Universidade de Coimbra

Faculdade de Ciências e Tecnologias

Mestrado em Engenharia Informática

Integração de sistemas – 2021/2022

Relatório do Trabalho 2

Three-tier Programming with Object-Relational Mapping

Coimbra, 15 de novembro de 2021

2016225648 Eurico José Pereira de Sousa uc2016225648@student.uc.pt

2017271132 Gustavo Miguel Martins Leite uc2017271132@student.uc.pt

Introdução

O problema apresentado é o desenvolvimento de uma aplicação web de gestão de uma companhia de autocarros. Os objetivos do projeto eram promover a familiarização com o desenvolvimento de aplicações de três camadas (apresentação, negócio e dados), o desenvolvimento de apps baseadas em EJBs, e o uso de JPA e JPQL e logging.

O trabalho foi desenvolvido, de acordo com as instruções do docente, com a integração do Docker com o Visual Studio Code. O Docker serviu assim de plataforma de suporte para Java 16, WildFly 24 (plataforma Jakarta EE), PostgreSQL 13 (base de dados) e Maven 3. A ORM utilizada foi o Java Persistence API a correr sobre o Hibernate visto que o JPA é o standard do Jakarta EE.

Camada de Apresentação

Layouts

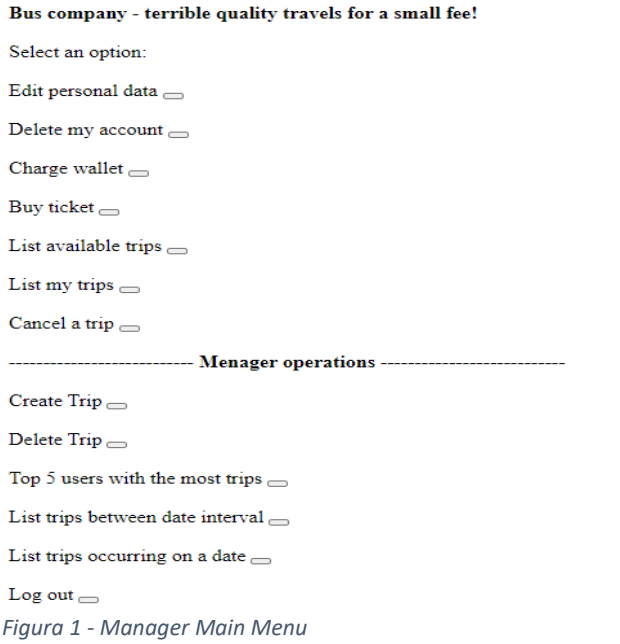

Figura 1 - Manager Main Menu

Figura 2 - Edit Data Menu

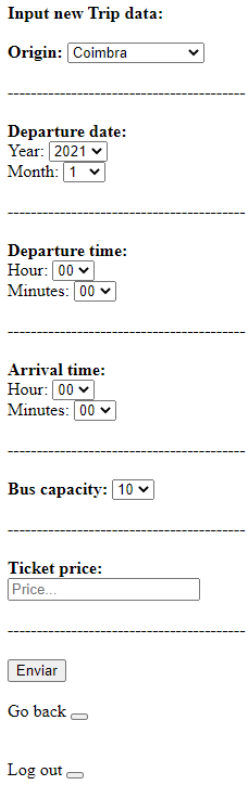

Figura 3 - Create Trip Menu

Na camada de apresentação usou-se para todas as páginas, estruturas baseadas em botões (ex: Figura 1), campos de inserção de texto (ex: Figura 2) e campos de seleção com opções pré-definidas na camada de negócio, com dropdown list (ex: Figura 3).

Nesta camada não houve nenhum aspeto que tenha sido complexo de implementar, sendo que a maior dificuldade encontrada foi aprender a sintaxe de JTLS.

Em relação à password nesta camada, é recebida através numa caixa de inserção de texto, após inserção pelo utilizador, sendo depois enviada para um servlet usando o método doGet(), pois o método doPost(), que seria o mais correto por não mostrar a pass no link, não se apresentava funcional no servlet (Authentication.java). Daqui a password é enviada para o método authentication() da camada de negócio que trata de fazer a verificação da password comparando-a com a que está na base de dados associada ao mail inserido.

Para além disto, as passwords são encriptadas através do método generateStrongPasswordHash() que usa o algoritmo PBKDF2WithHmacSHA1 para fazer hashing das palavras chave no processo de registo de um novo utilizador.

Para fazer a distinção entre, utilizadores manager e utilizadores normais, e utilizadores autenticados e não autenticados, usou-se um método doFilter() que faz estas filtragens com recurso a um atributo chamado "auth" no caso da autenticação e "managerAuth" no caso da distinção do tipo de utilizadores.

Camada de Negócio

Descrição dos EJBs

Em relação à camada de Negócio, foram criados três beans, sendo eles o IManageSystems.java que corresponde à interface dos métodos presentes no bean ManageSystem.java, encarregues de executar as funcionalidades especificadas no enunciado, e o ScheduleMailSender.java que está encarregue de fazer o envio de emails periódicos aos managers do sistema coma informação da rentabilidade diária.

Quanto ao bean ManageSystem.java, que contém os métodos de manipulação de dados recebidos na camada de apresentação assim como nos dados e entidades obtidos a partir da base de dados, definiu-se como stateless porque o sistema corre no localhost, sendo a interação entre clientes e sistema feita à vez, não havendo necessidade de criar múltiplas sessões separadas para vários clientes a interagir em simultâneo com a aplicação. Situação que não irá ocorrer no contexto apresentado no enunciado.

O tipo de interface definiu-se como local visto que não se está a usar um servidor remoto, mas sim uma aplicação corrida no localhost.

Quanto às transações, usou-se Typed Queries quando havia necessidade de ir buscar listas de entidades à base de dados, e Queries quando era necessário ir buscar valores isolados. Para ir buscar entidades específicas para executar nelas operações nelas de delete e update, usou-se o método find do entity manager.

Detalhes do Funcionamento da Camada de Negócios

Os métodos encarregues de executar as funcionalidades especificadas no enunciado, comunicam com camada de apresentação recebendo objetos do tipo String ou Integer como parâmetros, e devolvem sempre objetos do tipo String, List<String>, List< List<String>> ou Integer, nunca sendo passadas entre estas duas camadas entidades persistidas na base de

dados. Estes objetos representam diversos componentes necessários para o funcionamento da aplicação como datas, mails, nomes, passwords, listas de viagens, listas de bilhetes, listas de utilizadores e valores do tipo double para o carregamento da carteira da conta.

À camada de dados, a camada de negócio vai buscar entidades, listas de entidades ou valores isolados, como referenciado na descrição dos EJBs.

Camada de Dados

Diagrama ER

No nosso sistema temos 5 entidades: MyUser, Trip, Ticket, RegistUsers e Locations.

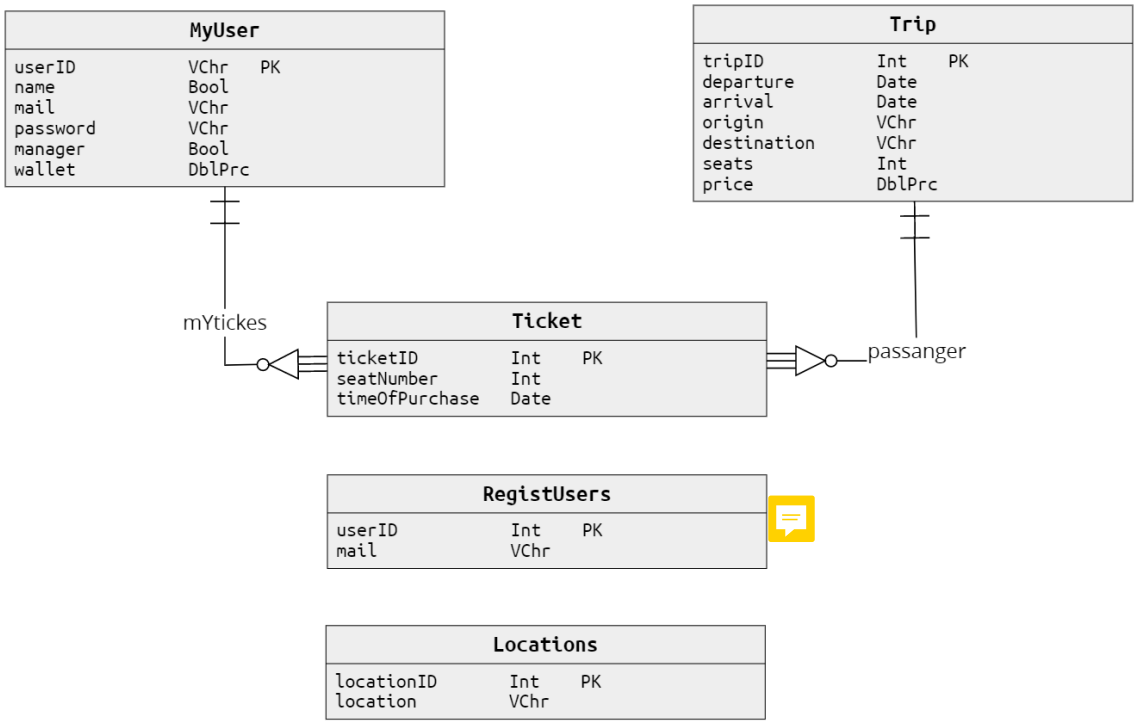

Figura 4 – Diagrama ER (VChr equivale a String, DblPrc corresponde a double e PK corresponde a Primary Key)

Packaging e Gestão do Projeto

Descrição do Package da App

O package do projeto segue a configuração descrita no capitulo 8 do livro fornecido pelo docente, sendo que a única dependência extra adicionada foi org.slf4j, no pom.xml do módulo do ejbs e web, para que ser possível a utilização do logger.

Estrutura do projeto

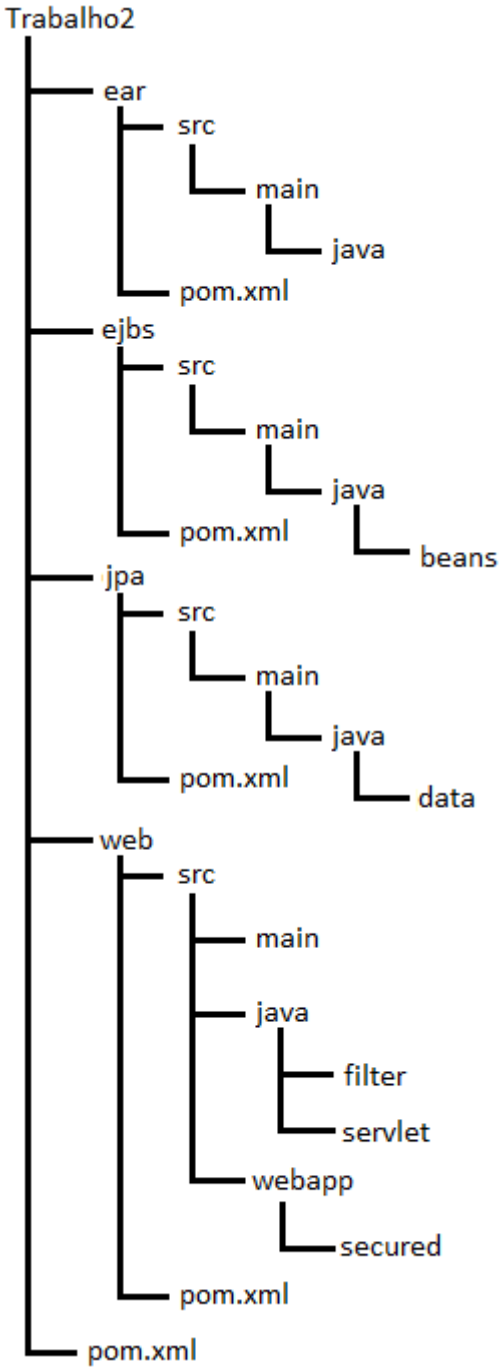

Figura 5 - Estrutura do Projeto

Notas Finais

- O início de sessão com um manager criado pelo script diretamente na base de dados, por vezes pode dar acesso negado sem razão aparente, uma forma de contornar este aspeto é criar um utilizador normal e depois, na base de dados mudá-lo para manager.

Referências

https://howtodoinjava.com/java/java-security/how-to-generate-secure-password-hash-md5-sha-pbkdf2-bcrypt-examples/

"Jakarta EE in Practice" por Filipe Araújo, Nuno Laranjeiro