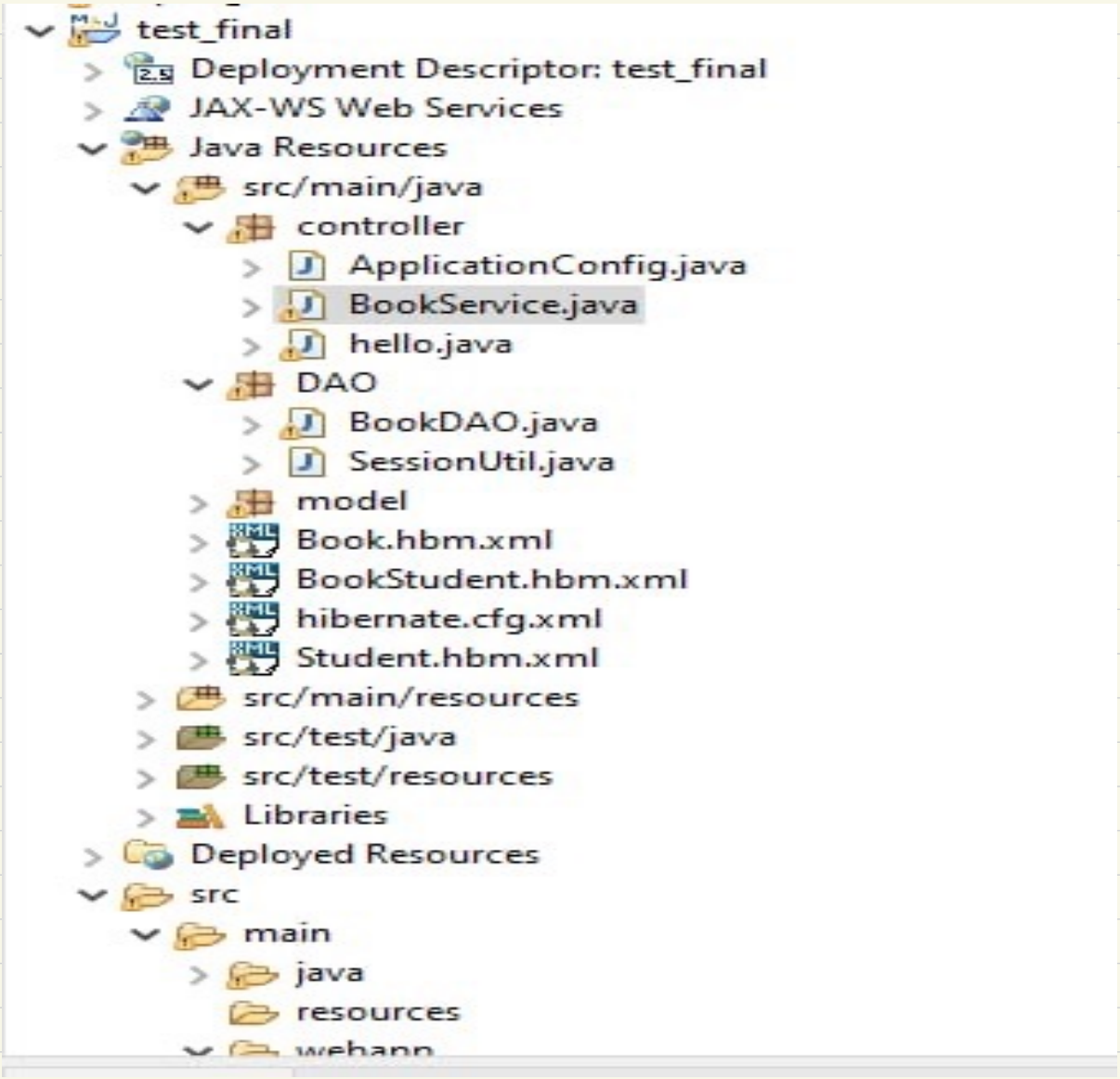
test_final
Deployment Descriptor: test_final
JAX-WS Web Services
Java Resources
src/main/java
controller
ApplicationConfig.java
BookService.java
hello.java
DAO
BookDAO.java
SessionUtil.java
model
Book.hbm.xml
BookStudent.hbm.xml
hibernate.cfg.xml
Student.hbm.xml
src/main/resources
src/test/java
src/test/resources
Libraries
Deployed Resources
src
main
java
resources

```java
package DAO;
import org.hibernate.Criteria;
import javax.persistence.criteria.CriteriaBuilder;
import javax.persistence.criteria.CriteriaQuery;
import javax.persistence.criteria.Root;
import java.util.ArrayList;
import java.util.List;

import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.TransactionException;


import model.Book;

public class BookDAO {

  public ArrayList<Book> getAllBooks() {
     Session session = SessionUtil.getSession();
     Query query = session.createQuery("from Book");
     ArrayList<Book> books = (ArrayList<Book>) query.list();

     System.out.println("Total Books: " + books.size());
     for (Book book : books) {
        System.out.println("Book ID: " + book.getIdBook() + ", Name: " + book.getName());
     }
     session.close();
     return books;
  }

  public int addBook(Book b) {
     try {
         System.out.println("Saving Book: " + b.getName());
        Session session = SessionUtil.getSession();
        Transaction tx = session.beginTransaction();

        session.save(b);
        tx.commit();
          System.out.println("Saving Book: " + b.getName());
        session.close();

```

```java
package controller;

import java.io.IOException;

import java.util.ArrayList;
import java.util.List;

import javax.ws.rs.ApplicationPath;
import javax.ws.rs.Consumes;
import javax.ws.rs.DELETE;
import javax.ws.rs.GET;
import javax.ws.rs.POST;
import javax.ws.rs.PUT;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.Application;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;

import model.Book;
import DAO.BookDAO;


@ApplicationPath("/eiei")
@Path("/services")
public class BookService extends Application {

   BookDAO bookDao = new BookDAO();
```

```
28
29    BookDAO bookDao = new BookDAO();
30
31    @GET
32    @Path("/books")
33    @Produces({MediaType.APPLICATION_JSON + ";charset=UTF-8"})
34    public Response getBooks() {
35        try {
36            List<Book> books = bookDao.getAllBooks(); // ✅ ดึงข้อมูลจาก DAO
37
38            // ✅ Debug Log เพื่อตรวจสอบข้อมูลก่อนส่งออก
39            System.out.println("Sending JSON Response: " + books);
40
41            if (books == null || books.isEmpty()) {
42                return Response.status(Response.Status.NOT_FOUND)
43                        .entity("No books found")
44                        .build();
45            }
46
47            return Response.ok().entity(books).build(); // ✅ ส่งข้อมูลเป็น JSON
48        } catch (Exception e) {
49            e.printStackTrace(); // ✅ แสดง Error ใน Console
50            return Response.status(Response.Status.INTERNAL_SERVER_ERROR)
51                    .entity("Error retrieving books: " + e.getMessage())
52                    .build();
53        }
54    }
55
56
57    @GET
58    @Path("/test")
59    @Produces(MediaType.TEXT_PLAIN)
60    public String testEndpoint() {
61        return "test";
62    }
63
```

```
4
5     @POST
6     @Path("/books")
7     @Consumes(MediaType.APPLICATION_JSON)
8     public Response createCustomer(Book book) throws IOException {
9     int i = bookDao.addBook(book);
0     if (i == 1)
1     return Response.status(201).entity(" create successfully").build();
2     else
3     return Response.status(201).entity(" create fail").build();
4     }
5
6     @PUT
7     @Path("/books/{param}")
8     @Produces(MediaType.APPLICATION_JSON)
9     public Book updateCustomer(@PathParam("param") Integer id , Book book)
0     {
1        Book newbook = bookDao.updateBook(book, id);
2        return book;
3
4     }
5     //
6     @DELETE
7     @Path("/books/{param}")
8     @Produces(MediaType.APPLICATION_JSON)
9     public Book updateCustomer(@PathParam("param") Integer id)
0     {
1        Book books = new Book();
2
3        books = bookDao.deleteBook(id);
4     return books;
5
6     }
7
8
```

```
@GET
@Path("/BooksName/{param}")
@Produces(MediaType.APPLICATION_JSON)
public Book getCustomerByname(@PathParam("param") String name ){
return bookDao.getBookByName(name);
}

@GET
@Path("/BooksId/{param}")
@Produces(MediaType.APPLICATION_JSON)
public Book getCustomerByname(@PathParam("param") Integer  id  ){
return bookDao.getBookById(id);
}
```

```
public int addBook(Book b) {
   try {
        System.out.println("Saving Book: " + b.getName());
      Session session = SessionUtil.getSession();
      Transaction tx = session.beginTransaction();

      session.save(b);
      tx.commit();
        System.out.println("Saving Book: " + b.getName());
      session.close();

   } catch (TransactionException e) {
      e.printStackTrace();
      return 0;
   }
   return 1;
}

public Book findCustomer(String userName) {
   Session session = SessionUtil.getSession();
   Criteria cr = (Criteria) session.createQuery("from Book where name="+userName);
   return (Book) cr.list().get(0) ;
   }

public Book updateBook(Book book, Integer id) {
   Session session = SessionUtil.getSession();
   Transaction tx = session.beginTransaction();
   List<Book> booklist = getAllBooks();
   for (Book b : booklist) {
      if (b.getIdBook() == id) {
         b.setIdBook(id);
         b.setName(book.getName());
         session.update(b);
         tx.commit();

         session.close();

         return b;
      }
```

```
97  public Book getBookByName(String  name) {
98      Session session = SessionUtil.getSession();
99      Transaction tx = session.beginTransaction();
00      List<Book> booklist = getAllBooks();
01      for (Book b : booklist) {
02          if (b.getName().equals(name))
03          {
04              tx.commit();
05              session.close();
06
07              return b;
08          }
09      }
10      tx.commit();
11      session.close();
12      return null;
13  }
14
15  public Book getBookById(Integer  id) {
16      Session session = SessionUtil.getSession();
17      Transaction tx = session.beginTransaction();
18      List<Book> booklist = getAllBooks();
19      for (Book b : booklist) {
20          if (b.getIdBook() == id)
21          {
22              tx.commit();
23              session.close();
24
25              return b;
26          }
27      }
28      tx.commit();
29      session.close();
30      return null;
31  }
32
33
34
35
36  }
```

```
57  public Book updateBook(Book book, Integer id) {
58      Session session = SessionUtil.getSession();
59      Transaction tx = session.beginTransaction();
60      List<Book> booklist = getAllBooks();
61      for (Book b : booklist) {
62          if (b.getIdBook() == id) {
63              b.setIdBook(id);
64              b.setName(book.getName());
65              session.update(b);
66              tx.commit();
67
68              session.close();
69
70              return b;
71          }
72      }
73      session.close();
74      return null;
75  }
76
77
78
79  public Book deleteBook(Integer id) {
80      Session session = SessionUtil.getSession();
81      Transaction tx = session.beginTransaction();
82      List<Book> booklist = getAllBooks();
83      for (Book b : booklist) {
84          if (b.getIdBook() == id) {
85
86              session.delete(b);
87              tx.commit();
88
89              session.close();
90
91              return b;
92          }
93      }
94      session.close();
95      return null;
96  }
```

```java
package model;
// Generated Mar 20, 2025, 3:27:53 PM by Hibernate Tools 5.5.9.Final

import java.util.HashSet;

/**
 * Book generated by hbm2java
 */
@XmlRootElement
@JsonIgnoreProperties({"bookStudents"})
public class Book implements java.io.Serializable {

    private Integer idBook;
    private String name;
    private Set bookStudents = new HashSet(0);

    public Book() {
    }

    public Book(String name) {
        this.name = name;
    }

    public Book(String name, Set bookStudents) {
        this.name = name;
        this.bookStudents = bookStudents;
    }

    public Integer getIdBook() {
        return this.idBook;
    }

    public void setIdBook(Integer idBook) {
        this.idBook = idBook;
    }

    public String getName() {
        return this.name;
    }

    public void setName(String name) {
        this.name = name;
    }
```