EAD Assignment1

IT No: IT 11 1609 08

Name ; P.K.D.S.Alahakoon

1. Use case diagram

User

Add Authors

View All Books

View All Authors

<<Include>>

Fill book information and submit

<<Include>>

Add a new Book

Select Search

Option

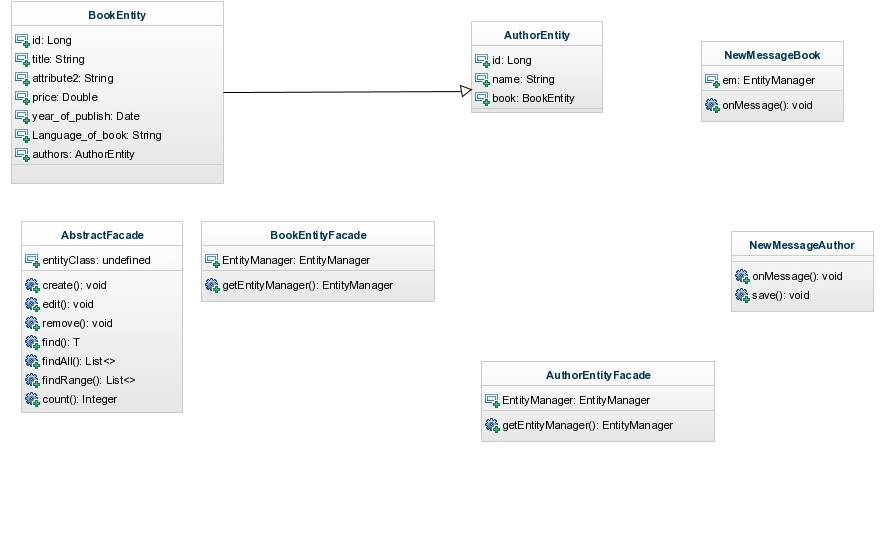
<<Include>>

Search a Book

Select a Book

Book Catalogue System

2. Class Diagram



3. Code explanation

3.1The structure of this application generally corresponds to the following tiers.

* Business Tier.
* Web Tier
* EIS Tier

Creating Entity classes

AuthorEntity and BookEntity are the class that

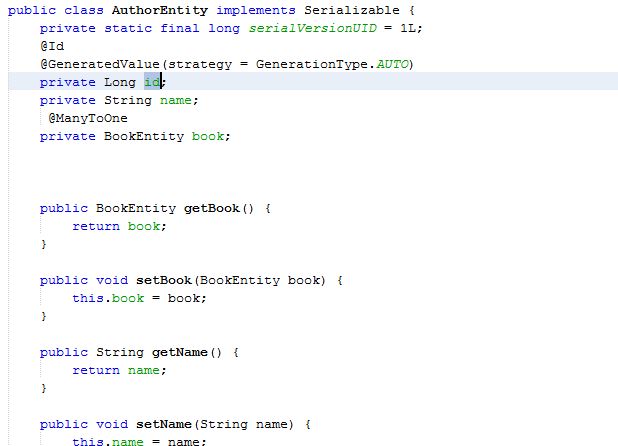
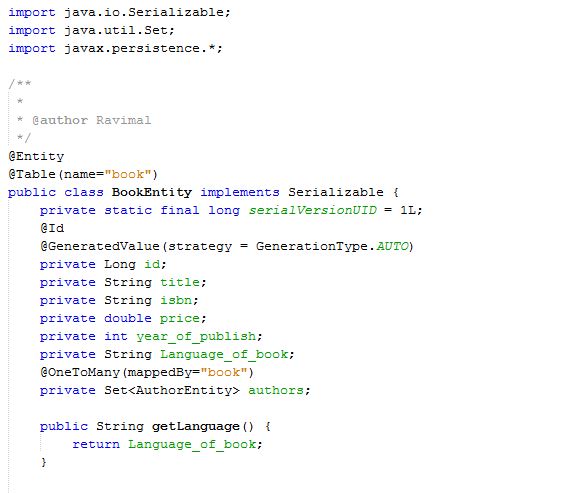
Represent “book” and “author” tables in database.

@Entity annotation to define the class as an entity class

@Table(name="book") annotation to define the name of the table

Each Entity class has its own attributes.

Getters and Setters has defined for each and every attribute in two entity classes.

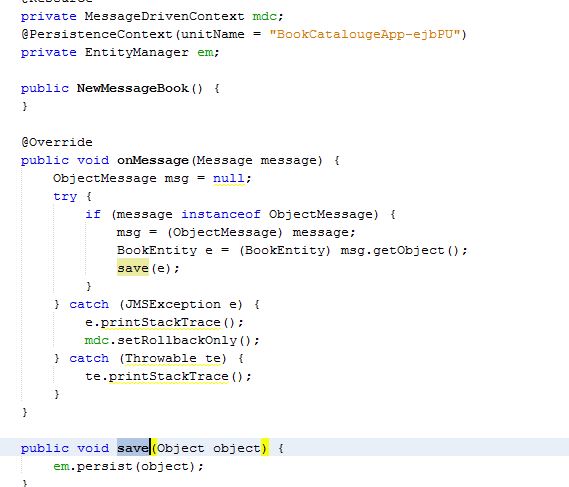


Getters and Setters are implemented to get and set attribute values of an Entity object.

3.2 Message-driven bean in the EJB module. The message-driven bean receives and processes messages sent to the queue by a servlet in the web module.

There are two Message-driven beans in this project .

1. NewMessageAuthor
2. NewMessageBook



When the queue receives a message, the EJB container invokes the message listener method or methods. For a bean that uses JMS, this is the onMessagemethod of the MessageListener interface.

A message listener method must follow these rules:

* The method must be declared as public.
* The method must not be declared as final or static.

The onMessage method is called by the bean’s container when a message has arrived for the bean to service. This method contains the business logic that handles the processing of the message. It is the message-driven bean’s responsibility to parse the message and perform the necessary business logic.

The onMessage method has a single argument: the incoming message.

The signature of the onMessage method must follow these rules:

* The return type must be void.
* The method must have a single argument of type javax.jms.Message.

@MessageDriven(mappedName = "jms/NewMessage", activationConfig = {

@ActivationConfigProperty(propertyName = "acknowledgeMode",

propertyValue = "Auto-acknowledge"),

@ActivationConfigProperty(propertyName = "destinationType",

propertyValue = "javax.jms.Queue")

})

public class NewMessageBook implements MessageListener {

The @MessageDriven annotation tells the container that the component is a message-driven bean

and specifies the JMS resource used by the bean. When the IDE generates the class, the Mapped

Name of the resource (jms/NewMessageBook) is derived from the name of the class

(NewMessageBook.java). The JMS resource is mapped to the JNDI name of the destination from which

the bean receives messages. The New Message-Driven Bean wizard also adds the information for the

JMS resources to sun-resources.xml.

In the NewMessageBook class, the onMessage method casts the incoming message to a ObjectMessage and and then cast it in to a BookEntity.

if (message instanceof ObjectMessage) {

msg = (ObjectMessage) message;

BookEntity e = (BookEntity) msg.getObject();

save(e);

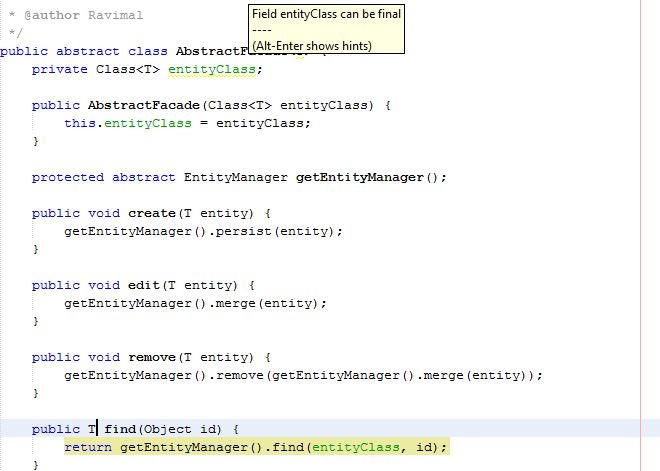
3.3 Class BookEntityFacade.java and AbstractFacade.java and opens the files in the editor. As

you can see in the following code, the annotation @Stateless is used to

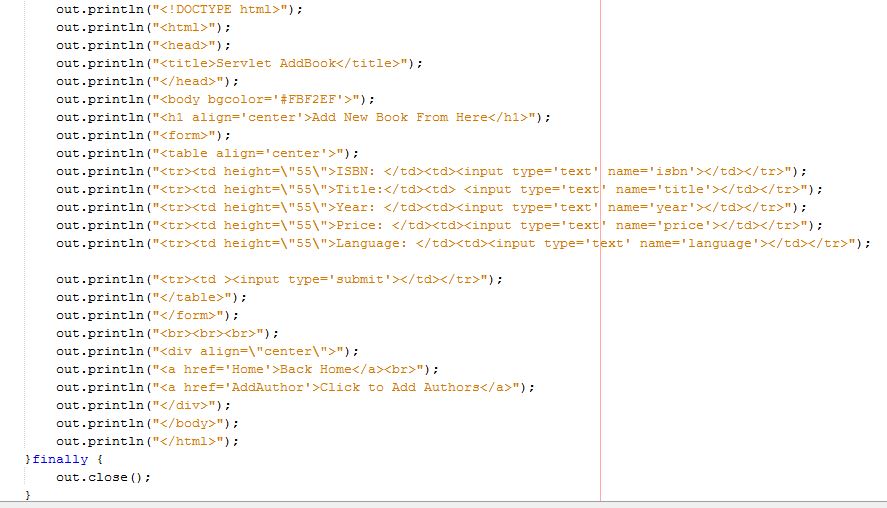
declare BookEntityFacade.java as a stateless session bean component.



BookEntityFacade.java



AbstractFacade.java



3.4 Public class AddBook extends HttpServlet is used to tell that AddBook.java class is a servlet. servlet is using for displaying the stored messages .

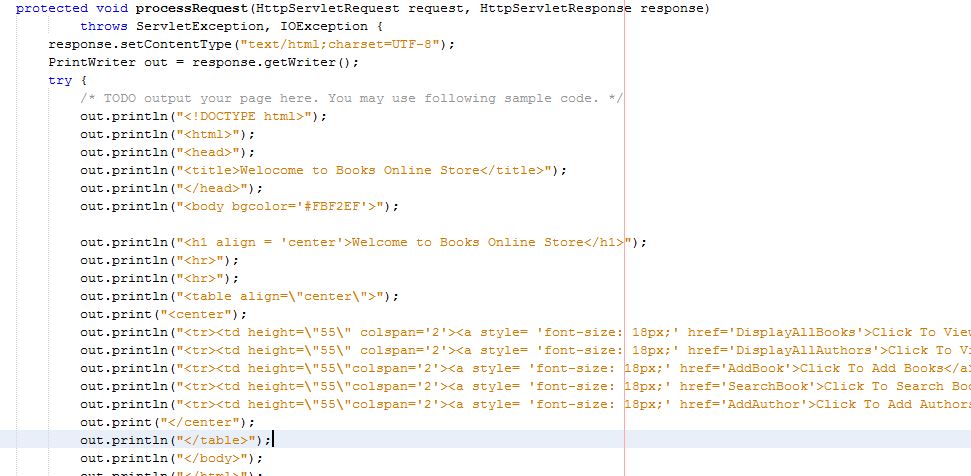
@EJB annotation to inject the enterprise bean

“@EJB

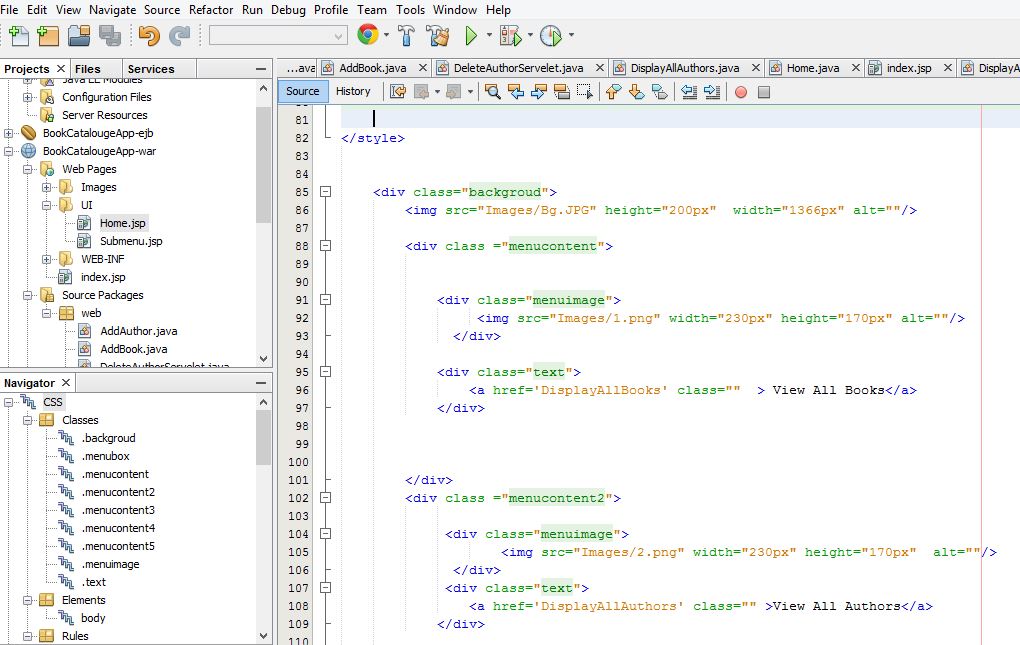
private BookEntityFacade bookEntityFacade”

Above code figure show how AddBook funvtion works.

Other HTML coding



Create a JSP page for Interface



4) User interfaces



