

Lab Exercise 5

Topic: JPA I

In this exercise, we will create a Book Store App with below core functions. The front-end uses **jsp** and back-end uses **Servlet, JPA, MySQL**. Depending on your performance, you might need to spend extra time outside the class to complete all functions of this exercise.

Note: It is important to complete this Lab exercise because the next Lab exercise depends on this Lab.

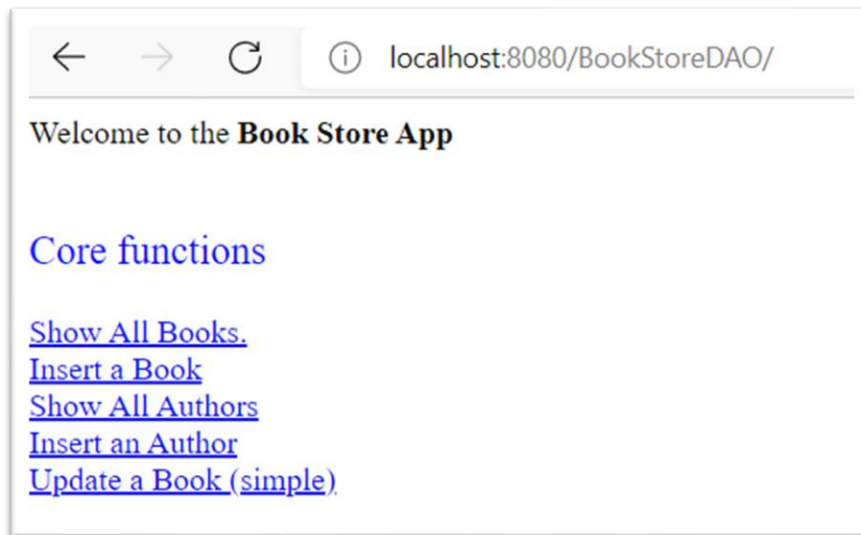


Figure 1 An example of GUI for the BookStore App

1. Download **bookdb.sql** file from the Moodle page
2. Set up **bookdb** database in MySQL server
<https://stumyadmin.cms.gre.ac.uk/>
Note: This MySQL server only works on the University network. You can create your own MySQL server by using Xampp (<https://www.apachefriends.org/download.html>)
3. Set up JNDI in Wildfly server for the connection to the **bookdb** database
4. Create a Dynamic Web Project named **BookStoreDAO**
5. Create an **index.html** with above hyperlinks to the correspondent functions of the Book Store App
6. Set up JPA to map to the **bookdb** database
7. Create packages:
 - a. model
 - b. dao
 - c. servlet
8. Create a servlet **BookStoreServlet** under the package **servlet**
9. Generate entities under the package model:
 - a. Author
 - b. Book
 - c. Bookcomment
 - d. Booksummary
10. Create a DAO named BookStoreDTO under package **dao**
11. Implement **Show All Books** of the **index.html**:

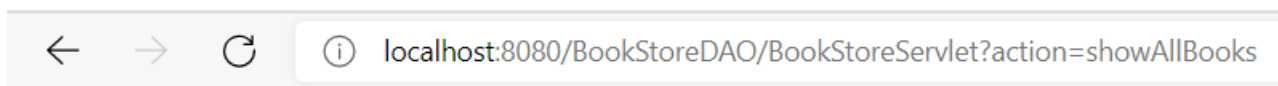
- a. Implement function `public List<Book> allBooks()` to use EntityManager to retrieve all the books from the bookdb database

```
public List<Book> allBooks() {
    List queryResults = em.createQuery("SELECT b FROM Book b").getResultList();
    List<Book> listResult = new ArrayList<Book>();

    for (int i = 0; i < queryResults.size(); i++) {
        Book b = new Book();
        b = (Book) queryResults.get(i);
        listResult.add(b);
    }

    return listResult;
}
```

- b. Implement **showAllBooks** in the BookStoreServlet and link this to the function **Show All Books** of the **index.html**



Displayed @ Sun Feb 06 02:26:27 GMT 2022

ID	Title
1	The Catcher in the Rye
2	Nine Stories
3	Franny and Zooey
4	The Great Gatsby
5	Tender id the Night
6	Pride and Prejudice
7	Professional ASP.NET 4.5 in C# and VB
12	The Book Thief
13	The Man Who Died Twice: Thursday Murder Club
14	Angel Maker: An Unputdownable Scandinavian Crime Thriller With A Chilling Twist

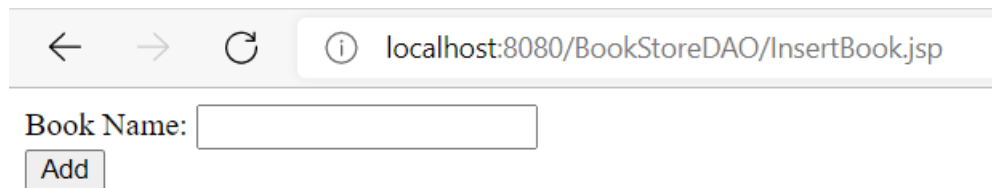
12. Implement **Insert Book** function:

- a. Implement the `public void insertBook(String name)` of the BookStoreDTO using EntityManager to insert a new book into the database

```
public void insertBook(String name) {
    Book b = new Book();
    b.setTitle(name);

    em.persist(b);
}
```

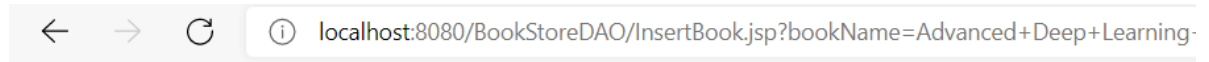
- b. Implement inserting book in BookStoreServlet
c. Create a jsp file named **InsertBook.jsp**



Book Name:

Add

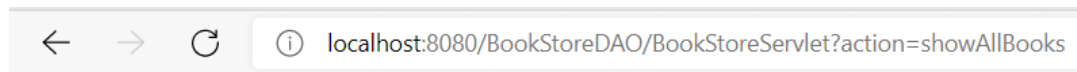
- d. When the Add button clicked, the data will be transferred to the BookStoreServlet to handle inserting a book. And show the status to the users, for example



Displayed @ Sun Feb 06 02:40:02 GMT 2022

Book inserted

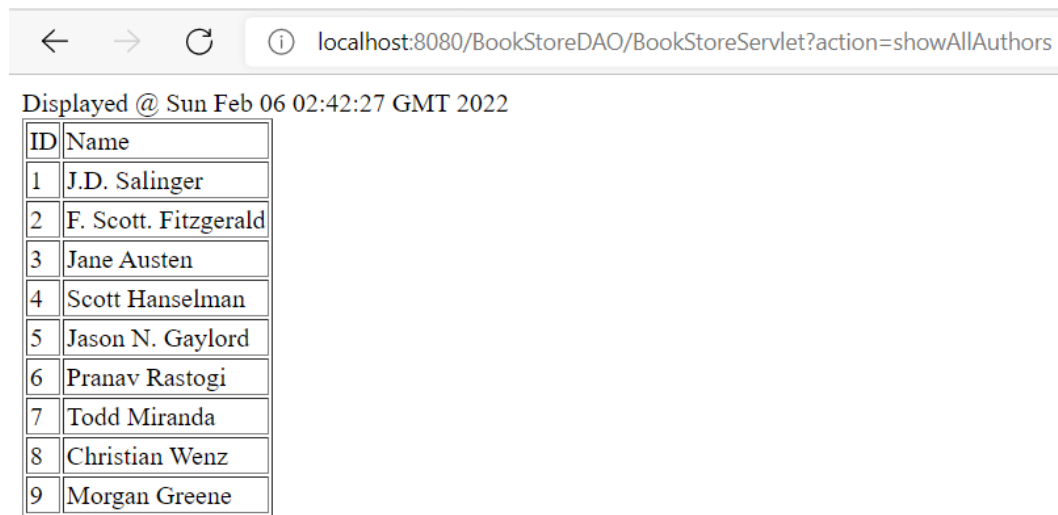
- e. Verify the book inserted by using **Show All Books** function



Displayed @ Sun Feb 06 02:41:50 GMT 2022

ID	Title
1	The Catcher in the Rye
2	Nine Stories
3	Franny and Zooey
4	The Great Gatsby
5	Tender id the Night
6	Pride and Prejudice
7	Professional ASP.NET 4.5 in C# and VB
12	The Book Thief
13	The Man Who Died Twice: Thursday Murder Club
14	Angel Maker: An Unputdownable Scandinavian Crime Thriller With A Chilling Twist
15	Advanced Deep Learning with TensorFlow 2 and Keras

13. Implement **Show All Authors** (similar to Show All Books)



Displayed @ Sun Feb 06 02:42:27 GMT 2022

ID	Name
1	J.D. Salinger
2	F. Scott. Fitzgerald
3	Jane Austen
4	Scott Hanselman
5	Jason N. Gaylord
6	Pranav Rastogi
7	Todd Miranda
8	Christian Wenz
9	Morgan Greene

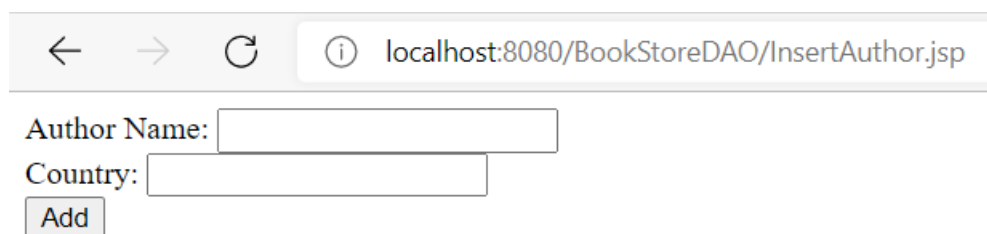
14. Implement **Insert an Author** function (similar to the Insert a Book):

- Implement the `public void insertAuthor(String name, String country)` of the BookStoreDTO using EntityManager to insert a new author into the database

```
public void insertAuthor(String name, String country) {
    Author a = new Author();
    a.setName(name);
    a.setCountry(country);

    em.persist(a);
}
```

- Implement inserting an author in BookStoreServlet
- Create a jsp file named **InsertAuthor.jsp**



Author Name:

Country:

- When the Add button clicked, the data will be transferred to the BookStoreServlet to handle inserting an author.
- Verify the author inserted by using **Show All Authors** function

15. Implement Update a Book

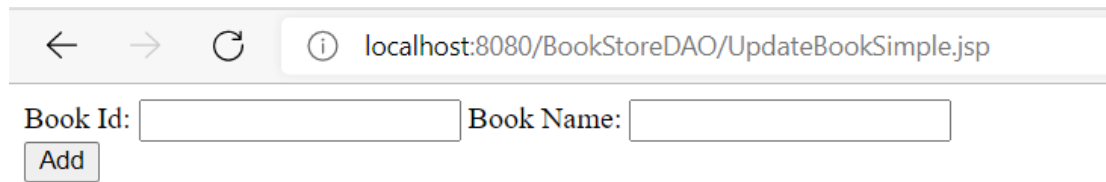
- Implement the `public void updateABookSimple(int bookId, String bookName)` of the BookStoreDTO using EntityManager to update the name of a book basing its Book ID

```
public void updateABookSimple(int bookId, String bookName)
{
    Book a = em.find(Book.class, bookId);

    a.setTitle(bookName);
}
```

```
        em.persist(a);  
    }
```

- b. Implement update a book in BookStoreServlet
- c. Create a jsp file named **UpdateBookSimple.jsp**



← → ↻ ⓘ localhost:8080/BookStoreDAO/UpdateBookSimple.jsp

Book Id: Book Name:

- d. When the Add button clicked, the data will be transferred to the BookStoreServlet to handle updating a book.
- e. Verify the book updated by using **Show All Books** function