

Licenciaturas em Engenharia Informática  
Programação Web  
Ano Lectivo de 2024/2025 – 1º Semestre  
**TRABALHO PRÁTICO A REALIZAR EM GRUPO EXTRA-AULAS**  
**TEMA B**

## 1. Introdução

Como trabalho prático a desenvolver fora das aulas e em grupo de dois alunos, crie uma aplicação designada por *QueVistoHoje* e que será uma aplicação *frontend* de uso “público” que poderá ser executada através de uma interface *Web*, de um device *Android*, *iOS*, *macOS* ou *Tizen*, e que é uma plataforma de comercialização e entrega de roupa produzida por um conjunto de outras empresas com as quais esta plataforma tem acordos de revenda e entrega. Para além da roupa, propriamente dita também é possível a compra e entrega de complementos tais como carteiras, cintos, perfumes e outros complementos. O código a desenvolver para esta aplicação “*frontend*” é um código único e desenvolvido utilizando um só projecto *Visual Studio* para qualquer uma das interfaces, aplicações, a desenvolver e atrás referidas.

Esta aplicação *frontend* tem de ser complementada por uma outra aplicação de *Gestão de Loja*, de uso exclusivo por administradores e gestores da empresa proprietária da aplicação e que comercializa as “roupas”, e que servirá para que esta empresa faça a gestão dos produtos, categorias dos produtos, modos de disponibilização dos mesmos bem, como os utilizadores, Clientes, Gestores e Administradores das aplicações e permitir ainda a gestão das compras de produtos feitas na aplicação *frontend* pelos clientes.

Tem de ser ainda criada uma outra aplicação API RestFull, que permitirá o acesso, controlado, aos dados de uma base de dados por parte da aplicação *frontend* “pública”. A base de dados desta *API*, será a mesma com a qual a aplicação de gestão de loja interage, isto é, os produtos, categorias e etc são os mesmos e estão por isso numa só Base de Dados. A aplicação “*frontend*” irá solicitar dados a esta *API* que os fornecerá. A aplicação de gestão da loja, interagirá directamente com esta Base de Dados e não através desta *API* ou de outra.

A aplicação “*frontend*“ a desenvolver terá um único código e esse código de programação da mesma irá ser utilizado quer para gerar a “página” *Web*, quer para gerar as aplicações em código nativo dos devices, *Windows Machine*, *Android*, *iOS*, *macOS*, *Tizen*.

Qualquer uma das três aplicações referidas, utilizarão a autenticação e a autorização de utilizadores, em particular e obrigatoriamente usando o *.Net 8 Identity Framework* no seu todo. No caso da aplicação *frontend*, esta utilização será feita com recurso à *API* desenvolvida e no caso da aplicação de gestão de loja será feita directamente pela aplicação e no seu todo, isto é, não utilizando somente as funções deste *framework Identity*. A mesma base de dados da aplicação já referida, também será usada para esta parte de autenticação e autorização. Os utilizadores da aplicação “*frontend*” pública só precisarão de se autenticar aquando da finalização da compra.

As soluções aplicacionais a construir devem representar uma situação real, ainda que dentro do contexto em que são desenvolvidas. As funcionalidades apresentadas neste enunciado são consideradas uma amostra primária de um conjunto de funcionalidades, no âmbito do problema,

podendo ser adicionadas outras funcionalidades que sejam consideradas necessárias e úteis à definição da solução.

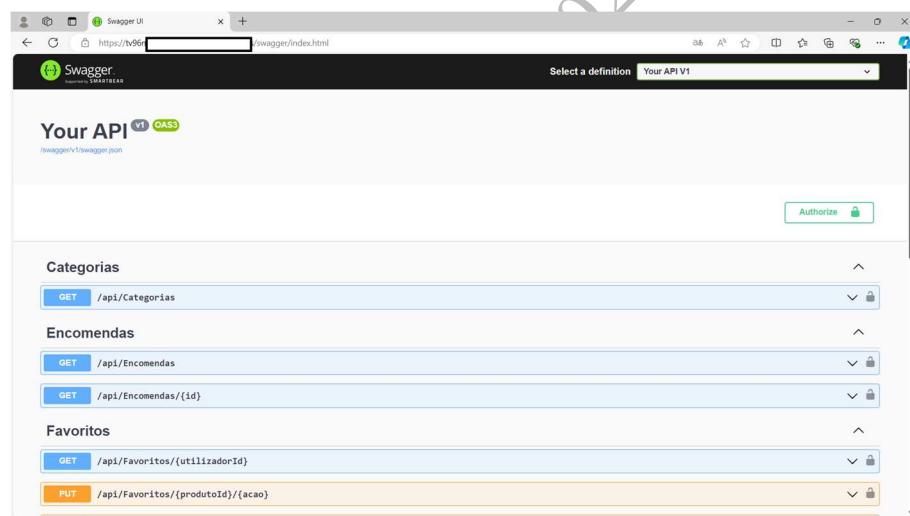
A solução final a apresentar no conjunto das três aplicações a desenvolver, deverão perspetivar uma aplicação de utilização real e prática e constituir no contexto da Engenharia o que neste contexto se entende como “*prova de conceito*”.

## 2. Ambiente de Desenvolvimento

É obrigatório implementar as aplicações com base no *framework .NET Core 8* utilizando *Blazor Web*, *Blazor Hybrid* e *RCL*, a linguagem de programação *C# 12*, *LINQ*, *.Net Core 8 Entity Framework* e *.NET Core 8 Identity Framework*, *SQL Server Local DB* e utilizando o *IDE Visual Studio Community 2022*.

Em cada uma das três partes, aplicações, a desenvolver utilizar-se-ão o ou os *templates .NET 8* adequados para essa parte utilizando como exemplo o trabalho desenvolvido durante as aulas laboratoriais.

Na aplicação *API Restfull*, é obrigatório utilizar o *Swagger* na API para efeito de testes à mesma. Na figura seguinte, mostra-se um exemplo de um ecran do *Swagger* da *API RestFull* a desenvolver.



Para a comunicação, na fase de desenvolvimento, entre a *API* e a aplicação *frontend*, tem de ser utilizado um *Dev Tunnel*.

## 3. Estruturas de Dados e Base de Dados das Aplicações

- A estrutura de dados utilizada tem de ter por base um Modelo-Entidade-Relacionamento, *MER*, também de entrega obrigatória, o qual deve reflectir todas as necessidades em termos de Entidades e respectivas Relações nas aplicações, concretizado nas aplicações através de classes de domínio codificadas em *C#* e que corresponderão a tabelas da base de dados criadas com recurso a instruções *migration*.

#### 4. Regras de Negócio consideradas no desenvolvimento das aplicações

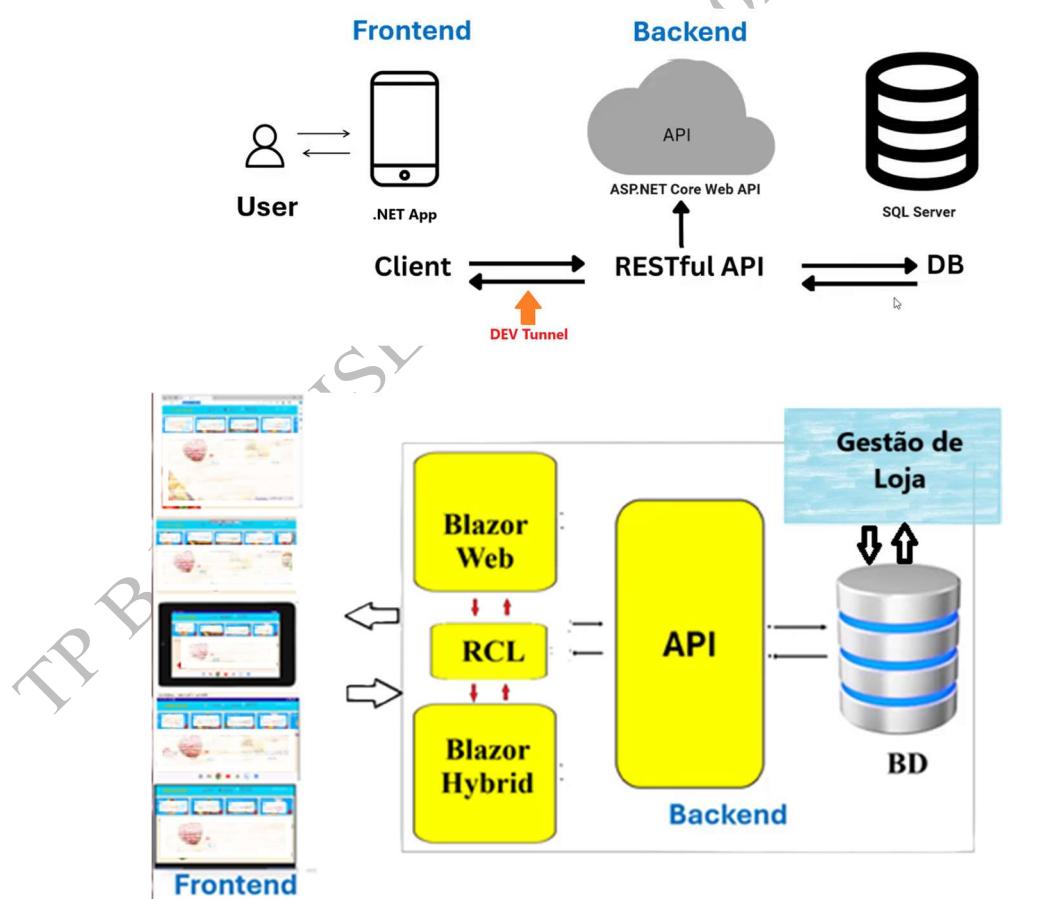
É obrigatória a especificação de um conjunto de especificações funcionais que traduzam as especificações definidas pela empresa que irá utilizar a aplicação e que constituirão as designadas “*Regras de Negócio*” a serem consideradas no desenvolvimento e em particular no funcionamento das aplicações a desenvolver.

No caso específico deste trabalho Prático, os alunos simularão a empresa que utilizará esta aplicação e serão eles que desse modo substituirão a empresa na especificação das regras de negócio a utilizar no desenvolvimento e funcionamento das aplicações.

Estas regras de negócio são de especificação e entrega obrigatória sendo essa entrega concretizada através da sua inclusão no relatório numa secção própria.

#### 5. Arquitectura de Desenvolvimento e Interacção entre as Partes

Como orientação para o referido, nas figuras seguintes é mostrada uma arquitectura exemplo. Na primeira figura foi excluída a aplicação de gestão de loja dadas as especificidades desta aplicação de gestão jáatrás referidas.



Trabalhos práticos que não utilizem as tecnologias referidas e a arquitectura indicada, serão liminarmente rejeitados e os alunos que os tenham desenvolvido sem o recurso a essas tecnologias e arquitectura estarão reprovados na componente Trabalho Prático e consequentemente na unidade curricular.

## 6. Interface Gráfica de Utilização Obrigatória

Será ainda obrigatório que o Trabalho Prático seja desenvolvido de acordo com o modelo e *template* exemplo indicados de seguida nas figuras, devidamente adaptados à aplicação desenvolvida, mas sem “fugir” ao esquema geral indicado.



Exemplo “Web”



Exemplo “Smartphone/Tablet”



Exemplo “Windows Machine”

Cores, imagens e outros pormenores gráficos poderão ser alterados de acordo com a sensibilidade gráfica e de *UI* do grupo que realizou o trabalho, mas sem “fugir” ao esquema geral indicado.

Poderá, caso seja necessário, ser acrescentado um outro “*slider*” para subcategorias.

Esta *UI* tem de ser a mesma em qualquer dos *devices* e “página” *Web* a usar para acesso à aplicação *frontend*, e suportada num único conjunto geral de instruções *CSS* e a serem utilizadas por todas as interfaces gráficas das aplicações de “*frontend*”, admitindo-se, no entanto, diferenças/falhas mínimas no que é apresentado em cada device ou Web atendendo-se a que quem realizou o trabalho prático não é um especialista em *design* gráfico. No entanto a *UI* e a *UX* deve ser bastante agradável e rica dentro das eventuais diferenças/mínimas já referidas.

Não se aceitarão trabalhos práticos desenvolvidos com base noutros modelos ou *templates* que não os exemplificados e indicados atrás e nesse caso esses eventuais trabalhos práticos serão liminarmente excluídos.

## 7. Descrição Genérica dos Produtos à Venda e suas classificações

Os produtos à venda podem ser disponibilizados de vários modos que se entenda possível e adequado de ser disponibilizado.

Uma encomenda pode ser constituída por um ou mais produtos à venda na aplicação e também pode ser adquirido mais do que um item de cada produto à venda.

Os produtos à venda podem ser categorizados e, eventualmente, sub-categorizados de mais do que uma maneira diferente, por exemplo: Empresa A → Vestidos; Empresa A → Vestidos → Cerimónia; Empresa B → Homem → Fatos; Empresa B → Homem ; Empresa B → Complementos → Carteiras; ; Empresa C → Camisas → Verão; Empresa C → Praia , etc.

## 8. Perfis de Utilizadores e respectivas funções

Existem quatro perfis/roles de utilizador:

- **Utilizador Anónimo**, só no caso da aplicação “*frontend*”;
- **Cliente**, só no caso da aplicação “*frontend*”;
- **Funcionário**, só no caso da aplicação de Gestão de Loja;
- **Administrador**, só no caso da aplicação de Gestão de Loja;

A. As principais funcionalidades/características da aplicação frontend, para cada tipo de utilizador, são:

- Utilizador com acesso livre **Utilizador Anónimo**
  - Visualizar e comprar (sem efectivar a compra) os produtos à venda;
  - Tem de ser possível visualizar os produtos por categoria, eventualmente por subcategorias, o seu preço e disponibilidade;
  - Tem de ser possível mostrar um produto em “destaque” escolhido aleatoriamente;
  - Seleccionar produtos e respectivas quantidades para compra;
  - Permitir o registo do utilizador como «Cliente» e estado “Pendente”.
- Utilizador com o role/perfil **Cliente**
  - As mesmas funcionalidades descritas atrás para o utilizador anónimo;
  - Logar-se como “**Cliente**”;
  - Efectivar a compra dos produtos, isto é encomendar e “pagar”, seleccionados para compra;
  - Consultar o histórico das compras que já efetuou;
- Utilizadores com o role/perfil **Funcionário ou Administrador**
  - Na aplicação “*frontend*” não têm nenhuma funcionalidade específica associada a estes dois tipos de utilizador.

B. As principais funcionalidades/características da aplicação de *Gestão de Loja*, para cada tipo de utilizador, são:

- Utilizadores com o role/perfil **Utilizador Anónimo ou Cliente**
  - Na aplicação *Gestão de Loja* não têm nenhuma funcionalidade específica associada a estes dois tipos de utilizador nem mesmo acesso à mesma.
- Utilizadores com o role/perfil **Funcionário ou Administrador**
  - Gestão dos produtos à venda e respectivos preços e stocks:
    - Listar os registos de produtos - com filtros (categorias, disponibilização, etc) e com ordenação e paginação;
    - Adicionar o registo de produtos;
    - Editar o registo de produtos;

- Apagar “físicamente” os registos de produtos (apenas se não existirem vendas desse produto);
  - Ativar ou inativar o registo de produtos de acordo com a sua disponibilidade;
  - Gestão dos preços e stocks dos produtos
  - Gestão das categorias:
    - O mesmo, adaptando a este caso, o indicado atrás para os produtos
  - Gerir os modos de disponibilização:
    - O mesmo, adaptando a estes casos, o indicado atrás para os produtos
  - Gerir as vendas:
    - Listar as vendas - com filtros adequados;
    - Confirmar uma venda;
    - Rejeitar uma venda;
    - Expedir os produtos ao cliente (simular expedição)
  - Gerir os pagamentos (simular pagamento)
  - Não pode apagar nem desativar o seu próprio registo de utilizador.
  - Só os utilizadores com o role/perfil de **Administrador** podem gerir o tipo de utilizador **Funcionário**, nomeadamente atribuir ou retirar esse tipo de perfil;
  - Só os utilizadores com o role/perfil de **Administrador** ou **Funcionário**, podem gerir o tipo de utilizador **Cliente**, nomeadamente atribuir ou retirar esse tipo de perfil;
  - Só os utilizadores com o role/perfil de **Administrador** ou **Funcionário**, podem alterar o estado de “Pendente” no tipo de utilizador **Cliente**, de “Pendente” para “Activo” e vice-versa;
- C. As principais funcionalidades/características da aplicação *API RestFull*, para cada tipo de utilizador, são:
- Nenhum tipo de utilizador interagirá directamente com esta aplicação. As interacções com a mesma serão somente feitas indirectamente pela aplicação “*frontend*” através de chamadas a esta *API*.
- ## 9. Operações Genéricas a serem realizadas por cada uma das aplicações a desenvolver e diferentes plataformas
- Em qualquer das aplicações *frontend* desenvolvidas, quer para a Web, Android, iOS, etc, as operações/funcionalidades devem ser exactamente as mesmas.
  - Para o utilizador **Utilizador Anónimo** na aplicação *frontend*, deve ser possível a visualização, selecção e aquisição de qualquer produto à venda e disponível para tal na aplicação.
  - Para o utilizador **Cliente** deve ser também possível que na aplicação *frontend*, além do referido, a efectivação da aquisição de produtos, ou seja, encomendar e pagar (pagamento simulado), de qualquer produto à venda e disponível para tal na aplicação.
  - Na aplicação de Gestão de Loja, tem de ser possível aos utilizadores com o role/perfil de **Administrador/Funcionário** executar todas as funções referidas no ponto 8 para estes dois tipos de utilizadores.

- Na aplicação **API RestFull**, nenhum tipo de utilizador terá acesso directo à aplicação

## 10. Elementos de Entrega Obrigatória

### a. Base de Dados

- Para as várias aplicações desenvolvidas e interfaces de UI, *Web + Android + iOS + macOS + Tizen*, a base de dados deve ser uma só e concretizada através do *SQLServer (localdb)*. A base de dados deve estar bem estruturada, completa e consistente, de forma a simplificar o desenvolvimento (atual e futuro) das aplicações ora desenvolvidas.
- Como já referido, a estrutura de dados utilizada tem de ter por base um Modelo-Entidade-Relacionamento, *MER*, também de entrega obrigatória, o qual deve reflectir todas as necessidades em termos de Entidades e respectivas Relações nas aplicações, concretizado nas aplicações através de classes de domínio codificadas em *C#* e correspondentes a tabelas da base de dados criadas com recurso a instruções *migration*.
- A edição/manipulação dos dados, relativamente à base de dados, tem de ser efetuada através da *Entity Framework Core* e *LINQ*.
- É obrigatória a entrega de uma base de dados submetida conjuntamente com o restante trabalho, a qual tem de estar preenchida/populada com uma quantidade suficiente de dados que permita a demonstração e a avaliação de todas as funcionalidades implementadas. O não cumprimento deste quesito obrigatório resultará numa rejeição liminar do trabalho submetido.

### b. Relatório Obrigatório

- O relatório é de realização e entrega obrigatória e deverá conter não somente a indicação de credenciais de acesso mas outros aspectos, como o objectivo do trabalho, se esse objectivo foi ou não alcançado, a completude do que foi realizado, as principais dificuldades sentidas na sua realização e o modo como foram superadas bem como uma análise autocritica do que foi implementado.
- No relatório, têm de ser ainda incluídas, em secção própria, as “regras de negócio” consideradas para a realização do trabalho.

### c. Código Desenvolvido

- Todo o código desenvolvido e necessário ao funcionamento da aplicação tem de ser obrigatoriamente entregue. A sua não entrega é motivo para exclusão liminar do trabalho prático submetido.

## 11.Avaliação

Critérios gerais para a avaliação do trabalho prático:

- A adequação e a abrangência da solução desenvolvida relativamente aos objetivos enunciados. A quantidade do trabalho realizado deve ser relevante e o âmbito do problema não deve estar reduzido de forma excessiva, valoração desta componente 5%.
- A completude e a consistência dos dados de demonstração das funcionalidades da aplicação. Os dados de demonstração têm de permitir, de um modo conveniente, o correcto funcionamento de todas as funcionalidades da aplicação dentro do domínio do problema apresentado, valoração desta componente 5%.
- A completude e a consistência do relatório entregue, valoração desta componente 15%.
- O desempenho individual de cada um dos dois alunos do grupo na defesa, nomeadamente a sua capacidade para explicar e apresentar o trabalho submetido, a sua capacidade para realizar pequenas alterações ao código entregue bem como a sua capacidade, se lhe for pedido, para apresentar, explicar e justificar trechos de código específico que o grupo tenha utilizado e entregue com a submissão do trabalho e a justificação das opções tomadas e implementadas serão objecto de avaliação e valoração específica, valoração desta componente 30%.
- A qualidade do código desenvolvido e a consistência entre as diversas “partes” das aplicações será bastante valorada e com parâmetros e avaliação e valoração específica. Note-se que a estrutura de classes e a estrutura de dados devem representar convenientemente o domínio do problema e definir uma solução consistente e com qualidade, valoração desta componente 45%.

## 12.Prazo de Entrega

A data-limite para a **entrega do trabalho prático** é o dia **29 de Dezembro de 2024**. Não serão aceites trabalhos entregues após essa data. Os alunos que não façam esta entrega até esta data limite serão considerados como não tendo realizado o trabalho prático. Após a entrega não é possível alterar o trabalho que já foi entregue. Os trabalhos serão entregues via plataforma *moodle*. Aquando da entrega o grupo de alunos que realizou o trabalho escolherá na plataforma *moodle* o dia e hora da defesa do trabalho dentro das opções disponíveis constantes nessa plataforma.

## 13.Modos de Entrega

A entrega do trabalho prático é realizada em formato digital no *Moodle* da UC (um ficheiro *.ZIP* com a seguinte designação obrigatória: nomealuno1numaluno1+ nomealuno2numaluno2.zip).

Para além do projeto, que tem de incluir todo o código fonte, a base de dados populada, as imagens, o relatório e outros elementos que tenham sido utilizados, o ficheiro *.ZIP* deve incluir um ficheiro em formato *PDF* com a seguinte informação:

1. Nomes completos + Números de aluno + Ramo + Curso (D, PL, CE).
2. Dados de acesso à aplicação *Web*, tais como o *username*, a *password* e o perfil, associados aos diversos “utilizadores exemplo”.