

STREAMINSIGHTS ANALYSER APPLICATION

IIHT

Time To Complete: 3 hrs

CONTENTS

1 Problem Statement	3
2 Business Requirements:	3
3 Implementation/Functional Requirements	3
3.1 Code Quality/Optimizations	3
3.2 Template Code Structure	4
a. Package: com.streaminsightsanalyserapplication	4
b. Package: com.streaminsightsanalyserapplication.model	4
c. Package: com.streaminsightsanalyserapplication.repository	4
4 Execution Steps to Follow	5

1 PROBLEM STATEMENT

The StreamInsights Analyzer Application provides users with the ability to perform not only basic CRUD (Create, Read, Update, Delete) operations and search functionalities in different criterias on movies, movie reviews, and user profiles but also provides the analytical operations like most watched movies, sorting movies as per their genres, getting lowest rated movie any many more for analysis and viewing.

2 BUSINESS REQUIREMENTS:

Screen Name	Console input screen
Problem Statement	<ol style="list-style-type: none">1. User needs to enter into the application.2. The user should be able to do the particular operations3. The console should display the menu<ol style="list-style-type: none">1) create a new movie2) create a new user3) create a movie review4) update a movie5) update a user6) get details of a movie7) get details of a user8) delete a movie9) delete a user10) get most watched movies11) sort movies by genre count12) get top movies watched by users of age between 25 - 3013) search movies with minimum rating of 4 by a director14) get lowest rated movie for an actor15) search movies with anime category and length under 150 minutes16) exit

3 IMPLEMENTATION/FUNCTIONAL REQUIREMENTS

3.1 CODE QUALITY/OPTIMIZATIONS

1. Associates should have written clean code that is readable.
2. Associates need to follow SOLID programming principles.

3.2 TEMPLATE CODE STRUCTURE

A. PACKAGE: COM.STREAMINSIGHTSANALYSERAPPLICATION

Resources

Class/Interface	Description	Status
StreamInsightsAnalyserApplication.java(class)	This represents bootstrap class i.e class with Main method, that shall contain all console interaction with the user.	Partially implemented

B. PACKAGE: COM.STREAMINSIGHTSANALYSERAPPLICATION.MODEL

Resources

Class/Interface	Description	Status
Movie.java(class)	This represents entity class for Movie	Partially Implemented
MovieReview.java(class)	This represents entity class for MovieReview	Partially Implemented
User.java(class)	This represents entity class for User	Partially Implemented

C. PACKAGE: COM.STREAMINSIGHTSANALYSERAPPLICATION.REPOSITORY

Resources

Class/Interface	Description	Status
MovieDao.java(interface)	This is an interface containing declaration of DAO method	Already Implemented
MovieDaoImpl.java(class)	This is an implementation class for DAO methods. Contains empty method bodies, where logic needs to be written by test taker	Partially Implemented
MovieReviewDao.java(interface)	This is an interface containing declaration of DAO method	Already Implemented
MovieReviewDaoImpl.java(class)	This is an implementation class for DAO methods. Contains empty method bodies, where logic needs to be written by test taker	Partially Implemented
UserDao.java(interface)	This is an interface containing declaration of DAO method	Already Implemented
UserDaoImpl.java(class)	This is an implementation class for DAO methods. Contains empty	Partially Implemented

	method bodies, where logic needs to be written by test taker	
--	--	--

4 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. To build your project use command:
mvn clean package -Dmaven.test.skip
4. This editor Auto Saves the code.
5. If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.
6. These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.
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 the second sql command (sudo systemctl start mysql), you may encounter a warning message like :

System has not been booted with systemd as init system (PID 1).
Can't operate. Failed to connect to bus: Host is down

>> 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. These are time bound assessments. The timer would stop if you logout (Save & Exit) and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.

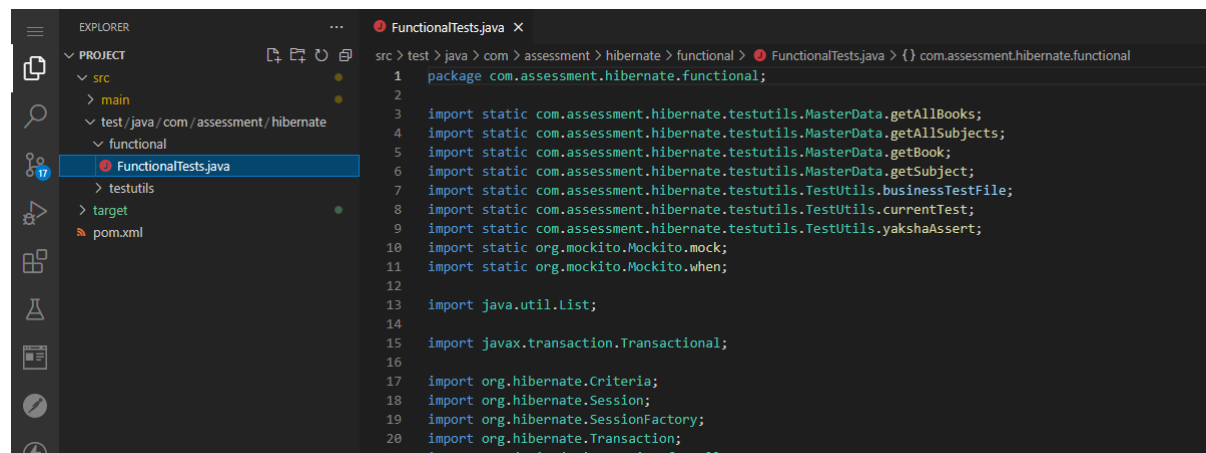
10. To run your project use command:

mvn clean install exec:java

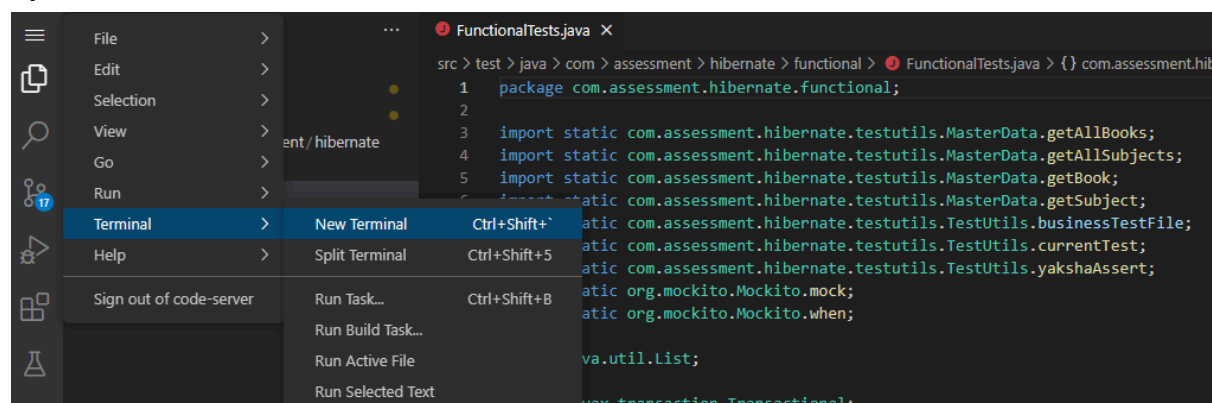
-Dexec.mainClass="com.streaminsightsanalyserapplication.StreamInsightsAnalyserApplication"

11. To test your project, use the command

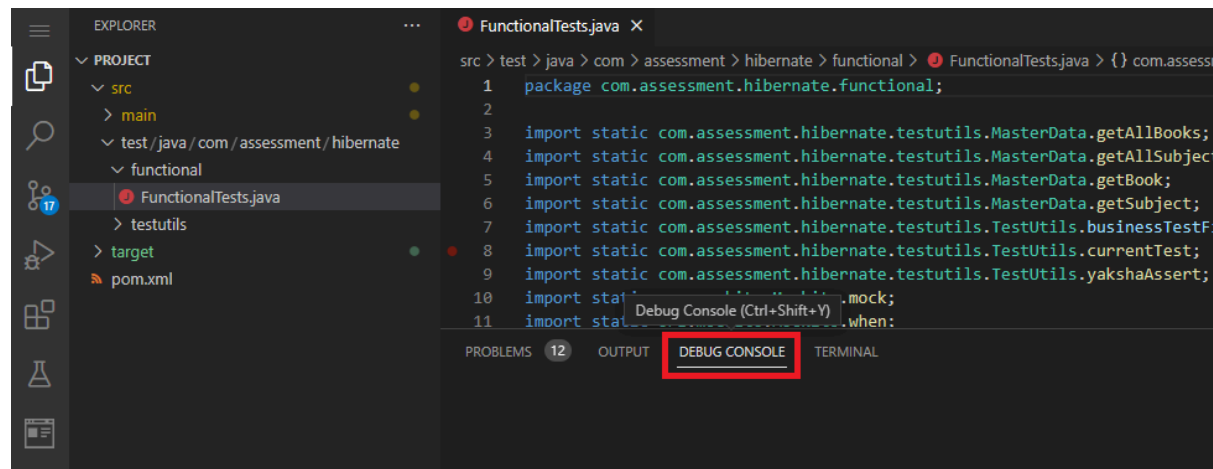
a. Open FunctionalTests.java file in editor



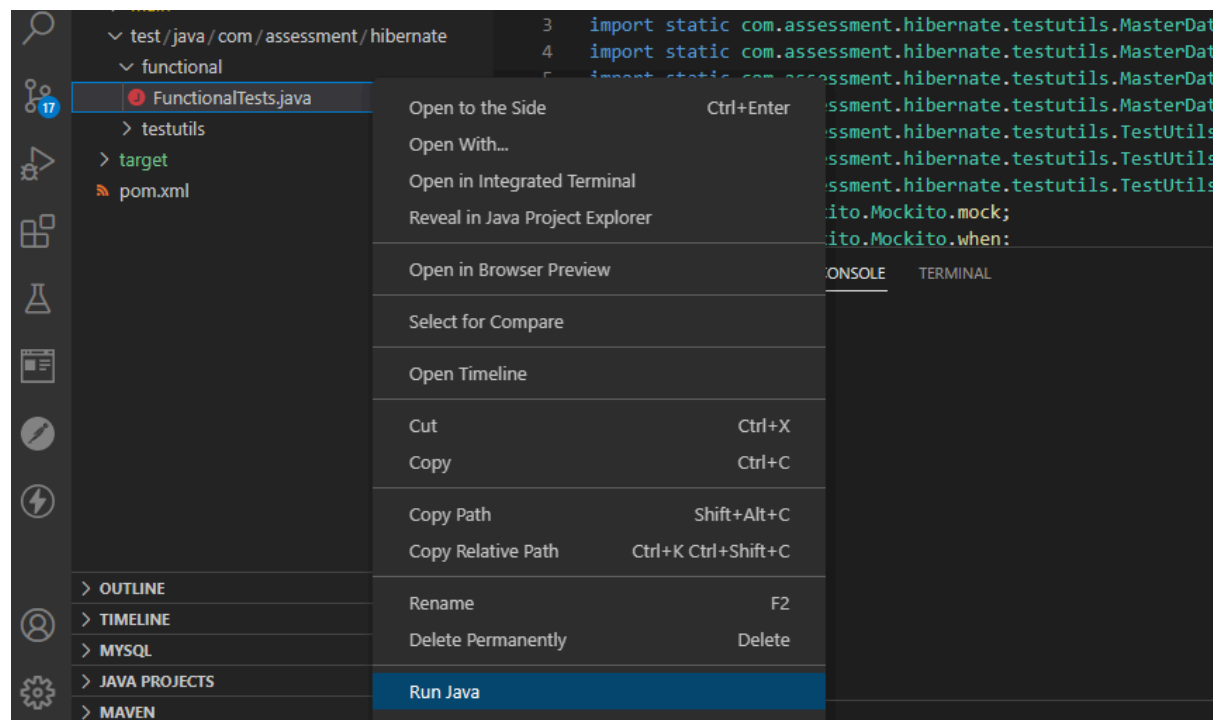
b. Open a new Terminal



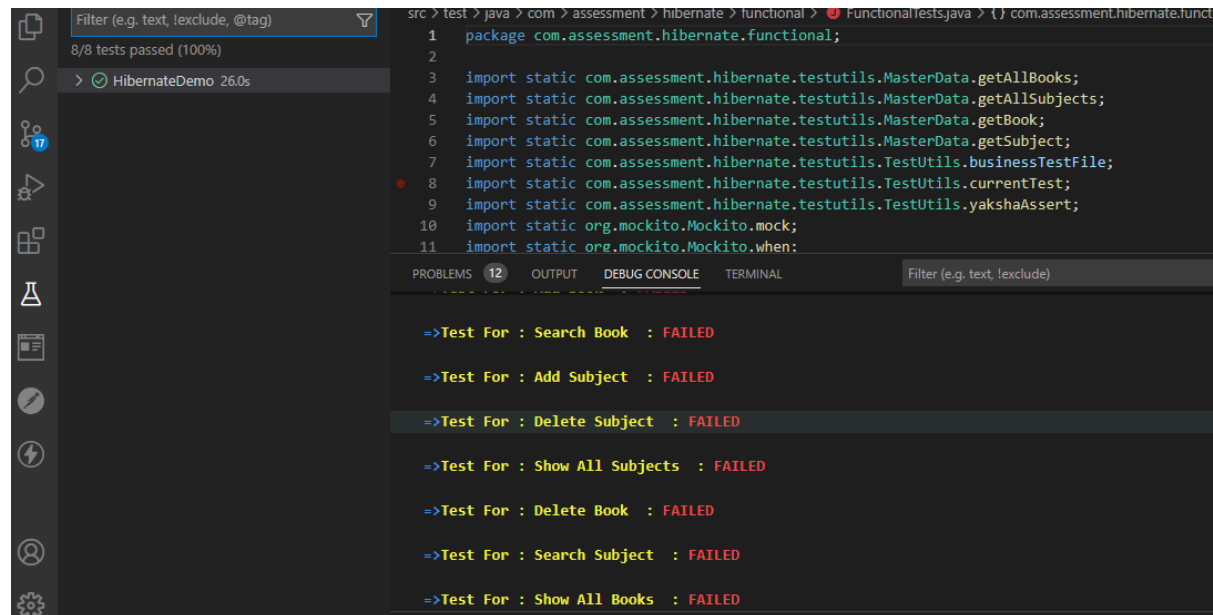
c. Go to Debug Console Tab



d. Right click on FunctionalTests.java file and select option Run Java



- e. This will launch the test cases and status of the same can be viewed in Debug Console



The screenshot shows an IDE interface with a dark theme. On the left, a sidebar contains icons for Explorer, Search, Run and Debug, and other tools. The main editor area displays a Java file named `FunctionalTests.java` with the following code:

```
1 package com.assessment.hibernate.functional;  
2  
3 import static com.assessment.hibernate.testutils.MasterData.getAllBooks;  
4 import static com.assessment.hibernate.testutils.MasterData.getAllSubjects;  
5 import static com.assessment.hibernate.testutils.MasterData.getBook;  
6 import static com.assessment.hibernate.testutils.MasterData.getSubject;  
7 import static com.assessment.hibernate.testutils.TestUtils.businessTestFile;  
8 import static com.assessment.hibernate.testutils.TestUtils.currentTest;  
9 import static com.assessment.hibernate.testutils.TestUtils.yakshaAssert;  
10 import static org.mockito.Mockito.mock;  
11 import static org.mockito.Mockito.when;
```

Below the code editor, the **DEBUG CONSOLE** tab is active, showing the following test results:

```
=>Test For : Search Book : FAILED  
  
=>Test For : Add Subject : FAILED  
  
=>Test For : Delete Subject : FAILED  
  
=>Test For : Show All Subjects : FAILED  
  
=>Test For : Delete Book : FAILED  
  
=>Test For : Search Subject : FAILED  
  
=>Test For : Show All Books : FAILED
```

12. You need to use **CTRL+Shift+B** - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.