



UNIVERSIDADE da MADEIRA

Aplicações Centradas em Redes 2017/2018

Erasmus Software

Docente: Filipe Quintal



Erasmus+

Grupo 4:

Pedro Mendes nº2022114

David Moniz nº2021914

João Silva nº2020114

Introdução

No âmbito da unidade curricular, Aplicações Centradas em Redes da Universidade da Madeira, foi-nos proposto a realização de um projeto, que consistia numa aplicação web, cujo tema era livre, e seria desenvolvida através da framework Laravel. O tema proposto pelo nosso grupo, grupo 4, consiste em desenvolver uma plataforma capaz de gerir candidaturas online, ao abrigo do programa de mobilidade Erasmus. O público-alvo da aplicação indicada são:

- aos estudantes que queiram candidatar-se ao programa.
- aos coordenadores do programa de cada universidade.
- aos diretores curso de cada universidade.
- às pessoas que pretendam obter mais informações sobre o Erasmus.

Pretendemos descrever as funcionalidades desta aplicação, a metodologia utilizada e análises ao desenvolvimento da própria aplicação. Iremos começar por identificar o problema a resolver, referindo os requisitos e futuras funcionalidades do nosso sistema. Depois iremos abordar a revisão da literatura de ação, que aborda a nossa aprendizagem individual e em grupo do framework laravel e a nossa estratégia de implementação da aplicação, desde o back-end até ao front-end.

Problema

Esta aplicação tem o objetivo de substituir o método tradicional das candidaturas feitas ao abrigo do programa de mobilidade Erasmus. O método tradicional de uma candidatura exige que o estudante tenha que estar constantemente a frequentar gabinetes, quer seja o coordenador do programa, quer seja o director de curso, para assinaturas, estampagens, confirmação do país de destino ou de anexos. O nosso método digitaliza o método tradicional, tornando-o assim mais rápido e eficaz.

Revisão da literatura e plano de ação

Antes de ser feita a implementação da aplicação web, decidimos fortalecer os nossos conhecimentos da framework laravel, utilizando a própria documentação online, da versão 5.3, e de vários vídeos do laracast.

Começamos por fazer um brainstorm de várias tarefas que cada utilizador teria que ter, e depois fizemos um sketch de tabelas para implementar numa base de dados, e utilizamos também o mySQL workbench para tornar mais fácil a visualização e formulação da base de dados.

Após concordarmos com o diagrama da base de dados, começamos por implementar a mesma, através das migrations do framework laravel.

Depois de suceder com as migrations, utilizamos os Models do Laravel para fazer operações CRUD referentes à base de dados. Após isto começamos a criar os controllers e as views que foram se apresentando necessárias ao longo do projeto, e fazendo algumas alterações às tabelas da base de dados. É importante referir que ao longo do projeto, recorremos bastante à documentação disponibilizada online do framework Laravel e aos documentos disponibilizados nas aulas referentes a algumas ferramentas de desenvolvimento web (por exemplo php, js, css).

Solução proposta

a. Descrição geral da solução

CountriesController - Neste controller, usamos as funções simples de inserir, modificar e eliminar dados na tabela Countries, da base de dados. A tabela countries é responsável por guardar os dados dos países.

CitiesController - Neste controller, usamos as funções simples de inserir, modificar e eliminar dados (operações CRUD) na tabela Cities, da base de dados. Este controller é também responsável por verificar se o país dessa cidade existe, caso exista utiliza esse país, se não existe, é responsável por inserir esse país. Esta tabela guarda os dados referentes a uma cidade.

AddressesController - Neste controller, usamos as funções simples de inserir, modificar e eliminar dados (operações CRUD) na tabela Addresses da base de dados. Este é também responsável por verificar se a cidade do address existe ao inserir, caso exista não insere uma nova cidade e utiliza a previamente inserida, caso não exista automaticamente cria uma nova cidade.

Esta três tabelas referidas anteriormente estão relacionadas entre si, através de chaves estrangeiras com id de cada tabela. Por exemplo: se quisermos saber o país de um endereço, teremos que aceder à tabela address, que tem uma chave estrangeira de cities, que se chama city_id, e por sua vez à tabela city tem uma chave estrangeira, country_id, e por sua vez à country através dessa country_id para obter informações acerca desse país.

ManagersController - Neste controller, usamos as funções simples de inserir, e eliminar dados na tabela Managers da base de dados. A tabela manager referencia um user através de uma chave estrangeira (user_id), esta tabela é também utilizada para podermos diferenciar as roles dos nossos utilizadores. É importante referir que uma universidade pode ter mais que um manager.

DirectorsController - Neste controller, usamos as funções simples de inserção e eliminação de dados na tabela Directors da base de dados. A tabela director referencia também um user através de uma chave estrangeira (user_id), e contem um atributo extra em relação à mesma, que é o program_id, que referencia a tabela programs, que permite identificar qual o curso que este utilizador é diretor. É importante referir que uma universidade só pode ter um único diretor de curso.

StudentsController - Neste controller, usamos as funções simples de inserção e eliminação de dados na tabela Students da base de dados. A tabela students referencia também um user através de uma chave estrangeira (user_id), e contem um atributo extra em relação à mesma, que é o program_id, que referencia a tabela programs, que permite identificar qual o curso a que este estudante pertence.

CandidatesController - Neste controller, usamos as funções simples de inserção, e eliminação na tabela Candidates da base de dados. É importante referir que esta tabela referencia o student através de uma chave estrangeira (student_id). O nosso pensamento na diferenciação entre student e candidate é que um estudante só se torna candidato quando é criado um processo.

ProgramsController - Neste controller, usamos as funções simples de inserção, modificação e eliminação de dados na tabela Programs, e ainda uma função que retorna uma view que é utilizada para um pedido AJAX que retorna os cursos de uma certa universidade.

InformationsController - Este controller, é responsável por organizar as informações das News dependendo da sua quantidade e redireciona para uma view com informações das News. É também responsável pela retorna da view específica dessa New.

UsersController - Neste controller, usamos as funções simples de inserção, modificação e eliminação de dados na tabela Users da base de dados. Também tem outras funções como o registo, logout, recuperação de conta, o login do utilizador e do admin. Tem também funções para editar informações pessoais dos utilizadores, submissão de imagens, retorna de views, redireccionamento se os utilizadores não estiverem autenticados. É importante referir que a tabela de users tem duas chaves estrangeiras address_id, que permite referenciar um endereço a um utilizador, e university_id, pois todos os utilizadores do sistema pertencerão a uma universidade (tanto os managers, como os estudantes, como os diretores de curso).

ProcessesController - Neste controller, usamos as funções simples de inserção, na tabela processes e da tabela Process_University. A tabela Process_University permite relacionar o processo em questão com a universidade de destino do candidato e a tabela processes permite ir buscar os utilizadores intervenientes estudante e o manager da universidade de origem que interage, também na tabela process contém um atributo files que guarda o path dos diferentes ficheiros inseridos desse processo. Também tem outras funções como o upload e o download dos ficheiros e a aprovação dos resultados do estudante na candidatura.

UniversitiesController - Neste controller, usamos as funções simples de inserção, modificação e eliminação de dados na tabela Universities da base de dados, e ainda algumas views para pedidos ajax, e obter informações sobre uma universidade em específico.

DashboardController - Neste controller, o dashboard tem como objetivo apresentar a view da dashboard após o utilizador ter feito o login na aplicação. Tem várias funções como a criação de um candidato e de um processo, a filtração dos processos dependendo do tipo de utilizador e a amostra de um processo e dos seus ficheiros anexados.

MessagesController - Neste controller, o objetivo é conseguir apresentar a view das mensagens do utilizador e poder dar a liberdade de enviar uma mensagem entre os diferentes tipos de utilizador que interagem no seu processo de candidatura (manager e diretor de curso da sua universidade), como também responder e visualizar mensagens recebidas.

b. Diagramas auxiliares

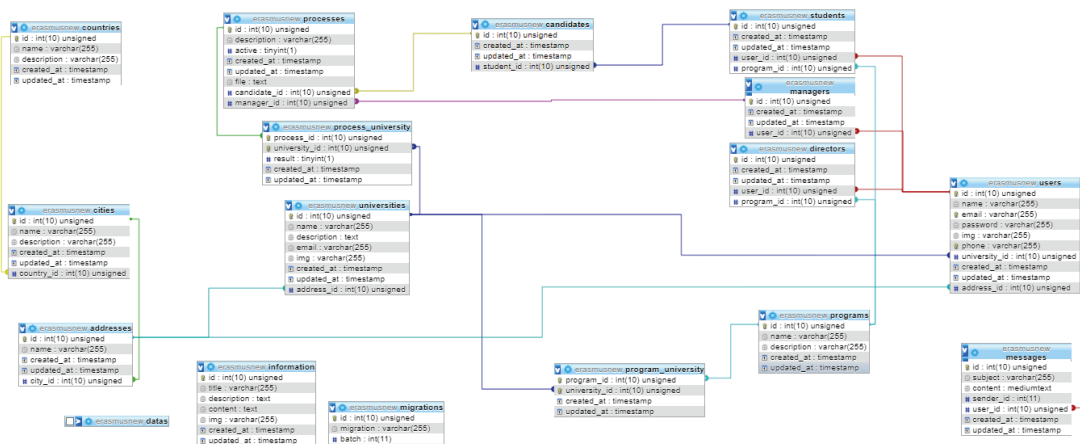


Figura 1 - Diagrama E-R da base de dados

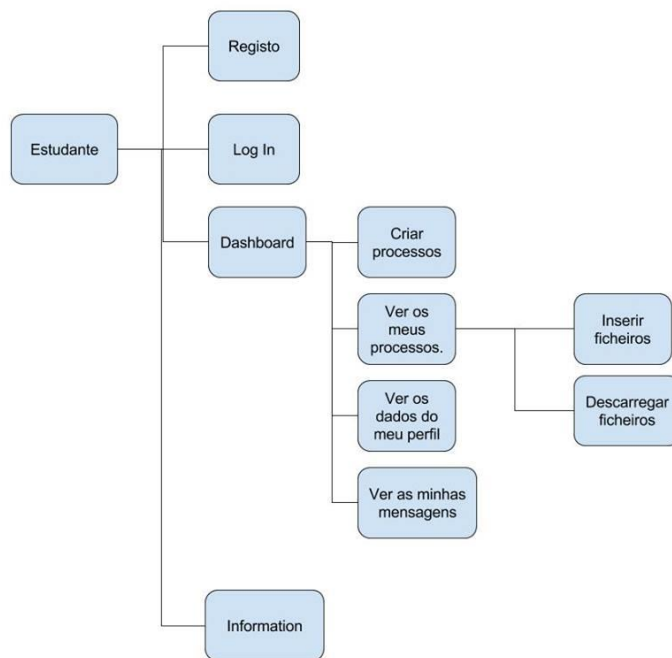


Figura 2 - Diagrama de casos utilização

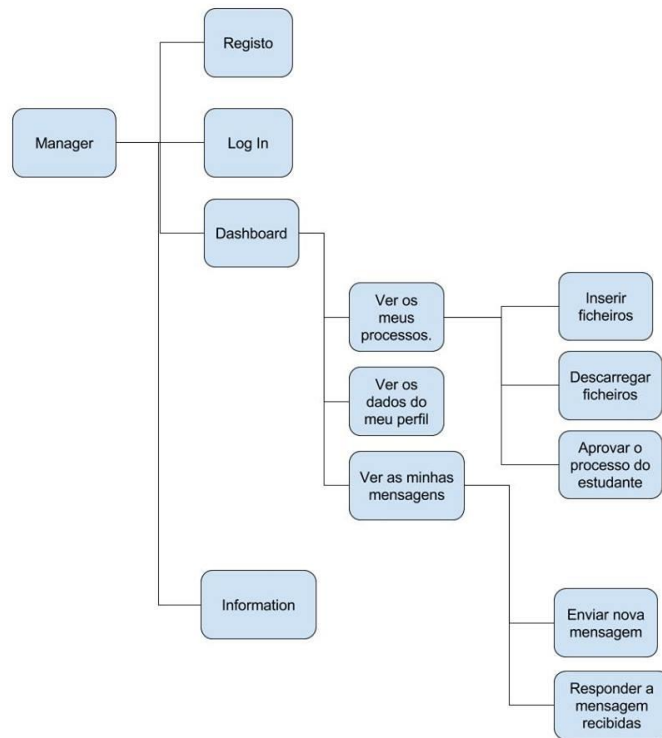


Figura 3 - Diagrama de casos utilização

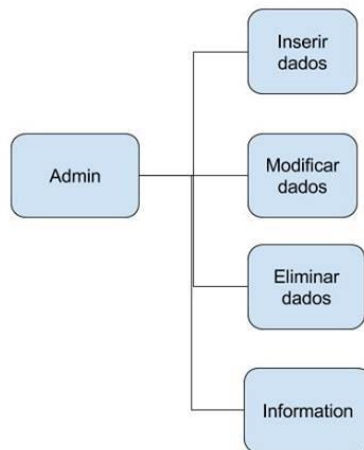


Figura 4 - Diagrama de casos utilização

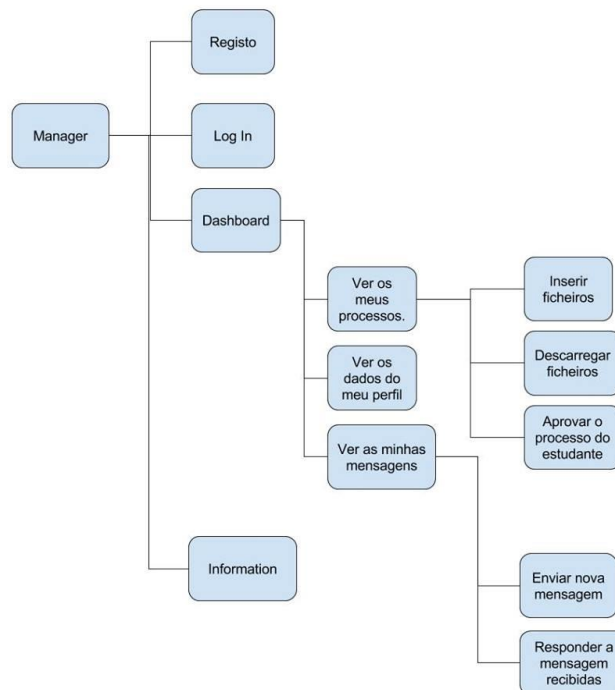


Figura 5 - Diagrama de casos utilização

c. Descrição de aspetos particulares relevantes

É importante referir que a autenticação de login não foi feita utilizando o auth do Laravel mas sim com a session global helper na qual enviamos o id do utilizador numa variável de sessão e a role. Resolvemos fazer com a session pois simplificou o processo de autenticação, pois temos 4 tabelas diferentes de utilizador (student, candidato, que também é um student, manager de uma universidade, e diretor de curso de uma universidade, no qual todas estas são também um user), e a funcionalidade é a mesma. Quando um utilizador tenta aceder a uma rota que não seja possível a este aceder caso não esteja autenticado este é redirecionado para o índice da aplicação.

Utilizamos o façade Storage do laravel, para nos auxiliar no upload de ficheiros e visualização dos mesmos.

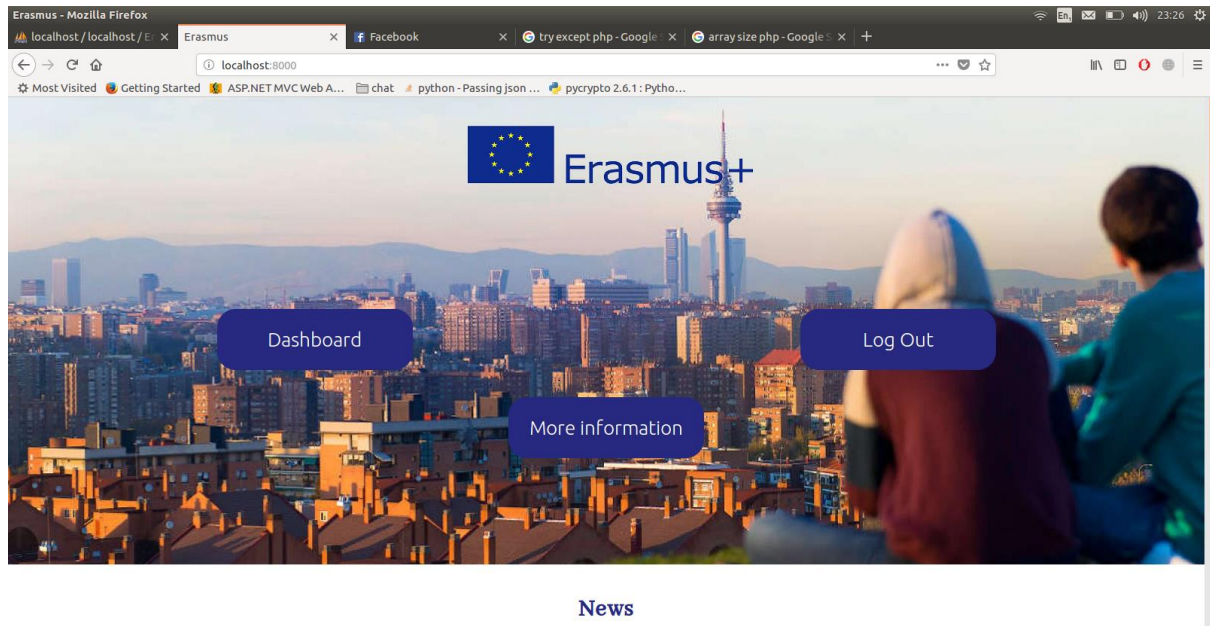
Utilizamos ainda um servidor nodejs para um chat, no qual tem o objetivo de fornecer um meio de comunicação entre pessoas do mesmo país de origem com um país de destino de Erasmus igual.

Um outro aspeto importante de referir é como o processo de candidatura funciona. Este é inicializado pela submissão de uma nova candidatura da parte do estudante de uma certa universidade. Quando este é submetido, este processo é atribuído a um manager (coordenador) da universidade do estudante (de origem), geralmente ao que tiver menos processos, e a um diretor de curso dessa universidade (de origem). O diretor de curso é responsável por introduzir o ficheiro Learning agreement, enquanto que o estudante é responsável por introduzir os restantes cinco ficheiros necessários para a candidatura (NIB, BI, etc). Após a submissão dos seis ficheiros e análise detalhada da parte do manager, este pode aceitar a candidatura ou recusar. Em ambos os casos o estado do processo passa para inativo, mas na aceitação o resultado do processo vai como aceite e no outro como recusado. Após a aceitação do processo, se o

estudante aceder à view desse processo este poderá aceder ao chat implementado em nodejs falado anteriormente.

Quanto a APIs externas utilizadas, utilizamos a API externa do google maps, para fornecer informação aos utilizadores acerca da localização das universidades.

Utilização da aplicação



Apresentação inicial da nossa aplicação.

A screenshot of the Erasmus+ application's registration page. The page has a header with the Erasmus+ logo and navigation links: 'Dashboard', 'More Info', and 'Log In'. The main content area is titled 'Register' and contains a form with the following fields: 'Select your role:' with a dropdown menu showing 'Student'; 'Name:' with a text input field; 'Password:' with a text input field; 'Confirm Password:' with a text input field; and 'Email:' with a text input field. On the right side of the page, there is a vertical banner image showing a group of people jumping in the air, with the Erasmus+ logo and text on the right.

Erasmus - Register - Mozilla Firefox

localhost:8000/register

Erasmus+

Country:

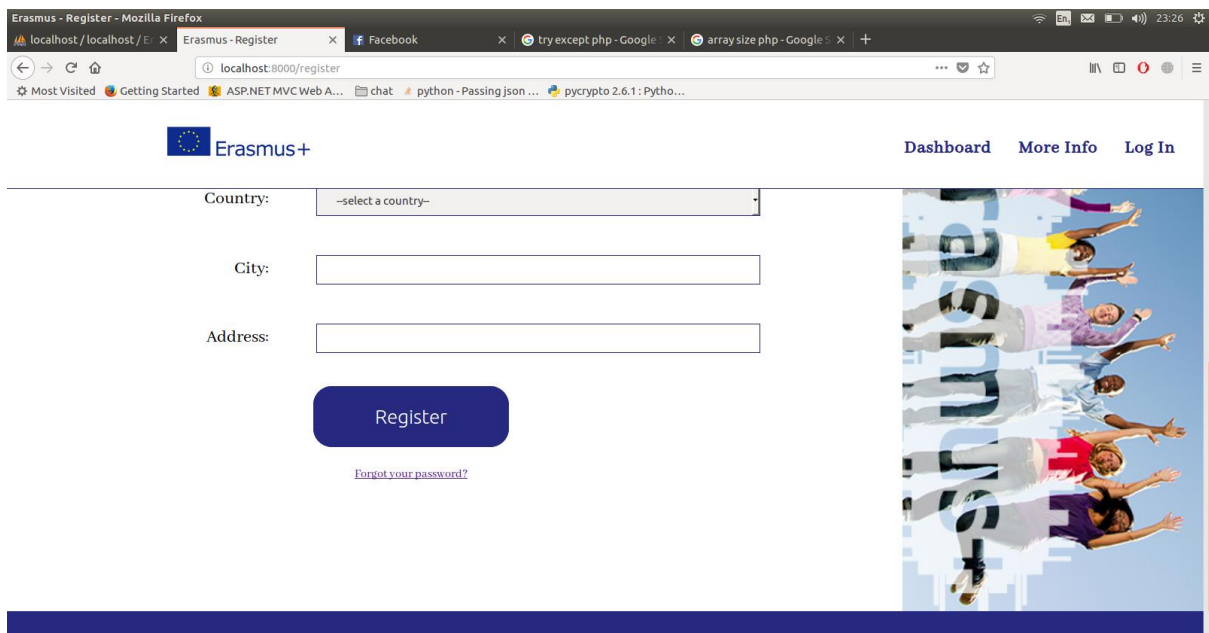
City:

Address:

[Register](#)

[Forgot your password?](#)

Dashboard More Info Log In



Registo da aplicação

Erasmus - Log In - Mozilla Firefox

localhost:8000/Login

Erasmus+

Dashboard More Info Log In

Login

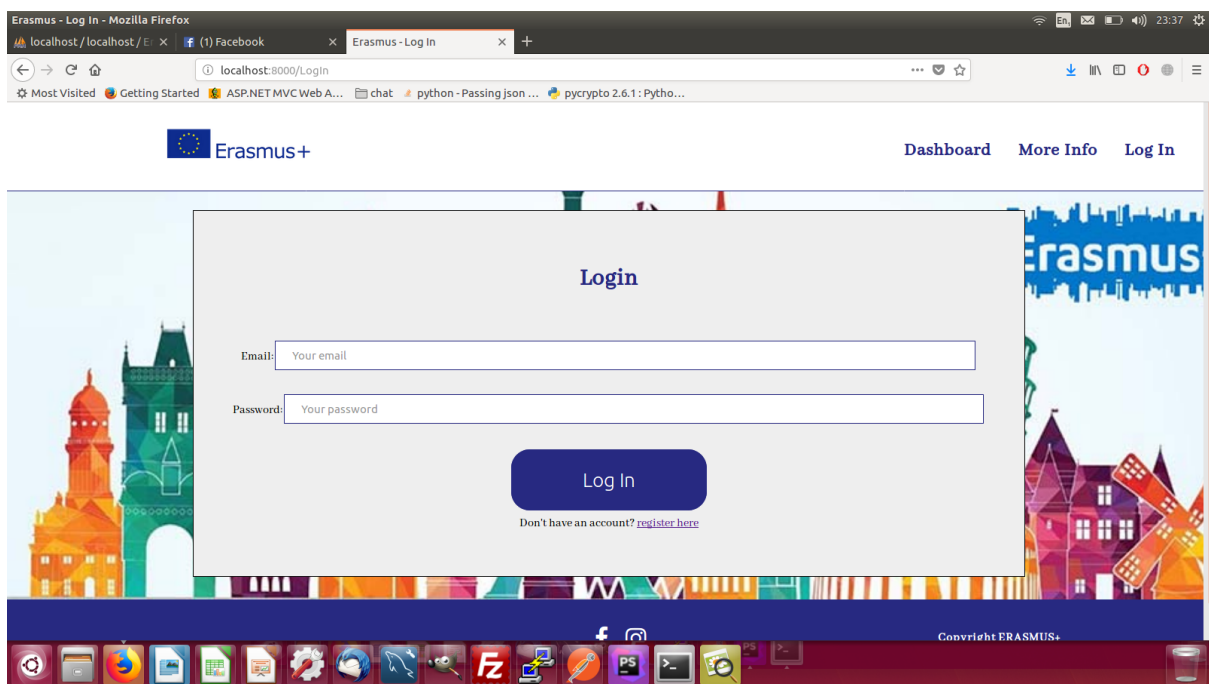
Email:

Password:

[Log In](#)

Don't have an account? [register here](#)

Copyright ERASMUS+



Log In da aplicação

Menu Revit Ar HF-UM API Aut A1 INTE localho how to Erasmus Erasmus ACR - R ACR-P Stove D + 127.0.0.1:8000/admin

Erasmus+ Home More Info Log Out

Add	Change	Remove
<h3>Add Country</h3> <p>Country Name:</p> <input type="text"/> <p>Description:</p> <input type="text"/>		

A view do admin que poderá inserir/modificar/apagar dados diretamente na base de dados.

Menu Revit Ar HF-UM API Aut A1 INTE localho how to Erasmus Erasmus ACR - R ACR-P Stove D + 127.0.0.1:8000/Information

Erasmus+ Home More Info Log In

O QUE E

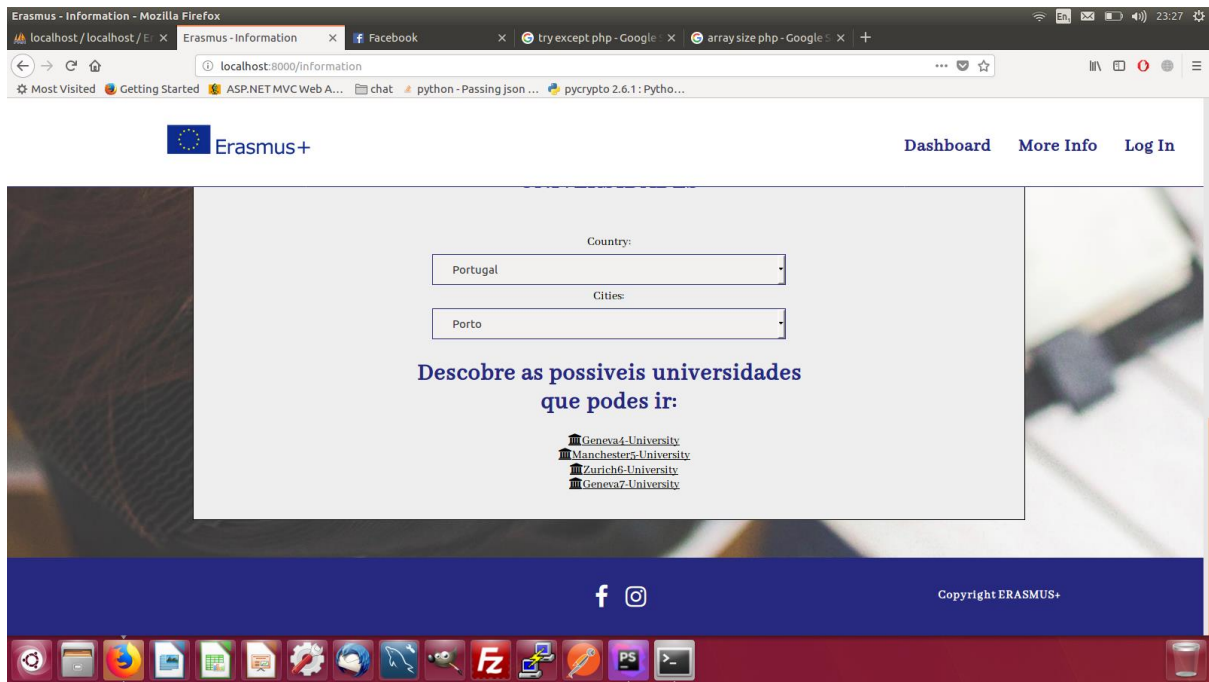
CONTEUDO

CONTEUDO

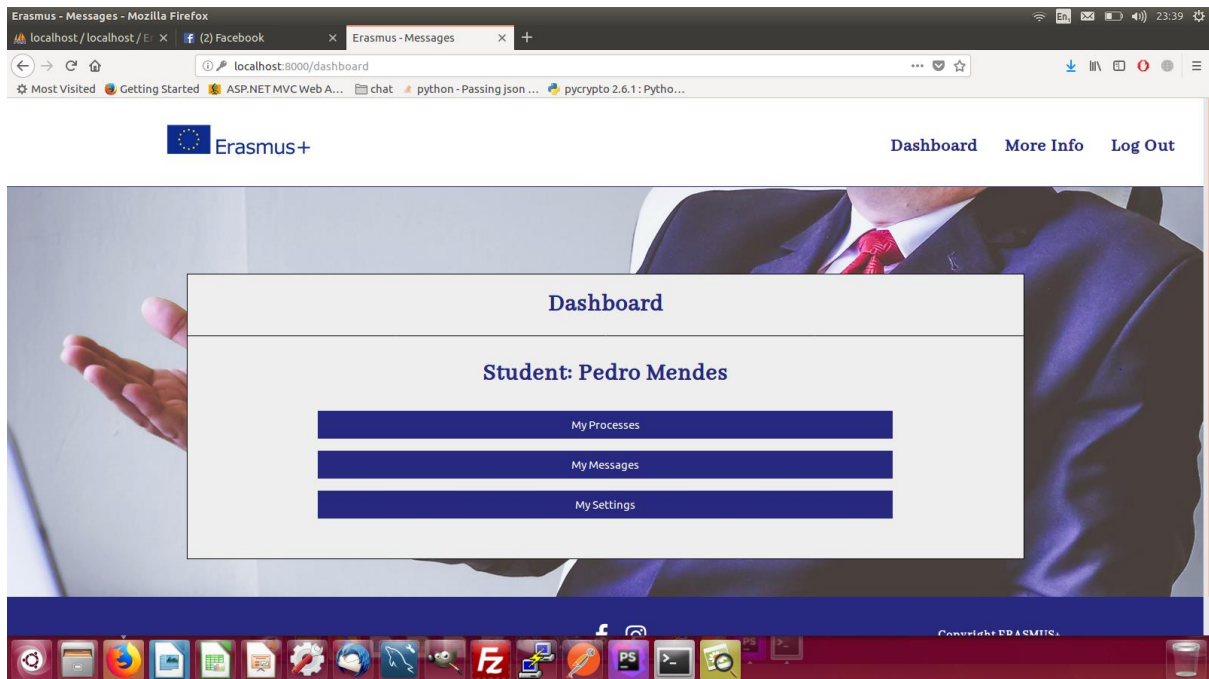
CONTEUDO

CONTEUDO

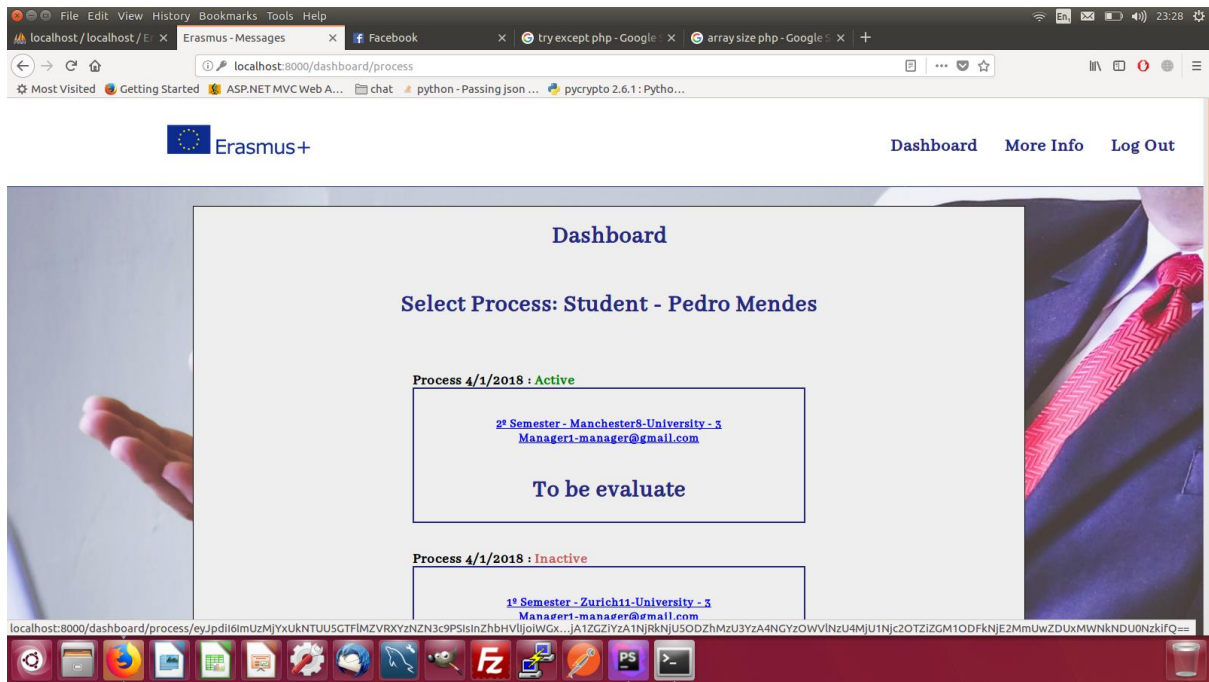
A view que mostra informação sobre as universidades.



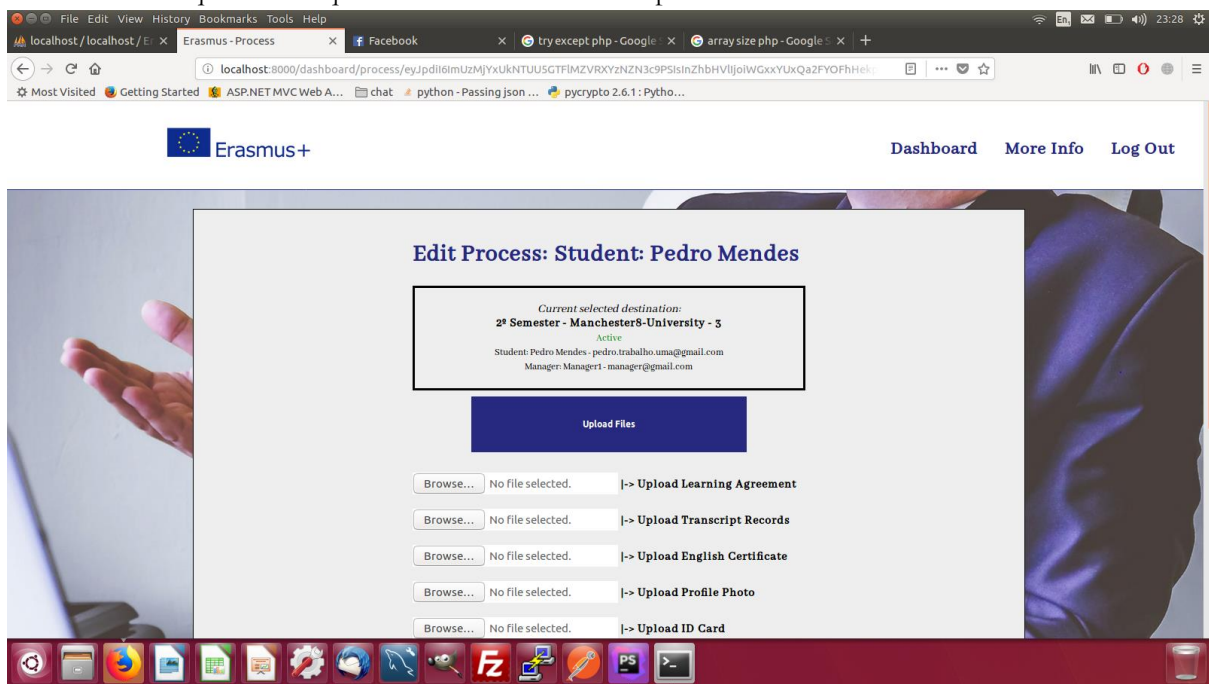
A view que mostra os links de informação sobre cada universidade.



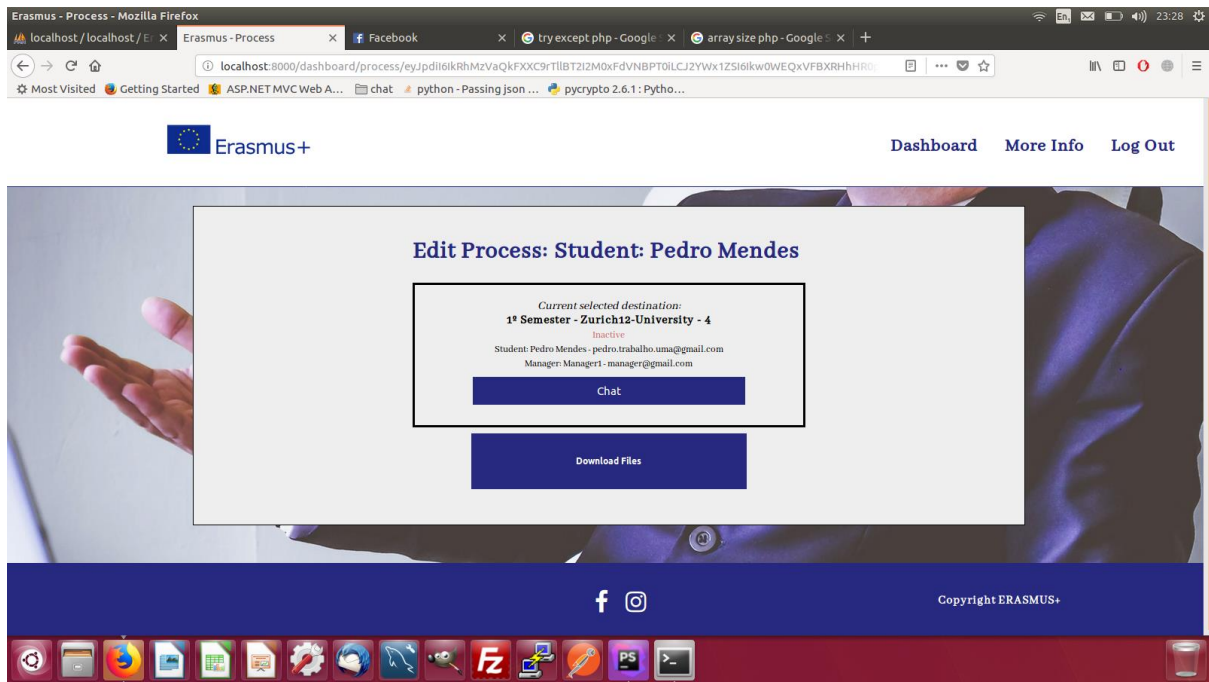
View principal apos a execução de login



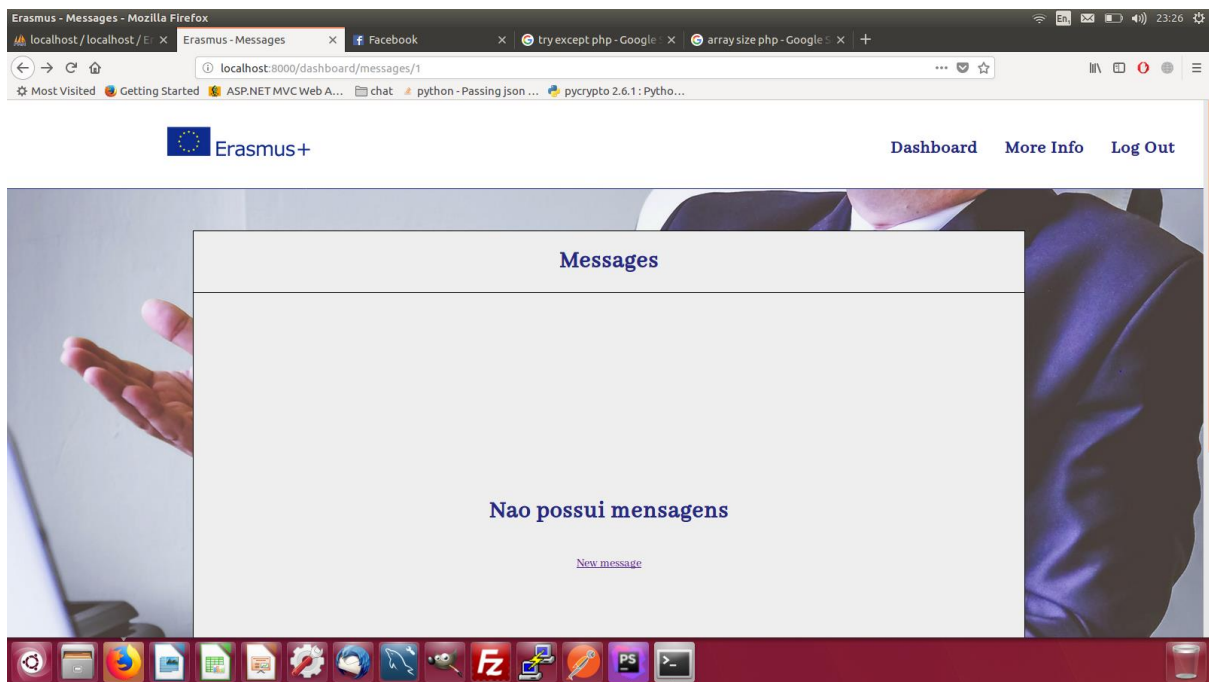
A view que mostra quando o estudante vê o seu processo ativo.



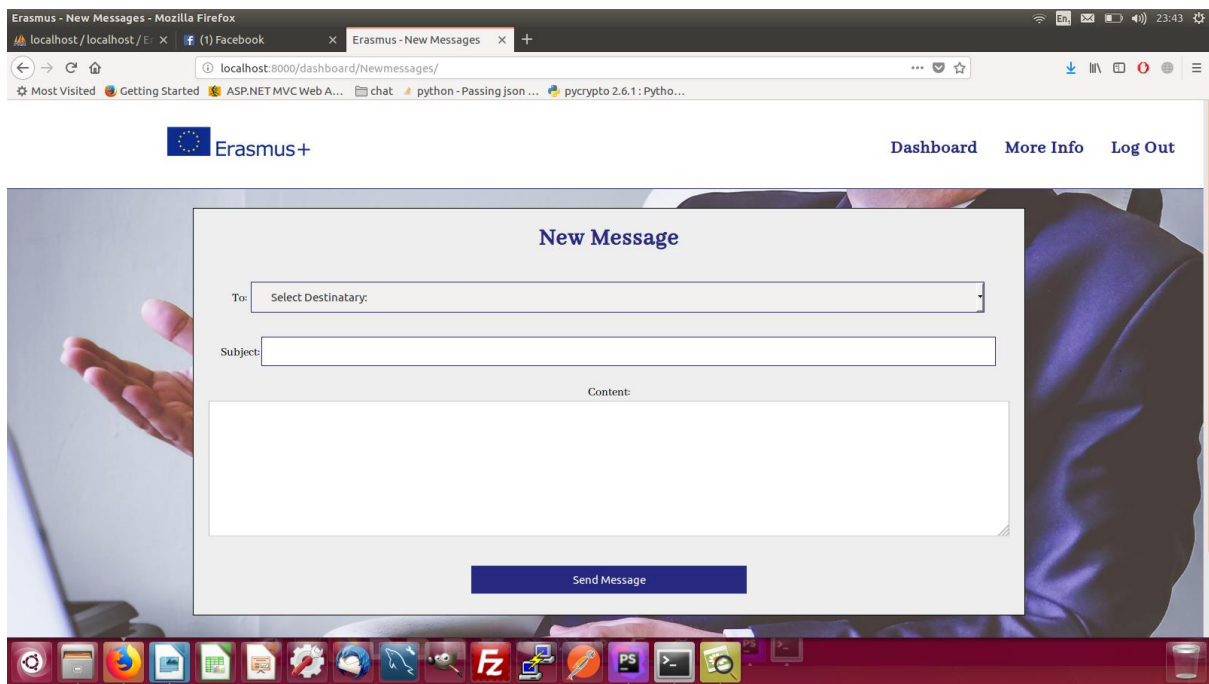
Ao clicar na view myprocesses mostra o botão de fazer upload dos ficheiros onde o estudante poderá escolher que ficheiro irá inserir o documento.



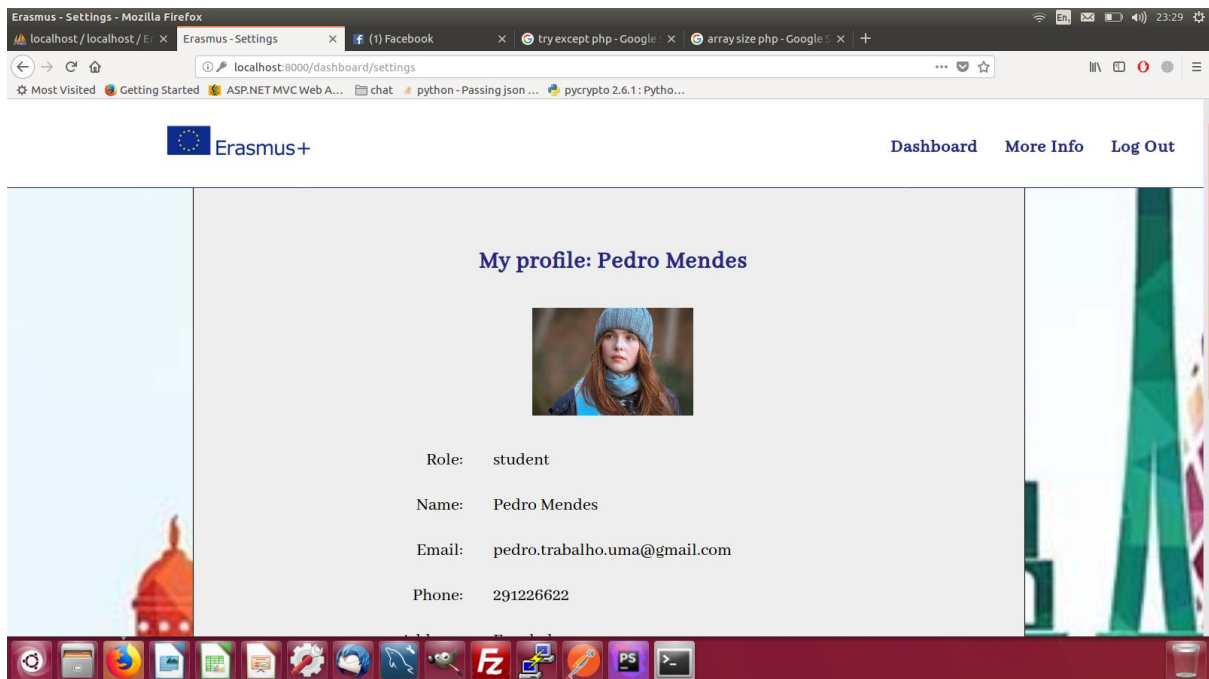
Caso este processo esteja aceita, aparece o botão que redireciona para a para o chat.



Ao clicar em my messages, direciona para a página onde é apresentado uma espécie de caixa de entrada de email.



Nesta pagina e possível criar uma nova mensagem, e depois confirmar o seu envio.



Pagina que mostra as configurações atuais e deixa alterar as mesmas.

Discussão

Quanto ao projeto, achamos que foi um projeto possível de ser utilizado e estendido futuramente, embora tenhamos tido alguns problemas inicialmente tanto na definição da base de dados como na habituação de utilização da framework laravel, acabámos por os superar e por desenvolver uma aplicação web com alguns pontos fortes e fracos.

Pontos fortes:

- A base de dados mostra-se consistente e robusta.
- Permite upload e download de ficheiros.
- Apresenta dados gerados dinamicamente.
- Utilização de pedidos AJAX em certas páginas (por exemplo: register).
- O admin tem liberdade de inserir, modificar e apagar dados da base de dados (que não são inseríveis por outros utilizadores) na própria aplicação.
- O estudante consegue candidatar-se online sem ter que se deslocar pessoalmente para falar com os intervenientes (manager e diretor de curso) no processo de candidatura de Erasmus, tornando assim o processo mais rápido e eficaz para todos os intervenientes.

Pontos fracos:

- O front-end não é por vezes consistente de página para página e poderia ser melhorado.
- Os países, as cidades e as universidades têm de ser inseridas um de cada vez pelo admin. Por exemplo, as coordenadas da universidade terão de ser inseridas cada vez que seja preciso adicionar uma nova universidade, uma vez que é utilizado o Google maps para visualizar a localização da universidade. Uma possível solução seria utilizar uma API para ir buscar os países, universidades, etc.

Conclusão

Após a conclusão deste projeto, sentimos que fortalecemos as nossas bases de conhecimento em relação a framework laravel, e que estamos mais confiantes e à vontade para desenvolver uma outra aplicação utilizando esta framework. Aprendemos que o front-end por vezes dá bastantes dores de cabeça, e que é bastante importante, e se uma aplicação web não for apelativa ao uso, é menos provável que as pessoas a utilizem. Quanto ao back-end achamos que também é bastante importante, pois permite providenciar uma aplicação web segura, dinâmica e com tratamento de falhas.

Embora tenhamos encontrado alguns problemas pelo caminho como referido anteriormente, achamos que o trabalho foi concluído com sucesso, com a maioria dos requisitos cumpridos, e que aumentamos a nossa capacidade de utilização de diferentes ferramentas para desenvolvimento web sejam elas de back-end ou front-end.

Referências

[https://laracasts.com/series/laravel-5-from-scratch;](https://laracasts.com/series/laravel-5-from-scratch)
[https://laravel.com/docs/5.3;](https://laravel.com/docs/5.3)