

Universidade do Minho

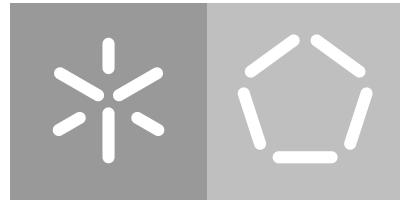
Escola de Engenharia

Departamento de Informática

Ana Catarina Gil, a85266
Ana Margarida Campos, a85166
Joana Afonso Gomes, a84912
Pedro Fernandes, a84313
Tânia Rocha, a85176

Projeto de Laboratórios de Informática IV
Plataforma de Aluguer e Venda *RentItAll*

Julho 2020



Universidade do Minho

Escola de Engenharia

Departamento de Informática

Ana Catarina Gil, a85266
Ana Margarida Campos, a85166
Joana Afonso Gomes, a84912
Pedro Fernandes, a84313
Tânia Rocha, a85176

Projeto de Laboratórios de Informática IV
Plataforma de Aluguer e Venda *RentItAll*

Laboratórios de Informática IV
Mestrado Integrado em Engenharia Informática

Trabalho efetuado sob orientação de
Daniela Sofia Rijo Oliveira

Julho 2020

A B S T R A C T

This report is based on a brief description and demonstration of a project for the course unit of Informatic Laboratories IV, which its objective was to implement a Web or Mobile Interface.

The developed application is called **RentItAll**, which has the purpose of making available a wide range of items to sell and to rent through a Web Interface. In the following sections of the report, we will be mentioning the development of the application from market analysis and initial study, accompanied by the appropriate diagrams, to the actual implementation of the website.

Application areas: Specification and development of Web and Mobile applications, using Software Engineering methodologies

Keywords: Software Engineering, Software Design and Development, Microsoft, Relational Database, Online Shopping

RESUMO

O presente relatório é constituído por uma breve descrição e demonstração do projeto realizado no âmbito da Unidade Curricular Laboratórios de Informática IV, no qual era pedido a implementação de uma Interface Web (responsive) ou Mobile.

A aplicação desenvolvida tem o título de **RentItAll**, que tem como propósito disponibilizar quaisquer itens/propriedades para vender ou alugar através de uma Interface Web.

Nas secções seguintes serão analisadas todas as etapas de desenvolvimento do projeto, desde a análise de mercado e estudo inicial, acompanhado dos devidos diagramas, até à própria implementação da página Web.

Área de Aplicação: Especificação e desenvolvimento de aplicações Web e Mobile, utilizando metodologias no âmbito da Engenharia de Software

Palavra-Chave: Engenharia de Software, Concepção e Desenvolvimento de Software, Microsoft, Base de Dados Relacional, Compras/Alugueres online

CONTÉUDO

1	INTRODUÇÃO	1
1.1	Contexto	1
1.2	Motivação	2
1.2.1	Análise do mercado	2
1.2.2	Análise do Meio Envolvente	2
1.3	Objetivos	3
1.4	Estrutura	4
2	ESTUDO INICIAL	5
2.1	Levantamento de Requisitos	5
2.2	Diagrama de Classes	10
2.3	Use Cases	10
2.4	Mockups	17
3	ESTRUTURA DA APLICAÇÃO	23
3.1	Proposta sugerida - soluções	23
3.1.1	Arquitetura do Sistema	24
4	BASE DE DADOS	27
4.1	Modelo Conceitual	27
4.2	Modelo Lógico	28
4.3	Dicionário de Dados	29
4.4	Povoamento	30
5	DESENVOLVIMENTO	31
5.1	Decisões	31
5.2	Implementação	31
5.2.1	Login	31
5.2.2	Adicionar Utilizador	32
5.2.3	Encriptação de Passwords	33
5.2.4	Recuperação de Password	34
5.2.5	Cookies	35
5.2.6	Pesquisa de Artigos	35
5.2.7	Artigos	37
5.2.8	Alterar Dados	40
5.2.9	Carrinho de Compras	41
5.2.10	Alugueres	45

5.2.11 Notificações	45
5.2.12 Histórico	47
5.2.13 Pontuação	48
5.2.14 Comentários	50
5.2.15 Denúncias	51
5.3 Métodos do Administrador	52
5.3.1 Denúncias	52
5.3.2 Utilizadores	53
5.3.3 Estatísticas	55
5.4 Métodos da Company	57
5.5 Outros resultados frontend	57
5.5.1 Botão para o topo da página	57
5.5.2 Contactos	57
5.5.3 Sobre Nós	57
6 CONCLUSÃO	58
6.1 Conclusões	58
6.2 Perspectiva de trabalho futuro	58
.1 Anexo 1: Página inicial das Company	60
.2 Anexo 2: Vista reduzida e não iterativa da página Sobre Nós	61

L I S T A D E F I G U R A S

Figura 1	Percentagem de compras online na Europa	2
Figura 2	Diagrama de Classes	10
Figura 3	Diagrama de Use Cases	16
Figura 4	Mockup da página inicial	17
Figura 5	Mockup do footer	18
Figura 6	Mockup da publicidade ao <i>voucher</i>	18
Figura 7	Mockup da view dos detalhes de um produto	19
Figura 8	Mockup do carrinho de compras	20
Figura 9	Mockup do carrinho de compras	21
Figura 10	Mockup da página de registo de um utilizador <i>single</i>	22
Figura 11	Mockup da página de registo de um utilizador <i>company</i>	22
Figura 12	Esquema dos Models do projeto	25
Figura 13	Esquema dos Controllers do projeto	26
Figura 14	Esquema das Views do projeto	26
Figura 15	Modelo lógico da Base de Dados	27
Figura 16	Modelo lógico da Base de Dados	28
Figura 17	Parcial da página de Login	32
Figura 18	View Adicionar Utilizador	33
Figura 19	Base de Dados: passwords de utilizadores	34
Figura 20	Email recebido via Gmail	35
Figura 21	Barra de pesquisa	36
Figura 22	Categorias	37
Figura 23	View <i>SearchCategoria</i>	37
Figura 24	Mais Vendidos	37
Figura 25	Melhor Classif.	37
Figura 26	Novidades	37
Figura 27	View que permite ver os artigos do Utilizador	39
Figura 28	Inserir novo Artigo	39
Figura 29	Exemplo de slider de um artigo com mais do que uma imagem	40
Figura 30	Utilizador: Alterar dados	40
Figura 31	Alterar Password	41
Figura 32	Adicionar artigo ao Carrinho na view <i>Details</i>	43

Figura 33	View <i>Adicionar Carrinho</i> , após Utilizador adicionar um produto ao carrinho	43
Figura 34	View <i>VendaInfo</i> , o carrinho de compras	44
Figura 35	Notificação de quantidade máxima	44
Figura 36	<i>Selector</i> com <i>voucher</i> disponíveis	44
Figura 37	Preço actualizado após desconto	44
Figura 38	Notificação da tentativa de uso de <i>voucher</i> com valor de oferta superior ao total do carrinho	44
Figura 39	Utilizador: selecionar datas de aluguer	45
Figura 40	Navbar sem notificações	47
Figura 41	Badge com notificações	47
Figura 42	Utilizador - Histórico	47
Figura 43	Utilizador - Histórico de Alugueres	48
Figura 44	Avaliação de um produto na view <i>Details</i> do Utilizador	49
Figura 45	Pontuação de um produto na view <i>Details</i> da Home (não logado)	49
Figura 46	Pontuação de produtos em views de amostragem de artigos	50
Figura 47	Envio de um comentário na View <i>Details</i> do Utilizador.	50
Figura 48	Botão para denunciar	51
Figura 49	Inserir denúncia	51
Figura 50	Aceitar ou rejeitar uma denúncia	53
Figura 51	Bloquear ou enviar aviso a um utilizador	54
Figura 52	Email de aviso recebido.	55
Figura 53	Numerário de elementos do site	55
Figura 54	Evolução de vendas comparativamente a alugueres ao longo do tempo	56
Figura 55	Listagem das denúncias recentes	56
Figura 56	Numerário de elementos mais recentes do site	56
Figura 57	Parcial de uma view com botão para regressar ao topo da página	57
Figura 58	Página de Contactos	57

L I S T A D E T A B E L A S

Tabela 1	Requisito 1	5
Tabela 2	Requisito 2	6
Tabela 3	Requisito 3	6
Tabela 4	Requisito 4	7
Tabela 5	Requisito 5	7
Tabela 6	Requisito 6	8
Tabela 7	Requisito 7	8
Tabela 9	Requisito 9	9
Tabela 8	Requisito 8	9
Tabela 10	Dicionário de Dados	29

1

INTRODUÇÃO

Este projeto teve como principal objectivo motivar-nos para a utilização de linguagens de programação e procedimentos próprios do desenvolvimento de uma aplicação Web.

Durante a sua execução foi desenvolvida uma página web capaz de representar um conjunto de artigos para um indivíduo ou empresa poder disponibilizar e para outras pessoas comprarem ou alugarem os respetivos itens.

Concebemos como requisitos para esta resolução a implementação de diversas funcionalidades, que serão descritas e apresentadas ao longo deste relatório.

Dado o relatório ter sido elaborado paralelamente à execução das diversas etapas do trabalho, a pormenorização e visualização das fases do projeto pode não transparecer fielmente o resultado final, visto que foi sendo aperfeiçoado conforme o seu desenvolvimento.

1.1 CONTEXTO

Decidimos nomear a nossa aplicação como RentItAll, uma vez que estamos a proporcionar um serviço que disponibiliza quaisquer artigos (itens ou propriedades) para aluguer, não obstante o facto de que também é possível, para além de alugueres, efetuar compras e vendas de artigos.

Determinámos que uma aplicação deste género é necessária para ajudar os utilizadores a deliberarem se pretendem ou não comprar artigos de uma gama de preços elevada, podendo assim utilizar esses itens a um preço acessível, simultaneamente seguindo um método de reutilizar materiais e desta forma originar um negócio sustentável.

1.2 MOTIVAÇÃO

1.2.1 Análise do mercado

De acordo com inúmeras pesquisas, o hábito de fazer compras *online* cresceu significativamente em Portugal desde o início da década. Hoje, perto de quatro em cada dez residentes compram produtos ou serviços através de um computador ou telemóvel. É uma prática impulsionada pela massificação da tecnologia, por um crescimento da literacia digital e por um aumento da oferta por parte das empresas.

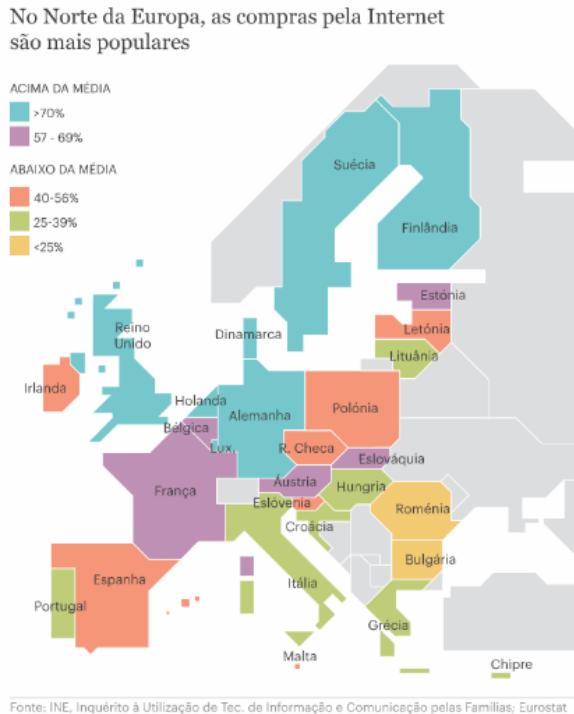


Figura 1: Percentagem de compras online na Europa

1.2.2 Análise do Meio Envolvente

- Clientes

Numa estratégia inicial, todos os cidadãos residentes em Portugal poderão ser possíveis clientes da nossa plataforma desde que possuam acesso à internet.

Como o propósito deste site é o aluguer ou venda de diferentes itens, os clientes podem recorrer a estes serviços contribuindo para a reutilização de produtos e **diminuindo**, consequentemente, a **pegada ecológica**.

- **Fornecedores**

De acordo com os princípios do projeto, e tal como referido anteriormente, também os fornecedores serão utilizadores do site.

Segundo dados mais recentes, o **negócio pessoal** tem crescido cada vez mais, entre os quais o investimento particular de produtos a preços elevados como casas, carros, itens luxuosos,etc, podendo lucrar posteriormente através do aluguer ou venda dos mesmos.

Também o acumulo e a compra de produtos desnecessários é algo com que a maioria das pessoas se pode identificar. A utilização desta plataforma proporciona um **novo uso dos produtos** acompanhado de um ganho monetário.

- **Concorrentes**

A grande parte de plataformas disponibilizadas para aluguer são, normalmente, centradas num determinado produto (casas, carros, roupas...). Com base nisto, alguns dos nossos concorrentes mais poderosos são:

- **Airbnb**, uma plataforma online direcionada ao arrendamento de alojamentos em qualquer parte do mundo;
- **Chic by Choice**, um site português com o propósito de alugar roupas de luxo;
- **Europcar** é uma plataforma reservada para o aluguer de veículos em todo o mundo.

Como principal concorrente temos o site **Rnters** que é direcionado para o aluguer de diferentes tipos de produtos. A nossa principal diferença em relação a esta plataforma é que no nosso caso, o utilizador não só pode alugar como permitir a venda.

1.3 OBJETIVOS

Os principais objetivos deste trabalho são os seguintes:

- Delineamento de uma estratégia logística e flexível de desenvolvimento;
- Desenvolvimento iterativo e incremental ao longo do projeto;
- Desenvolvimento ágil, progressivo e melhorado do produto;

- Coesão e melhoria progressiva da dinâmica da equipa e da sua comunicação interna diária;
- Adoção de uma metodologia própria de gestão do projeto para discussão quinzenal com o docente responsável.

1.4 ESTRUTURA

Este relatório encontra-se estruturado da seguinte forma:

- **Introdução:** Neste capítulo é efetuada uma contextualização do problema na qual é abordado o tipo de aplicação a desenvolver bem como o modelo de negócio associado.
- **Estudo Inicial:** Neste capítulo é demonstrado o processo de levantamento de requisitos, enumerados e especificados os diferentes Use-Cases, o diagrama de Classes inicial e os *mockups* usados como base para a constução da plataforma.
- **Estrutura da Aplicação:** Na Estrutura da Aplicação são identificados os atores do sistema assim como a sua divisão e arquitetura.
- **Base de Dados:** Neste capítulo é abordado o processo de desenvolvimento da Base de Dados onde se enquadra o modelo Conceptual e Lógico da mesma bem como um dicionário de dados e povoamento inicial.
- **Desenvolvimento:** Neste são especificadas todas as funcionalidades do nosso sistema e todas as decisões tomadas pelo grupo para a implementação da plataforma.
- **Conclusão:** Por último, é feita uma reflexão à cerca do trabalho desenvolvido e identificados os próximos passos a tomar face ao trabalho futuro.

2

ESTUDO INICIAL

2.1 LEVANTAMENTO DE REQUISITOS

Tabela 1: Requisito 1

Requisitos Funcionais							
F1: Criação de contas		Oculto: Não					
Descrição: O site suporta a criação de contas de utilizadores, associadas aos dados pessoais do utilizador, como o nome, morada, contacto, conta bancária, avaliação, etc...							
Requisitos Não Funcionais							
Nome	Restrição	Categoria	Desejável	Permanente			
NF1.1 Controlo de acesso	A função só pode ser acedida por usuários sem nenhum perfil atribuído.	Segurança	()	(X)			
NF1.2 Identificação da conta	As contas devem ser identificadas por um email, isto é, o contacto do utilizador.	Interface	()	(X)			
NF1.3 Tempo de registo	O tempo para registo de cada conta deve ser inferior a 'x' segundo(s)	Performance	(X)	()			
NF1.4 Janela única	Todas as funções relacionadas a criar contas devem ser efetuadas numa única janela.	Interface	(X)	(X)			
NF1.5 Guardar na base de dados	Na criação da conta, acede-se à base de dados e adiciona-se um novo objeto de classe Conta com o contacto como chave primária.	Dados	()	(X)			

Tabela 2: Requisito 2

Requisitos Funcionais								
F2: Gestão de contas	Oculto: Não							
Descrição: O site suporta a gestão de contas de utilizadores.								
Requisitos Não Funcionais								
Nome	Restrição	Categoria	Desejável	Permanente				
NF2.1 Controlo de acesso	A função só pode ser acedida por usuários com o perfil de utilizador e apenas ao perfil da própria conta	Segurança	()	(X)				
NF2.2 Alteração de dados	A função permite a alteração de qualquer parâmetro de dados pessoais com exceção o contacto principal, o email.	Interface	()	(X)				
NF2.3 Tempo de atualização	O tempo para atualizar os dados dentro da base de dados deve ser inferior a um segundo.	Performance	(X)	()				
NF2.4 Janela única	Todas as funções relacionadas á gestão de contas devem ser efetuadas numa única janela.	Interface	(X)	(X)				
NF2.5 Guardar na base de dados	Após a gestão da conta, acede-se à base de dados e atualiza-se o objeto de classe Conta com o contacto como chave primária, e altera-se o dados alterados pelo utilizador.	Dados	()	(X)				
NF2.6 Alteração de password	A função pede a confirmação da password anterior para o utilizador poder alterar para uma nova password.	Segurança	(X)	(X)				

Tabela 3: Requisito 3

Requisitos Funcionais								
F3: Alugar um item (<i>rent/buy</i>)	Oculto: Não							
Descrição: Um utilizador pode alugar um ou mais itens, de outro(s) utilizador(es), no modo <i>rent/buy</i> .								
Requisitos Não Funcionais								
Nome	Restrição	Categoria	Desejável	Permanente				
NF3.1 Controlo de acesso	A função pode ser acedida por qualquer usuário com perfil.	Segurança	()	(X)				
NF3.2 Identificação do item	O item tem um identificador próprio, uma sequência de caracteres.	Interface	()	(X)				
NF3.3 Tempo de aluguer	A duração de um aluguer é o tempo definido pelo utilizador dentro do periodo definido pelo dono do item em questão.	Interface	()	(X)				
NF3.4 Janela única	Todas as funções relacionadas a aluguer devem ser efetuadas numa única janela.	Interface	(X)	(X)				
NF3.5 Envio de mensagem	No alugamento do item, após o utilizador preencher os dados, irá ser enviado uma pedido de confirmação (F7) ao utilizador que é proprietário do item para confirmação.	Interface	(X)	(X)				
NF3.6 Pagamento	No item está associado um preço de aluguer que será o preço que o utilizador necessitará de pagar por mês pelo item.	Interface	()	(X)				
NF3.7 Atualização do estado	O estado do artigo na base de dados é atualizado para "Indisponível".	Dados	()	(X)				

Tabela 4: Requisito 4

Requisitos Funcionais							
F4: Alugar um artigo (owner)		Oculto: Não					
Descrição: Um utilizador põe a alugar um ou mais itens, no modo <i>owner</i> .							
Requisitos Não Funcionais							
Nome	Restrição	Categoria	Desejável	Permanente			
NF4.1 Controlo de acesso	A função pode ser acedida por qualquer usuário com perfil.	Segurança	()	(X)			
NF4.2 Identificação do item	O item tem um identificador próprio, uma sequência de caracteres.	Interface	()	(X)			
NF4.3 Tempo de aluguer	O utilizador define um período de tempo, no qual um outro utilizador pode alugar o artigo.	Interface	()	(X)			
NF4.4 Janela única	Todas as funções relacionadas a aluguer devem ser efetuadas numa única janela.	Interface	(X)	(X)			
NF4.5 Definição do preço	O utilizador define o preço mensal para o aluguer do artigo.	Interface	()	(X)			
NF4.6 Guardar item na base de dados	Na disponibilização de um item para aluguer, acede-se à base de dados e adiciona-se um novo objeto de classe Artigo com o identificador do artigo como chave primária e o parâmetro 'modo' como "Aluguer", como também os diferentes dados definidos pelo utilizador.	Dados	()	(X)			

Tabela 5: Requisito 5

Requisitos Funcionais							
F5: Pedido de confirmação		Oculto: Sim					
Descrição: O sistema envia um pedido de aluguer ou compra de um artigo para o vendedor para confirmação.							
Requisitos Não Funcionais							
Nome	Restrição	Categoria	Desejável	Permanente			
NF5.1 Envio	O pedido é enviado para o vendedor.	Especificação	()	()			
NF5.2 Confirmação	O vendedor confirma e aceita a proposta do utilizador.	Interface	()	(X)			
NF5.3 Resposta	A confirmação e página de pagamento é enviada para o utilizador.	Especificação	()	()			

Tabela 6: Requisito 6

Requisitos Funcionais								
F6: Vender um artigo (owner)	Oculto: Não							
Descrição: Um utilizador põe á venda um ou mais itens.								
Requisitos Não Funcionais								
Nome	Restrição	Categoria	Desejável	Permanente				
NF6.1 Controlo de acesso	A função pode ser acedida por qualquer usuário com perfil.	Segurança	()	(X)				
NF6.2 Identificação do item	O item tem um identificador próprio, uma sequência de caracteres.	Interface	()	(X)				
NF6.3 Tempo de aluguer	O utilizador define um período de tempo, no qual um outro utilizador pode alugar o artigo.	Interface	()	(X)				
NF6.4 Janela única	Todas as funções relacionadas a aluguer devem ser efetuadas numa única janela.	Interface	(X)	(X)				
NF6.5 Definição do preço	O utilizador define o preço de venda.	Interface	()	(X)				
NF6.6 Guardar item na base de dados	Na disponibilização de um item para aluguer, acede-se à base de dados e adiciona-se um novo objeto de classe Artigo com o identificador do artigo como chave primária e o parâmetro 'modo' como "Venda", como também os diferentes dados definidos pelo utilizador.	Dados	()	(X)				

Tabela 7: Requisito 7

Requisitos Funcionais								
F7: Definição de categorias	Oculto: Sim							
Descrição: O sistema associa uma ou mais categorias aos diferentes itens no sistema.								
Requisitos Não Funcionais								
Nome	Restrição	Categoria	Desejável	Permanente				
NF7.1 Atualização de dados	O sistema adiciona ao parâmetro de categorias uma ou várias, dependendo do item.	Dados	(X)	(X)				

Tabela 9: Requisito 9

Requisitos Funcionais					
F9: Comprar um artigo (<i>rent/buy</i>)	Oculto: Não				
Descrição: Um utilizador pode comprar um ou mais itens, de outro(s) utilizador(es), no modo <i>rent/buy</i> .					
Requisitos Não Funcionais					
Nome	Restrição	Categoria	Desejável	Permanente	
NF9.1 Controlo de acesso	A função pode ser acedida por qualquer usuário com perfil.	Segurança	()	(X)	
NF9.2 Identificação do item	O item tem um identificador próprio, uma sequência de caracteres.	Interface	()	(X)	
NF9.3 Preço	O preço de venda já se encontra associado ao item, mas em caso de o utilizador já ter alugado o mesmo item, o sistema determina o preço final tendo em conta o total já gasto no alugamento do artigo.	Interface	(X)	(X)	
NF9.4 Janela única	Todas as funções relacionadas a aluguer devem ser efetuadas numa única janela.	Interface	(X)	(X)	
NF9.5 Envio de mensagem	Na compra do item, após o utilizador preencher os dados, irá ser enviado uma pedido de confirmação (F7) ao utilizador que é proprietário do item para confirmação.	Interface	(X)	(X)	
NF9.6 Pagamento	Na resposta do pedido de confirmação irá vir uma página de pagamento para confirmação da compra, com ou sem recibo.	Interface	()	(X)	
NF9.7 Atualização do estado	Na base de dados, remove-se o artigo identificado pelo seu identificador.	Dados	()	(X)	
NF9.8 Devolução	Se o utilizador verificar que, na entrega, o artigo se encontra danificado, tem a possibilidade de submeter um pedido de devolução.	Interface	(X)	(X)	

Tabela 8: Requisito 8

Requisitos Funcionais					
F8: Pontuação	Oculto: Não				
Descrição: Após um aluguer, na devolução, o proprietário do artigo avalia o estado do item, dando uma avaliação ao utilizador que o alugou.					
Requisitos Não Funcionais					
Nome	Restrição	Categoria	Desejável	Permanente	
NF8.1 Controlo de acesso	A função pode ser acedida por um usuário que recebeu o seu artigo após o seu aluguer.	Segurança	()	(X)	
NF8.2 Identificação do item	O item tem um identificador próprio, uma sequência de caracteres.	Interface	()	(X)	
NF8.3 Janela única	Todas as funções relacionadas a aluguer devem ser efetuadas numa única janela.	Interface	(X)	(X)	
NF8.4 Pontuação	O utilizador define a pontuação a atribuir.	Interface	()	(X)	
NF8.5 Estragos	Em caso de o utilizador ver que o seu artigo está danificado, informa o sistema sobre o estragos e custos de arranjo.	Segurança	(X)	(X)	

2.2 DIAGRAMA DE CLASSES



Figura 2: Diagrama de Classes

2.3 USE CASES

Use Case: Criar Conta

Descrição: Utilizador cria uma conta no Sistema

Pré-condição: Aceder ao site

Pós-condição: Uma nova conta é criada com sucesso

Fluxo-Normal:

1. Utilizador clica no botão de criar conta do tipo single
 2. Sistema apresenta uma interface identificando as credenciais necessárias para o registo
 3. Utilizador preenche estas mesmas credenciais de acordo com a sua informação pessoal
 4. Sistema valida todos os dados
 5. Utilizador fica registado na plataforma.

Fluxo-Alternativo 1: [Utilizador quer criar conta do tipo company (passo 1)]

1.1 Utilizador clica no botão de criar conta do tipo company

1.2 Regressa ao passo 2

Fluxo-Exceção 1: [O email utilizado já está a ser usado] (passo 4)

1.1 Sistema não valida os dados e informa que o email já está a ser utilizado

1.2 Utilizador não fica registado

Fluxo-Exceção 2: [Campos vazios] (passo 4)

2.1 Sistema não valida os dados e informa que ainda existem campos para preencher

2.2 Utilizador não fica registado

Fluxo-Exceção 3: [Campos inválidos] (passo 4)

3.1 Sistema não valida os dados e informa que quais os campos inválidos

3.2 Utilizador não fica registado

Use Case: *Login*

Descrição: Utilizador realiza a ação de iniciar sessão na plataforma

Pré-condição: Aceder ao site

Pós-condição: O utilizador entra na sua conta

Fluxo-Normal:

1. Utilizador clica no botão de iniciar sessão
2. Sistema apresenta interface com os campos email e password por preencher
3. Utilizador preenche os campos
4. Sistema valida os dados
5. Utilizador entra na sua conta

Fluxo-Exceção 1: [Password incorreta] (passo 4)

1.1 Sistema informa que a password está incorreta

1.2 Sistema apresenta opção de recuperar conta

1.3 O utilizador não entra na conta

Fluxo-Exceção 2: [Email não se encontra registado] (passo 4)

2.1 Sistema informa que o respetivo email não se encontra registado na plataforma

2.2 O utilizador não entra na conta

Use Case: Inserir artigo para aluguer/compra**Descrição:** Utilizador deseja registar um novo artigo**Pré-condição:** Utilizador tem de estar com a sessão iniciada**Pós-condição:** Artigo é inserido na plataforma**Fluxo-Normal:**

1. Utilizador clica no botão Inserir novo artigo
2. Sistema remete-o para uma nova interface com campos para preencher sobre o artigo
3. Utilizador preenche esses mesmos campos e submete-os
4. Sistema valida dados
5. Novo artigo é adicionado ao sistema (associado ao respetivo dono)

Fluxo-Exceção 1: [Campos obrigatórios inválidos] (passo 4)

- 1.1 Sistema informa qual(ais) o(s) campo(s) que não estão corretamente preenchidos (incluindo campos vazios)
- 1.2 Artigo não é adicionado à plataforma

Use Case: Alugar um artigo**Descrição:** Utilizador aluga um artigo**Pré-condição:** Utilizador tem de ser do tipo single e selecionar previamente o artigo que pretende alugar**Pós-condição:** Utilizador alugaefetivamente o artigo **Fluxo-Normal:**

1. Utilizador seleciona as datas em que pretende fazer o aluguer
2. Utilizador escolhe o número de artigos que pretende alugar
3. Utilizador clica no botão Alugar
4. Sistema apresenta interface com campos por preencher
5. Utilizador deseja utilizar a conta bancária associada à sua conta
6. Sistema valida dados
7. Pedido de aluguer é efetuado
8. Proprietário do artigo em questão aceita o pedido
9. Aluguer é efetuado

Fluxo-Exceção 1: [Datas selecionadas não estão disponíveis] (passo 3)

- 1.1 As datas selecionadas não estão disponíveis
- 1.2 Aluguer não é efetuado

Fluxo-Exceção 2: [Números de artigos não disponível] (passo 2)

- 2.1 A quantidade de artigos selecionado não se encontra disponível
- 2.2 Aluguer não efetuado

Fluxo-Exceção 3: [Campos obrigatórios inválidos] (passo 6)

- 3.1 Sistema informa que existem dados por preencher ou que não são válidos
- 3.2 Aluguer não é efetuado

Fluxo-Exceção 4: [Proprietário não aceita o pedido] (passo 8)

- 4.1 Proprietário não aceita o pedido
- 4.2 Aluguer não é efetuado

Fluxo-Alternativo 1: [Utilizador deseja pagar com uma conta bancária diferente] (passo 5)

- 1.1 Utilizador deseja utilizar a conta bancária diferente da associada à sua conta
- 1.2 Regressa a 6

Use Case: *Comprar um artigo*

Descrição Utilizador compra um artigo

Pré-condição: Utilizador tem de ser do tipo single e selecionar previamente o artigo que pretende comprar

Pós-condição: Utilizador compra efetivamente o artigo

Fluxo-Normal:

1. Utilizador seleciona o número de artigos que deseja comprar
2. Utilizador clica no botão Comprar
3. Sistema apresenta interface com campos por preencher
4. Utilizador deseja utilizar a conta bancária associada à sua conta
5. Sistema valida dados
6. Pedido de compra é efetuado

7. Proprietário do artigo em questão aceita o pedido
8. Compra é efetuada

Fluxo-Exceção 1: [Números de artigos não disponível] (passo1)

- 1.1 A quantidade de artigos selecionado não se encontra disponível
- 1.2 Aluguer não efetuado

Fluxo-Exceção 2: [Campos obrigatórios inválidos] (passo 4)

- 2.1 Sistema informa que existem dados por preencher ou que não são válidos
- 2.2 Compra não é efetuada

Fluxo-Exceção 3: [Proprietário não aceita o pedido] (passo 6)

- 3.1 Proprietário não aceita o pedido
- 3.2 Compra não é efetuada

Fluxo-Alternativo 1: [Utilizador deseja pagar com uma conta bancária diferente] (passo 3)

- 1.1 Utilizador deseja utilizar a conta bancária diferente da associada à sua conta
- 1.2 Regressa a 4

Use Case: *Pesquisar por um artigo*

Descrição: Utilizador pesquisa por um artigo

Pré-condição: Aceder ao site

Pós-condição: Utilizador encontra o artigo

Fluxo-Normal:

1. Utilizador visualiza os artigos existentes na página inicial da plataforma online
2. Utilizador encontra o artigo que procurava

Fluxo-Alternativo 1: [Pesquisar artigo por categoria] (passo 1)

- 1.1 Utilizador escolhe e seleciona uma categoria associada ao artigo que procura
- 1.2 Regressa a 2

Fluxo-Alternativo 2: [Pesquisar artigo por etiquetas] (passo 1)

- 2.1 Utilizador procura artigo através do motor de busca
- 2.2 Regressa a 2

Fluxo-Exceção 1: [Utilizador não consegue encontrar o artigo em questão] (passo 2)

- 1.1 Utilizador não encontra o artigo

Use Case: *Eliminar um artigo*

Descrição: Utilizador deseja eliminar um artigo da sua conta

Pré-condição: Estar com sessão iniciada

Pós-condição: Artigo é eliminado

Fluxo-Normal:

1. Utilizador vai ao seu perfil e seleciona o artigo que deseja eliminar
2. Clica no botão Eliminar
3. Artigo é eliminado do sistema

- 1.1 Utilizador não possui nenhum artigo para aluguer/compra

- 1.2 Nenhum artigo é eliminado

Use Case: *Bloquear contas*

Descrição: Administrador gere denúncias de contas

Pré-condição: Administrador tem de estar autenticado

Pós-condição: Conta é bloqueada

Fluxo-Normal:

1. Administrador entra na lista de denúncias
2. Seleciona denúncia
3. Utilizador que está a ser denunciado tem 5 denúncias
4. Administrador bloqueia a conta

Fluxo-Exceção 1: [Utilizador tem menos de 5 denúncias] (passo 3)

- 1.1 Utilizador que está a ser denunciado tem menos de 5 denúncias

- 1.2 Administrador manda uma mensagem de aviso para o respetivo utilizador

- 1.3 Administrador não bloqueia a conta

Use Case: *Fazer denúncia*

Descrição: Utilizador faz denúncia de outro utilizador

Pré-condição: Utilizador tem de estar autenticado

Pós-condição: Denúncia é feita

Fluxo-Normal:

1. Utilizador entra no perfil do utilizador a quem deseja fazer uma denuncia contra e clica em denunciar
2. Sistema verifica que houve interação entre os dois utilizadores
3. Sistema verifica que é a primeira denúncia que faz contra o respetivo utilizador
4. Sistema apresenta interface com um breve questionário sobre a denúncia
5. Sistema valida dados
6. Denúncia é realizada

Fluxo-Exceção 1: [não houve interação entre os dois utilizadores] (passo 2)

- 1.1 Sistema verifica que não houve nenhuma interação entre ambos os utilizadores
- 1.2 Denuncia não é realizada

Fluxo-Exceção 2: [já foi realizada uma denúncia] (passo 3)

- 2.1 Sistema verifica que não é a primeira denúncia que faz contra o respetivo utilizador
- 2.2 Denúncia não é realizada

Fluxo-Exceção 3: [Dados não são válidos] (passo 5)

- 3.1 Sistema não valida dados (dados inválidos ou dados obrigatórios em branco)
- 3.2 Denúncia não é realizada

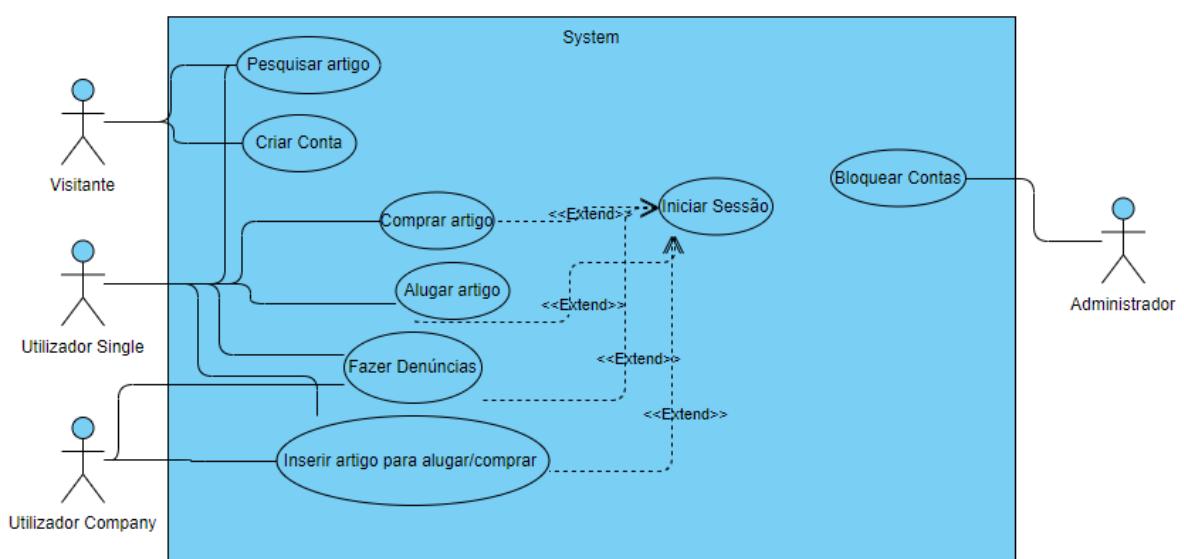


Figura 3: Diagrama de Use Cases

2.4 MOCKUPS

De maneira a começar a estruturar o front-end do website foram desenvolvidos mockups para permitir um melhor encaminhamento durante a fase de implementação. Para tal foi utilizada a ferramenta do *Adobe, Adobe XD*.

Primeiramente foi desenvolvida a estimativa de uma página inicial onde são apresentados vários produtos, nomeadamente organizados por categorias (como por exemplo *Produtos mais Vendidos*) e imagens de publicidade.

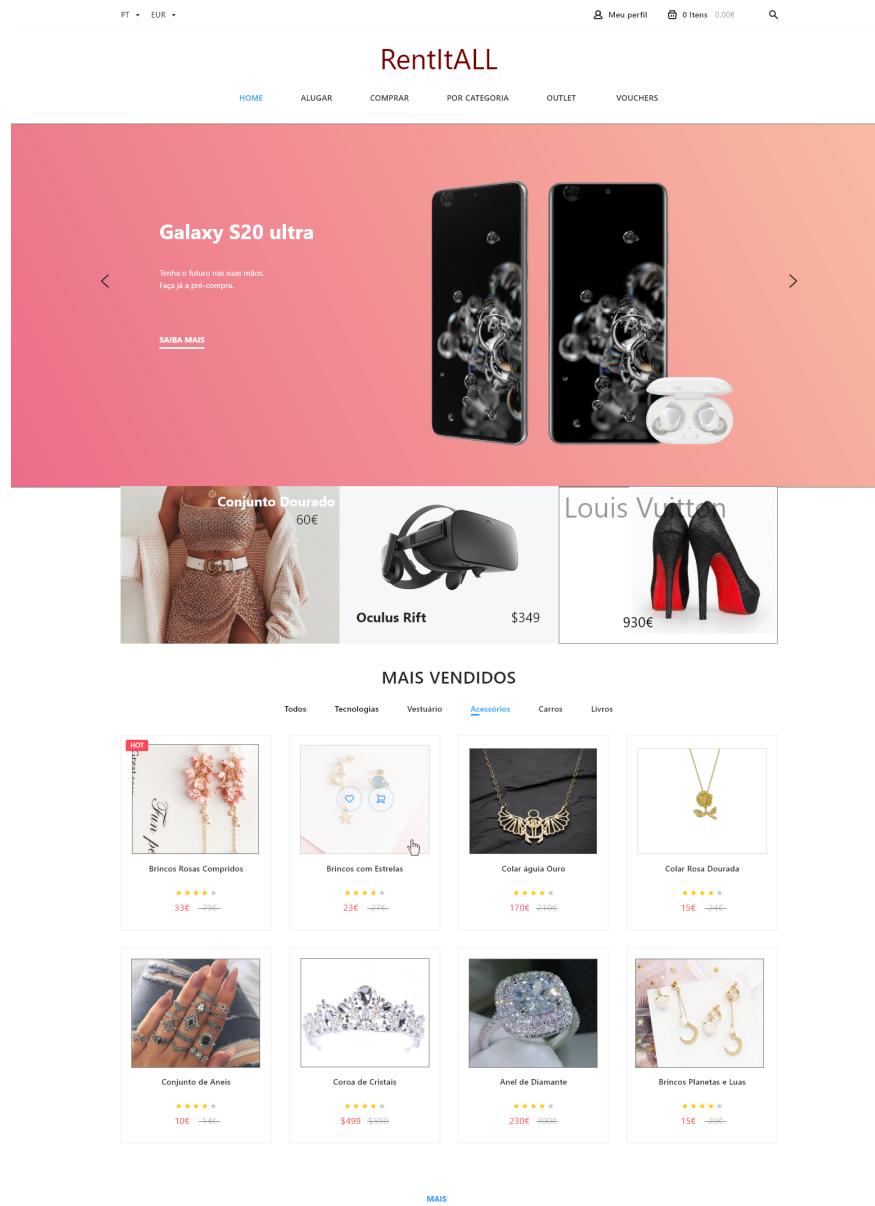


Figura 4: Mockup da página inicial

Foi desenhado também um exemplo de um *footer* que mais tarde seria implementado de maneira mais simples:

Como Funciona

Através de Portes

Placeholder text: Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor minim veniam, quis nostrud reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

Pessoalmente

Placeholder text: Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor minim veniam, quis nostrud reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

Por comunicação

Placeholder text: Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor minim veniam, quis nostrud reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

Search query...

Search

RentItALL

Recente mas inovador.

[Sobre Nós](#) [Informação](#) [Política de Privacidade](#) [Termos e Condições](#)

[Sobre Nós](#) [Informação](#) [Política de Privacidade](#) [Termos e Condições](#)

[Sobre Nós](#) [Informação](#) [Política de Privacidade](#) [Termos e Condições](#)

[Sobre Nós](#) [Informação](#) [Política de Privacidade](#) [Termos e Condições](#)

Follow Us

[f](#) [t](#)

Contacte-nos:

My Company , 4578 Marmora Road, Glasgow
D04 89GR
Call us now: 0123-456-789
Email: support@whatever.com

© 2020 Projeto de LI4 by Group1

Figura 5: Mockup do footer

Foi considerado que seria atribuído um *voucher* de desconto aquando do registo no website, por isso foi desenhado o mockup de "publicidade" para o mesmo que seria visível quando não se está com a sessão iniciada.



Figura 6: Mockup da publicidade ao *voucher*

Quanto à visualização dos detalhes de um determinado artigo, foi desenvolvido um mockup que apresenta as fotos do respectivo artigo, o seu preço, escolhas e características que essa tenha disponível, quantidade e os seus detalhes. Neste mesmo mockup está presente uma área de produtos semelhantes que possam interessar ao comprador.

PT ▾ EUR ▾

RentItALL

HOME ALUGAR COMPRAR POR CATEGORIA OUTLET VOUCHERS

Home / Vestuário / Conjunto Dourado

Conjunto Dourado

★★★★★ * 3 comentários Partilhar opinião

60€ 122€

Disponibilidade: Em Stock

Categoria: Vestuário

Portes Grátis

Tipo: Compra

Selecionar uma cor: (radio buttons)

Tamanho: XS

- + Adicionar ao carrinho

f Partilhar no Facebook t Partilhar no Twitter

Informação do Produto Comentários 3 Mais

Nunc facilisis sagittis ullamcorper. Proin lectus ipsum, gravida et mattis vulputate, tristique ut lectus. Sed et lorem nunc. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae: Aenean eleifend laoreet congue. Vivamus adipiscing nisi ut dolor dignissim semper. Nulla luctus malesuada tincidunt. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Integer enim purus, posuere at ultricies eu, placerat a felis. Suspensisse aliquet urna pretium eros convallis interdum. Quisque in arcu id dui vulputate mollis eget non arcu. Aenean et nulla purus. Mauris vel tellus non nunc mattis lobortis.

Nunc facilisis sagittis ullamcorper. Proin lectus ipsum, gravida et mattis vulputate, tristique ut lectus. Sed et lorem nunc. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae.

BEST SELLER

Apple MacBook Pro

★★★★★ \$499 4500+

GoPro Hero 6

Lorem ipsum dolor sit amet, consectetur adipiscing elit, labor

\$299

PRODUTOS RELACIONADOS

HOT

Brincos Rosas Compridos

★★★★★ 33€ 79€

Brincos com Estrelas

★★★★★ 23€ 47€

Colar águia Ouro

★★★★★ 170€ 340€

Colar Rosa Dourada

★★★★★ 15€ 24€

Figura 7: Mockup da view dos detalhes de um produto

Foi também criado um mockup para o carrinho de compras que lista os produtos que foram inseridos no carrinho, o preço total e a possibilidade de seguir para *checkout*. Esta mesma view demonstra a possibilidade de utilização de um Código de desconto aplicável à compra.

The mockup shows a shopping cart interface for the website RentItALL. At the top, there are language and currency dropdowns (PT, EUR), a user profile link ('Meu perfil'), a cart icon showing 2 items worth 0.00€, and a search bar. The main header is 'RentItALL' in red. Below it is a navigation menu with links: HOME, ALUGAR, COMPRAR, POR CATEGORIA, OUTLET, and VOUCHERS. A breadcrumb trail indicates the current location: Home / Vestuário / Conjunto Dourado. The main content area displays a table of products in the cart:

PRODUTOS	PREÇO	QTY
	60€	- 2 +
	\$998	- 2 +

Below the table, there is a section for applying a voucher code:

Código Voucher	Usar
----------------	-------------

At the bottom, the total amount is displayed as **TOTAL 1558€**, and a large blue button labeled **Check out** is present.

Figura 8: Mockup do carrinho de compras

Quanto à listagem de produtos por procura ou por categoria, foi desenhado um mockup de listagem que permite escolha de características.

PT ▾ EUR ▾

Meu perfil 0 Itens 0,00€

RentItALL

HOME ALUGAR COMPRAR POR CATEGORIA OUTLET VOUCHERS

Home / Acessórios / Todo

ACESSÓRIOS

- Tudo
- Coroas 6
- Brincos 150
- Pulseiras 166
- Colares 122
- Anéis 90
- Outros 179

PREÇO

Range: 9.99€ - 2578.99€

COR

● ● ● ● ● ●

MARCA

- Gucci 99
- Michael Kors 99
- UNBRANDED 99
- Outras 99

MAIS

Galaxy S20 ultra
Tenha o futuro nas suas mãos.
Faça já a pré-compra.

[SABER MAIS](#)

721 Itens Ordenar por: Nome Mostrar 8

Brincos Rosas Compridos
★★★★★
33€ ~~70€~~

Brincos com Estrelas
★★★★★
23€ ~~47€~~

Colar águia Ouro
★★★★★
170€ ~~210€~~

Conjunto de Aneis
★★★★★
10€ ~~14€~~

Coroa de Cristais
★★★★★
\$499 ~~\$500~~

Anel de Diamante
★★★★★
230€ ~~300€~~

Brincos Planetas e Luas
★★★★★
15€ ~~20€~~

Colar Rosa Dourada
★★★★★
15€ ~~24€~~

1 2 3 4 5

Figura 9: Mockup do carrinho de compras

Para a situação de registo de contas existem duas variantes, o registo de um utilizador *single* e um utilizador *company*. Para estes foram criadas dois mockups a demonstrar o formulário de registo e as suas diferenças.

O formulário para o registo de um utilizador *single* é intitulado "FORMULÁRIO EMPRESAS" e "FORMULÁRIO". Ele contém campos para Nome, Email, Morada, Código Postal e Informação Bancária (Número e Código). Um botão "CRIAR" está posicionado no lado esquerdo.

Nome		Email		Morada		Código Postal		Informação Bancária	
Nome		Email		Nome		Nome		Número	
Email		Email		Email		Código Postal		Número	
Morada		Morada		Morada		Código Postal		Número	
Morada		Morada		Morada		Morada		Número	
Nome		Nome		Nome		Nome		Número	
Código Postal		Código Postal		Código Postal		Código Postal		Número	
Número		Número		Número		Número		Número	
Informação Bancária		Informação Bancária		Informação Bancária		Informação Bancária		Validade	
Número		Número		Número		Número		Validade	
Validade		Validade		Validade		Validade			
CRIAR				CRIAR					

Figura 10: Mockup da página de registo de um utilizador *single*

Figura 11: Mockup da página de registo de um utilizador *company*

3

ESTRUTURA DA APLICAÇÃO

3.1 PROPOSTA SUGERIDA - SOLUÇÕES

Tal como explicado no capítulo anterior o nosso sistema vai ter quatro tipos de atores: **Visitante, Utilizador Single, Utilizador Company e Adminitrador.**

O **visitante** tem funcionalidades limitadas em relação aos dois tipos de utilizadores. A este apenas lhe é permitido visualizar e pesquisar produtos, assim como a possibilidade de criar uma conta. É lhe também permitido aceder ás páginas informativas, tal como a secção *Sobre Nós*, onde são divulgadas informações sobre a equipa por detrás da aplicação, e a secção de *Contactos*.

O **utilizador** do tipo *Single*, além das funcionalidades do visitante, pode comprar/alugar os produtos do sistema, assim como pode criar os seus próprios produtos. Posto isto, na sua *navbar* irão aparecer mais seções como o *Histórico*, onde é possível visualizar todo o seu histórico de compras e alugueres, *Notificações*, que é a secção referente ao aceitar/recusar um pedido de aluguer, *Meus Produtos*, secção responsável pelo gerenciamento dos próprios pordutos e o *Carrinho de Compras*.

O **utilizador** do tipo *Company*, visto que este se refere a um tipo de contas para empresas, apenas lhe são oferecidas funcionalidades referentes à venda/alugueres dos seus próprios produtos. Deste modo, as páginas de visualização e procura de produtos não estão disponíveis quando este tem sessão iniciada. Ao invés disso, na sua página inicial, o utilizador Company terá um conjunto de dados estatísticos sobre a sua conta, como por exemplo a visualização gráfica dos produtos vendidos e alugados, os produtos mais concorridos, o número e informação sobre os artigos, denúncias, comentários e *vouchers*. Posto isto, este vai ter apenas na sua *navbar* as secções : *Catálogo*, que funciona do mesmo modo que a secção *Meus Produtos* no utilizador single, e *Histórico*.

Por último temos então o **Administrador**. Este tal como o Utilizador *Company*, terá na sua página inicial um conjunto de dados estatísticos, mas desta vez referente a todo o site. Na sua *navbar* terá também como secções duas páginas relativas à gestão de utilizadores e denúncias. A secção *Denúncias* aceita/recusa pedidos de denúncias feitas pelos utilizadores sobre determinados artigos, uma vez que podem haver denúncias fraudulentas, para determinar se são denúncias válidas ou não, ou seja, se houve justificação para as denúncias. A secção *Ver Utilizadores* exibe uma lista com todos os utilizadores assim como o número de denúncias já aceites, tendo como opção enviar um aviso via *Email* ou até mesmo bloquear uma conta quando este valor começar a ser alarmante.

3.1.1 Arquitetura do Sistema

Para a concretização da nossa aplicação Web decidimos utilizar o *Microsoft Visual Studio*, sendo este destinado ao desenvolvimento de software especialmente dedicado ao *.NET Framework*.

Como referido anteriormente a linguagem usada para o desenvolvimento do projeto foi **C#**, utilizando o template *ASP .NET Web Application*, com o modelo *MVC*.

O *MVC* é uma arquitetura de software que separa a lógica da interface do usuário. Isto é conseguido através da separação da aplicação em três partes *Model*, *View* e *Controller*. De forma sucinta, o *Model* representa o comportamento lógico dos dados na aplicação, a *View* fornece a interface de usuário da aplicação e o *Controller* define o comportamento da aplicação.

MODELS

As classes presentes no *Model* são criadas através da Base de Dados, tal como é explicado no capítulo 4. Essencialmente estas correspondem a cada uma das entidades criadas na BD. Porém, ao longo do desenvolvimento do nosso projeto surgiu a necessidade de criar mais três classes, *AluguerInfo*, *VendaInfo*, *SpecialIndexes*. Estas foram criadas com vista a auxiliar as *Views*, ou seja, estas permitem agregar dados contidos em mais do que um model para posteriormente serem acedidos numa determinada view.

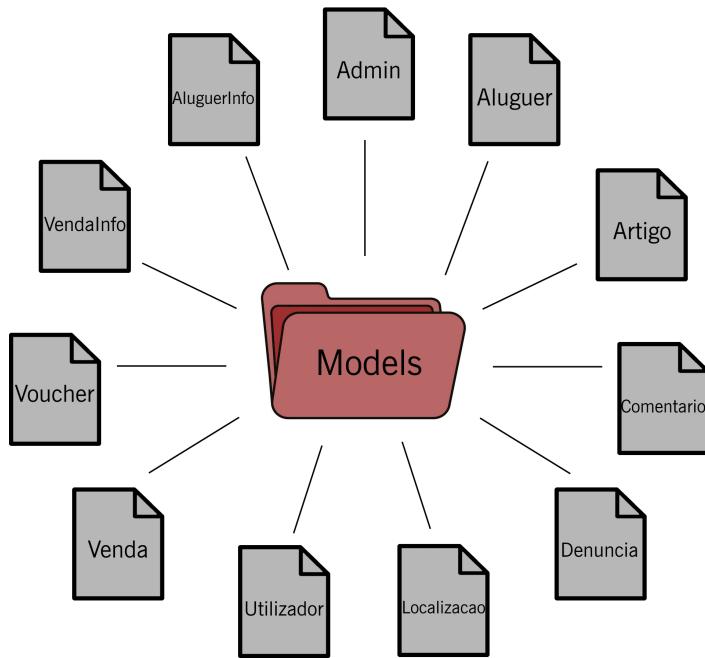


Figura 12: Esquema dos Models do projeto

CONTROLLERS

O *Controller* é responsável por interligar a *View* e o *Model*, dessa forma tornou-se fundamental criar várias classes controllers de forma a organizar e agregar funcionalidades e ações. Para isso temos uma classe para cada ator do nosso sistema, ou seja, o utilizador do tipo *single*, o utilizador do tipo *company* e o administrador, sendo as suas classes criadas respetivamente a *UtilizadorController*, *CompanyController* e *AdminController*. Temos também uma *HomeController* que é quem trata das ações a realizar quando um cliente da nossa aplicação não é ou não está logado como nenhum dos três atores anteriormente referidos. Por último temos a *ContasController*, esta é responsável por gerir as ações relativas às contas, como por exemplo, o método de *Login* e o de *Recuperação de Password* abordadas no capítulo 5.

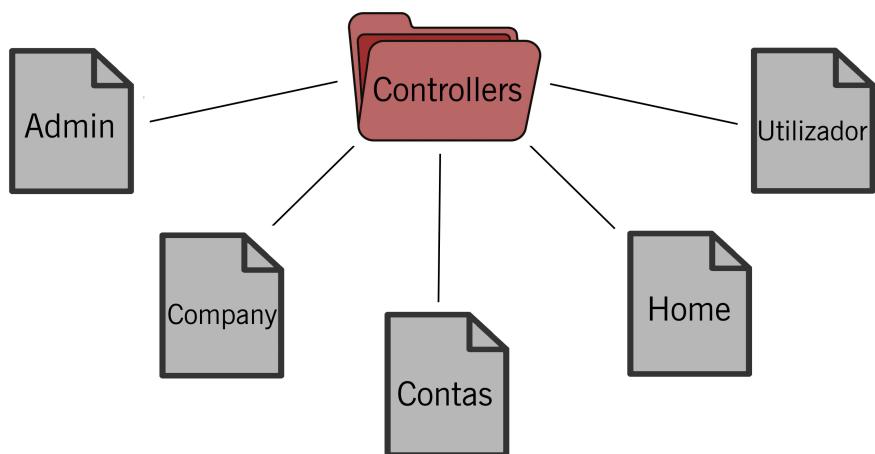


Figura 13: Esquema dos Controllers do projeto

VIEWS

As nossas *Views* estão divididas da seguinte forma:

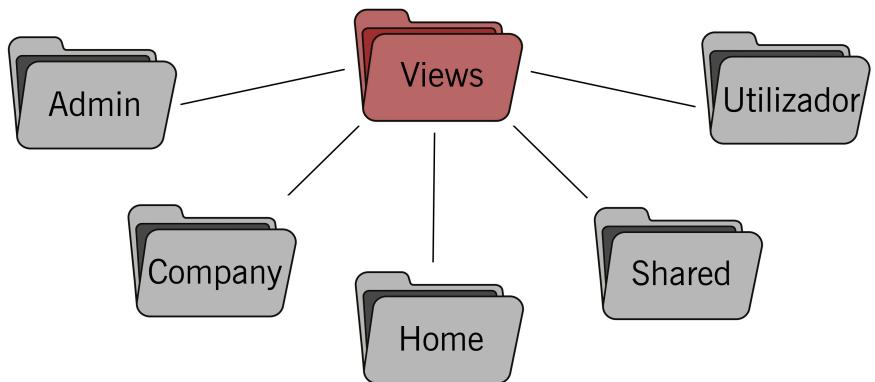


Figura 14: Esquema das Views do projeto

Cada uma das diferentes subpastas contém as *Views* referentes aos diferentes modos de *login*, excetuando as *views Shared*, que contêm CSS e *scripts* partilhados por *Views* do mesmo modo.

4

BASE DE DADOS

Após terem sido definidos os modelos que sustentarão a implementação da nossa aplicação foi necessário planificar a base de dados que apoiará o seu funcionamento. Tendo em consideração todos os requisitos definidos anteriormente, foi possível produzir modelos que mostram todas as entidades, atributos e dependências do sistema

4.1 MODELO CONCEITUAL

Primeiramente foi desenvolvido o modelo Conceitual. Esta modelação consiste no processo de construção de um modelo que é capaz de relacionar informação independentemente da implementação. A sua elaboração permitiu organizar todos os dados necessários para a aplicação .

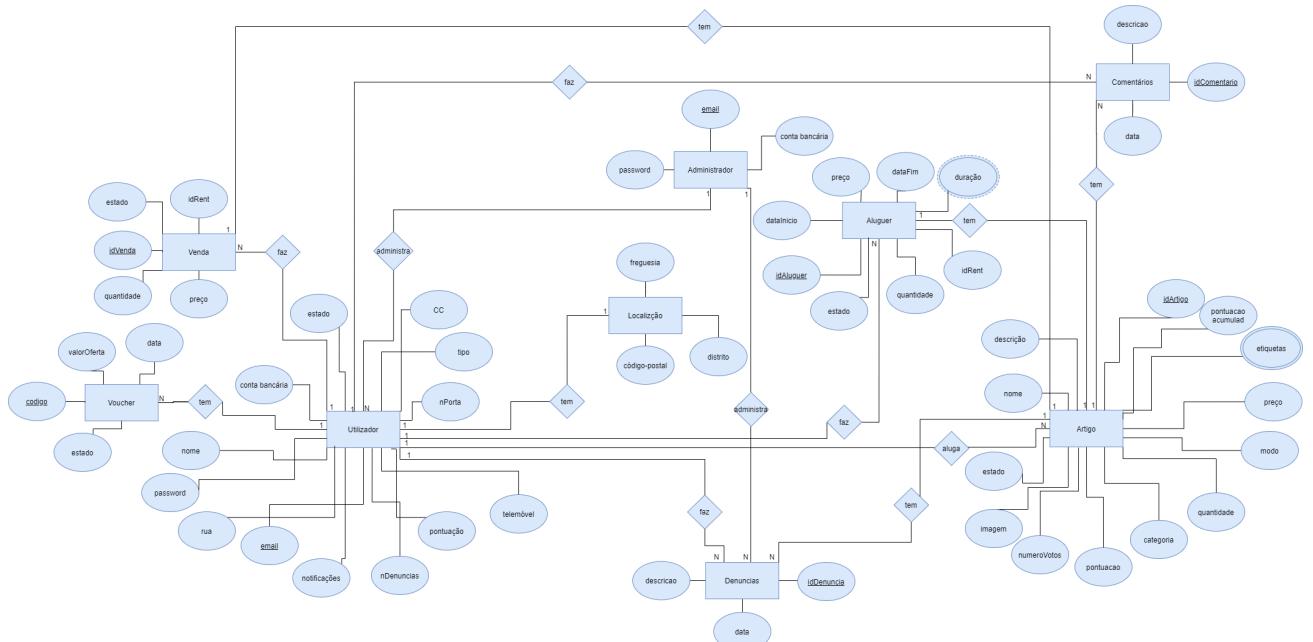


Figura 15: Modelo lógico da Base de Dados

4.2 MODELO LÓGICO

Com base no Modelo Conceptual apresentado anteriormente, foi desenvolvido o Modelo Lógico. Este foi criado com o software *SSMS* de maneira a ser possível mais tarde ligar a base de Dados à nossa aplicação e ao *Azure*.

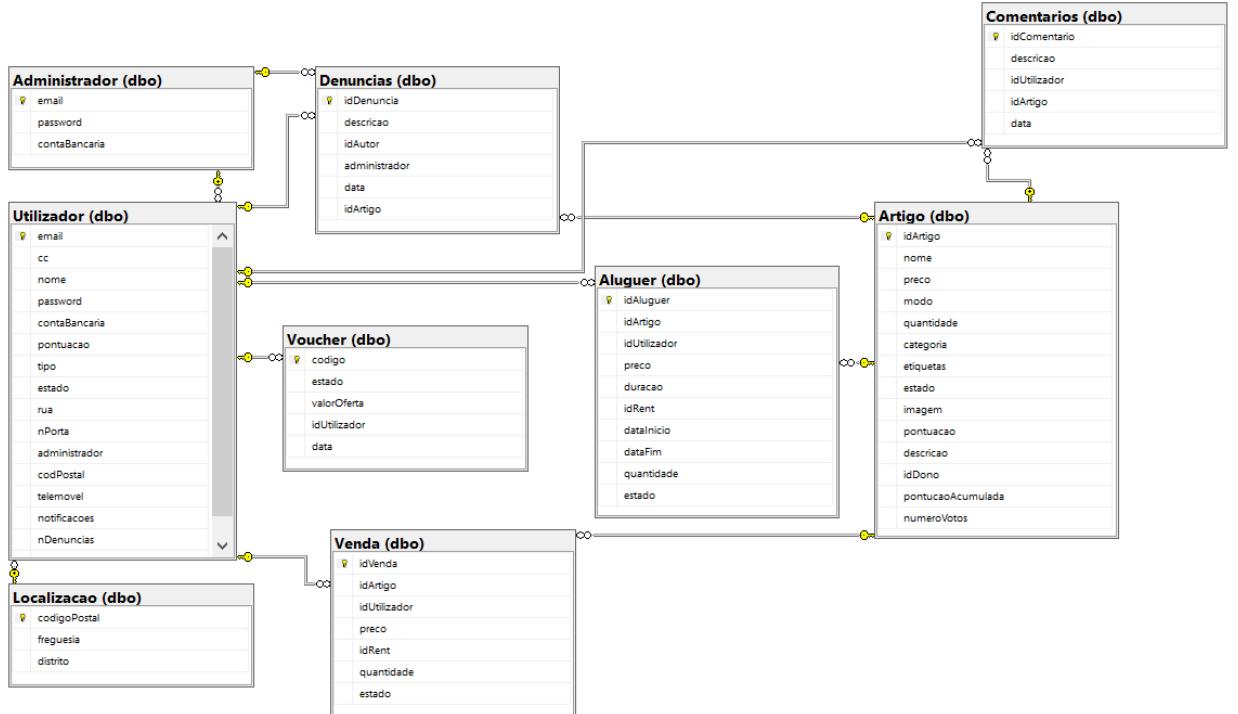


Figura 16: Modelo lógico da Base de Dados

Com o modelo lógico concluído e de maneira a começar a implementação da aplicação web, a base de dados foi conectada diretamente com o *Visual Studio*, software onde foi desenvolvido o nosso sistema, e esta conexão permitiu criar logo de início os principais *models* e todas as dependências de dados que eram necessárias.

Mais tarde, para permitir que a base de dados fosse escalável e aderida por todos, foi necessária a migração da mesma para a plataforma *Azure*.

4.3 DICIONÁRIO DE DADOS

De maneira a ser possível verificar a informação que é armazenada em cada atributo foi elaborado o seguinte dicionário de dados. Este explicita cada um dos atributos com o tipo de dados correspondentes e ainda uma breve descrição sobre os mesmos.

Tabela 10: Dicionário de Dados

Entidade	Atributos	Tipo de Dados	Descrição
Administrador	email	Variável até 45 caracteres	Email para permitir acesso ao site
	password	Variável até 45 caracteres	Password para permitir acesso
	contaBancaria	Inteiro longo	Conta bancária do Administrador
Utilizador	email	Variável até 45 caracteres	Email para permitir acesso ao site
	cc	Inteiro	Cartão de cidadão do Utilizador
	nome	Variável até 45 caracteres	Nome do Utilizador
	password	Variável até 45 caracteres	Password para permitir acesso
	contaBancaria	Inteiro longo	Conta bancária do Utilizador
	tipo	Variável até 45 caracteres	Tipo (se é single ou company)
	estado	Inteiro	Estado (Ativo ou bloqueado)
	rua	Variável até 45 caracteres	Rua do Utilizador
	nPorta	Inteiro	Número da porta da rua
	administrador	Variável até 45 caracteres	Email do administrador do site
	codPostal	Variável até 45 caracteres	Código Postal
	telemovel	Inteiro	Número de telemóvel
	notificacoes	Inteiro	Número de notificações
Localizacao	nDenuncias	Inteiro	Número de denúncias
	codigoPostal	Variável até 45 caracteres	Código Postal
	freguesia	Variável até 45 caracteres	Freguesia
Denuncias	distrito	Variável até 45 caracteres	Distrito
	idDenuncia	Inteiro	Id único de cada denúncia
	descricao	Variável até 2000 caracteres	Descrição da denúncia
	idAutor	Variável até 45 caracteres	Id do autor da denúncia
	administrador	Variável até 45 caracteres	Email do administrador
	data	Date	Data em que foi realizada
Voucher	idArtigo	Inteiro	Artigo que possui a denúncia
	codigo	Variável até 45 caracteres	Código do Voucher
	estado	Inteiro	Estado (se já foi utilizado ou não)
	valorOferta	Float	Valor de oferta do Voucher
	idUtilizador	Variável até 45 caracteres	Utilizador que usufrui do Voucher
Venda	data	Date	Data de expiração
	idVenda	Inteiro	Id único de cada venda
	idArtigo	Inteiro	Id do artigo a ser vendido
	idUtilizador	Variável até 45 caracteres	Email do dono do artigo
	preco	Float	Preço do artigo
	idRent	Variável até 45 caracteres	Email do utilizador que vai comprar
	quantidade	Inteiro	Quantidade de artigo que se vende
Estado	estado	Inteiro	Estado (se disponível ou não)

Aluguer	idAluguer	Inteiro	Id único de cada aluguer
	idArtigo	Inteiro	Id do artigo a alugar
	idUtilizador	Variável até 45 caracteres	Email do dono do artigo
	preco	Float	Preço de aluguer por dia
	duracao	Inteiro	Duração do aluguer
	idRent	Variável até 45 caracteres	Email do utilizador que vai alugar
	dataInicio	Date	Data de início do aluguer
	dataFim	Date	Data de fim do aluguer
	quantidade	Inteiro	Quantidade a alugar
	estado	Inteiro	Estado (se disponível ou não)
Artigo	idArtigo	Inteiro	Id único de cada artigo
	nome	Variável até 45 caracteres	Nome do Artigo
	preco	Float	Preço do artigo
	modo	Variável até 45 caracteres	Modo (aluguer ou venda)
	quantidade	Inteiro	Quantidade do artigo
	categoria	Variável até 45 caracteres	Categoria do artigo
	etiquetas	Variável até 45 caracteres	Etiquetas que definem o artigo
	estado	Inteiro	Estado (se visível ou não)
	imagem	Variável até 1000 caracteres	Imagens do artigo
	pontuacao	Float	Pontuação
	descricao	Variável até 1000 caracteres	Descrição do produto
	idDono	Variável até 45 caracteres	Email do dono do artigo
Comentarios	pontuacaoAcumulada	Float	Soma das pontuações
	numeroVotos	Inteiro	Número de pessoas que votaram
	idComentario	Inteiro	Id único de cada Comentário
	descricao	Variável até 1000 caracteres	Descrição do comentário
	idUtilizador	Variável até 45 caracteres	Utilizador que faz comentário
Artigo	idArtigo	Inteiro	Artigo que possui o comentário
	data	Date	Data do comentário

4.4 POVOAMENTO

Numa fase inicial foi também necessária a criação de um pequeno povoamento para ser possível a realização de testes enquanto decorria o desenvolvimento da aplicação web. Este povoamento foi criado a partir de uma série de queries Sql e consistiu na criação de algumas contas dos diferentes tipos (administrador, utilizador single e utilizador company) bem como na inserção de vários artigos, denúncias, comentários e *vouchers* e ainda vendas e alugueres já realizados.

5

DESENVOLVIMENTO

5.1 DECISÕES

Foi decidido pelo grupo que seria usado, como proposto, **C# no Backend**. Já para o **Frontend** recorremos à implementação de código **HTML**, **CSS**, *scripts* em **JavaScript** e alguma implementação básica de **PHP**. De seguida indicaremos algumas das vertentes mais relevantes e o enveradar para a sua implementação.

5.2 IMPLEMENTAÇÃO

Concluída a fase de especificação, segue-se a fase de implementação das funcionalidades com base nos requisitos propostos e na modelação anteriormente apresentada. De seguida são descritas todas as funcionalidades da aplicação *web* bem como a maneira de como foram implementadas no sistema.

5.2.1 *Login*

De maneira a que os utilizadores tenham acesso a determinadas funções que um convidado não pode ter, como alugar, vender e comprar produtos, foi necessária a criação do método de *login*. Este método consiste essencialmente em verificar se uma determinada conta existe na base de dados e se as suas credenciais (email e password) estão corretas. Para além desta verificação, este método também tem em conta os utilizadores que foram bloqueados devido a um número elevado de denúncias sendo estes impedidos de aceder ao site. Também é no *login* que são iniciadas as *cookies* de autenticação bem como a verificação da encriptação das passwords.

Após ser feito o *login* são apresentadas as views respetivas a cada tipo de utilizador (administrador, company ou single).

RESULTADOS



Figura 17: Parcial da página de Login

5.2.2 Adicionar Utilizador

Para que um Utilizador usufrua da maioria das funcionalidades da aplicação é necessário a criação de uma conta. Consequentemente, é necessário o preenchimento de um formulário com os dados do futuro utilizador.

No método correspondente a adicionar um cliente, o que ocorre é a inserção dos dados fornecidos pelo mesmo na base de dados.

RESULTADOS

The screenshot shows a registration form titled "Register". It contains 14 input fields, each with a placeholder label:

- Email
- CC
- Nome
- Password
- Conta Bancária
- Tipo (single/company)
- Telemovel
- Rua
- Número da porta
- Código Postal
- Confirme Código Postal
- Freguesia
- Distrito

At the bottom of the form is a large, dark blue rectangular button with the word "REGISTAR" in white capital letters.

Figura 18: View Adicionar Utilizador

5.2.3 *Encriptação de Passwords*

A encriptação é uma ferramenta essencial para a segurança dos dados. Esta é um processo de codificação de uma determinada informação, que no nosso caso será das passwords associadas ás diferentes contas, de modo a que só seja acessível a quem tenha o código que permite a descodificação da informação, revertendo-a para a sua forma original.

Para implementar esta funcionalidade no nosso projeto, foi usado o algoritmo de sintetização de mensagen MD5, que é uma função hash criptográfica. Para isso foram desenvolvidas duas funções. Uma que encripta a password inicial, *HashPassword*, outra que dada uma password encriptada e uma password desencriptada verifica se são compatíveis, *VerifyMd5H*.

Estas funções são usadas em todos os métodos que envolvem a password, nomeadamente, no Login, no Registo de contas, na opção de alterar a password e na recuperação da mesma.

RESULTADOS

	email	password
1	afonno.gommes@gmail.com	114021c5bbb64c605e4ff855a2bd7b65
2	catarina@gmail.com	ad38128a6606984904c2bd02b8c1a746
3	catarinaG1231@gmail.com	d6607a0d5fa9ddbc40d551a695a3ddee
4	company1@gmail.com	df655f976f7c9d3263815bd981225cd9
5	company2@gmail.com	d196a28097115067ef73d25b0c0be8
6	conta@gmail.com	1a9c91f6e0310d4f55b7ee7f22c2c9df
7	joana@gmail.com	18f01959ff46071d73905d549cafde20
8	margarida.maia.campos@gmail.com	dcf4acb874cfb973611f27e73795ada8
9	margarida@gmail.com	dcf4acb874cfb973611f27e73795ada8
10	meguiie.campos@gmail.com	114021c5bbb64c605e4ff855a2bd7b65
11	pedro@gmail.com	c6cc8094c2dc07b700ffcc36d64e2138
12	regina.sandra@gmail.com	221182760f5b980c97c7a74a94d57364
13	tani.spam33@gmail.com	916229a8d4ef288e7f4880ad356800aa
14	tania@gmail.com	916229a8d4ef288e7f4880ad356800aa
15	thisismydrive.joana@gmail.com	114021c5bbb64c605e4ff855a2bd7b65

Figura 19: Base de Dados: passwords de utilizadores

5.2.4 Recuperação de Password

Para implementar esta funcionalidade achamos por bem recorrer à API SmtpClient, que permite o envio de emails por parte dos aplicativos usando o SMTP (Simple Mail Transfer Protocol).

A nossa estratégia passou por enviar um email ao utilizador que pretende recuperar a password com um código de 8 bits aleatório. Após o envio, o usuário será redirecionado para uma nova página onde deverá preencher os parâmetros pedidos como o código de recuperação de password, enviado para o email, e a sua nova password. Se tudo for válido então esta ação é executada com sucesso.

RESULTADOS



Figura 20: Email recebido via Gmail

5.2.5 Cookies

As *cookies* implementadas nesta aplicação permitem ao utilizador, caso este não faça *logout*, manter sessão iniciada no *browser* durante um tempo pré-definido de 10 horas. Ao fim do tempo definido, caso o utilizador queira aceder a alguma funcionalidade do site necessita de fazer o *login* novamente. Isto traz segurança ao site uma vez que tenta impedir que outros indivíduos utilizem a conta de um determinado utilizador caso este se tenha esquecido de fazer *logout*.

Para a implementação desta funcionalidade foi necessário a adição de *cookies* na classe *startup.cs* do projeto. Lá são especificadas as principais características das mesmas tais como o nome, o *LoginPath* e o *CookiePath* que indicam onde a *cookie* vai ser iniciada. Também foi necessário acrescentar no método de *login Claims*. Estas são responsáveis por guardar o nome do utilizador autenticado bem como informações sobre as permissões e autorizações do mesmo. Ainda no *login* foi necessária a adição das propriedades de autenticação. É aí que é definido o período de tempo de 10 horas e mais algumas propriedades bem como se é permitido o *refresh* da página.

5.2.6 Pesquisa de Artigos

Dada a natureza da aplicação surgiu a necessidade de agilizar o processo de pesquisa por artigos. Este é feito de várias maneiras tendo em conta as necessidades do site.

O primeiro tipo de pesquisa a ser implementado foi a pesquisa pela *search bar*. Após dar entrada no site é possível escrever na barra de pesquisa do mesmo o que se pretende procurar. O Algoritmo usado consiste na comparação da palavra escrita pelo utilizador com todos os

nomes e etiquetas dos artigos que se encontram na base de dados. Após feita a pesquisa é apresentada uma *view* com todos os artigos que fizeram *match*.

Uma outra forma de pesquisa de artigos é a pesquisa por uma determinada categoria. Como todos os produtos possuem uma categoria, a implementação desta funcionalidade consiste em comparar a categoria escolhida pelo utilizador, na *sidebar* disponibilizada para isso, com a de todos os artigos e apresentar os que possuem essa mesma categoria.

Outra forma de pesquisa de artigos no nosso sistema é por *rankings*, ou seja, a partir dos botões *Maior Classificação*, *Mais Vendidos* e *Novidades* irão aparecer todos os produtos do nosso sistema tal como na página inicial, porém estes vão estar ordenados segundo a opção escolhida.

De forma geral, inicialmente é feita uma *querie* que devolve uma lista com todos os produtos presentes na nossa Base de Dados. No caso da *Maior Classificação*, esta lista vai ser ordenada decrescentemente pelo atributo *classificação* da entidade *Produto*, no *Mais Vendidos* é feita uma iteração para contar o número de vendas/alugueres de cada produto e só posteriormente é feita a ordenação de acordo com este valor calculado. Por último, no caso da opção *Novidades*, os produtos vão ser ordenados consoante o seu ID, uma vez que cada vez que se cria um produto, este inteiro vai ter o seu valor incrementado de forma a que os produtos com ID mais elevado são assim os artigos mais recentemente adicionados á Base de Dados.

RESULTADOS

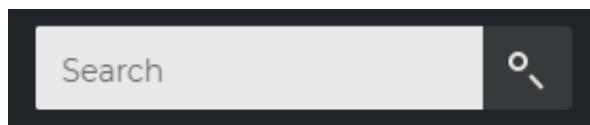


Figura 21: Barra de pesquisa



Figura 22: Categorias

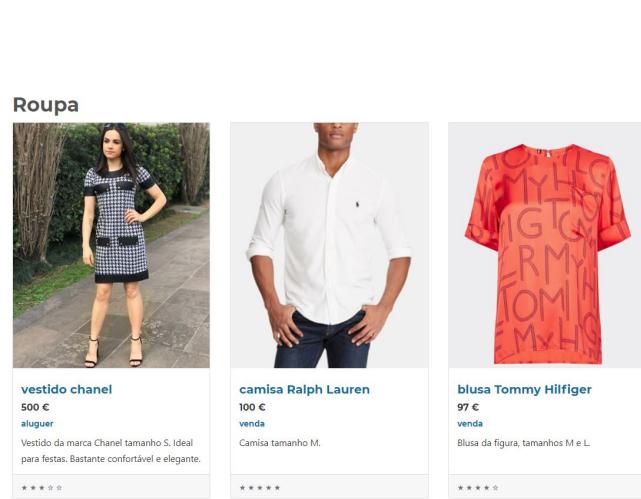


Figura 23: View SearchCategoria

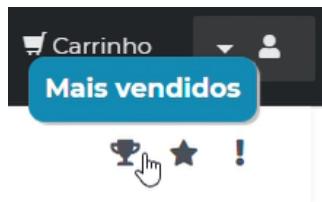


Figura 24: Mais Vendidos

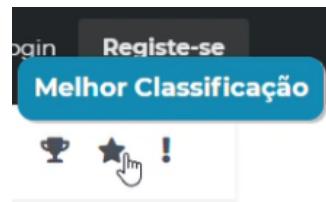


Figura 25: Melhor Classif.

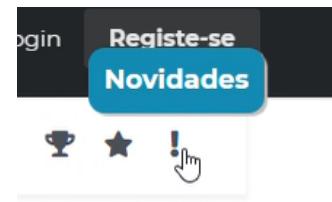


Figura 26: Novidades

5.2.7 Artigos

Na nossa aplicação, dentro da conta, tanto do utilizador do tipo single como no utilizador do tipo company, existe uma secção da navbar, *Meus Produtos*, que permite ter acesso aos produtos que o utilizador coloca à venda/aluguer assim como as suas principais informações.

Nesta página é também possível criar e remover produtos. No caso de criar um novo artigo o utilizador será redirecionado para uma nova página onde aparecerá um formulário por preencher sobre o novo artigo a adicionar. É também permitido o upload de imagens a partir dos ficheiros presentes no computador do utilizador. Posteriormente é criado efectivamente um artigo com esses dados e é adicionado o mesmo à Base de Dados. No caso de remover produto apenas é necessário clicar no botão para esse efeito e o mesmo será removido da Base de Dados.

Para visualização de um *slider* das fotos de um artigo na View Details dos modos de que lhe têm acesso, recorreu-se a um *script* de JavaScript, utilizando também C# e, claro, HTML, com a finalidade de ser possível, no caso de mais de um artigo possuir mais do que uma imagem associada, separar os *paths* das várias imagens correspondentes a esse artigo e apresentá-las num *slider*.

```

@{var fileName = Model.Imagem;
  var idd = Model.IdArtigo;
  string[] words = fileName.Split(" ");
  int i = 1;
  int count = words.Length();

  foreach (string s in words){
    
    if (i == count) break;
    i++;
  }
  string slides = "mySlides-" + idd;
  var sl = slides;
  string scr = "script-" + idd;
  var scri = scr;

  if (words.Length() > 1)
  {
    <button style="(...)" onclick="plusDivs(-1)">&#10094</button> &nbsp;
    <button style="(...)" onclick="plusDivs(1)">&#10095;</button>
  }

  <script name="script-@idd" type="text/javascript">
  var slideIndex = 1;
  showDivs(slideIndex);

  function plusDivs(n,) { showDivs(slideIndex += n); }

  function showDivs(n) {
    var i;
    var x = document.getElementsByClassName("@sl");
    if (n > x.length) { slideIndex = 1 }
    if (n < 1) { slideIndex = x.length }
    for (i = 0; i < x.length; i++) {
      x[i].style.display = "none";
    }
    x[slideIndex - 1].style.display = "block";
  }
  </script>
}

```

Listing 5.1: Implementação do slider das imagens nas views Details

RESULTADOS

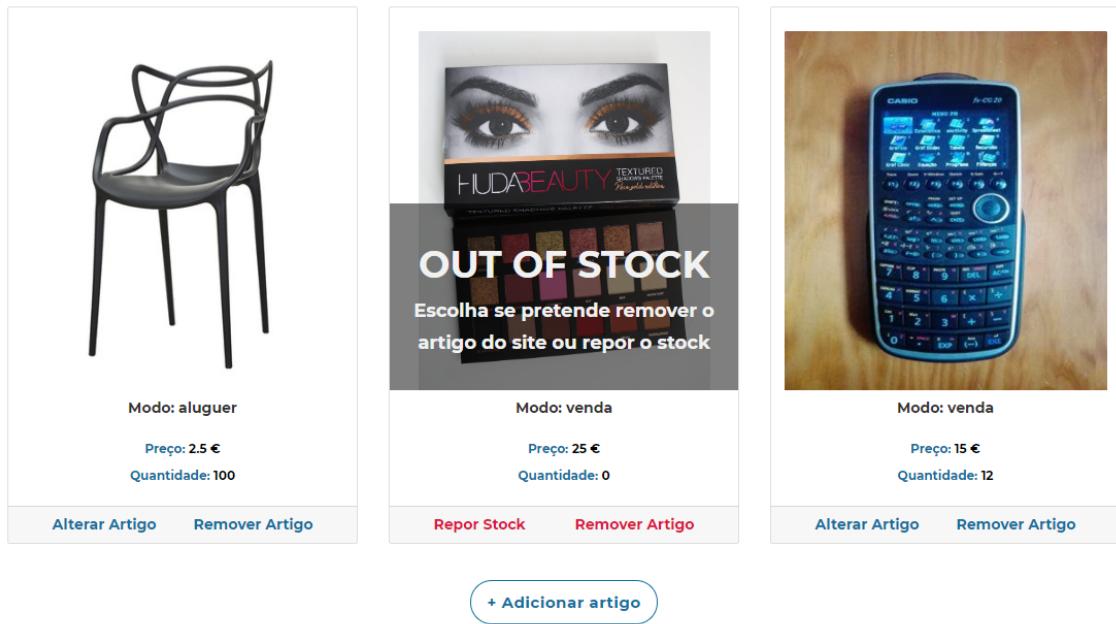


Figura 27: View que permite ver os artigos do Utilizador

Nome

Preço (€)

Modo:

Quantidade

Escolha uma categoria:

Descrição

Etiquetas

Faça upload da(s) imagem(ns)

Nenhum arq...lecionado

[INserir Artigo](#)

Figura 28: Inserir novo Artigo



Figura 29: Exemplo de slider de um artigo com mais do que uma imagem

5.2.8 Alterar Dados

No nossa aplicação é permitido alterar dados sobre duas entidades por parte do usuário. Uma sobre os seus próprios dados, como por exemplo, conta bancária, número de telemóvel, etc, e sobre os seus próprios produtos, como por exemplo, quantidade, descrição, etc.

Para isso o utilizador escolhe qual o parâmetro a alterar, identificando posteriormente o seu novo valor, sendo o mesmo atualizado na Base de Dados do sistema.

RESULTADOS

Tipo de informação	Informação atual	Alterar Dados
Email	joana@gmail.com	Alteração Não Permitida
Password	*****	Alterar
Conta Bancária	11445566187996	Alterar
Telemovel	989898989	Alterar
Código Postal	2900-003	Alterar
Freguesia	Setúbal	Alterar
Rua	Rua Fogos	Alterar
Número da porta	34	Alterar
Distrito	Setúbal	Alterar

Figura 30: Utilizador: Alterar dados

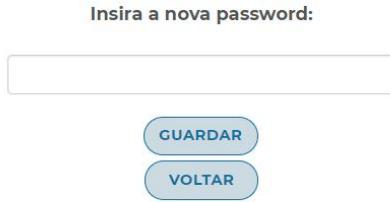


Figura 31: Alterar Password

5.2.9 Carrinho de Compras

Quando um Utilizador pretende adquirir artigos (em modo de venda), pode adicionar um ou mais itens ao carrinho até efetuar o *checkout* e finalizar a compra.

Para esse fim, existe um botão na página de um artigo (view *Details*) que redirecciona para a view *Adicionar Carrinho*, comprovando que o item foi de facto adicionado ao carrinho.

Depois, dentro do Carrinho de Compras (view *VendaInfo*), com hiperlink na *navbar* de um Utilizador *loggado*, é possível ter acesso a todos os artigos que foram inseridos no carrinho. Nesta área é possível remover qualquer artigo desta lista e aumentar ou diminuir a quantidade de um determinado produto até ao número máximo ou mínimo permitido, ou seja, a quantidade existente em stock. Na situação em que é aumentada a quantidade de um produto até este atingir o limite máximo é recebida uma pequena notificação a dar conhecimento que o numero escolhido é o mais alto o possível. Para estas duas interações, o preço total da compra é atualizado.

```
/* Actualiza Quantidade */
function updateQuantity(quantityInput) {
    /* Calcular preco por linha */
    var productRow = $(quantityInput).parent().parent();
    var price = parseFloat(productRow.children('.price').text());
    var quantity = $(quantityInput).val();
    var linePrice = price * quantity;

    /* Notificacao caso o numero seja o numero maximo definido*/
    var quantidadeMax = document.getElementById("getget");
    var noo = quantidadeMax.getAttribute("max");
    if (noo == quantity) {
        document.getElementById("resposta").innerHTML = "Quantidade maxima permitida!";
        document.getElementById("resposta").style.visibility = "visible";
    }
    else document.getElementById("resposta").style.visibility = "hidden";

    /* Recacular total do carrinho */
    productRow.children('.subtotal').each(function () {
        $(this).fadeOut(fadeTime, function () {
```

```

        $(this).text(linePrice.toFixed(2));
        recalculateCart();
        $(this).fadeIn(fadeTime);
    });
}

productRow.find('.item-quantity').text(quantity);
updateSumItems();
}

```

Listing 5.2: Método que permite aumentar ou diminuir a quantidade de um artigo e envia uma notificação caso seja atingido o limite.

Ainda nesta *view*, antes de seguir para *Checkout* é possível aplicar um *voucher* caso o utilizador o tenha. O *voucher* pode ser escolhido através de um *selector* que apresenta todos os *voucher* disponíveis. Para tal é enviado um *trigger* através do elemento *onclick* para atualizar o total do carrinho de cada vez que é escolhido um *voucher*.

```

<div class="summary-delivery">
    <select id="list" name="delivery-collection" amiga="@oioi" onchange="@getSelectValue()" onclick="recalculateCart(true)" class="summary-delivery-selection">
        @for (var item1 in Model.FirstOrDefault().IdUtilizadorNavigation.Voucher)
        {
            int index2 = 0;
            <option value=@index2 selected="selected">Selecione Voucher</option>
            @foreach (var item2 in Model.FirstOrDefault().IdUtilizadorNavigation.Voucher)
            {
                index2 = index2 + 1;
                <option value=@index2>@Html.DisplayFor(modelItem => item1.Codigo)</option>
            }
        }
    </select>
</div>

```

Listing 5.3: Código HTML do *select* que permite seleccionar um *voucher* disponível.

Após a escolha de um, o total do carrinho é actualizado apresentando o preço total com o desconto que o *voucher* tem. Na situação de escolha de um *voucher* com desconto igual ou superior à compra a efectuar, é recebida uma notificação de que é impossível aplicar este mesmo à compra final.

```

//Se existir uma promocao retira ao preco. No entanto apenas se a compra
ultrapassar o valor da promocao
var promoPrice = parseFloat(document.getElementById("gfg").innerText);
if (promoPrice) {
    if (subtotal >= promoPrice) {
        total -= promoPrice;
    }
}

```

```

} else {
    alert('A compra deve ser mais elevada para usar este cupo.');
    $('.summary-promo').addClass('hide');
}
}

```

Listing 5.4: Código Javascript que verifica se o voucher é aplicável e envia um notificação caso negativo.

RESULTADOS

Bicicleta montanha

200 €

venda

Bicicleta de montanha como nova.

[Adicionar ao Carrinho](#)

Figura 32: Adicionar artigo ao Carrinho na view *Details*

O produto foi adicionado ao seu carrinho

Nome	Quantidade	Preco (€)	Foto do Artigo
Bicicleta montanha	1	200	

[Ver Carrinho](#)

[Continuar a comprar](#)

Figura 33: View *Adicionar Carrinho*, após Utilizador adicionar um produto ao carrinho

Carrinho de Compras

Artigo	Preço	Quantidade	Total
Huda Paleta Maquilhagem Código do Artigo - 18	25 €	1	25.00
Brincos Almond Código do Artigo - 23	15 €	1	15.00

[REMOVER](#) [REMOVER](#)

2 Itens no Carrinho

Subtotal 40.00

Adicione um Voucher:
[Selecionar Voucher](#)

TOTAL 40.00

[CHECKOUT](#)

Figura 34: View *VendaInfo*, o carrinho de compras

Quantidade

54

Quantidade máxima permitida!

Adicione um Voucher:
[Selecionar Voucher](#)

Selecionar Voucher

DRT45H
OMR49W
OOKA76

2 Itens no Carrinho

Subtotal 40.00

Adicione um Voucher:
[OMR49W](#)

TOTAL 30.00

[CHECKOUT](#)

Figura 35: Notificação de quantidade máxima

Figura 36: Selector com voucher disponíveis

Figura 37: Preço actualizado após desconto

A compra deve ser mais elevada para usar este cupão.

[OK](#)

Carrinho de Compras

Artigo	Preço	Quantidade	Total
Marcadores Pastel Stabilo Código do Artigo - 30	8 €	1	8.00

[REMOVER](#)

1 Itens no Carrinho

Subtotal 8.00

Adicione um Voucher:
[Selecionar Voucher](#)

TOTAL 8.00

[CHECKOUT](#)

Figura 38: Notificação da tentativa de uso de *voucher* com valor de oferta superior ao total do carrinho

5.2.10 Alugueres

No caso do método de adquirir artigos em modo de aluguer o utilizador começa por escolher o intervalo de datas sobre o qual pretende usufruir do item em questão enviando posteriormente um pedido ao dono, uma vez que o item pode já estar indisponível, ou até mesmo, o dono pode não querer alugar para determinados utilizadores, fazendo com que este tenha total responsabilidade sobre o gerência dos seus alugueres.

Na parte do dono do item vai então existir uma secção da navbar relativa às notificações, sendo esta abordada com mais detalhes na subsecção 5.2.11.

RESULTADOS

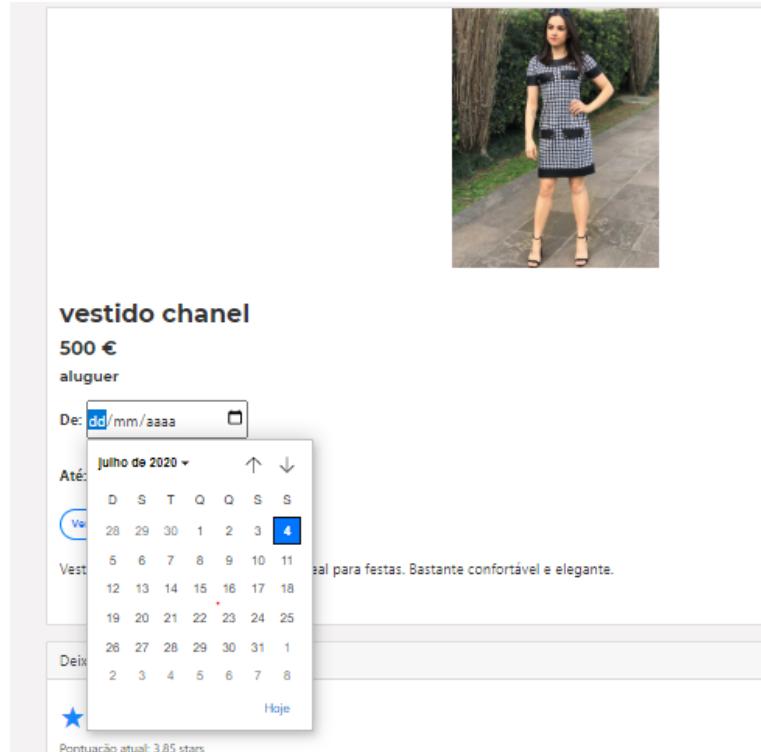


Figura 39: Utilizador: selecionar datas de aluguer

5.2.11 Notificações

Para ser possível as notificações do utilizador, recorremos ao método *ViewData* do ASP.NET, que é útil para passar informação direta de um Controller para uma View. Assim, em todas as

Views do Utilizador existe uma secção da *navbar* que mostra o *badge* das notificações junto ao seu hiperlink apenas quando são feitos pedidos de aluguer a esse Utilizador ou aceite/recusado um dos seus pedidos de aluguer, com o respetivo número de pedidos/respostas.

```
public ActionResult Contact()
{
    string user = Helpers.CacheController.utilizador;
    Utilizador u = model.Utilizador.Where(x => x.Email.Equals(user)).
    FirstOrDefault();
    int notifications = u.Notificacoes;

    ViewData["noti"] = notifications;

    return View();
}
```

Listing 5.5: Exemplo de passagem das notificações no Controller.

```
@section NotId {
<li>
<a href="@Url.Action("AluguerInfo", "Utilizador")">
    <span class="glyphicon glyphicon-bell"></span>
    Notificacoes
    @int n = (int)@ViewData["noti"];
    if (n > 0){
        <span class="badge" style="background-color: #138AB2">n</span>
    }
</a>
</li>
}
```

Listing 5.6: Notificações na View.

Nos métodos do *Controller*, existe uma variável que incrementa as notificações do utilizador (atributo da base de dados) cada vez que é feito um pedido de aluguer a um dos seus artigos, ou quando um outro utilizador responde a um dos seus pedido de aluguer. Cada vez que o utilizador vê as notificações, esse parametro retorna a 0.

Este processo foi reproduzido de igual modo para utilizadores *Company*, albergando os respetivos Controller e Views, excetuando claro as notificações a respostas a pedidos de aluguer, visto que essa funcionalidade não lhes é possibilitada.

RESULTADOS

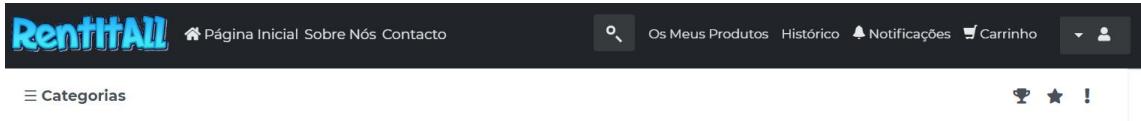


Figura 40: Navbar sem notificações

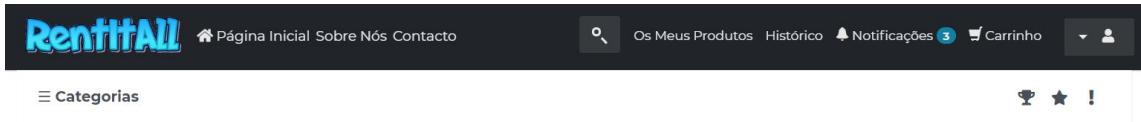


Figura 41: Badge com notificações

5.2.12 Histórico

O histórico é também uma secção da *navbar* do Utilizador que lhe permite ver quais as suas vendas e pedidos de aluguer efectuados e recusados tanto por ele como sobre ele.

A nível de implementação, tanto a entidade Venda como a entidade Aluguer têm um parâmetro do tipo int correspondente ao seu estado. No caso dos alugueres este tem o valor de 0 quando este ainda é um pedido, 1 se for aceite e 2 se for recusado. Assim sendo, é apenas feita uma querie que filtra os alugueres com estados 1 e 2. Já no caso das vendas, o estado tem valor de 0 caso o item ainda se encontre no carrinho de compras e 1 caso a compra tenha sido efetuada com sucesso. Nesta situação apenas iremos filtrar as vendas com estado 1.

RESULTADOS

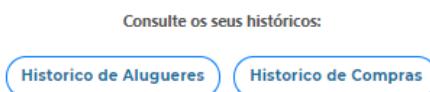


Figura 42: Utilizador - Histórico

Histórico de Alugueres								
	Nome	Quantidade	Preco (€)	Foto do Artigo	Data de Início	Data de fim	Email	Alugado por
Realizado	vestido chanel	1	300		03/01/2020	06/01/2020	margarida@gmail.com	Margarida Campos
Recebido	cadeira	1	200		20/01/2020	22/01/2020	tania@gmail.com	Tânia Rocha
Realizado	sistema de som	1	150		10/01/2020	10/01/2020	pedro@gmail.com	Pedro Fernandes

Figura 43: Utilizador - Histórico de Alugueres

5.2.13 Pontuação

Um utilizador modo *single* (inerentemente com login efetuado) pode avaliar os produtos de outros utilizador conforme a sua experiência no processo de compra/aluguer, através de um sistema de pontuação por estrelas. Na base de dados, a tabela *Artigo* tem os seguintes atributos, que contribuem para a implementação desta funcionalidade: *pontuação*, *pontuação acumulada*, *número de votos*. Assim, cada vez que é um utilizador atribui uma pontuação a um artigo, a média da pontuação é recalculada, através de operações do método *Stars* do *UtilizadorController*.

```
[HttpPost]
public IActionResult Stars(int IdArtigo, int pont){

    Artigo art = model.Artigo.FirstOrDefault(x => x.IdArtigo.Equals(IdArtigo));

    double pontuacao = art.Pontuacao;

    int nrVotos = art.NumeroVotos;
    int newNrVotos = nrVotos + 1;

    art.NumeroVotos = newNrVotos;

    double newPontuacaoAcumulada = art.PontuacaoAcumulada + pont;
```

```

        double newPontuacao = newPontuacaoAcumulada / (double)newNrVotos;

        art.Pontuacao = Math.Truncate(100 * newPontuacao) / 100; ;
        art.PontuacaoAcumulada = newPontuacaoAcumulada;

        if (ModelState.IsValid) model.SaveChanges();

        return RedirectToAction("Details", "Utilizador", new { IdArtigo = IdArtigo });
    }
}

```

Listing 5.7: Método para calcular a nova Pontuação de um artigo após uma nova avaliação.

Na View Details existe um dos mais complexos formulários HTML que permite a apresentação das estrelas com a pontuação atual e a submissão da pontuação aquando do clique do utilizador na estrela correspondente à avaliação pretendida.

RESULTADOS



Figura 44: Avaliação de um produto na view *Details* do Utilizador



Figura 45: Pontuação de um produto na view *Details* da Home (não logado)



Figura 46: Pontuação de produtos em views de amostragem de artigos

5.2.14 Comentários

Um utilizador *loggado* pode deixar comentários com o fim de partilhar a sua experiência com determinado artigo, contemplando tanto aspectos dirigidos diretamente ao artigo, como opiniões do processo de venda/aluguer com o dono.

É possível visualizar esta secção de comentários sem efetuar *login*, porém *postar* um comentário é uma funcionalidade reservada a utilizadores com conta no site.

RESULTADOS

Figura 47: Envio de um comentário na View *Details* do Utilizador.

5.2.15 Denúncias

De modo a permitir que um utilizador mostre a sua insatisfação de um determinado produto ou mesmo à maneira como este foi entregue pelo dono, foi criada a possibilidade de fazer denúncias.

Quando um utilizador deseja fazer uma denúncia terá de se dirigir à página do artigo que pretende denunciar e lá terá um botão que o levará para escrever a sua crítica. Na parte da implementação, as denúncias feitas são, através de uma querie, adicionadas à base de dados do sistema e o número de denúncias do dono do produto é incrementado.

O administrador tem a tarefa de verificar na sua conta as denúncias feitas aos artigos e decidir se as mesmas são válidas ou foram comprovadas para determinar se o dono do produto será notificado por email ou então bloqueado do sistema, conforme o número de denúncias e a importância das mesmas.

Todavia, o sistema não limita o número de denúncias para o mesmo artigo, isto é, nos casos onde um utilizador realize denúncias sem provas e várias para o mesmo artigo e/ou o mesmo dono. Mostrando assim uma pequena falha no sistema, embora que o administrador possa, depois de determinar o culpado, bloquear essa conta.

RESULTADOS

The screenshot shows a user interface for reporting an article. At the top right, there is a button labeled "Denunciar Artigo" with a warning icon. Below it, a large text input field is labeled "Insira a sua denúncia:". At the bottom right of the input field, there is a blue button labeled "Enviar a minha denúncia".

Figura 48: Botão para denunciar

Figura 49: Inserir denúncia

5.3 MÉTODOS DO ADMINISTRADOR

Passando para a implementação dos métodos do administrador foi tido em conta varias decisões quanto ao funcionamento das mesmas.

Primeiramente foi decidido que o administrador teria acesso a todas as denúncias existentes no site e poderia aceitar ou rejeitar após a avaliação respectiva de cada uma.

O administrador tem também a possibilidade de avaliar um determinado utilizador quanto ao número de denúncias que este tem e a partir delas decidir se a conta desse utilizador é bloqueada ou não. Por outro lado pode decidir enviar um aviso para o email do utilizador para o notificar que está em risco de ter a sua conta bloqueada devido ao numero de denuncias que lhe tem associadas.

Como página inicial do Administrador foram implementados métodos de estatísticas variados que permitem uma análise real dos dados presentes na página web, nomeadamente as vendas e alugueres que têm sido efectuados, o número de utilizador, artigos e denúncias e os artigos e denuncias mais recentes.

5.3.1 *Denúncias*

Dentro do domínio da página web do Administrador é possível então consultar os utilizadores e respectivas denuncias. Nesse mesmo local é possível então analisar cada denúncia através da sua descrição e posteriormente aceita-la, incrementando assim ao dono do artigo denunciado, o número de denúncias, ou rejeitar ficando esta sem efeito. Para rejeitar, isto é, remover denuncias utilizou-se o seguinte método:

```
public ActionResult rejeitarDenuncia(int idDenuncia)
{
    List<Denuncias> lista = (from den in model.Denuncias where (den.IdDenuncia
    >= idDenuncia) select den).ToList<Denuncias>();
    Denuncias k = lista.ElementAt<Denuncias>(0);
    lista.Remove(k);
    Artigo a = (from x in model.Artigo where (k.IdArtigo == x.IdArtigo) select
    x).FirstOrDefault();
    Utilizador u = model.Utilizador.FirstOrDefault(x => x.Email.Equals(a.
    IdDono));
    model.Denuncias.Remove(k);
    model.SaveChanges();
    return View();
}
```

No caso de aceitar um denúncia o método desenvolvido recebe o id da denúncia a aceitar e segue então para a incrementação do número de denúncias do utilizador respetivo:

```
public ActionResult AceitarDenuncia(int idDenuncia)
{
    List<Denuncias> lista = (from den in model.Denuncias where (den.IdDenuncia
    >= idDenuncia) select den).ToList<Denuncias>();
    Denuncias k = lista.ElementAt<Denuncias>(0);
    lista.Remove(k);
    Artigo a = (from x in model.Artigo where (k.IdArtigo == x.IdArtigo) select
    x).FirstOrDefault();
    Utilizador u = model.Utilizador.FirstOrDefault(x => x.Email.Equals(a.
    IdDono));
    u.NDenuncias++;
    if (k != null) model.Denuncias.Remove(k);
    model.SaveChanges();
    return View();
}
```

RESULTADOS

Denúncias					
Índice de Denúncia	Descrição	Autor da Denuncia	Artigo denunciado	Dono do Artigo	Num. de Denúncias do Dono
2	Deviam também informar que tem uma discoteca ao lado muito barulhenta até altas horas.	tania@gmail.com	Apartamento Espoende	Margarida Campos margarida@gmail.com	4

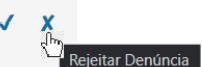


Figura 50: Aceitar ou rejeitar uma denúncia

5.3.2 Utilizadores

O Administrador tem como principal objetivo manter ordem e controlo no uso da página web, deste modo, o administrador tem as capacidades de bloquear utilizadores de utilizarem o serviço. Assim, foram criados métodos para bloquear o utilizador, como também enviar um

aviso por e-mail para notificar o utente da sua má conduta ou da sua possível suspensão de utilização do serviço.

Os métodos responsáveis de bloquear ou avisar um utilizador, por opção do administrador, são:

```

public ActionResult Warning(string email)
{
    Utilizador u = model.Utilizador.Where(x => x.Email.Equals(email)).
FirstOrDefault();
    PassRec pr = new PassRec();
    pr.Warning(email);
    return RedirectToAction("Index", "Admin");
}

public ActionResult Bloquear(string email)
{
    Utilizador u = model.Utilizador.Where(x => x.Email.Equals(email)).
FirstOrDefault();
    u.Estoado = 2;
    model.SaveChanges();
    return RedirectToAction("Index", "Admin");
}

```

RESULTADOS

Utilizadores na aplicação				
Nome	Email	Número de Denúncias		
Pedro Fernandes	pedro@gmail.com	4		
Margarida Campos	margarida@gmail.com	4		
Company1	company1@gmail.com	3		
Margarida Campos	margarida.maia.campos@gmail.com	2		

Figura 51: Bloquear ou enviar aviso a um utilizador

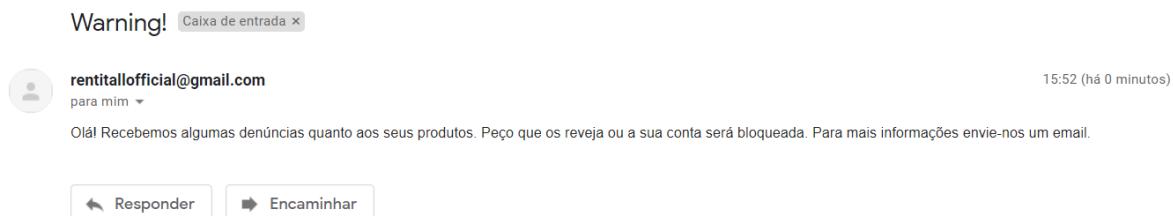


Figura 52: Email de aviso recebido.

5.3.3 Estatísticas

A página inicial do administrador é, ao contrário do utilizador *single*, constituída por vários elementos de estatística referente a todos os elementos do site, nomeadamente artigos, denúncias, *vouchers*, vendas e alugueres.

Existem três elementos de estatísticas principais no nosso site, o mais simples são apenas a contagem do número de utilizadores, artigos, denúncias e vouchers existentes no site.

O segundo elemento trata-se de um gráfico que demonstra o preço das vendas e alugueres comparativamente ao longo do tempo. Este é desenvolvido em javascript com informação obtida da nossa base de dados enviada a partir de elementos html denominados por ”one1”, ”tho3” e ”@Model.precario” que guardam informação dos alugueres, vendas e artigos respetivamente.

```
<script src="/js/fLOT/fLOT-active.js" one=@one2 two=@two3 three=@Model.precario>
</script>
```

O terceiro é uma tabela que lista os elementos mais recentes no website e um gráfico correspondente ao valor dos mesmos. Este elemento é enviado da mesma maneira que descrito anteriormente através do elemento ”@Model.precario”.

RESULTADOS

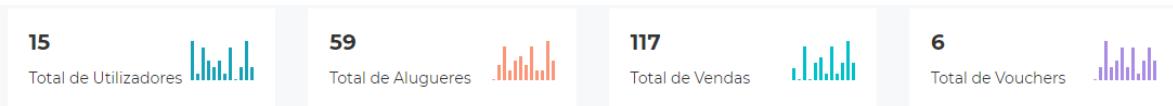


Figura 53: Numerário de elementos do site

Estatísticas de Compras e Alugueres

Representação ao longo do tempo



Figura 54: Evolução de vendas comparativamente a alugueres ao longo do tempo

Denúncias Recentes

margarida@gmail.com

Demorou demais a chegar. Ultrapassou...

joana@gmail.com

Faltam páginas do livro

tania@gmail.com

Vieram dois tamanhos diferentes!!!

tania@gmail.com

Deviam também informar que tem um...

[Ver mais](#)

Itens Recentes

ID	Nome	Preço
31	Rolex Sport GMT Master	150 €
30	Marcadores Pastel Stabilo	8 €
29	BLACK SATIN OFF SHOULDER DRESS	209 €
28	Sapato blucher pele	77 €
27	Conjunto de Acessórios de Diamantes	406 €



Figura 55: Listagem das denúncias recentes

Figura 56: Numerário de elementos mais recentes do site

5.4 MÉTODOS DA COMPANY

Para o último tipo de utilizador, *Company* foi desenvolvida uma área de semelhança ao do Administrador com estatísticas correspondentes ao artigos da empresa correspondente. Foi também desenvolvida uma *view Catálogo* que lista os artigos da empresa, permite a sua alteração, restock, caso estejam fora de stock e permite adicionar novos produtos. Este tipo de utilizador também recebe notificações quanto aos pedidos de alugueres e tem acesso ao histórico de compras e alugueres (*Cf. Anexo 1*).

5.5 OUTROS RESULTADOS FRONTEND

5.5.1 Botão para o topo da página

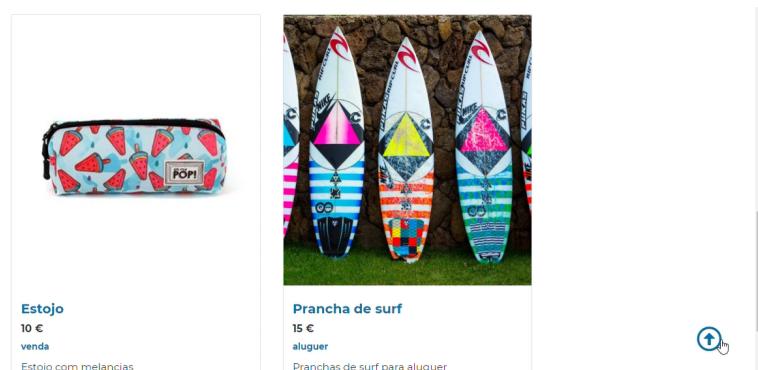


Figura 57: Parcial de uma view com botão para regressar ao topo da página

5.5.2 Contactos



Figura 58: Página de Contactos

5.5.3 Sobre Nós

Cf. Anexo 2: Vista reduzida e não iterativa da página Sobre Nós.

6

CONCLUSÃO

No final deste projeto, a equipa conclui que foi desenvolvido com sucesso uma página web capaz de disponibilizar o serviço de aluguer e compra de artigos.

Foram também implementadas várias funcionalidades para cada tipo de utilizador conforme os requisitos levantados no início do projeto.

6.1 CONCLUSÕES

Este trabalho proporciona assim as capacidades necessárias para um utilizador poder dar e tomar aluguer de um artigo, como também, vender e comprar outros itens. É também fornecida a possibilidade de uma empresa criar a sua página com os diversos artigos que a empresa disponibiliza. Por fim, o administrador tem como responsabilidade de moderar a página web, desta forma, foram disponibilizadas funcionalidades que possam lidar com quaisquer situações previstas nos requisitos.

6.2 PERSPECTIVA DE TRABALHO FUTURO

Numa perspetiva mais profissional, é notável que num projeto complexo como este, sejam necessárias algumas adições que iriam melhorar facetas na nossa plataforma, tais como: A criação de novos *vouchers* ser também a responsabilidade das companies e do administrador; também podia-se utilizar outros métodos de registar uma nova conta ou efetuar login, como por exemplo utilizando as contas de redes sociais (Facebook, Twitter, etc...); criação de uma *mailbox* que permitisse o contacto com o administrador do site através do mesmo e recessão de respostas; melhorar a dinâmica da página web, ou seja, desenvolver uma coluna com artigos semelhantes ao que foi carregado para que o cliente encontre mais artigos de que possa estar interessado;

B I B L I O G R A F I A

<https://www.w3schools.com>

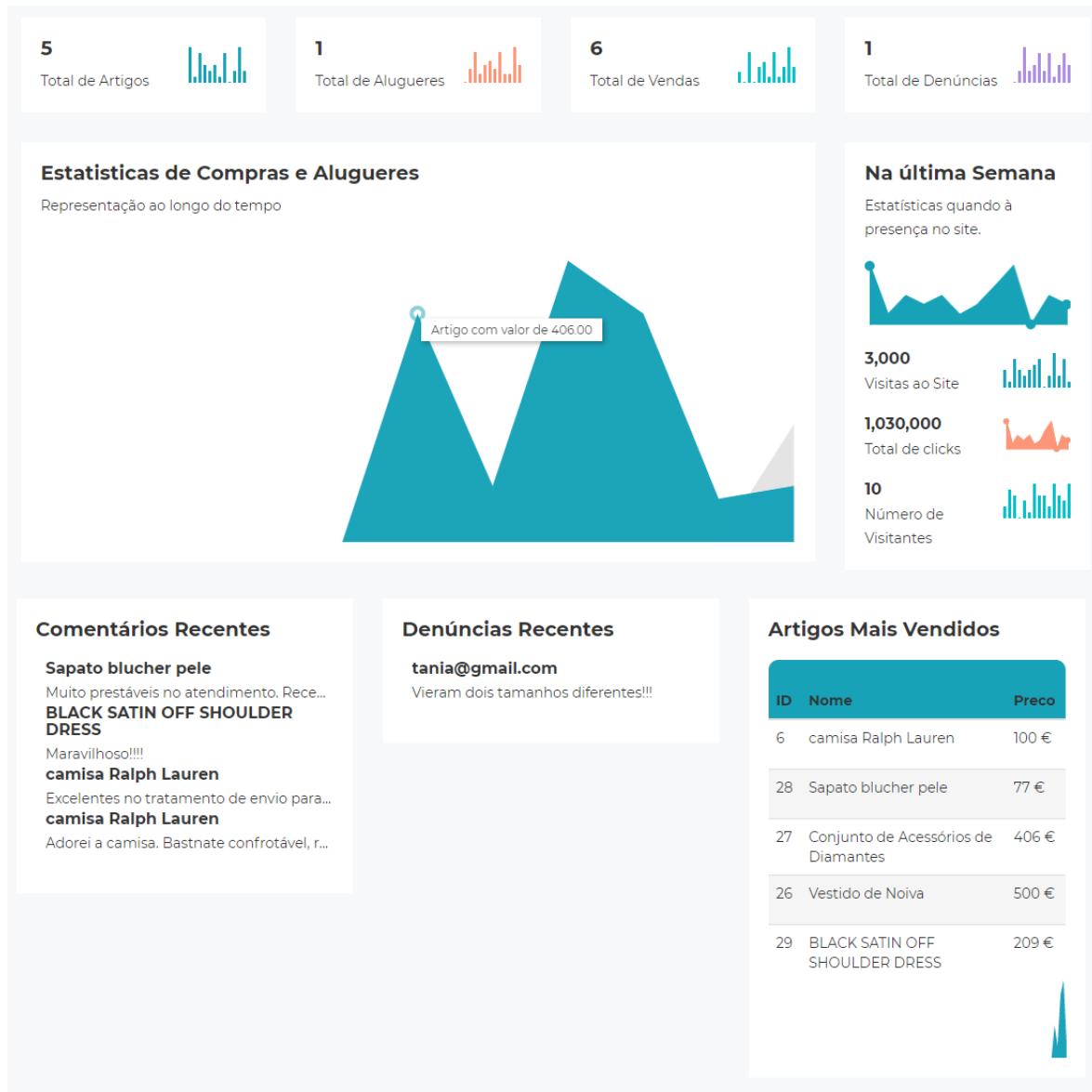
<https://stackoverflow.com>

<https://www.codecademy.com/learn/learn-html>

<https://docs.microsoft.com>

Anexos

.1 ANEXO 1: PÁGINA INICIAL DAS COMPANY



.2 ANEXO 2: VISTA REDUZIDA E NÃO ITERATIVA DA PÁGINA SOBRE NÓS

SOBRE NÓS

Conheça a história e a equipa.

2020

De onde somos

Estudantes da universidade do Minho situada em Braga, Portugal



2020

Área de Aprendizagem

Somos estudantes do curso de Engenharia Informática



Fevereiro de 2020

Contextualização

Este website está a ser construído por nós no âmbito da Unidade Curricular de Laboratórios de Informática IV e teve inicio em Fevereiro de 2020



Junho de 2020

Objetivos

Construção de um website que permita venda e aluguer dos mais variados itens!



A NOSSA EQUIPA

Conheça as pessoas por trás do website



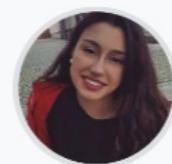
Ana Catarina Gil
Backend



Ana Margarida Campos
Backend



Tânia Rocha
FrontEnd



Joana Afonso Gomes
FrontEnd



Pedro Fernandes
Backend

