|  |  |
| --- | --- |
|  |  |

**Licenciatura em Engenharia Informática**

Escola Superior de Tecnologia e Gestão

*Instituto Politécnico de Viana do Castelo*

Projeto 2

Web + Acesso à BD

Gestão de um ginásio

Alexandre Santos - 24585

Sofia Sousa - 23435

2022/2023

Sumário

[1. Introdução 3](#_Toc137076257)

[2. Modelo Casos de Uso 3](#_Toc137076258)

[3. Modelo Físico (Base de Dados) 4](#_Toc137076259)

[4. Acesso à base de dados 5](#_Toc137076260)

[5. Aplicação Desktop 7](#_Toc137076261)

[5.1. Login 8](#_Toc137076262)

[5.2. Administrador 9](#_Toc137076263)

[5.3. Rececionista 11](#_Toc137076264)

[5.4. Instrutor 12](#_Toc137076265)

[6. Conclusão 14](#_Toc137076266)

[7. Bibliografia 15](#_Toc137076267)

# Introdução

Com este relatório, será apresentado o tema do nosso projeto, a estrutura do projeto, requisitos e atores. Vamos ainda incluir o diagrama final da base de dados após várias alterações ao longo do desenvolvimento do projeto. Por fim, incluímos a estrutura e implementação da nossa aplicação desktop em JavaFX.

# Modelo Casos de Uso

No modelo de casos de uso temos 4 atores:

* Administrador, cria utilizadores de todo o tipo e tem possibilidade de listar outros dados existentes no sistema como subscrições e planos de treino;
* Instrutor, pode gerir aulas de grupo e planos de treino;
* Rececionista, gere clientes, subscrições e pagamentos;
* Cliente, consegue ver e editar a sua conta, consultar a sua subscrição, aulas em que se inscreveu ou aulas que irão acontecer para se inscrever e por fim os planos de treino que já teve e que tem atualmente.

Uma imagem com diagrama

Descrição gerada automaticamente

# Modelo Físico (Base de Dados)

Uma imagem com texto, captura de ecrã

Descrição gerada automaticamenteA base de dados final, usada para armazenar toda a informação e relações entre entidades que são a base de funcionamento da aplicação, pode ser analisada na figura seguinte.

# 4. Acesso à base de dados

Para aceder à base de dados utilizamos as ferramentas *Hibernate* e *JPA*.

Começamos por criar um projeto vazio em *Maven*, onde criamos o ficheiro “*pom.xml*” que guarda informação sobre todas as dependências necessárias para o projeto, no nosso caso, é o *Hibernate* e os drivers para conexão com o *PostgreSQL*. De seguida criamos dentro da pasta “*resources*” o package “*META-INF*”, que deverá conter o ficheiro “*persistence.xml*”. Este ficheiro contém toda a informação relacionada com a conexão à base de dados, incluindo o nome da unidade do *persistence* que usamos para realizar todo o tipo de consultas e atualizações na base de dados.

Uma imagem com texto, captura de ecrã, software, Software de multimédia

Descrição gerada automaticamente

No acesso da base de dados, temos dois packages que contem a definição d base de dados e métodos básicos de CRUD.

O primeiro é o “*repository*”, onde são guardadas as entidades da base de dados, que correspondem às tabelas na BD. Em cada ficheiro são definidos os campos das tabelas, as relações e as restrições como chaves primárias, valores gerados automaticamente, etc.

De seguida, temos os ficheiros “*DAO*”, estes ficheiros vão ter os métodos principais para o *CRUD*, as *queries* de consulta, inserção e atualização de dados.

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamenteUma imagem com texto, captura de ecrã

Descrição gerada automaticamente

# 5. Aplicação Web

A aplicação web, foi desenvolvida com Java Spring Web MVC Thymleaf. Com isto, criámos dois tipos de ficheiros:

* No diretório “*com.example”* estão os três controllers que conseguimos desenvolver, estes são classes java que controlam o comportamento da aplicação;
* No package “*resources”* encontram-se os ficheiros html contém o código de desenho e configuração do aspeto das interfaces.

Uma imagem com texto, captura de ecrã, software

Descrição gerada automaticamente

## 5.1. Login

Para o login, o utilizador usa o seu email (único para cada utilizador) e password.

Uma imagem com texto, captura de ecrã, software, Sistema operativo

Descrição gerada automaticamente

## 5.2. Dashboard

Uma imagem com texto, captura de ecrã, file, software

Descrição gerada automaticamentePara o Administrador criámos um dashboard, onde pode ver e editar o próprio perfil, criar, editar e apagar funcionários e clientes. Por fim, tem outra secção com 4 botões para aceder às listagens, aqui são abertas páginas externas para cada listagem como será possível ver nas imagens seguintes:

# 6. Conclusão

Agora que estamos no final, ficamos com pena de não ter conseguido desenvolver a parte web, apenas as páginas principais ficaram desenvolvidas sem funcionalidades. Passámos muito tempo a desenvolver a versão desktop da aplicação e o acesso á base de dados, e isto deixou-nos quase nenhum tempo para desenvolver aa fase final. Apesar disso conseguimos aprender alguns conceitos básicos da tecnologia SpringMVC.

# 7. Bibliografia

<https://www.jetbrains.com/help/idea/ddl-data-sources.html#configuring-ddl-generation-settings>

<https://www.javaguides.net/2020/07/three-tier-three-layer-architecture-in-spring-mvc-web-application.html>

<https://www.jetbrains.com/datagrip/features/generation.html>

<https://cezarcruz.com.br/como-exportar-jar-utilizando-intellij/>

<https://www.geeksforgeeks.org/how-to-add-external-jar-file-to-an-intellij-idea-project/>

<https://www.jetbrains.com/help/idea/persistence-tool-window.html#generate_entities_and_mappings>

<https://www.geeksforgeeks.org/how-to-add-external-jar-file-to-an-intellij-idea-project/>