



UNIVERSIDADE LUSÓFONA  
de Humanidades e Tecnologias  
*Humani nihil alienum*

# TCG DECK BUILDER

Gestão e geração de baralhos para o jogo Magic – The Gathering

## Trabalho Final de curso



Nome do Aluno: Virgínio Vunda de Andrade (Nº 21402424)

Nome do Orientador: Bruno Cipriano

Trabalho Final de Curso | Matemática | \_\_/09/2016

# Índice Geral

<b>Índice de Figuras.....</b>	<b>2</b>
<b>Resumo.....</b>	<b>Erro! Marcador não definido.</b>
<b>Abstract.....</b>	<b>5</b>
<b>Agradecimentos.....</b>	<b>5</b>
<b>Introdução .....</b>	<b>7</b>
<b>2. Enquadramento teórico.....</b>	<b>8</b>
<b>2.1 Tecnologias.....</b>	<b>8</b>
2.1.1 PHP .....	8
2.1.2 HTML .....	8
2.1.3 Javascript .....	8
2.1.4 SGDB .....	8
2.1.5 Outros .....	8
<b>3. Método .....</b>	<b>9</b>
<b>3.1 Desenvolvimento.....</b>	<b>9</b>
<b>3.2. Modelação .....</b>	<b>9</b>
3.2.1 Análise.....	10
3.2.1.1 Magic .....	10
3.2.1.1.1 Características .....	11
3.2.1.1.1.1 Tipos de Cartas .....	11
3.2.1.1.1.2 Cores das Cartas .....	12
3.2.1.1.1.3 Custo das Cartas .....	12
3.2.1.2 Modelo específico para o site .....	14
3.2.2 Modelo de dados.....	14
3.2.2.1 Aplicação .....	14
3.2.2.2 Diagrama.....	16
<b>3.3 Gerador de baralhos .....</b>	<b>17</b>
3.3.1 Input.....	17
3.3.2 Processos de geração .....	18
3.3.3 Output .....	19
<b>4. Conclusões e trabalho futuro .....</b>	<b>20</b>
<b>5. Bibliografia .....</b>	<b>21</b>
<b>Anexo A - Requisitos.....</b>	<b>23</b>
<b>Anexo B - Manual do Programador.....</b>	<b>30</b>
<b>Glossário .....</b>	<b>34</b>

## Índice de Figuras

Figura 1 - Carta Armadura de Blanchwood	10
Figura 2 - Carta Firespout	10
Figura 3 - Carta Titã do Inferno	11
Figura 4 - Símbolo da cor Azul	12
Figura 5 - Símbolo da cor "Verde e Vermelha"	12
Figura 6 – Carta Selesnya Guildmage	13
Figura 7 - Carta Scourge of the Nobilis	13
Figura 8 - Carta Machado de Lava	13
Figura 9 - Modelo de Dados	16
Figura 10 - Formulário do Gerador de Baralhos	17
Figura 11 - Fluxograma do Gerador de Baralhos	19
Figura 12 - Formulário do Registo	24
Figura 13 - Formulário de Login	24
Figura 14 - Formulário para ver/editar dados do utilizador	25
Figura 15 - Menu Paginação	30
Figura 16 - Caixinhas para Ordenação, pesquisa e filtro	30
Figura 17 - Footer Administrador	31
Figura 18 - Footer Jogador	31
Figura 19 - Janela de confirmação para eliminar algo	31

# Resumo

Este projeto final de curso teve como objetivo fazer um "construtor de baralhos (*deck builder*)" para o jogo Magic – The Gathering.

O jogo Magic é um jogo de cartas colecionáveis criado por Richard Garfield, no qual os jogadores utilizam um baralho de cartas construído de acordo com o seu modo individual de jogo para tentar vencer o baralho adversário.

O **TCG Deck Builder**, permite aos utilizadores a gestão de cartas e a utilização das mesmas para a construção de baralhos, de maneira automática ou não, sendo que este projeto dispõe de um gerador de baralhos. Por outro lado, o projeto também permite aos utilizadores a gestão dos respetivos dados pessoais.

A **aplicação** dispõe de um **BackOffice** onde utilizadores administradores fazem comodamente a gestão de cartas e utilizadores/jogadores no sistema. A gestão do sistema, alimentado por uma base de dados criteriosamente e rigorosamente gerida, reserva aos administradores o direito de banir jogadores que, de uma maneira ou doutra, contribuem para o mau funcionamento da aplicação ou o desconforto dos outros utilizadores.

**Keyword:** BackOffice, Aplicação, Utilizador, Jogador, Administrador, Web, Base de dados, Modelo de dados.

# Abstract

This final year project aimed to make a Deck Builder for the Magic game – The Gathering.

The Magic game is a collectible card game created by Richard Garfield, in which players use a deck of cards built according to your individual way of playing to try to beat the opponent deck.

The TCG Deck Builder allows users cards management and the use thereof for the construction of decks, automatic way or not, being that this design comprises a generator decks. Moreover, the design also allows users to manage the respective personal data.

The application has a BackOffice where users administrators comfortably make the management cards and users/players in the system. The system management, powered by a database carefully and rigorously managed, reservation administrators the right to ban players who, in one way or another, contribute to the malfunction of the application or the discomfort of others.

**Keyword:** BackOffice, Aplicação, Utilizador, Jogador, Administrador, Web, Base de dados, Modelo de dados.

# Agradecimentos

Agradeço a **minha** querida e guerreira **mãe** por ter sempre se abdicado dos seus interesses pessoais para fornecer a ajuda necessária aos seus, por ter sido uma mãe altruísta.

Agradeço ao professor **Bruno Cipriano** por me ter proposto este projeto e ter tornado possível a concretização do mesmo, por meio da sua “interminável” paciência, dedicação e disponibilidade enquanto docente e orientador.

Agradeço a professora **Teresa Almada** por, direta ou indiretamente, ter sido a fonte das minhas “motivações académicas”, por ter sido, para mim, indiscutivelmente o/a melhor docente que já tive.

Agradeço ao meu caro colega **Ricardo Peres** que para mim é um verdadeiro símbolo de determinação, persistência e disciplina.

# Introdução

O desenvolvimento deste projeto foi providenciado no âmbito da disciplina de Projeto Final de Curso, na Licenciatura de Matemática (no Ramo das Ciências da Computação) da Universidade Lusófona de Humanidades e Tecnologias. O projeto concentra-se na construção de uma aplicação que permita ao utilizador a coleção de cartas, a construção de baralhos de cartas (*deck builder*) e o controlo sobre os baralhos gerados, através de um formulário que permite especificar as características dos baralhos pretendidos, respeitando as regras do jogo de cartas colecionáveis (TCG)

## **Magic - The Gathering.**

Um dos pontos mais alto do projeto é o gerador de baralhos. A aplicação terá de ser capaz de gerar baralhos de maneira automática, respeitando o critério do jogador.

O Magic é um jogo de cartas cujo a existência é imperativa no âmbito da indústria do entretenimento. É um jogo criado por Richard Garfield principalmente com o objetivo de satisfazer os bons apreciadores de cartas que promovem a arte no geral, sem qualquer tipo de preconceito.

Assim, para atingir os objetivos definidos acima, este projeto funciona como uma ferramenta eficaz de gestão de cartas colecionáveis, poupando tempo em processos manuais.

O projeto despertou o meu interesse por vários motivos, uns dos principais é porque se trata de uma plataforma *web*, de uma aplicação cujo o conteúdo programático (na minha opinião) envolve uma rigorosa preparação teórico-prática especializada em SGDB e principalmente por se tratar dum projeto que me ajudaria a aprender (pelo menos o básico de) três linguagens de programação muito solicitadas pelo empregadores: PHP, HTML e Javascript.

## 2. Enquadramento teórico

### 2.1 Tecnologias

Descrição das linguagens de programação e outras tecnologias usadas para a escrita do e execução do código fonte.

#### 2.1.1 PHP

O PHP é uma linguagem de *script open source* que coaduna principalmente com o desenvolvimento de plataformas *web* e que com facilidade pode ser utilizada juntamente com HTML. Por isso, escolhi esta linguagem para desenvolver a maior parte desta aplicação.

#### 2.1.2 HTML

O HTML é uma linguagem de formatação utilizada na construção de páginas na *web*.

#### 2.1.3 Javascript

O Javascript é uma linguagem *client-side*, em outras palavras, processada pelo próprio *browser* e foi usada para validar ações que intersejam os interesses principais da aplicação.

#### 2.1.4 SGDB

Como SGBD foi usado o MySQL. O MySQL é um sistema de gestão de base de dados que utiliza a linguagem SQL (*Structured Query Language*) como interface. Esta linguagem foi utilizada como parte indispensável na interação da aplicação com a sua base de dados e como expoente máximo da gestão de dados.

#### 2.1.5 Outros

Nesta secção apresentam-se outras ferramentas tecnológicas usadas para escrita e execução do código fonte.

- **Sublime Text 3**

O Sublime Text 3 é um editor de código-fonte multiplataforma. O Sublime Text 3 é considerado um editor de texto, e não uma IDE.

- **XAMPP**

O XAMPP, trata-se de um servidor independente de plataforma que consiste num pacote de *software* contendo um servidor Apache, o MySQL e os interpretadores para linguagens de script: PHP e Perl pré-configurados. Este *software* é gratuito, e funciona na maior parte dos sistemas operativos, sendo um grande apoio no desenvolvimento de *websites*.



## 3. Método

Nesta secção é apresentado o método seguido para o desenvolvimento do projeto.

### 3.1 Desenvolvimento

Este presente projeto de gestão de cartas colecionáveis é constituído por duas componentes distintas:

- **BackOffice** – Sistema de gestão de dados utilizados pela aplicação. Esta componente permite ao administrador a gestão dos dados utilizados pela aplicação.
- **Aplicação** – Utiliza os dados geridos no *BackOffice* para tornar possível o processamento pretendido para os mesmos. Esta componente permite ao utilizador ter a sua lista/coleção de cartas, usa-las para a construção dos seus baralhos, de maneira automática ou não, e gestão dos seus dados pessoais essenciais para o bom funcionamento da aplicação.

De maneiras a se conseguir atingir o grau de complexidade exigido num projeto desta dimensão ou a estar mais próximo daquilo que é a realidade do jogo Magic, numa primeira fase, fez-se primeiramente um estudo minucioso que visou a compreensão das regras, princípios e requisitos que fazem do Magic “um jogo de interesse comum”.

Nesta fase percebeu-se a necessidade de uma concentração mais ampla nos direitos do jogador de possuir uma lista/coleção de cartas, na gestão/manipulação das mesmas que envolve: adicionar, editar e construir baralhos de cartas que façam parte da sua lista/coleção de cartas, de acordo com as suas preferências de jogo para tentar vencer o baralho adversário. Por uma questão de lógica e consistência, viu-se também a necessidade de dar ao termo editar o seguinte significado: apagar carta, definir quantidade da carta na lista do utilizador, comentar e etc.

Após a esta primeira fase, já com uma ideia suficientemente apurada do Magic, trabalhou-se arduamente na elaboração de um esboço do modelo de dados e na implementação do *BackOffice* que incluiu uma série de testes quase que interminável, já que o projeto foi na maior parte desenvolvido por meio de duas linguagens de programação que eu desconhecia: HTML e PHP.

Depois do meu “à vontade” com o PHP e HTML, da definição e implementação do modelo de dados (construção das tabelas e preenchimento das mesmas para efeito de testes) e de parte do *BackOffice*, iniciou-se literalmente a implementação da aplicação com as seguintes funcionalidades: registo e *login*, lista de cartas e a sua gestão, criação de baralhos e a sua gestão, paginação, pesquisa, filtros, ordenação, dados reais e etc...

Também é digno de nota as muitas vezes que grandes quantidades de códigos tiveram de ser revisadas/desconsideradas para garantir o sucesso da aplicação.

### 3.2. Modelação

Segue-se uma demonstração de como foi construída o modelo de dados que suporta a aplicação.

### 3.2.1 Análise

A análise do modelo de dados consiste no processo de suporte a concepção, projeto e verificação do modelo de dados. Está dividido em duas partes:

- **Magic** – Onde descreve-se algumas características das cartas, segundo a realidade do jogo Magic.
- **Modelo específico para o site** – Onde descreve-se as relações existentes entre as entidades do modelo de dados, com base nos objectivos do *software*.

#### 3.2.1.1 Magic

Deixando-me guiar por alguns conceitos básicos do jogo Magic, encontrados nalguns manuais do jogo mencionados na Bibliografia, constatei nas cartas algumas características relevantes para a Modelação. Por exemplo, notei que todas as cartas tinham um nome, tipo, cor, descrição/texto. Notei também que todas as cartas com a exceção das do tipo “Terreno” tinham um custo.

O que também foi digno de nota é que as cartas podiam ter uma ou várias cores ou até mesmo serem incolores, conforme o exemplo nas figuras 1 e 2 abaixo:



Figura 1 - Carta Armadura de Blanchwood



Figura 2 - Carta Firespout

### 3.2.1.1.1 Características

Atributos que nos ajudam a identificar uma carta da outra. Podemos observar na figura 3 abaixo.



Figura 3 - Carta Titã do Inferno

Para continuarmos focados no real objetivo deste *software*, nem todos os atributos das cartas foram considerados. Por exemplo, não considerou-se a “Força/Resistência” e a “Expansão/Raridade”.

#### 3.2.1.1.1.1 Tipos de Cartas

Existem os seguintes tipos de cartas:

- Terreno
- Criatura
- Artefato
- Planeswalker
- Encantamento
- Magia instantânea
- Feitiço

### 3.2.1.1.1.2 Cores das Cartas

As cartas podem ter as seguintes cores:

- Branca
- Preta
- Vermelha
- Azul
- Verde
- Verde
- Vermelha
- Verde + Branca
- Vermelha + Branca
- Preta + Branca
- Verde + Azul

As cores têm um símbolo de representação. Nas figuras 4 e 5 abaixo veremos dois deles:



Figura 4 - Símbolo da cor Azul



Figura 5 - Símbolo da cor "Verde e Vermelha"

Por razões de simplificação do sistema, a modelação de cores mistas foi feita como se fosse uma cor. Por exemplo, em vez de considerar-se (por lista) “Verde” + “Vermelha” é considerado (por oposição à lista) “Verde + Vermelho”. Por outro lado, se a modelação de cores mistas fosse feita de uma outra maneira então exigiria muito trabalho que não justificaria o resultado que pode ser obtido de uma maneira simplificada, assim como foi o sucedido.

### 3.2.1.1.1.3 Custo das Cartas

Todas as cartas, com exceção das cartas do tipo “Terreno”, têm um custo que normalmente vai de um a cem. Assim, considerou-se que todas as cartas têm um custo e que o custo das cartas do tipo terreno é igual a zero.

Por razões de simplificação do sistema, o “custo” foi considerado apenas de forma numérica mas o custo real varia por cor. Para melhor percepção, consideremos alguns exemplos por meio das figuras 1, 2 e 3:



Figura 6 – Carta Selesnya Guildmage



Figura 7 - Carta Scourge of the Nobilis



Figura 8 - Carta Machado de Lava

### Figura 6

O custo real da carta é “2 (Verde + Branco)” mas para o sistema é considerado 2.

### Figura 7

O custo real da carta é “2 genérico” + “1 (Vermelho + Branco)” mas para o sistema é considerado 3.

### Figura 8

Para esta carta o custo real é “4 genérico” + “1 vermelho” mas para o sistema é 5.

### 3.2.1.2 Modelo específico para o site

Vejamos as relações estabelecidas entre as entidades do modelo de dados para alcançar os objetivos do *software*:

- Uma carta pode pertencer a zero ou mais jogadores.
- Um jogador pode ter zero ou mais cartas.
- Uma carta pode pertencer a zero ou mais baralhos.
- Um baralho pode ter zero ou mais cartas.
- Os utilizadores podem comentar as cartas e uma carta pode ter zero ou mais comentários.
- Os utilizadores precisam de dados singulares como *e-mail* e *password* para serem utilizados pelo mesmo para efetuar a sua autenticação no *site*. Precisam também de um *username* e uma fotografia para tornar mais simpática a sua descrição/representação no *site*.
- Os utilizadores, as cartas, os baralhos e os comentários tinham de ter um identificador único/id para que fossem devidamente identificados para uma eventual operação no *site*.
- Cada utilizador tem a sua lista/colecção de cartas devidamente identificada, onde poderá editar a quantidade das mesmas e ver as cartas que pode utilizar para formar os seus baralhos.
- Os baralhos pertencem sempre a um jogador e em parte representam uma lista de zero ou mais cartas editáveis, possuem uma descrição.

### 3.2.2 Modelo de dados

Em seguida, vamos apresentar uma explicação da aplicação e desenho da análise do modelo de dados.

#### 3.2.2.1 Aplicação

Primeiramente identificou-se a necessidade das seguintes entidades/tabelas: Utilizador, Carta e Baralho. Como as cartas têm alguns atributos que também têm os seus respetivos atributos (p.e. “cor”, “tipo”) viu-se a necessidade de acrescentar mais algumas entidades: Cor, Tipo e Comentário. Nota-se a necessidade de diferenciar os utilizadores que podem ser administradores ou jogadores, assim surge a tabela “TipoUtilizador”. Conforme vimos na análise, o jogador deve ter o privilégio de ter ou colecionar carta(s), por isso acrescenta-se ao modelo de dados a entidade “CartaUtilizador”. O utilizador também deve poder construir/ter os seus baralhos de cartas colecionáveis, assim acrescenta-se também a entidade “Baralho”. Um baralho pode ser gerado ou criado pelo utilizador e até mesmo arquivado, para lidar com esta realidade acrescenta-se a tabela “TipoBaralho”. Deve-se apresentar as configurações do baralho, por isso temos também a tabela “Configuracoes” que é ligada a tabela “Baralho” por meio da tabela “ConfiguracoesBaralho”. As configurações do baralho incluem as cores das cartas que foram usadas para gerar o baralho e o tipo de jogo, por isso se adiciona também as tabelas “CoresCartasBaralhoGerado” e “TipoJogo”.

As tabelas Carta, Utilizador, Baralho, TipoJogo, Cor e etc., devem possuir um identificador único “id”, uma vez que nestas tabelas cada linha representa uma entidade singular. O mesmo não acontece com as tabelas como “ConfiguracoesBaralho”, “CartaUtilizador” e etc.

Nas sub-seções seguintes descrevem-se em maior detalhe todas as tabelas (algumas em função das outras) do modelo de dados:

## Utilizador

A tabela “Utilizador” permite o registo dos dados dos utilizadores no sistema, de maneiras a estarem habilitados a realizar a autenticação no *site*, onde poderão efetivamente desfrutar de todas as funcionalidades com os direitos reservados ao utilizador. Por razões de segurança, a *password* é guardada na base de dados encriptada com o algoritmo sha256 e não em *plain text*. Para o *login* o utilizador precisa de um *e-mail* e uma *password*, precisa também de um nome e uma foto para estar devidamente representado no sistema. Por esta razão a tabela “Utilizador” tem os seguintes campos: “id”, “nome”, “e-mail”, “password”, “ficheiro (ou foto)”, “data\_registro” e “tipo”. O campo “tipo” da tabela “Utilizador” faz referência ao campo “id” da tabela “**TipoUtilizador**” que tem também um campo chamado “descricao”.

## Carta

Os campos da tabela “Carta” são os seguintes: id, nome, cor, custo, tipo, descricao e ficheiro. A carta podem ou não pertencer a um utilizador, por isso temos a tabela “**CartaUtilizador**” com os seguintes campos: id\_utilizador, id\_carta e quantidade. A carta pode ter zero ou mais comentários feitos pelo utilizador, por isso temos a tabela “**Comentario**” com os seguintes campos: id\_comentario, id\_carta, id\_utilizador, descricao, data e hora. Como vimos antes, a tabela “Carta” tem um campo “cor”. Este campo “cor” referência o campo “id” da tabela “**Cor**” que também tem os seguintes campos: nome e ficheiro. O campo “tipo” da tabela “Carta” faz referência ao campo “id” da tabela “**Tipo**” que tem um outro campo de nome “descricao”.

## Baralho

Caso exista um baralho, o mesmo terá de pertencer a um utilizador e ter algumas informações específica que o caracterizam. Da análise dos requisitos resultou a identificação dos seguintes campos: id\_baralho, id\_utilizador, nome, descricao, data\_criacao, hora\_criacao, data\_ultima\_alteracao, tipo. O campo “tipo” é chave estrangeira derivada da chave primária “id” da tabela “**TipoBaralho**” que também tem o seguinte campo: descrição. Cada baralho tem a sua configuração que pode ser igual a de um outro baralho, uma vez que o utilizador pode gerar mais de uma baralho, introduzindo o número 2 ou 3 no campo “Número maximo baralhos” do formulário do gerador de baralho(s). Assim, vi a necessidade de ter uma nova tabela “**ConfiguracoesBaralho**” com os seguintes campos: id\_configuracao e id\_baralho. O campo “id\_configuracao” faz referência ao campo “id\_configuracao” da tabela “**Configuracoes**” que tem também os seguintes campos: qtd\_cartas\_baralho, tipo\_jogo, custo\_max\_aceitavel, perc\_terrenos, terrenos\_doutras\_cores, num\_min\_copias\_cartas, num\_max\_copias\_cartas, num\_baralhos. Todos estes campos são criados para guardar as informações do formulário do gerador de baralhos. O campo “tipo\_jogo” guardo o “id” do tipo de jogo que representa uma linha da tabela “**TipoJogo**” que também o campo “descricao”. Um baralho pode ter várias cartas que podem ter cores diferentes, ou seja, as configurações de um baralho podem envolver várias. Assim, deu-se origem a tabela “**CoresCartasBaralhoGerado**” que tem os seguintes campos: id\_configuracao e id\_cor.

### 3.2.2.2 Diagrama

Apresenta as características de funcionamento e comportamento do *software*, conforme vemos abaixo na figura 9.

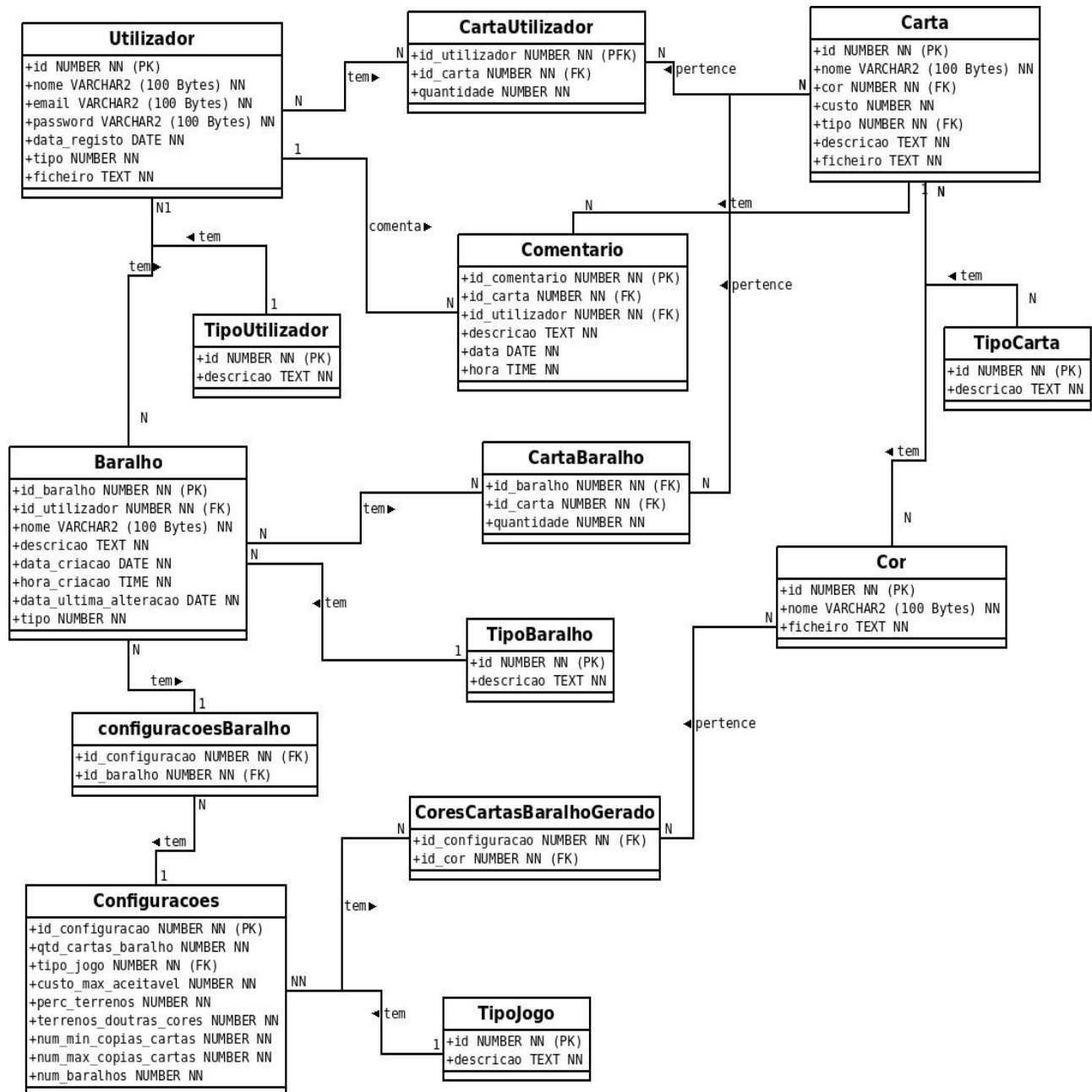


Figura 9 - Modelo de Dados



### 3.3 Gerador de baralhos

Nesta secção vão ser apresentados pormenores técnicos da implementação e do funcionamento do Gerador de Baralhos. Esta secção tem 3 sub-secções:

- **Input** – Onde se descreve a informação que serve de entrada ao gerador.
- **Processos de geração** – Onde se descreve detalhadamente os passos dados para a geração dos baralhos, assim também como a sequência destes mesmos passos.
- **Output** – Onde se descrevem os dados de saída do gerador.

#### 3.3.1 Input

O *input* é feito por meio do seguinte formulário:

### Gerador de Baralho(s)

Escolha as cores:

Quantidade de cartas Baralho(s)

Escolha o tipo de jogo

Custo máx. aceitável

Porcentagem de terreno (1-100)

Usar terrenos de outras cores?

Núm. mínimo cópias cartas (1-4)

Núm. máx. cóp. cartas (nMinCoCartas-4)

Número máximo de baralhos (1-3)

☐ Vermelha  
☐ Incolor  
☐ Preta  
☐ Branca  
☐ Azul  
☐ Verde  
☐ Verde e Vermelha  
☐ Verde e Branca  
☐ Vermelha e Branca  
☐ Branca e Preta  
☐ Verde e Azul

Escolha... ▾

Nao ▾

Gerar

Figura 10 - Formulário do Gerador de Baralhos

Seguem-se algumas explicações dos campos do formulário acima:

- **Escolha as cores:**

Neste campo o jogador pode escolher as cores das cartas que pretende ter no(s) baralho(s).

- **Quantidade de cartas Baralho(s)**

Aqui o jogador especifica o número de cartas que todos os baralhos devem ter. Este número não pode ser menor que um.

- **Escolha o tipo de jogo**

A escolha do tipo de jogo serve para, a medida que selecionamos as cartas para os baralhos, dar prioridade as cartas com determinados custos.

Conforme poderemos observar com mais detalhes no Anexo A, existem 3 tipos de jogos:

- **Rápido** – Onde o sistema privilegia cartas baratas.
- **Misto** – Onde o sistema privilegia cartas de custo médio.
- **Lento** – Onde o sistema privilegia cartas caras.

- **Custo máx. aceitável**

Serve para especificar o custo máximo que todas as cartas selecionadas para os baralhos devem ter.

- **Porcentagem de terreno (1-100)**

Aqui o jogador deve especificar qual deve ser a percentagem de terrenos em função do número de cartas que os baralhos terão. O número deve ir de 1 a 100.

- **Usar terrenos de outras cores?**

Serve para o jogador confirmar se pretende que os seus terrenos tenham cores diferentes das especificadas inicialmente, caso não tenha terrenos suficientes nas cores já mencionadas.

- **Núm. mínimo copias cartas (1-4)**

Aqui o jogador deve especificar o número mínimo de cópias que quer para cada carta dos seus baralhos. O número não pode ser inferior que 1 ou maior que 4.

- **Núm. máx. cop. Cartas (nMinCoCartas-4)**

Aqui o jogador deve especificar o número máximo de cópias que quer para cada carta dos seus baralhos. Este número não deve ser inferior ao número introduzido no campo anterior nem maior que 4.

- **Número máximo baralhos (1-3)**

Serve para especificar o número de baralhos que se pretende gerar. Este número deve estar entre 1 e 3.

### 3.3.2 Processos de geração

Por meio do formulário do gerador de baralhos, o jogador especifica o número de cartas que os baralhos devem ter, as cores das cartas, a percentagem das cartas do tipo terreno, o número mínimo e máximo de cópias das cartas, o número de baralhos a gerar, o custo máximo das cartas, o tipo de jogo, “usar terrenos doutras cores”.

Depois de clicar em "Gerar", verifica-se se o jogador preencheu devidamente todos os campos do formulário, garante-se que na sua lista/coleção de cartas temos o número de cartas que se pretende ter nos baralhos a gerar, de acordo com as características das cartas especificadas no formulário.

Recorrendo à base de dados, selecciona-se as cartas do jogador, dividindo-as em dois grupos ("Terrenos" e "não Terrenos") das cartas do tipo "Terreno" tira-se o número de cartas do tipo "Terreno" e analogamente faz-se o mesmo com as cartas que não são do tipo "Terreno".

Cria-se o baralho e a sua configuração, associa-se a configuração ao baralho, associam-se as cartas ao baralho, grava-se a configuração e apresentam-se os baralhos gerados ao utilizador.

O fluxograma (na figura 11 abaixo) ilustra o processo de geração dos baralhos.

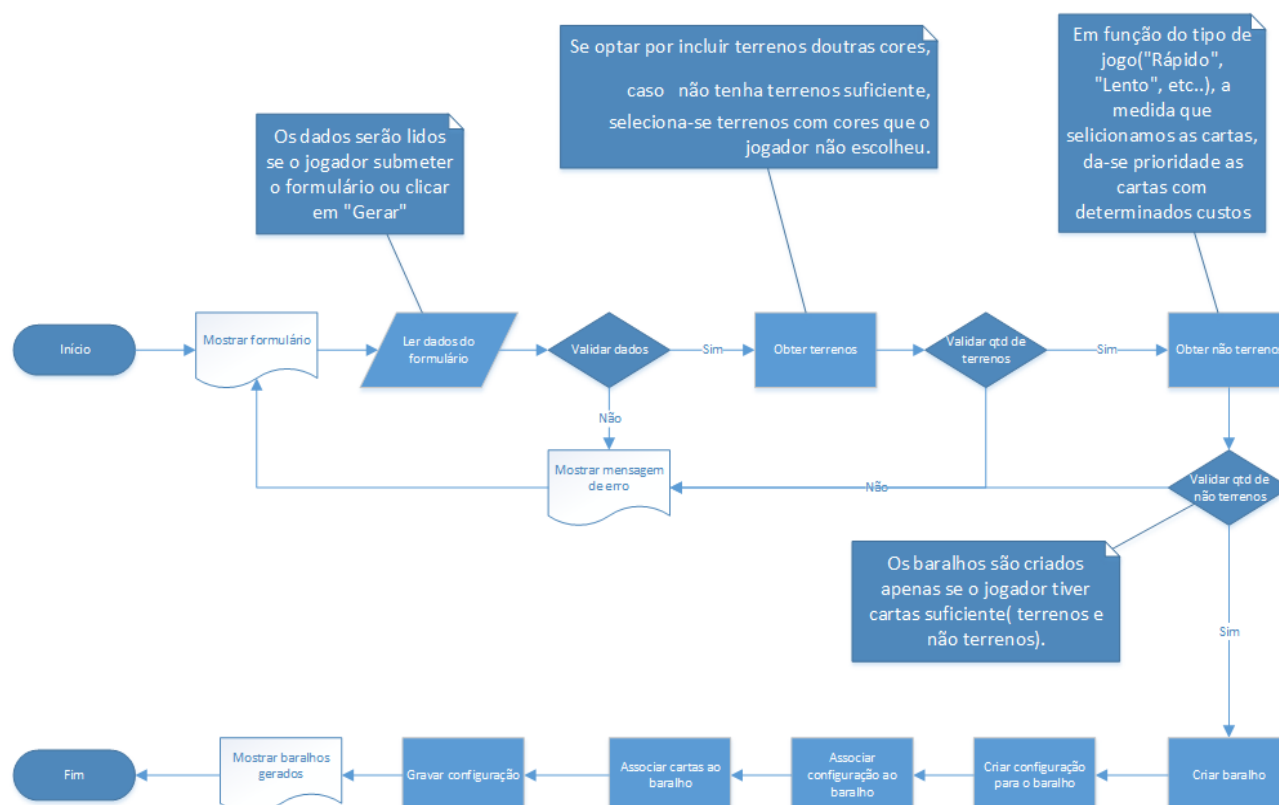


Figura 11 - Fluxograma do Gerador de Baralhos

### 3.3.3 Output

O *Output* do gerador de baralhos são os baralhos gerados. Vamos agora ver alguns aspetos interessantes para esta parte:

- **Número de baralhos gerados**

O número é especificado pelo utilizador no último campo do formulário do gerador de baralhos.

- **Diferença entre os baralhos**

Pode estar na quantidade de cópias de uma determinada carta em cada baralho, uma vez que o número de cópias de uma determinada carta pertencente a um baralho gerado é providenciado de maneira aleatória.

Os baralhos gerados ficam automaticamente associados ao jogador que os gerou/criou.

## 4. Conclusões e trabalho futuro

A finalidade deste projeto foi fazer um gerador de baralhos de cartas colecionáveis, permitir ao administrador a adição de características próprias do jogo (p.e. “tipo de carta”, “cor”, etc) por meio de um formulário que especifica objetivo, dar algum controle ao jogador sobre os baralhos gerados e gerar alguns gráficos com algumas estatísticas sobre os baralhos, cartas e etc. Assim, com este objetivo em mente, o foco foi construir um *software* que tornasse possível a automatização dos processos que permitem o alcance dos objetivos traçados, de maneiras a reduzir o tempo e o esforço derivado de trabalhos manuais.

Com muita persistência e disciplina, com o apoio incontestável do professor orientador Bruno Cipriano e muitas pesquisas, conseguiu-se alcançar não só os objetivos traçados inicialmente mas também muitos outros que tornaram o projeto/*software* mais consistente, de fácil utilização e mais credível.

Um dos maiores desafios que enfrentei foi ter de colocar a paginação, ordenação, pesquisas e filtros a funcionar de uma maneira sincronizada. Também foi bastante complicado projetar uma “tecnologia” que gerasse baralhos, esta parte exigiu de mim muita disciplina, concentração e paciência. A maior dificuldade que tive foi incluir a possibilidade do utilizador recuperar a sua *password*, desenvolvi todo o código fonte necessário mas não consegui configurar o XAMPP para fazer envios de *e-mails*. Como trabalho futuro, pretende-se estender esta aplicação com funcionalidades que a conferem um estado mais próximo da realidade do jogo oficial Magic. Por exemplo, o programa considera que o custo das cartas é um valor inteiro mas na realidade é algo muito mais complexo. Também pretendo futuramente conseguir incluir a possibilidade do utilizador recuperar a sua *password*.

## 5. Bibliografia

### Manual de PHP

<http://php.net/manual/en/function.time.php>  
<http://php.net/manual/en/function.header.php>  
<http://www.rapidtables.com/web/dev/php-redirect.htm>  
<http://php.net/manual/en/function.hash.php>  
<http://php.net/manual/en/function.hash.php>

### Manual de HTML

<http://www.rapidtables.com/web/dev/html-redirect.htm>  
[http://www.w3schools.com/charsets/ref\\_html\\_8859.asp](http://www.w3schools.com/charsets/ref_html_8859.asp)  
[http://www.w3schools.com/html/html\\_charset.asp](http://www.w3schools.com/html/html_charset.asp)

### Javascript

[http://www.w3schools.com/jsref/met\\_win\\_alert.asp](http://www.w3schools.com/jsref/met_win_alert.asp)

### Manual de SQL

[http://www.w3schools.com/sql/sql\\_delete.asp](http://www.w3schools.com/sql/sql_delete.asp)  
[http://www.w3schools.com/sql/sql\\_update.asp](http://www.w3schools.com/sql/sql_update.asp)  
[http://www.w3schools.com/sql/sql\\_func\\_max.asp](http://www.w3schools.com/sql/sql_func_max.asp)

### Manual do Magic

- **Resumido**  
Páginas 4 à 7 - [https://www.wizards.com/magic/rules/PT\\_MTGM12\\_Rulebook\\_web.pdf](https://www.wizards.com/magic/rules/PT_MTGM12_Rulebook_web.pdf)
- **Compreensivo**  
Capítulos 2 e 3 - [http://www.magicjebb.com.br/site/downloads/Magic\\_Core\\_Rules.pdfY](http://www.magicjebb.com.br/site/downloads/Magic_Core_Rules.pdfY)

### YouTube

“Passagem de valores p. url”, disponível em <https://www.youtube.com/watch?v=wo2aWbY9OKI>  
“Time and Date”, disponível em <https://www.youtube.com/watch?v=tbob0uSdSUg>  
“Corr. Erros de acentuação”, disponível em <https://www.youtube.com/watch?v=FRCljkAq54I>  
“Warning: Cannot modify”, disponível em <https://www.youtube.com/watch?v=OYyVpWn638Q>

### Wikipédia

<https://pt.wikipedia.org/wiki/SHA-2>  
[https://pt.wikipedia.org/wiki/Sistema de gerenciamnto de banco de dados](https://pt.wikipedia.org/wiki/Sistema_de_gerenciamnto_de_banco_de_dados)  
<https://pt.wikipedia.org/wiki/SQL>  
<https://pt.wikipedia.org/wiki/IDE>

## **Anexos**

# Anexo A - Requisitos

## Requisitos não-funcionais

### Requisitos de domínio:

#### Utilizador

- Um utilizador é quem faz uso da aplicação...
- Um utilizador é caracterizado por:
- Nome/*username*, foto, *e-mail*.

#### Jogador

- Um jogador é um utilizador com privilégios para colecionar cartas, construir baralhos de acordo com o seu modo individual de jogo para tentar vencer o baralho adversário.

#### Administrador

- Um administrador é quem faz a gestão do *site*: gestão de utilizadores, cartas, comentários, cores e tipos das cartas.

### Requisitos Técnicos

- O sistema deve oferecer a possibilidade de ser acedido remotamente, por meio da Internet.
- Só os utilizadores registados e autenticados podem beneficiar de todas as funcionalidades do *site*.

## Requisitos Funcionais do BackOffice

### Administrador - O sistema deve permitir:

- Adicionar, remover e editar cartas, cores e tipos das cartas.
- Banir/eliminar utilizador e editar os dados do mesmo.

## Requisitos Funcionais da Aplicação

### Utilizador - O sistema deve permitir:

- Efetuar a autenticação/*login* introduzindo o respectivo *e-mail* e *password*.
- Recuperar a palavra passe por meio do *e-mail*.
- registo de novos utilizadores no sistema pelo introdução do *e-mail*, *username* e *password*.
- Alteração da palavra passe.
- *logout* deve significar deixar de estar autenticado.
- Adicionar, remover, listar, editar dados pessoais.

**Jogador** - O sistema deve permitir:

- Coleccionar cartas e edita-las quando elas fazem parte da sua coleção.
- Gerar baralhos, de maneira automática ou não.
- Comentar cartas e apagar os seus comentários.

### Tipos de baralho

- **Actuais**

São os únicos baralhos que o jogador pode manipular e utilizar no jogo.

- **Gerados**

Podem ser apagados, visualizados e ativados.

- **Antigos**

Podem ser visualizados e reativados.

Nota: Os baralhos gerados e antigos devem ser ativados/passados para a lista de baralhos actuais para poderem ser editados.

## Funcionalidades públicas do *site*

- **Registo**

Deve solicitar-se ao utilizador o preenchimento de um formulário que permite o sistema obter do mesmo o *username*, o *e-mail* e a *password*. Assim, o *site* tem o seguinte formulário de registo:

### Registo utilizador

Username	<input type="text" value="Virginio Andrade"/>
Password	<input type="password" value="*****"/>
Email	<input type="text" value="exemplo@hotmail.com"/>
<input type="button" value="Gravar"/>	

Figura 12 - Formulário do Registo

Por questões de segurança, garante-se que o utilizador optará por uma *password* de pelo menos oito caracteres e um *e-mail* inexistente na base de dados do programa.

- **Login**

Deve solicitar-se ao utilizador o seu *e-mail* e a respetiva *password* que passarão por uma verificação com a finalidade de garantir que o utilizador está realmente registado no sistema. Assim, o *site* tem o seguinte formulário de *Login*:

### Iniciar sessão

Email	<input type="text" value="exemplo@hotmail.com"/>
Password	<input type="password" value="*****"/>
<input type="button" value="Login"/>	

Figura 13 - Formulário de Login



# Funcionalidades do utilizador

## Geral

- **Editar Perfil**

A página apresenta a foto do utilizador, um formulário com os dados editáveis como o seu *username*, *e-mail*, um campo de nome “Foto” que permite o *upload* da foto do perfil, um outro formulário para a mudança da *password* caso o utilizador pretenda, com três campos: “Password nova”, “Password nova (confirmação)” e “Password actual”. Para este formulário, garante-se que os dois primeiros campos têm o mesmo conteúdo e são preenchidos com pelo menos oito caracteres, segundo os critérios do sistema com relação a *password*. Os dois formulários são submetidos individualmente. A figura 14 abaixo ilustra o que está explicado acima.

### Ver/editar dados Perfil



<b>Username</b>	<input type="text" value="Mila Andrade"/>
<b>Email</b>	<input type="text" value="mila@hotmail.com"/>
<b>Foto</b>	<input type="button" value="Escolher ficheiro"/> Nenhum ficheiro selecionado
<input type="button" value="Gravar"/>	

<b>Password nova</b>	<input type="password" value="*****"/>
<b>Password nova (confirmação)</b>	<input type="password" value="*****"/>
<b>Password actual</b>	<input type="password" value="*****"/>
<input type="button" value="Alterar"/>	

Figura 14 - Formulário para ver/editar dados do utilizador

## Lista de cartas do jogo

### Jogador

- **Ver carta**

Apresentam-se os dados da carta e, depois de se garantir que o utilizador é um jogador, não permite-se que o mesmo os edite. Mas como os dados da carta incluem os comentários sobre a mesma... então esta página permite que o jogador comente a carta ou até mesmo apaga os seus comentários. Quando a carta já faz parte da lista do jogador, no lugar desta opção é apresentada a seguinte informação: “Existente na tua lista”.

- **Associar carta a minha lista**

Ao escolher esta opção, o sistema apresenta uma página com a imagem da carta e um formulário com o nome (não editável) da carta e um outro campo “Quantidade” onde o jogador pode especificar a quantidade de cópias da carta que pretende associar a sua lista de cartas.

### Administrador

- **Adicionar carta**

Para esta ação, o sistema apresenta ao administrador um formulário para o preenchimento dos dados relacionados a nova carta, dados como a foto (por meio de um *upload*), nome, tipo e etc.

- **Editar/ver carta**

Garante-se que se trata de um administrador, apresentam-se os dados da carta que podem ser editados. Para além de poder comentar a carta, o administrador pode apagar todos os comentários que não promovem o sucesso da aplicação, independentemente do utilizador.

- **Apagar carta**

Acrescentando ao que já se abordou acima sobre este tema, esta ação apaga a carta (e os seus respetivos comentários) do sistema, ou seja, apaga a carta até mesmo dos baralhos e listas de cartas dos jogadores.

### Jogador

#### Minha lista de cartas - Lista de cartas do utilizador

- **Ver carta**

Dá um tratamento específico por se tratar de um Jogador, conforme já foi frisado acima.

- **Editar carta**

Apresenta-se a foto da carta e um formulário com o nome (não editável) da carta e um outro campo “Quantidade” onde o jogador pode especificar a quantidade de cópias da carta que pretende associar a sua lista de cartas.

- **Associar carta a baralho**

Apresenta-se a foto da carta e um formulário com o nome (não editável), o campo “Baralho” e um campo “Quantidade” onde o jogador deve especificar o número de cópias da carta que pretende ter no seu baralho.

- **Apagar carta**

Incrementando ao que já se abordou acima a respeito deste tema, esta ação apaga a carta do histórico do jogador no sistema, ou seja, apaga a carta da sua coleção e dos seus baralhos. Também apaga os seus comentários sobre a carta.

## Lista de baralhos

### Geral

- **Ver**

Deve mostrar organizadamente numa tabela o conteúdo do baralho, nomeadamente as informação mais relevantes de uma carta (p.e. “cor”, “tipo”, “nome”).

- **Ver carta**

O jogador deve poder ver os dados das cartas (incluindo os comentários) comenta-las e apagar os seus respetivos comentários.

- **Apagar carta**

Incrementando ao que já se abordou acima a respeito deste tema, esta ação apaga a carta do baralho em questão.

### Actuais

- **Criar Baralho**

Por meio de um formulário com o campo “Nome do baralho” e “Descrição”, o jogador especifica as informações necessárias para criar um novo baralho, onde o utilizador pode adicionar cartas.

- **Editar**

Por meio de um formulário com o campo “Nome” e “Descrição”, onde o jogador pode editar os seus baralhos actuais.

- **Arquivar**

Quando arquivado, o baralho antigo passa a ser um “baralho antigo”.

### Antigos

- **Reativar**

Quando reativado, o baralho antigo passa a ser um “baralho actual”.

### Gerados

- **Ativar**

Quando ativado, o baralho gerado passa a ser um “baralho actual”.

- **Configurações**

Apresenta-se o formulário do gerador de baralhos preenchido com os dados editáveis que foram usados para o gerar. O jogador ainda deve poder gerar mais baralhos por meio destes dados.

- **Apagar**

Depois da confirmação do jogador, apaga o baralho e todo o seu conteúdo.

## Gerar baralhos

Por meio de um formulário específico, o jogador deve especificar as características do baralho. Caso o jogador não faça o preenchimento correto do formulário ou, por exemplo, não tenha cartas suficiente deve também receber mensagens de erros específicos.

O campos do formulário (input) do gerador de baralhos devem ser os seguintes:

- **Escolha as cores:**
- **Quantidade de cartas Baralho(s)**
- **Escolha o tipo de jogo**
- **Custo máx. Aceitável**
- **Porcentagem de terrenos (1-100)**
- **Usar terrenos de outras cores?**
- **Núm. mínimo cópias cartas (1-4)**
- **Núm. máx. Cóp. Cartas (nMinCoCartas-4)**
- **Número máximo de baralhos (1-3)**

## Tipos de jogo

- **Rápido**

Quando o utilizador escolhe jogo rápido, ao selecionar as cartas, a sequência de prioridade é a seguinte: as cartas “baratas” (com o custo inferior que três) - custo médio (custo de 3 a 5) - cartas caras (custo maior que 5).

- **Misto**

A sequência de seleção das cartas é a seguinte: cartas com custo médio - cartas baratas - cartas caras.

- **Lento**

Sequência: cartas caras - cartas com custo médio - cartas baratas.

Exemplo: suponhamos que o utilizador tem na sua coleção: 2 cartas baratas, 1 com o custo médio e 1 carta cara e pretende gerar um baralho com 3 cartas. Se escolher o jogo rápido, ele terá no seu baralho o seguinte: 2 cartas baratas e 1 de custo médio. Se escolher o jogo misto, ele terá no seu baralho o seguinte: 1 carta de custo médio e 2 cartas baratas. Se escolher o jogo lento, ele terá nos seus baralhos o seguinte: 1 carta cara, 1 de custo médio e 1 carta barata.

## Administrador

### Lista de utilizadores

- **Editar/saber mais**

Os administradores devem ter o privilégio de ver/editar os dados pessoais de todos os utilizadores, para banir determinado utilizador ou evitar que o mesmo promova atitudes que perturbam o bom funcionamento do *site*.

- **Banir**

Para além de banir-mos os utilizadores por meio do impedimento de acederem ao *site* e etc., podemos fazer isto por eliminar-mos todo o histórico do utilizador do sistema.

## **Lista de cores (das cartas)**

- **Adicionar cor**

Uma página com um formulário onde o administrador introduz os dados da nova cor.

- **Editar**

Para editar um determinada cor, o sistema fornece uma página com o formulário ideal.

- **Apagar**

Após a confirmação do administrador, elimina-se todo o histórico da cor no sistema.

## **Lista de tipos (de cartas)**

- **Adicionar tipo**

Uma página com um formulário onde administrador introduz os dados do novo tipo.

- **Editar**

Para editar um determinado “tipo”, o sistema fornece uma página com o formulário ideal.

- **Apagar**

Após a confirmação do administrador, elimina-se todo o histórico do “tipo” no sistema.

# Anexo B - Manual do Programador

## Introdução

Nesta secção vão ser apresentados pormenores técnicos da implementação do *site*.

## Paginação

A paginação é feita pela função “`paginacaoComplexa()`” que como todas as outras funções encontra-se no ficheiro “`funcoes.php`”. A função recebe como argumento o mínimo e o máximo que serão usados na cláusula LIMIT (do SQL) para apresentar apenas um determinado número de linhas de uma tabela. Esta função também recebe o nome do ficheiro que apresenta resultados/listas organizadas numa tabela por meio de uma *query* que seleciona criteriosamente da base de dados os elementos a serem apresentados. Como a paginação também se faz sentir mesmo quando ordenamos, filtramos e fazemos pesquisas... então a função também recebe como argumentos elementos identificativos como: cor, tipo, nome do que se pretende pesquisar e o “TipoOrdenacao” que serão depois passados por URL e recebidos na página em questão pela função “`$_GET`” ou “`$_POST`” (quando se trata da primeira pesquisa) do PHP. O que também é recebido pela função é o id para, por exemplo, quando se tratar de listas de resultados pertencente a uma entidade específica. O utilizador beneficia-se da paginação por meio de um menu (abaixo de cada lista de resultados) com 4 opções, conforme podemos ver na figura 15 abaixo:

[Primeira](#) | [Anterior](#) | [Próxima](#) | [Última](#)

Figura 15 - Menu Paginação

## Ordenação, pesquisa e filtro

Na página em questão, de acordo com o conteúdo que se apresenta, disponibiliza-se ao utilizador uma ou duas caixinha(s)/tabela(s) onde o mesmo pode comodamente especificar o tipo de ordenação (p.e. “custo carta - ascendente ou descendente”, “nome”), a cor, nome ou o tipo da(s) carta(s) que pretende encontrar. Estes valores serão usados na *query* em SQL para satisfazer o desejo do utilizador. Seguidamente, poderemos observar um exemplo na figura 16.

Cor	Escolha... ▼
Tipo	Escolha... ▼
Procurar	<input type="text"/>
<input type="button" value="Filtrar"/>	

Tipo de ordenação	Escolha... ▼
<input type="button" value="Ordenar"/>	

Figura 16 - Caixinhas para Ordenação, pesquisa e filtro

## Rodapé (Footer)

É um componente utilizado pelo *software* com a finalidade de permitir uma ação rápida por parte do utilizador, permitindo o acesso a funções/operações mais corriqueiras do programa. Esta componente gerida por meio do ficheiro “footer.php” que apresenta organizadamente um “*link*” (para cada operação) onde o utilizador pode clicar para aceder respectivamente as funcionalidades da aplicação. Como poderemos constatar nas figuras 17 e 18 abaixo, os administradores e os jogadores têm **Footer** diferente um do outro.

[Lista Cartas](#) | [Lista Utilizadores](#) | [Página principal](#) | [Editar Perfil](#) | [Lista Cores](#) | [Lista Tipos](#)

[Logout](#)

Figura 17 - Footer Administrador

[Baralhos actuais](#) | [Baralhos antigos](#) | [Baralhos gerados](#) | [Gerar Baralho\(s\)](#) | [Página principal](#) | [Lista cartas - Jogo](#) | [Editar Perfil](#) | [Minhas Cartas](#)

[Logout](#)

Figura 18 - Footer Jogador

## Apagar/eliminar carta, baralho e utilizador

Apresenta-se uma janela (em Javascript) de confirmação do ato para garantir que o utilizador tem noção da ação que pretende realizar. O utilizador pode clicar em “Cancelar” para retroceder do ato de eliminar/apagar certos elementos ou pode clicar em “Ok” para continuar a operação, conforme a figura 19 abaixo:

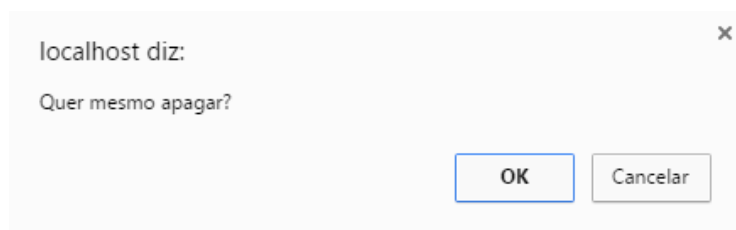


Figura 19 - Janela de confirmação para eliminar algo

## Listagem de ficheiros

Esta secção apresenta uma tabela com a lista dos nomes (e uma respectiva pequena descrição) de todos os ficheiros que contém o código fonte do *software*.

Nome do ficheiro	Descrição
adiciona-cor.php	Fornece o formulário para adicionar uma nova cor de carta no sistema.
adiciona-tipo.php	Fornece o formulário para adicionar um novo tipo de carta no sistema.
apaga-carta-baralho.php	Recebe o <i>id</i> da carta e o <i>id</i> baralho e a apaga do baralho.
arquivo-baralho.php	Recebe o <i>id</i> do baralho e do jogador e passa o baralhos aos seus baralhos antigos.
associa-carta-administrador.php	Fornece o formulário para adicionar uma nova carta no sistema.
associa-carta-baralho.php	Recebe o <i>id</i> da carta e o do baralho e associa a carta ao baralho.
associa-carta-utilizador.php	Recebe o <i>id</i> da carta e o do jogador e associa a carta ao jogador.
ativar-baralho.php	Recebe o <i>id</i> do baralho gerado e o <i>id</i> do jogador e associa o baralho aos seus baralhos actuais.
baralhos-actuais-utilizador.php	Apresenta a lista dos baralhos actuais do jogador e as respectivas opções sobre os mesmos.
baralhos-antigos-utilizador.php	Apresenta a lista dos baralhos antigos do jogador e as respectivas opções sobre os mesmos.
baralhos-gerados-utilizador.php	Apresenta a lista dos baralhos gerados do jogador e as respectivas opções sobre os mesmos.
cartas-utilizador.php	Apresenta a lista de cartas do jogador e as respectivas opções sobre os mesmos.
comentar-carta.php	Permite o utilizador comentar as cartas.
config.php	Contém o código fonte da configuração/conexão da base de dados.
cria-baralho.php	Fornece o formulário para a criação de baralho.
edita-baralho.php	Fornece o formulário ao jogador para o mesmo editar o seu baralho.
edita-carta.php	Fornece um formulário para editar carta.
edita-carta-utilizador.php	Fornece um formulário ao jogador para que o mesmo edite a sua carta.
edita-cor.php	Fornece um formulário para editar os dados da cor de carta.
edita-perfil.php	Fornece um formulário para que o utilizador edite ou observe os seus dados ou para que o administrador edite ou observe dados pessoais dos outros utilizadores.
edita-tipo.php	Fornece um formulário para editar os dados do tipo de carta.
elimina-utilizador.php	Elimina o utilizador do sistema.
footer.php	Fornece o rodapé para o jogador e o administrador
funcoes.php	Contém todas as funções que são chamadas nos outros ficheiros.
gerador-baralhos.php	Fornece o formulário do gerador de baralhos.
index.php	Fornece o formulário de <i>Login</i> .
lista-cartas.php	Apresenta a lista de cartas do sistema e as devidas opções aos



	jogadores e administradores.
lista-cores.php	Apresenta a lista de cores das cartas e as respectivas opções sobre as mesmas.
lista-tipos.php	Apresenta a lista dos tipos de cartas e as respectivas opções sobre os mesmos.
lista-utilizadores.php	Apresenta a lista de utilizadores e as respectivas opções sobre os mesmos.
pagina-inicial.php	Contém o código fonte da página inicial.
pagina-utilizador.php	Contém o código fonte da página do utilizador.
recuperar-password.php	Fornece o formulário para o recuperação da <i>password</i> (o utilizador recebe um e-mail com um <i>link</i> e um código).
registo.php	Fornece o formulário para o registo do utilizador.
reset-password.php	Fornece um formulário para o <i>reset</i> da <i>password</i> .
ver-baralho.php	Apresenta a lista de cartas do baralho e as respectivas opções sobre as mesmas.

# Glossário

IDE (*Integrated Development Environment*) - Ambiente de Desenvolvimento Integrado, um ambiente integrado para desenvolvimento de *software*.

*Query* - Uma pergunta ou um pedido de informação sobre algo. Uma *Query* em SQL é uma instrução cujo a execução visa a obtenção de uma informação da base de dados.

SGDB - Sistema de Gestão de Bases de Dados.

SQL (*Structured Query Language*) - É uma linguagem padrão para fazer pesquisa na base de dados.

TCG - *Trading Card Game*.