Relatório
Assignment #2

Integração de Sistemas

Carolina Afonso Leite Montezuma de Carvalho nº2014225913
João André Nunes Agnelo nº2017297643

Índice

1.Introdução

No âmbito da unidade curricular de Integração de Sistemas criámos um projeto denominado **Webflix** dividido em três camadas: *Data*, *Business* e *Presentation*.

Na camada *Data* utilizamos a tecnologia *Java Persistance API* (JPA), que trata do mapeamento objeto-relacionamento (ORM), e faz a ligação entre a aplicação e a base de dados (MySQL).

Na camada *Business* é implementada toda a lógica da aplicação e serve como intermediário entre a camada Data e a camada *Presentation.*

A camada *Presentation* serve de interface entre os utilizadores e a aplicação. Esta trata de redirecionar os utilizadores para as respetivas páginas consoante os pedidos por estes efetuados.

A secção 2 descreve a camada *Presentation*. A secção 3 descreve a camada *Business*. A secção 4 descreve a camada Presentation.

A secção 5 trata da gestão de projetos e packaging.

2. Camada Presentation

2.1 Servlet

Para este projeto utilizámos um servlet “PlayersTallerThan” recorre aos DTOS provenientes da camada *Business* para fornecer conteúdo *web* aos utilizadores.

Adicionalmente, é responsável por controlar a navegação dos utilizadores pelas várias páginas *web* tendo em conta as permissões dos mesmos. Por exemplo, os utilizadores não registados não deverão ter acesso a qualquer outra página à excepção das páginas de registar e login.

Figura 1: Ecrã principal de um utilizador registado

Na figura 1 podemos observar a página principal dos utiizadores através da qual estes podem consultar todos os conteúdos existentes na aplicação, consultar a sua *watch list* ou editar os seus detalhes de conta (ou até apagar a mesma).

2.2 Encriptação de *Passwords*

Para a encriptação de passwords recorremos ao algoritmo de *Hash MD5. Quando* cria uma conta a password por este inserida é convertida pelo algoritmo MD5 e de seguida guardada na base de dados.
Sempre que este faz login a password inserida é convertida para Hash pelo algoritmo e comparada com a já guardada na base de dados.
A lógica por trás deste processo é nunca guardar as passwords na base de dados em *plaintext.*

2.3 Príncipais dificuldades

Relativamente à interface foi particularmente díficil familiarizar-mo-nos com a tecnologia Java Server Pages nomeadamente em situações nas quais relacionávamos conteúdo HTML com código Java.

3. Camada *Business*

O objetivo da camada *Business* prende-se principalmente em determinar como é que toda a informação presente na base de dados pode ser criada, guardada e possivelmente alterada.

Para a realização deste projeto criámos 3 *Enterprise Java Beans* sendo que todos estes estão implementados como sendo *Stateless Session Beans*.
A razão da nossa escolha prende-se com o facto de os stateless session bean não conservarem a ligação com o cliente e, consequentemente, não terem nenhum estado associado (i.e, *stateless*), ou seja, o *Bean* pode ser reutilizado entre vários utilizadores. Uma vez que a nossa camada business apenas é apenas utilizada para a realização de querys que passam a informação da camada Data para a camada Presentation, achámos que esta seria a opção mais adequada.

A transferência de dados da camada *Business* para a camada *Data* é feita através de *Data Transfer Objects* (DTOs) e para a camada *Presentation* é feita através de *ids*.

3.1 Serviços príncipais

ManagerEJB

- addAccount(String username, String password, String email) - cria uma conta para um administrador;
- deleteAccount(int userID) - Apaga a conta do administrador com o respetivo *id*;
- managerLoggedIn(int userID) -
- managerLoggedOut(int userID)

UserEJB

- addAcount(String username, String password, String email, String creditCard) - Cria uma conta para um username;
- deleteAccount(int userID) - Apaga a conta de um utilizador;
- editPersonalInformation(int userID, String username, String password, String email, String creditCard) - Função para editar a informação de um utilizador.
- userLoggedIn(int userID)
- userLoggedOut(int userID)

ContentEJB

- addNewContent(String title, String director, int year, String category) - Cria uma conta para um username;
- removeContent(int contentD) - Apaga um conteúdo;
- editContent(int contentID, String title, String director, String category, int year) - Função para editar a informação de um utilizador.

- removeContenFromWatchList(int contentID, int userID)

- getSuggestedContent(int userID);

4. Camada *Data*

O objetivo da camada Business prende-se principalmente em determinar como é que toda a informação presente na base de dados pode ser criada, guardada e possivelmente alterada.

Para a realização deste projeto criámos 3 Enterprise Java Beans sendo que todos estes estão implementados como sendo Stateless Session Beans.
A razão da nossa escolha prende-se com o facto de os stateless session bean não conservarem a ligação com o cliente e, consequentemente, não terem nenhum estado associado (i.e, stateless), ou seja, o Bean pode ser reutilizado entre vários utilizadores. Uma vez que a nossa camada business apenas é apenas utilizada para a realização de querys que passam a informação da camada Data para a camada Presentation, achámos que esta seria a opção mais adequada.

A transferência de dados da camada Business para a camada Data é feita através de Data Transfer Objects (DTOs) e para a camada Presentation é feita através de ids.

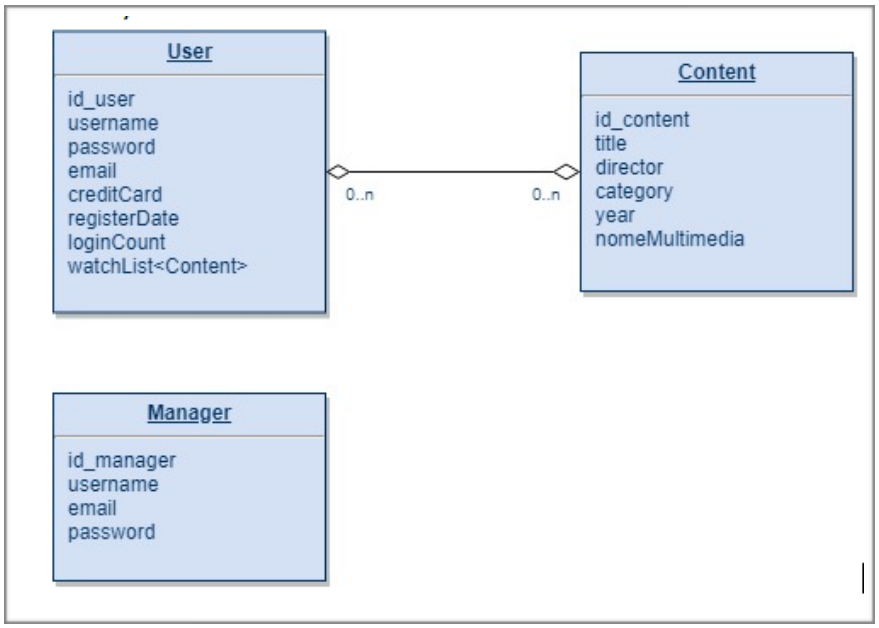

Figura 2: Diagrama ER

Como podemos observar na figura, o ER que representa as relações entre as três entidades existentes no nosso projeto é bastante simples. Na verdade, a única relação existe entre a entidade User e Content, e entende-se por "Vários Contentes podem estar associados a vários Users". Esta relação diz respeito à watch list de cada utilizador. É lógico que a entidade Manager (representa os administradores do sistema com privilégios máximos) não tenha qualquer relação com as outras entidades, visto que estes apenas existem para administrar o sistema (e.g., gerir conteúdos). Na verdade, Manager é um tipo de User com privilégios superiores, no entanto optou-se por separar estes dois tipos de utilizador por duas entidades distintas.
Tal como o User, o Manager possui um login (i.e., email e palavra passe) e um nome de utilizador. O User, pelo outro lado, possui variáveis adicionais como a data de registo, o número de vezes que efetuou login e o número de cartão de crédito. A entidade Content representa um conteúdo (i.e., filme ou série), sendo que cada conteúdo possui um título, o nome do respetivo diretor/realizador, o ano em que foi lançado e a categoria na qual se insere (e.g., drama, horror, ação, fantasia).

5. Gestão de projeto e packages

Neste projeto recorremos ao *maven* para tratar do *packaging* e da gestão de dependências, permitindo assim libertar-nos de tal trabalho. A estrutura do projeto é a seguinte:

- Projeto *parent*
    - Projeto *data*
    - Projeto *business*

- Projeto *ear*

- Projeto *web*

É de notar que o projeto *ear* serve de *placeholder*, dado que o projeto é, após a compilação, exportado para um ficheiro *.ear. O projeto *parent* serve apenas para agrupar os outros projetos.