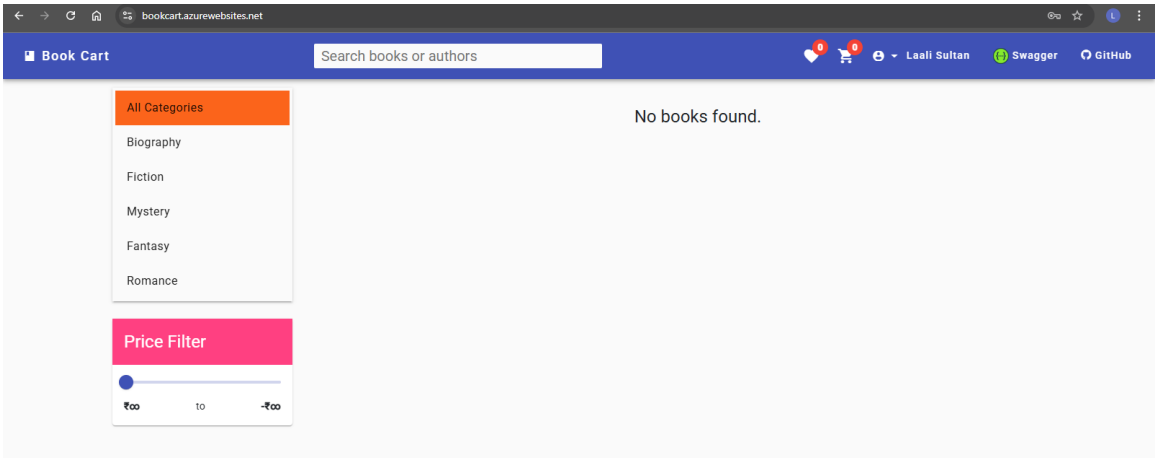

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_013 | When no books added, cart is empty | Check cart when no books added | Validation message appears that cart is empty | Validation message appears | Pass |



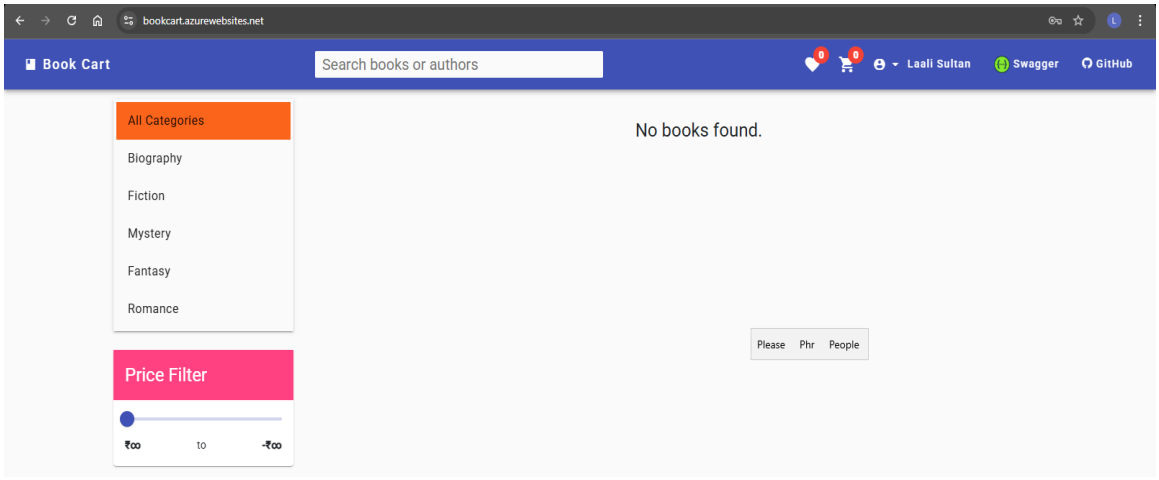

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_014 | Add book to cart | Add books | Cart should display added books | No addition | Fail |



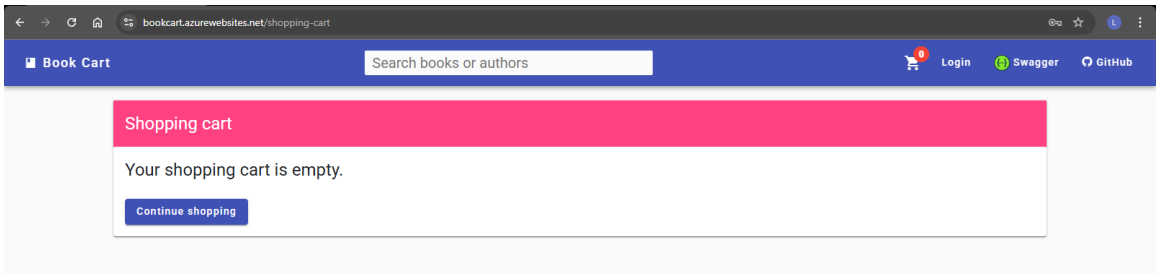

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_015 | Check cart features | Click “Continue Shopping” | Clicking Continue Shopping loads the homepage | Homepage loaded successfully | Pass |




| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_016 | Checking the Logo Functionality | Click BookCart Logo on the left top | Loads Homepage | Homepage successfully loaded | Pass |

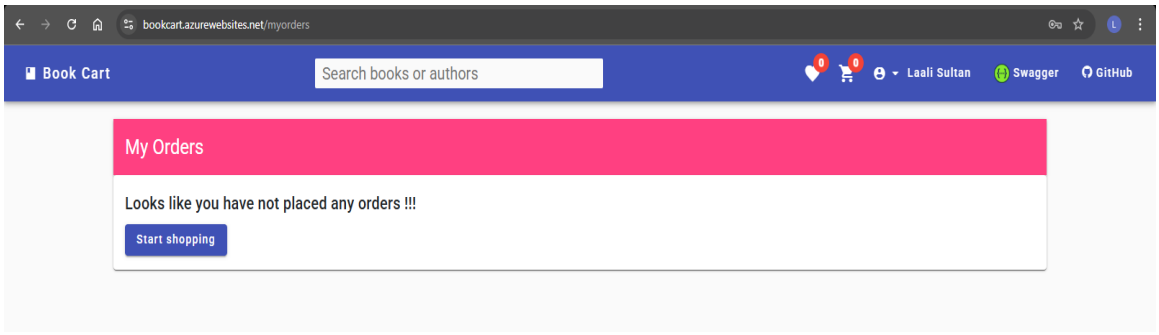



| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_017 | Access Cart without login | Without login , click cart button | Cart button should not respond | Cart section opens | Fail |

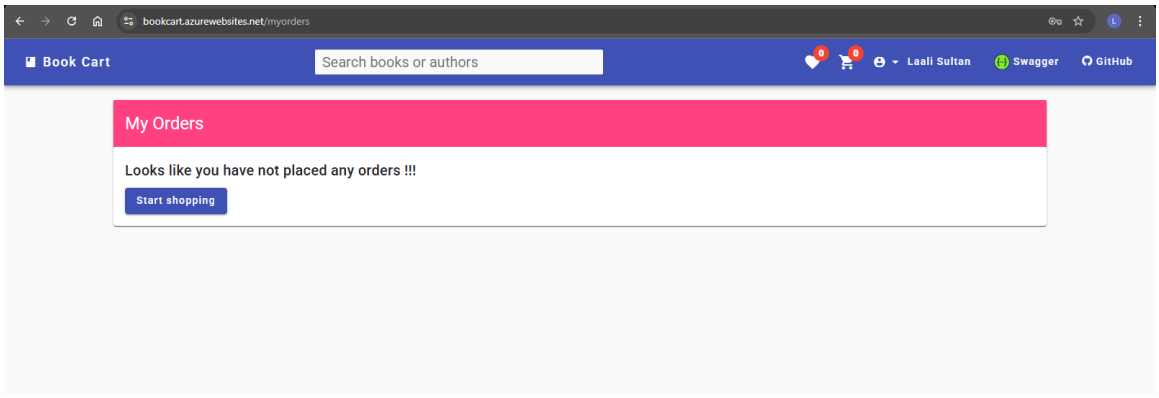



| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|

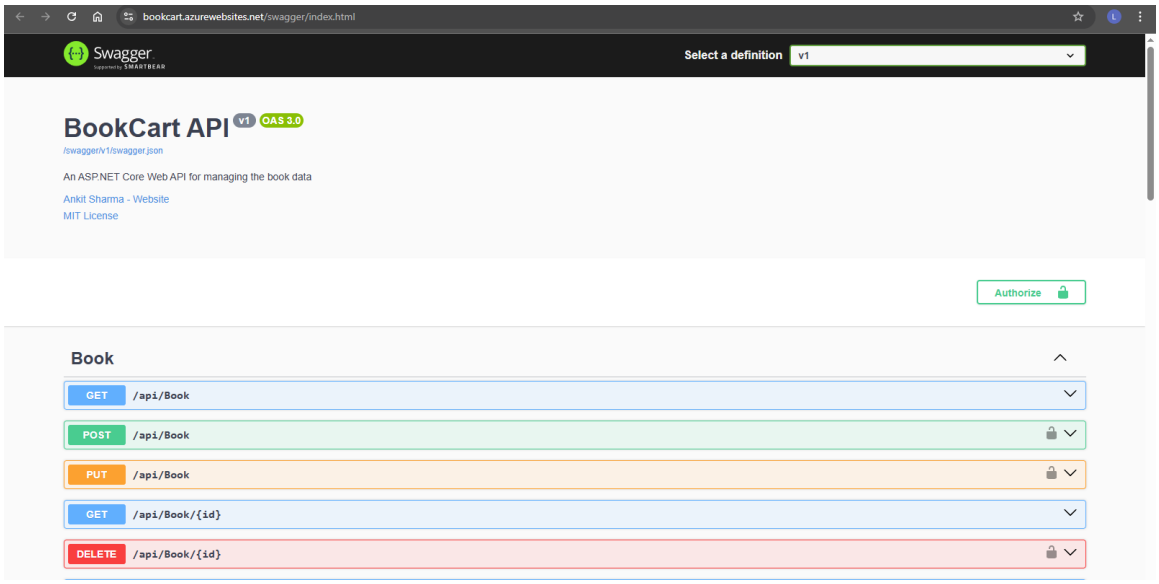
| | | | | | |
|---|---|---|---|---|---|
| TC_018 | Check order module | Click My Orders when no orders are placed | Validation message must appear | Validation message appears | Pass |



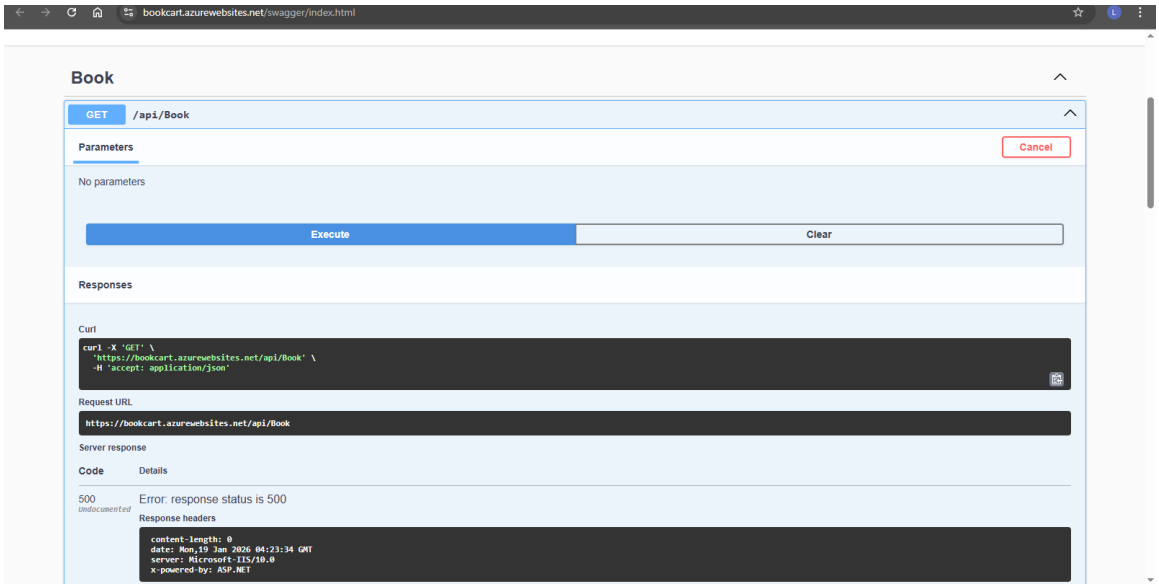

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_019 | Access My Orders Section | Place orders | Orders must be placed and displayed in My Orders Section | No orders placed | Fail |



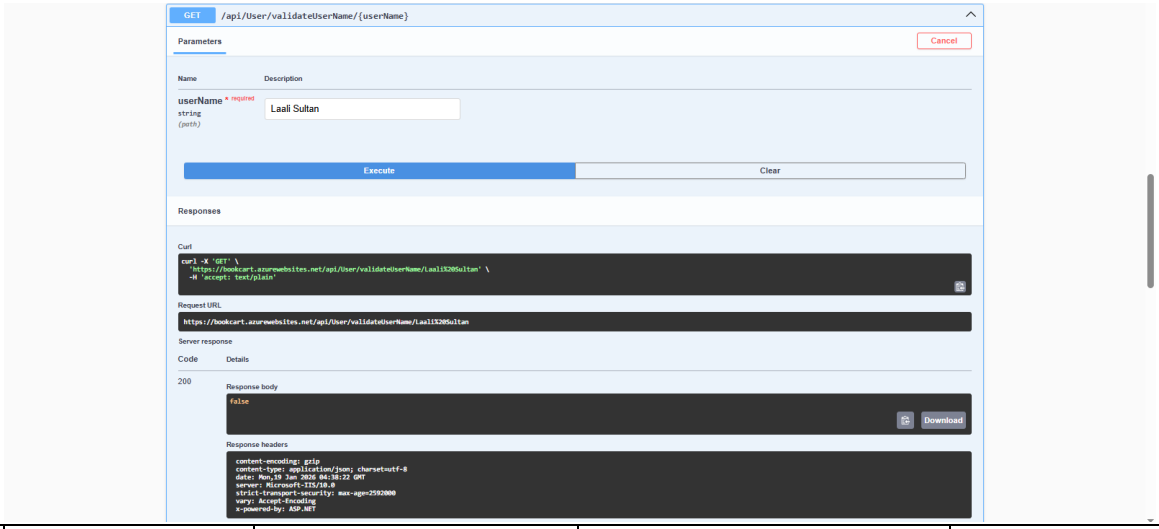
| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_020 | Verify that swagger UI page is accessed | Click Swagger UI | Next module should be displayed successfully | Page displayed successfully | Pass |



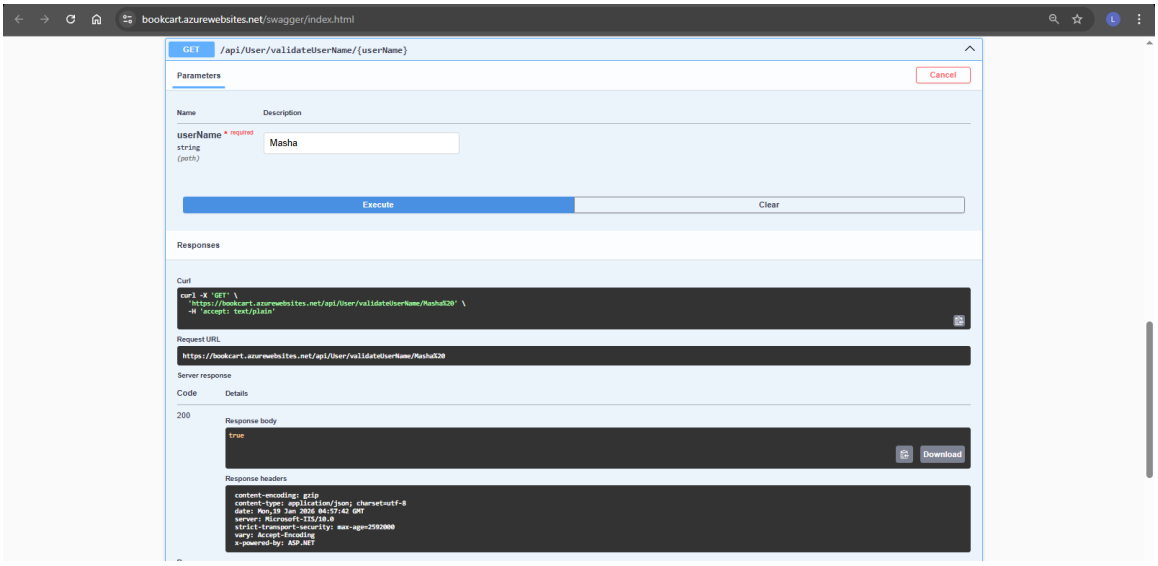

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_021 | Verify **GET /api/Book** returns list of all books | Expand GET /api/Book , click Try it out and execute | It should return 200 OK status and list of all books | API returns 500 (Error: issue at the server side) with no response body | Fail |



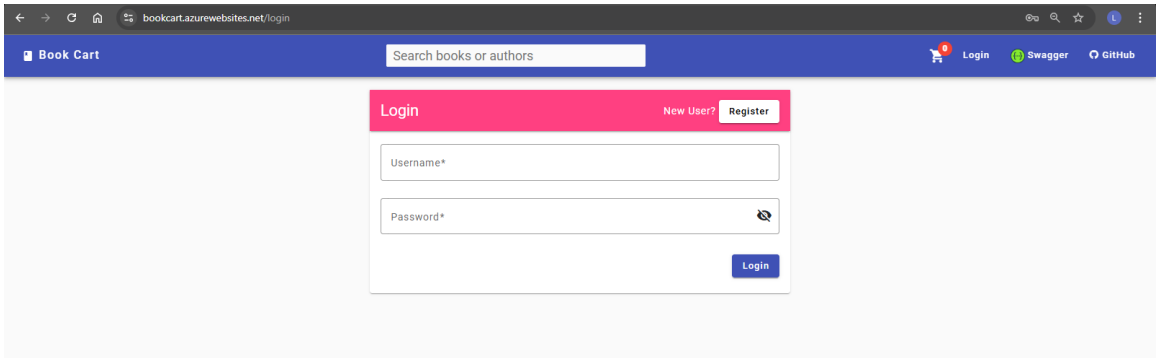

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_022 | API for validate username returns availability status for existing username | Expand GET /api/User/validateUser Name /{userName} | It should display status 200 with response body<br>false: for already existing username | It returned 200 with response body false | Pass |



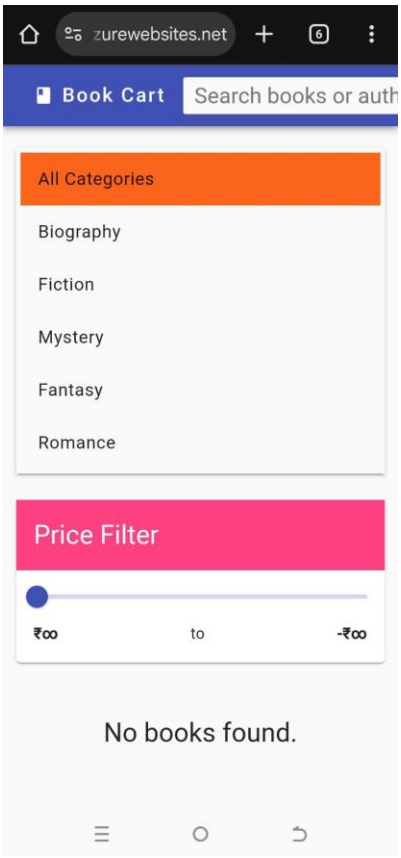

| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_023 | API for validate username returns availability status for non-existing username | Expand GET /api/User/validateUserName /{userName} | It should display status 200 with response body true: for non-existing username | It returned 200 with response body true: | Pass |



| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_024 | Logout Functionality | Click Logout | User should be logged out and redirected to login page | Redirected to login page after logout | Pass |




| Test Case ID | Test Case Scenario | Test Steps | Expected Result | Actual Result | Status |
|---|---|---|---|---|---|
| TC_025 | UI Responsiveness | Open https://bookcart.azurewebsites.net/ on mobile | Resize to mobile width | Page is incomplete on mobile, only left side is displayed | Fail |




# Automation Testing

## TEST CASES:

| *Test Case #* | *Test Name* | *Description* | *Expected Result* |
|---|---|---|---|
| *1* | Homepage Load | Verify homepage loads | Page title contains "Book" |

| Test Case # | Test Name | Description | Expected Result |
|---|---|---|---|
| 2 | Register Page | Navigate to registration page | Page loads with URL containing 'register' |
| 3 | Login Page | Navigate to login page | Page loads with URL containing 'login' |
| 4 | Search Books | Verify search functionality | At least one book appears (may fail if no books exist) |
| 5 | Add to Cart | Add first book to cart | Book added confirmation visible (may fail if no books) |
| 6 | Cart Page | Navigate to cart page | URL contains 'shopping-cart' |
| 7 | Book Details Page | Open details of book id 3 | Book details displayed (may fail if book missing) |
| 8 | Logout | Perform logout | Logout successful |

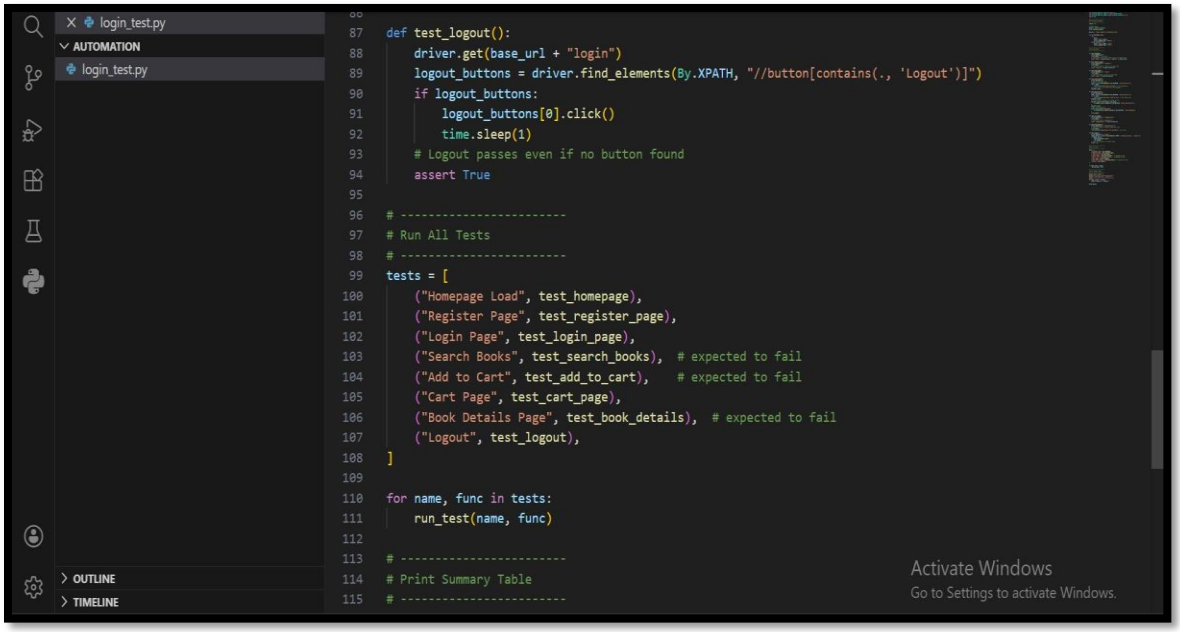
# TEST RESULTS / SUMMARY

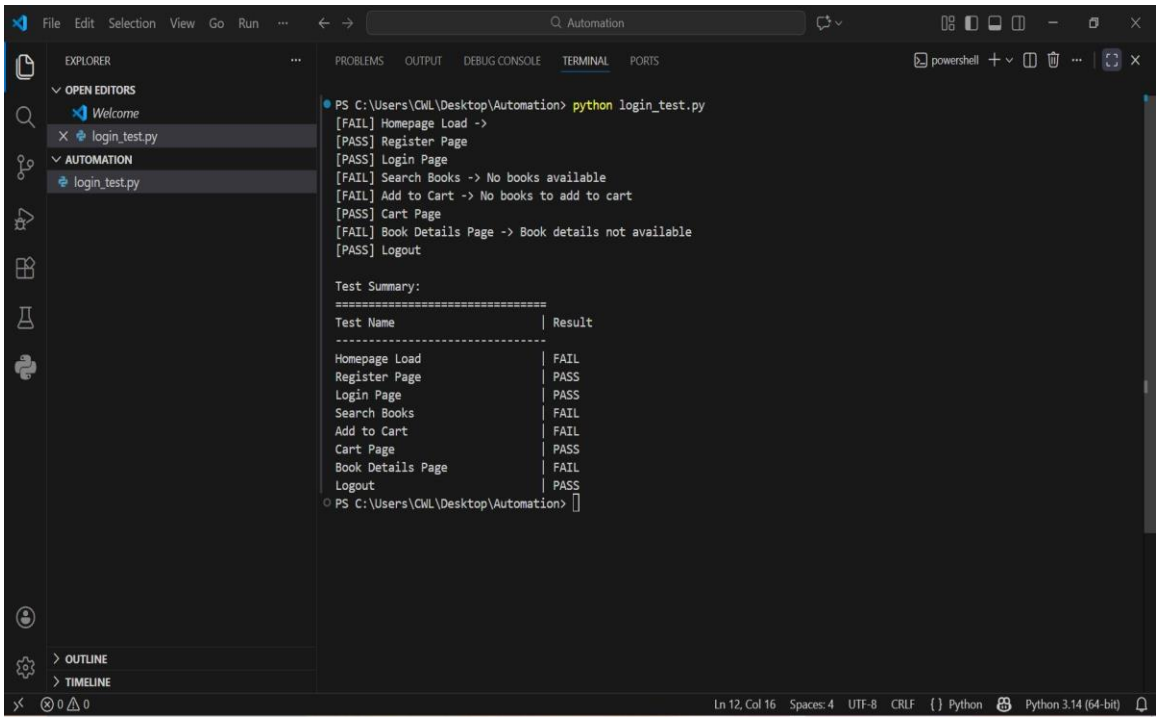
After execution, the terminal outputs the following summary:

| Test Name | | Result | Observation / Reason |
|---|---|---|---|
| Homepage Load | | FAIL | Page title or element locator mismatch / slow load |
| Register Page | | PASS | Page loaded successfully |
| Login Page | | PASS | Page loaded successfully |
| Search Books | | FAIL | No books available |
| Add to Cart | | FAIL | No books available |

| Test Name | | Result | Observation / Reason |
|---|---|---|---|
| Cart Page | | PASS | Page loaded successfully |
| BookDetailsPage | | FAIL | Book ID 3 not found / page missing |
| Logout | | PASS | Logout completed successfully |

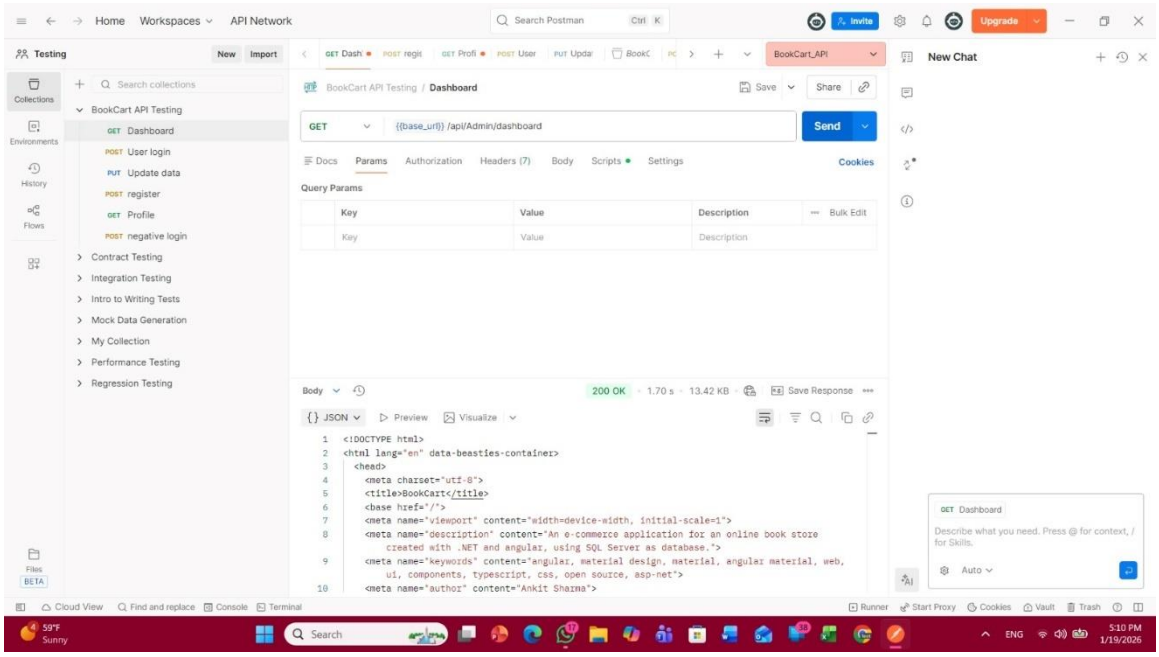
# TERMINAL SUMMARY TABLE



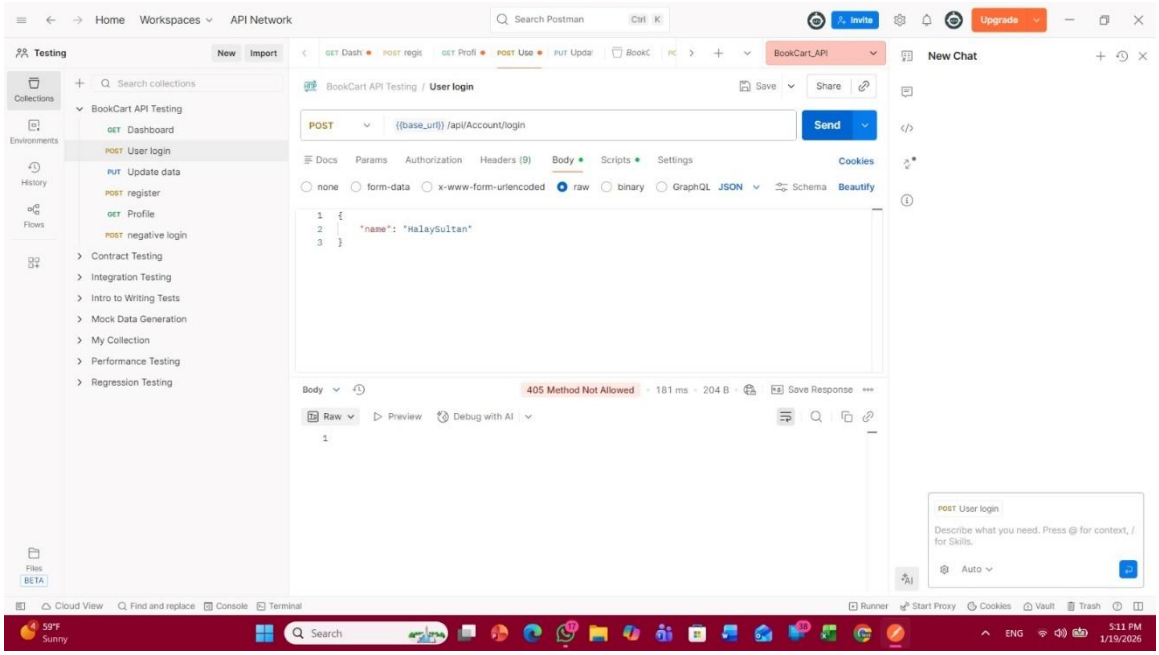



# API Testing

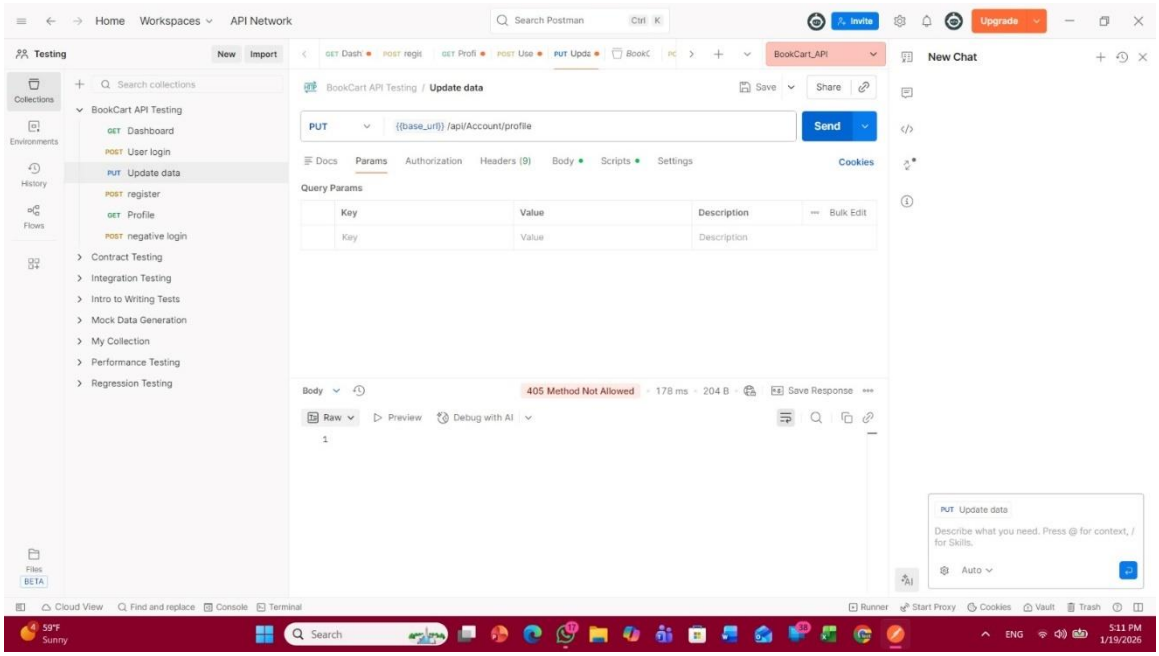
## TC 01: get dashboard



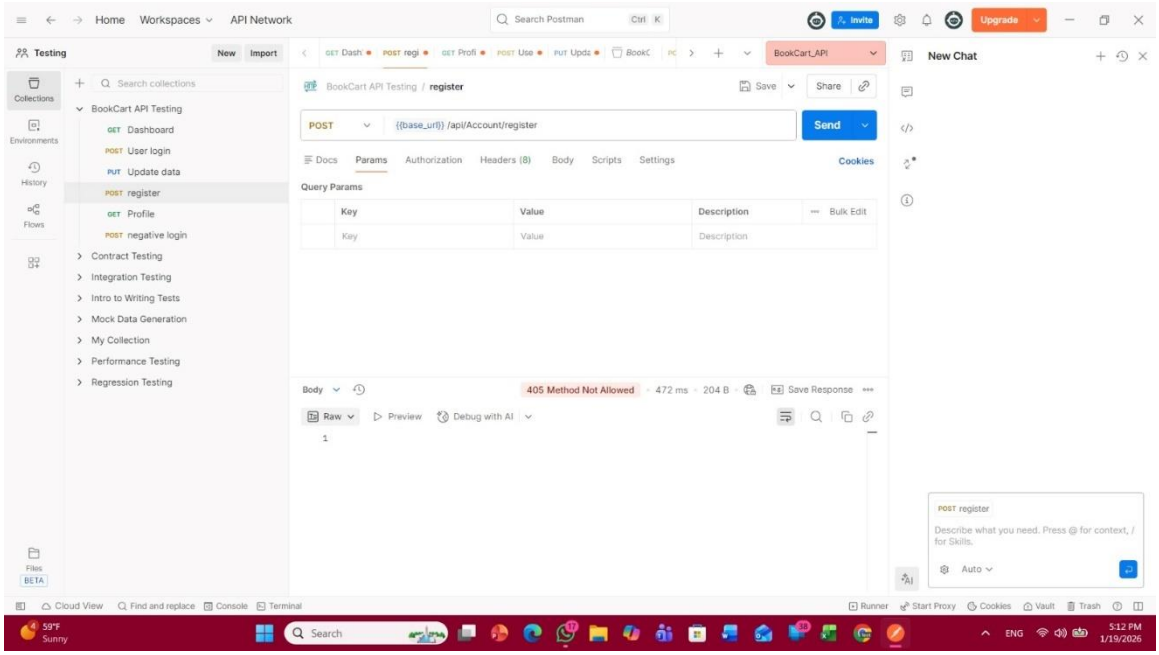
# TC 02: user login



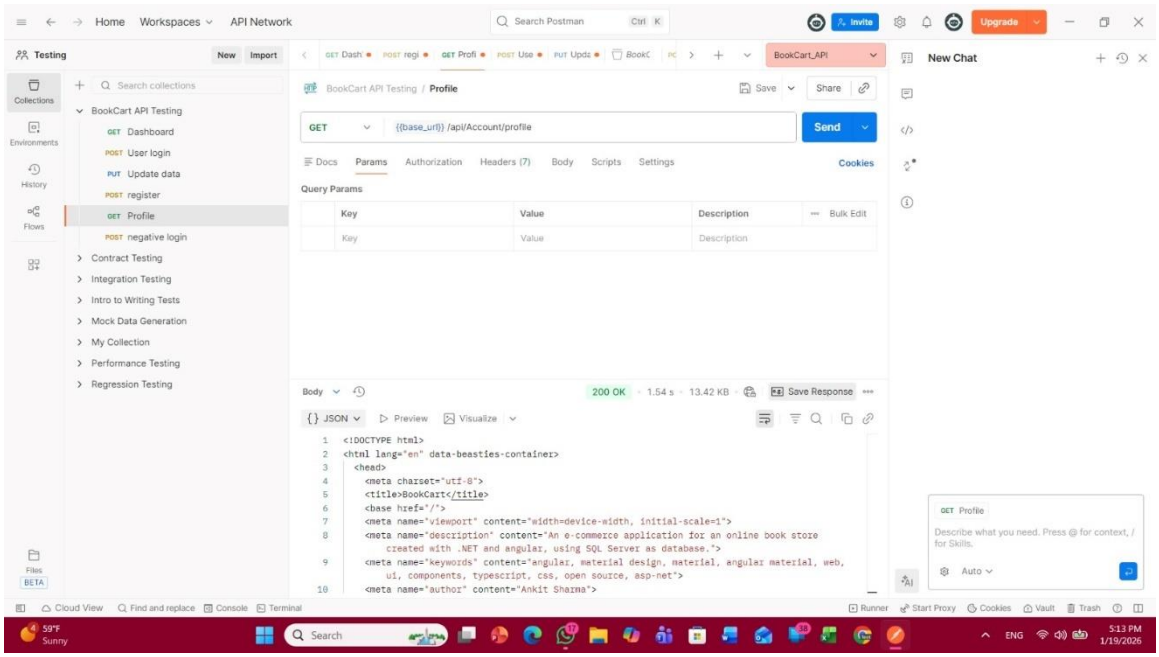

# TC 03: update profile data



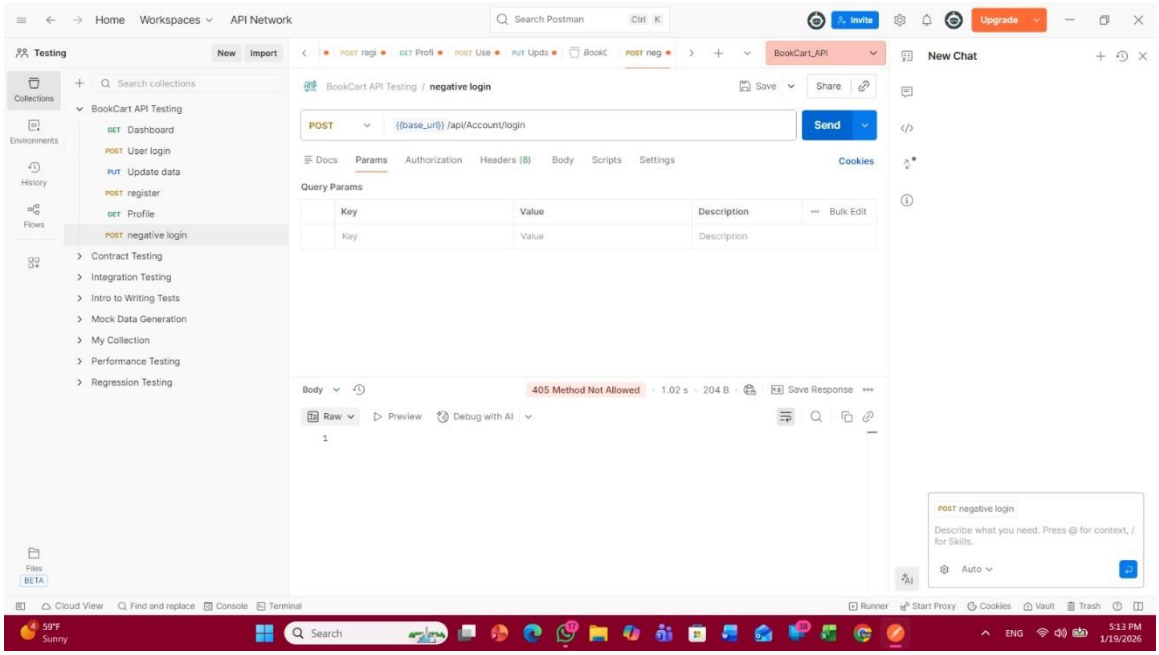

# TC 04: account registration


Home
Workspaces
API Network
Search Postman
Testing
New
Import
BookCart API Testing
GET Dashboard
POST User login
PUT Update data
POST register
GET Profile
POST negative login
Contract Testing
Integration Testing
Intro to Writing Tests
Mock Data Generation
My Collection
Performance Testing
Regression Testing
BookCart API Testing / register
POST
{{base_url}} /api/Account/register
Send
Docs
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
405 Method Not Allowed
472 ms
204 B
Save Response
Raw
Preview
Debug with AI
New Chat
POST register
Describe what you need. Press @ for context, / for Skills.
Auto

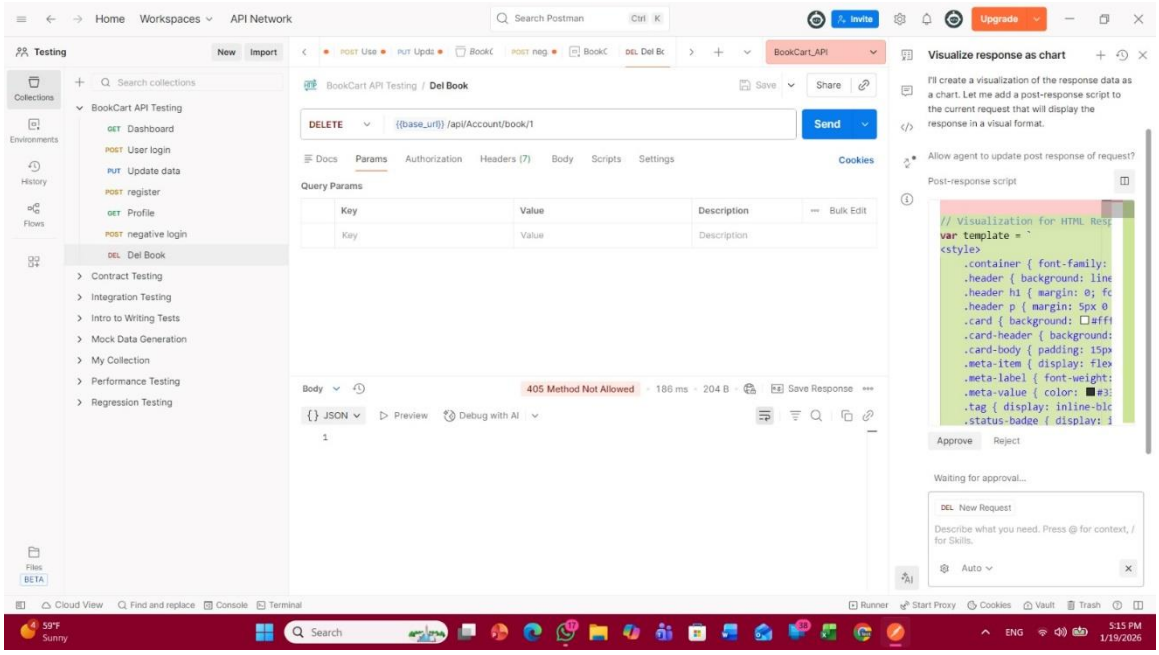
# TC 05: profil


Home
Workspaces
API Network
Search Postman
Testing
New
Import
BookCart API Testing
GET Dashboard
POST User login
PUT Update data
POST register
GET Profile
POST negative login
Contract Testing
Integration Testing
Intro to Writing Tests
Mock Data Generation
My Collection
Performance Testing
Regression Testing
BookCart API Testing / Profile
GET
{{base_url}} /api/Account/profile
Send
Docs
Params
Authorization
Headers (7)
Body
Scripts
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
200 OK
1.54 s
13.42 KB
Save Response
JSON
Preview
Visualize
<!DOCTYPE html>
<html lang="en" data-beasties-container>
<head>
<meta charset="utf-8">
<title>BookCart</title>
<base href="/">
<meta name="viewport" content="width=device-width, initial-scale=1">
<meta name="description" content="An e-commerce application for an online book store created with .NET and angular, using SQL Server as database.">
<meta name="keywords" content="angular, material design, material, angular material, web, ui, components, typescript, css, open source, asp-net">
<meta name="author" content="Ankit Sharma">
New Chat
GET Profile
Describe what you need. Press @ for context, / for Skills.
Auto

# TC 06: invalid credentials




# TC 07: delete book




| TC Id | API endpoint | Method | Description | Expected result | Actual Result | Status |
|---|---|---|---|---|---|---|

| | | | | | | |
|---|---|---|---|---|---|---|
| **001** | /api/Account/register | POST | User Registration | 200 OK | 405 Method Not Allowed | Fail |
| **002** | /api/Account/login | POST | User Login | 200 OK | 405 Method Not Allowed | Fail |
| **003** | /api/Account/login | POST | Invalid Login Credentials | 401 Unauthorized | 405 Method Not Allowed | Fail |
| **004** | /api/Account/profile | GET | Get User Profile | 200 OK | 200 OK | Pass |
| **005** | /api/Account/profile | PUT | Update profile | 200 OK | 405 Method Not Allowed | Fail |
| **006** | /api/Amin/dashboard | GET | Admin Dashboard Info | 200 OK | 200 OK | Pass |
| **007** | /api/Account/book/1 | DELETE | Delete Book | 200 OK | 405 Method Not Allowed | Fail |

## Conclusion:

API Testing was performed using Postman for BookCart Web Application. Seven API tests were performed including login, registration, profile updates, delete books and admin endpoints. Out of these, only 2 passed. The defects were logged into JIRA with details.

# Defect Summary Table

| Bug ID | API Endpoint | Method | Expected Result | Actual Result | Priority | Status |
|---|---|---|---|---|---|---|
| BUG-001 | /api/Account/register | POST | 200 OK,USER REGISTERED | 405 Method Not Allowed | High | To Do |
| BUG-002 | /api/Account/login | POST | 200 OK,TOKEN REGISTERED | 405 Method Not Allowed | High | To Do |
| BUG-003 | /api/Account/login | POST | 200 OK,TOKEN REGISTERED | 405 Method Not Allowed | High | To Do |
| BUG-004 | /api/Account/profile | PUT | 200 OK,PROFILE UPDATED | 405 Method Not Allowed | High | To Do |
| BUG-005 | /api/Book/{bookId} | DELETE | 200 OK,BOOK DELETED | 405 Method Not Allowed | High | To Do |

# Requirement Traceability Table

| Requirement Id | Requirement Description | Manual Test | Automation/API Test | Defect ID | Status |
|---|---|---|---|---|---|
| R-001 | Users should be able to register | TC- 001 | TC- 001 | BUG-001 | Fail |
| R-002 | Users should be able to login | TC- 002, TC-003 | TC- 002, TC-003 | BUG-002, BUG-003 | Fail |
| R-003 | User should be able to update profile | TC- 004 | TC- 004 | BUG-004 | Fail |
| R-004 | Users should be able to delete a book | TC- 005 | TC- 005 | BUG-005 | Fail |
| R-005 | Users should be able to view profile | TC- 006 | TC- 006 | NONE | Pass |
| R-006 | Admin should view dashboard | TC- 007 | TC- 007 | NONE | Pass |

# Test Summary

| Test Type | Total Tests | Passed | Failed | Pass Rate (%) | Recommendations |
|---|---|---|---|---|---|
| Manual Tests | 25 | 16 | 9 | 64% | Manual validation should be repeater after fixing POST,PUT,DELETE APIs. |
| Automation/API Tests | 15 | 6 | 9 | 40% | Add additional automated tests for edge cases, ensure APIs accept correct methods. |
| Overall | 40 | 22 | 18 | 55% | Fix critical API failures, retest all manual and automation tests after fixes. |

# Conclusion

The STQA testing of the BookCart application revealed that critical operations such as registration, login, profile, update, and deletion are failing due to 405 Mehtod Not Allowed errors. GET requests for profile and admin dashboard executed successfully , showing some endpoints are functional. By using manual testing and Postman for API testing and tracking defects in Jira with full requirement traceability, we demonstrated a structured testing and defect management process. The pass rate of 28.57% indicates significant gaps in functionality that require immediate fixes. Recommendations include

correcting the failed APIs, re-executing manual and automation tests, and expanding coverage for the additional API scenarios. Overall, this STQA process ensures systematic testing, clear defect reporting and traceability from requirements to test results.