Spring Library Management Project

Overview

In this project, you will implement a Library Book and Patron Management System using Spring Framework. The system manages the interaction between books and patrons, with operations such as borrowing books, adding books to patrons, and displaying details. The project will use XML configuration for defining Spring beans and will verify correct bean definitions and functionality using test cases.

Project Structure

1. **XML Configuration File (`applicationContext.xml`)**: The XML file defines beans for `Book` (with prototype scope) and `Patron` (with prototype scope). **[This should be defined by you]**
2. **Beans (`Book`, `Patron`)**: Java classes representing books and patrons with properties such as `title`, `author`, and `id`**.[Already defined**]
3. **Main Application Class (`LibraryApplication`)**: Boots the Spring context, retrieves the `Book` and `Patron` beans, and simulates the borrowing process. **[This should be defined by you]**
4. **Test Cases**: A set of pre-written tests that validate the correctness of your Spring configuration, properties, and scopes. **[Already defined]**

Steps to Complete the Project

Step 1: Understand the Given Code

Before you start, ensure that you understand the provided classes:
- **Book Class**: Represents a book with properties `title` and `author`.
- **Patron Class**: Represents a patron, which can borrow books. It has properties `name` and `id`.
- **Main Application Class (`LibraryApplication`)**: Loads the Spring context and simulates borrowing a book by a patron.

Step 2: Implement the XML Configuration File (`applicationContext.xml`)

In the `applicationContext.xml` file, you will define the following beans:
1. **Book Beans**: Define two `Book` beans with prototype scope:
- bean id: `book1`: A `Book` titled `The Great Gatsby` by `F. Scott Fitzgerald`.
2. **Patron Beans**: Define one `Patron` bean with prototype scope:
- bean id: `patron1`: A `Patron` named `John Doe` with ID `12345`.

Step 3: Implement the Main Application Class (`LibraryApplication.java`)

In the `LibraryApplication.java` class:
1. **Load the Spring Context**: Use `ClassPathXmlApplicationContext` to load the Spring context from the `applicationContext.xml` configuration file.
2. **Retrieve Beans**: Retrieve the `book1` and `patron1` beans from the Spring container.
3. **Display Book and Patron Details**: Print details of the `Book` (title and author) and `Patron` (name and id) to the console.
4. **Simulate Borrowing a Book**: Have the `Patron` borrow the `Book` using the `borrowBook()` method in the `Patron` class.

Execution Steps to Follow

1. **All actions like build, compile, running application, running test cases will be through Command Terminal.**
2. **To open the command terminal the test takers need to go to the Application menu (Three horizontal lines at left top) -> Terminal -> New Terminal.**
3. **cd into your backend project folder.**
4. **To build your project use command:**

   sudo JAVA_HOME=$JAVA_HOME /usr/share/maven/bin/mvn clean package -Dmaven.test.skip

   ***If it asks for the password, provide password : pass@word1**
5. **To launch your application, move into the target folder (cd target). Run the following command to run the application:**

   java -jar <your application jar file name>
6. **To test any Restful application, the last option on the left panel of IDE, you can find ThunderClient, which is the lightweight equivalent of POSTMAN. Please use 127.0.0.1 instead of localhost to test rest endpoints.**
7. **Default credentials for MySQL:**
   a. **Username: root**
   b. **Password: pass@word1**
8. **To login to mysql instance: Open new terminal and use following command:**
   a. **sudo systemctl enable mysql**
   b. **sudo systemctl start mysql**

NOTE: **After typing any of the above commands you might encounter any warnings.**

>> Please note that this warning is expected and can be disregarded. Proceed to the next step.

c. **mysql -u root -p**

The last command will ask for password which is 'pass@word1'

9. **Mandatory: Before final submission run the following command:**

sudo JAVA_HOME=$JAVA_HOME /usr/share/maven/bin/mvn test

***If it asks for the password, provide password : pass@word1**