# Sistema de Simulação de Campeonatos de Automobilismo

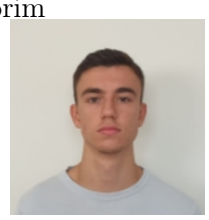
## Trabalho realizado por:


Afonso Laureano Barros Amorim
Beatriz Ribeiro Monteiro
Bianca Araújo do Vale
Daniel José Silva Furtado
Telmo José Pereira Maciel

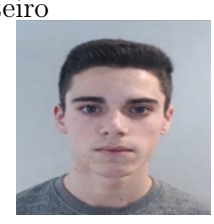

A97569
Afonso Amorim

A95437
Beatriz Monteiro

A95835
Bianca Vale



A97327
Daniel Furtado



A96569
Telmo Maciel

Fase 3
Grupo 5
https://github.com/5ditto/DSS-PROJECT.git
Desenvolvimento de Sistemas de Software 2022/2023
Universidade do Minho

# Índice



# Índice de Imagens

# 1 Introdução

Este relatório destina-se à apresentação e exposição do trabalho desenvolvido na terceira e última fase do projeto desenvolvido no âmbito da Unidade Curricular de *Desenvolvimento de Sistemas de Software*.

Este projeto consiste na implementação de um sistema de Simulação de Campeonatos de Automobilismo similar ao *F1 Manager*.

Nas fases anteriores tivemos de desenvolver diferentes diagramas necessários para o desenvolvimento desta aplicação. Entre os diagramas desenvolvidos podemos destacar o Modelo Domínio, o Diagrama de Classes, o Diagrama de *Use Cases* e alguns Diagramas de Sequência que, na nossa opinião, são fulcrais para o desenvolvimento deste projeto.

# 2 Alterações às fases anteriores

Ao prepararmos o desenvolvimento desta última fase do projeto apercebemo-nos de erros que foram cometidos ao longo das duas fases anteriores.

Para esta fase tivemos de adaptar o diagrama de classes para suportar os DAOS e, porque, ao começarmos a implementar as classes apercebemo-nos que haviam inconsistencias no diagrama que desenvolvemos na fase anterior.

## 2.1 Diagrama de Classes

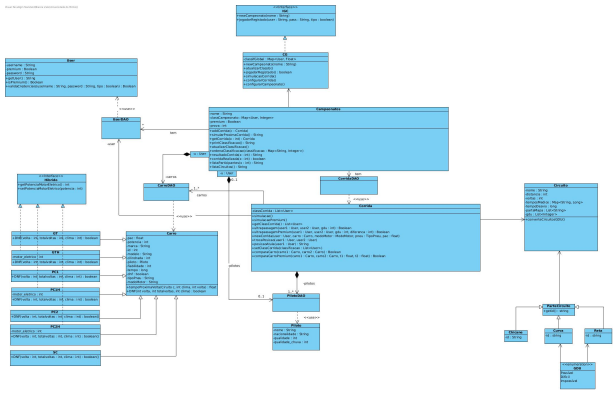


Figure 1: Diagrama de Classes

# 3 Base de Dados

## 3.1 Modelo Lógico

Tendo por base o Diagrama de Classes, os *Use Cases* e os Diagramas de Sequência, construímos a camada Lógica de Negócios do nosso projeto. A camada de Negócos é responsável por fazer a ponte entre a camada de dados(DAO) e a interface (UI). Assim os dados são processados de forma consistente e correta.

## 3.2 Povoamento da Base de Dados

O povamento da nossa Base de Dados é feito localmente e cada utilizador consegue povoar a Base de Dados. O objetivo era guardar a informação localmente para que não houvesse perda de memória. A base de dados é baseada nos DAO's realizados que criam tabelas e seriam povoadas automaticamente.

# 4 Descrição da Implementação

Com a implementação dos DAO's a nossa forma de trabalhar e pensar para realizar e implementar o trabalho teve que mudar. Para conseguir implementar uma base de dados tivemos que alterar bastante algumas partes do código. O projeto está dividido em camadas. Exsite uma camada de dados na qual está implementada a base de dados que está encapsulada e persistente para que assim não haja perda de dados e informação.

Para implementar a base de dados tivemos que implemenar vários ficheiros DAO's com várias queries que realizam operações na base de dados. As queries realizadas são pedidas pela camada de Lógica de Negócios de forma a encapsular assim a informação.

```
    private PilotoDAO(){
        try(Connection conn = DriverManager.getConnection(DataBaseConfig.URL,
DataBaseConfig.USERNAME, DataBaseConfig.PASSWORD);
            Statement stm = conn.createStatement()){

            String sql =    "CREATE TABLE IF NOT EXISTS Pilotos (" +
                            "Nome varchar(100) NOT NULL PRIMARY KEY," +
                            "Niveis_pericia varchar(45) DEFAULT NULL," +
                            "CTS decimal(2,1) DEFAULT NULL," +
                            "SVA decimal(2,1) DEFAULT NULL";
            stm.executeUpdate(sql);

        }catch (SQLException e){
            // Erro a criar tabela
            e.printStackTrace();
            throw new NullPointerException(e.getMessage());
        }
    }
```

Figure 2: Exemplo da criação da tabela de Piloto

```
public class CarroDAO implements Map<Integer, Carro> {

    private static CarroDAO singleton = null;

    private CarroDAO(){
        try(Connection conn = DriverManager.getConnection(DataBaseConfig.URL,
DataBaseConfig.USERNAME, DataBaseConfig.PASSWORD);
            Statement stm = conn.createStatement()){

            String sql =    "CREATE TABLE IF NOT EXISTS Carros (" +
                            "ID int NOT NULL PRIMARY KEY"+
                            "Modelo varchar(50) NOT NULL ," +
                            "Marca varchar(50) NOT NULL ," +
                            "cilindrada int NOT NULL "+
                            "fiabilidade int DEFAULT NULL" +
                            "potencia int DEFAULT NULL," +
                            "tipoPneu varchar(5) DEFAULT NULL," +
                            "modoMotor varchar(5) DEFAULT NULL," +
                            "CONSTAINT CheckTipoPneu CHECK(tipoPneu in ('macio', 'duro', 'chuva')";

            stm.executeUpdate(sql);
            //CONSTRAINT `MarcaModelo` PRIMARY KEY (`marca`, `modelo`),
                        //CONSTRAINT `MarcaModelo` PRIMARY KEY (`marca`, `modelo`),
                        //CONSTRAINT `CheckTipo` CHECK(`tipoCarro` IN ('C1', 'C1H', 'C2', 'C2H',
'GT', 'GTH', 'SC')),
        }catch (SQLException e){
            // Erro a criar tabela
            e.printStackTrace();
            throw new NullPointerException(e.getMessage());
        }
    }
```

Figure 3: Exemplo da criação da tabela do Carro

# 5 Conclusão

O trabalho da Unidade Curricular de Desenvolvimento de Sistemas de Software permitiu-nos um maior conhecimento e aprendizagem, a nível individual e de grupo, acerca dos diferentes modelos e diagramas, bem como a implementação do código e da base de dados.

Na primeira fase do presente trabalho, foi realizada, pelo grupo, a análise de requisitos do projeto. De modo a obtermos uma melhor interpretação das entidades relevantes do mesmo e as relações que estabelecem entre elas, procedemos à elaboração do Modelo de Domínio. Após a consolidação deste Modelo, procedemos à constatação dos requisitos funcionais e, para tal, realizamos o Modelo de *Use Cases*. Assim, conseguimos perceber quais as ações que os administradores ou jogadores iam desempenhar no trabalho.

Na fase dois, de modo a otimizar o nosso trabalho, procedemos à alteração de alguns *Use Cases* previamente apresentados. Nesta etapa, foi-nos pedida a realização de diversos diagramas: Diagrama de Componentes, Diagrama de Classes, Diagrama de Packages e Diagrama de Sequência para alguns *Use Cases*. Estes permitiram-nos uma elaboração e organização a um nível mais avançado de todas as funcionalidades para a nossa aplicação.

No decorrer desta última fase, apercebemo-nos de algumas falhas no nosso Diagrama de Classes e Diagrama de Sequência e, portanto, procedemos à alteração das mesmas de maneira a alcançarmos um projeto eficaz e funcional.

Em jeito de conclusão, o facto de já termos implementado os diagramas previamente permitiu-nos um avanço e um planeamento, facilitando, assim, a correção de erros. Deste modo, com este trabalho, apesar de termos passado por algumas dificuldades tais como a implementação da ligação da base de dados à camada de negócios e as alterações que fizemos no código e no Diagrama de Classes. Assim, finalizado o projeto e apesar dos diversos precalços, o grupo está satisfeito com o resultado final, considerando ir de encontro ao que foi solicitado por parte dos docentes da Unidade Curricular.