Université de Lorraine
DIPLOME: MASTER Informatique
Date : 27 février 2017

Nom:.....

Prénom:.....

GLA

1 Library management

The library of our University wishes to computerize the management of its books using an approach based on the JEE technology. The application must allow the display of the bibliographic fund, the registration of any new book, the possibility to submit (to the administrative-financial service) purchase requests and the management of reply messages for this kind of requests, the possibility to send messages to users who borrow books.

Each book (class `Book`) has a title and a certain number of authors. We can have two books with the same title but not two authors with the same name. We want to know at any time the user (class `User`) who borrows the book (if any) and, if applicable, the start date of the loan. We also wish to record the number of loans since the book has been aquired by the library.

You should develop the beans necessary for the business logic of the application; we assume that the presentation part (JSF and NamedBean) of the application is already developed or will be developed later on. The application uses a database available in the container via the Persistence Unit called `ExamBibliothequePU`.

1.1 Data model

Propose a data model for the application by making explicit the necessary entities and their associations. Propose a JPA implementation of this model. For each proposed class you must detail all the attributes and all the annotations needed to persist with a JPA approach (we assume that getters/setters are generated automatically for each attribute).

1.2 Add books

The application proposes a web page to add books to the bibliographic fund. You should develop an EJB `BookManagerBean` implementing this feature. More specifically, the bean should implements the method

```
void addBook(Book book);
```

that will persist in the database the book provided as parameter. You should detail the Java code and the necessary annotations. You must also identify and detail any modifications that must be made to the classes of question 1.1.

1.3 Borrow a book

The library staff uses an interface allowing them to handle book loans. Once a book is borrowed by a subscriber the borrow counter of the book is incremented in the database and a mail is sent to the subscriber to confirm the loan. An EJB for sending mails is already available but you should be aware that the method `sendMail` for sending mails can take a long time:

```
public class MailSenderBean implements MailSender {

    public void sendMail(String address, String message) {
        // l'envoi de ce mail va prendre beaucoup de temps
        ...
        System.out.println("  Mail to "+ address + " finally sent : " + message);
    }
}
```

The bean `BookManagerBean` of question 1.2 should now implement the method

```
void borrowBook(Long idBook, String clientAddress)
```

which allows one to record that the book with the identifier `idBook` has beeb borrowed by the user with the id `idUser` and to send a confirmation email to the user.

Complete the bean `BookManagerBean` to implement this functionality. You should detail the Java code and the annotations needed for the concerned classes and methods. Complete **directly on this page** the class `MailSenderBean` with all the necessary annotations for avoiding unnecessary waiting when calling its long-running methods.

1.4 Statistiques

The application provides statistics (number of books, most demanded book, etc.) which are used in different interfaces of the application. The statistics are computed in the method `updateStatistics` of the bean `Statistics` and can be accessed through the getters supposed available for all of the attributes of the bean.

We assume that the method `getAllBooks` of the bean `BookManagerBean` allows one to retrieve all the books in the library but it is potentially time consuming. You must use a caching mechanism to optimize the response speed and for this purpose the statistics will be computed only at application startup and every day at midnight.

```
public class Statistics {
    private int nbBooks;
    private Book mostDemanded;


    private BookManager bookManager;



    public void updateStatistics() {
        List<Book> res = bookManager.getAllBooks();
        // compute statistics: nbBooks, mostDemanded, ...
        ...
    }
    ...
}
```

Complete **directly on this page** the bean `Statistics` with the necessary annotations to implement such a behaviour.

1.5 Purchase requests

The purchase requests for new books are done by the library staff and sent in a JMS topic to be handled asynchronously by the SAF application (administrative-financial service). We suppose that the bean `BookManagerBean` of question 1.2 implements a method which allows the sending of requests as objects of type

```
public class Request implements Serializable {
    private String from;
    private String description;
    private String reply;
    ...
}
```

with the attributes `from` and `description` filled according to the attributes of the book passed as parameter. We suppose that all the getters/setters and constructors of the class `Request` are already implemented.

The responses to these requests are messages with a content of type `Request` sent to the application SAF in a queue `ExamBibliothequeQueue`. Our application will use a bean `ReplyHandler` to get these messages and print their content as follows:

```
Reply to request message: «description» - «reply»
```

Implement the bean `ReplyHandler` to accomplish the reception of responses from SAF. You should detail the Java code as well as the annotations necessary for the concerned methods and classes.