



Universidade do Minho
Escola de Engenharia
Licenciatura em Engenharia Informática

Unidade Curricular de Laboratórios de Informática IV

Ano Letivo de 2022/2023

eFeiras

João Luís Silva Ribeiro, A69858
Miguel Silva Pinto, A96106
Orlando José da Cunha Palmeira, A97755
Pedro Miguel Castilho Martins, A97613

15 de janeiro de 2023

L | 4

Data de Recepção	
Responsável	
Avaliação	
Observações	

eFeiras

João Luís Silva Ribeiro, A69858
Miguel Silva Pinto, A96106
Orlando José da Cunha Palmeira, A97755
Pedro Miguel Castilho Martins, A97613

15 de janeiro de 2023

Resumo

No âmbito da unidade curricular de Laboratórios de Informática IV foi-nos proposto o desenvolvimento de uma aplicação de feiras online.

Numa primeira fase, analisámos o problema de maneira a caracterizá-lo, identificando os motivos, objetivos e a viabilidade da implementação desta aplicação.

Numa fase posterior, definimos os requisitos funcionais e não funcionais da aplicação bem como a sua especificação. Na especificação foram abordados os aspetos estruturais e comportamentais do sistema.

Na última fase, implementamos a aplicação que idealizamos nas duas fases anteriores, seguindo a documentação criada. Na implementação da aplicação tivemos o foco em apenas algumas das funcionalidades da aplicação uma vez que não tivemos tempo de implementar tudo aquilo que foi planeado.

Área de Aplicação: Engenharia de Software, Bases de Dados Relacionais, Programação Web

Palavras-Chave: Feiras *Online*, Planeamento de software, UML, C#, Bases de Dados Relacionais, SQL Server, Plataforma .NET Core, Desenvolvimento Web.

Índice

1	Introdução	1
1.1	Contextualização	1
1.2	Fundamentação	1
1.3	Objectivos	1
1.4	Viabilidade	2
1.5	Recursos a utilizar	3
1.6	Equipa de trabalho	5
1.7	Plano de execução de trabalho	6
2	Levantamento e Análise de Requisitos	7
2.1	Apresentação da estratégia e método	7
2.2	Descrição geral dos requisitos levantados	8
2.2.1	Requisitos funcionais	8
2.2.2	Requisitos não funcionais	14
2.3	Validação dos requisitos estabelecidos	15
3	Especificação e Modelação do Software	16
3.1	Apresentação geral da especificação	16
3.2	Aspetos estruturais	17
3.3	Aspetos comportamentais	19
3.3.1	<i>Use cases</i> de utilização geral	20
3.3.2	<i>Use cases</i> do cliente	21
3.3.3	<i>Use cases</i> do vendedor	22
3.3.4	<i>Use cases</i> do administrador	24
3.3.5	Diagramas de atividades	26
4	Conceção do sistema de dados	29
4.1	Apresentação geral da estrutura do sistema de dados	30
4.2	Descrição detalhada dos vários elementos de dados e seus relacionamentos	31
4.2.1	Utilizador	31
4.2.2	Feira	32
4.2.3	Categoria	32
4.2.4	Subcategoria	33
4.2.5	Compra	33
4.2.6	Produto	34
4.2.7	Banca	35
4.2.8	Relação M-N <i>Banca_has_Produto</i>	35

4.2.9 Relação M-N <i>Compra_has_Produto</i>	36
5 Esboço dos interfaces do sistema	37
5.1 Estrutura geral das interfaces do sistema	37
5.2 Caracterização das interfaces	38
5.2.1 Página de apresentação de feiras (comprador e vendedor)	38
5.2.2 Páginas de apresentação das bancas de uma feira (comprador e vendedor)	39
5.2.3 Página dos produtos de uma banca (comprador)	40
5.2.4 Configuração de uma banca (vendedor)	41
5.2.5 Carrinho de compras	42
6 Implementação da Aplicação	43
6.1 Apresentação e descrição do processo de implementação realizado	43
6.2 Apresentação dos serviços implementados e estrutura final da aplicação	44
6.2.1 Serviços implementados	44
6.2.2 Estrutura final da aplicação	49
6.3 Análise e avaliação da aplicação desenvolvida	50
6.4 Ferramentas Utilizadas	51
7 Conclusões e trabalho futuro	53
Lista de Siglas e Acrónimos	56
Anexos	58
Use case - Criar conta	58
Use case - <i>Login</i>	59
Use case - Adição de produtos no carrinho	59
Use case - Alterar carrinho de compras	60
Use case - Finalização de uma compra	60
Use case - Criar produto	61
Use case - Configuração de um produto	61
Use case - Remoção de um produto	61
Use case - Solicitação de uma banca de uma feira	62
Use case - Configuração de uma banca de uma feira	62
Use case - Aprovação da conta de um vendedor	63
Use case - Criação de uma feira	63
Mockup - Página de autenticação	64
Mockup - Página de criação de conta	64
Mockup - Histórico de compras	65
Instruções SQL para a criação das tabelas da base de dados	65

Lista de Figuras

1.1	Plano de trabalho - Definição e Fundamentação	6
1.2	Plano de trabalho - Especificação	6
1.3	Plano de trabalho - Construção	6
3.1	Modelo de domínio	17
3.2	Diagrama de <i>use cases</i>	19
3.3	Diagrama de atividades do cliente	26
3.4	Diagrama de atividades do vendedor	27
3.5	Diagrama de atividades do administrador	28
4.1	Modelo lógico da base de dados do sistema	30
4.2	Dicionário de dados - Utilizadores	31
4.3	Dicionário de dados - Feiras	32
4.4	Dicionário de dados - Categorias	32
4.5	Dicionário de dados - Subcategorias	33
4.6	Dicionário de dados - Compras	33
4.7	Dicionário de dados - Produtos	34
4.8	Dicionário de dados - Bancas	35
4.9	Dicionário de dados - relação <i>Banca_has_Produto</i>	35
4.10	Dicionário de dados - relação <i>Compra_has_Produto</i>	36
5.1	Esquema das interfaces do sistema	37
5.2	Página de apresentação de feiras	38
5.3	Página de apresentação das bancas de uma feira (comprador)	39
5.4	Página de apresentação das bancas de uma feira (vendedor)	39
5.5	Página dos produtos de uma banca (comprador)	40
5.6	Configuração de uma banca (vendedor)	41
5.7	Carrinho de compras	42
6.1	Estrutura e comunicação das camadas da aplicação	43
6.2	Página de <i>login</i>	44
6.3	Página de criação da conta	45
6.4	Página principal	45
6.5	Página de visita de uma feira (comprador)	46
6.6	Página de visita de uma banca (comprador)	46
6.7	Página de visita do carrinho de compras	47
6.8	Página de visita de uma feira (vendedor)	48

6.9	Página de visita de uma feira (vendedor)	48
6.10	Página de configuração de uma banca	49
6.11	Estrutura final da aplicação	50

Lista de Tabelas

1.1	Equipa de trabalho.	5
-----	---------------------	---

1 Introdução

1.1 Contextualização

A definição de feira diz-nos que é um evento onde se expõe e se vende variados produtos, normalmente ligado a um âmbito (ex: electrónica, gastronomia,...) e possivelmente também ligado a um tema, com uma data ou intervalo de datas definidas. A Câmara de Braga organiza variados tipos de feiras, como a feira semanal ou a Braga Romana, de modo a suprimir os desejos comerciais e culturais da sua população. Devido ao crescimento da popularidade do comércio digital nas faixas etárias mais jovens, e sabendo que o Concelho de Braga tem uma grande população jovem, a Câmara de Braga viu uma oportunidade de inovar e trazer uma nova modalidade de feiras à sua população, que sirva como uma alternativa às feiras tradicionais. Assim nasce este projeto de uma plataforma para sustentar estas feiras.

1.2 Fundamentação

Devido ao aumento da disponibilidade de serviços que fornecem produtos na comodidade das casas dos seus clientes e o número crescente de utilizadores de Internet, a popularidade de feiras presenciais tem vindo a decrescer. Para além do mais, a chegada da pandemia trouxe ainda mais problemas a este tipo de atividade devido à sua natureza presencial. Todos estes problemas poderiam vir a ser resolvidos com a criação de uma aplicação de feiras online que juntava a comodidade do online com as características acolhedoras e sociais de uma feira.

1.3 Objectivos

Através do desenvolvimento desta aplicação, pretendemos formar um meio de realização de feiras mais desenvolvido e eficiente, permitindo:

- Colmatar dificuldades relacionadas com o espaço físico necessário para a realização de feiras.
- Fornecer alternativas de comércio aos vendedores.
- Poder alargar as fronteiras dos vendedores bracarenses para outros municípios do país

(ou até numa visão mais ambiciosa, para outros países).

- Conhecer os hábitos de compras dos clientes permitindo saber quais os produtos que agradam mais (ou menos).
- Atrair as gerações mais novas a este tipo de comércio.
- Poder aumentar a variedade de produtos vendidos além dos que se vendem em feiras tradicionais.
- Permitir a obtenção de lucros à Câmara de Braga, uma vez que os vendedores terão de pagar a utilização deste serviço.

1.4 Viabilidade

Segundo as estatísticas do site AppBrain, a categoria *Shopping* encontra-se em 6.º lugar na loja de aplicações Google Play. Já na loja de aplicações App Store (Apple), esta categoria encontra-se em 7.º lugar, segundo o site Statista. É importante referir que existem cerca de 49 categorias diferentes de aplicações.

Ainda nas estatísticas, algumas das aplicações mais populares na categoria *Shopping* chegam a ter mais de 500 milhões de downloads, segundo informações do site AppBrain.

A partir do que foi referido anteriormente, verificamos que a categoria *Shopping* (onde o nosso projecto se encontra) é bastante popular e tem uma forte adesão por parte dos utilizadores.

Partindo agora para o nosso projecto em concreto, tentámos encontrar aplicações que se encontrem dentro do nosso tema, as Feiras. Após muito tempo de pesquisa, não conseguimos encontrar nenhuma aplicação que permita a realização de feiras online, o que pode indicar que o nosso projecto será pioneiro nesta matéria. Para além disso, dado que esta aplicação se encontra numa categoria muito popular, aborda um tema novo e é financiada por uma entidade governamental, prevemos que a sua popularidade atinja um patamar satisfatório num prazo relativamente curto. Desta forma, será possível atrair um grande número de utilizadores (nomeadamente vendedores, uma vez que irão pagar pela utilização deste serviço) de Braga, mas também de outros municípios do país, potencializando o retorno financeiro à Câmara Municipal de Braga.

1.5 Recursos a utilizar

O desenvolvimento e manutenção de uma aplicação exigem uma série de recursos diferentes, cada um deles crucial para garantir o sucesso do projeto. Estes recursos podem incluir pessoas, tecnologias e infraestruturas.

No nosso projeto, será necessária uma equipa de desenvolvimento composta por quatro elementos para criar e manter uma aplicação de qualidade. Além disso, é importante envolver os intervenientes da **CMB** responsáveis pelo projeto que serão responsáveis por fornecer informações sobre o que pretendem da aplicação desenvolvida bem como aprovar tudo o que vai sendo feito ao longo de todo o processo de desenvolvimento. Também serão necessários funcionários de gestão dos utilizadores, vendedores e feiras que são essenciais para garantir que a aplicação esteja a funcionar de forma otimizada e para resolver quaisquer problemas que possam ocorrer.

Tecnologias como sistemas de gestão de bases de dados e ambientes de desenvolvimento são também cruciais para o bom funcionamento da aplicação. Um **SGBD** é necessário para se poder armazenar e gerir de forma eficiente os dados da aplicação e um ambiente de desenvolvimento é necessário para auxiliar no processo de construção.

Finalmente, será necessário ter servidores suficientes para suportar o funcionamento da aplicação e garantir a disponibilidade para os utilizadores. Prevemos que, no início, serão necessários três servidores.

Todos estes recursos são fundamentais para assegurar que a aplicação seja desenvolvida e mantida de forma eficiente, e é importante garantir que estão disponíveis e que são utilizados de forma adequada para garantir o sucesso do projeto.

Sucintamente, para a aplicação que vamos desenvolver, serão necessários os seguintes recursos:

- Humanos:
 1. Equipa com quatro elementos responsáveis pelo desenvolvimento e implementação da aplicação.
 2. Intervenientes da CMB responsáveis pelo projeto.
 3. Funcionários responsáveis pela gestão de clientes, vendedores e feiras.
- Materiais:
 1. *Software*:
 - Um sistema de gestão de bases de dados.
 - Um ambiente de desenvolvimento.

- Ferramentas de apoio à gestão do projeto e de elaboração da documentação.

2. *Hardware:*

- Três servidores.

1.6 Equipa de trabalho

Para o desenvolvimento do nosso projeto, foi organizada uma equipa de trabalho composta por vários elementos com funções em diferentes áreas, em que a colaboração de todos é essencial para o desenvolvimento bem sucedido da aplicação.

Apresenta-se, de seguida, a equipa organizada para este projeto:

Pessoal interno:

Intervenientes da Câmara Municipal de Braga responsáveis pelo projeto (são responsáveis por informar o que pretendem da aplicação desenvolvida bem como aprovar tudo o que vai sendo feito ao longo de todo o processo de desenvolvimento).

Pessoal externo:

Uma equipa de trabalho constituída por 4 alunos de Engenharia Informática da Universidade do Minho:

Número	Nome	Função	Foto
A97755	Orlando José da Cunha Palmeira	Coordenador de projeto	
A97613	Pedro Miguel Castilho Martins	Testador	
A96106	Miguel Silva Pinto	Engenheiro de Software	
A69858	João Luís Silva Ribeiro	Analista	

Tabela 1.1: Equipa de trabalho.

Outros:

- Visitantes/Cientes de feiras (essenciais na obtenção de informações de como deve funcionar a aplicação no lado do cliente)
- Vendedores de feiras (essenciais na obtenção de informações de como deve funcionar a aplicação para os vendedores)

1.7 Plano de execução de trabalho

De maneira a planejar uma boa execução do projeto, identificaram-se etapas distintas, nas três etapas do projeto, para dividir o trabalho em tarefas mais elementares, e assim se poder estimar melhor datas da duração de uma dada tarefa. Também se indicou um ou mais elementos da equipa de trabalho, para serem responsáveis pela correta e atempada execução de cada uma dessas mesmas tarefas.

Com o intuito de registar esse mesmo planeamento, usamos diagramas de Gantt, os quais se apresentam a seguir.

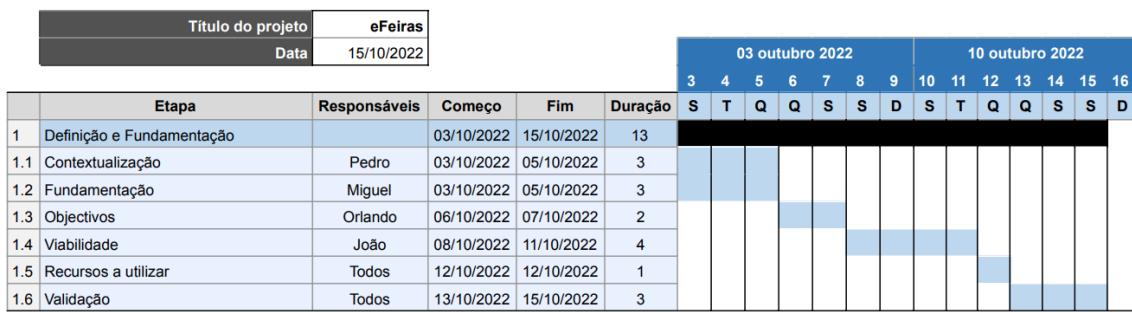


Figura 1.1: Plano de trabalho - Definição e Fundamentação



Figura 1.2: Plano de trabalho - Especificação



Figura 1.3: Plano de trabalho - Construção

2 Levantamento e Análise de Requisitos

2.1 Apresentação da estratégia e método

O levantamento e análise de requisitos é um dos aspetos mais importantes e cruciais no desenvolvimento de um projeto de *software*. Para termos uma boa base de desenvolvimento do projeto, adotámos vários métodos de levantamento de requisitos.

Primeiramente, foi organizada uma ida a uma feira local com o objetivo de se entender melhor o modo como este tipo de comércio se desenrola de maneira presencial, o que pode auxiliar no surgimento de ideias e requisitos e na familiarização com o processo que vamos implementar no nosso projeto.

De seguida, recorremos a entrevistas, questionando feirantes (a entidade mais importante no nosso sistema), uma vez que sem eles, a nossa aplicação não existe, sendo de extrema importância obter opiniões, bem como saber quais as rotinas destes utilizadores. Ocasionalmente também conversamos com os clientes, de maneira a ter uma melhor perspetiva da experiência que os compradores gostariam de ter na nossa aplicação.

2.2 Descrição geral dos requisitos levantados

Enumeram-se, de seguida, os diversos requisitos levantados pela equipa. Os requisitos encontram-se separados por requisitos **funcionais** e **não funcionais** e foram devidamente numerados para facilitar a sua referência quando necessário.

2.2.1 Requisitos funcionais

RF01 - Registo do utilizador

- Requisitos do utilizador:
 1. A aplicação deve permitir a um utilizador criar uma conta para utilizar este serviço.
- Requisitos do sistema:
 1. O sistema deverá solicitar o nome, CC, NIF, morada, *username*, *password*, *e-mail*, tipo de conta (cliente, vendedor).
 2. O sistema não deverá permitir a existência de utilizadores com o mesmo *username*, *e-mail*, CC ou NIF e deverá notificar o utilizador dessa ocorrência.
 3. Caso a conta seja de vendedor, o sistema deverá registá-la com aprovação pendente para depois ser aprovada por um administrador.
 4. O sistema deverá atribuir um ID único ao novo utilizador.

RF02 - Autenticação na aplicação

- Requisitos do utilizador:
 1. A aplicação deve permitir ao utilizador aceder à sua conta.
- Requisitos do sistema:
 1. O sistema deverá validar os dados introduzidos pelo utilizador e fornecer o acesso à sua conta.
 2. Caso os dados sejam inválidos (*password* incorreta, utilizador inexistente, etc...), o sistema deverá apresentar uma mensagem de erro ao utilizador impedindo o acesso à conta.

RF03 - Adicionar um produto ao carrinho de compras

- Requisitos de utilizador:
 1. A aplicação deve permitir a um cliente adicionar um produto ao seu carrinho de compras.
- Requisitos do sistema:
 1. O sistema deverá adicionar o(s) produto(s) selecionado(s) no carrinho de compras do utilizador.
 2. O sistema deverá garantir que a quantidade pretendida pelo utilizador não é superior à quantidade de produto(s) disponível.

RF04 - Finalizar uma compra

- Requisitos de utilizador:
 1. A aplicação deve permitir a um utilizador autenticado finalizar a sua compra dos produtos no seu carrinho.
- Requisitos do sistema:
 1. O sistema deverá garantir que o comprador tem as condições para efetuar a compra (tais como: cartão de crédito válido; ter, pelo menos, um produto no seu carrinho de compras).
 2. O sistema deverá ainda garantir que a compra não deve ser realizada se algum produto constante no carrinho de compras do utilizador não se encontrar disponível, isto é, a quantidade pretendida pelo utilizador é superior à disponível.

RF05 - Criação de produtos

- Requisitos do utilizador:
 1. A aplicação deve permitir a um vendedor criar um produto para vender.
- Requisitos do sistema:
 1. O sistema deverá garantir que o utilizador autenticado é um vendedor.
 2. O sistema deverá solicitar o tipo de produto, nome, descrição, preço, quantidade e uma imagem.
 3. O sistema deverá garantir que a conta do utilizador não tem um produto igual ao que pretende adicionar.

- O sistema deverá armazenar o produto, atribuindo-lhe um ID único, e associá-lo à conta do vendedor.

RF06 - Configuração de produtos

- Requisitos do utilizador:
 - A aplicação deve permitir a um vendedor alterar certos parâmetros do seu produto.
- Requisitos do sistema:
 - O sistema deve garantir que o vendedor só pode alterar a quantidade disponível, o nome, a descrição e o preço.
 - O sistema deve registar todas as alterações efetuadas pelo vendedor na base de dados.

RF07 - Remoção de produtos

- Requisitos do utilizador:
 - A aplicação deve permitir a um vendedor remover um produto da sua conta.
- Requisitos do sistema:
 - O sistema deve garantir que o utilizador autenticado é um vendedor.
 - O sistema deve garantir que o produto é removido da base de dados.

RF08 - Configurar catálogo de produtos de uma banca

- Requisitos do utilizador:
 - A aplicação deve permitir a um vendedor adicionar e remover produtos na banca que lhe está alocada numa feira.
- Requisitos do sistema:
 - O sistema deverá garantir que o utilizador autenticado é um vendedor.
 - Se o vendedor efetuar uma remoção de um produto da banca, o sistema deverá garantir que esse mesmo produto é dissociado da banca na base de dados.
 - Se o vendedor adicionar um produto, o sistema deverá garantir que a categoria do produto é coerente com a categoria da feira, impedindo a adição desse produto

caso essa condição não se verifique. Se a condição se verificar, o sistema deve garantir que o produto é associado à banca na base de dados.

RF09 - Criação de um vendedor

- Requisitos do utilizador:
 1. A aplicação deve permitir a criação de contas de vendedores mediante a aprovação de um administrador.
- Requisitos do sistema:
 1. O sistema deverá garantir que o utilizador autenticado é um administrador.
 2. O sistema deverá permitir ao administrador visualizar as contas de vendedores que ainda não foram aprovadas.
 3. O sistema deverá permitir ao administrador aprovar uma conta de vendedor caso considere que essa conta reúne condições para ser aprovada. Nesse caso, o sistema deve marcar a conta de vendedor como aprovada.
 4. O sistema deverá eliminar uma conta de vendedor da base de dados caso o administrador não a aprove.

RF10 - Criação de feiras

- Requisitos de utilizador:
 1. A aplicação deve permitir aos administradores criarem novas feiras.
- Requisitos do sistema:
 1. O sistema deverá garantir que o utilizador autenticado é um administrador.
 2. O sistema deverá solicitar o nome, âmbito, data de início, data de fim, uma imagem ilustrativa e a capacidade da feira (n.º de bancas).
 3. O sistema não deverá permitir a criação de feiras com o mesmo nome.
 4. O sistema deverá avisar o administrador caso exista outra feira com o mesmo nome.
 5. O sistema deverá atribuir um ID único à nova feira.

RF11 - Visualização do catálogo de feiras

- Requisitos de utilizador:
 1. A aplicação deve permitir a um utilizador visualizar as feiras que estão em curso.
- Requisitos do sistema:
 1. O sistema deve garantir que o utilizador autenticado é um cliente ou vendedor.
 2. O sistema deve obter as feiras que ainda estão em curso, isto é, as feiras cujo prazo ainda não venceu.
 3. Se o utilizador autenticado for um cliente, o sistema deve mostrar as diversas feiras em curso, mostrando o intervalo de datas, o título e a categoria.
 4. Se o utilizador autenticado for um vendedor, o sistema deve mostrar as diversas feiras em curso, mostrando o intervalo de datas, o título, a categoria e o número de bancas disponíveis.

RF12 - Visualização das bancas de uma feira

- Requisitos de utilizador:
 1. Quando um utilizador entra numa feira, a aplicação deve permitir visualizar as bancas que estão disponíveis. .
- Requisitos do sistema:
 1. O sistema deverá garantir que o utilizador autenticado é um cliente ou vendedor.
 2. Se o utilizador autenticado for um cliente, o sistema deverá apresentar todas as bancas disponíveis, indicando o nome do seu responsável (vendedor que possui a banca).
 3. Se o utilizador autenticado for um vendedor, o sistema deverá apresentar todas as bancas disponíveis, indicando o nome do seu responsável, bem como o número de bancas que ainda podem ser adquiridas por vendedores.

RF13 - Visualização dos produtos de uma banca

- Requisitos de utilizador:
 1. Quando um utilizador entra numa banca de uma determinada feira, a aplicação deverá permitir visualizar os produtos disponíveis nessa banca.
- Requisitos do sistema:

1. O sistema deve garantir que o utilizador autenticado é um cliente ou vendedor.
2. O sistema deverá apresentar todos os produtos disponíveis numa banca, mostrando o seu nome, descrição, custo unitário e a respetiva imagem.

RF14 - Visualização do histórico de compras

- Requisitos de utilizador:
 1. A aplicação deve permitir a um cliente consultar o histórico de todas as suas compras.
- Requisitos do sistema:
 1. O sistema deverá garantir que o cliente autenticado é um cliente.
 2. O sistema deverá obter da base de dados todas as compras feitas pelo cliente e apresentá-las.

RF15 - Configuração do carrinho de compras

- Requisitos de utilizador:
 1. A aplicação deve permitir a um cliente alterar o conteúdo do seu carrinho de compras, isto é, mudar a quantidade de um produto presente e remover algum dos produtos presentes.
- Requisitos do sistema:
 1. Quando o cliente altera a quantidade de um produto no carrinho, o sistema deverá garantir que a nova quantidade presente no carrinho deverá ser igual ou inferior à quantidade disponível desse produto.
 2. Quando um utilizador decrementa a quantidade de um certo produto, o sistema deverá remover esse produto do carrinho de compras se a nova quantidade for zero.

RF16 - Obtenção de uma banca numa feira

- Requisitos de utilizador:
 1. A aplicação deve permitir a um vendedor solicitar uma banca de uma feira.
- Requisitos do sistema:
 1. O sistema deve garantir que o utilizador autenticado é um vendedor.
 2. O sistema deverá garantir que não atribui nenhuma banca se a capacidade da feira

- atingiu o seu limite.
3. Caso a capacidade da feira não tenha atingido o seu limite, o sistema deve atribuir uma banca ao vendedor.
 4. Se o vendedor já tiver uma banca na feira, o sistema deve impedir que lhe seja atribuída outra banca.

RF17 - Visualização do histórico de vendas

- Requisitos de utilizador:
 1. A aplicação deve permitir a um vendedor visualizar o histórico de todas as suas vendas.
- Requisitos do sistema:
 1. O sistema deve garantir que o utilizador autenticado é um vendedor.
 2. O sistema deverá obter da base de dados todas as vendas feitas pelo vendedor e apresentá-las.

2.2.2 Requisitos não funcionais

RNF01 - Ambiente de execução do programa

- A parte do sistema que interage com os utilizadores deverá executar a partir de um *browser* de internet, e a parte do armazenamento de dados e da lógica de negócio deverá correr em ambiente Microsoft Windows.

RNF02 - Ferramentas a usar no desenvolvimento do programa

- Para a elaboração do sistema, terá de se usar obrigatoriamente a framework .NET e por consequente a linguagem de programação C#. Também deverá ser usado o sistema Microsoft SQL Server para a gestão da base de dados necessária para o funcionamento do sistema.

RNF03 - Dados armazenados em conformidade com o RGPD

- O armazenamento de dados pessoais dos utilizadores, necessários para o funcionamento do sistema deverá, obrigatoriamente, de estar de acordo com o Regulamento Geral sobre a Proteção de Dados.

RNF04 - Tempo de execução das funcionalidades interativas

- Todas as funcionalidades do sistema que incluam interação direta com um utilizador, deverão de ser executadas, preferencialmente num tempo menor que 3 segundos, e obrigatoriamente em menos de 10 segundos.

RNF05 - Uso do método PBKDF2 para a proteção das passwords

- As passwords dos utilizadores deverão ser *hashed* usando a implementação nativa .NET do método PBKDF2, para garantir segurança.

RNF06 - As contas de administrador não são criadas pelo sistema

- O sistema não deverá conseguir criar contas de administrador. Estas contas deverão ser criadas com a inserção manual na base de dados, por um perfil com privilégios para tal.

2.3 Validação dos requisitos estabelecidos

Por forma a validar os requisitos levantados, levamos todas a ideias que angariamos através das nossas experiências e entrevistas ao cliente para obter uma avaliação inicial das ideias bem como receber novas indicações e sugestões para o nosso projeto.

Após todos os requisitos serem validados pelo cliente passamos à etapa seguinte do projeto, a especificação e modelação do Software.

3 Especificação e Modelação do Software

3.1 Apresentação geral da especificação

A partir dos requisitos definidos, é possível perceber como a aplicação deve estar construída bem como definir qual deve ser o seu comportamento.

Para isso, recorremos à notação **UML** para auxiliar no desenvolvimento de diagramas que suportem a especificação do *software* ao nível estrutural e ao nível comportamental.

Em termos estruturais, construímos o **modelo de domínio** para ilustrar quais são as entidades envolvidas no nosso sistema bem como as relações existentes entre elas.

Em termos comportamentais, começamos por definir um conjunto de *use cases* que especificam as funcionalidades principais do programa. Seguidamente, construímos um **diagrama de use cases** que indica quais os atores envolvidos em cada *use case*. Finalmente, construímos **diagramas de atividades** para demonstrar o funcionamento da aplicação para os clientes, vendedores e os administradores.

3.2 Aspectos estruturais

Na definição dos aspectos estruturais, é necessário identificar e compreender o conjunto de entidades e relações que existem no domínio a que o sistema se destina.

Por esse motivo, decidimos elaborar o modelo de domínio, uma vez que fornece uma representação visual bastante clara das entidades do sistema e das relações entre elas. Além disso, o modelo de domínio pode ser visto como um planeamento a alto nível do sistema a desenvolver e, por isso, terá uma grande utilidade na fase de implementação para garantir que a aplicação está a ser desenvolvida como foi planeado.

Apresenta-se, de seguida, o modelo de domínio do nosso sistema.

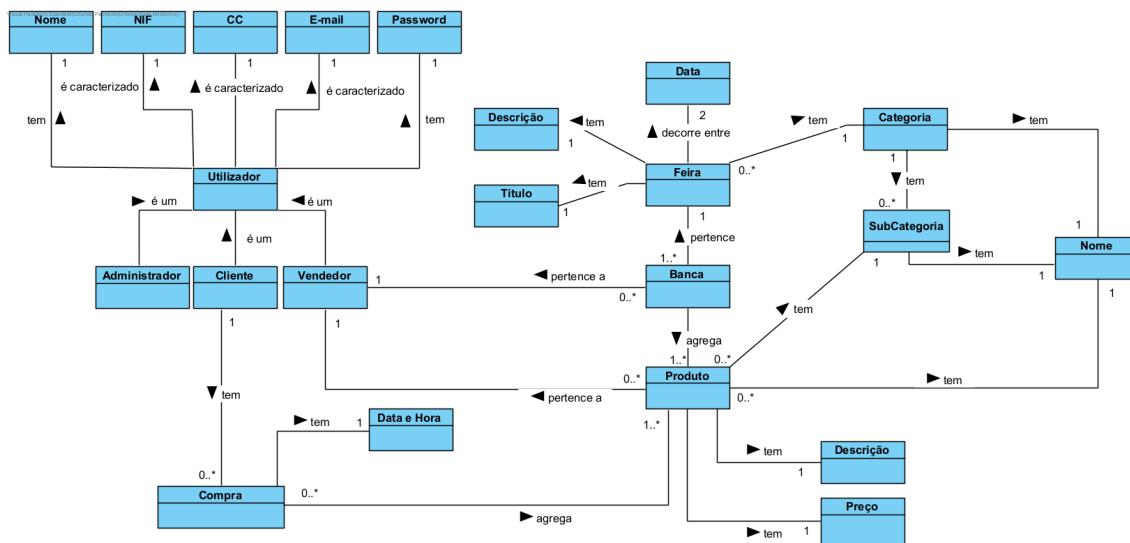


Figura 3.1: Modelo de domínio

Consideraremos como entidades mais relevantes do sistema o Utilizador (e as respetivas especializações), a Feira, a Banca, o Produto, a Categoria e a Subcategoria.

No nosso sistema, o vendedor poderá adquirir bancas para expor os seus produtos nas feiras que estão em curso, sendo que a feira terá uma categoria e o produto uma subcategoria que deverá estar adequada à categoria da feira.

O comprador poderá visitar as diversas feiras em curso e respetivas bancas para ver os produtos expostos.

Existem algumas considerações a ter em conta neste modelo de domínio. A primeira é que não foi possível representar a restrição de que um vendedor só tem acesso a uma banca em cada feira, isto é, o vendedor pode ter, em simultâneo, mais do que uma banca, mas não pode ter mais do que uma banca numa só feira (este aspeto será abordado nos *use cases* e no modelo lógico da base de dados).

Outra consideração a ter em conta são as entidades *Categoria* e *Subcategoria*. No modelo de domínio é indicado que um produto tem apenas uma subcategoria mas não tem uma categoria. No mundo real, isto pode se parecer incorreto. No entanto, no contexto do nosso sistema, as entidades *Categoria* e *Subcategoria* servem para adequar os produtos às feiras.

Segue-se um exemplo de aplicação desta estratégia: Existe a categoria «Alimentos» e uma subcategoria «Legumes». Um produto da subcategoria «Legumes» pode ser colocado numa feira cuja categoria é «Alimentos». Contudo, um produto da subcategoria «Perfumes» não pode ser colocado numa feira da categoria «Legumes». Estas entidades servem para impedir que o sistema coloque produtos numa feira inadequada ao seu tipo.

Obviamente, o utilizador não vai saber que um produto só tem uma subcategoria e não tem uma categoria. O utilizador verá sempre que, tanto as feiras como os produtos, apenas têm categorias, e que as categorias das feiras são mais abrangentes que as categorias dos produtos.

3.3 Aspectos comportamentais

A aplicação que vamos desenvolver foi projetada para atender dois tipos de utilizadores: os compradores e os vendedores (note-se que também existem os administradores, mas estes utilizadores serão apenas responsáveis pela manutenção do sistema). Por isso, é importante prever quais as funcionalidades que cada tipo de utilizador irá ter para poder satisfazer as suas necessidades.

Para isso, iremos apresentar um conjunto de casos de uso da nossa aplicação para os diferentes tipos de utilizadores conforme descrito no diagrama da figura 3.2.

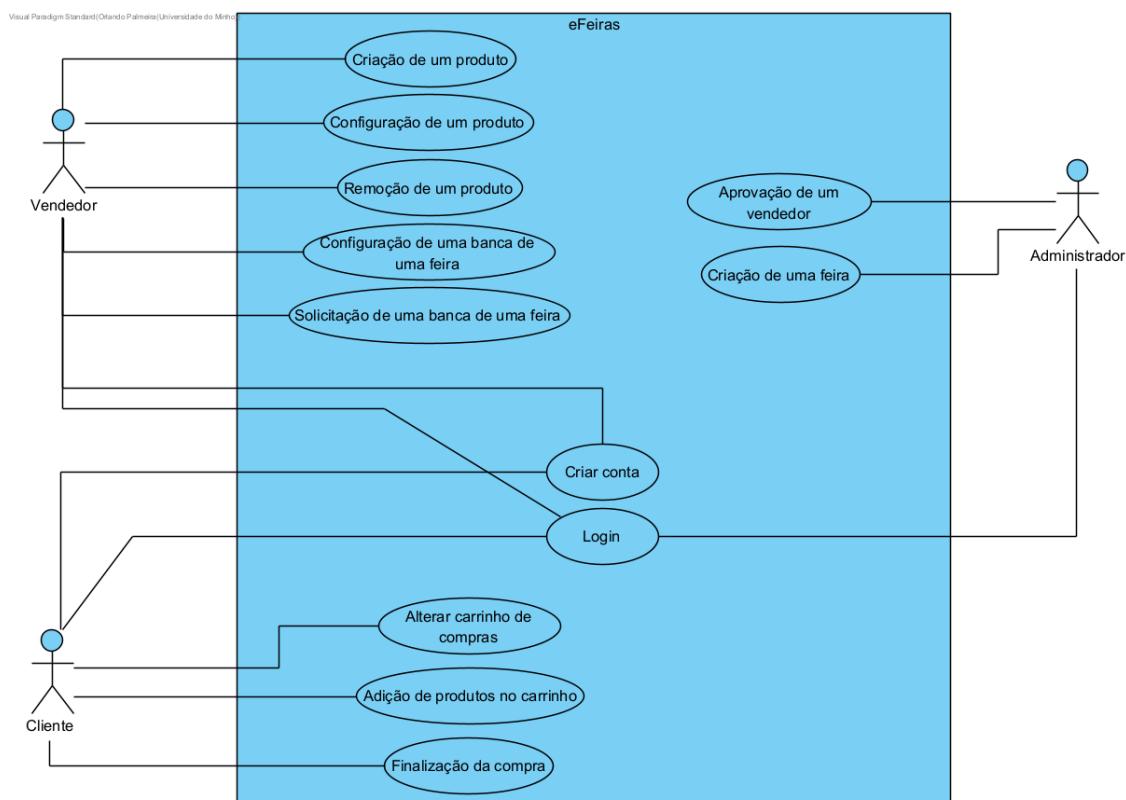


Figura 3.2: Diagrama de *use cases*

Nas próximas secções, iremos descrever cada um dos *use cases* e, nos anexos, estes serão apresentados num formato mais estruturado.

3.3.1 Use cases de utilização geral

Os *use cases* de utilização geral são aqueles que não são exclusivos de um tipo utilizador.

Na nossa aplicação existirão apenas dois *use cases* deste tipo: Criar conta e *login*. Note-se que o *use case* **Criar conta** apenas existe para os clientes e vendedores.

Criar conta

Para poder utilizar a aplicação, um utilizador deverá estar registado. Para isso, na área de registo o utilizador terá de fornecer os seguintes dados:

- Nome
- Número do cartão de cidadão
- Número de identificação fiscal
- Nome de utilizador
- *Password*
- *E-mail*
- Tipo de conta (cliente ou vendedor)
- Apresentação (no caso de ser um vendedor)

No sistema não podem existir dois utilizadores com alguns dos seguintes dados repetidos: número de cartão de cidadão, número de identificação fiscal, nome de utilizador e *e-mail*. Para isso, antes da criação da conta, o sistema vai verificar se já existe algum utilizador com algum dos dados mencionados acima. Se isso acontecer, a aplicação informa o utilizador dessa situação acrescentando que a conta não pode ser criada.

Quando a conta está a ser criada, o sistema ainda verifica qual será o seu tipo. Se for um comprador, a conta é imediatamente criada. Caso seja um vendedor, a conta é criada mas é marcada com **aprovação pendente**, isto é, a conta ficará bloqueada até ser aprovada por um administrador.

Este *use case* satisfaz o requisito **RF01**.

Login

Para poder utilizar as funcionalidades disponibilizadas pela aplicação, o utilizador deverá entrar na sua conta com as respetivas credenciais (*e-mail* e *password*).

Quando as credenciais são fornecidas, o sistema começa por verificar a existência da conta. Se a conta não existir, a aplicação informa que a conta não existe. Caso contrário, o sistema verifica se a *password* fornecida está correta e, se isso não acontecer, o sistema impede o acesso à conta.

Este *use case* satisfaz o requisito **RF02**.

3.3.2 Use cases do cliente

Adição de produtos no carrinho

Ao entrar numa banca de uma feira, o sistema apresenta ao cliente todos os produtos presentes nessa banca. Seguidamente, o utilizador consegue definir as quantidades dos produtos que pretende adicionar ao carrinho de compras.

Após isso, o utilizador informa que pretende adicionar esses produtos ao carrinho e o sistema irá verificar se as quantidades que informou poderão ser satisfeitas tendo em conta o *stock* existente de cada produto. Se a quantidade disponível de algum produto não conseguir satisfazer a pretendida pelo cliente, a aplicação informa o cliente dessa situação e o produto não é adicionado.

Este *use case* satisfaz o requisito **RF03**.

Alterar carrinho de compras

Quando o cliente visita o seu carrinho de compras, a aplicação apresenta todos os produtos que o cliente adicionou e ainda fornece opções para alterar as quantidades pretendidas e até remover produtos.

Se o cliente pretender incrementar a quantidade de um produto, o sistema irá verificar se a quantidade pode ser incrementada, isto é, se a quantidade disponível é igual ou superior à nova quantidade após o incremento. Se for esse o caso, então a alteração é efetuada. Caso contrário a alteração é impedida.

Se o cliente pretender decrementar a quantidade de um produto, o sistema irá verificar se a quantidade após o decremento é igual a 0. Se isso acontecer, o sistema irá eliminar automaticamente o produto do carrinho de compras. Caso contrário, o sistema simplesmente aplica a alteração pretendida pelo utilizador.

Finalmente, se o utilizador pretender eliminar o produto do carrinho de compras, o sistema

elimina-o.

Este *use case* satisfaz o requisito **RF15**.

Finalização da compra

Quando o utilizador se encontra na página do carrinho de compras, a aplicação apresenta-lhe todos os produtos que adicionou bem como o montante total até ao momento.

Se o utilizador confirmar que quer finalizar a compra, o sistema começa por verificar se as quantidades dos produtos constantes no carrinho de compras são possíveis de serem compradas. Se isso não acontecer, a aplicação informa o cliente que existem produtos que não podem ser comprados e informa quais são esses produtos. Caso contrário, a aplicação solicita ao utilizador os dados do cartão de crédito (número, CVV, data de validade e nome do titular).

Se o cartão de crédito for válido, então o sistema regista a compra e subtrai as quantidades compradas às quantidades disponíveis dos produtos comprados na base de dados.

Este *use case* satisfaz o requisito **RF04**.

3.3.3 Use cases do vendedor

Criar produto

Um vendedor tem a possibilidade de adicionar produtos seus na base de dados da aplicação para os poder vender.

Quando o utilizador se dirige à pagina dos seus produtos e indica que pretende criar um produto, a aplicação solicita as seguintes informações acerca do produto a adicionar:

- Nome
- Descrição
- Categoria
- Preço unitário
- Quantidade inicialmente disponível
- Imagem ilustrativa do produto

Após o utilizador preencher todos os campos com as informações do produto, o sistema verifica se este produto não existe produto na conta do utilizador. Se assim for, então o produto é adicionado ao sistema. Caso contrário, a aplicação informa o utilizador que já tem este produto associado à sua conta.

É importante referir duas observações neste *use case*. A primeira é que o campo **Categoría**, no contexto da lógica de negócio, é a **Subcategoria**. Este aspecto é abordado com detalhe na secção 3.2. A segunda consideração é o **preço unitário**. A aplicação exige que os preços de um produto sejam referentes a uma unidade desse produto. Se uma unidade equivaler a um certo peso (em gramas, quilogramas, etc...), é da responsabilidade do vendedor mencionar esse detalhe na descrição do produto.

Este *use case* satisfaz o requisito **RF05**.

Configuração de um produto

Um vendedor tem a possibilidade de alterar certos aspectos dos seus produtos.

Quando o utilizador se dirige à pagina dos seus produtos, este pode selecionar um deles e alterar os seguintes campos:

- Descrição
- Preço
- Imagem ilustrativa
- Quantidade disponível

Após confirmar as alterações ao produto, estas são registadas na base de dados.

A configuração de um produto não exige muita lógica nem verificações por parte do sistema. Por este motivo, é normal que este *use case* seja relativamente pequeno.

Este *use case* satisfaz o requisito **RF06**.

Remoção de um produto

Um vendedor pode ter a necessidade de descontinuar a venda de algum dos seus produtos.

Quando o utilizador se dirige à pagina dos seus produtos, este seleciona o produto a remover e o mesmo é removido da base de dados do sistema.

Tal como o *use case* anterior, a remoção de produtos é uma operação simples e sem verificações por parte do sistema.

Este *use case* satisfaz o requisito **RF07**.

Solicitação de uma banca de uma feira

Quando um vendedor visita uma feira, o sistema apresenta as diversas bancas dessa feira. Se

o utilizador não tiver uma banca nessa feira, a aplicação disponibiliza a opção do utilizador adquirir uma banca para vender os seus produtos.

Quando o utilizador solicita uma banca numa feira, o sistema começa por verificar se o número de bancas ocupadas naquela feira ainda não foi atingido. Se assim for, a aplicação solicita ao vendedor o título para a sua banca e, a partir de agora, o utilizador tem uma banca na feira para expor os seus produtos.

Este *use case* satisfaz o requisito **RF16**.

Configuração de uma banca de uma feira

Quando um vendedor já possui uma banca numa feira e visita essa banca, este pode adicionar ou remover produtos da banca.

Se o utilizador pretender adicionar um produto, a aplicação apresenta ao utilizador todos os produtos associados a sua conta. Quando o utilizador selecionar um dos produtos, o sistema verifica se este pode ser adicionado à feira tendo em conta a sua categoria. Se assim for, o produto é adicionado à banca. Caso contrário, a aplicação informa o utilizador que o produto não pode ser adicionado.

Se o utilizador pretender remover um produto, o sistema dissocia esse produto da banca na base de dados.

Este *use case* satisfaz o requisito **RF08**.

3.3.4 Use cases do administrador

Aprovação da conta de um vendedor

Os administradores são responsáveis por controlar as contas de vendedores que podem ser adicionadas ao sistema.

Quando o administrador pretende ver quais as contas de vendedores que ainda não estão aprovadas, a aplicação apresenta essas mesmas contas. Seguidamente, o utilizador seleciona uma das contas e a aplicação apresenta as informações da conta selecionada e, após avaliar essas informações, o administrador decide se a conta é aprovada ou não.

Se for aprovada, a mesma fica pronta a utilizar pelo seu dono. Caso contrário, a conta é eliminada.

Este *use case* satisfaz o requisito **RF09**.

Criação de uma feira

Os administradores são responsáveis pela abertura de feiras na aplicação.

Quando o administrador pretende adicionar uma feira ao sistema, a aplicação solicita os seguintes dados:

- Nome da feira
- Categoria
- Número de limite de bancas
- Data de início
- Data de fim

Quando o utilizador confirma que quer adicionar a feira, o sistema verifica que não existem outras feiras com o mesmo nome. Se assim for, a nova feira é adicionada ao sistema. Caso contrário, a aplicação informa o utilizador que a feira não pode ser criada.

Este *use case* satisfaz o requisito **RF10**.

3.3.5 Diagramas de atividades

Os diagramas de atividades são uma ferramenta muito útil para representar a sequência de atividades que ocorrem num sistema. No caso do nosso projeto, utilizámos estes diagramas para perceber como é que os *use cases* se encaixam na utilização da aplicação e o conjunto de ações necessário para chegar até eles.

Neste relatório, vamos apresentar três diagramas que representam, cada um, o funcionamento da aplicação para os clientes, vendedores e administradores

Diagrama de atividades do cliente

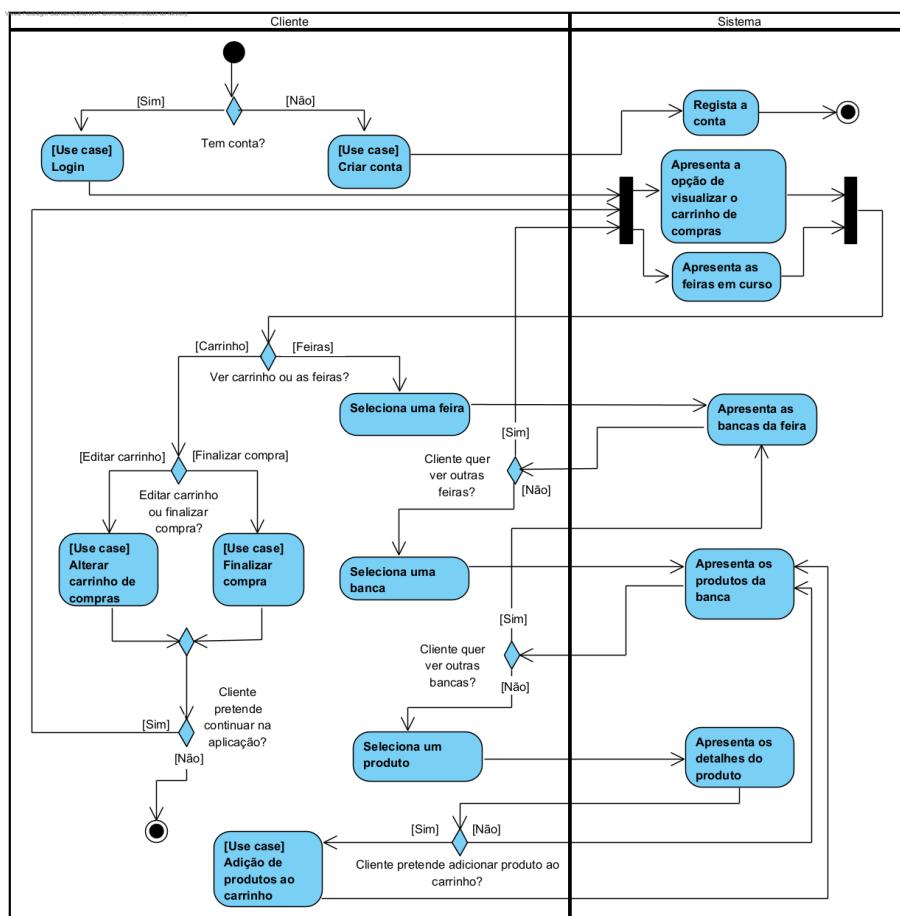


Figura 3.3: Diagrama de atividades do cliente

Este diagrama representa o funcionamento do programa para um utilizador cliente.

Inicialmente, se o utilizador não tiver uma conta, este deve visitar a página de criação de conta para se registrar na aplicação. Se o utilizador tiver conta, este é redirecionado para a página principal onde são apresentadas as feiras em curso bem como algumas opções como, por exemplo, visitar o carrinho de compras.

Quando o utilizador é redirecionado para a página principal, pode visitar todas as feiras em curso ou ver o seu carrinho de compras. Quando visita uma feira, o utilizador pode visitar todas as bancas e adquirir produtos nelas presentes. Quando visita o carrinho de compras, o utilizador pode alterá-lo, removendo ou alterando as quantidades de produtos presentes bem como fazer o *check-out* da compra para adquirir os produtos.

Diagrama de atividades do vendedor

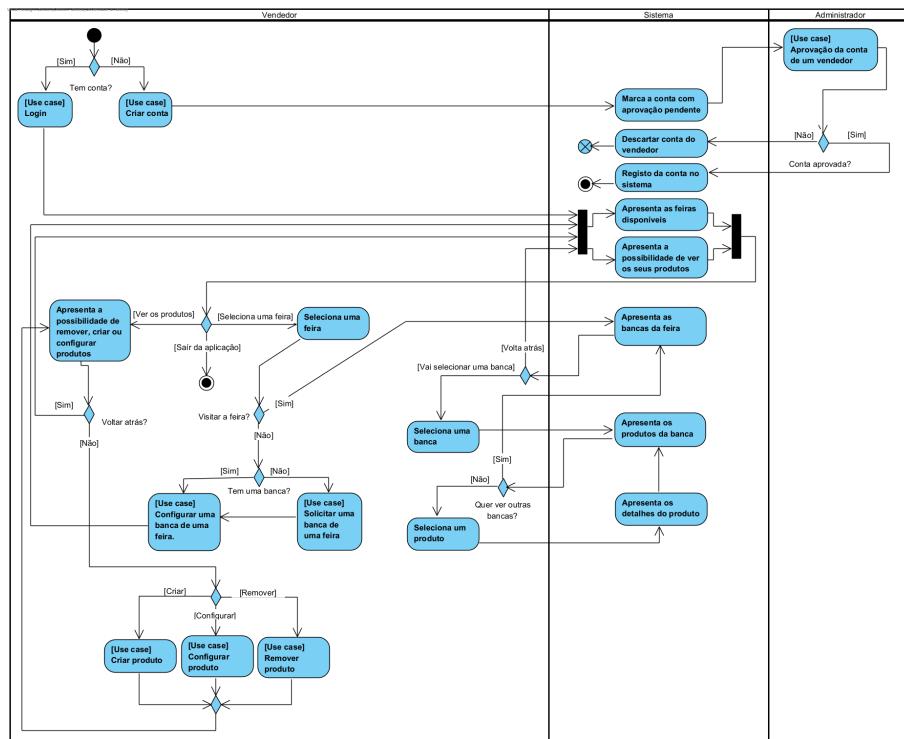


Figura 3.4: Diagrama de atividades do vendedor

Este diagrama representa o funcionamento do programa para um utilizador vendedor.

Inicialmente, se o utilizador não tiver uma conta, este deve visitar a página de criação de conta para se registar na aplicação e, posteriormente, deve aguardar pela aprovação da conta. Se o utilizador tiver conta, este é redirecionado para a página principal onde são apresentadas as feiras em curso bem como algumas opções como, por exemplo, ver quais são os produtos que detém.

Quando o utilizador é redirecionado para a página principal, pode visitar as todas as feiras em curso ou ver os seus produtos. Quando visita uma feira, o vendedor pode configurar a sua banca ou pode solicitar uma banca para vender produtos na feira que está a visitar. O utilizador pode também optar por apenas visitar a feira e ver que produtos estão a ser vendidos nas bancas.

Se o vendedor visitar a página dos seus produtos, a aplicação apresenta-lhe todos os produtos que detém bem como proporciona as funcionalidades de adicionar produtos ou remover e

alterar algum dos seus produtos.

Diagrama de atividades do administrador

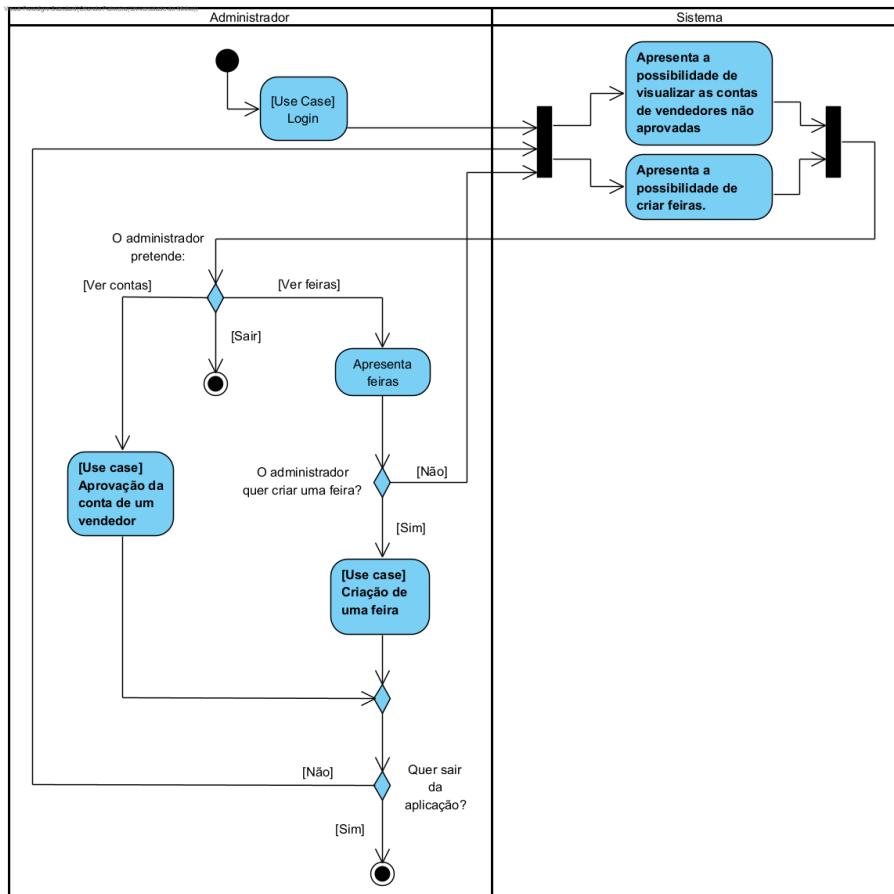


Figura 3.5: Diagrama de atividades do administrador

Este diagrama representa o funcionamento do programa para um utilizador administrador do sistema.

Quando o utilizador é redirecionado para a página principal, pode verificar quais são as contas de vendedores que foram criadas mas que ainda não foram aprovadas e, para além disso, também pode criar feiras.

Se o utilizador pretender aprovar uma conta de vendedor, este é direcionado para uma página onde lhe são apresentadas as contas de vendedores que ainda não foram aprovadas e, nessa página, o utilizador pode aprovar essas contas.

Se o utilizador pretender criar uma feira, este é direcionado para uma página onde lhe são apresentadas as feiras em curso bem como lhe é apresentada a possibilidade de criar uma feira.

4 Conceção do sistema de dados

A nossa aplicação vai possuir diversos dados acerca dos clientes, vendedores, feiras, transações comerciais, entre outros. Com o passar do tempo, é normal que o sistema comece a armazenar uma grande quantidade de informação e, por esse motivo, é extremamente importante implementar uma base de dados que organize todas essas informações de forma eficiente.

Além disso, uma base de dados é extremamente útil para gerar relatórios que possuam informações úteis. Por exemplo, podem ser gerados relatórios que informem qual a categoria de produtos que foi mais comprada. Esses relatórios podem ser fornecidos aos vendedores para estes ajustarem os seus produtos às exigências do mercado.

Finalmente, as bases de dados possuem implementações muito eficientes de armazenamento de dados, o que permite uma consulta rápida dos mesmos permitindo proporcionar aos utilizadores finais uma experiência de utilização mais agradável.

4.1 Apresentação geral da estrutura do sistema de dados

Após definirmos os modelos que irão sustentar a implementação do nosso sistema, foi necessário planificar uma base de dados que apoiará o seu funcionamento.

Após a análise dos requisitos recolhidos e com o auxílio dos modelos apresentados na especificação do sistema, foi possível reproduzir o modelo lógico apresentado a seguir:

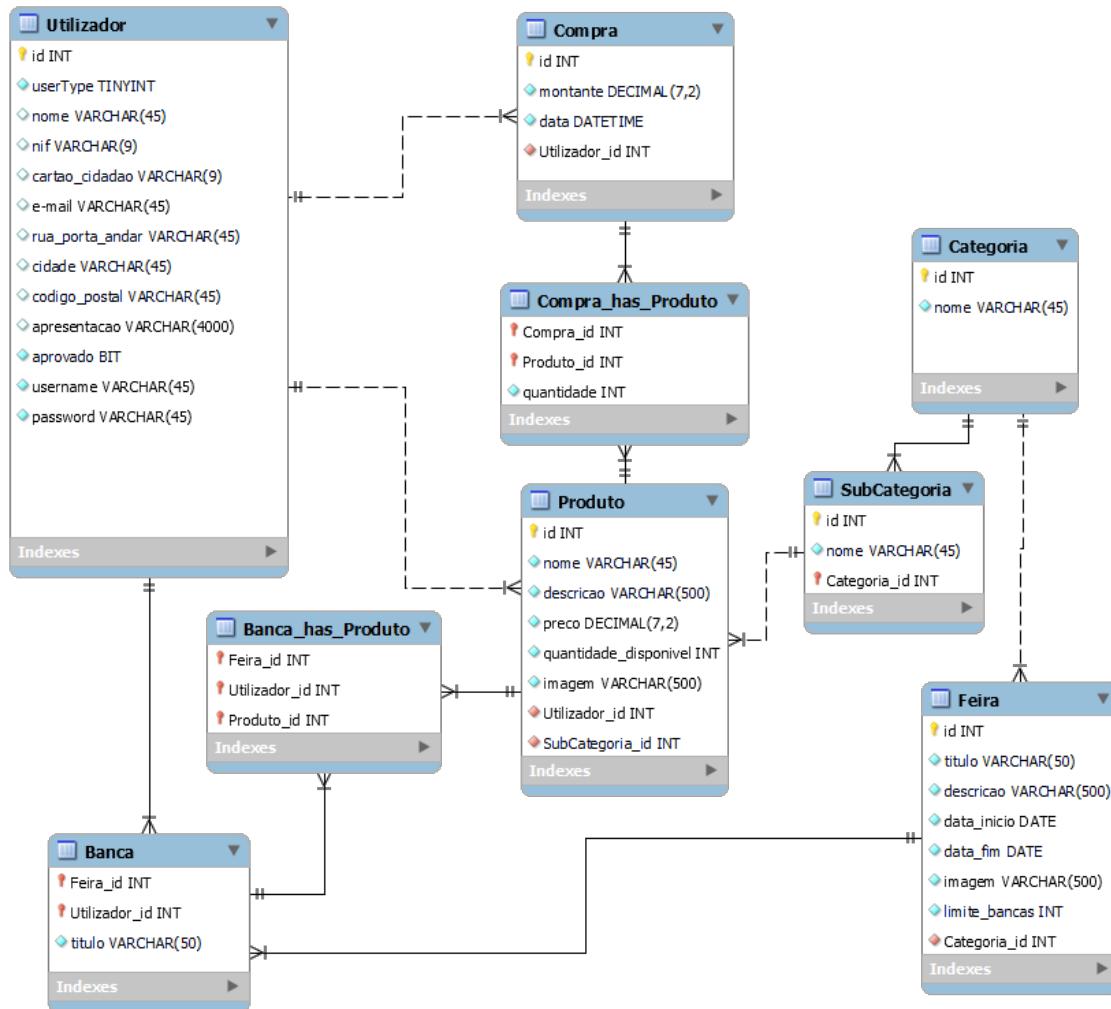


Figura 4.1: Modelo lógico da base de dados do sistema

4.2 Descrição detalhada dos vários elementos de dados e seus relacionamentos

Para uma melhor compreensão das entidades na base de dados, as suas relações e os seus atributos, elaborou-se o dicionário de dados em que explicamos a função de cada campo e damos exemplos de valores que esses campos podem tomar.

4.2.1 Utilizador

Um utilizador é identificado por um número inteiro incrementado automaticamente pela base de dados. Os restantes atributos são apenas dados simples que caracterizam o utilizador.

No atributo **userType** é possível ter um de três valores: 0 indica que é um administrador, 1 indica que é um comprador e 2 indica que é um vendedor.

Entidade	Atributo	Tipo de dados	Espaço Ocupado (bytes)	Descrição	Exemplo
Utilizador	id	Int	4	Id único de um utilizador	1
	userType	Tinyint	1	Tipo de utilizador (cliente, vendedor, admin)	cliente
	nome	VARCHAR(45)	47	Nome do utilizador	José Agostinho
	nif	VARCHAR(9)	11	NIF do utilizador	222444333
	cartao_cidadao	VARCHAR(9)	11	Cartão de cidadão do utilizador	111333222
	e-mail	VARCHAR(45)	47	E-mail do utilizador	zezinho_feirante@gmail.com
	rua_porta_andar	VARCHAR(45)	47	Morada do utilizador	Largo do Paço
	cidade	VARCHAR(45)	47	Cidade do utilizador	Braga
	codigo_postal	VARCHAR(45)	47	Código postal do utilizador	4704-536
	apresentacao	VARCHAR(4000)	4002	Texto apresentativo de um vendedor para aprovação	Pretendo vender produtos alimentícios que, geralmente, possuem uma grande procura.
	aprovado	Bit	1	Bit que indica se a conta está aprovada ou não	1
	username	VARCHAR(45)	47	Username usado no login	zezinho123
	password	VARCHAR(45)	47	Password do utilizador	321zezinho123

Figura 4.2: Dicionário de dados - Utilizadores

4.2.2 Feira

Uma feira é identificada por um número inteiro incrementado automaticamente pela base de dados. Os restantes atributos são apenas dados simples que caracterizam a feira.

O atributo **Categoria_id** representa a relação entre as entidades feira e categoria. Esta relação serve para indicar qual é a categoria da feira e é útil para filtrar quais os produtos que podem ser adicionados à feira.

Entidade	Atributo	Tipo de dados	Espaço Ocupado (bytes)	Descrição	Exemplo
Feira	id	Int	4	Id único de uma feira	3
	titulo	Varchar(50)	52	Título de uma feira	Feira de Natal
	descricao	Varchar(500)	502	Descrição de uma feira	Feira com o tema Natalício onde são vendidos todo o tipo de produtos típicos da época natalícia.
	data_inicio	Date	3	Data de início de uma feira	2022/11/28
	data_fim	Date	3	Data de término de uma feira	2023/01/06
	imagem	Varchar(500)	502	Path de imagem descriptiva de uma feira	./media/feiras/feira_do_livro_2022.png
	limite_bancas	Int	4	Número limite de bancas de uma feira	50
	Categoria_id	Int	4	Id da categoria associada à feira	5

Figura 4.3: Dicionário de dados - Feiras

4.2.3 Categoria

Uma categoria é identificada por um número inteiro incrementado automaticamente pela base de dados. O campo **nome** é apenas o nome da categoria.

Esta entidade é completamente independente de todas as outras e é por isso que não possui, por exemplo, chaves estrangeiras.

Entidade	Atributo	Tipo de dados	Espaço Ocupado (bytes)	Descrição	Exemplo
Categoria	id	Int	4	Id único de uma categoria	12
	nome	Varchar(45)	47	Nome de uma categoria	Alimentos

Figura 4.4: Dicionário de dados - Categorias

4.2.4 Subcategoria

Uma subcategoria é identificada por um número inteiro incrementado automaticamente pela base de dados. O campo **nome** é apenas o nome da subcategoria.

O campo **Categoria_id** serve para identificar a categoria à qual a subcategoria pertence. Com esta relação, conseguimos saber, a partir da categoria, quais são as suas subcategorias.

Entidade	Atributo	Tipo de dados	Espaço Ocupado (bytes)	Descrição	Exemplo
SubCategoria	id	Int	4	Id único de uma subcategoria	4
	nome	Varchar(45)	47	Nome de uma subcategoria	Fruta
	Categoria_id	Int	4	Id da categoria associada a uma subcategoria	12

Figura 4.5: Dicionário de dados - Subcategorias

4.2.5 Compra

Uma compra é identificada por um número inteiro incrementado automaticamente pela base de dados. Os atributos **montante** e **data** indicam, respetivamente, o preço pago pelo utilizador e a data e hora do momento da realização da compra.

O atributo **Utilizador_id** identifica o utilizador que efetuou a compra.

Entidade	Atributo	Tipo de dados	Espaço Ocupado (bytes)	Descrição	Exemplo
Compra	id	Int	4	Id único de uma compra	24
	montante	Decimal(7,2)	5	Custo total da compra	6,98
	data	Datetime	8	Altura em que foi realizada a compra	2022-11-26 18:44:32
	Utilizador_id	Int	4	Id de utilizador que efetuou a compra	16

Figura 4.6: Dicionário de dados - Compras

4.2.6 Produto

Um produto é identificado por um número inteiro incrementado automaticamente pela base de dados. Os restantes atributos são apenas dados simples para caracterizar o produto.

O atributo **Utilizador_id** é uma chave estrangeira que irá tomar o valor da chave primária de um utilizador. Esta relação entre o utilizador e o produto serve para indicar quem é o dono do produto.

O atributo **SubCategoria_id** é uma chave estrangeira que identifica a subcategoria à qual este produto pertence.

Entidade	Atributo	Tipo de dados	Espaço Ocupado (bytes)	Descrição	Exemplo
Produto	id	Int	4	Id único de um produto	6
	nome	Varchar(45)	47	Nome de um produto	Bananas
	descricao	Varchar(500)	502	Descrição de um produto	Bananas colhidas na Madeira ao pôr do sol.
	preco	Decimal(7,2)	5	Preço de um produto	3.99
	quantidade_disponivel	Int	4	Quantidade disponível de um produto	10
	imagem	Varchar(500)	502	Path da imagem de um produto	./media/products/liaao.png
	Utilizador_id	Int	4	Id do vendedor a quem pertence o produto	19
	SubCategoria_id	Int	4	Id da subcategoria associada ao produto	4

Figura 4.7: Dicionário de dados - Produtos

4.2.7 Banca

Uma banca é identificada pelas chaves primárias da feira e do utilizador. O campo **título** é o nome da banca atribuído pelo vendedor que a adquiriu.

O atributo **Feira_id** identifica a feira à qual a banca pertence e o atributo **Utilizador_id** identifica o utilizador que é dono da banca.

Ao definirmos a chave primária da banca como sendo as chaves da feira e do utilizador, estamos a impedir que um utilizador tenha mais que uma banca numa feira que é uma das condições do requisito **RF16**.

Entidade	Atributo	Tipo de dados	Espaço Ocupado (bytes)	Descrição	Exemplo
Banca	Feira_id	Int	4	Id da feira em que se encontra a banca	23
	Utilizador_id	Int	4	Id do utilizador a quem pertence a banca	12
	título	Varchar(50)	52	Titulo da banca	Banca Gourmet

Figura 4.8: Dicionário de dados - Bancas

4.2.8 Relação M-N *Banca_has_Produto*

A relação *Banca_has_Produto* serve, essencialmente, para saber quais são os produtos contidos numa banca.

A chave primária desta tabela é formada pela chave da banca (id do utilizador e da feira) e pelo id do produto. Deste modo, podemos impedir que um produto seja adicionado mais do que uma vez na mesma banca, evitando repetições.

Entidade	Atributo	Tipo de dados	Espaço Ocupado (bytes)	Descrição	Exemplo
Banca_has_Produto	Feira_id	Int	4	Id da feira em que se encontra a banca	12
	Utilizador_id	Int	4	Id do utilizador a quem pertence a banca	14
	Produto_id	Int	4	Id do produto que está na banca	51

Figura 4.9: Dicionário de dados - relação *Banca_has_Produto*

4.2.9 Relação M-N *Compra_has_Produto*

A relação *Compra_has_Produto* serve, essencialmente, para saber quais são os produtos presentes numa compra.

A chave primária desta tabela é constituída pelo id da compra e do produto para evitar associar o mesmo produto à mesma compra mais do que uma vez.

Entidade	Atributo	Tipo de dados	Espaço Ocupado (bytes)	Descrição	Exemplo
Compra_has_Produto	Compra_id	Int	4	Id de uma determinada compra	15
	Produto_id	Int	4	Id do produto que foi adquirido na compra	35
	quantidade	Int	4	Quantidade do produto comprada	4

Figura 4.10: Dicionário de dados - relação *Compra_has_Produto*

5 Esboço dos interfaces do sistema

As interfaces gráficas devem ajudar os diversos utilizadores de uma aplicação a ter acesso às funções ou informações pretendidas, de uma forma clara e intuitiva.

Por esta razão foi decidido criar esboços das interfaces do programa, para poderem ser analisadas pela equipa e pelos clientes, antes da implementação do sistema, e assim criar uma versão mais adequada ao contexto.

5.1 Estrutura geral das interfaces do sistema

De modo a perceber o modo de transição de umas interfaces para as outras, elaborou-se o seguinte esquema:

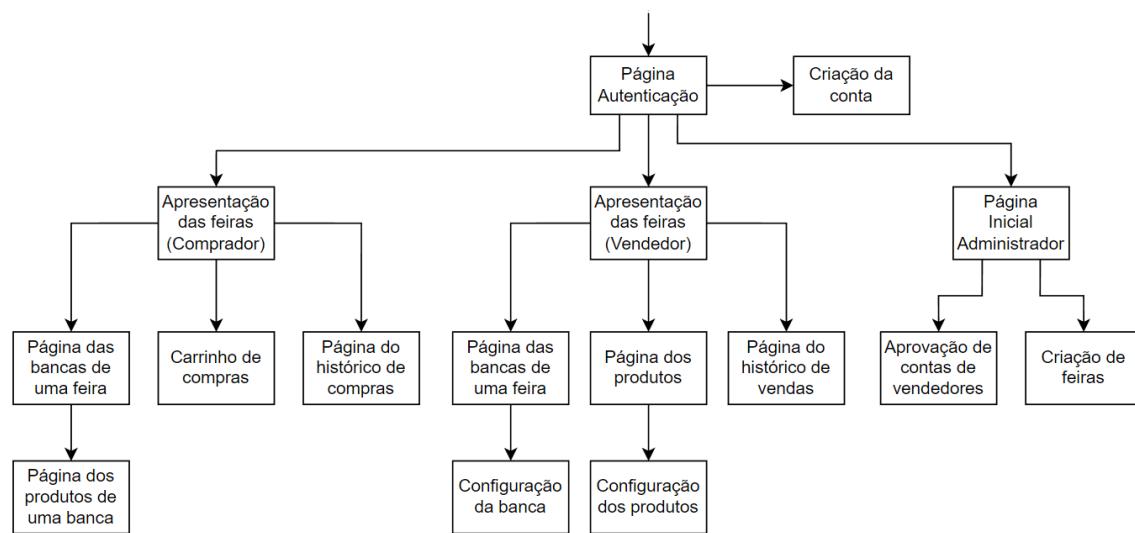


Figura 5.1: Esquema das interfaces do sistema

5.2 Caracterização das interfaces

Apresentam-se de seguida alguns *mockups* das interfaces que consideramos que representam as funcionalidades principais da aplicação. Existem alguns mockups que são muito semelhantes ao que é feito em quase todas as aplicações de áreas de atuação, similares a este projeto e, por esse motivo, escolhemos colocá-los nos anexos.

5.2.1 Página de apresentação de feiras (comprador e vendedor)

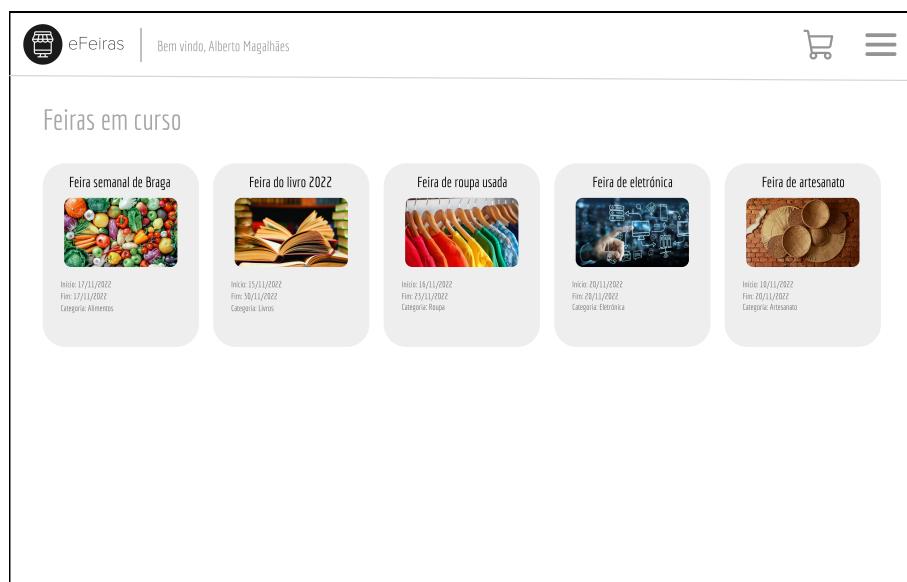


Figura 5.2: Página de apresentação de feiras

Neste mockup, que representa a visualização da página inicial (visualização de feiras em curso) após a autenticação de um comprador ou um vendedor, o foco foi criar um aspeto simples e informativo, mas também apelativo.

Para esse efeito achamos um boa solução, apresentar as feiras que estão a decorrer numa grelha, com as informações essenciais e com imagens descriptivas dos âmbitos das feiras. Os elementos que aglomeram as informações de uma feira também servem de botão para navegar para a página de apresentação das bancas disponíveis na feira selecionada.

É importante referir um elemento que é comum a todas as interfaces, o cabeçalho. Este cabeçalho foi pensado para albergar o logótipo da aplicação, para estar presente durante toda a utilização do programa, uma mensagem para identificar o utilizador autenticado e botões para a navegação rápida para funções consideradas importantes (dependentes do tipo de utilizador).

5.2.2 Páginas de apresentação das bancas de uma feira (comprador e vendedor)

The screenshot shows a user interface for a food market. At the top, there's a header with the logo 'eFeiras' and a welcome message 'Bem vindo, Alberto Magalhães'. On the right side of the header are icons for a shopping cart and a menu. Below the header, there's a banner for 'Feira semanal de Braga' with dates 'Início: 17/11/2022' and 'Fim: 17/11/2022', categorized under 'Alimentos', and a small image of various vegetables. The main content area displays three food stall entries in rounded rectangular boxes:

- A Banca do Sr. Alberto** - Proprietário: Alberto Magalhães
- Frutas e Companhia** - Proprietário: Cristina Pais
- Banca Gourmet** - Proprietário: Ermelinda Silva

Figura 5.3: Página de apresentação das bancas de uma feira (comprador)

This screenshot shows the same market listing as Figure 5.3, but from the perspective of a vendor ('vendedor'). In addition to the stall details and owner information, there are two buttons on the right side of the screen: 'Solicitar banca' (Request stall) and 'Voltar' (Back).

Figura 5.4: Página de apresentação das bancas de uma feira (vendedor)

Para esta página, que serve para apresentar as bancas disponíveis de uma feira, e também, no caso de um vendedor, solicitar uma banca nessa mesma feira, foi decidido apresentar as mesmas de uma forma muito simples, sob um formato de lista, com apenas o nome da banca e do seu proprietário. Desta maneira mantemos a informação fácil de captar.

É importante referir que, apesar de se pretender uma interface mais limpa e com poucos elementos, é necessário um elemento identificativo da feira que o utilizador está a explorar, e para ser mais facilmente identificável, foi dada a liberdade de se usar a imagem associada à mesma feira.

Os itens da lista também servem como botões para navegar para a página de exploração dos produtos da banca em questão.

Esta esta página é muito similar tanto para compradores como para vendedores. No entanto, existe uma pequena diferença, que é a inclusão de um botão para um vendedor solicitar uma banca na feira que está a explorar. Caso o vendedor já tenha banca na feira, este botão será substituído por um para configurar a sua banca.

5.2.3 Página dos produtos de uma banca (comprador)

Item	Preço(€/un)	Quantidade
Banana importada	1,99 €	<input type="button" value="-"/> 3 <input type="button" value="+"/>
Pimento vermelho	3,49 €	<input type="button" value="-"/> 2 <input type="button" value="+"/>
Batata vermelha	2,99 €	<input type="button" value="-"/> 0 <input type="button" value="+"/>
Manga	6,29 €	<input type="button" value="-"/> 0 <input type="button" value="+"/>

Voltar Adicionar ao carrinho

Figura 5.5: Página dos produtos de uma banca (comprador)

Na página de visita de uma banca é necessário apresentar uma maior quantidade de informação, uma vez que os produtos são entidades mais detalhadas. Por esse motivo, é necessário que esta página apresente apenas informações relevantes de cada produto (tais como, a imagem, o nome e o preço). Para aceder a informações complementares existe uma hiperligação para aceder, por exemplo, à descrição do produto.

Para além disso, seguindo o estilo apresentado anteriormente, na parte superior da página manteve-se o elemento identificativo da feira, que agora também contém o nome da banca que se está a visitar.

Como a finalidade desta página é explorar uma banca, esta também necessita de poder dar

acesso ao comprador à funcionalidade de adicionar produtos ao seu carrinho de compras e, para isso, também foi necessário criar elementos para esse efeito. Criaram-se botões para poder aumentar ou diminuir a quantidade pretendida dos produtos, e um botão para adicionar as quantidades selecionadas ao carrinho.

5.2.4 Configuração de uma banca (vendedor)

The screenshot shows a web interface for managing a stall. At the top, there's a header with the logo 'eFeiras', a welcome message 'Bem vindo, Alberto Magalhães', and a shopping cart icon. Below the header, the title 'Feira semanal de Braga' and 'Categoria: Alimentos' is displayed, along with a small image of various vegetables. The main content area is titled 'A Banca do Sr. Alberto'. It lists four items with their prices:

Item	Preço(€/un)
Banana importada	1,99 €
Pimento vermelho	3,49 €
Batata vermelha	2,99 €
Manga	6,29 €

Each item row includes a 'Ver detalhes' link and a delete button (a small 'X'). To the right of the table, there's a button labeled 'Adicionar um produto'. At the bottom left, there's a 'Voltar' button.

Figura 5.6: Configuração de uma banca (vendedor)

A página para para um vendedor configurar a sua banca assemelha-se, em termos estéticos, à página dos produtos de uma banca. A grande diferença é que aqui o vendedor pode alterar os produtos expostos, removendo algum dos que já se encontram na banca ou adicionando outros que tenha associados à sua conta.

5.2.5 Carrinho de compras



The screenshot shows a shopping cart interface for 'eFeiras'. At the top, there's a logo, a welcome message 'Bem vindo, Alberto Magalhães', and icons for a shopping cart and a menu. The main title is 'O seu carrinho (2 itens)'. Below this, there's a table with the following data:

Item	Preço(€/un)	Quantidade	Total
Banana importada	1,99 €	(<input type="button" value="1"/> <input type="button" value="2"/> <input type="button" value="3"/> <input type="button" value="4"/>)	5,97 € <input type="button" value="X"/>
Pimento vermelho	3,49 €	(<input type="button" value="1"/> <input type="button" value="2"/> <input type="button" value="3"/> <input type="button" value="4"/>)	6,98 € <input type="button" value="X"/>
Total:			12,95 €

A 'Check-out' button is located at the bottom left.

Figura 5.7: Carrinho de compras

No carrinho de compras houve um foco na clareza da demonstração dos produtos que se encontram no carrinho, as sua quantidades, o seu preço unitário e o seu preço total, e também o preço total final do carrinho. Além disso, há uma funcionalidade para remover itens clicando num botão em forma de caixote de lixo.

Também se optou, por uma questão de facilidade e conforto para o comprador, dar a opção de mudar a quantidade dos produtos no carrinho.

6 Implementação da Aplicação

6.1 Apresentação e descrição do processo de implementação realizado

No desenvolvimento desta aplicação, optou-se por começar com a implementação da base de dados e a respetiva API (IDatabaseFacade). Isso permitiu garantir que, antes da construção da aplicação, os dados pudessem ser corretamente armazenados e que a comunicação entre a base de dados e o resto da aplicação funcionasse corretamente.

Em seguida, desenvolveu-se a lógica de negócio e a respetiva API (IEfeirasFacade). Isso incluiu a implementação dos métodos da lógica de negócio, validações e processamento de dados para garantir que a aplicação esteja a funcionar corretamente. Toda a lógica de negócio foi implementada em paralelo com o desenvolvimento da interface gráfica para se ir percebendo quais as necessidades que a interface gráfica ia exigindo da lógica de negócio.

No final, conseguimos verificar que a base de dados, a lógica de negócio e a interface gráfica estavam a trabalhar em conjunto e a comunicar corretamente.

A estrutura final das três camadas e o modo como comunicam entre si podem ser traduzidos no seguinte esquema:

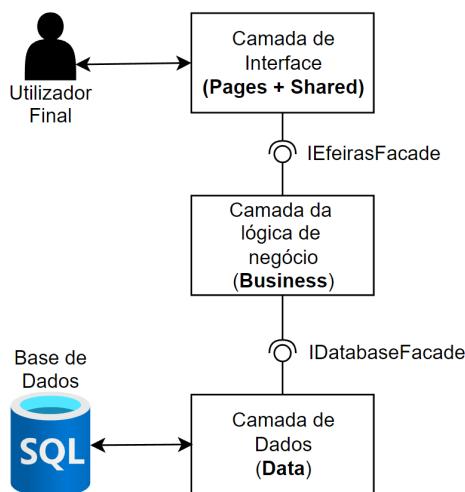


Figura 6.1: Estrutura e comunicação das camadas da aplicação

No esquema da figura 6.1 pretendemos demonstrar com quais camadas o utilizador e a base de dados interagem, bem como as interfaces utilizadas para as diferentes camadas comunicarem entre si. No nosso caso, existem duas API's que servem para comunicar com a lógica de negócio e com a base de dados (IEFeirasFacade e IDatabaseFacade).

Finalmente, é importante referir que no esquema apresentado são fornecidos os nomes dos *packages* que contém o código fonte das respetivas camadas.

6.2 Apresentação dos serviços implementados e estrutura final da aplicação

Durante a fase de implementação da aplicação **eFeiras**, a equipa de desenvolvimento esforçou-se para que, no final, se conseguisse obter um *software* semelhante ao que foi planeado nas fases anteriores.

Vamos apresentar de seguida todas os serviços implementadas.

6.2.1 Serviços implementados

Autenticação

Quando o utilizador inicia a aplicação, é abordado com a página de *login*. Nessa página o utilizador pode entrar na sua conta ou pode optar por ser redirecionado para a página da criação de conta caso não tenha uma.

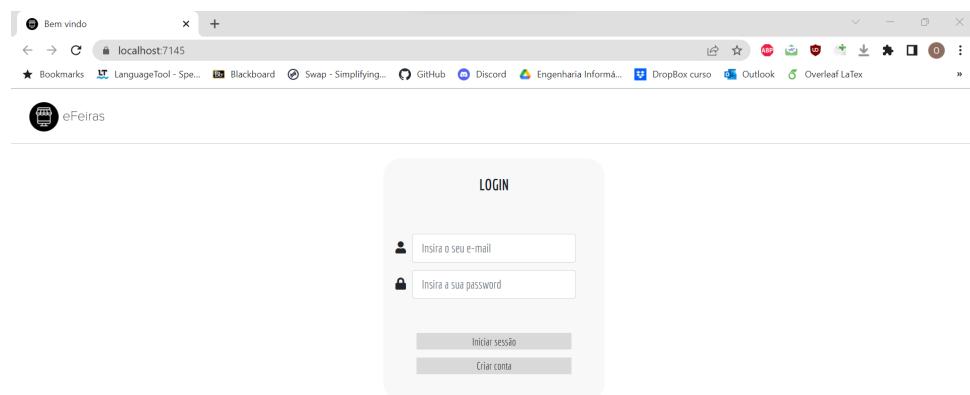


Figura 6.2: Página de *login*

Caso o utilizador queira criar uma conta, este é encaminhado para a seguinte página:

The screenshot shows a web browser window with the URL `localhost:7145/createaccount`. The page title is "eFeiras". The main content is a form titled "Crie a sua conta". It contains several input fields: "Nome", "E-mail", "Nome de utilizador", "Password", "Número do cartão de cidadão", "Número de identificação fiscal", "Rua, n.º de porta, andar", "Cidade", and "Código postal". Below these fields are two radio buttons: "Comprador" (unchecked) and "Vendedor" (checked). At the bottom of the form is a "Criar conta" button. To the right of the form, there is a note in a box: "Caso pretenda ser um vendedor, apresente-se. Diga quais são os seus objectivos e que tipos de produto pretende vender. Se deixar este espaço em branco, a sua conta não será aprovada."

Figura 6.3: Página de criação da conta

Nota: Nesta página, o campo que aparece do lado direito, apenas está disponível para a criação de contas de vendedores. Este campo serve para um vendedor se apresentar para posterior aprovação por um administrador.

Página principal

Após iniciar sessão, o utilizador é redirecionado para a página principal. Nesta página, são apresentadas todas as feiras em curso bem como é apresentada a possibilidade de visitar o carrinho de compras.

The screenshot shows a web browser window with the URL `localhost:7145/CompPage`. The page title is "eFeiras". The top navigation bar shows the user "Bem-vindo(a), Orlando Palmeira" and includes icons for home, shopping cart, and exit. Below the header, there is a section titled "Feiras em curso" (Upcoming Fairs) which lists five events:

- Feira semanal de Braga (Category: Alimentos)
- Feira do livro 2022 (Category: Livros)
- Feira de roupa usada (Category: Roupa)
- Feira de eletrónica (Category: Eletronica)
- Feira de artesanato (Category: Artesanato)

Each event listing includes a small thumbnail image, the fair name, start date, end date, and category.

Figura 6.4: Página principal

Visita a uma feira (comprador)

Quando um comprador entra numa feira, a aplicação mostra todas as bancas que estão abertas. Nesta página são apresentadas algumas informações sobre a feira visitada e, para cada banca, é apresentado o seu nome e o nome do seu proprietário.

The screenshot shows a web browser window with the URL `localhost:7145/VisitaFeiraComp`. The title bar says "eFeiras". The main content area displays information about a "Feira semanal de Braga" from 14-12-2022 to 20-01-2023. It shows three open stalls:

- A Banca do Sr.Alberto** (Owner: Alberto Magalhães) - Category: Alimentos, with a small image of various vegetables.
- Frutas e Companhia** (Owner: Cristina Pais) - Category: Alimentos.
- Banca Gourmet** (Owner: Ermelinda Silva) - Category: Alimentos.

At the bottom left is a "Voltar" button, and at the bottom right is an "Adicionar ao carrinho" button.

Figura 6.5: Página de visita de uma feira (comprador)

Visita a uma banca (comprador)

Quando um comprador entra numa banca, a aplicação apresenta todos os produtos expostos e as suas informações como, por exemplo, nome e preço unitário. Também são apresentados os botões de incrementar e decrementar a quantidade de produtos que o utilizador quer colocar no carrinho de compras.

The screenshot shows a web browser window with the URL `localhost:7145/VisitaBancaComp`. The title bar says "eFeiras". The main content area displays information about the "Frutas e Companhia" stall at a "Feira semanal de Braga". It shows five fruit products:

Item	Preço(€/un)	Quantidade
Maçã verde	1,5 €	- 0 +
Pêra rocha	2 €	- 0 +
Kiwi	1,75 €	- 0 +
Manga	2,25 €	- 0 +
Ameixa	1,5 €	- 0 +

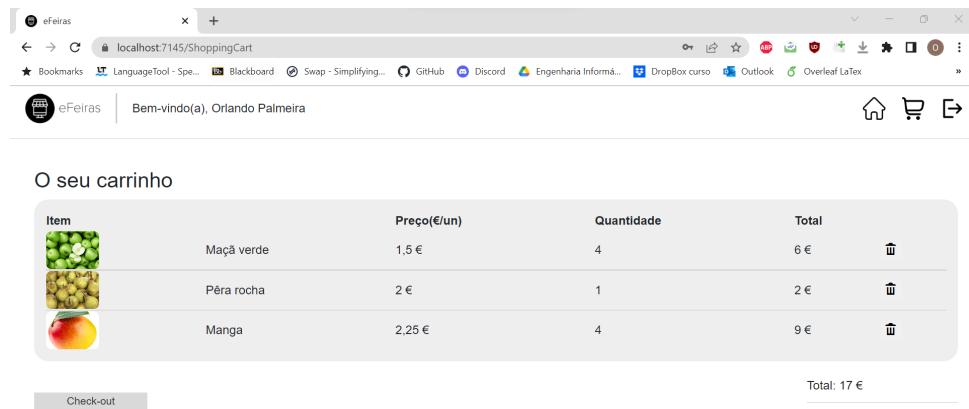
Each product row includes a "Ver detalhes" link. At the bottom left is a "Voltar" button, and at the bottom right is an "Adicionar ao carrinho" button.

Figura 6.6: Página de visita de uma banca (comprador)

Carrinho de compras

Na página do carrinho de compras, o comprador consegue ver quais os produtos que selecionou ao longo das visitas que fez nas feiras, a quantidade de cada produto e o montante total que tem de pagar.

Nesta página o comprador pode finalizar a compra ou eliminar algum produto caso já não o queira comprar.



O seu carrinho

Item	Preço(€/un)	Quantidade	Total
 Maçã verde	1.5 €	4	6 €
 Pêra rocha	2 €	1	2 €
 Manga	2.25 €	4	9 €

Total: 17 €

[Check-out](#)

Figura 6.7: Página de visita do carrinho de compras

Visita a uma feira (vendedor)

Na página da feira, a aplicação apresenta ao vendedor algumas informações sobre a feira bem como as bancas que estão abertas. O vendedor pode solicitar uma banca ou configurar a que já tem.

The screenshot shows a web browser window for the 'Feira semanal de Braga' website. The URL is 'localhost:7145/VisitaFeiraVend'. The page title is 'eFeiras | Bem-vindo(a), Cristina Pais'. It displays information about the fair: 'Feira semanal de Braga' (Start: 14-12-2022, End: 20-01-2023), 'Categoria: Alimentos', and a small image of various fruits and vegetables. Below this, three open stalls are listed: 'A Banca do Sr.Alberto' (Owner: Alberto Magalhães), 'Frutas e Companhia' (Owner: Cristina Pais), and 'Banca Gourmet' (Owner: Ermelinda Silva). A 'Configurar banca' button is visible on the right. At the bottom left is a 'Voltar' button.

Figura 6.8: Página de visita de uma feira (vendedor)

Visita a uma banca (Vendedor)

Na página de visita de uma banca, o vendedor consegue ver algumas informações sobre os produtos expostos. Como o vendedor não faz compras, a finalidade desta página é apenas para ajudar um vendedor a ver o que os seus concorrentes estão a vender.

The screenshot shows a web browser window for the 'Feira semanal de Braga' website. The URL is 'localhost:7145/VisitaBancaVend'. The page title is 'eFeiras | Bem-vindo(a), Cristina Pais'. It displays information about the fair: 'Feira semanal de Braga' (Category: Alimentos) and a small image of various fruits and vegetables. Below this, a table lists products from the 'Banca Gourmet':

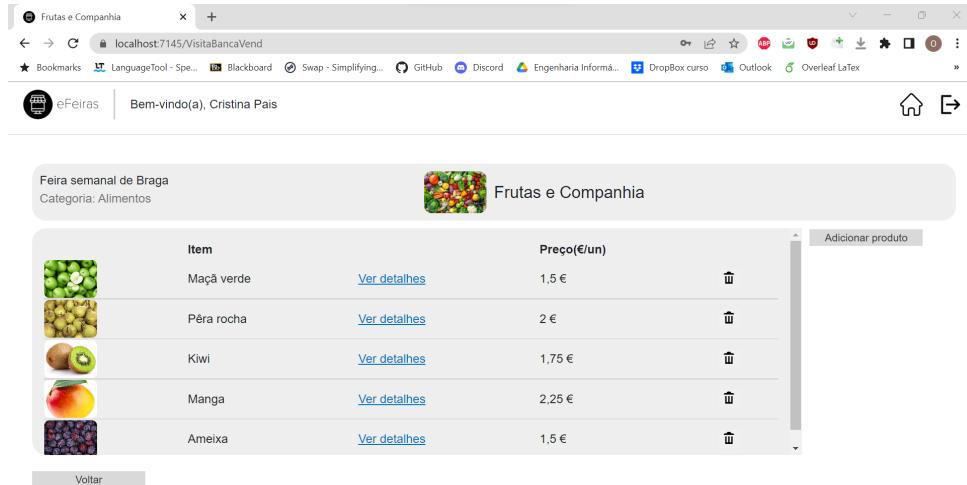
Item	Preço(€/un)
Amendoins	3,5 €
Pistácios	4 €
Nozes	3,75 €
Avelãs	4,25 €
Castanhas	3,5 €

Each product row includes a small image of the item and a 'Ver detalhes' link. A 'Voltar' button is at the bottom left.

Figura 6.9: Página de visita de uma feira (vendedor)

Configuração da banca

A página da configuração da banca serve para o vendedor adicionar e remover produtos da sua banca.



The screenshot shows a web browser window with the title 'Frutas e Companhia'. The URL is 'localhost:7145/VisitaBancaVend'. The page header includes 'eFeiras' and 'Bem-vindo(a), Cristina Pais'. The main content area is titled 'Feira semanal de Braga' and 'Categoria: Alimentos'. It features a logo of various fruits. A table lists five fruit items:

Item	Preço(€/un)
Maçã verde	1,5 €
Pêra rocha	2 €
Kiwi	1,75 €
Manga	2,25 €
Ameixa	1,5 €

Buttons for 'Adicionar produto' and 'Voltar' are visible at the bottom right and bottom left respectively.

Figura 6.10: Página de configuração de uma banca

6.2.2 Estrutura final da aplicação

No desenvolvimento da aplicação, foi seguido o modelo de três camadas (como foi referido na figura 6.1), composto pela interface de utilizador, lógica de negócio e acesso a dados.

Para garantir uma comunicação simplificada entre as camadas da lógica de negócio e de acesso aos dados, foi criada uma série de classes (DAO's) para permitir acesso às diversas entidades do sistema armazenadas na base de dados. Essas classes foram encapsuladas na interface IDatabaseFacade, que abstrai a implementação das mesmas e facilita a comunicação com a base de dados.

Na camada de lógica de negócio, foram implementadas todas as classes que representam as entidades concretas da aplicação, tais como bancas, produtos, feiras e utilizadores. Adicionalmente, foi implementada uma interface IFeirasFacade, que implementa todas as operações necessárias à camada de interface do utilizador. Estas operações incluem validações, consultas, inserções e remoções de dados.

A estrutura final da aplicação pode ser resumida no seguinte esquema:

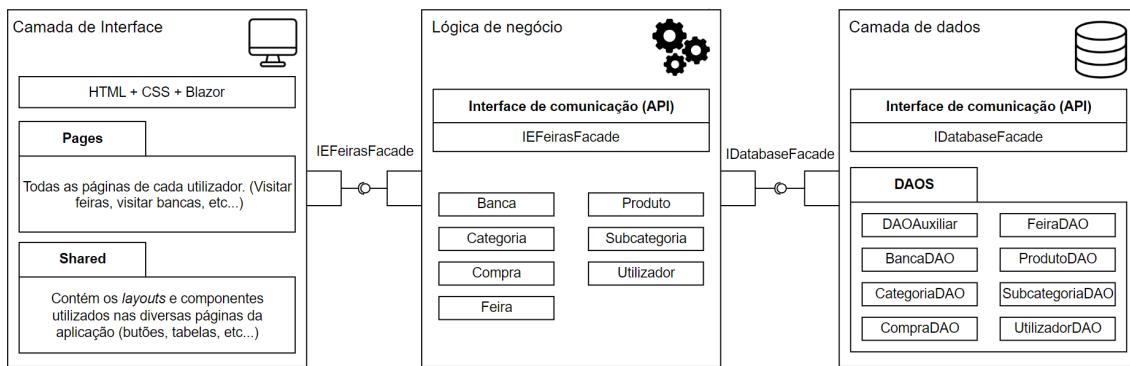


Figura 6.11: Estrutura final da aplicação

6.3 Análise e avaliação da aplicação desenvolvida

Tendo em conta os requisitos funcionais e os *use cases* estabelecidos na especificação, a aplicação deveria proporcionar aos utilizadores uma série de funcionalidades relevantes, a saber:

- Comprador
 - Navegação pelas diversas feiras e respetivos produtos.
 - Adição de produtos no carrinho de compras.
 - Alterar o seu carrinho de compras.
 - Realização de compras.
 - Visualização do histórico de compras.
- Vendedor
 - Navegação pelas diversas feiras e respetivos produtos.
 - Criação de produtos.
 - Configuração de produtos.
 - Remover/descontinuar produtos.
 - Configurar o catálogo das suas bancas.
 - Adquirir bancas.
 - Visualização do histórico de vendas.
- Administrador

- Criação de feiras.
- Aprovação das contas de vendedor.

No caso do comprador, a navegação nas feiras, adição de produtos ao carrinho de compras e a realização de compras encontra-se em pleno funcionamento. No entanto, a alteração do carrinho de compras não está completamente funcional uma vez que apenas permite eliminar os produtos do carrinho e não alterar sua quantidade. Quanto ao histórico de compras, esta funcionalidade não foi implementada.

Infelizmente, o vendedor está bastante incompleto. Este tipo de utilizador já tem a capacidade de navegar pelas feiras. No entanto, ainda não consegue criar, configurar e remover produtos, adquirir bancas e visualizar o histórico de vendas. Quando à configuração da banca, o vendedor só consegue eliminar os produtos da sua banca.

O administrador é um tipo de utilizador que não é muito relevante, tendo em conta o tipo da aplicação desenvolvida. Assim sendo, a implementação das suas funcionalidades não foi priorizada, o que implicou que este tipo de utilizador não está operacional.

Quanto aos requisitos não funcionais, conseguimos cumprir a maioria, mas infelizmente não cumprimos dois na versão atual. É algo que esperávamos conseguir atingir com mais experiência e tempo. Esses requisitos são o RF03 e o RF05.

RF03 - Dados armazenados em conformidade com o RGPD.

Dada a natureza do RGPD, e a falta de experiência do grupo de trabalho com cumprimentos de normas legislativas, o tempo que tivemos foi demasiado curto para fazer o estudo necessário para conseguir fazer cumprir de maneira satisfatória este requisito. Com isto, este requisito não foi cumprido em prol da execução do resto do projeto.

RF05 - Uso do método PBKDF2 para a proteção das passwords.

Devido à falta de tempo para estudar mais a fundo a proteção de *passwords* armazenadas numa base de dados, não nos foi possível usar nenhum método para cumprir este requisito. Este era um requisito que esperávamos conseguir cumprir, e gostávamos desta oportunidade para aprender um pouco mais sobre segurança, mas infelizmente, não conseguimos implementar.

6.4 Ferramentas Utilizadas

Durante a introdução da aplicação destacamos os recursos humanos e materiais que iriam ser utilizados para a realização do projeto. Os recursos materiais de softwares que consideramos necessários foram:

- Um sistema de gestão de bases de dados
- Um ambiente de desenvolvimento

- Ferramentas de apoio à gestão do projeto

Como sistema de gestão de base de dados foi utilizado o Microsoft SQL Server, que foi a ferramenta que nos foi proposta para a realização da aplicação e permite gerir toda a informação da aplicação de forma segura e com um ótimo desempenho.

Como ambiente de desenvolvimento foi utilizado Microsoft Visual Studio 2022, um IDE, que foi utilizado para a implementação do código da aplicação, esta ferramenta também oferece recursos de *debug* que permite um desenvolvimento orientado a testes. Foi utilizado também a plataforma .NET, juntamente com a framework ASP.NET e Blazor que permitiu-nos desenvolver a interface gráfica da aplicação. O Blazor é uma tecnologia de desenvolvimento web baseada em C# e foi a nossa escolha devido à sua capacidade de permitir escrever código C# tanto no lado do cliente como no lado do servidor.

As ferramentas de apoio á gestão do projeto utilizadas foram, o Github que é uma plataforma essencial para o desenvolvimento de qualquer software em equipa, a plataforma Overleaf que foi de grande auxilio na criação da documentação do projeto, como linguagem de modelação dos diagramas elaborados na especificação foi utilizada o UML juntamente com a ferramenta Visual Paradigm que permite de forma gráfica criar os diagramas UML.

7 Conclusões e trabalho futuro

De maneira a haver um maior rigor no planeamento e consequente implementação do projeto, o seu desenvolvimento foi dividido em três fases distintas. Com este modelo de trabalho, foi possível planear o desenvolvimento do software de maneira mais rigorosa e estruturada, sendo cada fase dedicada a definir os aspetos centrais do projeto mais concretamente a Definição e Fundamentação, a Especificação da estrutura e comportamento, e a Implementação da Aplicação.

Na fase inicial do projeto, a Definição e Fundamentação, foram definidas as bases do nosso projeto passando pela contextualização do problema, objetivos finais do sistema, análise de viabilidade e alocação de recursos para todos os aspetos do seu desenvolvimento.

Na segunda fase, Requisitos e Especificação, passou-se a determinar os vários aspetos estruturais e comportamentais, desde o levantamento dos requisitos à elaboração de diagramas UML com o intuito de auxiliar a especificação do sistema.

Estas duas primeiras fases tiveram uma especial importância em garantir que o projeto tem um planeamento fundamentado, fiável e bem estruturado, permitindo assim uma melhor ideia do que se quer com o nosso produto de software, e consequentemente um melhor entendimento do que se pretende implementar.

Com a estrutura e requisitos do projeto especificados, passamos para a fase de implementação que consistiu no desenvolvimento do software tendo como suporte a especificação elaborada na fase anterior. Dada a dimensão deste projeto, do tempo disponível para o concluir e das ferramentas novas que tivemos de aprender a utilizar, não foi possível implementar todas as funcionalidades previstas na especificação. Em primeiro lugar, gostaríamos de ter o funcionamento do comprador completo, ou seja, permitir que visualizasse o histórico de compras e que pudesse ter mais opções de alteração do carrinho de compras. Quanto ao vendedor, gostaríamos de ter implementado as funcionalidades que lhe permitem submeter os seus produtos no sistema, adquirir bancas e configurar o respetivo catálogo.

Apesar das dificuldades, a implementação do sistema foi feita de acordo com a especificação definida nas fases anteriores e consideramos o desenvolvimento deste projeto bastante positivo, estando satisfeitos com a fase de planeamento pensada, a qual a disciplina destacou a sua importância.

Para trabalho futuro gostaríamos de terminar a implementação das funcionalidades relativas ao vendedor que não foram totalmente implementadas, de acordo com a especificação previamente

elaborada. Com o passar do tempo, serão necessárias novas funcionalidades para manter os utilizadores interessados na aplicação, como por exemplo, a implementação de uma lista de produtos favoritos, compatibilidade com novos métodos de pagamento, um conjunto de estatísticas para mostrar aos vendedores a sua atividade mensal, entre outras. Para além disso, à medida que o número de utilizadores da aplicação aumenta, será necessário um trabalho de vigilância e manutenção do sistema de base de dados, tendo que atender às eventuais necessidades futuras que se manifestem ao longo do tempo de vida da aplicação.

Referências

Apple: most popular app store categories 2022. (s.d.). Obtido 14 outubro 2022, de <https://www.statista.com/statistics/270291/popular-categories-in-the-app-store/>

Top categories on google play. (s.d.). Obtido 14 outubro 2022, de <https://www.appbrain.com/stats/android-market-app-categories>

Lista de Siglas e Acrónimos

BD Base de Dados

RGPD Regulamento Geral sobre a Proteção de Dados

CMB Câmara Municipal de Braga

SGBD Sistema de Gestão de Bases de Dados

HTML HyperText Markup Language

CSS Cascading Style Sheets

PBKDF2 Password-Based Key Derivation Function 2

UML Unified Modeling Language

CVV Card Verification Value

ASP Active Server Pages

NET Network Enabled Technologies

API Application Programming Interface

SQL Structured Query Language

CC Cartão de Cidadão

NIF Número de Identificação Fiscal

RF Requisito Funcional

RNF Requisito Não Funcional

DAO Data Access Object

IDE Integrated Development Environment

Anexos

Use case - Criar conta

Use case: Criar conta

Descrição: Este *use case* aborda a funcionalidade em que um utilizador pretende criar uma conta no serviço.

Pré-condição: True

Pós-condição: A conta é adicionada ao sistema.

Fluxo normal (1):

1. O sistema solicita o nome, CC, NIF, *username*, *password*, *e-mail* e o tipo de conta (cliente, vendedor).
2. O utilizador fornece essas informações.
3. O sistema verifica que não existem mais utilizadores com o mesmo CC, NIF, *username* e *e-mail*.
4. O sistema verifica que o tipo de conta pretendido é "cliente".
5. A conta é criada e adicionada ao sistema.

Fluxo de exceção (2) [Já existe um utilizador com o CC, NIF ou *e-mail* mencionado] (passo 3):

- 3.1. O sistema avisa o utilizador que já existe uma conta com o CC, NIF ou *e-mail* mencionado.
- 3.2. A conta não é adicionada ao sistema.

Fluxo alternativo (3) [O tipo de conta pretendido é "vendedor"] (passo 4):

- 4.1. O sistema marca a conta com aprovação pendente
- 4.2. A conta é criada e adicionada ao sistema a aguardar aprovação.

Use case - *Login*

Use case: Login

Descrição: Este *use case* aborda a funcionalidade em que um utilizador pretende aceder à sua conta.

Pré-condição: O utilizador tem uma conta no sistema.

Pós-condição: O utilizador entra na sua conta.

Fluxo normal (1):

1. O sistema solicita o *e-mail* e a *password*.
2. O utilizador fornece essas informações.
3. O sistema verifica que o *e-mail* existe.
4. O sistema verifica que a *password* está correta.
5. O sistema concede acesso à conta

Fluxo de exceção (2) [*E-mail* inexistente] (passo 3):

- 3.1. O sistema informa que desconhece o *e-mail* fornecido.
- 3.2. O sistema não concede o acesso ao utilizador.

Fluxo de exceção (3) [*Password* incorreta] (passo 4):

- 3.1. O sistema avisa o utilizador que a *password* inserida não é válida.
- 3.2. O sistema não concede acesso à conta.

Use case - Adição de produtos no carrinho

Use case: Adição de um ou mais produtos ao carrinho de compras

Descrição: Este *use case* aborda a funcionalidade em que um cliente adiciona produtos ao seu carrinho de compras.

Pré-condição: O utilizador está autenticado como cliente e entrou numa banca de uma feira.

Pós-condição: Os produtos selecionados são adicionados ao carrinho de compras do utilizador.

Fluxo normal (1):

1. O sistema apresenta todos os produtos presentes na banca.
2. O utilizador seleciona um produto da banca.
3. O utilizador define a quantidade que pretende adicionar ao carrinho de compras.
4. O sistema verifica que a quantidade disponível desse produto é igual ou superior à solicitada pelo utilizador.
5. O sistema adiciona o produto com a quantidade definida no carrinho de compras do utilizador.

Fluxo de exceção (2) [A quantidade disponível do produto selecionado é inferior à solicitada] (passo 4):

- 4.1. O sistema informa que a quantidade pedida pelo utilizador é superior à disponível.
- 4.2. O produto não é adicionado ao carrinho de compras do utilizador.

Use case - Alterar carrinho de compras

Use case: Alterar carrinho de compras

Descrição: Este *use case* aborda a funcionalidade em que um cliente edita os produtos que estão no seu carrinho de compras.

Pré-condição: O utilizador está autenticado como cliente e tem produtos no seu carrinho de compras.

Pós-condição: As alterações que o cliente desejou foram efetuadas.

Fluxo normal (1):

1. O sistema apresenta todos os produtos presentes no carrinho de compras do utilizador.
2. O sistema indica que o utilizador pode alterar a quantidade dos produtos presentes no seu carrinho de compras ou até removê-los.
3. O utilizador indica que pretende incrementar a quantidade de um certo produto no carrinho de compras.
4. O sistema verifica que a quantidade de produto disponível ainda consegue satisfazer a vontade do cliente.
5. O sistema regista as alterações pretendidas pelo utilizador.

Fluxo alternativo (2) [O utilizador pretende decrementar a quantidade de um certo produto] (passo 3):

- 3.1.a O sistema decremente uma unidade do produto selecionado no carrinho de compras do utilizador.
- 3.2.a O sistema verifica que a quantidade presente no carrinho de compras é superior a 0.
- 3.3.a O sistema regista a alteração efetuada.

Fluxo alternativo (3) [O sistema verifica que a quantidade presente no carrinho de compras é 0] (passo 3.2):

- 3.2.1 O sistema remove o produto do carrinho de compras.

Fluxo alternativo (4) [O utilizador pretende remover o produto do seu carrinho de compras] (passo 3):

- 3.1.b O sistema remove o produto do carrinho de compras do utilizador.

Fluxo de exceção (5) [A quantidade de produto disponível não consegue satisfazer a vontade do cliente] (passo 4)

- 4.1. O sistema informa que o utilizador não pode incrementar a quantidade daquele produto.
- 4.2. O sistema impede a alteração pretendida pelo cliente.

Use case - Finalização de uma compra

Use case: Finalização da compra

Descrição: Este *use case* aborda a funcionalidade em que um cliente finaliza uma compra.

Pré-condição: O utilizador está autenticado como cliente e tem produtos no seu carrinho de compras.

Pós-condição: A compra é finalizada.

Fluxo normal (1):

1. O sistema apresenta todos os produtos presentes no carrinho de compras do utilizador, bem como o montante total da compra.
2. O utilizador confirma a compra.
3. O sistema verifica que a quantidade de cada produto no carrinho de compras é igual ou inferior à quantidade disponível de cada produto.
4. O sistema solicita ao utilizador que forneça os dados do cartão de crédito (Número, CVV, data de validade e titular)
5. O utilizador fornece os dados do seu cartão de crédito.
6. O sistema verifica que o cartão de crédito é válido.
7. O sistema subtrai à quantidade disponível de cada produto a quantidade pretendida pelo utilizador.
8. A compra é efetuada.

Fluxo de exceção (2) [A quantidade pretendida pelo utilizador em um ou mais produtos é superior à quantidade disponível] (passo 3):

- 3.1. O sistema indica qual o produto cuja quantidade pretendida é superior à disponível.
- 3.2. O sistema indica que a compra não pode ser realizada indicando quais os produtos indisponíveis.
- 3.3. A compra não é efetuada.

Fluxo de exceção (3) [O cartão de crédito do utilizador é inválido] (passo 6):

- 6.1. O sistema avisa o utilizador que o seu cartão de crédito é inválido.
- 6.2. A compra não é efetuada.

Use case - Criar produto

Use case: Criação de um produto

Descrição: Este *use case* aborda a funcionalidade em que um vendedor adiciona um produto seu, ao sistema.

Pré-condição: O utilizador está autenticado como vendedor.

Pós-condição: O produto é adicionado ao sistema.

Fluxo normal (1):

1. O sistema solicita o nome, descrição, categoria, preço por unidade, quantidade disponível e uma imagem do produto.
2. O utilizador fornece essas informações.
3. O sistema verifica que o produto que se pretende adicionar não existe na conta do utilizador
4. O produto é adicionado ao sistema.

Fluxo de exceção (2) [O produto que pretende adicionar já existe na conta do vendedor] (passo 3):

- 3.1. O sistema avisa o utilizador que este produto já existe na sua conta.
- 3.2. O produto não é adicionado ao sistema.

Use case - Configuração de um produto

Use case: Configuração de um produto

Descrição: Este *use case* aborda a funcionalidade em que um vendedor altera um produto seu.

Pré-condição: O utilizador está autenticado como vendedor.

Pós-condição: O produto é alterado conforme o utilizador pretende.

Fluxo normal (1):

1. O sistema apresenta os produtos associados à conta do utilizador.
2. O utilizador seleciona o produto que pretende alterar.
3. O sistema indica que o utilizador pode alterar a quantidade disponível, a descrição, o preço ou a imagem.
4. O utilizador altera os campos que pretende.
5. O sistema regista as alterações pretendidas pelo utilizador.

Use case - Remoção de um produto

Use case: Remoção de um produto

Descrição: Este *use case* aborda a funcionalidade em que um vendedor remove um produto seu.

Pré-condição: O utilizador está autenticado como vendedor e tem produtos associados a si.

Pós-condição: O produto é removido.

Fluxo normal (1):

1. O sistema apresenta os produtos associados à conta do utilizador.
2. O utilizador seleciona o produto que pretende remover.
3. O produto é removido do sistema.

Use case - Solicitação de uma banca de uma feira

Use case: Solicitação de uma banca de uma feira

Descrição: Este *use case* aborda a funcionalidade em que um vendedor pretende adquirir uma banca de uma feira.

Pré-condição: O utilizador está autenticado como vendedor e não possui uma banca na feira.

Pós-condição: Uma banca é atribuída ao vendedor

Fluxo normal (1):

1. O sistema apresenta as feiras disponíveis ao utilizador.
2. O utilizador seleciona uma das feiras.
3. O utilizador indica que pretende obter uma banca na feira selecionada.
4. O sistema verifica se o nº limite de bancas não é ultrapassado.
5. O sistema solicita um título para a banca.
6. O sistema atribui uma banca ao utilizador.

Fluxo de exceção (2) [O número limite de bancas nesta feira já foi atingido] (passo 3):

- 3.1. O sistema informa que o número limite de bancas da feira já foi atingido.
- 3.2. Nenhuma banca é atribuída ao utilizador.

Use case - Configuração de uma banca de uma feira

Use case: Configuração de uma banca de uma feira.

Descrição: Este *use case* aborda a funcionalidade em que um vendedor adiciona ou remove um produto seu na banca de uma feira.

Pré-condição: O utilizador está autenticado como vendedor.

Pós-condição: As alterações pretendidas pelo utilizador são realizadas.

Fluxo normal (1):

1. O sistema apresenta ao utilizador os produtos presentes na sua banca.
2. O sistema indica que o utilizador pode adicionar ou remover produtos da sua banca.
3. O utilizador indica que pretende adicionar um produto à sua banca.
4. O sistema apresenta ao utilizador os produtos associados à sua conta.
5. O utilizador seleciona um dos seus produtos
6. O sistema verifica que a categoria do produto selecionado é coerente com a categoria da feira.
7. O produto é adicionado à banca do utilizador.

Fluxo alternativo (2) [O utilizador indica que pretende remover um produto da sua banca] (passo 3):

- 3.1. O utilizador seleciona o produto que quer remover da banca.
- 3.2. O sistema remove o produto da banca.

Fluxo de exceção (3) [A categoria do produto a adicionar não é coerente com a categoria da feira] (passo 6):

- 6.1. O sistema indica que o produto não pode ser adicionado à banca.
- 6.2. O produto não é adicionado.

Use case - Aprovação da conta de um vendedor

Use case: Aprovação da conta de um vendedor

Descrição: Este *use case* aborda a funcionalidade em que um administrador aprova uma conta de vendedor.

Pré-condição: O utilizador está autenticado como administrador

Pós-condição: A conta de vendedor pendente de aprovação é aprovada.

Fluxo normal (1):

1. O utilizador solicita ao sistema para apresentar as contas de vendedores pendentes de aprovação.
2. O sistema apresenta ao utilizador todas as contas de vendedores que ainda não foram aprovadas.
3. O utilizador seleciona uma das contas.
4. O sistema apresenta ao utilizador as informações da conta selecionada
5. O utilizador aprova a conta.
6. A conta é adicionada ao sistema.

Fluxo de exceção (2) [A conta não é aprovada pelo utilizador] (passo 5):

5.1 O utilizador não aprova a conta.

5.2 A conta não é adicionada ao sistema e é descartada da lista de contas de vendedores pendentes.

Use case - Criação de uma feira

Use case: Criação de uma feira

Descrição: Este *use case* aborda a funcionalidade em que um administrador adiciona uma feira ao sistema.

Pré-condição: O utilizador está autenticado como administrador.

Pós-condição: A feira é adicionada ao sistema.

Fluxo normal (1):

1. O sistema solicita ao utilizador o nome, âmbito/categoria, limite de bancas, a data de início e a data de fim da feira.
2. O utilizador fornece essas informações.
3. O sistema verifica que o nome da feira não é igual ao de outras feiras no sistema.
4. A feira é adicionada ao sistema.

Fluxo de exceção (2) [O nome da feira que se está a criar já existe numa feira presente no sistema] (passo 3):

3.1. O sistema avisa o utilizador que a feira não pode ser adicionada ao sistema.

3.2. A feira não é adicionada ao sistema.

Mockup - Página de autenticação

The mockup shows a login form titled "LOGIN". It features two input fields: one for "Insira o e-mail" (Email) with a user icon and another for "Insira a password" (Password) with a lock icon. Below these are two buttons: "Iniciar sessão" (Start session) and "Criar conta" (Create account).

Mockup - Página de criação de conta

The mockup shows a form titled "Crie a sua conta" (Create your account). It includes several input fields: "Nome" (Name), "E-mail" (Email), "Nome de utilizador" (User name), "Número do cartão de cidadão" (Social security number), "Número de identificação fiscal (NIF)" (Tax identification number), "Rua, n.º porta, andar" (Street, number, floor), "Cidade" (City), and "Código postal (XXXX-XXX)" (Postcode). To the right of the form is a note: "Caso pretenda ser um vendedor, apresente-se. Diga quais são os seus objectivos e que tipos de produtos quer vender. Se deixar este espaço em branco, a sua conta não será aprovada." (If you want to be a seller, present yourself. Tell us what your objectives are and what types of products you want to sell. If you leave this space blank, your account will not be approved.). At the bottom is a "Criar conta" (Create account) button.

Mockup - Histórico de compras

The mockup shows a user interface for a shopping application. At the top, there is a header with a logo (eFeiras), a welcome message ('Bem vindo, Alberto Magalhães'), a shopping cart icon, and a menu icon. Below the header, the main content area is titled 'As suas compras' (Your purchases). It displays a table of three recent purchases:

Número	Data	Total	Ação
301103831	05/11/2022	44,32€	Ver detalhes
300872464	23/10/2022	12,95€	Ver detalhes
300287222	13/10/2022	30,00€	Ver detalhes

At the bottom of the content area, there is a 'Voltar' (Back) button.

Instruções SQL para a criação das tabelas da base de dados

-- Este ficheiro serve para criar todas as tabelas da base de dados.

```
USE eFeiras;

-- 1
CREATE TABLE Utilizador (
    id int not null identity(1,1) PRIMARY KEY,
    userType tinyint not null,
    nome varchar(45),
    nif varchar(9),
    cartao_cidadao varchar(9),
    e_mail varchar(45),
    rua_porta_andar varchar(45),
    cidade varchar(45),
    codigo_postal varchar(45),
    apresentacao varchar(4000),
    aprovado bit not null,
    username varchar(45) not null,
```

```

        password_ varchar(45) not null
);

-- 2
CREATE TABLE Categoria (
    id int not null identity(1,1) PRIMARY KEY,
    nome varchar(45)
);

-- 3
create table Subcategoria (
    id int not null identity(1,1) PRIMARY KEY,
    nome varchar(45),
    categoria_id int not null FOREIGN KEY REFERENCES Categoria(id)
);

-- 4
create table Feira (
    id int not null identity(1,1) PRIMARY KEY,
    titulo varchar(50) not null,
    descricao varchar(500) not null,
    data_inicio date not null,
    data_fim date not null,
    imagem varchar(500) not null,
    limite_bancas int not null,
    categoria_id int not null FOREIGN KEY REFERENCES Categoria(id)
);

-- 5
create table Banca(
    Feira_id int not null foreign key references Feira(id),
    Utilizador_id int not null foreign key references Utilizador(id),
    titulo varchar(50) not null,
    primary key (Feira_id,Utilizador_id),
);

-- 6
create table Produto(
    id int not null identity(1,1) PRIMARY KEY,

```

```

        nome varchar(45) not null,
        descricao varchar(500) not null,
        preco decimal(7,2) not null,
        quantidade_disponivel int not null,
        imagem varchar(500) not null,
        Utilizador_id int not null foreign key references Utilizador(id),
        SubCategoria_id int not null foreign key references Subcategoria(id)
    );

-- 7
create table banca_has_produto(
    Feira_id int not null foreign key references Feira(id),
    Utilizador_id int not null foreign key references Utilizador(id),
    Produto_id int not null foreign key references Produto(id),
    primary key (Feira_id,Utilizador_id,Produto_id)
);

-- 8
create table Compra(
    id int not null identity(1,1) primary key,
    montante decimal(7,2) not null,
    data datetime not null,
    Utilizador_id int not null foreign key references Utilizador(id)
);

-- 9
create table compra_has_produto(
    Compra_id int not null foreign key references Compra(id),
    Produto_id int not null foreign key references Produto(id),
    quantidade int not null,
    primary key(Compra_id,Produto_id)
);

```