

SPAD

RELATÓRIO

TPSIP\_10.17

Bruno Ferreira

João Santos

Luís Passeira

Outubro de 2018

# Índice

[Índice 3](#_Toc493864898)

[1. Introdução 5](#_Toc493864899)

[1.1 Enquadramento 5](#_Toc493864900)

[1.2 Objetivos a Cumprir 5](#_Toc493864901)

[1.3 Organização do Relatório 5](#_Toc493864902)

[2 Análise de Sistemas 7](#_Toc493864903)

[2.1 Requisitos Funcionais 7](#_Toc493864904)

[2.2 Requisitos Não Funcionais 7](#_Toc493864905)

[*2.3* Diagrama de *Use Cases* 7](#_Toc493864906)

[2.4 *Use Case XPTO – Descrição Estruturada* 7](#_Toc493864907)

[2.5 Diagrama de Classes 7](#_Toc493864908)

[*2.6* Diagrama de Sequência para o *Use Case XPTO* 7](#_Toc493864909)

[2.7 Fluxograma para o *Use Case XPTO* 7](#_Toc493864910)

[*2.8* Diagrama de Fluxo de Dados para o Processo XPTO 7](#_Toc493864911)

[2.9 Diagrama E-R 7](#_Toc493864912)

[2.10 Modelo de Dados 7](#_Toc493864913)

[2.11 Dicionário de Dados 7](#_Toc493864914)

[3 Desenvolvimento da Aplicação 9](#_Toc493864915)

[3.1 Arquitetura da Aplicação 9](#_Toc493864916)

[3.2 Tecnologias 9](#_Toc493864917)

[*3.3* *Pontos Importantes* 9](#_Toc493864918)

[3.4 Descrição do uso da aplicação e suas funcionalidades 9](#_Toc493864919)

[4 Conclusão 11](#_Toc493864920)

[4.1 Objectivos Cumpridos 11](#_Toc493864921)

[4.2 Contratempos 11](#_Toc493864922)

[4.3 Dificuldades Encontradas 11](#_Toc493864923)

[4.4 Apreciação pessoal 11](#_Toc493864924)

[5 Referências 13](#_Toc493864925)

[6 Anexos 15](#_Toc493864926)

# Introdução

Com o advento da internet surgiu também um mundo sem fim de novas oportunidades.

Entre elas a disponibilização de serviços on-line de casas de apostas de futebol, as quais se têm multiplicado nos últimos anos.

O homem, é, historicamente um ser atraído pelo jogo e pelo risco, consequentemente o mercado das apostas desportivas online, é um mercado que vem crescendo consistentemente ao longo do tempo.

Aliado ao facto do forte investimento que estas casas de apostas realizam em publicidade, juntamente com a cada vez maior facilidade de acesso à internet que as pessoas em todo mundo possuem, é de prever que este mercado continue a crescer nos próximos anos.

Atualmente existem milhões de pessoas em todo o mundo que realizam apostas desportivas online diariamente, a grande maioria sem conhecimento nem ferramentas adequadas para realizar uma aposta desportiva consciente.

Assim fica claro que este facto representa uma oportunidade de negócio ainda pouco explorada, gerada pelo mercado “principal”, o mercado das apostas desportivas online.

O projeto SPAD é, portanto, uma iniciativa de criação de um serviço online baseado em análise estatística, de prognósticos de apostas desportivas, que visa fornecer aos apostadores online, uma ferramenta que os ajude a apostar de forma mais consciente e consistente.

## Enquadramento

SPAD ou Sistema de Prognósticos de Apostas Desportivas, enquadra-se no âmbito do Projeto Final do curso TPSIP\_10.17.

## Objetivos a cumprir

Entre os vários objetivos que este projeto se propõe a alcançar, destacam-se os seguintes:

- Obter dados de eventos desportivos de futebol de forma automática (Web-API).

- Validar a fidedignidade dos dados obtidos através da Web-API.

- Persistir os dados no nosso sistema (Base de dados MS SQL Server).

- Aplicar algoritmos estatísticos sobre os dados recolhidos

- Gerar prognósticos para futuros eventos desportivos de futebol.

- Guardar o histórico dos prognósticos gerados pelo sistema.

- Permitir o registo de utilizadores no sistema e a sua gestão.

- Permitir a criação e envio de newsletter para utilizadores registados.

- Permitir a consulta do histórico dos prognósticos gerados pelo sistema.

- Criar duas Aplicações Web (Front Office; Back Office) que permitam a automatização e a fácil gestão de todo o sistema.

- Garantir o acesso ao Front Office através de dispositivos móveis.

## Organização do Relatório

Nota: deve descrever-se a estrutura do relatório em termos de capítulos principais, explicando o que se vai abordar em cada um.

# Análise de Sistemas.

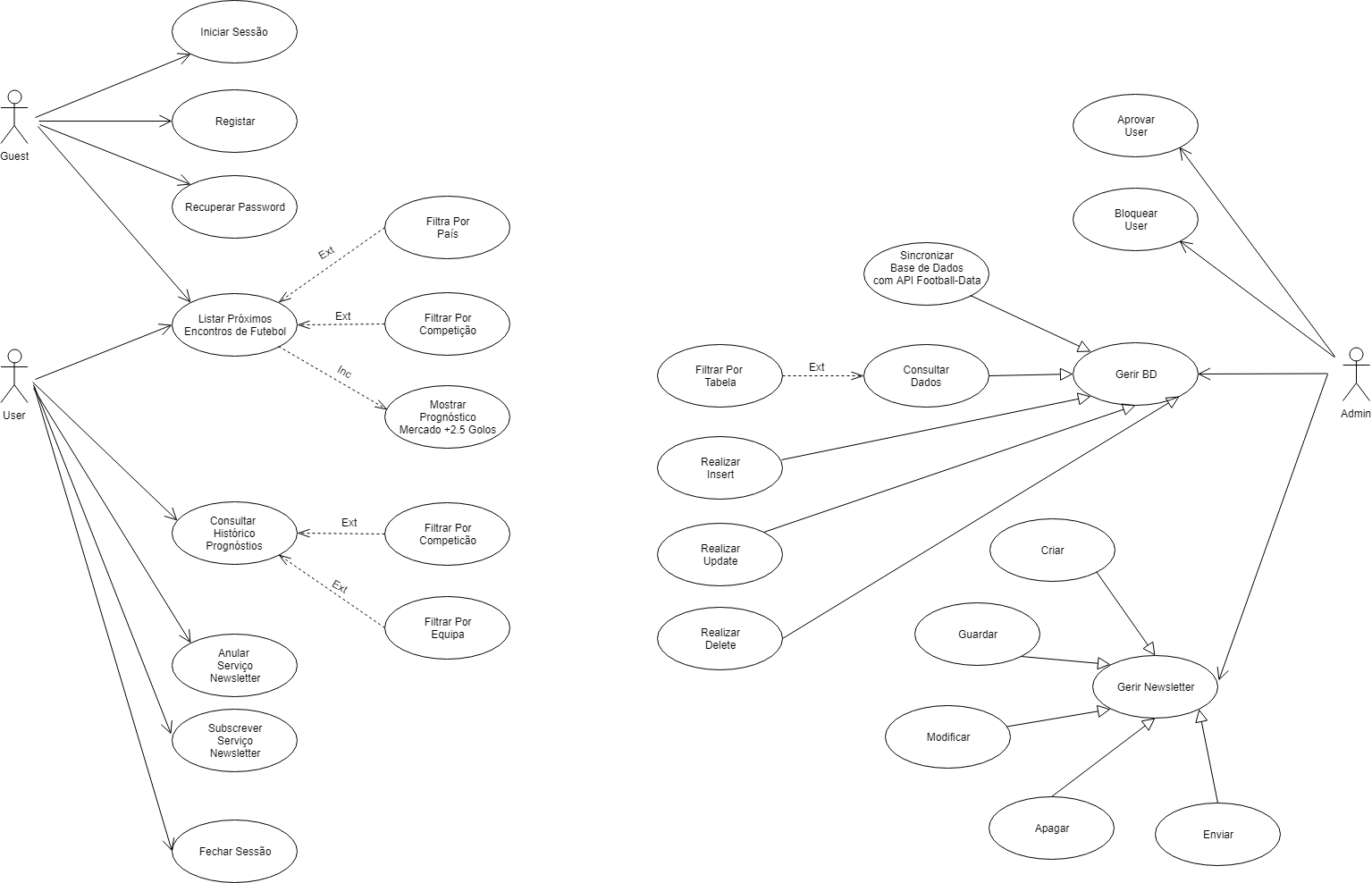
## Requisitos Funcionais

* Cada Utilizador deve poder fazer registo.
* Cada Utilizador deve poder iniciar sessão e terminar a mesma, caso já a tenha iniciado.
* Cada Utilizador deve poder recuperar a password da sua conta, caso não tenha sessão iniciada.
* Cada Utilizador pode guardar uma fotografia associada ao seu perfil.
* Um Utilizador não autenticado (sem sessão iniciada) pode ver a listagem de encontros de futebol e pode também fazer filtragem por competição e/ou equipa, mas não têm acesso aos prognósticos disponíveis no serviço.
* Um Utilizador autenticado (com sessão iniciada) tem acesso a todas as funcionalidades anteriormente mencionadas, e também à subscrição da Newsletter.
* Cada utilizador deve confirmar o e-mail após o registo para que um administrador o possa aprovar. Depois de aprovado, o utilizador consegue ter acesso às funcionalidades do sistema.
* Cada Administrador pode aprovar Utilizadores.
* Cada Administrador pode bloquear Utilizadores.
* Cada Administrador pode gerir a base de dados, de acordo com as seguintes ações:
  + Sincronizar com a API;
  + Filtrar dados;
  + Introduzir dados;
  + Modificar dados;
  + Eliminar dados.
* Cada Administrador pode gerir o serviço de Newsletter, de acordo com as seguintes ações:
  + Criar Newsletter;
  + Guardar Newsletter na base de dados;
  + Modificar Newsletter;
  + Apagar Newsletter;
  + Enviar Newsletter.

## Requisitos Não Funcionais

* A Base de Dados deve ser desenvolvida em *Transact-SQL*.
* A aplicação deve seguir uma arquitetura N-TIER de 3 camadas (*User Interface, Business Logic Layer e Data Access Layer*).
* Deve ser utilizada a programação orientada a objetos (*Object-oriented Programming*).

## Diagrama de Use Cases



## Use Case Listar Próximos Encontros – Descrição Estruturada

Pré-Condição:

O dispositivo deve ter conexão à internet.

Descrição:

1. O Utilizador seleciona uma Liga.
2. Apenas o botão da liga selecionada é que fica destacado.
3. O sistema procura a próxima partida de cada equipa distinta dessa liga.
4. O sistema vai obter os prognósticos correspondentes às partidas obtidas.

Pós-Condição:

Os jogos são apresentados com os respetivos prognósticos, com indicação de “Bet/No Bet” no mercado 2,5.

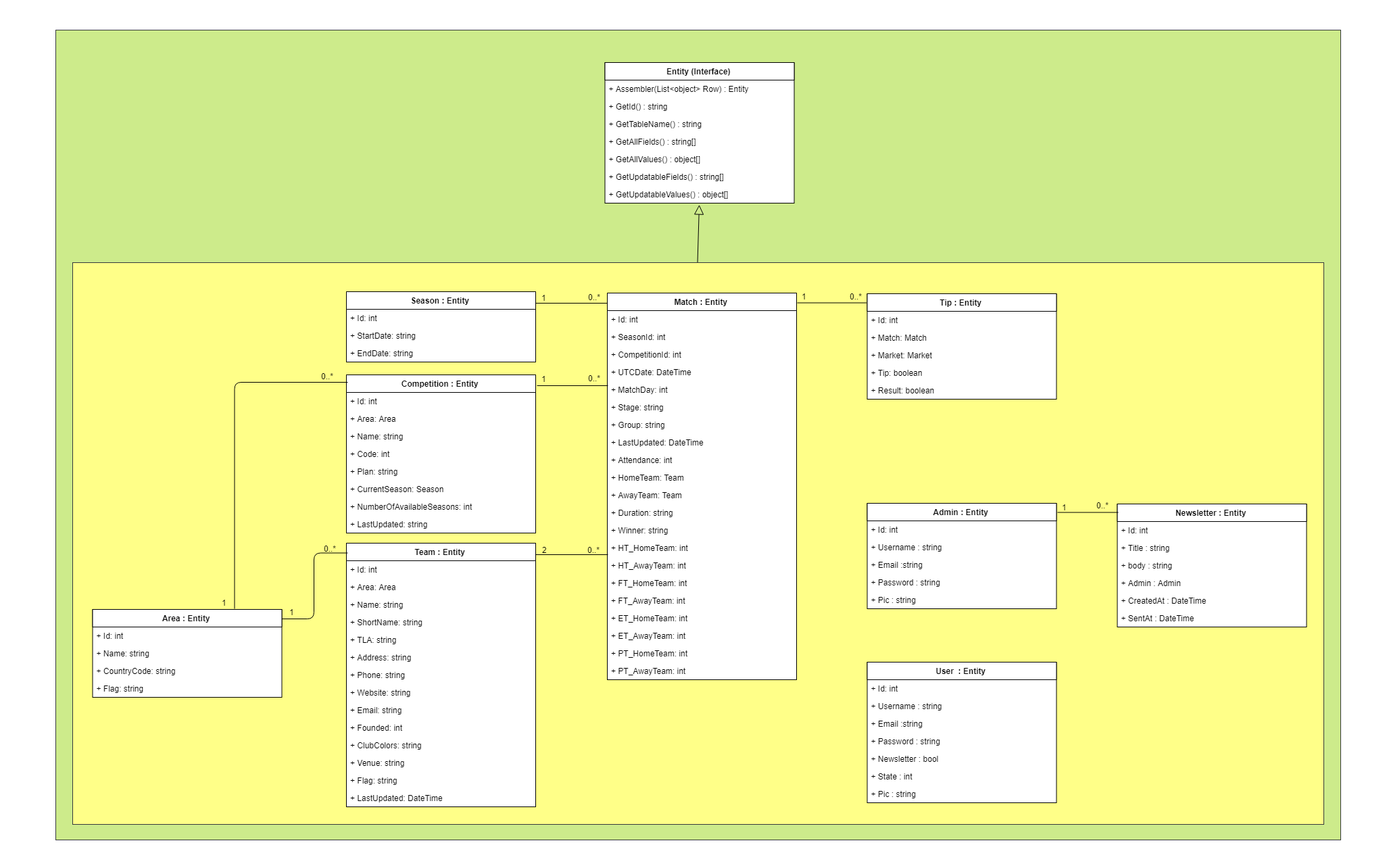
Cenário Secundário:

* 1. O utilizador pode selecionar várias ligas ao mesmo tempo.
  2. No caso de o utilizador ter selecionado várias ligas os botões das mesmas permanecerão destacados.
  3. Se não houver encontros a aplicação mostra uma mensagem de erro e volta ao ponto 1.

## Diagrama de Classes

- Representação das classes que mapeiam as tabelas mais importantes da nossa base de dados.

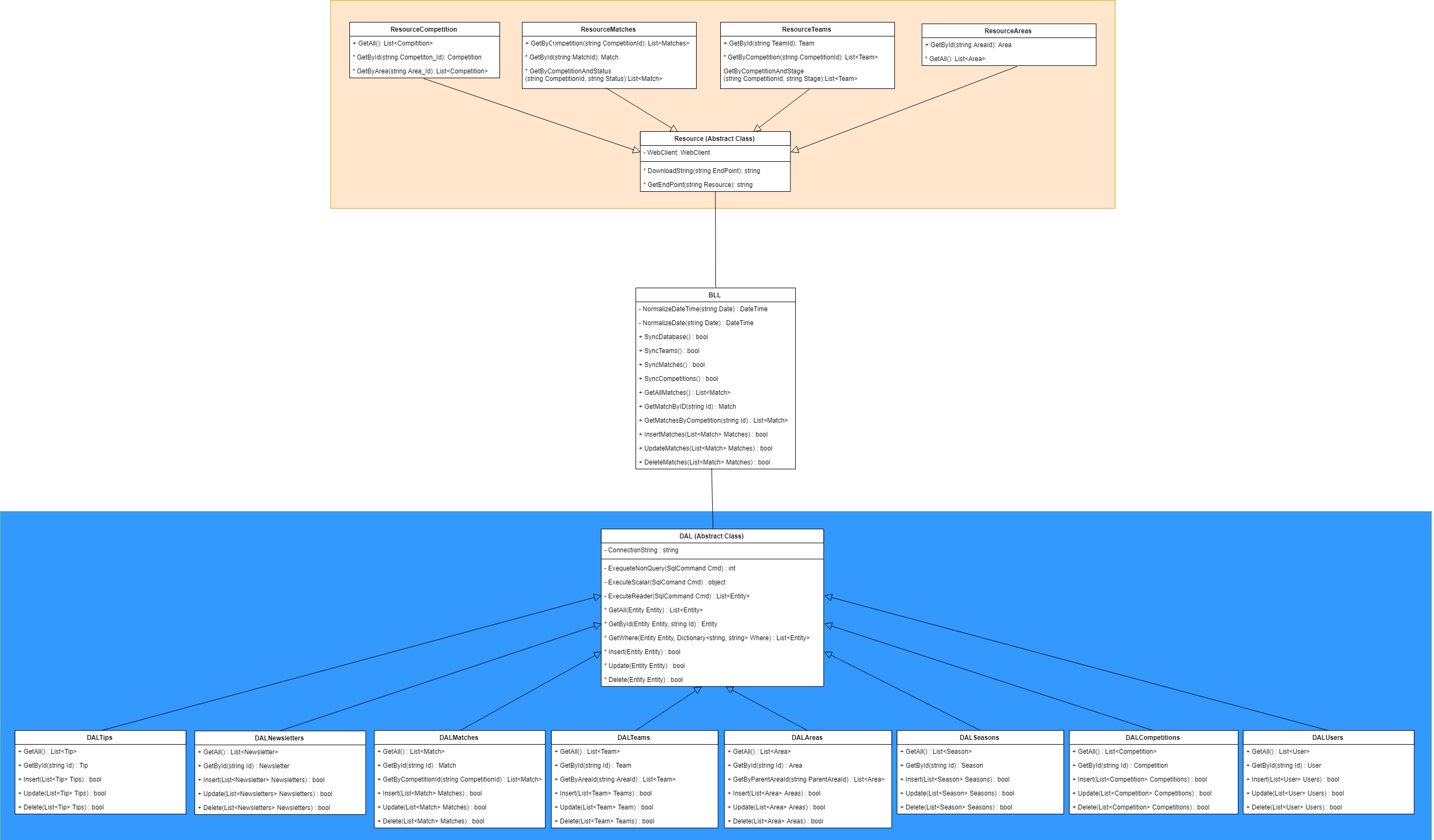
- Todas estas classes implementam a interface Entity permitindo assim o desenvolvimento de uma DAL genérica que trabalha apenas com objetos do tipo Entity.



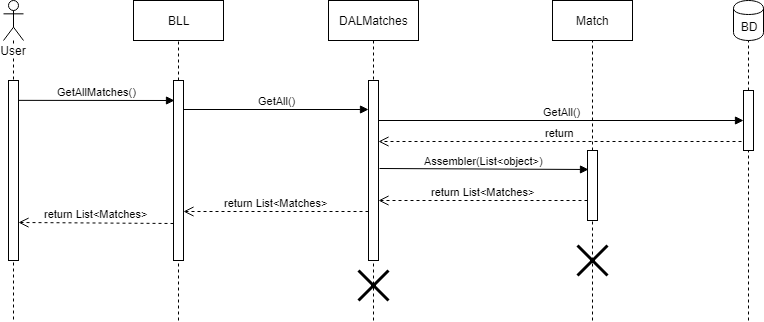
- Representação da classe abstrata DAL genérica, e das suas classes filhas. Cada uma responsável por expor os métodos necessários para comunicar com a tabela correspondente da base de dados.

- Representação da classe abstrata Resource genérica, e das suas classes filhas. Cada uma responsável por expor os métodos necessários para comunicar com os diferentes “endpoints” da API Football-Data.Org.

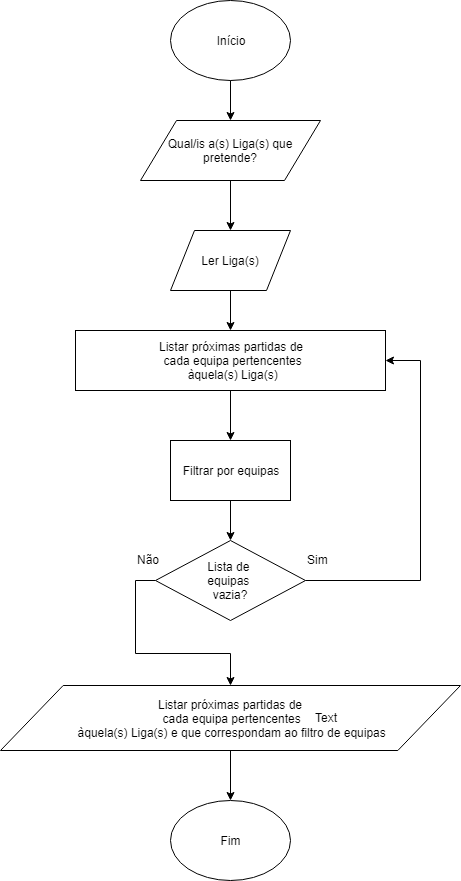
- Representação da BLL que expões os mais variados métodos para comunicar com a API, DAL e também métodos de validação.



## Diagrama de Sequência para o Use Case Listar Próximos Jogos

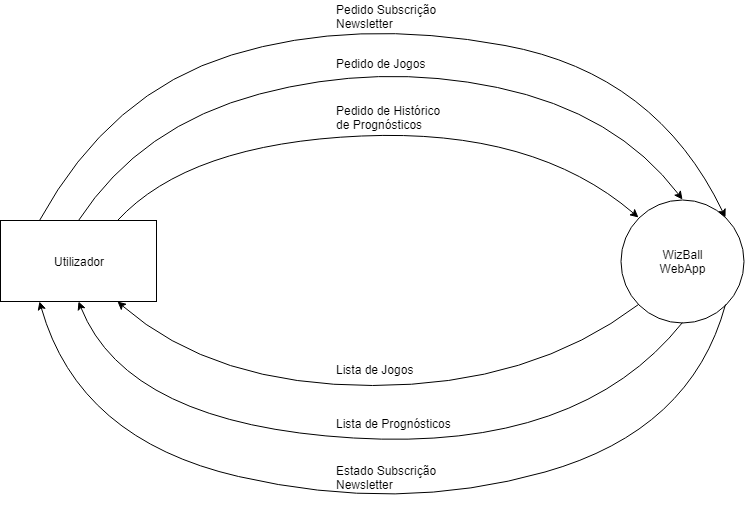


## Fluxograma para o Use Case Listar Próximos Jogos

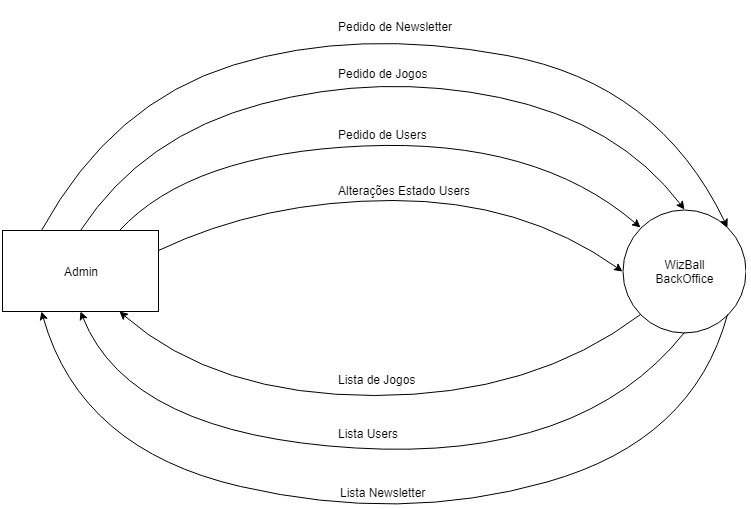


## Diagrama de Contexto

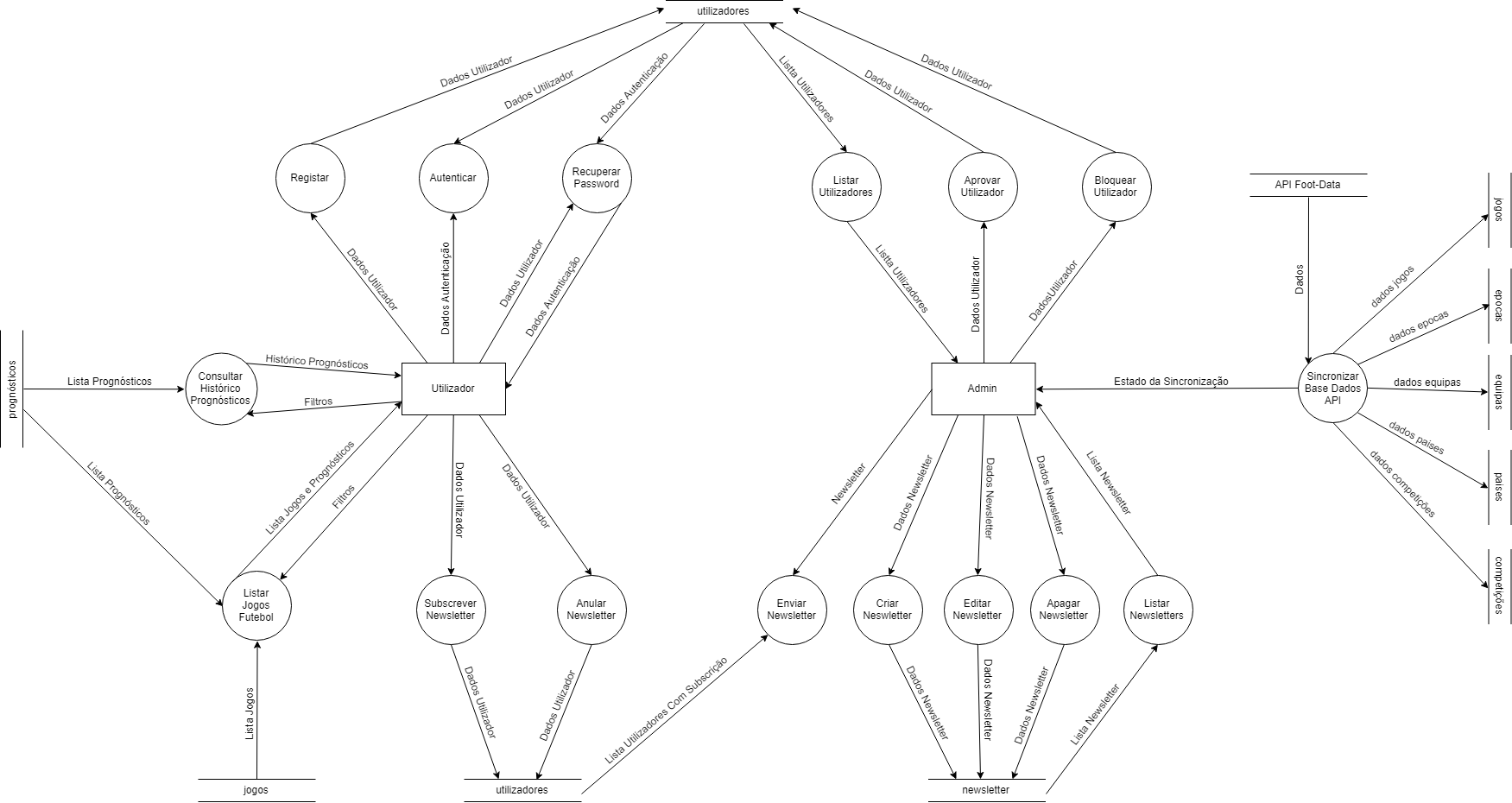
**Front Office**



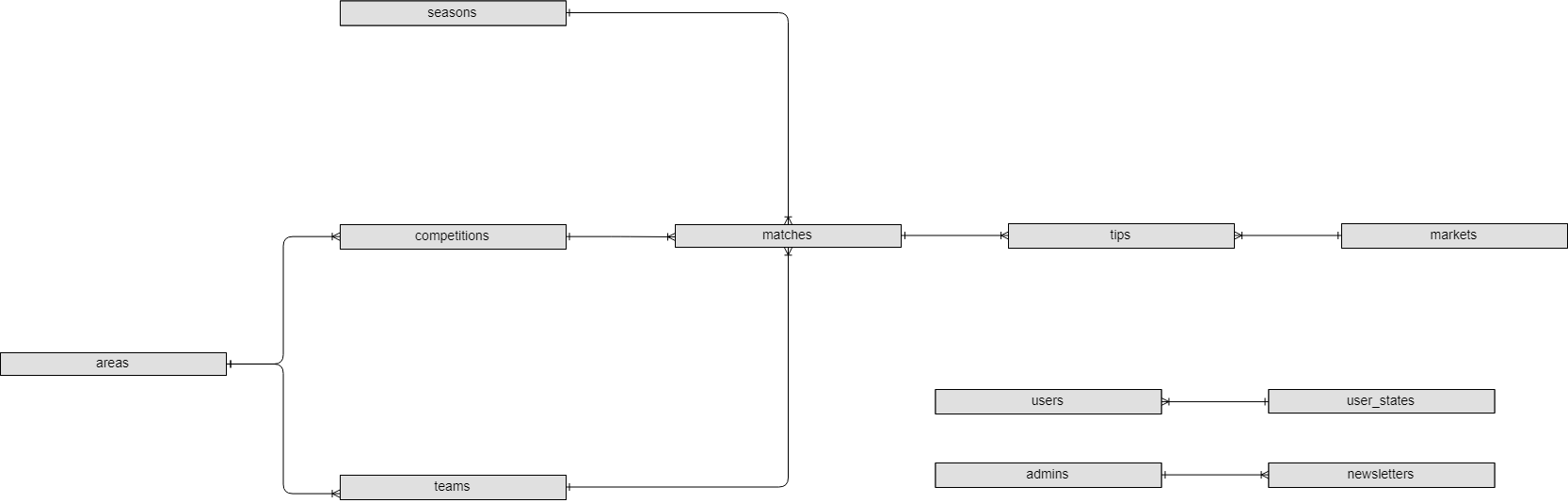
**Back Office**



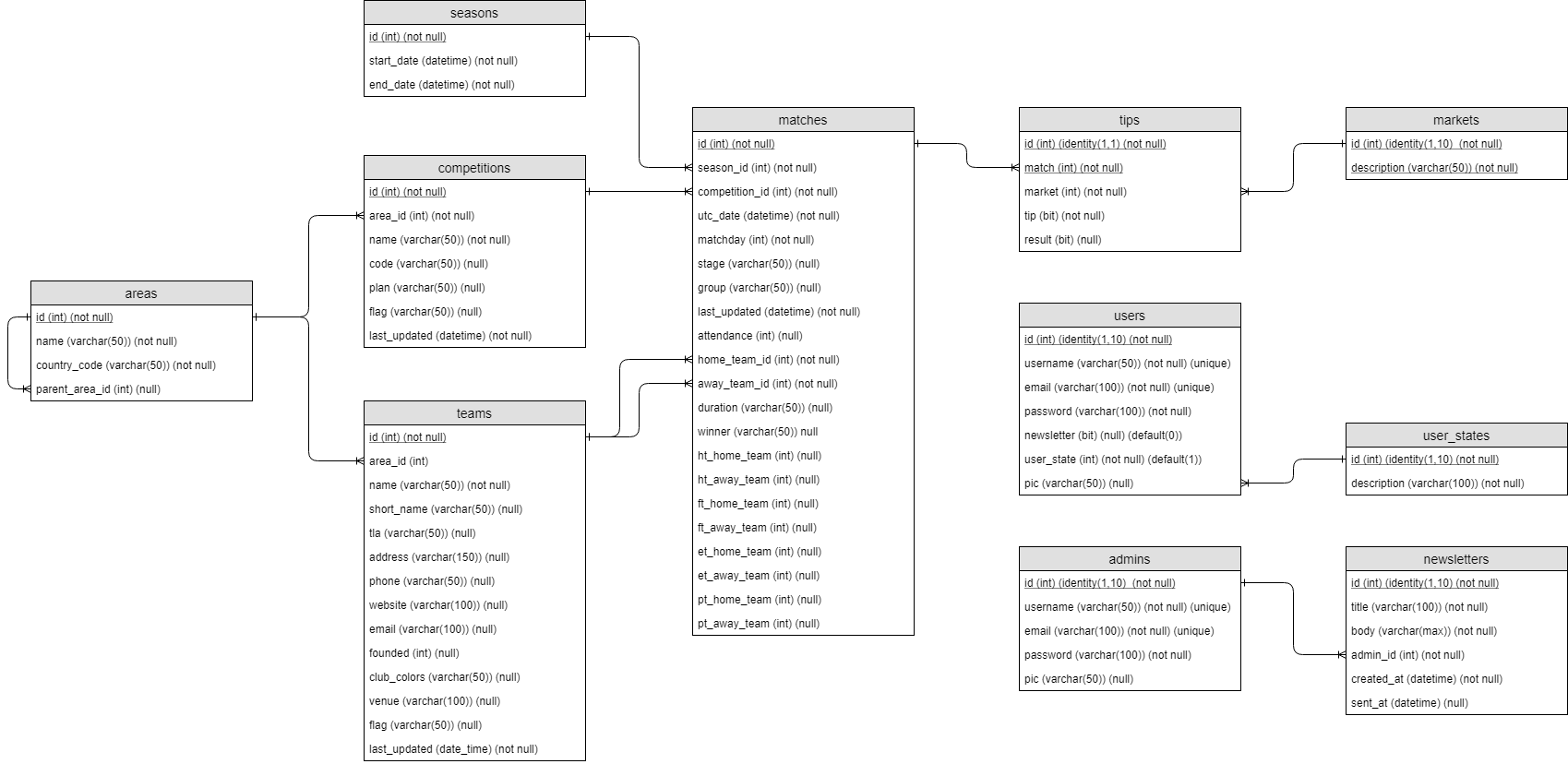
## Diagrama de Fluxo de Dados



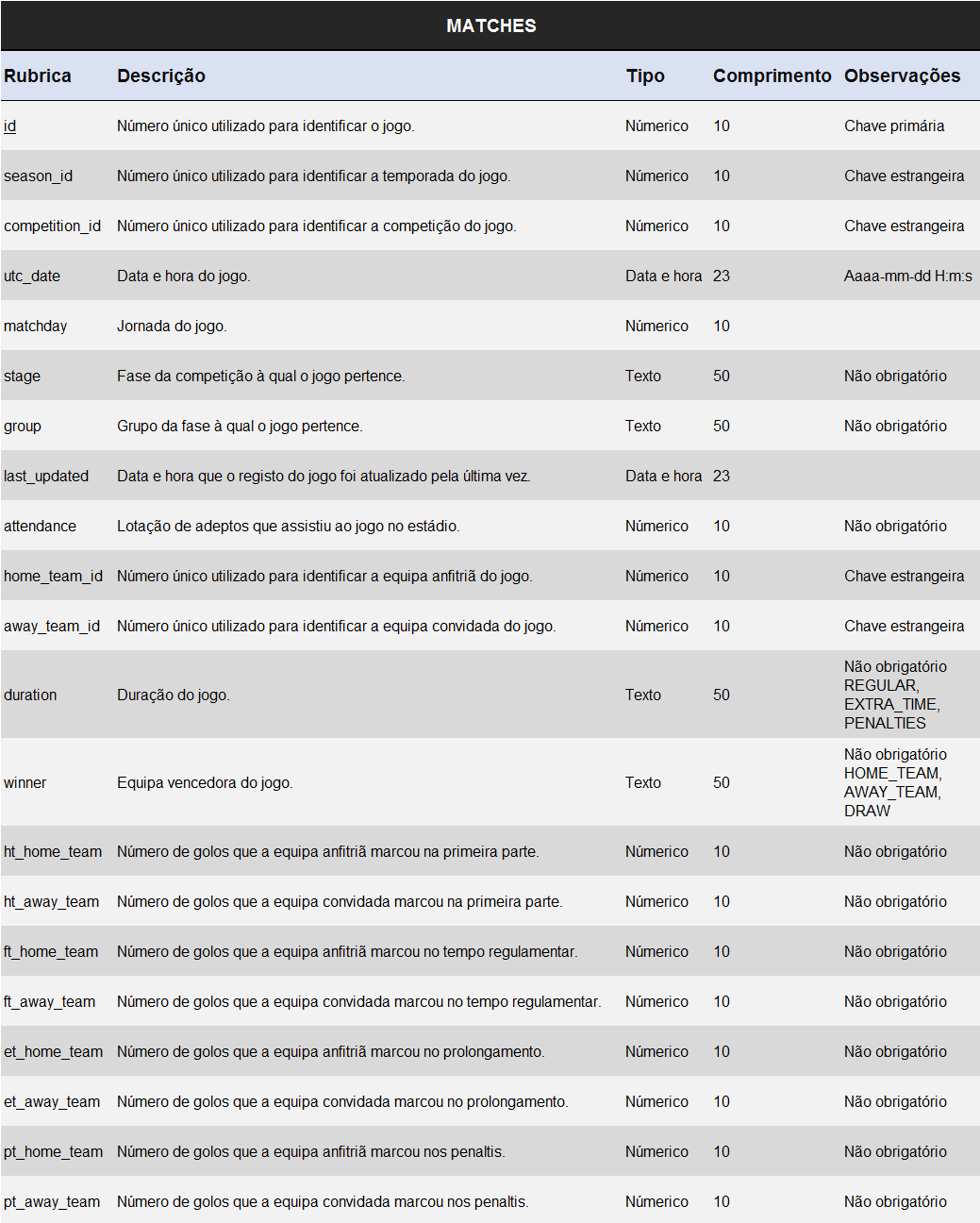
## Diagrama E-R

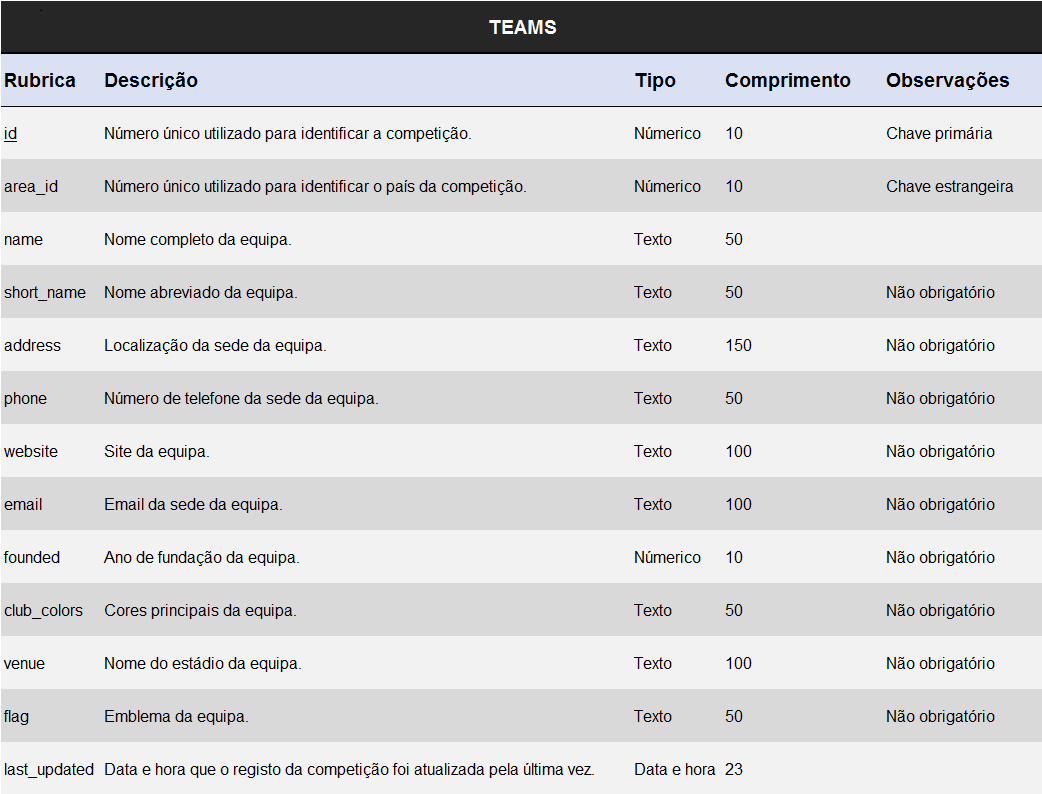


## Modelo de Dados



## Dicionário de Dados







# Desenvolvimento da Aplicação

## Módulos

Após a primeira reunião com o nosso cliente Ricardo Baptista, ficou claro que o projeto SPAD teria de ser segmentado em vários subprojectos, para que desta forma pudéssemos atender de forma mais eficiente e versátil às necessidades colocadas pelo nosso cliente. Portanto o sistema SPAD é composto por 3 módulos principais.

### Módulo - Business Logic

O módulo Business Logic é um projeto do tipo Class Library, que pode ser entendida como a espinha dorsal do to o projeto SAPD.

A responsabilidade desta class library é encapsular toda a lógica de negócios da aplicação, e disponibilizar as funcionalidades, que, tanto o Front Office, como o Back Office vão necessitar.

Por sua vez o módulo Business Logic é também ele dividido em partes lógicas menores:

1. Entities (Conjunto de classes que mapeiam as entidades da base de dados).
2. DAL (Conjunto de classes que providenciam todos os métodos necessários para interagir com a base de dados).
3. API (Conjunto de classes que providenciam todos os métodos necessários para interagir com Football-data.org).
4. BLL (Classe que expõe todos os métodos tanto da DAL como da API).
5. Web Services (Classe que expõe parcialmente os métodos da DAL e da API, para ser possível realizar chamadas AJAX apenas quando apropriado e apenas para os métodos que façam sentido).

Desta forma a Business Logic encapsula toda a lógica de negócios bem como o acesso aos dados, tanto à API como à base de dados, permitindo aos desenvolvedores que usarem esta Business Logic, concentrarem-se apenas na implementação das funcionalidades que esta expõe.

### Módulo – Back Office

O módulo Back Office é um projeto do tipo Web Application, que pode ser entendida como o cérebro do projeto.

A responsabilidade desta Web Application é a de gerir todo sistema SPAD. Ou seja, nesta aplicação podem-se realizar tarefas como:

1. Gerir utilizadores.
2. Sincronização de dados entre a API e a Base de Dados.
3. Consultar os dados da Base de Dados.
4. Enviar Newsletters.

### Módulo – Front Office

O módulo Front Office é um projeto do tipo Web Application, que pode ser entendida como a cara do projeto.

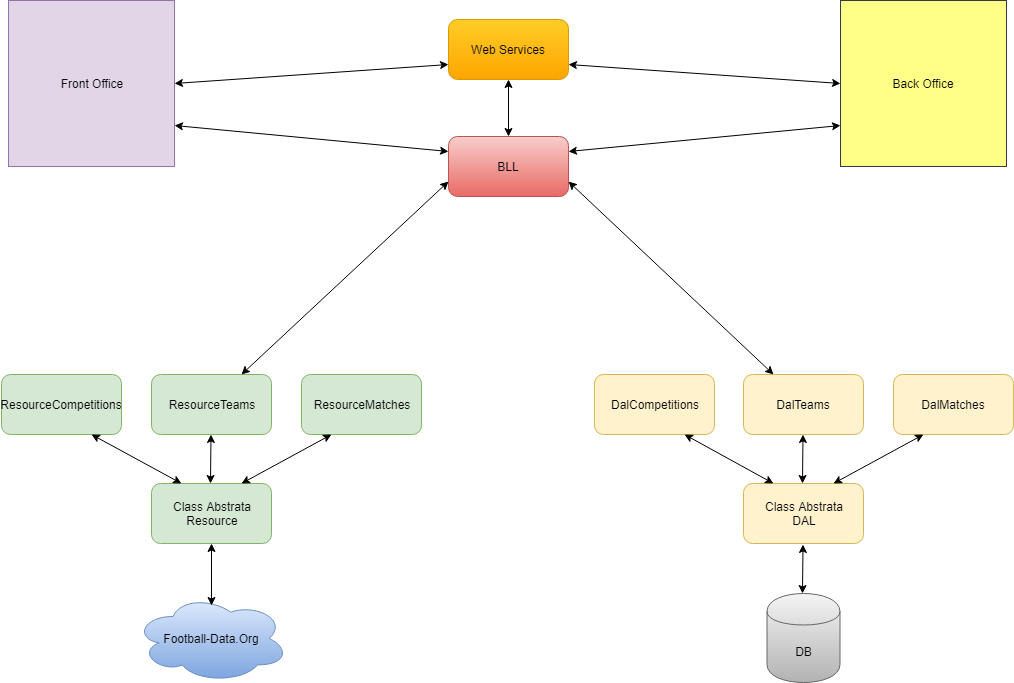
A responsabilidade desta Web Application é permitir a consulta de prognósticos de resultados de eventos desportivos, de forma rápida, simples e eficiente, a partir de qualquer lugar através da internet, e de ser capaz de adaptar à dimensão de qualquer dispositivo, garantido uma experiência de utilização sempre fluida.

## Arquitetura da Aplicação

Aplicação em 5 camadas.

1. User interface (Front Office; Back Office)
2. Business Logic Layer (Modelo de Negócio)
3. Data Access Layer (Acesso a Dados: Ms Sql Server)
4. API Access Layer (Acesso a Dados: API Football-Data.org)
5. Web Services Layer (Acesso a Dados Assíncronos)

A imagem seguinte é uma representação das diferentes camadas da aplicação e das suas relações.



Como se pode concluir, tanto o Front Office como o Back Office podem comunicar ambas com a BLL e WS, no entanto será sempre a BLL a comunicar com as camadas mais baixas da aplicação.

## Tecnologias

- Microsoft SQL Server, T-SQL.

- ASP.NET, C#.

- API football-data.org.

- HTML 5, Javascript, jQuery, CSS 3, Bootstrap 4.

- Font Awesome, Sweet Alert.

- Git.

- Gantt.

## *Pontos Importantes*

### **Business Logic**

#### DAL Genérica

De forma a implementar os conhecimentos adquiridos no curso, tentamos implementar a DAL de forma genérica. Digo tentamos, porque apenas os métodos mais requisitados foram implementados de forma genérica. Esta opção deveu-se essencialmente à complexidade que tal desenvolvimento acresce em termos de tempo, ao total de tempo disponível para desenvolver o resto da aplicação.

Então o conceito a que chegamos foi o seguinte: Existe uma class, de nome DAL. Esta class é abstrata e é a responsável por configurar toda a ligação à base de dados, e também por implementar e expor todos os métodos genéricos, que posteriormente as DALs especificas vão utilizar.

Método genérico que permite executar qualquer Stored Procedure na base de dados e converter os dados retornados na Entity passada por parâmetro. Tem ainda a possibilidade de passar os SqlParameters que o ST possa eventualmente necessitar.

protected List<Entity> ExecuteStoredProcedure(Entity Entity, string StoredProcedure, SqlParameter[] Parameters)

Método genérico que permite passar uma expressão SQL pura e converter o resultado para a Entity passada por parâmetro.

protected List<Entity> GetRaw(Entity Entity, string QueryRaw)

Método que permite obter registos da base de dados, relativos à tabela correspondente à Entity passada por parâmetro, bem como indicar uma lista de condições que satisfaçam as nossas necessidades.

protected List<Entity> GetWhere(Entity Entity, List<DbWhere> Where)

Esta é apenas uma amostra dos diversos métodos genéricos implementados na class abstrata DAL. As DALs especificas devem herdar a class abstrata DAL, e assim, desta forma, poderem consumir os recursos disponibilizados na class abstrata.

#### Lazy Loading

A DAL foi desenvolvida segundo o conceito de Lazy Loading, isto significa que todos os métodos das diferentes DALs devolvem dados “puros”. Ou seja, os dados vêm da base de dados exatamente como estão guardados.

No entanto na aplicação os objetos do tipo Entity (que mapeiam as tabelas da base de dados), são objetos complexos.

Portanto se em alguma parte da aplicação for necessário aceder a propriedades dos objetos complexos que compõem o objeto Entity, deve-se então chamar o método **EntityBuilder (Entity Obj).**

Este método tem várias versões sobrecarregadas. Após a conclusão deste método, os objetos Entities ficam complemente carregados.

#### API football-data.org

Esta aplicação perderia todo o sentido se os dados de que precisa fossem manualmente introduzidos na base de dados, uma vez que uma das necessidades do nosso cliente era exatamente automatizar o processo de obtenção de dados. Assim sendo a conceção e implementação deste módulo foi muito semelhante à da DAL genérica.

Assim, existe também uma classe abstrata de nome Resource, a qual implementa e expõe todos os métodos genéricos que os Resource específicos vão necessitar para comunicar com a API. Desta forma a classes abstrata Resource encapsula em si toda a lógica, devendo-se apenas implementar as chamadas especificas a cada End Point nos Resource específicos.

Método genérico que recebe um EndPoint ex: http:// api.football-data.org /v2/matches e que retorna uma string no formato JSON pronta a ser utilizada nas classes Resources especificas.

protected virtual string DownloadString(string EndPoint)

### **Back Office**

### **Front Office**

#### Vanilla Javascript e CSS 3 puro.

Toda a aplicação Front Office utiliza apenas Vannila Javascript e CSS 3 puro, uma decisão que provavelmente atrasou o desenvolvimento geral da aplicação, mas que foi uma decisão consciente, e que resultou numa aprendizagem adicional muito grande, que de outra forma seria muito difícil de alcançar.

Exceção são os botões e os inputs que utilizam Bootstrap 4.

#### Validações

Todas as validações necessárias no Front Office, são numa primeira instância realizadas via JS, e só quando todas as validações são positivas, se necessário, se realiza um Post Back ao servidor. Ainda assim é sempre feita dupla validação no servidor.

#### AJAX

O Front Office faz recurso de AJAX em situações especificas como por exemplo, quando um utilizador está a escolher o seu username, o Front Office faz validação via AJAX para informar o utilizador on-the-fly se o username é válido. O mesmo acontece com o email, com o login, recuperar password etc… evitando-se chamadas desnecessárias ao servidor com Post Backs.

#### Responsividade

Um dos requisitos mais importantes do projeto SPAD era que este fosse acessível via dispositivos móveis. Como tal, foi dada grande enfase à responsividade do Front Office, estando este totalmente responsivo na página Home (a mais importante) e na página Tips History (segunda mais importante). As restantes páginas ficaram parcialmente responsivas.

#### Imagens com preview sem necessidade de servidor

Uma das features que poderá ser diferenciadora no Front Office, é a funcionalidade de preview de imagem que o utilizador escolher no seu perfil, sem necessidade de a guardar previamente no servidor. Esta funcionalidade foi alcançada via Vanilla JS.

#### Bubble Sort

Sendo que o Front Office é totalmente gerado de forma dinâmica, sem recurso a qualquer template ou plug-in, e sendo o mesmo verdade para as tabelas que nele se encontram, tornou-se imperativo desenvolver uma forma de aplicar ordenação às tabelas por coluna. Assim o algoritmo escolhido, por simplicidade e rapidez de implementação, foi o Bubble Sort.

#### Utc Date to Local Date

O Front Office é uma Web Application que inevitavelmente lista vários conjuntos de informação, onde a data é extremamente importante. Por conveniência todas as datas são guardadas no base de dados no formato Utc Date. Ter a datas neste formato possibilita a conversão da mesma para qualquer data de uma região diferente, sendo apenas necessário saber a hora padrão local.

Assim foi desenvolvido do lado do cliente (pois queremos converter a Utc Date para Local Date, e apenas estando do lado do cliente conseguimos saber através de informações do seu sistema, qual é a sua hora padrão local) um método em Javascript, de nome UtcDateToLocalDate(), que acuta imediatamente após qualquer tabela ser gerada. Assim o cliente vê sempre a data dos jogos que vão decorrer, por exemplo, na sua hora local, sem necessidade do utilizador intervir com a aplicação.

#### Modals personalizados

No sentido de melhorar a experiência de utilização da Web Application e de conseguir alcançar um design diferenciado, foram desenvolvidos vários modelos de Modals para o Front Office, tendo sido para tal necessário o recurso à manipulação de HTML CSS e Javascript.

## Descrição do uso da aplicação e suas funcionalidades

Nota: Utilização de caso de sucesso principal

# Conclusão

Nota: o capítulo de conclusões é um dos mais importantes do relatório, no qual devem ser apresentados os resultados do trabalho elaborado. As conclusões devem focar o sucesso / insucesso do trabalho, revendo as dificuldades encontradas, e basear-se nos resultados realmente obtidos. Devem relacionar-se os resultados obtidos com os objectivos.

## Objectivos Cumpridos

Nota: devem ser repetidos os objectivos apresentados no capítulo de introdução e, para cada um deles, deve ser descrito o seu grau de realização na solução conseguida.

## Contratempos

Texto

## Dificuldades Encontradas

Texto

## Apreciação pessoal

Texto