*Trabalho de Programação Avançada*

*Programa de Gestão de*

*Coleções*

*Resumo*

Foi proposto, como projeto final da disciplina de Programação Avançada , lecionada pelo Professor Fernando Barros , o seguinte desafio:

Desenvolver um sistema de gestão de coleções implementado em linguagem Python, utilizando os princípios da Programação Orientada a Objetos.

O programa foi projetado para atender às necessidades administrativas e de gestão de um grupo de coleções , oferecendo funcionalidades abrangentes para o gerenciamento de varias coleções e de todos os campos inerentes a cada uma delas.

Tem funcionalidades de o utilizador dentro de cada coleção , poder efetuar diversas funcionalidades de criar , apagar , modificar , e exportar e importar

Este projeto será realizado pelos alunos William Breunig, Jorge Medina, Filipa Coutinho e Bernardo Anjos

Índice

[Introdução 4](#_Toc199284232)

[Descrição do Desafio 5](#_Toc199284233)

[1 – Ambiente Gráfico 6](#_Toc199284234)

[1.1 – Desenvolvimento 6](#_Toc199284235)

[1.2 – Testes 6](#_Toc199284236)

[1.3 –Documentação 6](#_Toc199284237)

[2 - Implementar a base de dados e funcionalidade de categorização 7](#_Toc199284238)

[2.1 – Base de Dados 7](#_Toc199284239)

[2.2 – Funcionalidades de Organização 7](#_Toc199284240)

[2.3 – Testes 7](#_Toc199284241)

[2.4 – Documentação 7](#_Toc199284242)

[3 - Adicionar sistema de pesquisa avançada na coleção 8](#_Toc199284243)

[3.1 Pesquisa avançada 8](#_Toc199284244)

[3.2 – Testes 9](#_Toc199284245)

[3.3 –Documentação 11](#_Toc199284246)

[4 - Exportação e importação de coleções em ficheiros 12](#_Toc199284247)

[4.1 – Exportação e Importação 12](#_Toc199284248)

[4.2 – Testes 12](#_Toc199284249)

[4.3 – Documentação 12](#_Toc199284250)

[Conclusão 13](#_Toc199284251)

[Anexos 14](#_Toc199284252)

# Introdução

Num mundo cada vez mais digital e organizado, existe a necessidade de **gerir coleções de forma eficiente** , seja para colecionadores particulares, museus, bibliotecas ou empresas. Um **Programa de Gestão de Coleções** surge como uma ferramenta indispensável para pesquisar, atualizar , inserir e partilhar itens de maneira estruturada, garantindo não só a preservação da coleção em si , mas também a sua acessibilidade.

Este tipo de software permite:

* **Organização centralizada** de itens (livros, obras de arte, vinis, jogos, etc.).
* **Busca rápida** através de filtros e categorias personalizáveis.
* **Registo detalhado** de informações como data de aquisição, observações e data de aquisição.
* **Partilha segura** com outros utilizadores.

Com uma interface intuitiva, um **Programa de Gestão de Coleções** não só simplifica o dia a dia de colecionadores , como também transforma uma simples acumulação de objetos num coleção devidamente documentado e valorizado.

Seja para fins pessoais, académicos ou profissionais, esta ferramenta abre portas a uma nova forma de interagir com coleções, garantindo que cada item seja facilmente localizável, bem preservado e devidamente apreciado.

# Descrição do Desafio

O principal desafio do projeto consiste em desenvolver uma aplicação robusta e interface intuitiva e que pudesse:

* Gerir varias coleções .
* Fácil de utilizar , com uma interface intuitiva.
* Existência de varias opções de gestão , como sendo inserção , e eliminar elementos e atualização das coleções
* Pudesse ser importada e exportada , de uma forma fácil e rápida e segura

Este processo foi desenvolvido em quatro etapas , até ao resultado , final .

Essas etapas foram as seguintes :

1. Criação do ambiente gráfico
2. Criação de opções de gestão das coleções ( Apagar , alterar , modificar e atualizar )
3. Criação de opções de pesquisa avançada
4. Exportação e importação

# 1 – Ambiente Gráfico

## 1.1 – Desenvolvimento

Para a criação de o nosso ambiente gráfico , optamos pela a utilização de diversas bibliotecas , que nos permitem tornar o nosso projeto de fácil entendimento e de fácil navegação.

Essas

## 1.2 – Testes

Com o objetivo de tornar o programa mais intuitivo e fácil de utilizar, foi dada especial atenção ao design da interface gráfica, tornando-a apelativa e funcional para o utilizador. Foram criadas janelas que reúnem toda a informação necessária para que a gestão e manutenção das coleções possam ser realizadas de forma simples e eficiente. Além disso, foram adicionados botões que facilitam o acesso às funcionalidades implementadas, tornando a sua utilização mais rápida e direta.

## 1.3 –Documentação

A criação da interface gráfica apresentou algumas dificuldades de implementação, nomeadamente no posicionamento de certas janelas, especialmente quando uma nova era aberta e a anterior era encerrada. Foi necessário realizar diversos ajustes até que as janelas fossem corretamente posicionadas no ecrã.  
Outra dificuldade encontrada foi na criação e disposição dos botões, de forma a garantir que se encaixassem adequadamente no espaço disponível em cada janela, mantendo a estética e a funcionalidade da interface.

# 2 - Implementar a base de dados e funcionalidade de categorização

## 2.1 – Base de Dados

Para podermos gerir o nosso projeto , e no que se refere a gestão das nossas colecções, optamos de criar uma base de dados , ao qual optamos por utilizar o PhpMYAdm, para criar uma base de dados, como os referidos campos de cada coleção .Desta forma , conseguimos guardar toda a informação que e inserida , e aletrada pelo o utilizador , evitando desta forma a perda de dados

## 2.2 – Funcionalidades de Organização

## 

Para efetuarmos uma organização mais dinamica , foi criado diversas funcionalidades que colocamos a disponibilidade do utilizador , como sendo a inserção de novos elementos em cada uma das colecções , alterar campos de elementos já inseridos , e apagar , quando algum desses elementos já não pertencer a colecção

## 2.3 – Testes

Despois de implementar a funcionalidade de organização conseguimos com que o utilizador possa inserir novos elementos nas coleções e também para poder alterar esses elementos da coleção quando o utilizador quiser e também poder apagar esses elementos das coleções sempre que quiser

## 2.4 – Documentação

Apos implementar esta funcionalidade surgiu alguns problemas relacionados a base de dados como a ligação a base de dados que não estava a funcionar porque tinha um erro na função obter\_ ligação mas foi resolvido tirando o comentário onde esta programado a porta outro problema foi a parte da data que estava a dar erro no formato dela foi resolvido acrescentando uma verificação que verifica se o formato da data e correto ou não

# 3 - Adicionar sistema de pesquisa avançada na coleção

## 3.1 Pesquisa avançada

Implementação de pesquisa avançada . Esta funcionalidade permitirá ao utilizador pesquisar dados específicos, sendo que optámos por possibilitar a pesquisa através de dois campos distintos, deixando ao utilizador a liberdade de escolher o critério que melhor se adequa às suas necessidades.

Este código em Python define uma função chamada pesquisar que serve para ir buscar informações a uma tabela específica numa base de dados.

A função permite que se procurem dados de duas maneiras opcionais:

* **Por nome:** Se fornecer um filtro\_nome, a função vai procurar registos na tabela onde a coluna "nome" contenha o texto que indicou (mesmo que seja só uma parte do nome).
* **Por tipo/tema:** Se fornecer um filtro\_tipo, a função vai procurar registos onde a coluna "tipo" (se a tabela for "jogos") ou a coluna "tema" (para outras tabelas) contenha o texto que indicou.

A função começa por ligar-se à base de dados e preparar um comando SQL para selecionar tudo da tabela. Depois, verifica se foram dados filtros. Se sim, adiciona condições à procura (cláusula WHERE) para filtrar os resultados pelo nome e/ou pelo tipo/tema, usando o operador LIKE para encontrar correspondências parciais.

Finalmente, executa o comando na base de dados, recolhe os resultados encontrados e fecha a ligação, devolvendo uma lista com esses resultados. Se não encontrar nada que corresponda aos filtros (ou se não houver filtros), devolve uma lista vazia. Em resumo, é uma ferramenta flexível para pesquisar dados numa base de dados com a possibilidade de usar filtros por nome e por tipo/tema.

## 3.2 – Testes

**Relatório de Testes – Função “pesquisar”**

**1. Informações Gerais**

* **Contexto:** Este relatório detalha os testes realizados na função pesquisar, responsável por efetuar procuras parametrizadas numa base de dados, utilizando filtros opcionais por nome e tipo/tema.

**2. Objetivo do Teste**

O principal objetivo deste teste é verificar a correta funcionalidade da função pesquisar em diversos cenários de utilização, garantindo que:

* A função retorna os dados esperados quando nenhum filtro é aplicado.
* A função filtra corretamente os dados com base no filtro\_nome.
* A função filtra corretamente os dados com base no filtro\_tipo (ou filtro\_tema para a tabela "jogos").
* A função combina corretamente ambos os filtros (filtro\_nome e filtro\_tipo/filtro\_tema).
* A função lida adequadamente com casos onde não são encontrados resultados.
* A função aplica o campo correto (tipo ou tema) dependendo da tabela.

**3. Ambiente de Teste**

* **Sistema Operativo:** Windows 10
* **Linguagem de Programação:** Python 3.13
* **Base de Dados:** MySQL

**4. Cenários de Teste e Resultados**

Para os testes, assumimos uma base de dados com as seguintes tabelas e dados

**Tabela: livros – dados -país /título / autor / tema / ano**

**Tabela: jogos – dados – nome / título / tema / ano / observações**

**Cenário de Teste 1: Procura sem filtros**

* **Descrição:** Chamar a função pesquisar sem especificar filtro\_nome ou filtro\_tipo.
* **Input:** pesquisar("produtos")
* **Resultado Esperado:** Todos os registos da tabela jogos/livros
* **Resultado Obtido:** Todos os registos da tabela jogos/livros
* **Estado:** **PASS**

**Cenário de Teste 2: Procura com filtro por nome**

* **Descrição:** Chamar a função pesquisar com filtro\_nome especificado.
* **Input:** pesquisar("jogos", filtro\_nome="Need for Speed")
* **Resultado Esperado:** Registos da tabela jogos onde o nome contém " Need for Speed ".
* **Resultado Obtido:** Registos que contêm nome “Need for Speed” aparecem na lista de resultados de pesquisa
* **Estado:** **PASS**

**Cenário de Teste 3: Procura com filtro por tema (tabela jogos)**

* **Descrição:** Chamar a função pesquisae com filtro\_tipo para a tabela "jogos" (onde o campo é "tema").
* **Input:** pesquisar("jogos", filtro\_tipo="Aventura")
* **Resultado Esperado:** Registos da tabela jogos onde o tema contém "Aventura".
* **Resultado Obtido:** Nos resultados da pesquisa aparecem todos os jogos com o tema “Aventura”
* **Estado:** **PASS**

**Cenário de Teste 4: Procura com ambos os filtros (nome e tipo/tema)**

* **Descrição:** Chamar a função pesquisa com filtro\_nome e filtro\_tipo (ou filtro\_tema).
* **Input:** pesquisar("jogos", filtro\_nome="Aventura", filtro\_tipo="Aventura")
* **Resultado Esperado:** Registos da tabela jogos onde o nome contém "Aventura" E o tema contém "Aventura".
* **Resultado Obtido:** Todos os jogos que contém o nome “Aventura” e o tema é “Aventura” aparecem nos resultados
* **Estado:** **PASS**

**Cenário de Teste 6: Procura com filtros que não encontram resultados**

* **Descrição:** Chamar a função pesquisar com filtros que não correspondem a nenhum registo.
* **Input:** pesquisar("jogos", filtro\_nome="frog factions", filtro\_tipo="matematica")
* **Resultado Esperado:** Uma lista vazia.
* **Resultado Obtido:** Uma lista vazia
* **Estado:** **PASS**

**5. Conclusões**

A função pesquisar demonstrou um comportamento correto em todos cenários de procura com e sem filtros, tanto para o campo nome como para o campo tipo/tema. A lógica de construção da query SQL e a passagem de parâmetros estão a funcionar como esperado.

## 3.3 –Documentação

**Função:** O código “pesquisar” permite ao utilizador procurar jogos ou livros numa colecção numa base de dados.

**Como utilizar:** O utilizador usa os campos de pesquisa integrados na interface gráfica, sendo eles “nome” e “tipo”, introduzindo texto para ser procurado na base de dados fornecida, recebendo resultados que correspondam ao que tenham procurado.

Recursos utilizados - <https://dev.mysql.com/doc/>

- <https://www.red-gate.com/simple-talk/databases/mysql/retrieving-mysql-data-python/>

<https://dev.mysql.com/doc/connector-python/en/connector-python-example-cursor-select.html>

# 4 - Exportação e importação de coleções em ficheiros

## 4.1 – Exportação e Importação

## 

Para que o nosso projeto seja de fácil transporte e acesso noutros dispositivos, foi implementada uma funcionalidade de importação e exportação através de ficheiros JSON. Desta forma, qualquer alteração efetuada na base de dados pode ser facilmente importada para o projeto, assim como é possível exportar todos os dados existentes para um ficheiro JSON, garantindo portabilidade e segurança da informação.

## 4.2 – Testes

Após a implementação das funcionalidades de importação e exportação em JSON, estas encontram-se plenamente operacionais. Com isso, é gerado um ficheiro que o utilizador pode guardar no local que desejar, permitindo não só o transporte fácil dos dados, mas também a criação de cópias de segurança (backups) das informações inseridas.

Desta forma podemos ter

## 4.3 – Documentação

A implementação desta funcionalidade apresentou alguns desafios e problemas até que fosse possível colocá-la em pleno funcionamento. O erro mais persistente ocorreu durante a exportação dos dados para ficheiros JSON: os campos definidos como datas não eram corretamente interpretados, resultando em erros. Para resolver este problema, criei uma função chamada converter, que utiliza o método isoformat() para transformar os objetos de data em strings compatíveis com o formato JSON.

# Conclusão

Este projeto proporcionou uma experiência completa na aplicação das competências adquiridas na disciplina de Programação Avançada, desde o planeamento até à execução.

A criação de interfaces gráficas garantiu que o sistema fosse funcional e intuitivo para os utilizadores. Durante o desenvolvimento, surgiram algumas dificuldades, nomeadamente a ligação à base de dados que alimenta os campos das nossas coleções e a implementação de funcionalidades específicas para cumprir os requisitos solicitados — especialmente no que diz respeito à importação e exportação de dados em formato JSON. No entanto, com persistência e esforço, esses obstáculos foram superados.

Outro desafio enfrentado foi a implementação da pesquisa avançada, mais concretamente a criação de campos de pesquisa independentes para cada coleção.

No geral, o projeto atingiu os objetivos propostos e contribuiu significativamente para o desenvolvimento das competências dos alunos na criação e execução de código em Python.