Enterprise Application Integration
Practical Assignment 2

João Ferreira & João Silva
Departamento de Engenharia Informática
Universidade de Coimbra
jpbat@student.dei.uc.pt | jfmsilva@student.dei.uc.pt
2009113274 | 2008111448

November 2013

Contents

1 Introduction **3**

2 Data Model **4**

3 Enterprise JavaBeans **5**

4 Navigation **7**
4.1 Login 7
4.2 Index 7
4.3 Me 7
4.4 Protections 7

5 Division of Work **8**

6 Conclusion **9**

1 Introduction

In this practical assignment it was requested that we reused the first one (just the crawler) to populate a database using JPA.

As the both of us had made the first assignment separated, our first step was to decide wich one of the crawlers to use.

Once the database was populated we used that data to later build a web site, in wich the user could log in and browse through the movies and sort them alfabetically or filter them either by score (bigger then, smaller then, or between an interval).

Besides that the user could also set his favorite list wich alowed him to receive email updates, each time a movie that belonged to at least one of his favorite catogories was added.

We'll now briefly explain some decisions that were taken and how we solved some issues that appeard along the way.

2 Data Model

We used the classes that were generated on the first assignment, and we added some anotations, so that the tables were automaticly generated.

So that we can understand better we attach an ER model from our database.

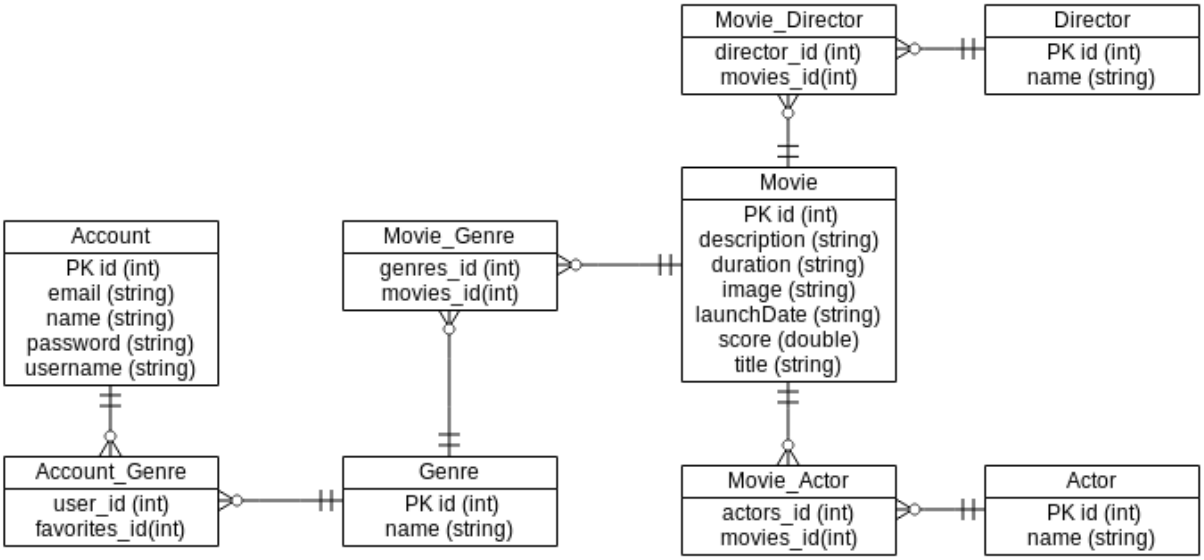

Figure 1: ER Model.

As we can see we have some weak entities that make the connection between our main entities (Acount, Movie, Genre). We decided that that was the best decision to take, once the other option was to put the full string in each one of the entities. With that option we can make any conection between the entities in a much simpler and faster way.

For instance, instead of searching the substring "Drama" in each of the Movies genre string, we can just request to the database all the movies id that had that connection in the Genre_Movie table.

Since we needed the foreign relations from the movies and the users in most cases, we decided not to use the lazy loading from the database.

3 Enterprise JavaBeans

While creating the business logic layer, we implemented the CRUD (Create, Read, Update and Delete) class. This class contains all the primary functionalities and initialisations used by the Beans. We created the following Beans:

AccountService : Statefull Bean to store the User information on log.

AccountManagerService : Stateless Bean used to Search users in the Database.

EmailService : Singleton Bean used to create an EmailDipatcher when the server starts and to give access to the mail queue.

MovieManagerService : Message Driven Bean responsible for receiving new movie lists from the topic and sending the new Movies to the Database and to ask the Email dispatcher to send Emails to the users reporting new movies.

DirectorService : Stateless Bean Used to perform basic operations related to Actors.

GenreService : Stateless Bean Used to perform basic operations related to Genres.

MovieService : Stateless Bean Used to perform basic operations related to Movie. We also added custom query's to access the database like search by genre and filter by score.

All the queries to the database are created using the CriteriaBuilder to prevent SQL injection.

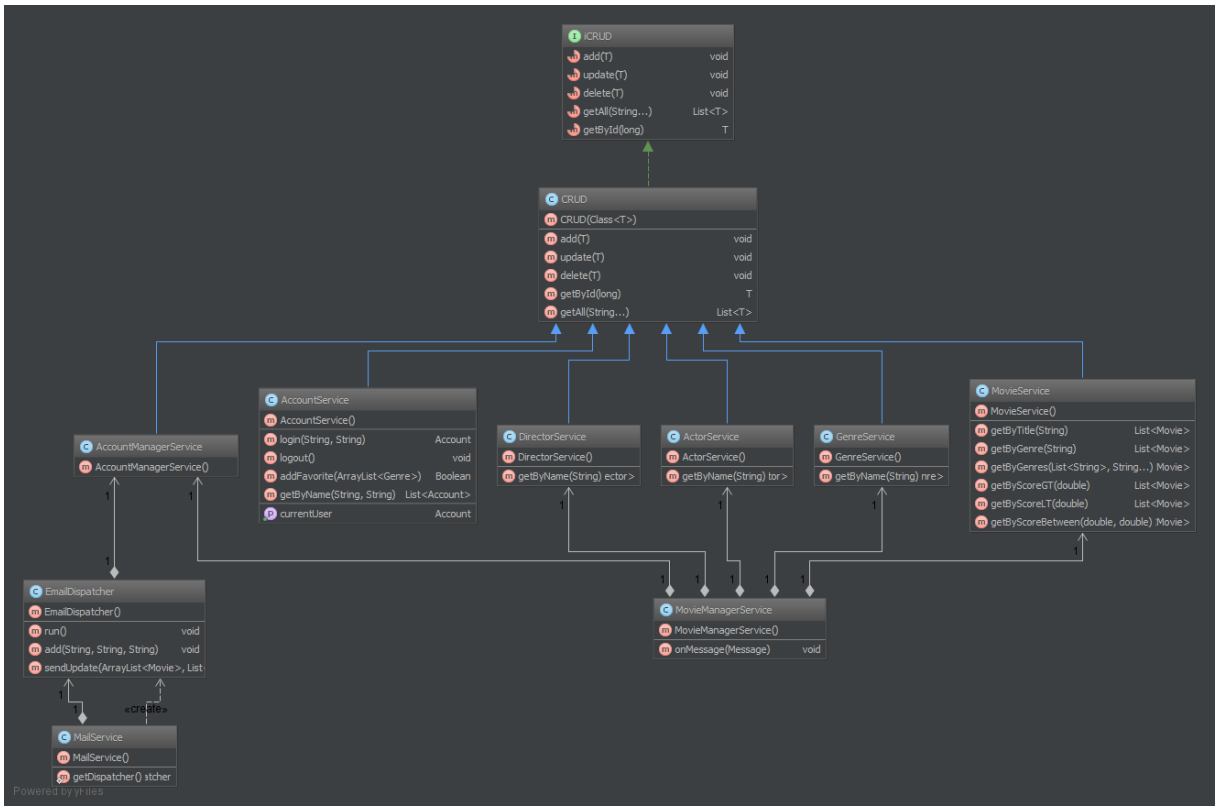

Figure 2: Services Diagram.

4 Navigation

To navigate through the website the user can access to three diferent pages. The mapping of the pages is made with Java Servlets and the pages are built with JSP.

In the top of the page, there is always either the information about the logged in user or a login form.

4.1 Login

In this page the user can either register filling a small form with important information. Besides that an already registred user can login into the application, by filling the form available in the top of the page.

4.2 Index

In this page the user can filter the movies that are showed, order them in alfabetically, access to his personal page by clicking his name, and logout by hovering the mouse over his picture.

The filtering and sorting are accessed by clicking on the button available on the top of the page. A modal is showed in wich the user can select the desired option.

4.3 Me

In this page the user can edit his favorites and see the movie with the best score that belongs to one of his favorite categories.

4.4 Protections

Besides what was already said about the SQL Injection, we also protected the aplication against either access to unauthorized pages and scripting injection in the forms.

5 Division of Work

The work was, in what appeard to us, equally divided.

João Ferreira navigation, business model. (40h)

João Silva entities, business model. (40h)

6 Conclusion

With this project we reforced idea that the abstraction of comunication is really important.

We also understood the importance of the usage of statefull and stateless beans, as well as the importance of keeping the implementation simple and as abstract from what we are trying to do as possible.

With this we consider that the execution of this assignment was a success, for wich the both of us learned some important aspects of the importance of using JPA to communicate with a database, along other things.

References

[1] Professor Filipe Araújo blog.

[2] jQuery website.

[3] Twitter's Bootstrap website.

[4] W3Schools website.

[5] JBoss community.