

**Carolo Farmacêutica**

TeSP em Programação de Sistemas de Informação

Hugo Emanuel Da Luz Moreira Dias

Pedro Miguel Ideias Francisco

Tiago Santos Da Silva

Leiria, Janeiro de 2024



**Carolo Farmacêutica**

2220853 – Hugo Emanuel Da Luz Moreira Dias

2220879 – Pedro Miguel Ideias Francisco

2220864 – Tiago Santos Da Silva

Trabalho de Projeto em Sistemas de Informação do curso TeSP em Programação de Sistemas de Informação.

Leiria, Janeiro de 2024

# Originalidade e Direitos de Autor

O presente relatório de projeto é original, elaborado unicamente para este fim, tendo sido devidamente citados todos os autores cujos estudos e publicações contribuíram para o elaborar.

Reproduções parciais deste documento serão autorizadas na condição de que seja mencionado/a o/a Autor/a e feita referência ao ciclo de estudos no âmbito do qual a/o mesma/o foi realizado, a saber, TeSP em Programação de Sistemas de Informação, no ano letivo 2023/2024, da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria, Portugal, e, bem assim, à data das provas que visaram a avaliação destes trabalhos.

# Agradecimentos

Agradecemos aos professores do Curso Técnico Superior Profissional de Programação de Sistemas de Informação, que desempenharam um papel fundamental no nosso desenvolvimento académico e no sucesso do nosso projeto. Agradecemos à Professora Dulce Gonçalves, Joana Pedrosa, Cátia Ledesma, Sónia Luz, Marisa Maximiano, e aos Professores Mário Fernandes e Sílvio Mendes pela sua orientação e apoio ao longo desta jornada.

# Resumo

O presente relatório, descreve o desenvolvimento de um projeto designado “Carolo Farmacêutica”, realizado no âmbito das Unidades Curriculares de Plataformas de Sistemas de Informação, Acesso Móvel a Sistemas de Informação e Serviços e Interoperabilidade de Sistemas que consiste na realização de uma aplicação web e móvel.

Este projeto tem como objetivo criar uma representação de uma farmácia online, permitindo a venda de produtos farmacêuticos aos clientes, e possibilitando a gestão da farmácia pelos seus administradores e funcionários.

Com a realização deste projeto, foram aprofundados alguns conhecimentos já adquiridos ao longo do curso, bem como foram adquiridos novos conhecimentos para a sua realização.

**Palavras-chave:** saúde, bem-estar, aprendizagem

# Abstract

The report describes the development of a project called “Carolo Farmacêutica”, carried out for the Curricular Units Plataformas de Sistemas de Informação, Acesso Móvel a Sistemas de Informação e Serviços e Interoperabilidade de Sistemas consists of the implementation of a web and mobile application.

This project aims to establish an online pharmacy, allowing the sale of pharmaceutical products to its users and management to its staff and administrators.

While developing Carolo Farmacêutica, we deepened the knowledge gained throughout the course and gained substantial new knowledge for its development.

**Keywords:** health, well-being, learning

Índice

[Originalidade e Direitos de Autor iii](#_Toc158374577)

[Agradecimentos iv](#_Toc158374578)

[Resumo v](#_Toc158374579)

[Abstract vi](#_Toc158374580)

[Lista de Figuras ix](#_Toc158374581)

[Lista de tabelas xi](#_Toc158374582)

[Lista de siglas e acrónimos xii](#_Toc158374583)

[1 Introdução 1](#_Toc158374584)

[1.1 Objetivos 2](#_Toc158374585)

[1.2 Planeamento 2](#_Toc158374586)

[2 Enquadramento 3](#_Toc158374587)

[3 Tecnologias Utilizadas 4](#_Toc158374588)

[3.1 GitHub 5](#_Toc158374589)

[3.1.1 Branches 5](#_Toc158374590)

[3.1.2 Commits 6](#_Toc158374591)

[3.1.3 Merge 6](#_Toc158374592)

[3.1.4 Pull Requests 6](#_Toc158374593)

[4 Arquitetura do Sistema 7](#_Toc158374594)

[5 Metodologias 9](#_Toc158374595)

[6 Análise e Desenho 12](#_Toc158374596)

[6.1 Requisitos do Sistema 12](#_Toc158374597)

[6.2 User Stories 14](#_Toc158374598)

[6.3 Casos de Uso 15](#_Toc158374599)

[6.4 Modelo de Dados 17](#_Toc158374600)

[6.5 Mockups/wireframes 18](#_Toc158374601)

[6.5.1 WebSite 18](#_Toc158374602)

[6.5.2 Aplicação Móvel 22](#_Toc158374603)

[7 Implementação 26](#_Toc158374604)

[7.1 WebSite 26](#_Toc158374605)

[7.1.1 Yii2 26](#_Toc158374606)

[7.1.2 MVC 26](#_Toc158374607)

[7.1.2.1 Model (Modelo) 26](#_Toc158374608)

[7.1.2.2 View (Vista) 26](#_Toc158374609)

[7.1.2.3 Controller (Controlador) 26](#_Toc158374610)

[7.1.3 Backend 27](#_Toc158374611)

[7.1.4 Frontend 31](#_Toc158374612)

[7.2 Aplicação Mobile 38](#_Toc158374613)

[7.2.1 Android Studio 38](#_Toc158374614)

[7.3 API 44](#_Toc158374615)

[8 Testes 45](#_Toc158374616)

[8.1.1 Testes Unitários 45](#_Toc158374617)

[8.1.2 Testes Funcionais 45](#_Toc158374618)

[8.1.3 Testes de Aceitação 46](#_Toc158374619)

[8.1.4 Execução dos testes 46](#_Toc158374620)

[8.1.5 Resultados obtidos e correção de erros 47](#_Toc158374621)

[9 Conclusões ou Conclusão 48](#_Toc158374622)

[Bibliografia ou Referências Bibliográficas 49](#_Toc158374623)

[Glossário 50](#_Toc158374624)

[Anexos 51](#_Toc158374625)

# Lista de Figuras

[Figura 1 - Ferramenta Android Studio 4](#_Toc158374626)

[Figura 2 - Ferramenta Jira 4](#_Toc158374627)

[Figura 3 - Ferramenta Restful 5](#_Toc158374628)

[Figura 4 - Ferramenta PhpStorm 5](#_Toc158374629)

[Figura 5 - Yii2 PHP Framework 5](#_Toc158374630)

[Figura 6 - Esquema de um conjunto de Branchs 6](#_Toc158374631)

[Figura 7 - Diagrama da Arquitetura do Sistema 8](#_Toc158374632)

[Figura 8 - Gráfico de burndown 10](#_Toc158374633)

[Figura 9 - Diagrama Casos de Uso 15](#_Toc158374634)

[Figura 10 - Modelo de Dados (Base Dados) 17](#_Toc158374635)

[Figura 11 - Wireframe ecrã login 18](#_Toc158374636)

[Figura 12 - Wireframe ecrã homepage 18](#_Toc158374637)

[Figura 13 - Wireframe ecrã checkout (Carrinho de Compras) 19](#_Toc158374638)

[Figura 14 - Wireframe ecrã após checkout (Dados de Entrega) 19](#_Toc158374639)

[Figura 15 - Wireframe ecrã homepage (admin) 20](#_Toc158374640)

[Figura 16 - Wireframe ecrã faturas 20](#_Toc158374641)

[Figura 17 - Wireframe ecrã gerir funcionários 21](#_Toc158374642)

[Figura 18 - Wireframe ecrã gerir medicamentos 21](#_Toc158374643)

[Figura 19 - Wireframe ecrã gerir receitas médicas 22](#_Toc158374644)

[Figura 20 - Wireframe Janela Inicial 22](#_Toc158374645)

[Figura 21 - Wireframe Receitas Médicas 23](#_Toc158374646)

[Figura 22 - Wireframe slidebar 24](#_Toc158374647)

[Figura 23 - Wireframe Carrinho de Compras 24](#_Toc158374648)

[Figura 24 - Wireframe pagamento (multibanco) 25](#_Toc158374649)

[Figura 25 - Página inicial Backend 27](#_Toc158374650)

[Figura 26 - Gestão de produtos 28](#_Toc158374651)

[Figura 27 - Seleção de cliente e estabelecimento 29](#_Toc158374652)

[Figura 28 - Inserção de serviços ou receitas 29](#_Toc158374653)

[Figura 29 - Ecrã de emissão da fatura 30](#_Toc158374654)

[Figura 30 - Página inicial do frontend 31](#_Toc158374655)

[Figura 31- Ecrã de inserção de receita-médica 32](#_Toc158374656)

[Figura 32- Página detalhes de um medicamento 32](#_Toc158374657)

[Figura 33- Medicamentos - página inicial 33](#_Toc158374658)

[Figura 34- Menu produtos 33](#_Toc158374659)

[Figura 35 - Dados de entrega 34](#_Toc158374660)

[Figura 36- Menu cliente - faturas 34](#_Toc158374661)

[Figura 37- Página Faturas 35](#_Toc158374662)

[Figura 38- Fatura 36](#_Toc158374663)

[Figura 39- Fatura Impressa 36](#_Toc158374664)

[Figura 40- Receita médica - cliente 37](#_Toc158374665)

[Figura 41 - Login da app mobile 38](#_Toc158374666)

[Figura 42 - Janela Inicial 39](#_Toc158374667)

[Figura 43 - Lista de receitas médicas 40](#_Toc158374668)

[Figura 44 - Carrinho de Compras 41](#_Toc158374669)

[Figura 45 - Opção de Pagamento 42](#_Toc158374670)

[Figura 46 - Opção pagamento com cartão 42](#_Toc158374671)

[Figura 47 – Opção de pagamento com multibanco 43](#_Toc158374672)

[Figura 48 - Exemplo de um pedido à API 44](#_Toc158374673)

# Lista de tabelas

Tabela 1- Identificação de stakeholder e equipa de desenvolvimento 10

Tabela 2 - Tabela de sprints 11

Tabela 3 - Roles dos diferentes end-users do SI 12

Tabela 4 - Requisitos funcionais Front-Office 13

Tabela 5 - Requisitos funcionais Back-Office 13

Tabela 6 - Requisitos funcionais App mobile 13

Tabela 7 - Cruzamento dos requisitos funcionais e respetivos roles 13

Tabela 8 - Requisitos Não funcionais 13

Tabela 9 - Tabela de testes unitários 45

Tabela 10 - Tabela de testes funcionais 45

Tabela 11 - Tabela de testes de aceitação 46

# Lista de siglas e acrónimos

|  |  |
| --- | --- |
| API | Application Programming Interface - Interface de programação de aplicação |
| CSS  ESTG | Cascading Style Sheets  Escola Superior de Tecnologia e Gestão |
| HTML | HyperText Markup Language |
| IDE | Integrated Development Environment – Ambiente de desenvolvimento integrado |
| JSON | JavaScript Object Notation |
| MVC | Model-View-Controller |
| PHP | Hypertext Preprocessor |
| REST | Representational State Transfer |
| SQL | Structured Query Language - Linguagem de Consulta Estruturada |
| XML | Extensible Markup Language |
|  |  |
|  |  |
|  |  |

# Introdução

Este projeto consiste num projeto realizado por três alunos, Hugo Dias, Pedro Francisco e Tiago Silva, no âmbito de um projeto final abrangendo as Unidades Curriculares Plataformas de Sistemas de Informação, Serviços de Informação e Interoperabilidade e Sistemas Móveis de Acesso à Informação. Insere-se no Curso Técnico Superior Profissional em Programação de Sistemas de Informação oferecido pela Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria.

O projeto chama-se “Carolo Farmacêutica” e consiste na criação de uma aplicação web e mobile. Esta aplicação tem como objetivo facilitar o acesso às farmácias e pode ter um impacto muito positivo na vida dos clientes, por exemplo ao facilitar a compra de medicamentos sem sair de casa.

Este tema foi escolhido para o projeto devido ao facto de ser algo útil para o utilizador, e também por ser algo que não é muito comum, permitindo-nos adquirir e aprofundar conhecimentos no desenvolvimento de aplicações móveis, criação de um website e criação de uma **API**. Estas aplicações permitem a venda de produtos farmacêuticos aos clientes, bem como a gestão da farmácia por parte dos seus administradores e funcionários.

Esta relatório discute métodos, tecnologias e linguagens como Yii2, **MVC**, **HTML**, **CSS** e **PHP** usados ​​no desenvolvimento de aplicações, especialmente na web. Usamos Java para dispositivos móveis e Yii2, **JSON** e **PHP** para nossa **API**. O Jira é usado para distribuição de tarefas e o GitHub é usado para controlo de versões. O trabalho é distribuído por web, dispositivos móveis e **API**s, com a web dividida em *backend* e *frontend*.

O presente documento encontra-se subdividido em:

* Tecnologias Utilizadas;
* Arquitetura do Sistema;
* Metodologias (Gestão do projeto e desenvolvimento);
* Análise e Desenho;
* Implementação;
* Testes.

## Objetivos

Este Sistema tem como objetivo ajudar a tornar a experiência do Cliente única, com o foco principal em evitar que o Cliente precise se deslocar para adquirir os medicamentos necessários, criando um Sistema que contém um website com *frontend* e *backend* e uma aplicação móvel.

Os objetivos principais do *backend* do website é gerir uma farmácia como faturas, contas dos utilizadores, serviços, medicamentos, categorias, despesas, entre outros, enquanto o *frontend* tem como principal objetivo mostrar a farmácia aos Utilizadores, apresentando os medicamentos, os serviços disponíveis pela farmácia e os medicamentos por categorias, também é possível os clientes visualizarem o seu histórico de faturas e as suas receitas médicas.

Os objetivos principais da aplicação móvel são o cliente ter a possibilidade de comprar medicamentos, ver as suas receitas médicas e ver o seu histórico de faturas.

## Planeamento

|  |  |  |
| --- | --- | --- |
| Número da Tarefa | Descrição | Duração |
| 1 | Estudo e elaboração da proposta | 1 Semana |
| 2 | Análise Concorrencial | 1 Semana |
| 3 | Especificação de requisitos | 1 Semana |
| 4 | User Stories | 1 Semana |
| 5 | Mockups | 2 Semanas |
| 6 | Análise de dados | 1 Semana |
| 7 | Implementação/desenvolvimento | 3 Meses |
| 8 | Testes de software | 2 Semanas |
| 9 | Escrita do relatório de projeto | 1 Mês |

# Enquadramento

O Sistema consiste na implementação de uma aplicação web, móvel e **API** com o objetivo de facilitar o acesso de uma farmácia. O Sistema será útil e poderá ser utilizado tanto por Clientes, como também pelos Funcionários e Administradores.

Os Utilizadores podem aceder ao Sistema através de um computador ou de um dispositivo móvel, no caso de se aceder através de um computador o Utilizador tem acesso aos três cargos existentes, nomeadamente, o Administrador, Funcionário e Cliente, já quem acede através de um dispositivo móvel só o Cliente é que tem acesso ao mesmo.

No que diz respeito ao Administrador tem apenas acesso à web App e dá a possibilidade de gerir funcionários, clientes e a ele mesmo, gerir medicamentos, gerir as categorias dos produtos, gerir os serviços, gerir os fornecedores, gerir as receitas médicas dos Clientes, gerir os estabelecimentos, gerir as despesas, criar faturas, gerir ivas e gerir os serviços que os estabelecimentos realizam.

O funcionário também só tem acesso à web App, onde pode realizar tudo o que o Administrador realiza, apenas não pode gerir funcionários e apagar qualquer tipo de dados da web App.

Através da aplicação o Administrador e o Funcionário passam a ter a informação facilitada podendo consultar em qualquer parte onde possam estar, qual o produto mais vendido, o número de vendas no presente mês, o estabelecimento que mais vendeu no presente mês, entre outras informações.

O Cliente também com acesso à web App e à app móvel, onde tem a possibilidade de comprar medicamentos, consultar as suas receitas médicas, consultar os serviços disponíveis e ver o seu histórico de compras.

Por mais funcional que seja este Sistema é inevitável que não resulte em desvantagens, nomeadamente para as pessoas mais céticas às novas tecnologias que poderão ter mais dificuldades em dominar o Sistema.

# Tecnologias Utilizadas

As tecnologias utilizadas no desenvolvimento do projeto foram:

* **Android Studio** (Figura 1) - ferramenta de desenvolvimento da aplicação;
* **Volley** – é uma biblioteca **HTTP** que facilita os pedidos de rede entre a App e a **API** de destino de forma automática;
* **Jira** (Figura 2) – ferramenta utilizada para o planeamento das tarefas de cada elemento do grupo;
* **REST** (Figura 3) – aplicado para a execução e criação da **API**;
* **PhpStorm** (Figura 4) – ferramenta de desenvolvimento do website;
* **Mosquitto** – é um servidor de mensagens que implementa um protocolo de comunicação, projetado para a troca de mensagens entre sistemas;
* **Yii2 PHP Framework** (Figura 5) – framework em **PHP** de desenvolvimento do website;
* **Wamp64** – software que permite simular em servidor web real;
* **SQL** – ferramenta utilizada para a manipulação da Base de Dados;
* **Servidor** – ferramenta utilizada para armazenar todo o Sistema, onde ao longo do Semestre realizamos um conjunto de deploys, que consistia em colocar o Sistema no Servidor, ao ter o Sistema no Servidor permitiu o acesso simultâneo a vários dispositivos, no entanto, se não estiver conectado à rede da escola, será necessário a utilização de VPN.

|  |  |
| --- | --- |
| Uma imagem com Gráficos, Tipo de letra, design gráfico, logótipo  Descrição gerada automaticamente  Figura 1 - Ferramenta Android Studio | Uma imagem com captura de ecrã, Gráficos, logótipo, Azul elétrico  Descrição gerada automaticamente  Figura 2 - Ferramenta Jira |
| Uma imagem com preto, escuridão  Descrição gerada automaticamente  Figura 3 - Ferramenta Restful | Uma imagem com Gráficos, design gráfico, Tipo de letra, logótipo  Descrição gerada automaticamente  Figura 4 - Ferramenta PhpStorm |
| Uma imagem com Gráficos, design gráfico, Tipo de letra, design  Descrição gerada automaticamente  Figura 5 - Yii2 PHP Framework | |

## GitHub

O GitHub é um serviço baseado em uma nuvem que hospeda um sistema de controlo de versões (VCS) chamado Git que permite aos desenvolvedores colaborar e fazer alterações nos projetos compartilhados enquanto mantêm um registo detalhado do seu progresso. Podem acompanhar e gerir as mudanças feitas para o código-fonte em tempo real, e ao mesmo tempo fornecer acesso a todos os outros recursos do Git disponíveis no mesmo local. Além disso, a interface de utilizador do GitHub é mais simples do que a do Git, tornando-a mais acessível para pessoas que possuem pouco ou nenhum conhecimento técnico. Isso significa que mais membros da equipa possam ser incluídos na elaboração e na gestão de projetos, fazendo com que o processo seja mais eficiente.

### Branches

A *branch* é utilizada para isolar o trabalho de desenvolvimento sem afetar outras *branches* do repositório. Cada repositório possui uma *branch* *main*/*master* e pode ter outros *branches* (ramificações). Pode-se fazer *merge* de uma *branch* noutra *branch*, usando um *pull* *request*.

### Commits

Um *commit* adiciona as alterações mais recentes do código-fonte para o repositório, tornando essas alterações parte da revisão principal do repositório. É importante reforçar que os *commits* nos sistemas de controlo de versões são armazenados no repositório indefinidamente. Assim, quando outros utilizadores fizerem uma atualização ou check-out no repositório, eles irão receber a última versão confirmada.

### Merge

Um *merge* é quando juntamos duas *branches* em apenas uma e recebemos o código de uma *branch* e pedimos ao git para adicionar esse código a outra *branch*. *Merges* podem funcionar perfeitamente, mas algumas vezes o git pode ficar confuso e gerar um conflito quando se tenta fazer um *merge*, nessas situações temos de abrir o ficheiro em conflito e corrigir manualmente o *merge*.

### Pull Requests

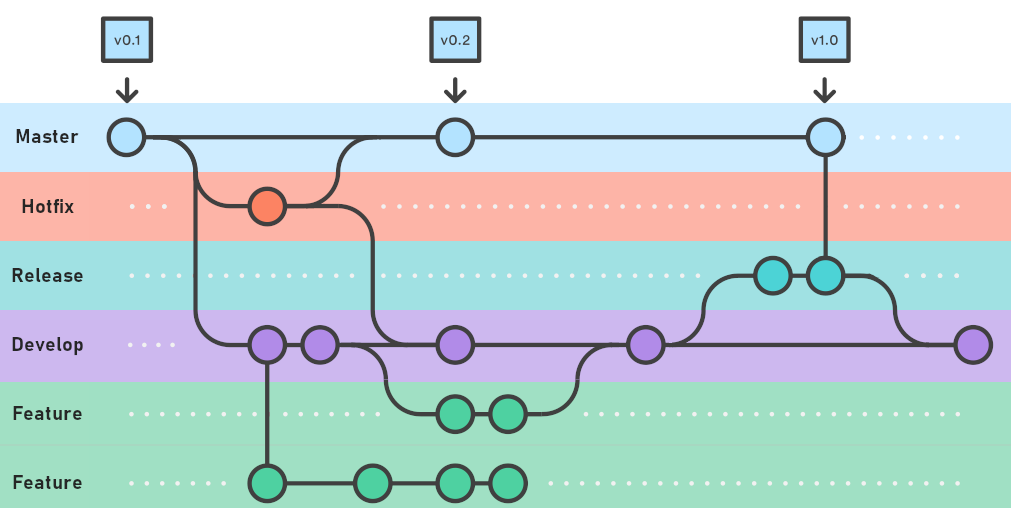
Os *Pull* *Requests* permitem que outras pessoas sejam informadas sobre as alterações das quais se fez *push* para um *branch* de um repositório no GitHub. Depois que um **Pull** *Request* é aberto, pode-se discutir e rever as possíveis alterações com colaboradores e adicionar *commits* de acompanhamento antes que as alterações sofram *merge* na *branch* base.

Figura 6 - Esquema de um conjunto de Branchs

# Arquitetura do Sistema

O Sistema é composto por duas principais plataformas: o website e a aplicação móvel. A arquitetura do sistema é delineada pelos seguintes componentes técnicos, descrevendo como interagem entre si:

* **Website:**
* Produtos: Responsável por gerir a apresentação de produtos, fornecendo informações detalhadas sobre cada item disponível na farmácia.
* Carrinho de Compras: Encarregue de gerir a adição, remoção e compra de produtos, mantendo uma interação eficaz com o Utilizador.
* Receitas Médicas: Responsável pela gestão das receitas médicas, permitindo aos Utilizadores visualizá-las.
* Faturas: Permite mostrar as transações efetuadas, proporcionando uma visão detalhada do histórico de compras.
* **Aplicação Móvel:**
* Produtos: Permite apresentar uma seleção de produtos disponíveis na farmácia, oferecendo informações essenciais aos Utilizadores.
* Carrinho de Compras: Responsável por gerir a adição, remoção e compra de produtos, proporcionando uma experiência de compra móvel eficiente.
* Faturas: Encarregue de apresentar um histórico de transações realizadas através da aplicação, possibilitando uma visão detalhada das compras efetuadas.
* Receitas Médicas: Responsável pela gestão de informações sobre receitas médicas, permitindo aos Utilizadores visualizá-las.

A comunicação entre as funcionalidades do website e da aplicação móvel são realizadas através de uma **API** que interage com o servidor Mosquitto e a Base de Dados. Esta comunicação garante a sincronização adequada das informações, proporcionando uma experiência de utilização consistente e facilitando a manutenção do sistema de forma coesa.

Uma imagem com círculo, diagrama, texto, captura de ecrã

Descrição gerada automaticamente

Figura 7 - Diagrama da Arquitetura do Sistema

# Metodologias

O planeamento foi realizado através da ferramenta Jira, utilizando a framework Scrum, com as devidas sprints com a duração de duas semanas, reuniões semanais e sprints com prazos a cumprir.

Para o planeamento do sistema, o Scrum foi utilizado para um melhor planeamento do que iria ser implementado durante as duas semanas de cada sprint, também foram atribuídos os papéis de cada um neste Sistema como é apresentado na *Tabela 1* e foram construídos wireframes, mockups e um diagrama Entidade-Relacionamento.

|  |  |  |
| --- | --- | --- |
| Cargos | Nome | Funções |
| Cliente | Professores e encarregados | * Utilizar a web app/app * Pedir a construção e os requisitos do mesmo |
| Utilizadores finais | Utilizadores | * Utilizar web app/App |
| Gestor de projeto | Tiago Silva | * Planeamento do projeto via Jira * Planear as Sprints * Garantir o comprimento das sprints |
| Analista(s) de sistema | Tiago Silva  Hugo Dias  Pedro Francisco | * Fazer testes ao longo do desenvolvimento de modo a verificar se está tudo a funcionar corretamente. * No final, dar feedback para saber se o website está a funcionar. |
| Programador(es) | Tiago Silva  Hugo Dias  Pedro Francisco | * Desenvolver a web app/app de acordo com as sprints definidas no Jira. * Testar web app/app |

Tabela 1- Identificação de stakeholder e equipa de desenvolvimento

O Jira também oferece uma série de recursos, relatórios e visualizações, incluindo gráficos de burndown, relatórios de progresso do projeto e relatórios de tendências. Isto permite ter uma visão clara do progresso do projeto e tomar decisões informadas sobre o próximo passo.

Uma imagem com texto, captura de ecrã, file, diagrama

Descrição gerada automaticamente

Figura 8 - Gráfico de burndown

|  |  |  |  |
| --- | --- | --- | --- |
| Nº | Inicio da Sprint | Fim da Sprint | Nº de tarefas |
| 1 | 03/10/2023 | 17/10/2023 | 10 |
| 2 | 17/10/2023 | 31/10/2023 | 3 |
| 3 | 31/10/2023 | 14/11/2023 | 6 |
| 4 | 14/11/2023 | 28/11/2023 | 4 |
| 5 | 28/11/2023 | 12/12/2023 | 5 |
| 6 | 12/12/2023 | 26/12/2023 | 4 |
| 7 | 26/12/2023 | 09/01/2024 | 6 |

Tabela 2 - Tabela de sprints

# Análise e Desenho

## Requisitos do Sistema

Os requisitos do sistema consistem nas especificações mínimas e recomendadas que um sistema deve conter para garantir o seu funcionamento adequado. Estes são os seguintes:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Role | Descrição detalhada | Back-office | Front-office | App mobile |
| Administrador | Gestão do backoffice:   * Gerir medicamentos * Gerir funcionários * Gerir vendas * Gerir stocks * Gerir fornecedores * Gerir receitas médicas * Gerir estabelecimentos * Gerir despesas | X  X  X  X  X  X  X  X |  | X  X |
|  |  |  |  |  |
| Cliente | Front-office:   * Comprar medicamentos * Ver as suas estatísticas (dinheiro gasto em compras, medicamentos comprados recentemente) * Ver as suas receitas médicas * Ver as suas faturas e carrinho de compras |  | X  X  X  X  X  X | X  X  X  X  X  X |
|  |  |  |  |  |

Tabela 3 - Roles dos diferentes end-users do SI

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Funcionário | Gestão do backoffice:   * Gerir medicamentos * Emitir faturas * Gerir vendas * Gerir stocks * Gerir fornecedores * Gerir receitas médicas * Gerir estabelecimentos * Gerir despesas | X  X  X  X  X  X  X  X |  | X |

|  |  |  |
| --- | --- | --- |
| # | Requisito | Prioridade |
| RF-FO-01 | O sistema deve possibilitar a venda de medicamentos e afins ao cliente. | Alta |
| RF-FO-02 | O sistema deve demonstrar ao cliente as suas receitas médicas e a validade das mesmas. | Média |

Tabela 4 - Requisitos funcionais Front-Office

|  |  |  |
| --- | --- | --- |
| # | Requisito | Prioridade |
| RF-BO-01 | O sistema deve possibilitar a gestão de funcionários ao administrador. | Alta |
| RF-BO-02 | O sistema deve possibilitar a emissão de faturas, onde irá mostrar os medicamentos comprados, o seu preço, com e sem iva, o subtotal e o total. | Alta |

Tabela 5 - Requisitos funcionais Back-Office

|  |  |  |
| --- | --- | --- |
| # | Requisito | Prioridade |
| RF-AM-01 | O sistema deve possibilitar a venda de produtos farmacêuticos ao cliente. | Alta |
| RF-AM-02 | O sistema deve apresentar ao cliente um carrinho de compras onde mostra todos os produtos adicionados ao mesmo. | Alta |
| RF-AM-03 | O sistema deve possibilitar a emissão de faturas, onde irá mostrar os medicamentos comprados, o seu preço, com e sem iva, o subtotal e o total. | Alta |

Tabela 6 - Requisitos funcionais App mobile

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | Requisito | Cliente | Administrador | Funcionário |  |
| RF-FO-01 | O sistema deve possibilitar a venda de medicamentos e afins ao cliente. | X |  | X |  |
| RF-FO-02 | O sistema deve apresentar ao cliente os últimos produtos adquiridos pelo mesmo. | X |  | X |  |

Tabela 7 - Cruzamento dos requisitos funcionais e respetivos roles

|  |  |  |  |
| --- | --- | --- | --- |
| # | Requisito | Tipo | Prioridade |
| RNF-01 | O sistema deve ser capaz de executar no Sistema Operativo Windows, acompanhado com uma aplicação Android. | Ambiente | Alta |
| RNF-02 | O sistema (Web App) deve ser desenvolvido com a linguagem de programação PHP (yii2). | Desenvolvimento | Alta |
| RNF-03 | O sistema (App Móvel) deve ser desenvolvido com as linguagens de programação Java e XML. | Desenvolvimento | Alta |

Tabela 8 - Requisitos Não funcionais

Para mais requisitos, verifique o anexo: [Anexo B](#_Anexos)

## User Stories

No que concerne aos Casos de Uso, estes possibilitam a descrição de funcionalidades do ponto de vista do Utilizador, incluindo a descrição de uma funcionalidade numa linguagem simples e compreensível para qualquer Utilizador. No âmbito deste projeto, os Casos de Uso definidos são os seguintes:

|  |  |
| --- | --- |
|  |  |
| Título: US1 – Adquirir medicamentos | **SP**: 5 |
| Descrição: Como cliente, quero poder adicionar produtos ao carrinho de compras, de modo a adquiri-lo posteriormente.  Critérios de Aceitação:   * Não pode ser adicionado um produto sem stock existente | |
| Título: US2 – Gerir cliente | **SP**: 3 |
| Descrição: Como funcionário, desejo criar e gerir contas de clientes, a fim de proporcionar um ambiente de utilizador mais agradável.  Critérios de Aceitação:   * Os campos nome, morada, contato e NIF têm de ser obrigatoriamente preenchidos. * Não pode haver um cliente com o mesmo NIF. | |

Para mais informações, verifique o anexo: [Anexo A](#_Anexos)

## Casos de Uso

Figura 9 - Diagrama Casos de Uso

Uma imagem com texto, diagrama, Esquema, Paralelo

Descrição gerada automaticamente

Este diagrama (*Figura 5*) ilustra os casos de uso no projeto da Carolo Farmacêutica.

Como evidenciado no diagrama acima, o Utilizador deve ter acesso a várias funcionalidades, incluindo visualizar os medicamentos disponíveis para venda, visualizar o seu carrinho de compras e permitir a adição, edição e remoção de produtos do seu carrinho de compras. Além disso, pode visualizar as suas receitas médicas declaradas, visualizar os serviços disponíveis em cada estabelecimento da Carolo Farmacêutica, visualizar as categorias dos produtos farmacêuticos onde se encontram inseridos, visualizar as faturas emitidas pelas suas compras e realizar o registo e login.

Os funcionários têm acesso a diferentes funcionalidades, como gerir produtos (inserir, atualizar e apagar), gerir categorias (inserir, atualizar e apagar), gerir serviços (inserir, atualizar e apagar), visualizar todas as receitas médicas, visualizar a lista de estabelecimentos, gerir despesas (inserir, atualizar e remover), visualizar e apagar as faturas emitidas, e visualizar ivas.

O administrador possui todas as permissões e acesso a todas as funcionalidades do sistema, incluindo a gestão de utilizadores, medicamentos, categorias, serviços, fornecedores, receitas médicas, estabelecimentos, despesas, faturas e ivas.

A **API** é um elemento essencial para o funcionamento do sistema, permitindo a interligação com a Base de Dados para realizar o armazenamento de todos os dados que envolvam nas funcionalidades implementadas e o acesso aos recursos necessários para o correto funcionamento do sistema.

## Modelo de Dados

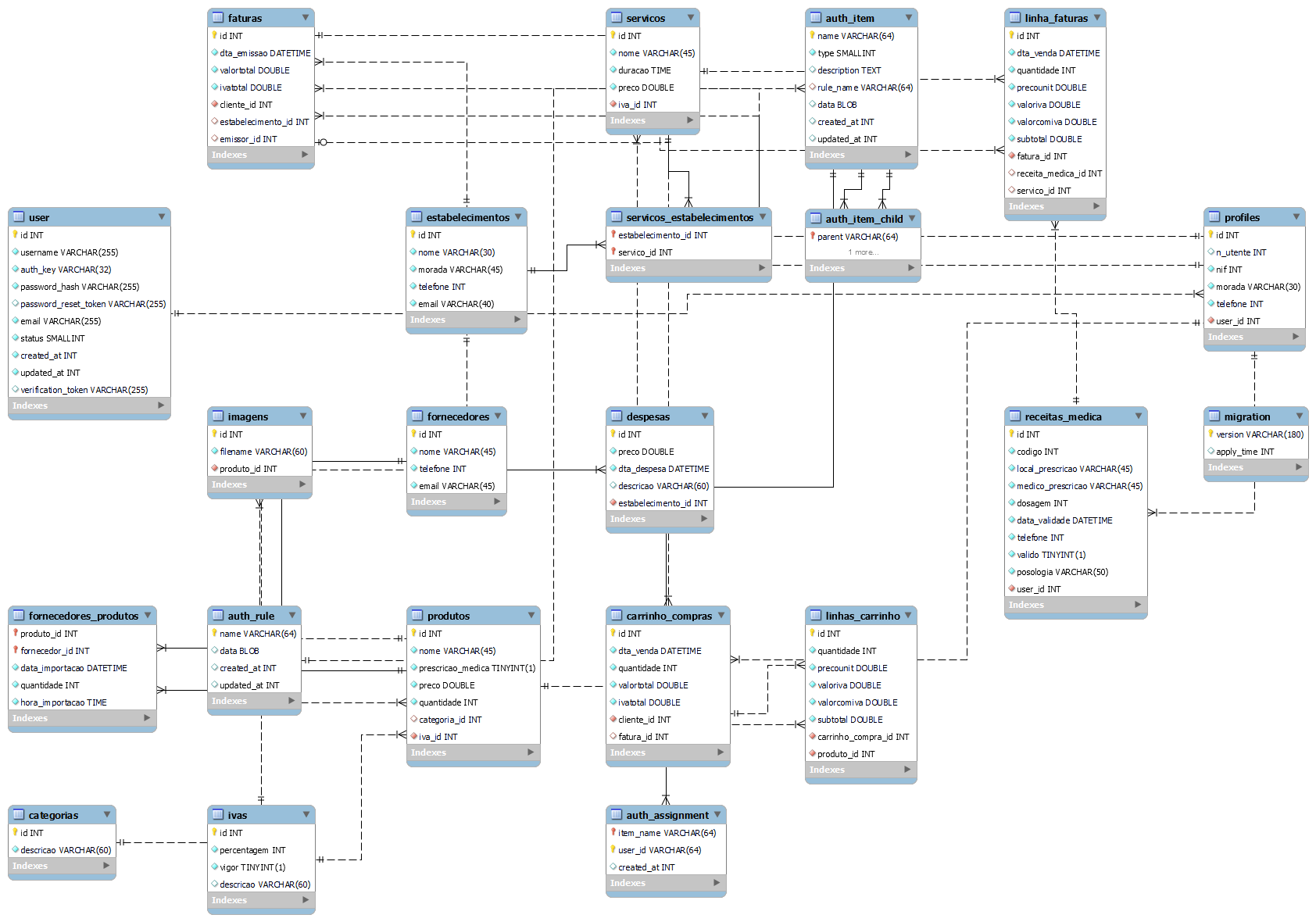


Figura 10 - Modelo de Dados (Base Dados)

A imagem apresentada anteriormente representa o modelo de dados da base de dados do projeto, que é composta por diversas tabelas. Estas tabelas desempenham papéis específicos no armazenamento e organização dos dados necessários para o correto funcionamento do sistema. A seguir, são detalhadas as tabelas presentes no modelo:

A tabela **user** destina-se a armazenar os dados de acesso de cada Utilizador. A tabela **profiles** guarda as informações relacionadas aos dados pessoais de cada Utilizador. A tabela **faturas** armazena as faturas declaradas por cada Utilizador, enquanto a tabela **linha\_faturas** contém os itens específicos de cada fatura.

A tabela **estabelecimentos** armazena os dados relativos aos estabelecimentos existentes no sistema. A tabela **servicos** regista as informações sobre os serviços disponíveis, e a tabela **servicos\_estabelecimentos** relaciona os serviços disponíveis em cada estabelecimento.

Para mais informações, verifique o anexo: [Anexo D](#_Anexos)

## Mockups/wireframes

Os mockups/wireframes planeados para a implementação do projeto foram os seguintes layouts representados abaixo:

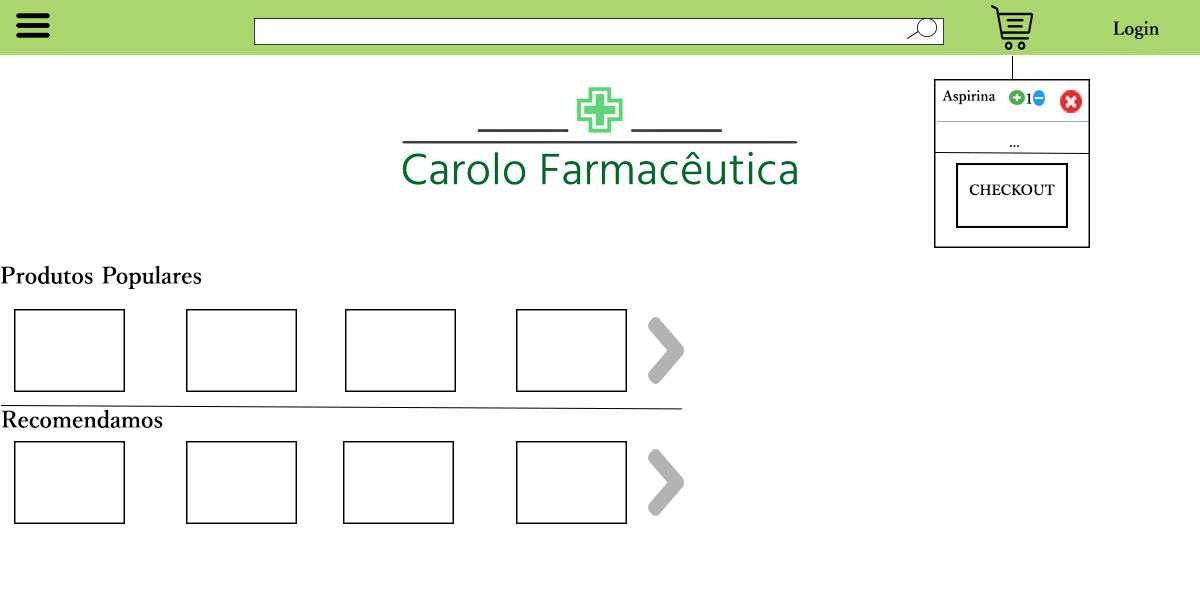
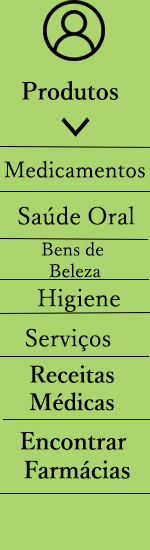
### WebSite

Uma imagem com texto, captura de ecrã, diagrama, Tipo de letra

Descrição gerada automaticamente

Figura 11 - Wireframe ecrã login

Figura 12 - Wireframe ecrã homepage



Uma imagem com texto, captura de ecrã, diagrama, ecrã

Descrição gerada automaticamente

Figura 13 - Wireframe ecrã checkout (Carrinho de Compras)

Figura 14 - Wireframe ecrã após checkout (Dados de Entrega)

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

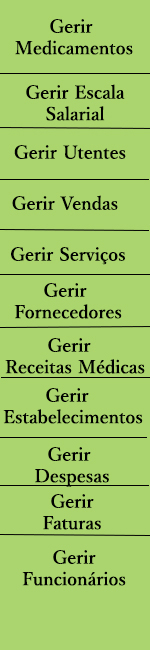
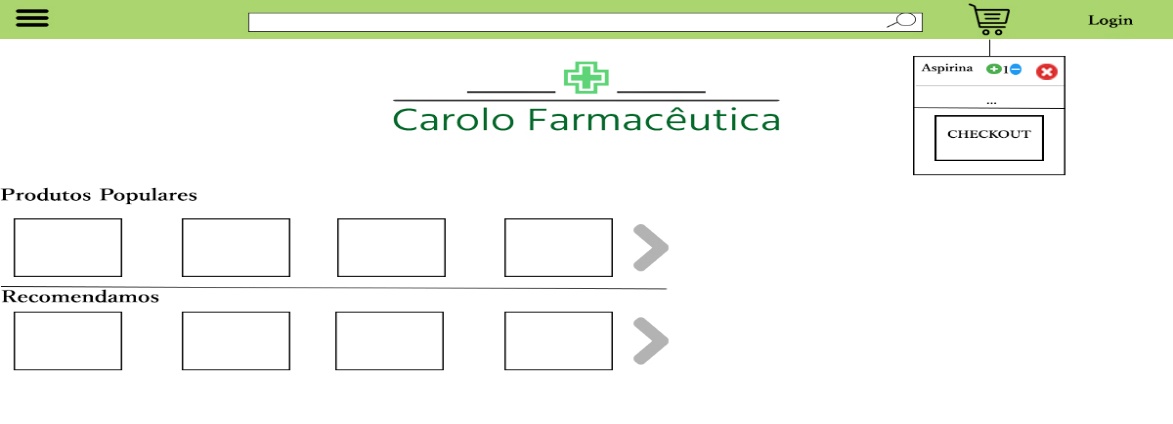


Figura 15 - Wireframe ecrã homepage (admin)

Uma imagem com texto, captura de ecrã, ecrã, diagrama

Descrição gerada automaticamente

Figura 16 - Wireframe ecrã faturas

Figura 17 - Wireframe ecrã gerir funcionários

Uma imagem com texto, captura de ecrã, ecrã, diagrama

Descrição gerada automaticamente

Uma imagem com texto, captura de ecrã, diagrama, ecrã

Descrição gerada automaticamente

Figura 18 - Wireframe ecrã gerir medicamentos

Uma imagem com texto, captura de ecrã, ecrã, diagrama

Descrição gerada automaticamente

Figura 19 - Wireframe ecrã gerir receitas médicas

### Aplicação Móvel

Figura 20 - Wireframe Janela Inicial

Uma imagem com texto, captura de ecrã, software, Página web

Descrição gerada automaticamente

Atividade Página Principal

Atividade Login

Figura 21 - Wireframe Receitas Médicas

Uma imagem com texto, captura de ecrã, diagrama, Retângulo

Descrição gerada automaticamente

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamente

Figura 22 - Wireframe slidebar

Uma imagem com texto, captura de ecrã, software, Sistema operativo

Descrição gerada automaticamente

Figura 23 - Wireframe Carrinho de Compras

Uma imagem com texto, captura de ecrã, software, Ícone de computador

Descrição gerada automaticamente

Figura 24 - Wireframe pagamento (multibanco)

Para mais informações, verifique o anexo: [Anexo C](#_Anexos)

# Implementação

## WebSite

Na unidade curricular Plataformas de Sistemas de Informação foi desenvolvida a aplicação web usando o Yii2 Advanced com a arquitetura MVC.

### Yii2

Yii2 é um framework PHP para desenvolvimento de aplicações web. É baseado no padrão de arquitetura MVC e fornece uma estrutura sólida para a criação de aplicações web avançados.

O Yii2 é uma ferramenta poderosa e flexível, que oferece muitos recursos ajudando os desenvolvedores a criar aplicações de alta qualidade de maneira rápida e eficiente.

### MVC

O MVC é um padrão de arquitetura de software amplamente utilizado na programação de aplicações. O objetivo do MVC é separar os componentes de uma aplicação em três partes distintas, cada uma com as seguintes responsabilidades:

#### Model (Modelo)

Representa a camada de dados e a lógica de negócios da aplicação. É responsável por armazenar e gerir informações, como por exemplos as informações da base de dados.

#### View (Vista)

Representa a camada de apresentação, ou seja, o que o utilizador vê. É responsável por armazenar e gerir informações, como por exemplo as informações da base de dados.

#### Controller (Controlador)

É a camada intermediária que coordena a interação entre o Modelo e a Vista. Recebe input do utilizador, manipula dados com base na lógica de negócios (Modelo) e atualiza a Vista com os resultados.

O benefício de usar o padrão MVC reside na sua capacidade de permitir que as equipas de desenvolvimento colaborem de forma mais eficaz. É mais eficiente porque cada componente da aplicação é responsável por uma tarefa específica. Além disso, se possível, facilitará a manutenção e o desenvolvimento de aplicações, uma vez que alterações podem ser feitas em uma classe sem afetar outras classes.

### Backend

O *backend* é uma componente da aplicação web que se concentra em tarefas relacionadas com o servidor, como armazenamento e manipulação dos dados, processamento de pedidos e geração de respostas. Geralmente, o *backend* é "invisível" para o utilizador final e executa tarefas que não requerem interação direta com o Cliente.

No *backend* da Carolo Farmacêutica, os Utilizadores com acesso incluem os administradores e os funcionários. Estes utilizadores têm a capacidade de realizar a gestão de clientes, ivas, estabelecimentos, medicamentos, categorias, serviços, fornecedores, receitas médicas e despesas. Outra funcionalidade importante trata-se da emissão de faturas de serviços e receitas médicas e a gestão da mesma.

Uma imagem com texto, captura de ecrã, software, Ícone de computador

Descrição gerada automaticamente

Figura 25 - Página inicial Backend

Uma imagem com texto, captura de ecrã, software, Ícone de computador

Descrição gerada automaticamenteUm exemplo da gestão da parte administrativa é a gestão de produtos, onde é possível criar um produto, visualizá-lo em detalhe, editá-lo, removê-lo e adicionar imagens ao mesmo.

Figura 26 - Gestão de produtos

A gestão das faturas é a funcionalidade que mais se difere das outras, onde para criar uma fatura primeiramente são escolhidos o cliente e o estabelecimento, após isso é possível selecionar um serviço ou uma receita médica, depois irá aparecer a página onde são criadas/emitas as faturas onde tem a opção de alterar a quantidade dos serviços, eliminar uma linha, adicionar uma linha ou concluir a fatura.

Uma imagem com texto, software, Ícone de computador, Software de multimédia

Descrição gerada automaticamente

Figura 27 - Seleção de cliente e estabelecimento

Uma imagem com texto, software, Página web, Ícone de computador

Descrição gerada automaticamente

Figura 28 - Inserção de serviços ou receitas

Uma imagem com texto, software, Ícone de computador, Página web

Descrição gerada automaticamente

Figura 29 - Ecrã de emissão da fatura

### Frontend

O *frontend*, também conhecido como "client-side", diz respeito à componente de uma aplicação web encarregue de fornecer a interface do utilizador e interagir com ele. É responsável por apresentar a informação de forma atrativa e intuitiva, possibilitando que o utilizador realize ações como preencher formulários, selecionar opções ou navegar entre páginas.

Uma imagem com texto, captura de ecrã

Descrição gerada automaticamente

Figura 30 - Página inicial do frontend

No *frontend* da Carolo Farmacêutica, todos os Clientes têm acesso ao *frontend*, onde é possível ver os medicamentos disponíveis e adicioná-los ao carrinho de compras. Este medicamento também pode ter receita médica, se for esse o caso para adicionar ao carrinho de compras é necessário ao cliente ter acesso à receita e inserir o código da mesma.

Uma imagem com texto, captura de ecrã, Tipo de letra, software

Descrição gerada automaticamente

Figura 31- Ecrã de inserção de receita-médica

Cada medicamento tem uma página de detalhes com a quantidade disponível, se tem receita médica, o preço, a categoria e as imagens do produto.

Uma imagem com texto, captura de ecrã, Marca, design gráfico

Descrição gerada automaticamente

Figura 32- Página detalhes de um medicamento

Para encontrar um medicamento, podemos utilizar a página inicial e procurar o medicamento desejado, ou pode-se procurar por categoria via o menu produtos.

Uma imagem com texto, captura de ecrã, refrigerante, design

Descrição gerada automaticamente

Uma imagem com texto, captura de ecrã, Tipo de letra, Página web

Descrição gerada automaticamente

Figura 33- Medicamentos - página inicial

Figura 34- Menu produtos

Após os medicamentos estarem inseridos no carrinho de compras, é possível removê-los do carrinho e atualizar a quantidade do medicamento (só possível mudar a quantidade se não tiver receita). Após clicar em concluir carrinho, é possível inserir os dados de entrega, e escolher o método de pagamento (multibanco, cartão de crédito).

Figura 35 - Dados de entrega

Uma imagem com texto, captura de ecrã, número, Tipo de letra

Descrição gerada automaticamente

O cliente após a compra, se desejar ver a fatura da mesma, tem de aceder ao menu do cliente, na secção de faturas e clicar em uma fatura para a visualizar, onde também tem a opção de imprimir a fatura em pdf.

Uma imagem com texto, captura de ecrã, software, Tipo de letra

Descrição gerada automaticamente

Figura 36- Menu cliente - faturas

Uma imagem com texto, captura de ecrã, número, software

Descrição gerada automaticamente

Figura 37- Página Faturas

Figura 38- Fatura

Uma imagem com texto, Tipo de letra, file, captura de ecrã

Descrição gerada automaticamente

Figura 39- Fatura Impressa

Uma imagem com texto, captura de ecrã, Tipo de letra, número

Descrição gerada automaticamente

Para o cliente ver as suas receitas médicas, tem de clicar no menu de cliente e no item receitas médicas.

Uma imagem com texto, Tipo de letra, número, software

Descrição gerada automaticamente

Figura 40- Receita médica - cliente

## Aplicação Mobile

Na Unidade Curricular Acesso móvel a Sistemas de Informação e Serviços foi desenvolvida a app mobile usando o Android Studio com a linguagem de programação **Java**.

### Android Studio

O Android Studio é um ambiente de desenvolvimento integrado desenvolvido pela Google para a plataforma Android. É utilizado por desenvolvedores para criarem aplicações para dispositivos Android, como telemóveis, tablets entre outros.

O **Java** é uma linguagem de programação de alto nível, amplamente utilizada para desenvolver aplicações em várias plataformas, incluindo Android. A plataforma Android suporta o desenvolvimento de aplicações em **Java** e o Android Studio com o **Java** permite aos desenvolvedores criarem aplicações robustas e avançadas para a plataforma Android.

O Android Studio oferece uma ampla gama de recursos para ajudar os desenvolvedores a criarem aplicações Android, incluindo uma interface de utilizador visual, ferramentas de depuração, recursos de gestão de versões, integração com serviços do Google e muito mais.

Uma imagem com texto, captura de ecrã, Tipo de letra, design

Descrição gerada automaticamenteAlém disso, o Android Studio é compatível com outras linguagens de programação, como o **Kotlin**.

Figura 41 - Login da app mobile

A aplicação mobile da Carolo Farmacêutica está apenas disponível para Clientes, o poderá ver os medicamentos existentes para compra, as suas receitas médicas e as suas respetivas faturas e também pode adquirir os medicamentos.

Uma imagem com texto, captura de ecrã

Descrição gerada automaticamente

Figura 42 - Janela Inicial

A Janela Inicial permite apresentar todos os produtos disponíveis para venda, disponibilizando ao Utilizador as seguintes ações: um botão de lupa para pesquisa do produto pretendido, a opção do menu de *overflow*(“três pontinhos) que permite apresentar um menu de itens no canto superior direito, cujas opções são as categorias existentes dos produtos, de forma a serem visualizados ao selecionar uma opção, e ao selecionar apenas um item pretendido pelo Utilizador, que encaminhará para uma nova janela onde mostrará os detalhes desse produto.

Uma imagem com texto, captura de ecrã, software, Página web

Descrição gerada automaticamente

Figura 43 - Lista de receitas médicas

Nesta Janela, apresentam-se todas as receitas médicas existentes do Cliente, neste caso, serão apresentadas as receitas atribuídas ao Utilizador com a sessão iniciada no momento. Estão disponíveis as seguintes ações: a ação da lupa, que permite pesquisar a receita médica pretendida através do código da mesma, e ainda uma lista que apresenta de forma clara ao Utilizador as suas receitas, de modo que o mesmo possa selecionar apenas um item na lista, o qual o encaminhará para uma nova janela, apresentando mais detalhes da receita médica selecionada.

Uma imagem com texto, captura de ecrã, software, Sistema operativo

Descrição gerada automaticamente

Figura 44 - Carrinho de Compras

A Janela do Carrinho de Compras permite ao Utilizador visualizar o carrinho com as suas compras que pretende realizar, disponibilizando as seguintes funcionalidades: a opção de eliminar o produto do carrinho, a opção de subtrair para diminuir a quantidade pretendida do produto e a opção de adicionar mais quantidade ao produto que pretende comprar. Para finalizar a sua compra, o Utilizador deve clicar no botão Checkout para concluir a sua compra e automaticamente gera a sua fatura. A fatura emitida pode ser consultada a qualquer momento na secção de Faturas da aplicação.

|  |  |
| --- | --- |
| Uma imagem com texto, captura de ecrã, software, Página web  Descrição gerada automaticamente  Figura 45 - Opção de Pagamento | Uma imagem com texto, captura de ecrã, software, Página web  Descrição gerada automaticamente  Figura 46 - Opção pagamento com cartão |
| Uma imagem com texto, captura de ecrã, software, Página web  Descrição gerada automaticamente  Figura 47 – Opção de pagamento com multibanco | |

A Janela de pagamento permite ao Utilizador escolher o método de pagamento desejado para efetuar o pagamento das suas compras. Na *figura 42*, representa-se uma caixa de diálogo para o Utilizador, permitindo a seleção do método de pagamento desejado. Se for escolhida a opção de Pagar com Cartão de Crédito (*figura 43*), serão apresentados campos de preenchimento obrigatório. Por último, se a opção escolhida for Pagar com Multibanco, serão geradas as referências de pagamento e apresentadas ao Utilizador para serem utilizadas num terminal de Multibanco, a fim de efetuar o pagamento das suas compras. Após a seleção do método de pagamento desejado, estarão disponíveis as opções Não e Sim. A opção NÃO impede o prosseguimento com a compra, enquanto a opção SIM permite concluir a compra.

## API

Na Unidade Curricular Interoperabilidade de Sistemas foi desenvolvida a **API** usando o Yii2.

Uma imagem com texto, captura de ecrã, Tipo de letra

Descrição gerada automaticamenteCom Yii2, pode-se criar uma **API** RESTful, que é uma **API** que segue o padrão REST. Esta situação permite o acesso aos dados da sua aplicação de forma fácil, usando as operações HTTP, como GET, POST, PUT e DELETE.

Figura 48 - Exemplo de um pedido à API

# Testes

Para garantir a consistência dos resultados e a usabilidade pelo Utilizador, foram desenvolvidos testes autónomos que avaliam as principais funcionalidades do Sistema (website).

Quanto à consistência da aplicação móvel, foram realizados testes manuais, nos quais cada um dos desenvolvedores testava as funcionalidades implementadas na aplicação móvel.

### Testes Unitários

Foram testadas as regras implementadas no modelo para validar os dados de cada um dos campos, de modo a obter um resultado positivo e negativo para cada um deles como referência.

|  |  |
| --- | --- |
| Testes Unitários | |
| Estabelecimentos | Criar/Ler/Editar/Remover |
| Fornecedores | Criar/Ler/Editar/Remover |
| Categorias | Criar/Ler/Editar/Remover |
| Despesas | Criar/Ler/Editar/Remover |
| Ivas | Criar/Ler/Editar/Remover |

Tabela 9 - Tabela de testes unitários

### Testes Funcionais

Foram implementados cinco testes funcionais, sendo um deles um login, enquanto os restantes foram registos nas funcionalidades mais importantes do projeto.

|  |
| --- |
| Testes Funcionais |
| Login da parte da administração |
| Inserção de um cliente em ambiente administrativo |
| Emissão de fatura de um serviço e/ou receita médica a um cliente |
| Login na parte de Clientes |
| Compra de um medicamento com e sem receita médica |

Tabela 10 - Tabela de testes funcionais

### Testes de Aceitação

Foi implementado um teste de aceitação que fez várias ações a algumas das

funcionalidades mais importantes do projeto.

|  |
| --- |
| Testes de Aceitação |
| 1. Criar conta 2. Login 3. Inserir dados adicionais (perfil do utilizador) 4. Logout 5. Entrar na parte administrativa 6. Criar receita para o novo utilizador 7. Logout da parte administrativa 8. Login novamente com o novo utilizador 9. Adicionar o produto com a receita ao carrinho 10. Adicionar o produto sem receita ao carrinho 11. Concluir carrinho 12. Checkout |

Tabela 11 - Tabela de testes de aceitação

### Execução dos testes

Para a execução dos testes funcionais e unitários, apenas e necessários dois comandos, onde para executar o teste funcional o comando é o seguinte:

“php vendor/bin/codecept run functional --steps -c *frontend*“ – este comando executa todos os testes funcionais do *frontend*, caso queria executar os testes da parte administrativa, é necessário substituir “*frontend*” por *backend*.

Já para executar os testes unitários o comando é o seguinte:

“php vendor/bin/codecept run unit --steps -c *backend* “ – este comando executa todos os testes unitários do *backend*, caso queria executar os testes da parte comum, é necessários substituir “*backend*” por common.

Por fim para os testes aceitação o comando é o seguinte:

“php vendor/bin/codecept run acceptance --steps -c *frontend*” – é necessário antes de executar este comando, abrir a linha de comandos, entrar na pasta do projeto, depois na pasta do selenium e executar o seguinte comando:

“java -jar selenium-server-4.16.1.jar standalone”

### Resultados obtidos e correção de erros

Após a execução dos testes mencionados anteriormente, verifica-se que as funcionalidades testadas estão a funcionar corretamente, e no final, surge a seguinte mensagem:



No que diz respeito à correção de erros, durante o desenvolvimento dos testes deparámo-nos com falhas que nos motivaram a corrigi-las, com o objetivo de tornar o sistema mais completo e sem falhas para o Utilizador.

# Conclusões ou Conclusão

Na nossa perspetiva, os objetivos delineados para o desenvolvimento deste Sistema foram, de um modo geral, claros e bem definidos e, na nossa opinião, cumpridos com sucesso. Conseguimos tirar o máximo proveito e adquirir conhecimentos e competências não só na vertente profissional, mas também no âmbito pessoal e na capacidade de trabalhar em equipa.

O desenvolvimento do Sistema foi gradual, permitindo a implementação das funcionalidades consideradas mais importantes. Este progresso foi possível devido à distribuição organizada do trabalho entre os membros do grupo, o que facilitou o desenvolvimento do Sistema.

Como finalistas do Curso Técnico Superior Profissional em Programação de Sistemas de Informação, com o desenvolvimento deste projeto, pretendemos demonstrar autonomia, criatividade e capacidade. Apesar de todas as dificuldades encontradas, que foram sendo superadas, acreditamos que o principal objetivo de aplicar e desenvolver as competências adquiridas ao longo deste curso foi alcançado, tendo sido uma experiência muito gratificante tanto a nível pessoal como profissional.

# Bibliografia ou Referências Bibliográficas

*PHP: Hypertext Preprocessor*. (s.d.). Obtido de https://www.php.net/

*Selenium*. (s.d.). Obtido de https://www.selenium.dev/

*Stack Overflow*. (s.d.). Obtido de https://stackoverflow.com/

*Yii PHP Framework*. (s.d.). Obtido de https://www.yiiframework.com/

# Glossário

Base de Dados

As bases de dados são conjuntos de ficheiros/tabelas relacionados entre si onde contêm registos. As mesmas são organizadas e relacionam-se de forma a organizar a informação e dar mais eficiência a uma pesquisa.

API

Uma API define métodos e padrões para a comunicação entre diferentes softwares. Esta permite que um programa solicite e receba informações ou execute ações em outro programa, serviço ou sistema de maneira eficiente e consistente.

# Anexos

