



Escola Superior de Tecnologia e Gestão  
Licenciatura em Engenharia Informática

Desenvolvimento de Aplicações Web

Projeto – Fase de Implementação

My Anime Collection

Martinho José Novo Caeiro  
Paulo António Tavares Abade



Beja, janeiro de 2025

**Instituto Politécnico de Beja**  
**Escola Superior de Tecnologia e Gestão**  
**Licenciatura em Engenharia Informática**

**Desenvolvimento de Aplicações Web**

**Projeto – Fase de Implementação**

**My Anime Collection**

**Martinho José Novo Caeiro**  
**Paulo António Tavares Abade**

**DOCENTE**

**Professor Luís Carlos Bruno**

Trabalho realizado no âmbito da unidade curricular Desenvolvimento de  
Aplicações Web (Bruno, 2025)

**Beja, janeiro de 2025**

# Índice

<b>1</b>	<b>Introdução</b>	<b>3</b>
<b>2</b>	<b>Análise do Sistema</b>	<b>3</b>
<b>3</b>	<b>Caracterização dos Atores</b>	<b>3</b>
3.1	Persona N <sup>o</sup> 1 - Miguel Silva . . . . .	4
3.2	Persona N <sup>o</sup> 2 - Catarina Silvestre . . . . .	4
<b>4</b>	<b>Diagrama de Casos de Uso</b>	<b>4</b>
4.1	Caso de Uso N <sup>o</sup> 1 - Consultar Animes . . . . .	6
4.2	Caso de Uso N <sup>o</sup> 2 - Adicionar Animes a uma Lista . . . . .	6
4.3	Caso de Uso N <sup>o</sup> 3 - Consultar o Perfil de outro Utilizador . . . . .	7
4.4	Caso de Uso N <sup>o</sup> 4 - Avaliar uma Lista de Animes de outro Utilizador	7
4.5	Caso de Uso N <sup>o</sup> 5 (Partilhado) - Visualizar os animes com melhor avaliação . . . . .	8
<b>5</b>	<b>Desenho do Sistema</b>	<b>9</b>
5.1	Modelação da Base de Dados . . . . .	9
5.2	Diagrama E/R . . . . .	9
5.3	Modelo Relacional . . . . .	9
5.4	Modelo Físico . . . . .	10
<b>6</b>	<b>Modelação de Interfaces Gráficas com o Utilizador</b>	<b>11</b>
6.1	Storyboard(s) . . . . .	11
6.2	Interfaces do Caso de Uso 1 . . . . .	12
6.3	Interfaces do Caso de Uso 2 . . . . .	12
6.4	Interfaces do Caso de Uso 3 . . . . .	13
6.5	Interfaces do Caso de Uso 4 . . . . .	13
6.6	Interfaces do Caso de Uso 5 . . . . .	14
6.7	Protótipo de Média Fidelidade . . . . .	15
<b>7</b>	<b>Melhorias efetuadas na Análise e Desenho do Sistema</b>	<b>16</b>
<b>8</b>	<b>Implementação</b>	<b>16</b>
8.1	Arquitetura do Sistema . . . . .	16
8.2	Tecnologias Usadas . . . . .	18
8.3	Desenvolvimento da API . . . . .	18
8.4	Especificação da Interface . . . . .	19

8.5	Decisões de Implementação . . . . .	19
8.6	Principais Casos Relevantes de Programação . . . . .	20
<b>9</b>	<b>Desenvolvimento da App Frontend/MVC</b>	<b>20</b>
9.1	Decisões de Implementação . . . . .	20
9.2	Principais Casos Relevantes de Programação . . . . .	20
<b>10</b>	<b>Conclusão e Trabalho Futuro</b>	<b>21</b>
	<b>Bibliografia</b>	<b>22</b>

## Índice de Figuras

1	Diagrama de Casos de Uso . . . . .	5
2	Modelo Físico . . . . .	10
3	Interface do Caso de Uso 1 . . . . .	12
4	Interface do Caso de Uso 2 . . . . .	12
5	Interface do Caso de Uso 3 . . . . .	13
6	Interface do Caso de Uso 4 . . . . .	13
7	Interface do Caso de Uso 5 . . . . .	14
8	Protótipo de Média Fidelidade . . . . .	15
9	Arquitetura do Sistema . . . . .	17

## 1 Introdução

Neste relatório iremos fazer a Implementação de uma aplicação de gestão de media, que tem como objetivo ajudar a organizar o conteúdo dos fãs de anime, ao ser possível categorizar os animes entre: Por visualizar, a visualizar, visualizados e também ser possível dar uma classificação aos mesmos ou às listas de outros utilizadores. Irá ser feita uma API que irá conter todos os animes e os seus dados relacionados, que será acompanhada de uma base de dados que irá guardar as informações dos utilizadores. Neste relatório será dividido em três grandes partes, sendo estas a Análise e Desenho do Sistema cujas quais já foram desenvolvidas no relatório anterior, seguida da parte de Implementação onde serão referidos os metodos de construção utilizados para fazer os casos de usos referidos na Analise e Desenho.

## 2 Análise do Sistema

Para analisar o sistema, é necessário caracterizar os atores que realizarão as tarefas neste sistema, e quais serão as tarefas que serão realizadas pelos mesmos, para isso serão utilizados os Casos de Uso, considerando a notação UML.

## 3 Caracterização dos Atores

Neste sistema o mesmo utilizador terá a posição de dois atores diferentes, sendo estes: **Visualizador** e **Gestor de Listas**.

Inicialmente, todos os utilizadores assumem o papel de “Visualizador” e depois irão assumir temporariamente o papel de gestor de listas. Para explicar como funciona, será feito o uso de Personas, que são personagens fictícios para representar o futuro utilizador deste sistema.

### **3.1 Persona Nº1 - Miguel Silva**

O Miguel tem 18 anos e entrou no MyAnimeCollection para descobrir novos animes. Ao visualizar o anime “One Piece” e ter-se interessado pelo mesmo, adicionou este anime à sua lista “A Visualizar”.

### **3.2 Persona Nº2 - Catarina Silvestre**

A Catarina tem 19 anos e descobriu o MyAnimeCollection pelo Miguel, e então quis verificar o perfil dele para ver que animes ele tem nas suas listas. Ao aceder o perfil do Miguel, viu as listas que ele tinha disponíveis no seu perfil. Após ver a lista de “A Visualizar” do Miguel, avaliou-a com 5 estrelas.

## **4 Diagrama de Casos de Uso**

Para criar o diagrama de casos de uso, foi utilizado a plataforma draw.io, respeitando a notação de casos de uso do UML.

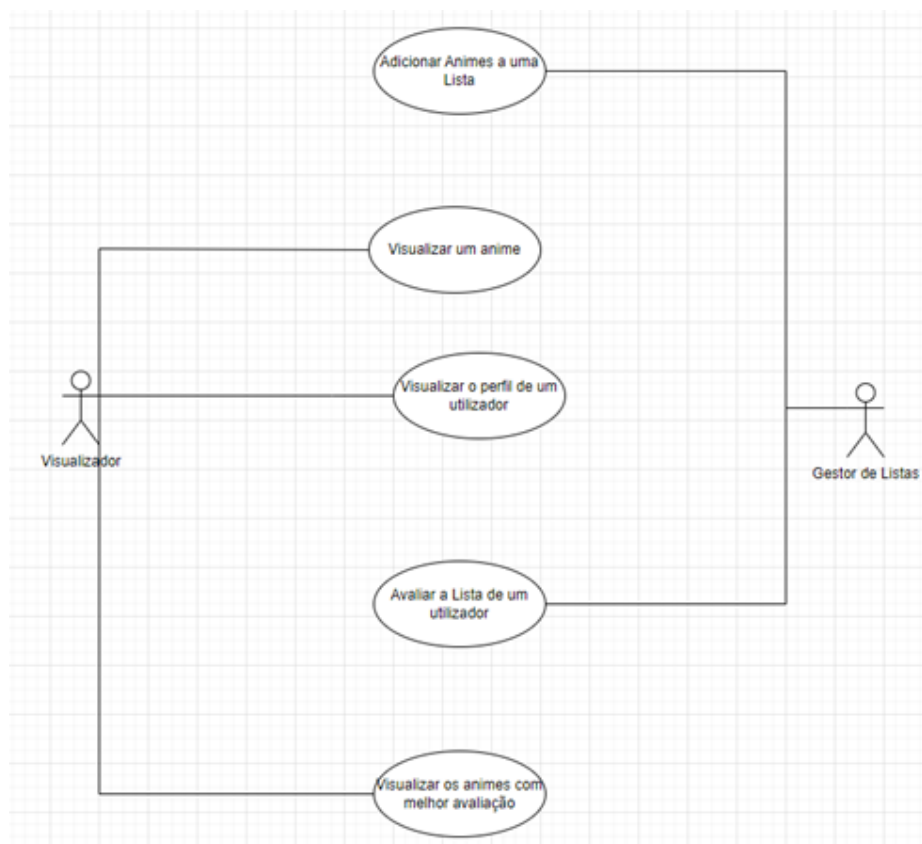


Figura 1: Diagrama de Casos de Uso

#### 4.1 Caso de Uso Nº1 - Consultar Animes

<b>Descrição</b>	Permite ao utilizador pesquisar informações sobre um anime específico, como sinopse, episódios, etc.
<b>Atores</b>	Visualizador
<b>Pré-Condições</b>	O utilizador deve estar conectado à internet e estar com a sessão iniciada
<b>Pós-Condições</b>	O anime é exibido com detalhes como sinopse, episódios, etc.
<b>Cenário Principal</b>	O Visualizador pesquisa um anime pelo nome, e o sistema retorna as informações detalhadas
<b>Situação de Falha</b>	Mostra um erro que identifica se o anime não foi encontrado ou foi um erro de conexão

#### 4.2 Caso de Uso Nº2 - Adicionar Animes a uma Lista

<b>Descrição</b>	Permite ao utilizador adicionar um anime à sua lista pessoal de animes para organizar seu progresso de visualização
<b>Atores</b>	Gestor de Lista
<b>Pré-Condições</b>	O utilizador deve estar conectado à internet e estar com a sessão iniciada
<b>Pós-Condições</b>	O anime é adicionado à lista com sucesso
<b>Cenário Principal</b>	O Gestor de Lista navega até o anime desejado e seleciona a opção "Adicionar à lista", categorizando-o (Por Visualizar, A Visualizar, Visualizados, ou uma criada pelo utilizador)
<b>Situação de Falha</b>	Mostra um erro que identifica uma falha na adição à lista por erro de sistema ou da conexão



#### 4.3 Caso de Uso Nº3 - Consultar o Perfil de outro Utilizador

<b>Descrição</b>	Permite visualizar o perfil e listas de outros utilizadores
<b>Atores</b>	Visualizador
<b>Pré-Condições</b>	O perfil do outro utilizador deve ser acessível.
<b>Pós-Condições</b>	As informações do perfil e listas são exibidas
<b>Cenário Principal</b>	O Visualizador busca pelo nome de outro utilizador e, se o perfil for acessível, o sistema exibe as informações do perfil, como listas e avaliações
<b>Situação de Falha</b>	Mostra um erro que identifica se a falha foi por inexistência do utilizador ou falha de conexão

#### 4.4 Caso de Uso Nº4 - Avaliar uma Lista de Animes de outro Utilizador

<b>Descrição</b>	Permite ao utilizador avaliar as listas de animes de outros utilizadores
<b>Atores</b>	Gestor de Lista
<b>Pré-Condições</b>	O utilizador deve estar conectado e estar com a sessão iniciada e a lista de outro utilizador deve ser acessível
<b>Pós-Condições</b>	A avaliação é submetida e exibida
<b>Cenário Principal</b>	O utilizador acessa a lista de outro utilizador, avalia e submete a avaliação, que é registada com sucesso
<b>Situação de Falha</b>	Mostra um erro que identifica uma falha ao enviar a avaliação devido a erro no sistema ou na conexão

#### 4.5 Caso de Uso Nº5 (Partilhado) - Visualizar os animes com melhor avaliação

<b>Descrição</b>	Exibe uma lista dos animes mais bem avaliados com base nas classificações de todos os utilizadores
<b>Atores</b>	Visualizador (não precisa estar com sessão iniciada)
<b>Pré-Condições</b>	Deve haver dados suficientes de avaliações para gerar uma lista
<b>Pós-Condições</b>	Lista dos animes mais bem avaliados é exibida
<b>Cenário Principal</b>	O Visualizador acessa a página de rankings, e o sistema exibe os animes com as melhores avaliações
<b>Situação de Falha</b>	Mostra um erro que identifica um erro de sistema

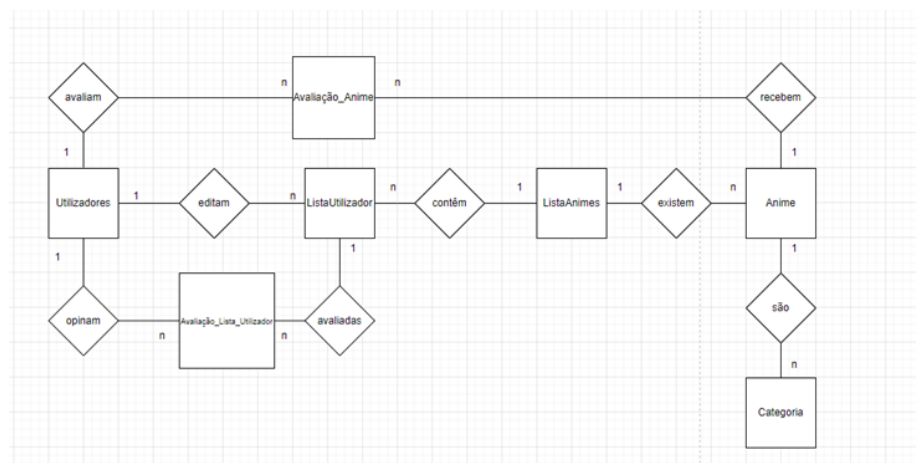
## 5 Desenho do Sistema

### 5.1 Modelação da Base de Dados

Nesta parte será mostrada como será o sistema, considerando as suas tabelas e relações entre elas, e depois será mostrado o modelo físico que indica as relações entre as tabelas, os atributos e o tipo dos mesmos.

### 5.2 Diagrama E/R

Este é o diagrama de entidade-relação utilizado pelo sistema do MyAnimeCollection:



### 5.3 Modelo Relacional

- UserList (userlist\_id, user\_id, animelist\_id, nome, description)
- AnimeList (animelist\_id, anime\_id)
- UserListAvaliation (userlistavaliation\_id, user\_id, id\_listaUtilizador, avali-  
ation)

- AnimeAvaliation (animelistavaliation\_id, user\_id, anime\_id, avaliation)
- User (user\_id, nome, age, email, password, biography)
- Anime (anime\_id, synopsis, categories, n\_episodes, n\_seasons)

## 5.4 Modelo Físico

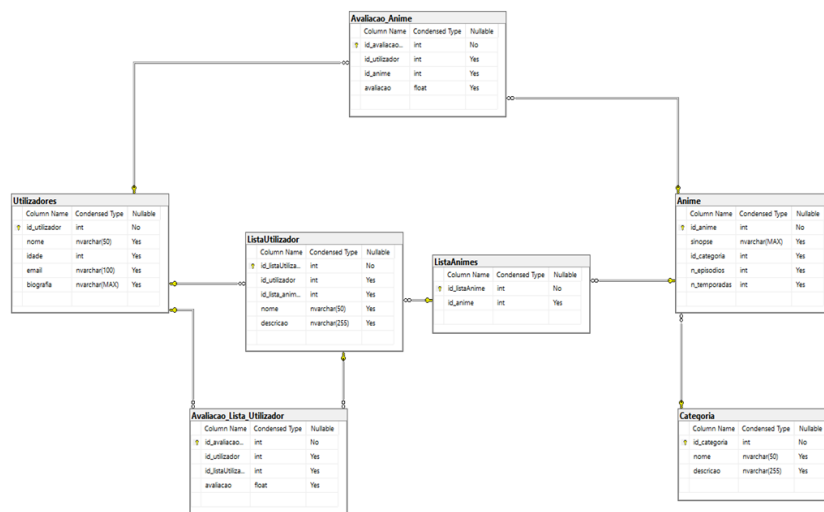
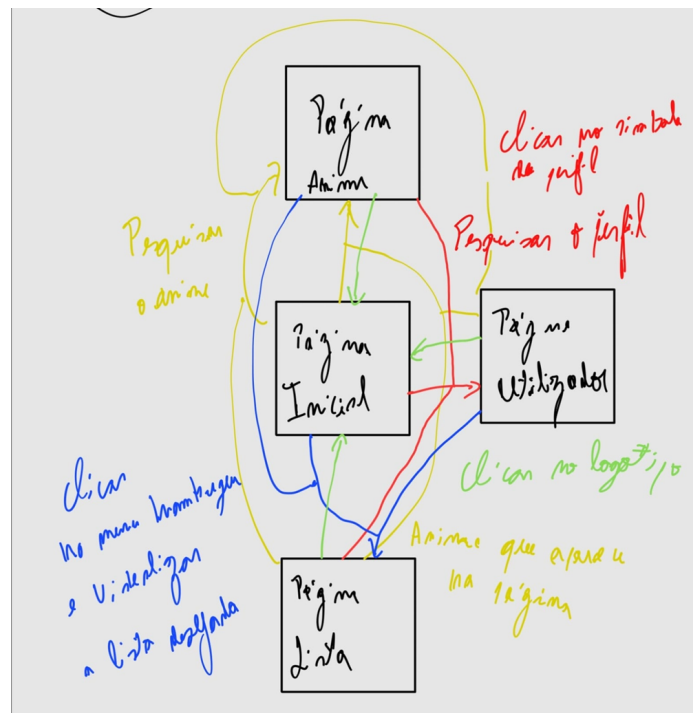


Figura 2: Modelo Físico

## 6 Modelação de Interfaces Gráficas com o Utilizador

### 6.1 Storyboard(s)

Neste Storyboard podemos visualizar as movimentações entre páginas principais. Existem mais opções que podem ser feitas nas páginas, porém serão apenas exploradas posteriormente.



## 6.2 Interfaces do Caso de Uso 1

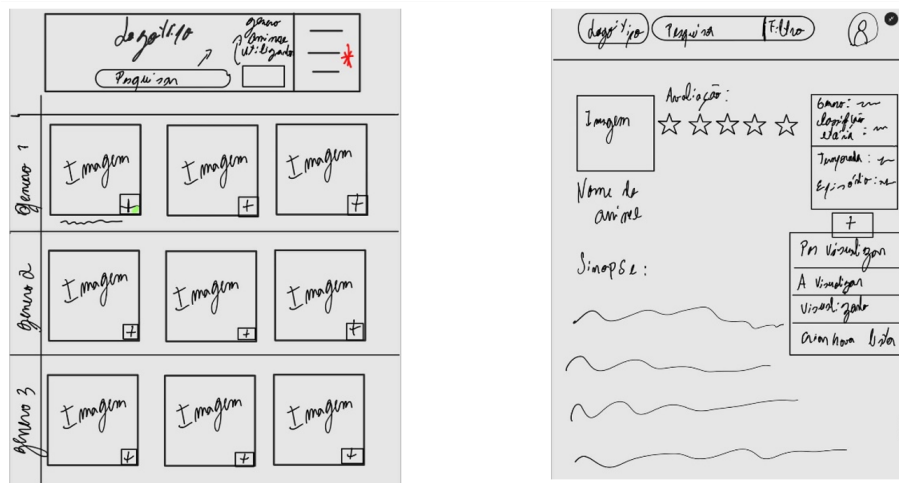


Figura 3: Interface do Caso de Uso 1

## 6.3 Interfaces do Caso de Uso 2

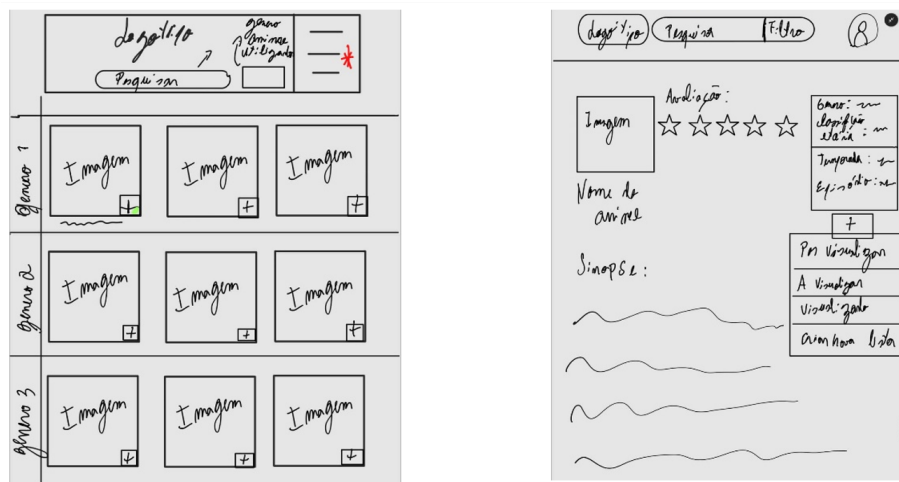


Figura 4: Interface do Caso de Uso 2

## 6.4 Interfaces do Caso de Uso 3

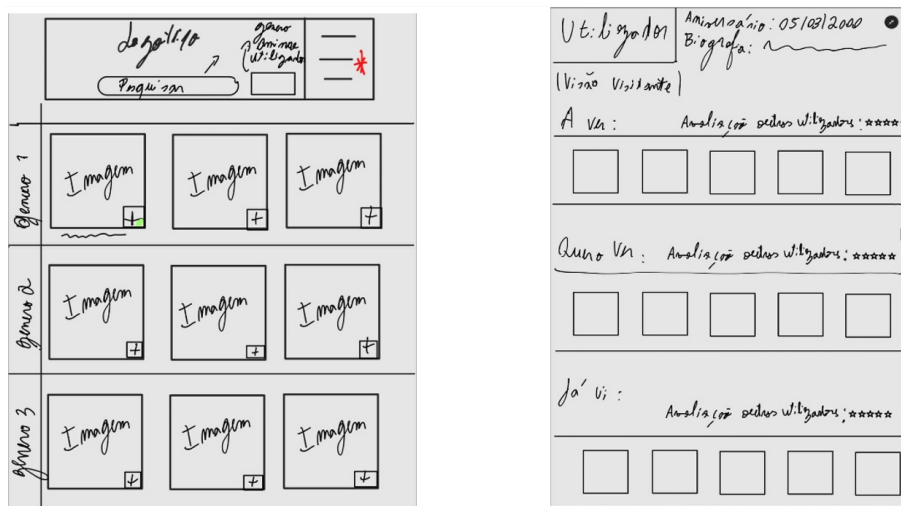


Figura 5: Interface do Caso de Uso 3

## 6.5 Interfaces do Caso de Uso 4

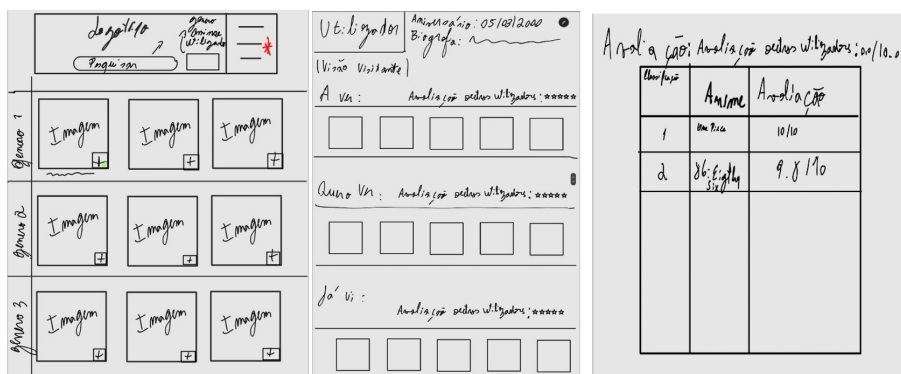


Figura 6: Interface do Caso de Uso 4

## 6.6 Interfaces do Caso de Uso 5

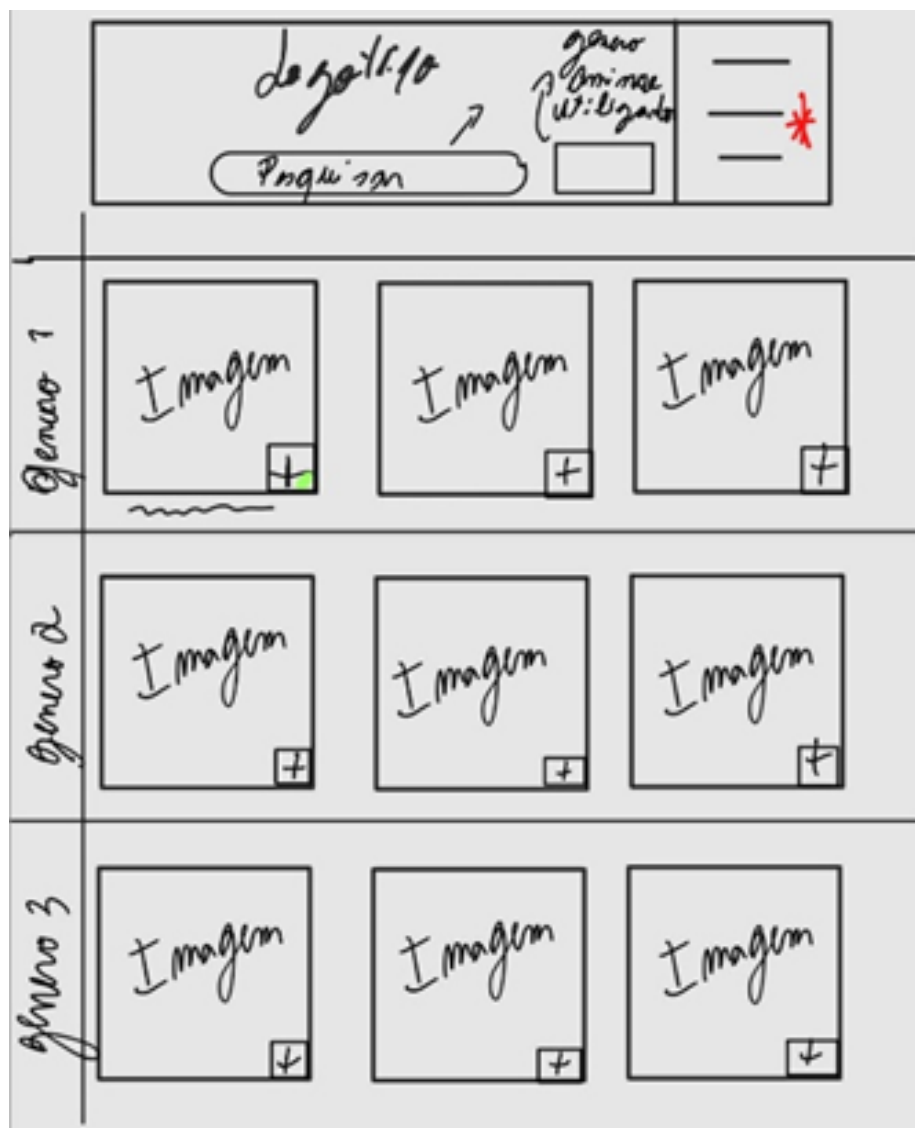


Figura 7: Interface do Caso de Uso 5



## 6.7 Protótipo de Média Fidelidade

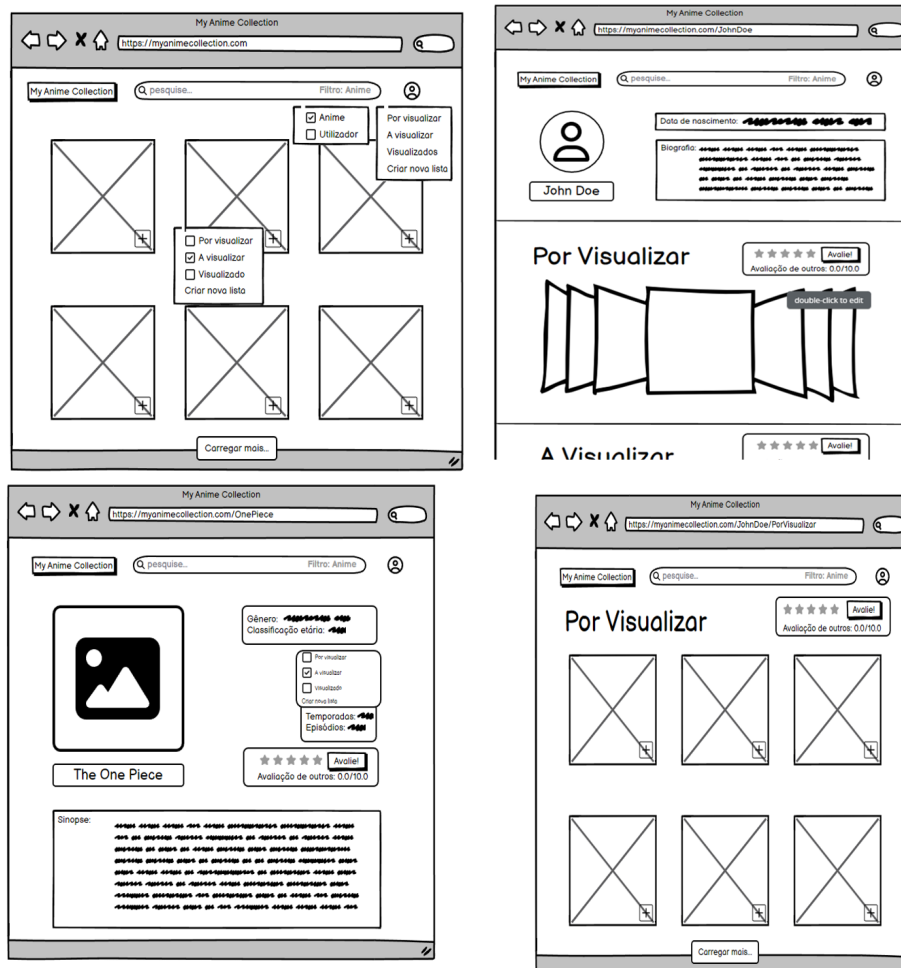


Figura 8: Protótipo de Média Fidelidade

## **7 Melhorias efetuadas na Análise e Desenho do Sistema**

O Caso de Uso 5 foi aprofundado, agora sendo possível visualizar animes por uma data específica em vez de um período predefinido. Esta melhoria foi feita para que o utilizador possa ter uma maior flexibilidade na pesquisa de animes com melhor avaliação. Esta implementação torna o sistema mais completo e com mais funcionalidades, podendo assim ser utilizado como uma aplicação de Sistemas de Informação, já que possui uma funcionalidade CUBO que permite ao utilizador visualizar os dados de uma forma mais eficiente, na questão de tempo. É limitado a animes, porém pode ser expandido para outras especificações, como por exemplo, o género do anime, ou o estúdio que o produziu com melhor avaliação num determinado período de tempo.

## **8 Implementação**

### **8.1 Arquitetura do Sistema**

Para o bom funcionamento do sistema, foi necessário criar uma arquitetura que permitisse a comunicação entre o Frontend e o Backend, e entre o Backend e a Base de Dados. A arquitetura do sistema foi feita com recurso ao padrão MVC, que permite separar o Frontend do Backend, e o Backend da Base de Dados. A aplicação comunica com a API e a Base de Dados e funciona como uma ponte entre os dois, permitindo ao utilizador interagir com o sistema. A arquitetura do sistema é a seguinte:

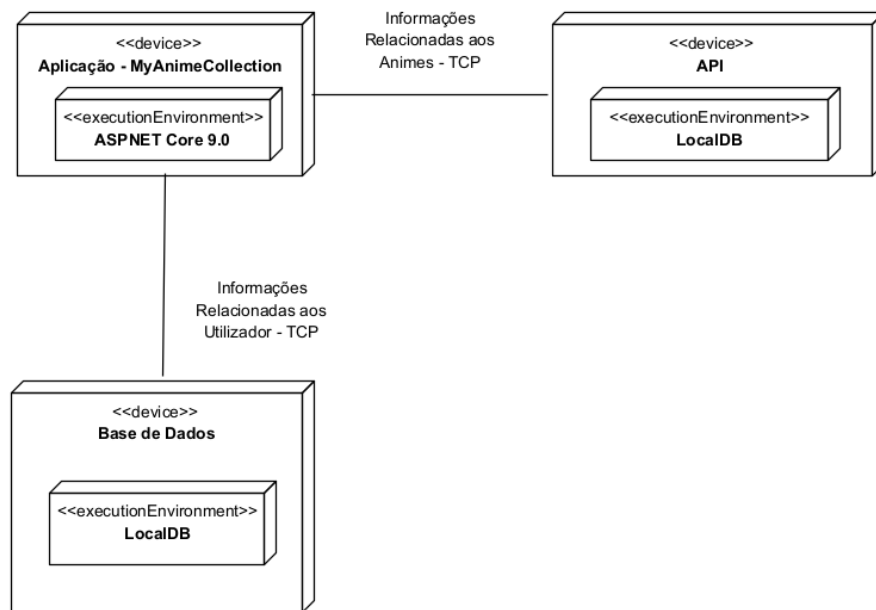


Figura 9: Arquitetura do Sistema

A aplicação representa o Frontend/Computador do utilizador do sistema, que se baseia no ASPNET Core 9.0, que é a aplicação que o utilizador interage. A aplicação comunica com a API, que possui a sua base de dados em LocalD e a Base de Dados do Sistema, possuiu outra base de dados em LocalDB.

## 8.2 Tecnologias Usadas

As tecnologias utilizadas para a implementação do sistema foram:

- ASP.NET Core 9.0 (Microsoft, 2025a)
- Entity Framework Core (Microsoft, 2025b)
- SQL Server (Microsoft, 2025d)
- Razor, HTML, CSS, HTML Helpers (Microsoft, 2025c)
- Swagger (Swagger, 2025)
- Bootstrap (Bootstrap, 2025)

Este sistema inicialmente foi desenvolvido em React, no âmbito da Unidade Curricular de Tecnologias Web e Desenvolvimento de Aplicações Móveis, porém, para implementar a API e a base de dados, o grupo decidiu mudar para ASP.NET Core 9.0, devido à facilidade de implementação da API e da base de dados, que foi feita em SQL Server, porém com uma metodologia de Code First. O Frontend foi desenvolvido em Razor, utilizando HTML, CSS, Bootstrap e HTML Helpers. O Backend foi desenvolvido em C#.

## 8.3 Desenvolvimento da API

A API foi desenvolvida em ASP.NET Core 9.0, utilizando Entity Framework Core para a comunicação com a base de dados em LocalDB. A API foi desenvolvida com o objetivo de fornecer os dados dos animes para o Frontend. A API foi alimentada com a API da Jinkan, que fornece dados de animes, como sinopse, número de episódios, número de temporadas, entre outros. Foi decidido limitar os dados fornecidos pela API da Jinkan, para que a API do MyAnimeCollection fosse mais fácil de implementar e mais leve. A API foi desenvolvida com os seguintes endpoints:

- GET /api/animés - Retorna todos os animés
- GET /api/animés/id - Retorna um anime específico
- POST /api/animés - Adiciona um anime
- PUT /api/animés/id - Atualiza um anime
- DELETE /api/animés/id - Apaga um anime

## 8.4 Especificação da Interface

Para facilitar a compreensão do utilizador, a interface foi desenvolvida com o apoio do Swagger, que permite visualizar os endpoints da API e testá-los. A interface foi desenvolvida com o objetivo de ser simples e intuitiva, para que o utilizador possa facilmente perceber como utilizar a API. Ao utilizar o Swagger, o programador está a fazer alterações diretas na base de dados, da API. Como a base de dados é em LocalDB, as alterações são feitas diretamente na base de dados local, e não na base de dados da API, que pode um dia ser implementada num servidor.

## 8.5 Decisões de Implementação

Para criar a base de dados utilizada pelo sistema, foi utilizada a metodologia Code First, que permite criar a base de dados a partir do código. A base de dados foi criada depois com recurso ao Entity Framework Core, mais especificamente a parte das Migrations.

As rotas do sistema foram criadas de maneira a ser intuitivas e fáceis de perceber, para que o programador que venha a trabalhar no sistema no futuro possa facilmente perceber como o sistema funciona. Por exemplo, a rota */user/id* permite ao utilizador ver o perfil de um utilizador, enquanto a rota */user/id/id\_lista* permite ao utilizador ver a lista de um utilizador.

Os controladores foram criados com base nos casos de uso, para que o programador possa facilmente perceber como o sistema funciona. Por exemplo, o

controlador *UserController* possui todos os métodos relacionados com o utilizador, como *Register*, *Login*, *Logout*, *Profile*, entre outros.

Na questão de segurança, foi implementado um sistema de autenticação, que permite ao utilizador registar-se e iniciar. O sistema de autenticação foi feito com recurso ao Identity, que é uma biblioteca do ASP.NET Core que permite criar um sistema de autenticação de forma fácil e rápida.

## 8.6 Principais Casos Relevantes de Programação

Explicação dos principais e mais complexos casos de codificação, exemplificando com pequenos trechos de código que permitam a um outro programador no futuro perceber esses casos.

# 9 Desenvolvimento da App Frontend/MVC

Descrever as principais decisões e casos de implementação da App frontend/MVC.

## 9.1 Decisões de Implementação

Descrever as principais decisões globais usadas na codificação da App, justificando a escolha, para os casos seguintes: escolha das rotas, definições dos controladores, definições dos modelos e segurança do sistema. Definições dos layouts das vistas (componentes das vistas).

## 9.2 Principais Casos Relevantes de Programação

Explicação dos principais casos de codificação para os casos de uso em causa, exemplificando com pequenos trechos de código e imagens das interfaces geradas pelas vistas.

## 10 Conclusão e Trabalho Futuro

Nesta etapa do projeto, obtivemos resultados que consideramos bastante satisfatórios e acreditamos que este projeto apresenta um grande potencial para o futuro. A utilização do ASP.NET, em vez do Laravel como inicialmente planejado, proporcionou-nos maior facilidade na implementação da API, uma vez que o ASP.NET nos permitiu criar uma API e um projeto MVC num único projeto, simplificando assim a comunicação entre a API e o Frontend. Estamos confiantes de que a aplicação tem um maior nível de fidelidade em relação à versão feita no semestre anterior na Unidade Curricular de Tecnologias Web e Desenvolvimento de Aplicações Móveis, onde encontramos dificuldades significativas devido às limitações da RestDB, que possuía um limite que prejudicava imenso devido à nossa extrema necessidade para garantir ao utilizador algumas funcionalidades que tornavam a páginas mais intuitiva, como por exemplo, mostrar se o anime já estava adicionado a alguma lista.

## Bibliografia

Bootstrap. (2025). *Bootstrap*. Obtido janeiro 17, 2025, de <https://getbootstrap.com/>

Bruno, L. (2025). *Página da Unidade Curricular de Desenvolvimento de Aplicações Web*. Obtido janeiro 17, 2025, de <https://cms.ipbeja.pt/course/view.php?id=1451>

Microsoft. (2025a). *ASP.NET*. Obtido janeiro 17, 2025, de <https://learn.microsoft.com/en-us/aspnet/core/?view=aspnetcore-9.0>

Microsoft. (2025b). *Entity Framework Core*. Obtido janeiro 17, 2025, de <https://learn.microsoft.com/en-us/ef/>

Microsoft. (2025c). *Razor*. Obtido janeiro 17, 2025, de <https://learn.microsoft.com/en-us/aspnet/core/mvc/views/razor?view=aspnetcore-9.0>

Microsoft. (2025d). *SQL Server*. Obtido janeiro 17, 2025, de <https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/sql-server-express-localdb?view=sql-server-ver16>

Swagger. (2025). *Swagger*. Obtido janeiro 17, 2025, de <https://swagger.io/docs/>