Relatório do Trabalho Prático de Programação Web

Professores:
Prof. Coordenador Doutor Jorge Augusto Castro Neves Barbosa
Prof. Adjunto Doutor Tiago André Ferreira de Almeida das Neves Figueira

ANO LETIVO 2025/2026

Da autoria de:
Fábio Oliveira 2022145902
Sebastian Vigas 2023155905

Instituto Superior de Engenharia
Politécnico de Coimbra

Índice

1. Introdução

Este relatório documenta o desenvolvimento da plataforma MYCOLL, um sistema integrado para a listagem e comercialização de peças colecionáveis (moedas, selos, camisolas desportivas, etc.). O projeto foi desenvolvido no âmbito da unidade curricular de Programação Web e baseia-se numa arquitetura moderna composta por três pilares: uma API RESTful, uma aplicação de Gestão de Loja (Backend) e uma aplicação Frontend Multiplataforma (Web via Blazor e Mobile via .NET MAUI).

2. Especificações Funcionais

O sistema MYCOLL foi concebido para centralizar todas as etapas do ciclo de vida de um produto colecionável, desde a submissão pelo fornecedor até ao envio do item ao cliente. Para tal, as funcionalidades foram divididas em dois ambientes distintos e complementares:

2.1. Aplicação Frontend (Público Geral: Clientes e Fornecedores)

Esta aplicação, disponível em ambiente Web e Mobile (.NET MAUI), serve como a "montra" e ferramenta de trabalho dos utilizadores externos.

- **Utilizadores Anónimos:**
  - Navegação na página inicial com todos os produtos ativos.
  - Visualização de um produto em destaque escolhido aleatoriamente pelo sistema.
  - Visualizar detalhes de um produto ao clicar nele.
  - Registo de nova conta (Cliente ou Fornecedor).
- **Utilizadores Clientes:**
  - Autenticação segura via Token JWT.
  - Gestão de Carrinho de Compras: Adição de produtos, ajuste de quantidades e remoção de itens.
  - Finalização de Encomendas: Processo de *checkout* com simulação de pagamento/encomenda.
- **Utilizadores Fornecedores:**
  - Gestão de Inventário: Criação de novos produtos para venda (submetidos para aprovação), edição e remoção dos seus próprios produtos.

2.2. Aplicação de Gestão de Loja (Administração e Backend)

Esta aplicação é de uso exclusivo da equipa interna e interage diretamente com a base de dados para garantir o controlo operacional da plataforma.

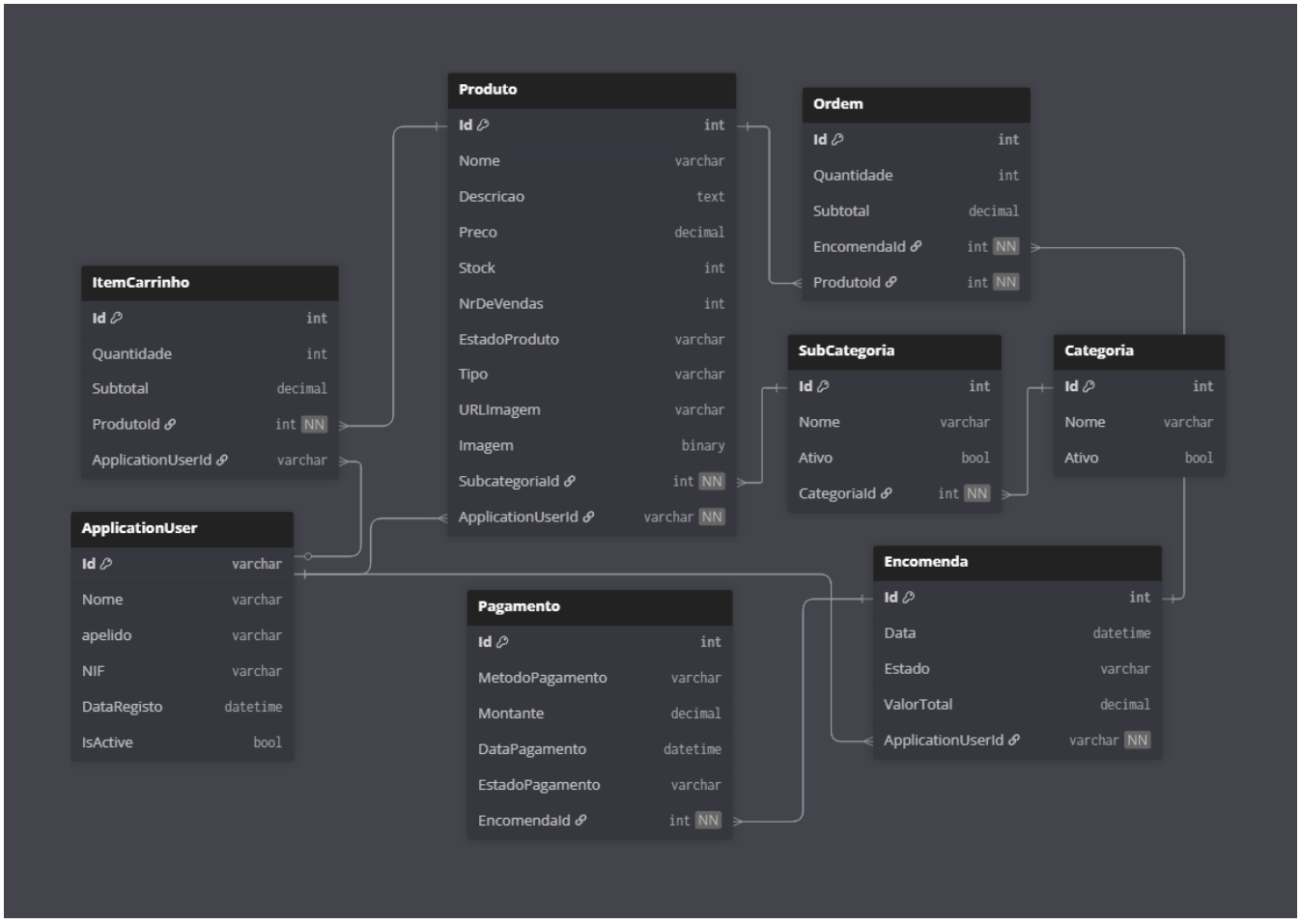
- **Gestão de Utilizadores e Roles:**
  - Administração de contas: Ativação ou desativação de utilizadores registados (Aprovação de contas).
  - Controlo de Acessos: Criação de novos perfis de acesso e atribuição de *Roles* (Admin, Funcionário, Fornecedor, Cliente).

- **Gestão de Catálogo:**
  - CRUD completo (Criar, Listar, Editar e Eliminar) de Produtos, Categorias e Subcategorias.
- **Gestão de Produtos:**
  - Controlo total sobre a lista global de produtos.
  - **Aprovação de Produtos:** Validação técnica e comercial de itens submetidos por fornecedores, permitindo a sua passagem para o estado "Ativo" para que fiquem visíveis na loja e para que sejam taxados.
- **Gestão de Encomendas e Vendas:**
  - Monitorização de todas as encomendas realizadas.
  - Fluxo Logístico: Alteração de estados de encomendas (Expedir ou Cancelar).
  - Atualização Automática de Stocks: Abate automático de quantidades no inventário aquando da expedição.
- **Configuração de Taxas e Regras de Negócio:**
  - Definição da taxa de lucro global (%) que o sistema deve aplicar automaticamente aos preços base definidos pelos fornecedores no momento da ativação do produto.

3. Modelo Entidade-Relacionamento (MER)

A base de dados, implementada em SQL Server LocalDB, assenta nas seguintes entidades principais:

- **Utilizadores (ApplicationUser): Herda de IdentityUser, incluindo Nome, NIF e estado de atividade.**
- **Produtos: Contém preço base, stock, descrição e imagem (armazenada em formato binário byte[]).**
- **Categorias e Subcategorias: Estrutura hierárquica para organização do catálogo.**
- **Encomendas e Ordens: Registo das transações e dos itens específicos de cada venda.**
- **Pagamentos: Registo da simulação de transações financeiras.**
- **Ordem: Regista o produto que foi encomendado.**
- **ItemCarrinho: São os items que estão no carrinho do Utilizador**



4. Perfis de Utilizadores

- **Anónimo**: Visualiza o catálogo, pode clicar em produtos e vê o produto em destaque.
- **Cliente**: Efetua compras, gere o seu carrinho e faz encomendas.
- **Fornecedor**: Gere os seus próprios produtos, mas estes nascem como "Pendentes".
- **Administrador/Funcionário**: Tem controlo total sobre o inventário, aprova utilizadores e produtos, e altera os estados das encomendas (Expedida/Cancelada).

5. Implementação Técnica

5.1. Estrutura de Dados

Utilizou-se o Entity Framework Core 8 com a abordagem *Code-First*. As migrações garantiram a consistência entre as classes C# e as tabelas SQL. O acesso aos dados no Backend é feito diretamente via DbContext para maior eficiência administrativa.

5.2. Comunicação e Segurança (API & JWT)

A comunicação entre o Frontend e a Base de Dados é mediada por uma API ASP.NET Core. A segurança é garantida por tokens JWT (JSON Web Tokens). Implementámos um sistema onde os Roles (Papéis) do utilizador são injetados nos *Claims* do token, permitindo que a API autorize ou bloqueie ações com base no perfil (ex: [Authorize(Roles = "Fornecedor")]).

5.3. Partilha de Código (RCL & Blazor Hybrid)

Seguindo as diretrizes das Fichas 3 e 8, criámos uma Razor Class Library (RCL). Isto permitiu que 100% da interface e lógica do utilizador (Pages, Services, DTOs) fossem partilhadas entre o projeto Blazor Web e o .NET MAUI. Assim, a aplicação mobile e o site usam exatamente o mesmo código.

6. Regras de Negócio

As seguintes regras foram codificadas para refletir o cenário real proposto:

1. **Aprovação de Contas**: Novos utilizadores registados via API ficam com *IsActive = false* e não podem fazer login até serem aprovados pelo Admin.
2. **Filtro de Visibilidade de Montra:** Um produto só aparece para os clientes quando este estiver Ativo por um Admin.
3. **Margem de Lucro (Taxa):** Implementámos um *ConfigService* onde o Admin define uma taxa global. Quando um produto é aprovado (passa de Pendente a Ativo), o sistema aplica automaticamente essa percentagem sobre o preço base do fornecedor.
4. **Gestão de Stock:** Ao marcar uma encomenda como "Expedida" no Backend, o sistema abate automaticamente as quantidades ao stock disponível dos produtos.
5. **Regra de Propriedade Privada:** Um Fornecedor está impedido de visualizar ou editar produtos que não lhe pertençam.

7. Funcionalidades Implementadas

A plataforma MyCOLL foi desenvolvida focando na robustez técnica e na experiência do utilizador. Abaixo detalham-se as principais implementações:

Pesquisa Inteligente em Cascata:
Para facilitar a navegação num catálogo potencialmente vasto, implementámos um sistema de filtragem onde o utilizador pode optar filtrar os produtos por uma categoria geral (ex: "Moedas") e uma subcategoria específica (ex: "Jogos Olímpicos"), permitindo uma navegação intuitiva e rápida.

Armazenamento Binário de Imagens:
As imagens são convertidas em byte[] e armazenadas diretamente na base de dados, garantindo a portabilidade dos dados.

Algoritmo de Destaque Aleatório:
Para cumprir os requisitos do frontend, implementámos um componente que, a cada carregamento da *Home Page*, seleciona um produto aleatório do repositório (apenas entre os que estão no estado "Ativo").

Fluxo de Encomenda:
O processo de compra foi desenhado da seguinte forma:

- Carrinho: O utilizador pode gerir itens e quantidades como bem entender.
- Transição de Estados: Ao finalizar a compra, o sistema executa três ações automáticas:
    1. Geração da Encomenda e cálculo do total.
    2. Criação de um registo que espera confirmação de um admin/funcionário.
    3. Limpeza imediata do carrinho de compras.

8. Conclusão

O desenvolvimento da plataforma MYCOLL permitiu a consolidação de conhecimentos críticos sobre o ciclo de vida de aplicações multiplataforma modernas. A integração bem-sucedida de uma API RESTful segura com um frontend partilhado (RCL) entre as vertentes Web e MAUI demonstrou de forma mais prática, fácil e versátil, demonstrando o potencial do ecossistema .NET 8.

Este projeto constituiu um desafio técnico recompensador, exigindo soluções criativas para problemas de persistência, segurança e gestão de dados binários. O resultado final é uma solução alinhada com os requisitos e boas práticas da engenharia de software modernas, cumprindo a maioria dos objetivos propostos.

9. Anexo (Dados de Exemplo)

Role	Email	Password
Admin	**admin@localhost.com**	**Is3C..00**
Fornecedor	**pweb2@localhost.com**	**Password.1**
Cliente	**pweb1@localhost.com**	**Password.1**
Cliente	**pweb3@localhost.com**	**Password.1**
Cliente	**pweb4@localhost.com**	**Password.1**