

**Trabalho prático**

**Sistemas de Informação**

**Fase 2**

|  |  |
| --- | --- |
| 47224 | André Graça |
| 49149  48459 | Diogo Guerra  Diogo Santos |

|  |  |
| --- | --- |
| docente | Walter Vieira |
|  |  |

Relatório final realizado no âmbito do trabalho prático de Sistemas de informação, do

curso de licenciatura em Engenharia Informática e de Computadores  
Semestre de Verão 2022/2023

Junho de 2023

# Resumo

Texto do resumo.

Breve descrição do projecto, dos resultados importantes e das conclusões: o objectivo é dar ao leitor uma visão global do projecto (não deve exceder uma página).

**Palavras-chave:** lista de palavras-chave, ordenadas alfabeticamente, separadas por ;.

# Abstract

Abstract text (1 page).

**Keywords:** sorted keyword list, delimited by ;.

**Índice**

[Resumo v](#_Toc417484088)

[Abstract vii](#_Toc417484089)

[Agradecimentos ix](#_Toc417484090)

[Lista de Figuras xiii](#_Toc417484091)

[Lista de Tabelas xv](#_Toc417484092)

[1. Introdução 1](#_Toc417484093)

[1.1 Nome da secção deste capítulo 1](#_Toc417484094)

[1.2 A segunda secção deste capítulo 1](#_Toc417484095)

[1.2.1 A primeira sub-secção desta secção 1](#_Toc417484096)

[1.2.2 A segunda sub-secção desta secção 1](#_Toc417484097)

[1.3 Organização do documento 2](#_Toc417484098)

[2. Formulação do Problema 3](#_Toc417484099)

[2.1 Nome da secção deste capítulo 3](#_Toc417484100)

[2.2 Análise do problema - enumeração 3](#_Toc417484101)

[2.3 Outro problema - tabela 4](#_Toc417484102)

[2.4 Expressões matemáticas 4](#_Toc417484103)

[2.5 Figuras de grande dimensão 4](#_Toc417484104)

[3. Solução Proposta - Grandes Ideias 7](#_Toc417484105)

[3.1 Nome da primeira secção deste capítulo 7](#_Toc417484106)

[3.2 A segunda secção deste capítulo 8](#_Toc417484107)

[3.2.1 A primeira sub-secção desta secção 8](#_Toc417484108)

[3.2.2 A segunda sub-secção desta secção 8](#_Toc417484109)

[3.3 Descrição detalhada da solução 8](#_Toc417484110)

[4. Avaliação Experimental 11](#_Toc417484111)

[4.1 Nome da primeira secção deste capítulo 11](#_Toc417484112)

[4.2 A segunda secção deste capítulo 11](#_Toc417484113)

[4.2.1 A primeira sub-secção desta secção 11](#_Toc417484114)

[4.2.2 A segunda sub-secção desta secção 11](#_Toc417484115)

[4.3 Análise de resultados 11](#_Toc417484116)

[5. Conclusões 13](#_Toc417484117)

[Referências 14](#_Toc417484118)

[A.1 Diagramas da Aplicação 15](#_Toc417484119)

[A.2 Modelos de dados 17](#_Toc417484120)

# 

# Lista de Figuras

[Figura 1 - Legenda da figura com o logotipo do ISEL. 1](#_Toc416101905)

[Figura 2 - Legenda da figura com o logotipo do ISEL – versão 2. 5](#_Toc416101906)

[Figura 3 – Diagrama de casos de utilização. 15](#_Toc416101907)

# Lista de Tabelas

[Tabela 1 -Um exemplo de legenda de tabela. Prazos de entrega de Projecto e Seminário, 4](#_Toc416101908)

# Introdução

# Camada de acesso a dados

Nesta secção, iremos explorar os métodos adotados para a construção de uma cada de acesso a dados. Iremos também abordar a construção das entidades em JPA.

## 2.1 Implementação do Mapper

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

Descrição gerada automaticamentePara a construção da camada de acesso a dados, optamos primeiramente pela implementação de um *Mapper* genérico. A interface Mapper tem métodos equivalentes às operações CRUD que são a create, read, update e delete para manipulação de dados.

Figura 1 - Interface Mapper

A implementação desta interface foi realizada de forma genérica, pois assim não é necessário a implementação de *Mappers*, igual ao número de entidades, visto que a class que implementa esta interface tem natureza genérica, sempre que houver necessidade de criar um Mapper, basta fornecer á class Mapper o tipo da chave primária da entidade e o tipo da entidade.

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

Descrição gerada automaticamente

Figura 2 - Implementação da class Mapper

Importante notar que na figura 2 apenas está exposto a implementação do método “create”, todos os 4 métodos foram implementados de forma semelhante. As variáveis *tClass* e *tIdclass* são o tipo da entidade e o tipo da chave primária que são passados ao construtor do Mapper.

## 2.2 Implementação do Repository

## 2.3 Implementação das entidades em JPA

Nesta secção, iremos observar como converter as tabelas da base de dados em entidades do JPA. Relações de grau 1:1 não serão abordadas pois o nosso modelo EA não possui nenhuma.

### Entidades

Uma imagem com texto, captura de ecrã, diagrama, círculo

Descrição gerada automaticamentePara a conversão de entidades do Modelo EA para entidades do JPA, é necessário para cada entidade criar uma class com fields equivalentes às suas colunas, e usar anotações que ajudem a associar as colunas na base de dados aos fields da class.

Figura 3 - Entidade Jogo JPA.

Na figura 3, está representado do lado direito a entidade **Jogo** do modelo EA, que tem 3 atributos id, nome e URL, sendo nome a chave primária. No lado direito, temos o código java equivalente a entidade **Jogo**, a class tem anotações como:

- **@Entity** -> Indica que representa uma entidade.

- **@Table**(name, schema) -> Indica que tabela esta entidade representa, dado o nome da tabela e o schema onde esta se encontra.

- **@Id** -> Indica qual dos fields representa a chave primária.

- **@Column**(name) -> Indica que coluna da tabela o field representa, dado o nome da coluna, esta anotação também recebe outros valores para manter total coerência com a base dados, isto é, outras restrições que possam ter sido impostas á coluna.

Para cada um destes fields também deve ser criado um getter e um setter para fornecer a possibilidade de aceder e alterar o valor durante a execução da aplicação.

No caso de a chave primária ser composta por mais do que um atributo de deve ser feito uma class que representa essa chave, com fields que representam as colunas que formam a chave e getters e setters. Na class deve ser criado um field que é do tipo da chave primária.

### Entidades Fracas e relações de grau 1:N

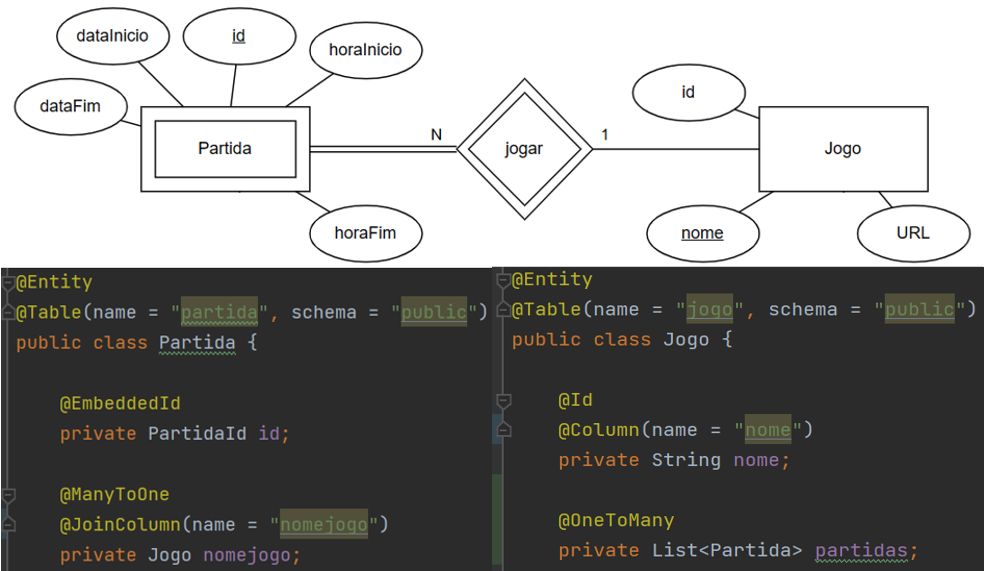
As entidades fracas do modelo EA quando convertidas para JPA, tem de ser indicado a usa dependência através das anotações do JPA. Visto que as relações de grau 1:N e entidades fracas seguem as mesmas estratégias de conversão, serão ambas abordadas nesta secção.

Figura 4 - Entidade Fraca Partida JPA

Na figura 4, podemos observar que partida tem um field do tipo **Jogo** com a anotação **@ManyToOne** que indica que o “N” está do lado da entidade **Partida**. Na class **Jogo** iremos ter uma lista de Partida com a anotação **@OneToMany** que representa que o jogo pode ter várias partidas. A class **Partida** irá conter, tal como mencionado anteriormente, fields equivalentes às colunas da tabela **Partida**, com getters e setters. Isto também se aplica a relações de grau 1:N.

### Relações de grau N:N

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

Descrição gerada automaticamente

Na figura 5, podemos observar que ambas as entidades **Jogador** e **Jogo** tem um field com a anotação **@ManyToMany** que indica a relação de N:N e este tem o tipo da entidade do qual partilha essa relação. Na class que representa a relação N:N, tem de ser colocada a anotação **@ManyToOne** no field que faz referência á entidades que usufrui desta relação, logo, a class **Comprar** irá ter 2 fields, um que referencia **Jogador** e outro que referencia **Jogo**. Ambos terão as anotações **@ManyToOne** e **@JoinColumn** para indicar qual coluna é que esta ser feita a referencia.

# Acesso às funcionalidades da BD

Nesta secção iremos apresentar as técnicas usadas para acesso às funcionalidades da base de dados desenvolvidas na fase anterior do trabalho prático.

## 3.1 Registo dos parâmetros das funcionalidades

## 3.2 Execução de funções

## 3.3 Execução de procedimentos armazenados

# Controlo de concorrência

## 4.1 Controlo de concorrência otimista

## 4.2 Controlo de concorrência pessimista

## 4.3 Análise de resultados

# Conclusões