

Laboratório em Engenharia Informática

UNIVERSIDADE DO MINHO

GESTOR WEB DE CVS E EPORTEFOLIOS

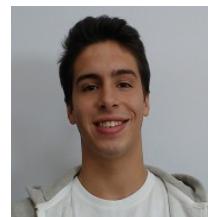
Projeto 56



Ricardo Costa
a85851



Rita Rosendo
a84475



Rui Oliveira
a83610

December 20, 2022

Contents

1	Introdução	2
2	Arquitetura	2
3	Base de Dados	4
4	Backend	4
5	Frontend	8
5.1	Estrutura do projeto Vue	8
5.2	Funcionalidades implementadas	10
5.3	Componentes Vue	11
6	Documentação da API	12
7	Docker	12
8	Conclusão	14
9	Referências bibliográficas	14
10	Anexos	15

1 Introdução

No âmbito da Unidade Curricular de Laboratório em Engenharia Informática, foi-nos proposta a elaboração de um projeto que consiste na criação de um Gestor Web de CVs e ePortefolios com base no *EuroPass*.

Como recursos tecnológicos foram usadas as seguintes ferramentas: o *MongoDB* para suporte da base de dados, o *Strapi* como gerador da API de dados e o *Vue* como gerador da interface.

2 Arquitetura

Para a realização do projeto tivemos de, inicialmente, direcionar a nossa atenção relativamente à arquitetura do sistema, isto é, decidir que modelo de dados usar assim como o que utilizar para suportar o *backend* e o *frontend*.

Primeiramente, como proposto pelo professor José Carlos Ramalho, decidimos utilizar como suporte do modelo de dados uma base de dados *NoSQL*. Esta decisão foi tomada devido a diversas razões. Numa base de dados *SQL* os dados são armazenados em tabelas relacionadas entre si e devem seguir uma mesma estrutura, porém, numa base de dados *NoSQL*, os dados podem ser guardados de diversas formas (quer seja orientado a coluna, ao documento, baseado em grafos ou organizados como chave-valor) e não é necessário seguir uma estrutura inicial o que dá mais liberdade, velocidade e flexibilidade para alterar o esquema a qualquer momento para satisfazer os requisitos. Outra razão é o facto das bases de dados *NoSQL* conseguirem processar quantidades enormes de dados não estruturados rapidamente e terem capacidade para responder a situações não planeadas permitindo um desenvolvimento mais rápido e ágil de aplicações. Por fim, apesar de inicialmente ponderarmos a utilização do **ArangoDB**, de entre as diversas opções de bases de dados *NoSQL*, decidimos escolher o **MongoDB** por esta ser suportada pelo **Strapi**, ferramenta que abordaremos já de seguida. Além disso, o facto de já termos usado no passado e considerarmos que responderia de forma excelente ao que pretendemos, foi também um fator decisivo.

Para o *backend*, decidimos utilizar o **Strapi**, ferramenta proposta pelo professor, que permite gerar API's de dados de forma relativamente simples e rápida. Trata-se de uma ferramenta *open-source*, desenvolvida em *nodejs* e de fácil uso com bastante documentação disponível na sua página principal. Para além de facilitar bastante a criação de uma API de dados, uma vantagem

do *Strapi* é o facto da funcionalidade de autenticação já estar implementada.

Como suporte da *frontend*, decidimos utilizar o **Vue** que é um *framework Javascript open-source* para gerar interfaces. Esta é uma ferramenta que requer uma configuração mínima na criação de um projeto e que pode ser facilmente integrada com uma aplicação já existente. Trata-se também de uma ferramenta com uma baixa curva de aprendizagem e com uma documentação oficial bastante extensa. Além disso, disponibiliza uma incrível biblioteca *UI* chamada *Vuetify*, que fornece todos os componentes necessários para a estilização de aplicações.

Por fim, podemos expor a arquitetura do sistema através da seguinte figura:

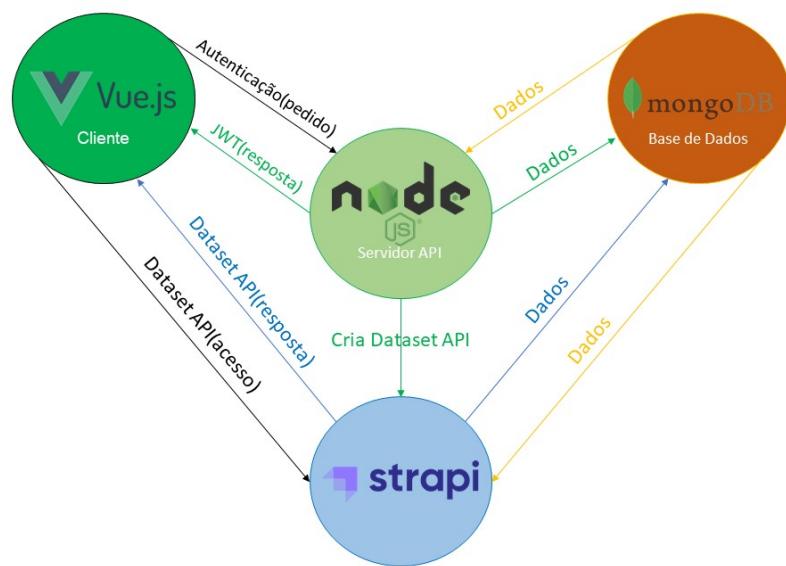


Figure 1: Arquitetura

3 Base de Dados

Tal como foi dito anteriormente, a base de dados foi implementada em **MongoDB**.

Nesta base de dados, existem vários tipos de documentos: usernames, tipo, exp_prof, eportefolio, endereço, edu_form e comp_pess.

- **Usernames:** documento que contém toda a informação relativa aos utilizadores
- **Tipo:** documento que contém a informação que possuí um tipo que não se enquandra num outro documento, isto é, um tipo de dados novo a inserir na base de dados
- **Exp_prof:** documento que contém a informação relativa à experiência profissional de cada utilizador
- **Eportefolio:** documento que contém a informação relativa ao eportefolio de cada utilizador
- **Endereço:** documento que contém toda a informação do endereço de cada utilizador
- **Edu_form:** documento que contém a informação relativa à formação educacional de cada utilizador;
- **Comp_pess:** documento que contém toda a informação relativa às competências pessoais de cada utilizador

4 Backend

Para a geração da API de dados utilizamos o **Strapi**, que poupará o trabalho relativo ao *backend*.

Primeiramente, após criarmos o projeto *strapi* onde escolhemos qual a base de dados que queremos usar assim como outras funcionalidades, foi necessário criar um utilizador administrador que contém o acesso ao *backend* e, portanto, contém os privilégios necessários para modificar as estruturas de dados do *strapi* assim como o adição de conteúdo, por exemplo.

Posteriormente, criamos as estruturas de dados para manipular os dados presentes na base de dados. Para tal, recorremos ao plugin *Content-type builder*. Desta forma, as seguintes coleções tipo foram criadas:

- **Tipo:** Esta coleção possui como campos os atributos que fazem parte da coleção de dados "tipo" da base de dados.
 - nome
 - campo_int
 - campo_txt
 - eportefolio, que armazena o id do eportefolio a que pertence
- **User:** Esta coleção possui como campos os atributos que fazem parte da coleção de dados "usernames" da base de dados.
 - username
 - email
 - password
 - eportfolios, que armazena o id dos eportfolios do user
 - cvs, adicionados ou criados pelo user
 - certificados, adicionados pelo user
 - cartas, que armazena a carta de apresentação do user
 - outros, que armazena outros tipos de documentos que o user ache pertinente acrescentar ao seu eportfólio
 - feed, que armazena a informação das operações do user no sistema
- **Competências Pessoais** Esta coleção possui como campos os atributos que fazem parte da coleção de dados "comp_pess" da base de dados.
 - carta de condução
 - língua materna
 - outra língua
 - competências digitais
 - descrição

- eportefolio, que armazena o id do eportefolio a que pertence
- **Educação:** Esta coleção possui como campos os atributos que fazem parte da coleção de dados ”edu_form” da base de dados.
 - título
 - organização
 - em curso, booleano que indica se o user ainda está a frequentar o ensino ou não
 - área
 - classificação final
 - tese
 - assunto principal
 - qeq, que armazena o nível a que pertence ao quadro europeu de qualificações
 - classificação nacional
 - tipo de créditos
 - número de créditos
 - válido até
 - sítio web
 - hiper
 - descrição
 - data, que armazena as datas de início e fim do percurso académico
 - endereço, que armazena o id do endereço a que pertence onde contém o endereço detalhado do estabelecimento de ensino frequentado
 - eportefolio, que armazena o id do eportefolio a que pertence
- **Experiência Pessoal:** Esta coleção possui como campos os atributos que fazem parte da coleção de dados ”exp_prof” da base de dados.
 - função

- entidade empregadora
 - em curso, booleano que indica se o user ainda trabalha no estabelecimento em questão
 - competências digitais
 - descrição
 - eportefolio, que armazena o id do eportefolio a que pertence
- **Eportefolio:** Esta coleção possui como campos os atributos que fazem parte da coleção de dados ”eportefolio” da base de dados.
- nome
 - género
 - nacionalidade
 - email
 - telemóvel
 - profissão
 - user, que armazena o id do user a que pertence
 - endereço, que armazena o id do endereço que o user possui
 - tipos, que armazena o id da coleção tipos que o user possui
 - trabalhos, que armazena os ids das coleções de experiência pessoal que o user possui
 - educações, que armazena os ids das coleções de educação que o user possui
 - competências pessoais, que armazena os ids das coleções de competências pessoais que o user possui
 - avatar
 - data de nascimento

No que diz respeito às rotas de acesso à informação, o *strapi* oferece o plugin *Roles & Permissions* onde se pode verificar quais as rotas que existem para cada pedido. De notar que, para um utilizador poder fazer pedidos utilizando essas rotas e ter acesso às respostas, o utilizador administrador

criado no início da criação do projeto *strapi* necessita de, antes de mais, definir as permissões necessárias.

Por fim, o **Strapi** através do plugin de autenticação facilita também a mesma, retornando o json web token(*JWT*) ao *frontend* em cada pedido de autenticação, garantindo assim uma ao cliente uma sessão segura dentro da aplicação.

5 Frontend

Tal como foi referido anteriormente, para o desenvolvimento do *frontend*, sendo este responsável por interagir diretamente com o utilizador, utilizamos a *framework* **Vue**.

5.1 Estrutura do projeto Vue

O nosso projeto *vue* divide-se, principalmente, nas seguintes partes: *helpers*, *services*, *store* e *views*.

A diretoria *helpers* contém os headers necessários para utilizar nos pedidos feitos ao *backend* assim como a costumização das rotas (informação presente no ficheiro *router.js*). Neste caso, temos dois tipos de *headers*, *auth-header.js* e o *special_auth-header.js*. O *auth-header.js* é utilizado apenas em pedidos em que apenas é necessário passar o *JWT* como o caso da remoção de uma conta de utilizador ou a criação, edição e remoção de um eportefolio, por exemplo. Por sua vez, o *special_auth-header.js* é utilizado em pedidos onde, para além do *JWT*, é necessário definir o *Content-Type*.

A pasta *services* contém as funções responsáveis por comunicar com o *backend*. Nesta comunicação com o *backend* é utilizada a API *fetch* que fornece uma interface para buscar recursos. O método *fetch()* tem como argumento obrigatório o caminho para o recurso que queremos obter.

A pasta *store* é a ”ponte” entre a comunicação com o *backend* e o agrupamento da informação que vem das páginas *vue*. Neste caso, é utilizado o *Vuex* que é um gestor do estado da aplicação + biblioteca e que serve como um armazenamento centralizado para todos os componentes da aplicação com regras que garantem que o estado só pode ser alterado de maneira previsível.

Por fim, a pasta *views* contém os componentes *vue* necessários para o desenvolvimento das páginas da aplicação.

Podemos observar com clareza a estrutura do frontend na figura seguinte:

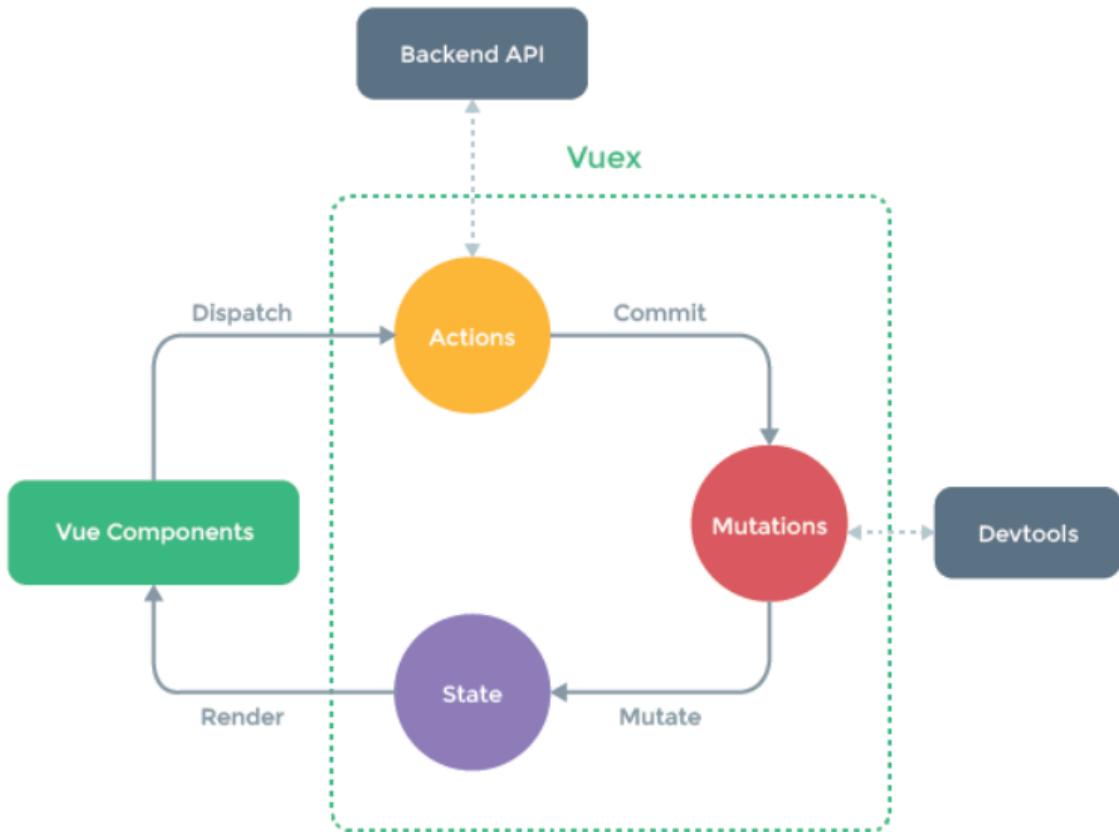


Figure 2: Estrutura do projeto vue

5.2 Funcionalidades implementadas

Nesta secção, procuramos seguir a estrutura do site [Europass](#) e encontrar possíveis inovações. Assim, decidimos implementar as seguintes funcionalidades:

- Login
- Logout
- Registar utilizador
- Editar e remover conta
- Criar, editar e remover um ePortefolio
- Ver informações do ePortefolio
- Adicionar novos tipos de informação aos dados do ePortefolio
- Converter um ePortefolio para CV (PDF)
- Visualizar, apagar, criar, fazer download e upload de CVs
- Biblioteca com todos os documentos do utilizador
- Adicionar diferentes tipos de documentos/certificados de atividade profissional e outros tipos de documentos na biblioteca
- Histórico de ações
- Páginas informativas

De todas estas funcionalidades, a mais difícil de ser implementada foi a conversão de um ePortfolio para o formato de PDF, devido ao *plugin HTML2PDF*. Este *plugin*, apesar de já se encontrar em vigor, continua a ser um *plugin* experimental que não nos permitiu realizar a conversão da forma que desejariam, sendo uma ferramenta que ainda precisa de ser mais testada e melhorada.

5.3 Componentes Vue

Para o desenvolvimento das diversas páginas da aplicação, foram desenvolvidos diversos componentes *vue*. Cada componente, normalmente, divide-se em três partes: *template*, *script* e *style*.

- *HomePage.vue*: componente responsável por criar a página inicial onde o utilizador pode escolher fazer login ou então registar-se caso ainda não tenha conta
- *LoginPage.vue*: componente responsável por criar a página de login do utilizador
- *RegisterPage.vue*: componente responsável por criar a página de registo de um novo utilizador
- *EportfolioPage.vue*: componente responsável por criar a página default para onde o utilizador irá após fazer login caso tenha um eportefolio criado. Nesta página, o utilizador tem diversas opções como apenas visualizar o seu eportefolio ou então editar a sua conta, editar o seu eportefolio ou apagar o seu eportefolio
- *CreatePage.vue*: componente responsável por criar a página para onde o utilizador irá após fazer login caso não tenha um eportefolio criado. Nesse caso, esta página dá a opção ao utilizador de criar um eportefolio
- *CreateEportPage.vue*: componente responsável por criar a página onde o utilizador pode criar um eportefolio
- *EditEportPage.vue*: componente responsável por criar a página onde o utilizador edita o seu eportefolio
- *EditPerfilPage.vue*: componente responsável por criar a página onde o utilizador edita o seu perfil
- *AccountPage.vue*: componente responsável por criar a página onde o utilizador pode consultar e editar os dados da sua conta
- *FeedPage.vue*: componente responsável por criar a página onde o utilizador pode consultar a sua atividade na aplicação

- *LibraryPage.vue*: componente responsável por criar a página onde o utilizador visualiza os diferentes tipos de documentos/certificados de atividade profissional armazenados assim como outros tipos de documentos . Nesta página, o utilizador também tem como opções carregar um CV novo, um certificado/diploma novo, uma carta de apresentação nova ou outro tipo de documento novo. Para além disto, em cada ficheiro presente, o utilizador tem a opção de editar o nome desse ficheiro ou então removê-lo
- *AboutPage.vue*: componente responsável por criar uma página informativa onde o utilizador pode perceber melhor a aplicação
- *HelpPage.vue*: componente responsável por criar a página informativa a que o utilizador pode recorrer caso tenha dúvidas de como utilizar a aplicação

6 Documentação da API

O *strapi* oferece um *plugin* para a geração da documentação. Esse *plugin*, *documentation*, usa *SWAGGER UI* para o utilizador visualizar a documentação da API. Para o utilizador o poder usar, este necessita de o instalar correndo o seguinte comando no terminal do projeto *strapi* que criou inicialmente ”`npm run strapi install documentation`”. Quando instalado, o *plugin* irá percorrer as rotas do projeto que se encontram na pasta *api* e irá tentar criar a documentação apropriada.

Para terceiros poderem ter acesso à documentação gerada do nosso projeto, decidimos criar uma conta no *Swagger* onde partilhamos a documentação previamente gerada pelo *strapi*. De seguida, partilhamos o *link* de acesso à documentação: [Documentação da API](#).

7 Docker

Para realizarmos o *deployment* da nossa aplicação decidimos que iríamos utilizar a ferramenta *Docker*. Esta ferramenta utiliza virtualização a nível do sistema operativo para permitir fornecer, por assim dizer, *software* em certos pacotes denominados de *containers*.

O principal objetivo de realizar *deployment* de uma aplicação é permitir que esta fique disponível de uma forma simples a quem pretenda usá-la. A ideia é esse utilizador ter de correr apenas alguns comandos relacionados com a configuração dos *containers* no nosso caso em específico e ser-lhe possível usufruir da aplicação, sendo apenas necessário escrever o *site* no seu motor de busca.

Para este trabalho apenas conseguimos efetivamente dar *deployment* do *frontend* da nossa aplicação através do *docker*, sendo na mesma necessário as partes relativas à base de dados e ao *backend* serem realizadas de acordo com o procedimento normal, que nós também tivemos de realizar. Isto aconteceu devido a alguns problemas que surgiram aquando de conectar os diversos *containers*, um com a base de dados, outro com o *backend* e outro com o *frontend* através do *docker-compose* que permite inicializar estes *containers*. Acreditamos que o problema seja relativo ao facto de inicializarmos o *container* relativo ao *backend* primeiro ou ao mesmo tempo que o *container* da base de dados e então, apesar do *container* da base de dados já tiver sido até inicializado, o *Mongo* em si, ainda não está operacional no *container*. Isto resulta em não conseguirmos estabelecer a relação com a base de dados quando tentamos realizar a fase de *deployment*.

Então basicamente conseguimos apenas dar *deployment* do frontend da nossa aplicação. Desde que o *backend* esteja a correr, o *container* fica a funcionar em *background* e é possível aceder ao nosso *site* escrevendo apenas no motor de busca "localhost/8080/", neste caso. Como apenas é um *container* acabamos por não utilizar o *docker-compose* e é somente necessário dar *build* da *docker image* e corrê-la. Além disto também é necessário utilizar um *web service* denominado *Nginx* que é basicamente um servidor HTTP que irá correr também no *docker container*.

8 Conclusão

No geral acreditamos ter realizado um bom trabalho, tendo conseguido atingir a maioria dos objetivos propostos e desenvolvido uma aplicação que permite a um utilizador criar e gerar os seus *ePortfolios*, além de poder também converter os *ePortfolios* que criou para formato de ficheiro PDF e fazer o seu *download* que era o objetivo principal deste projeto.

Relativamente a trabalho futuro que permitiria melhor o nosso projeto seria por exemplo permitir vários tipos diferentes de utilizador, como empresas que iriam possuir funcionalidades específicas e que, por exemplo, conseguiram ter acesso aos *ePortfolios* do utilizador e pesquisar por filtragem também. Basicamente remetendo para o que acontece também no *CV Europass* onde existem parcerias com empresas. Outro funcionalidade que poderia ter sido implementada seria por exemplo diferentes *templates* para a conversão para formato de ficheiro PDF, que permitissem ao utilizador da aplicação customizar, por assim dizer, o seu ficheiro PDF a seu gosto. Algumas destas customizações podiam ser, por exemplo, alterar o local da foto, alterar a ordem pela qual a informação é apresentada, dar destaque a alguma informação em específico, etc.

Por fim, e também como já foi mencionado na secção do *Docker*, um dos aspetos a melhorar era sem dúvida realizar o *deployment* completo da nossa aplicação, o que infelizmente não foi possível.

Apesar destes aspetos, consideramos que este projeto nos ajudou a aprender novas tecnologias e a trabalhar com novas ferramentas que não foram exploradas em outras unidades curriculares, além da dificuldade que surgiu por trabalharmos com um projeto de maior escala e que nos permitiu desenvolver competências de comunicação e organização em grupo.

9 Referências bibliográficas

- <https://strapi.io/documentation/developer-docs/latest/development/plugins/documentation.html#installation>
- https://developer.mozilla.org/pt-BR/docs/Web/API/Fetch_API
- <https://vuex.vuejs.org/#what-is-a-state-management-pattern>
- <https://azure.microsoft.com/pt-pt/overview/nosql-database/>

10 Anexos

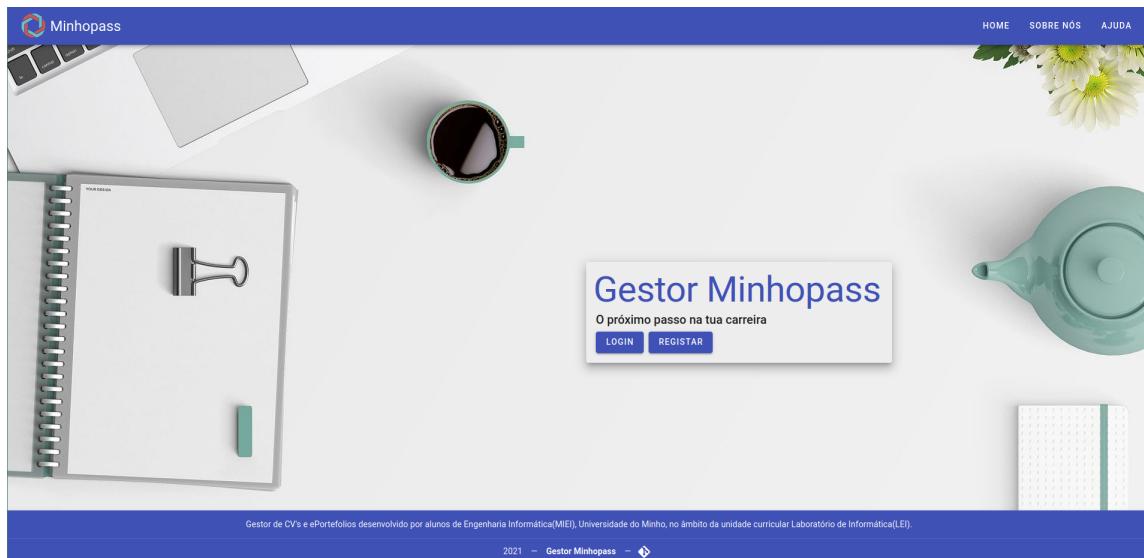


Figure 3: Página Inicial

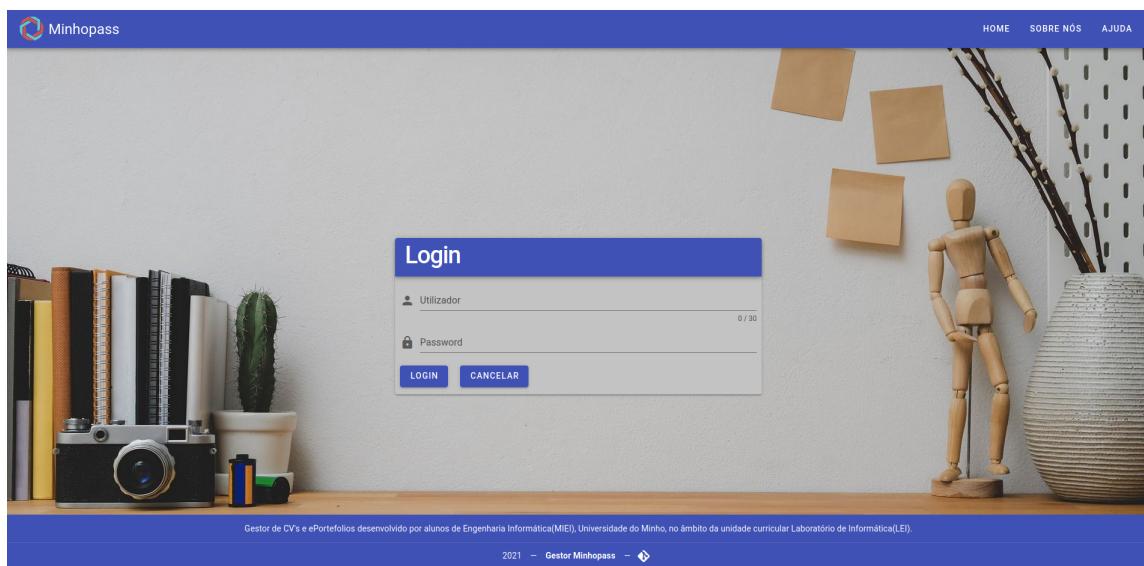


Figure 4: Página de Login

The screenshot shows the 'Biblioteca' (Library) section of the Minhopass application. At the top, there is a sidebar with icons for profile, documents, settings, and notifications. The main area has a blue header with the Minhopass logo and navigation links: HOME, SOBRE NÓS, AJUDA, and LOGOUT.

Biblioteca

Use a biblioteca para armazenar e organizar os seus documentos.

CVs

+ ADICIONAR | + CRIAR CV

Two CV entries are displayed:

- eportefolio (Thumbnail: red shirt)
- eportefolio2 (Thumbnail: red shirt)

Certificados/diplomas

+ ADICIONAR

Three certificates are listed:

- beta (Thumbnail: white certificate)
- cert_engenharia (Thumbnail: red seal)
- cert_southAfrica (Thumbnail: grey certificate)

Gestor de CV's e ePortfolios desenvolvido por alunos de Engenharia Informática(MIE), Universidade do Minho, no âmbito da unidade curricular Laboratório de Informática(LE).

2021 - Gestor Minhopass - 🔍

Figure 5: Biblioteca

The screenshot shows the 'Biblioteca' section with a different layout. The sidebar and top navigation are identical to Figure 5.

Certificados/diplomas

+ ADICIONAR

Three certificates are listed:

- beta (Thumbnail: white certificate)
- cert_engenharia (Thumbnail: red seal)
- cert_southAfrica (Thumbnail: grey certificate)

Cartas de apresentação

+ ADICIONAR

A message indicates: Sem documentos nesta secção.

Outros documentos

+ ADICIONAR

Gestor de CV's e ePortfolios desenvolvido por alunos de Engenharia Informática(MIE), Universidade do Minho, no âmbito da unidade curricular Laboratório de Informática(LE).

2021 - Gestor Minhopass - 🔍

Figure 6: Biblioteca

The screenshot shows the Minhopass ePortfolio interface. At the top, there's a blue header bar with the logo 'Minhopass' and navigation links: HOME, SOBRE NÓS, AJUDA, and LOGOUT. On the left, a vertical sidebar features icons for profile, dashboard, settings, and notifications. The main content area has a title 'O meu ePortefolio' and a sub-instruction 'Mantém o teu ePortefolio atualizado, adiciona novas secções e informação a qualquer altura.' Below this, there's a circular profile picture of a man with a beard, followed by sections for 'Dados pessoais' (Personal Data) and 'Contacto' (Contact). Under 'Dados pessoais', it lists: Nome: Pedro Nuno, País de origem: Portugal, Género: Outro, Data de nascimento: 1998-12-13, and Profissão: Maquilhador. Under 'Contacto', it lists: Email: pedro@gmail.com and Telefone: 914323565. A section for 'Endereço' (Address) follows, listing: Morada: Ribeirão, Trofa; Código postal: 4300-003; Cidade: Porto; and País: Portugal. At the bottom, there's a checkbox for 'Experiência profissional' (Professional Experience) which is currently unchecked. The footer contains the text 'Gestor de CV's e ePortefolios desenvolvido por alunos de Engenharia Informática(MIE), Universidade do Minho, no âmbito da unidade curricular Laboratório de Informática(LE).', the year '2021', and the link 'Gestor Minhopass'.

Figure 7: Eportefolio

This screenshot shows the 'Experiências profissionais' (Professional Experience) and 'Educação e formação' (Education and Training) sections of the ePortfolio. The 'Experiências profissionais' section includes a card for 'Serviço ao cliente' (Customer Service) at IKEA from September 2019 to now, describing it as providing the best service to clients. The 'Educação e formação' section includes a card for a 'Mestrado' (Master's degree) at the University of Minho from September 2015 to September 2020, mentioning functional, imperative, and object-oriented programming. Both sections have a 'Mais informações' (More information) button at the bottom right. The footer is identical to Figure 7.

Figure 8: Eportefolio

A minha conta

Consulta e edita os dados da tua conta sempre que quiseres.

Dados

username	janes51
email	janes@hotmail.com
EDITAR	

Alterar Password

Nova password
ALTERAR

Gestor de CV's e ePortefólios desenvolvido por alunos de Engenharia Informática(MIE), Universidade do Minho, no âmbito da unidade curricular Laboratório de Informática(LE).

2021 — Gestor Minhopass — 🔍

Figure 9: Página de consulta e edição da conta do utilizador

Feed de operações

Consulta o registo da tua atividade no gestor Minhopass.

Operação	Quando	Onde	Nota	Ações
Edição	22/6/2021 22:38	Conta	Shhhh, alteraste a tua password	🔗
Adição	22/6/2021 22:45	Biblioteca	Introduziste novo documento na biblioteca(cvs)	🔗
Adição	22/6/2021 22:49	Biblioteca	Introduziste novo documento na biblioteca(cvs)	🔗
Remoção	22/6/2021 22:52	Biblioteca	Removeste um documento da tua biblioteca(cvs)	🔗
Remoção	22/6/2021 23:1	Biblioteca	Removeste um documento da tua biblioteca(cartas)	🔗
Edição	22/6/2021 23:2	Conta	Alteraste os dados da tua conta.	🔗

Gestor de CV's e ePortefólios desenvolvido por alunos de Engenharia Informática(MIE), Universidade do Minho, no âmbito da unidade curricular Laboratório de Informática(LE).

2021 — Gestor Minhopass — 🔍

Figure 10: Página do Feed

Criar ePortefolio

Dados pessoais Experiência profissional Educação e formação Competências pessoais Informação extra

Informação extra

Adiciona outros tipos de informação ou cria novos.

Tipo	participação em conferências	
Nome do campo	123 300	
Aa	Pessoas envolvidas	
Nome do campo	123 32	
Aa	Empresas envolvidas	
ADICIONAR CAMPO NÚMÉRICO		
Nome do campo	Host	
Descrição do campo	Site Círculo	
ADICIONAR CAMPO TEXTUAL		
!		
ADICIONAR NOVA INFORMAÇÃO EXTRA		
CONTINUAR	APAGAR	VOLTAR

Gestor de CV's e ePortefolios desenvolvido por alunos de Engenharia Informática(MIEI), Universidade do Minho, no âmbito da unidade curricular Laboratório de Informática(LE).

Figure 11: Página da criação de um eportefolio

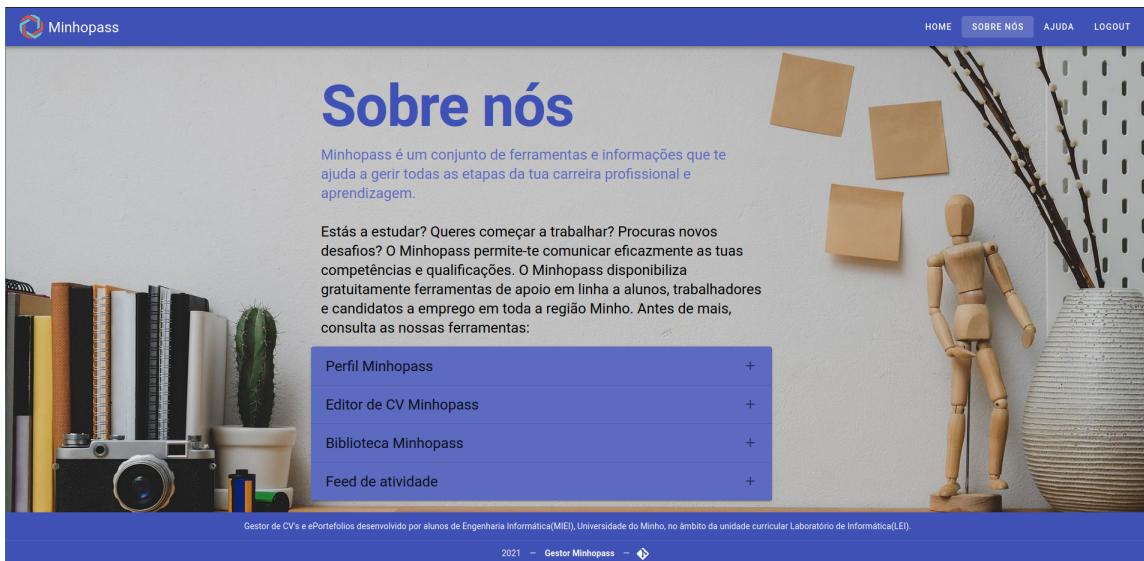


Figure 12: Página informativa

FAQ

Tens dúvidas relativamente ao funcionamento do gestor Minhopass?
Não te preocupes, não és o primeiro.

O que é o Minhopass? +

Porque devo utilizar o MinhoPass? +

Quanto custa o registo no Minhopass? +

Posso personalizar o meu CV Minhopass? +

Onde posso guardar o meu CV Minhopass? +

Em que formato posso guardar o meu CV Minhopass? +

Gestor de CV's e ePortfolios desenvolvido por alunos de Engenharia Informática(MIE), Universidade do Minho, no âmbito da unidade curricular Laboratório de Informática(LE).
2021 - Gestor Minhopass -

Figure 13: Página informativa