1 FAILED 52 PASSED

98% passed
53 executed

8 minutes ago
last run

8 minutes 3 seconds
duration

Mac OS X

Java HotSpot(TM) 64-Bit Server VM 17.0.9+11-LTS-201

cucumber-jvm 7.3.2

Search with text or @tags
You can search with plain text or Cucumber Tag Expressions to filter the output

☑ failed ☑ passed

✓ file:///Users/umute/IdeaProjects/LibraryApp-DB-Automation/src/test/resources/features/US_00_books.feature

@smoke

Feature: Book Category

Scenario: verify book categories with UI

- ✓ **Given** the user logged in as "librarian"
- ✓ **When** the user navigates to "Books" page
- ✓ **And** the user gets all book categories in webpage
- ✓ **Then** user should be able to see following categories

Action and Adventure
Anthology
Classic
Comic and Graphic Novel
Crime and Detective
Drama
Fable
Fairy Tale
Fan-Fiction
Fantasy
Historical Fiction
Horror

Science Fiction
Biography/Autobiography
Humor
Romance
Short Story
Essay
Memoir
Poetry

@db

Scenario: verify book categories with DB

- ✓ **Given** the user logged in as "librarian"
- ✓ **When** the user navigates to "Books" page
- ✓ **And** the user gets all book categories in webpage
- ✓ **Then** verify book categories must match book categories table from db

@db

Scenario: Verify book information with db

- ✓ **Given** the user logged in as "librarian"
- ✓ **And** the user navigates to "Books" page
- ✓ **When** I open book "Agile Testing"
- ✓ **Then** book information must match the database for "Agile Testing"

✓ file:///Users/umute/IdeaProjects/LibraryApp-DB-Automation/src/test/resources/features/US_01_dashboard.feature

Feature: Dashboard Page

@db

Scenario: Dashboard data verification

- ✓ **Given** the user logged in as "librarian"
- ✓ **When** user gets all information from modules
- ✓ **Then** the information should be same with database

✗ file:///Users/umute/IdeaProjects/LibraryApp-DB-Automation/src/test/resources/features/US_02_category.feature

Feature: As a data consumer, I want to know category of books are being borrowed the most

@db

Scenario: verify the the most popular book category that’s being borrowed

- ✓ **Given** Establish the database connection
- ✓ **When** I execute query to find the most popular book category
- ✓ **Then** verify "Fantasy" is the most popular book category.

@db @fail

Scenario: verify the most popular book category (expected: Science Fiction → Fail)

- ✓ **Given** Establish the database connection
- ✓ **When** I execute query to find the most popular book category
- ✗ **Then** verify "Science Fiction" is the most popular book category.

```
org.junit.ComparisonFailure: expected:<[Science Fiction]> but was:<[Fantasy]>
        at org.junit.Assert.assertEquals(Assert.java:117)
        at org.junit.Assert.assertEquals(Assert.java:146)
        at
com.libraryApp.steps.LiveLabsStepDefs.verify_is_the_most_popular_book_category(Li
        at ✽.verify "Science Fiction" is the most popular book category.
(file:///Users/umute/IdeaProjects/LibraryApp-DB-
Automation/src/test/resources/features/US_02_category.feature:14)
```

▶ screenshot

✓ file:///Users/umute/IdeaProjects/LibraryApp-DB-Automation/src/test/resources/features/US_03_counting.feature

Feature: Books module: As a librarian, I should be able to get the book count of each category to manage library

@wip @db

Scenario Outline: Librarian should able to get book count based on category

- ✓ **Given** the user logged in as "librarian"
- ✓ **And** the user navigates to "Books" page
- ✓ **When** the user gets "<category>" book count
- ✓ **Then** the "<category>" book count should be equal with database

Examples:

	category
✓	Action and Adventure
✓	Anthology
✓	Classic
✓	Comic and Graphic Novel

	category
✓	Crime and Detective
✓	Drama
✓	Fable
✓	Fairy Tale
✓	Fan-Fiction
✓	Fantasy
✓	Historical Fiction
✓	Horror
✓	Science Fiction
✓	Biography/Autobiography
✓	Humor
✓	Romance
✓	Short Story
✓	Essay
✓	Memoir
✓	Poetry

✓ file:///Users/umute/IdeaProjects/LibraryApp-DB-Automation/src/test/resources/features/US_04_userStatus.feature

Feature: User module

As a librarian, I should be able to see ACTIVE/INACTIVE user count to manage library

@db

Scenario: Librarian should be able to see ACTIVE/INACTIVE user count

✓ **Given** the user logged in as "librarian"
✓ **And** the user navigates to "Users" page
✓ **When** the user gets "ACTIVE" user count
✓ **And** the "ACTIVE" user count should be equal database
✓ **When** the user gets "INACTIVE" user count
✓ **And** the "INACTIVE" user count should be equal database

@db

Scenario Outline: Librarian should able to get book count based on category

✓ **Given** the user logged in as "librarian"
✓ **And** the user navigates to "Books" page
✓ **When** the user gets "<category>" book count
✓ **Then** the "<category>" book count should be equal with database

Examples:

	category
✓	Action and Adventure
✓	Anthology
✓	Classic
✓	Comic and Graphic Novel
✓	Crime and Detective
✓	Drama
✓	Fable
✓	Fairy Tale
✓	Fan-Fiction
✓	Fantasy
✓	Historical Fiction
✓	Horror
✓	Science Fiction
✓	Biography/Autobiography
✓	Humor
✓	Romance
✓	Short Story
✓	Essay
✓	Memoir
✓	Poetry

✓ file:///Users/umute/IdeaProjects/LibraryApp-DB-Automation/src/test/resources/features/US_05_login.feature

Feature: Login Functionality

@db

Scenario: Login with valid credentials

- ✓ **Given** the user logged in "librarian56@library" and "libraryUser"
- ✓ **When** user gets username from user fields
- ✓ **Then** the username should be same with database

@db

Scenario Outline: Login with valid credentials <email>

- ✓ **Given** the user logged in "<email>" and "<password>"
- ✓ **When** user gets username from user fields
- ✓ **Then** the username should be same with database

Examples:

	email	password
✓	librarian55@library	libraryUser
✓	librarian56@library	libraryUser
✓	student5@library	libraryUser
✓	student6@library	libraryUser

✓ file:///Users/umute/IdeaProjects/LibraryApp-DB-Automation/src/test/resources/features/US_06_userManagement.feature

Feature: User management

@db

Scenario: Updating user status as librarian should change current user status in DB

- ✓ **Given** the user logged in as "librarian"
- ✓ **And** the user navigates to "Users" page
- ✓ **When** the user clicks Edit User button
- ✓ **And** the user changes user status "ACTIVE" to "INACTIVE"
- ✓ **And** the user clicks save changes button
- ✓ **Then** "The user updated." message should appear
- ✓ **And** the users should see same status for related user in database
- ✓ **And** the user changes current user status "INACTIVE" to "ACTIVE"