



IPG Politécnico
|da|Guarda
Polytechnic
of Guarda

RELATÓRIO DE PROJETO

Licenciatura em Engenharia Informática

Leandro José Lopes Fernandes

julho | 2018





Escola Superior de Tecnologia e Gestão
Instituto Politécnico da Guarda

RELATÓRIO DE PROJETO

LEANDRO JOSÉ LOPES FERNANDES

RELATÓRIO PARA A OBTENÇÃO DO GRAU DE LICENCIADO
EM ENGENHARIA INFORMÁTICA

Julho/2018

Agradecimentos

Gostaria de deixar o meu mais humilde e sincero agradecimento a todos os que me ajudaram e contribuíram para o meu crescimento e conhecimento.

À minha família. Devo-lhes o meu percurso académico.

À minha namorada Elisa, e amigos, pelo constante apoio incondicional e paciência nos momentos mais difíceis.

Agradeço ao Pplware pela oportunidade oferecida, era de facto uma empresa já conhecida e poder estagiar lá foi um sonho concretizado.

Agradeço também ao meu supervisor e coordenador pelo apoio, profissionalismo, compreensão e conhecimentos partilhados comigo.

Aos meus colegas e todos os professores do curso de Engenharia Informática principalmente à Prof^a Natália Gomes, um muito obrigado por todo o apoio, atenção e disponibilidade que me foi prestada.

Ficha de Identificação

Aluno

Nome: Leandro José Lopes Fernandes

Número de aluno: 1012177

Licenciatura: Engenharia Informática

Estabelecimento de Ensino

Escola Superior de Tecnologia e Gestão

Instituto Politécnico da Guarda

Instituição Acolhedora do Estágio

Nome: Pplware

Morada: Av. Narciso Ferreira, 138 – 4º, 4765-202, Riba de Ave

Telefone: 914 215 491

Duração do Estágio

Início: 14 de maio de 2018

Fim: 13 de julho de 2018

Supervisor na Instituição

Nome: Vítor Martins

Orientador na ESTG-IPG

Nome: Eng.º Fernando Melo Rodrigues

Resumo

O presente projeto foi desenvolvido com a parceria e em regime de estágio na empresa Pplware.com – Tecnologias de Informação e Serviços Web, LDA.

Este consiste no desenvolvimento de uma aplicação *Web* baseada na tecnologia MERN, que tem como objetivo a partilha de experiências, que os utilizadores gostariam de recomendar, realizadas nas suas viagens.

Para o desenvolvimento do projeto foi utilizada a metodologia de desenvolvimento ágil, SCRUM. A aplicação web foi desenvolvida em React para a parte do *front-end*, NodeJS e Express para o *back-end* e com recurso a MongoDB para base de dados.

Palavras-Chave:

Express, MongoDB, NodeJS, *Social Media*, *Web Application*.

Abstract

This project was developed with the partnership and internship at the company Pplware.

This is the development of a web application based on MERN technology, which aims to share experiences, which users would like to recommend, made in their travels.

For the development of the project was used the SCRUM methodology. The web application was developed in React for the front-end part, NodeJS and Express for the back-end and with MongoDB as database.

Keywords:

Express, MongoDB, NodeJS, *Social Media, Web Application.*

Índice

Agradecimentos	II
Ficha de Identificação.....	III
Resumo	IV
Abstract.....	V
Índice	VI
Índice de Figuras	IX
Índice de Tabelas	XI
1. Introdução.....	1
1.1. Caracterização sumária da instituição.....	1
1.2. Motivação	1
1.3. Descrição do Problema	2
1.4. Objetivos.....	2
1.5. Etapas do Estágio.....	3
1.6. Estrutura do documento	3
2. Estado da Arte	5
2.1. Aplicações existentes	5
2.1.1 TripAdvisor Attractions	5
2.1.2. Google Trips.....	6
2.2. Análise crítica das aplicações analisadas	6

3. Metodologia	8
3.1. Características do Scrum	8
4. Análise de Requisitos	10
4.1. Diagrama de Contexto	10
4.2. Atores e respectivos casos de uso	11
4.3. Diagrama de Casos de Uso	12
4.4. Descrição dos Casos de Uso	13
4.5. Diagrama de Sequência	14
4.6. Diagrama de Classes	15
4.7. Dicionário de Dados	15
5. Tecnologias	18
5.1. JavaScript.....	18
5.2. NodeJS	18
5.3. Express.....	19
5.4. MongoDB	19
5.5. MongoDBCompass.....	19
5.6. Yarn	20
5.7. Github	20
5.8. Postman.....	21
5.9. Mailtrap.....	21
6. Implementação	22

6.1.	Sign Up	22
6.2.	Password	23
6.3.	Confirmação de Email	23
6.4.	Login	24
6.5.	Tokens.....	25
6.6.	Inserir Viagens	25
6.7.	Informações sobre as viagens realizadas	26
6.8.	Adicionar Imagem	28
7.	Verificação e Validação	29
8.	Conclusão.....	32
	Bibliografia.....	Erro! Marcador não definido.
	Anexos.....	33

Índice de Figuras

Figura 1 - Etapas do Estágio.....	3
Figura 2 - Web Site do TripAdvisor Attraction.....	5
Figura 3 - Aplicação Google Trips	6
Figura 4- Metodologia SCRUM.....	9
Figura 5- Diagrama de Contexto	10
Figura 6 - Diagrama de Casos de Uso	12
Figura 7- Diagrama de Sequência	14
Figura 8 - Diagrama de Classes.....	15
Figura 9 - MongoDB Compass.....	20
Figura 10 - Mailtrap.....	21
Figura 11- User Schema	22
Figura 12 - Metodo para Password Hash	23
Figura 13 - Configuração da Biblioteca "Mailer"	24
Figura 14 - Login.....	24
Figura 15 - Metodo para Gerar um Token.....	25
Figura 16 - Rota para Criar uma Viagem	26
Figura 17 - Mapa Mundo de Passport	27
Figura 18 - Rota para Obter as Viagens Realizadas	27
Figura 19 - Upload de Imagem.....	28
Figura 20 - Campos Obrigatórios	29

Figura 21 - Registo com email Indisponível	30
Figura 22 - Login com Password Incorreta	30
Figura 23 - Login com Email Incorreto.....	31
Figura 24 - Inserção de Viagem sem todos os Campos Obrigatórios Preenchidos.....	31

Índice de Tabelas

Tabela 1 - Comparação das Funcionalidades	7
Tabela 2 - Atores e Casos de Uso	11
Tabela 3 - Descrição do Caso de Uso "Registrar Utilizador"	13
Tabela 4 - Dicionário de Dados	16
Tabela 5 – Dicionário de Dados 2	17

1. Introdução

O curso de Engenharia Informática é uma licenciatura lecionada na Escola Superior de Tecnologia e Gestão do Instituto Politécnico da Guarda. O presente relatório foi realizado no âmbito da unidade curricular de Projeto de Informático, no meu caso em contexto de Estágio.

Esta unidade curricular tem como objetivo a realização de um projeto ou um projeto em contexto de estágio, consoante o interesse do aluno. No caso de um projeto em contexto de estágio este terá a duração mínima de 280 horas.

Foi, portanto, elaborado um projeto em contexto de estágio para avaliação da unidade curricular, sendo este realizado na empresa Pplware. Fui integrado numa equipa, constituída por dois elementos e um supervisor. O projeto foi batizado como *Passport* e consiste numa rede social com o intuito de os seus utilizadores serem capazes de partilhar as suas viagens bem como deixar *feedback* sobre elas para permitir que outros tenham opiniões sobre as suas próximas viagens.

1.1. Caracterização sumária da instituição

O projeto Pplware.com, que engloba vários canais de comunicação, pertence à empresa Pplware.com – Tecnologias de Informação e Serviços Web, LDA. Teve início a 18 de abril de 2005 e desde então tem expandido a sua área de ação para vários segmentos.

O *site* Pplware.com é atualmente o principal canal de tecnologia do Portal Sapo.

1.2. Motivação

Devido à grande afluência nas redes sociais de hoje em dia, à vontade de querer partilhar experiências e aventuras, junto com a minha paixão de viajar, notamos que os únicos sítios onde nos poderíamos aconselhar sobre viagens a fazer e sítios a visitar era em blogs e sites do mesmo género, em que as pessoas simplesmente limitavam-se a comentar a sua experiência. Com isto surgiu a ideia de criar uma plataforma onde poderíamos anotar as nossas viagens e por fim comentá-las dando a saber aos outros o que nos encantou ou não,

na cidade ou país que visitámos. Permitindo assim uma melhor análise do que fazer, do que visitar na nossa próxima viagem. Sendo possível também partilhá-las noutras redes sociais como Facebook e Instagram.

1.3. Descrição do Problema

Como referido em cima, sempre que procurávamos informação sobre a próxima viagem a concretizar, ou sobre o próximo país a visitar percebemos que essa pesquisa era um pouco difícil de ser eficazmente realizada, ou porque a informação não estava completa ou porque era necessário “saltar” de página em página, para coletar um pouco de informação em cada uma e com isto juntar todos os pontos de interesse em cada local.

1.4. Objetivos

Após algumas reuniões com todas as pessoas envolvidas no projeto, foram definidos os seguintes objetivos para o desenvolvimento da aplicação *web*.

- Plataforma online;
- Registrar utilizadores;
- *Login* do utilizador;
- Email de confirmação do registo;
- Associar a conta do utilizador às redes sociais;
- Inserção de países visitados para utilizadores registados;
- Consultar países visitados para utilizadores registados;
- Ver mapa mundo com países visitados para utilizadores registados;
- Inserir *feedback* na viagem realizada;
- Partilhar as viagens nas redes sociais;

- Criação de um *web* site responsivo;
- Validação de alguns campos;
- Otimização para melhor performance;

1.5. Etapas do Estágio

Todas as etapas para o estágio foram definidas como mostra a seguinte tabela.

Tarefas	Maio				Junho				Julho		
	S	1	2	3	4	1	2	3	4	1	2
0 Investigação				■							
1 Estado da Arte				■							
2 Análise das Tecnologias						■					
3 Aprendizagem das Tecnologias						■					
4 Sistema de Autenticação com <i>Tokens</i>							■				
6 Registo de Viagens									■		
7 Ambiente Gráfico										■	
8 Testes Finais e Conclusão											■

Figura 1 - Etapas do Estágio

1.6. Estrutura do documento

Este relatório é constituído por 8 capítulos e está estruturado da seguinte forma: no presente capítulo foi feita uma introdução sobre a aplicação *Web* desenvolvida, a motivação e os objetivos previstos para o desenvolvimento da mesma.

No capítulo seguinte apresenta-se o estado da arte para avaliar as aplicações similares, de forma a acrescentar valor à solução desenvolvida.

Quanto ao capítulo 3 descreve-se a metodologia usada pela equipa que foi definida tendo em vista o tempo que se tinha para alcançar os objetivos desejados.

No capítulo 4, é apresentada a análise de requisitos, contendo diagrama de casos de uso, diagrama de sequência, descrição de casos de uso, diagrama de classes do projeto, bem como o dicionário de dados.

No capítulo 5 são apresentadas as metodologias aprendidas e utilizadas para desenvolver a aplicação *Passport*.

O capítulo 6 refere-se à implementação e esta foi dividida em módulos para ser mais fácil implementar, testar e compreender como foi realizado neste projeto.

No capítulo 7 realizam-se alguns testes de validação e verificação da aplicação com vista a determinar o seu bom funcionamento de acordo com o planeado bem.

Por fim, no último capítulo, encontra-se a conclusão que descreve o trabalho desenvolvido como também alguma consideração mais relevante a ter, e as perspetivas de desenvolvimento que se pretendem efetuar no futuro.

2. Estado da Arte

Neste capítulo pretende-se analisar outras aplicações ou *web sites* com funcionalidades idênticas ao projeto desenvolvido, de modo a analisar a viabilidade e funções deste.

Após uma vasta pesquisa foram encontrados vários resultados que continham similaridades com o projeto apresentado, que, de seguida, serão analisados.

2.1. Aplicações existentes

Da pesquisa realizada e das várias aplicações analisadas conseguiu-se encontrar duas como sendo as mais interessantes para a comparação pretendida, sendo estas o TripAdvisor Attractions e o Google Trips.

2.1.1 TripAdvisor Attractions

O TripAdvisor Attractions [1] apresenta algumas funcionalidades que auxiliam o utilizador na pesquisa de locais turísticos bem como atividades tanto de lazer como de desporto. Nesta secção o utilizador tem a possibilidade de avaliar e ler as avaliações de outras pessoas e também publicar e ver fotografias. Para além destas funcionalidades, o utilizador pode também consultar preços, horários e outros detalhes sobre os locais pesquisados.

A Figura 2 apresenta a página inicial do TripAdvisor Attraction, com a sua barra de pesquisa de locais.



Figura 2 - Web Site do TripAdvisor Attraction

2.1.2. Google Trips

As principais funcionalidades do Google Trips [2] consistem no planeamento e organização de viagens. A criação de rotas para um determinado dia e local também é uma possibilidade, sendo que esta aplicação apresenta todos os pontos turísticos nas proximidades.

Assim como no TripAdvisor, o Google Trips permite o utilizador avaliar e ler as avaliações dos pontos de interesse bem como outras informações acerca do local.

Uma importante funcionalidade é a possibilidade de fazer o *download* do plano da viagem, possibilitando, assim, o seu acesso pelo utilizador, sem a necessidade de ligação à internet.

Na figura 3 são visíveis algumas das funcionalidades da aplicação

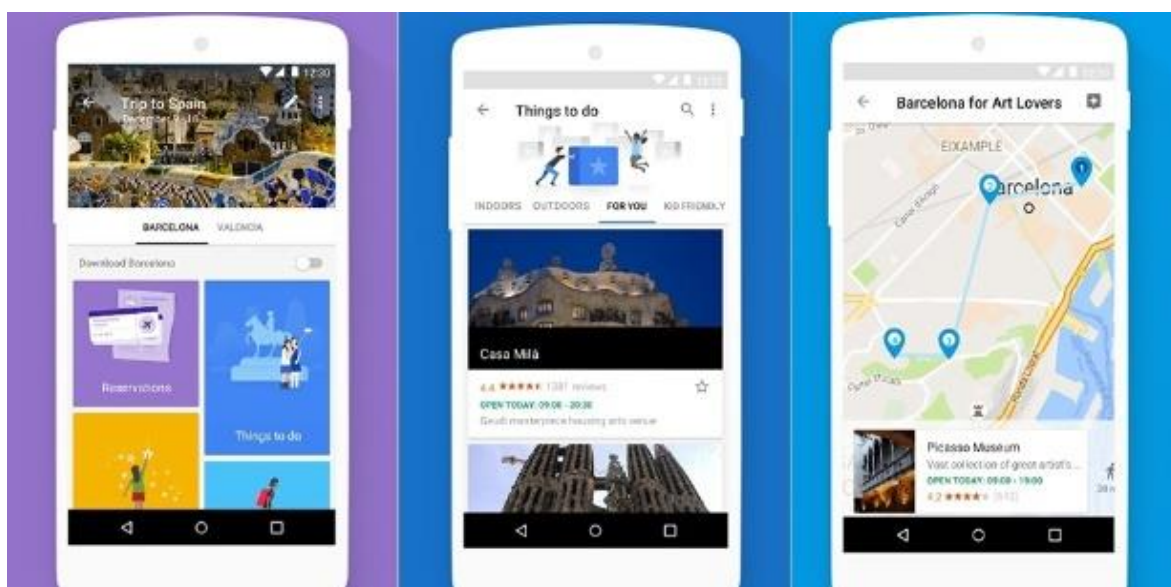


Figura 3 - Aplicação Google Trips

2.2. Análise crítica das aplicações analisadas

Após a análise das duas aplicações acima referidas, foi observado que estas contêm funcionalidades idênticas ao pretendido com o presente projeto. No entanto, nenhuma delas reúne os requisitos necessários para a resolução do problema já apresentado.

Desta forma, avançou-se com o projeto, uma vez que este se apresenta como viável e inovador, pelo facto de apresentar, de forma clara, uma resolução ao problema inicial.

A Tabela 1 compara de forma resumida as funcionalidades das aplicações analisadas com a aplicação a ser desenvolvida.

<i>Funcionalidades/Aplicações</i>	<i>TripAdvisor Attractions</i>	<i>Google Trips</i>	<i>Passport</i>
<i>Aplicação mobile</i>	✓	✓	✗
<i>Web Site</i>	✓	✗	✓
<i>Partilha de fotografias</i>	✓	✗	✗
<i>Mapa com países visitados realçados</i>	✗	✗	✓
<i>Feedback</i>	✓	✓	✓
<i>Seriação de utilizadores</i>	✗	✗	✓
<i>Métricas de comparação de utilizadores</i>	✗	✗	✓

Tabela 1 - Comparação das Funcionalidades [3]

Após a análise das duas aplicações existentes, foi necessário escolher a metodologia que melhor se adequa com as tarefas necessárias para o desenvolvimento do projeto com o tempo que nos foi disponibilizado para tal.

3. Metodologia

No desenvolvimento de uma aplicação *Web* é necessário seguir algumas normas e critérios. Existem diferentes tipos de metodologias de desenvolvimento de software, tais como, a metodologia orientada a objetos, metodologias de desenvolvimento ágil e metodologias estruturadas.

Optou-se pela escolha da metodologia de desenvolvimento ágil, mais precisamente pelo processo SCRUM [1], muito utilizado na gestão de projetos.

A metodologia SCRUM assume-se como uma metodologia extremamente flexível, que tem como objetivo definir um processo de desenvolvimento iterativo e incremental que pode ser aplicado a qualquer produto, proporcionando uma excelente interação entre as equipas de desenvolvimento. Com toda esta ligação entre membros e com a participação ativa dos clientes, obtém-se melhores resultados no desenvolvimento do projeto.

Jeff Sutherland aplicou a primeira conceção do SCRUM na Easel Corporation em 1993, mais tarde, por volta de 1995, Ken Schwaber melhorou essa metodologia devido à sua experiência no desenvolvimento de sistemas e processos.

3.1. Características do Scrum

- Os projetos são divididos em ciclos chamados de *Sprints*. O *Sprint* representa um espaço de tempo no qual um conjunto de tarefas deve ser executado;
- As funcionalidades a serem implementadas no projeto são mantidas numa lista que é conhecida como *Product Backlog*;
- No início de cada *Sprint*, é realizada uma reunião na qual o *Product Owner* prioriza os itens do *Product Backlog*, e a equipa seleciona as atividades que serão capazes de implementar durante essa *Sprint*;
- Realização de reuniões/discussões diárias sobre o progresso do trabalho no projeto, com o intuito de responder a 3 questões:

- O que foi realizado desde o dia anterior em direção ao objetivo final da *Sprint*?
- O que está planeado fazer no dia seguinte para atingir o objetivo da *Sprint*?
- Existe algum problema a impedir de atingir o objetivo?

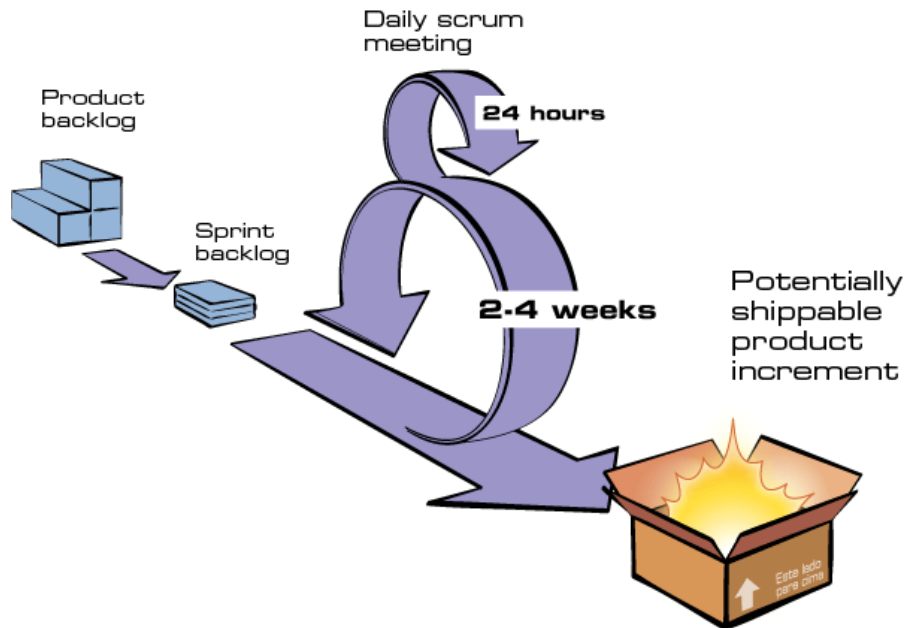


Figura 4- Metodologia SCRUM

A metodologia em causa tem várias secções, sendo estas parcialmente modificáveis, de modo a adequar o processo a cada projeto. A primeira secção é o *Product Backlog*. A segunda é o *Sprint Backlog* que consiste nos itens que têm maior prioridade a serem desenvolvidos. A terceira secção é o resultado conseguido durante a *Sprint*, de onde fará parte um protótipo do módulo que foi desenvolvido.

Por norma, cada *Sprint* tem uma duração de 2 a 4 semanas, contudo essa não foi a variação presente no projeto e por o tempo de desenvolvimento ser curto adotou-se intervalos de 1 a 2 semanas.

No estágio existiram pontos de situação diários com os membros da equipa e pontos de situação semanais com os membros superiores do projeto.

4. Análise de Requisitos

A análise é o processo de observação e levantamento dos elementos do ambiente onde o *software* será implementado. Devem-se identificar as pessoas que terão contacto com o *software*, quer seja um contacto operacional ou para o fornecimento de informações relevantes para o seu desenvolvimento.

Com base nos objetivos propostos pelo Pplware foi realizada uma análise que define quais os requisitos do sistema a desenvolver. No capítulo seguinte vamos apresentar os diagramas de casos de uso, diagramas de classes, algumas descrições de casos de uso, os principais diagramas de sequência e o dicionário de dados.

4.1. Diagrama de Contexto

O Diagrama de Contexto consiste numa representação dos fluxos de dados trocados entre o sistema (Passport) e as entidades externas (utilizadores), permitindo assim, identificar os módulos a serem realizados.

A figura 5 seguinte ilustra o diagrama de contexto relativamente à nossa aplicação.

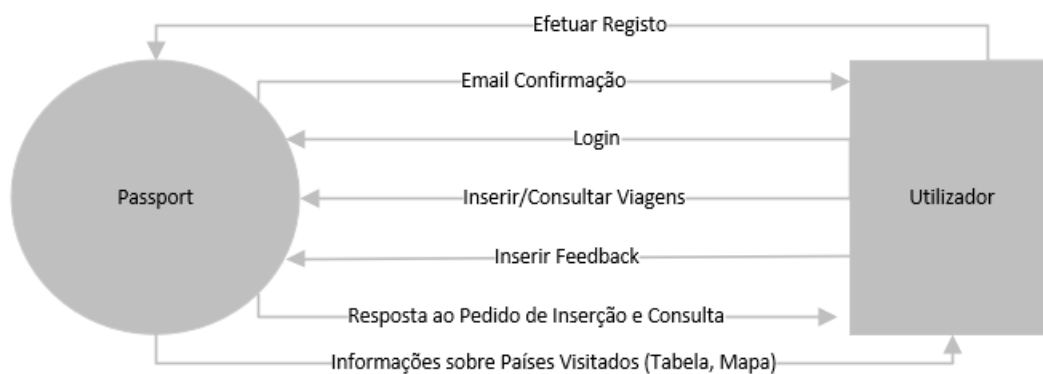


Figura 5- Diagrama de Contexto

4.2. Atores e respetivos casos de uso

A seguinte tabela representa os atores, que são quem interage com o sistema, neste caso o utilizador, e os casos de uso que representam os requisitos de um sistema computacional.

Ator	Objetivos – Casos de Uso
Utilizador	Registo
	Email de Confirmação
	<i>Login</i>
	Inserir Países Visitados
	Consultar Países Visitados
	Ver Mapa Mundo c/ Países Visitados
	Inserir <i>Feedback</i>
	Métricas
	Serição
	Associar conta às Redes Sociais
	Partilhar Viagens nas Redes Sociais

Tabela 2 - Atores e Casos de Uso

4.3. Diagrama de Casos de Uso

Um diagrama de casos de uso descreve a funcionalidade proposta de um novo sistema, ou seja, permite ver de forma simples e rápida todas as funcionalidades do projeto, assim como todas as interações com os seus atores.

Na figura seguinte está demonstrado o diagrama de casos de uso para a aplicação Passport.

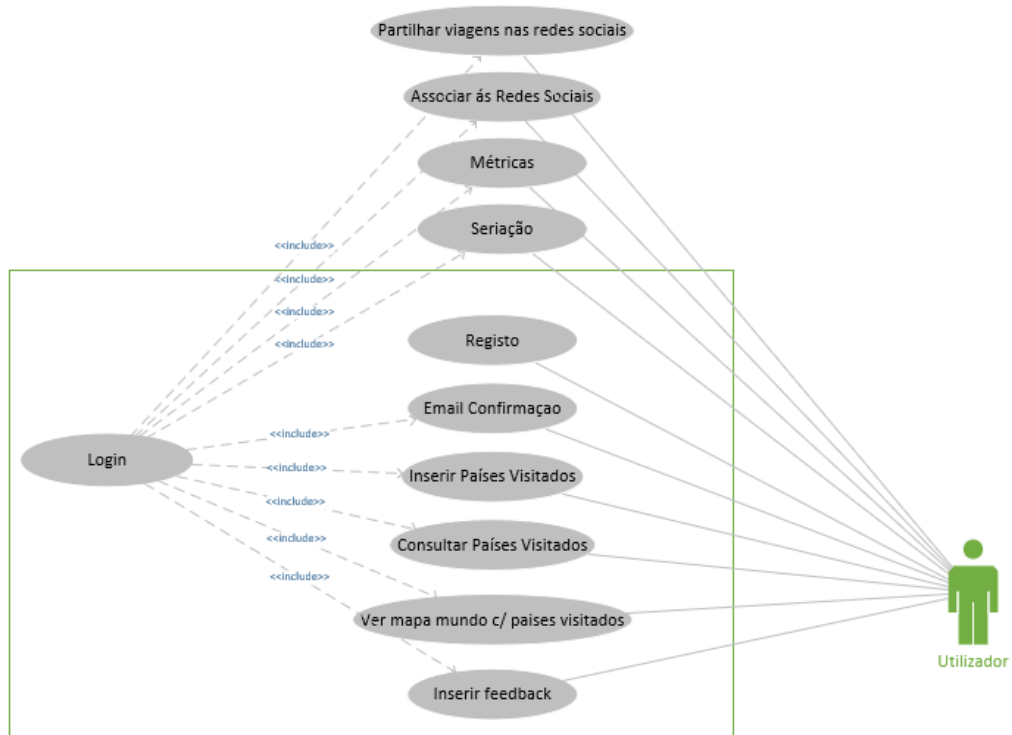


Figura 6 - Diagrama de Casos de Uso

4.4. Descrição dos Casos de Uso

Os casos de uso descrevem uma interação entre o utilizador e o sistema. Na descrição de cada caso entende-se que este segue o caminho principal, ou seja, as condições para que tudo corra bem estão reunidas. No entanto, poderá haver a necessidade de fazer uma descrição de caminhos alternativos.

4.4.1. Registrar utilizador

Nome	Registrar Utilizador
Descrição	Neste caso de uso o ator efetua o registo na aplicação <i>Passport</i>
Pré-Condição	--
Caminho Principal	<ol style="list-style-type: none"> 1. O ator escolhe a opção “<i>SignUp</i>”. 2. O sistema mostra um formulário para criar uma conta na aplicação. 3. O ator preenche todos os campos do formulário (<i>First Name, Last Name, Email, Birthday Date, Password</i>) 4. O sistema aceita o registo e direciona o ator para o <i>Dashboard</i>
Caminho Alternativo	<ol style="list-style-type: none"> 4.1. O ator não preenche todos os campos. 4.2. O ator introduz um email que já está em uso 4.3. O ator introduz a <i>password</i> incorreta
Pós-Condição	No fim de o utilizador estar devidamente registado, o sistema envia uma mensagem para o email a pedir para efetuar a sua verificação.
Suplementos ou Adornos	Realizar um teste em que se verifique que o utilizador introduziu um email que já está a ser usado na aplicação.

Tabela 3 - Descrição do Caso de Uso "Registrar Utilizador"

4.5. Diagrama de Sequência

Um diagrama de sequência descreve a maneira como os grupos de objetos colaboram nos comportamentos entre o sistema e os atores. Nele são registados os comportamentos de um único caso de uso e mostra os objetos e as mensagens passadas entre estes.

Na figura 7 está representado o diagrama de sequência do caso de uso “Registrar Utilizador”.

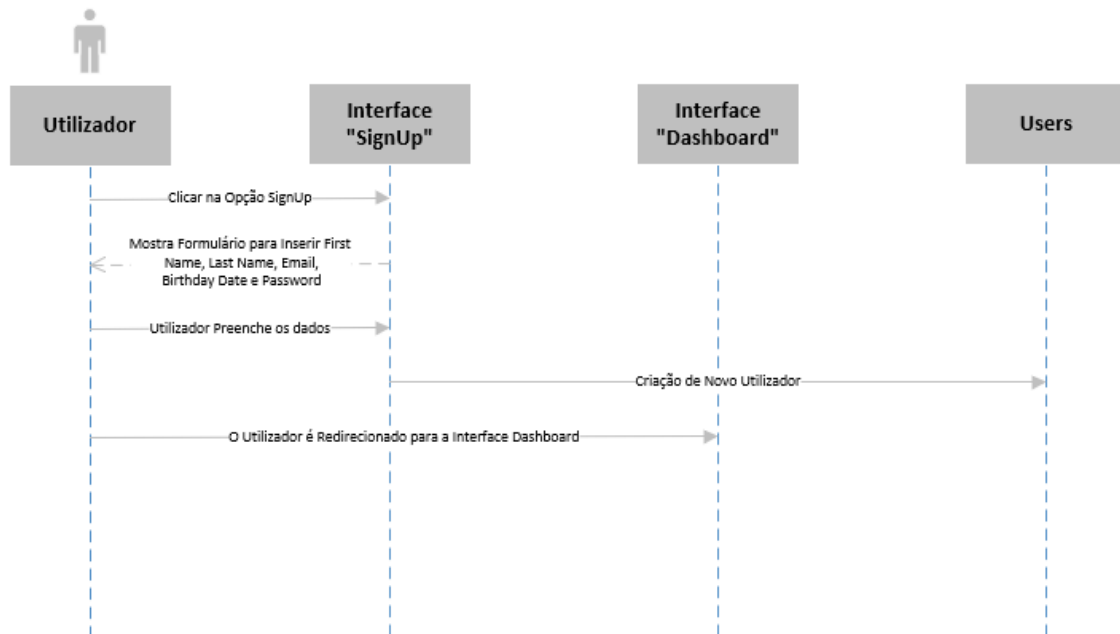


Figura 7- Diagrama de Sequência

4.6. Diagrama de Classes

Nesta secção é possível ver o diagrama de classes referente ao nosso projeto, representado pela figura 8. O diagrama mostra como as diferentes classes se relacionam entre si, cada classe é constituída por um nome, atributos e por fim as operações dos atores no sistema.

O nosso projeto está dividido em duas classes até ao momento. A classe “Users”, onde se junta todos os dados que se achou pertinente guardar relativamente a cada utilizador da nossa aplicação. E por fim, a classe “viagens” que está relacionada à outra classe existente no projeto, pelo facto de cada utilizador ter as suas próprias viagens, guardamos também os dados referentes a cada aventura realizada pelo público do Passport.

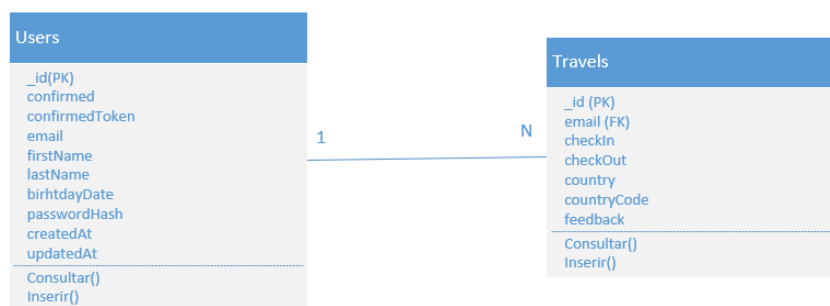


Figura 8 - Diagrama de Classes

4.7. Dicionário de Dados

Nesta secção é possível ver o dicionário de dados (Tabela 4 e Tabela 5) referente à aplicação Passport, que consiste em especificar todos os campos que são guardados, o tipo de dados desses campos, uma breve descrição de cada um deles, bem como todos os valores que podem ser introduzidos e as restrições que achamos por bem cada campo ter.

Nome do Campo	Tipo de Dados	Descrição	Valores Válidos	Restrições
ID_User (PK)	ObjectId	Chave primária gerada automaticamente pela base de dados;	Letras e números	Único; Criado pelo sistema; Não alterável;
FirstName	String	Corresponde ao primeiro nome do utilizador;	Letras e espaços	Obrigatório;
LastName	String	Corresponde ao último nome do utilizador;	Letras e espaços	Obrigatório;
Email	String	Corresponde ao email do utilizador;	Letras, números, pontos e arroba	Obrigatório;
BirthdayDate	Date	Corresponda à data de nascimento do utilizador;	Datas	Obrigatório; Idade mínima de 16 anos;
PasswordHash	String	Corresponde ao Hash da password do utilizador;	Letras, números, pontos e símbolos;	Obrigatório;
Confirmed	Boolean	Corresponde ao estado de confirmação da conta do utilizador;	True ou false	Alterável pelo sistema;
ConfirmationToken	String	Corresponde ao token para validar a conta;	Letras, números, pontos e símbolos	Único; Gerado pelo sistema;
CreatedAt	Date	Corresponde à data de criação da conta do utilizador;	Datas	Não alterável; Gerado pelo sistema;
UpdatedAt	Date	Corresponde à data de alteração da conta do utilizador;	Datas	Alterável; Gerado pelo sistema;

Tabela 4 - Dicionário de Dados

Nome do Campo	Tipo de Dados	Descrição	Valores Válidos	Restrições
ID_Travel (PK)	ObjectId	Chave primária gerada automaticamente pela base de dados;	Letras e números	Único; Criado pelo sistema; Não alterável;
Email (FK)	String	Corresponde ao email do utilizador;	Letras, números, pontos e arroba	Obrigatório;
CheckIn	Date	Corresponda à data de início da viagem	Datas	Obrigatório; Datas anteriores à data do sistema;
CheckOut	Date	Corresponda à data de fim da viagem	Datas	Obrigatório; Datas anteriores à data do sistema;
Country	String	Corresponda ao país visitado	Letras e espaços	Obrigatório;
CountryCode	String	Corresponda ao código do país visitado	Leras e espaços	Obrigatório; Inserido automaticamente com base no país selecionado;
Feedback	String	Corresponda ao feedback da viagem	Letras, números, espaços e símbolos	Obrigatório;

Tabela 5 – Dicionário de Dados 2

5. Tecnologias

Com base no estudo realizado na análise de requisitos, procedemos à escolha das tecnologias que se melhor adequavam ao desenvolvimento deste projeto, mas tendo em atenção as tendências da atualidade e com o pensamento no futuro.

Neste projeto foram usadas as tecnologias que compõem o MERN [4], que tem em comum a linguagem de programação JavaScript [5]. Na parte do *back-end* do *Passport* foi utilizado NodeJS [6], Express [7] e MongoDB [8] para base de dados.

5.1. JavaScript

Como linguagem de programação optou-se por JavaScript. Esta não foi uma linguagem que aprendemos durante a licenciatura, contudo era a linguagem em comum do MERN mas também pela sua versatilidade. Devido à facilidade em encontrar tanta informação sobre a linguagem podemos considerar que foi de fácil aprendizagem.

É atualmente a principal linguagem para programação do lado do cliente, contudo começa a ser bastante utilizada também do lado do servidor através de ambientes como o NodeJS.

5.2. NodeJS

Como tecnologia para executar no servidor, optou-se por NodeJS que é *open source* e corre em várias plataformas como Windows, Linux, Unix, entre outros. Interpreta código JavaScript e é baseado no V8 JavaScript Engine que é um compilador de JavaScript da Google utilizado pelo Chrome, permitindo assim a utilização de outras linguagens de programação como TypeScript e CoffeeScript.

Devido à maneira única de gerir pedidos, o NodeJS torna-se mais eficiente que a sua grande concorrência, Apache HTTP Server, ASP.NET e Ruby on Rails.

O NodeJS é a tecnologia mais adequada por suportar vários protocolos como HTTP, HTTPS, DNS, entre outros e seguir um modelo não-bloqueante, ou seja, os pedidos serão feitos e entregues apenas e quando estiverem prontos.

5.3. Express

O Express é um *dos frameworks* mais populares, escrito em JavaScript e hospedado dentro do próprio ambiente de execução NodeJS. Tem-se tornado a *framework standard* para o NodeJS, fazendo parte do *back-end* numa aplicação MEAN [9] ou MERN, a única diferença entre estes dois de stack é no MERN o *front-end* usar a tecnologia AngularJS. [10]

O Express permite gerar todo o “esqueleto” de uma aplicação *web* com o mínimo de linhas de código escritas por parte do programador, não está limitado a um sistema operativo, e suporta a arquitetura MVC (Model-View-Controller).

5.4. MongoDB

MongoDb é um software de base de dados, *open source* e multiplataforma, não relacional, guarda dados de modo flexível, em documentos do tipo JSON, o que significa que a estrutura dos dados pode ser alterada facilmente quando necessário.

Permite *index* automático sem a introdução de nenhum atributo dentro da coleção referente a uma chave primária. Permite também relacionar duas tabelas de uma forma fácil e sem grande esforço por parte do programador como podemos ver na Figura 9 (página seguinte).

5.5. MongoDBCompass

O MongoDB Compass [11] é uma interface gráfica onde é possível observar a base de dados, realizar *queries* e testes. Teve uma grande utilidade no do projeto na fase de realização dos testes, no registo de um utilizador para verificar o sucesso da adição na base de dados e a verificação da correção dos atributos, o mesmo se aplica ao adicionar viagens.

De seguida é apresentada está uma imagem onde é possível ver os documentos da coleção de “Users” que são criados quando é gerado um novo utilizador na aplicação *Web*.

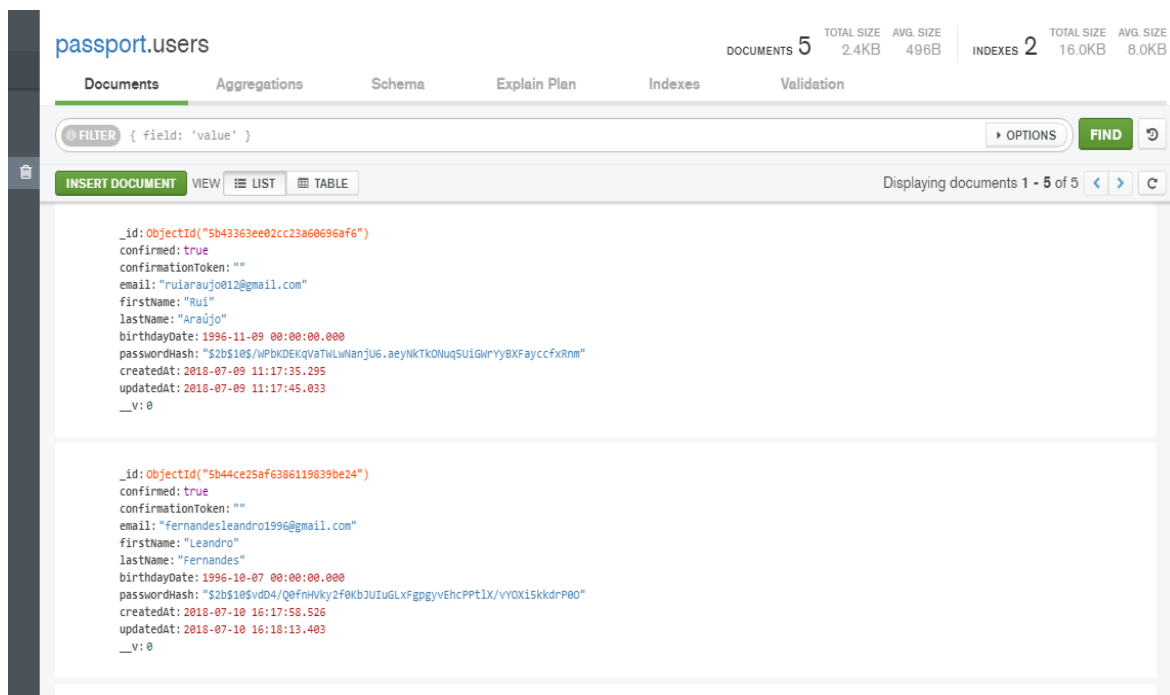


Figura 9 - MongoDB Compass

5.6. Yarn

O Yarn [12] é um gestor de pacotes para JavaScript bem como o NPM [13], contudo o gestor utilizado neste projeto traz alguns benefícios em relação ao seu concorrente, começando por guardar os *packages* em cache, não sendo necessário fazer o *download* deles de novo. Consegue correr operações em paralelo maximizando assim os recursos. As dependências vão ser instaladas sempre da mesma forma independentemente da ordem de instalação delas ou de qualquer que seja a máquina onde estão a ser instaladas. É capaz de identificar dependências e as suas versões, não permitindo assim que seja possível existir as mesmas dependências com versões diferentes causando assim conflitos na aplicação.

5.7. Github

O Github [14] é um repositório *open source*, baseado em tecnologia *cloud* para a gestão do código das nossas aplicações. É capaz de guardar o código numa grande variedade de linguagens de programação e preservar todas as alterações que são realizadas durante o desenvolvimento do projeto.

Tratando-se o presente trabalho, de um trabalho em equipa tornou-se mais fácil pelo facto de sermos capazes de estar sempre em sintonia no que cada um dos membros da equipa

realizou mesmo quando não era possível estar em contacto de momento, permitindo gerir versões e também estar a trabalhar ao mesmo tempo conseguindo qualquer elemento fazer o *download* da última versão realizada pelo outro, e assim, executar os testes necessários na aplicação. De facto, uma das ferramentas mais indispensáveis na realização deste projeto.

5.8. Postman

O Postman [15] tem várias utilidades, contudo no presente caso foi utilizado principalmente para testar rotas *post*, *get*, entre outras, que foram implementadas na aplicação *Web*. Como por exemplo, se por ventura a parte do *back-end* estava a avançar mais rapidamente que o *front-end*, era utilizado o Postman para testar se a rota estava corretamente construída, economizando tempo que poderia ser aproveitado para implementar novas *features*.

5.9. Mailtrap

O Mailtrap [16] tornou-se para a equipa uma ferramenta muito útil, poupando muito tempo em testes. Esta consiste em simplesmente guardar todos os emails que são enviados do servidor, ou seja, em vez dos emails de confirmação serem diretamente enviados para o nosso utilizador eram guardados na *inbox* do Mailtrap. Isto tornava possível criar um utilizador com um email que não existe, contudo iríamos receber na mesma no Mailtrap o conteúdo do email enviado, permitindo validar a conta do utilizador, realizando assim todos os testes necessários.

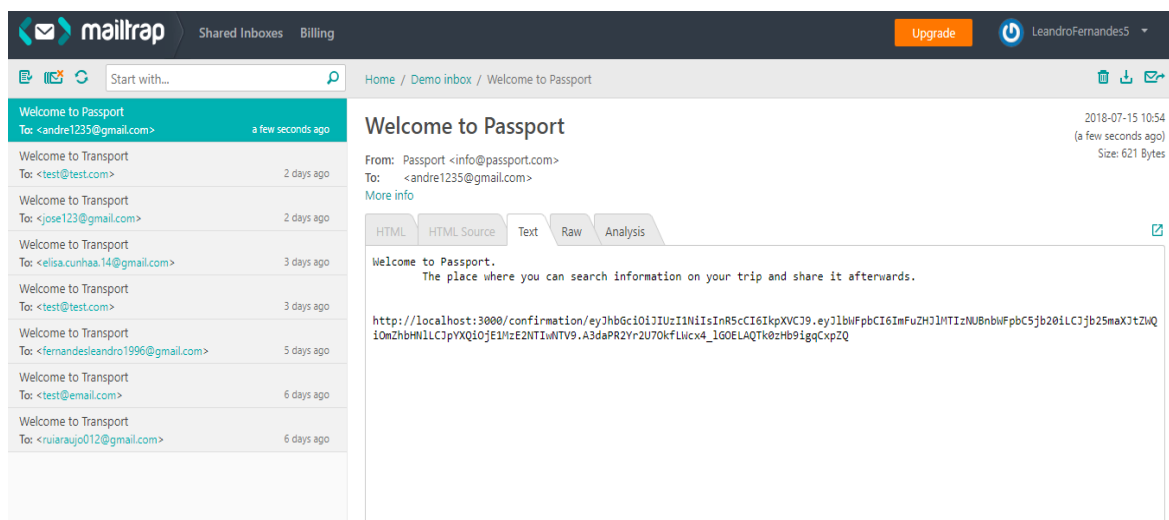


Figura 10 - Mailtrap

6. Implementação

Com base no estudo elaborado na análise de requisitos e à escolha das tecnologias a adotar no projeto, procedeu-se ao seu desenvolvimento. Algumas das tecnologias escolhidas não foram apresentadas durante a licenciatura, contudo, o foco foi utilizar tecnologias em ascendência com olhos no futuro da aplicação em desenvolvimento e num possível crescimento da mesma.

Durante a implementação da aplicação *Web* foi necessário fazer alguns ajustes aos requisitos inicialmente definidos, no entanto procurou-se sempre efetuar estas alterações visando uma melhor e mais rápida aplicação.

Neste capítulo irá ser apresentado tudo que foi realizado no curto espaço de tempo definido para o estágio curricular. Foram definidos vários módulos que se apresentam de seguida.

6.1. Sign Up

Neste módulo foi desenvolvido um registo de conta, onde se decidiu pedir ao utilizador o primeiro e último nome, o email, a data de nascimento, e uma *password*. Foi também realizada a validação destes campos (Figura 11).

```
const schema = new mongoose.Schema({
  email: {
    type: String,
    required: true,
    lowercase: true,
    index: true,
    unique: true
  },
  passwordHash: {type: String, required: true},
  confirmed: {type: Boolean, default: false},
  confirmationToken: {type: String, default: ""},
  firstName: {type: String, required: true},
  lastName: {type: String, required: true},
  birthdayDate: {type: Date, required: true}
},
  {timestamps: true}
);
```

Figura 11- User Schema

6.2. Password

Com o desenvolvimento do módulo referido acima surgiu um outro módulo dentro do anterior, que consiste na *password* não ser guardada em "plain text" na base de dados, ou seja, por razões de segurança decidiu-se ao invés guardar a *hash* da *password*. Este submódulo foi facilmente alcançado através da biblioteca *bcrypt* [17] e da criação de uma simples função que é depois chamada quando é inserido um novo utilizador na base de dados (Figura 12). Depois desta inserção é enviado um email de confirmação para o utilizador e enquanto esse email não for validado é negada algumas funções na aplicação, como poderemos ver no módulo seguinte.

```
schema.methods.setPassword = function setPassword(password) {  
  this.passwordHash = bcrypt.hashSync(password, 10); //hashing  
};
```

Figura 12 - Método para Password Hash

6.3. Confirmação de Email

Neste módulo foi gerado um email de confirmação depois de utilizador efetuar o seu registo. Enquanto esse email não for confirmado é negada a principal função da aplicação que é adicionar viagens, para gerir e testar este módulo foi usado o Mailtrap, tecnologia referida no capítulo anterior. O email contém algumas frases relativas à aplicação Passport bem como um *link* que permite confirmar a sua conta.

Para alcançar o objetivo deste módulo foi utilizada a biblioteca "mailer" [18] que trata do envio do email para o destinatário, onde é possível definir o conteúdo deste bem como alguns outros parâmetros padrão (Figura 13).

```
export function sendConfirmationEmail(user) {
  const transport = setup();
  const email = {
    from,
    to: user.email,
    subject: "Welcome to Transport",
    text: `Welcome to Transport.
    The place where you can search information on your trip and share it afterwards.

    ${user.generateConfirmationUrl()}
  `
  }
  transport.sendMail(email, function (err, info) {
    if(err)
      console.log(err)
    else
      console.log(info);
  });
}
```

Figura 13 - Configuração da Biblioteca "Mailer"

6.4. Login

Neste módulo, como se definiu que o email numa fase inicial seria único para cada utilizador decidiu-se então ser esse o atributo que se iria ter em conta quando fosse necessário encontrá-lo na base de dados (Figura 14). Ao efetuar o *login* caso o email já exista na coleção, significa que esse utilizador já está registado, então, e se a *password* estiver correta o utilizador é redirecionado para o *dashboard*.

```
router.post('/', (req, res) => {
  const { credentials } = req.body;
  User.findOne({ email: credentials.email }).then(user => {
    if (user && user.isValidPassword(credentials.password)) {
      res.json({ user: user.toAuthJSON() });
    } else {
      res.status(400).json({ errors: { global: "Invalid credentials" }});
    }
  });
});
```

Figura 14 - Login

6.5. Tokens

Como não poderia, decidiu-se implementar, talvez a medida de segurança mais importante em qualquer aplicação, os *tokens*.

Sempre que o utilizador efetua o *login* é gerado um *token* e todas as comunicações entre a *front-end* e o servidor necessita de *token* válido para ser permitida a troca de informação. Se o utilizador fizer *logout* e de novo *login*, é gerado um novo *token* (Figura 15), que é atualizado e guardado na base de dados. Dentro dele, vão alguns dados importantes para a troca de informação, como o email do utilizador e os dados da viagem se for, por exemplo, um pedido “post”. Para alcançar esta medida, foi utilizada a biblioteca `JsonWebToken` [19].

```
schema.methods.generateJWT = function generateJWT() {
  return jwt.sign(
    {
      email: this.email,
      confirmed: this.confirmed
    },
    process.env.JWT_SECRET
  );
};
```

Figura 15 - Método para Gerar um Token

6.6. Inserir Viagens

No módulo seguinte, (Figura 16) é mostrada a forma que foi implementada para adicionar a viagem. Esta forma consiste no recebimento por parte do *front-end* um *header* que contém um *token*, nesse *token* está o email e assim foi conseguido introduzir na base de dados as viagens respetivas àquele utilizador. Definiu-se que os parâmetros a serem guardados seria a data de início da viagem (*checkIn*), data de fim (*checkOut*), o país visitado (*country*), o código desse país (*countryCode*), o *feedback* que os utilizadores gostariam de partilhar com os outros utilizadores e o email referente à pessoa com *login* efetuado, para ser possível efetuar a ligação com a coleção "*Users*".

```
router.post('/', /*upload.single('travelImage'),*/ (req, res) => {  
  
  const header = req.headers.authorization;  
  let token;  
  
  if (header) token = header.split(" ")[1];  
  
  const decoded = jwt.decode(token);  
  
  const { checkIn, checkOut, country, countryCode, feedback } = req.body.travels;  
  const email = decoded.email;  
  //const travelImage = req.file.path;  
  
  const travels = new Travels({  
    |   checkIn,  
    |   checkOut,  
    |   country,  
    |   countryCode,  
    |   email,  
    |   feedback  
  });  
  travels.save()  
  .then(() => res.json({  
    |   message: 'Added Travel Successfully'  
  }));  
});
```

Figura 16 - Rota para Criar uma Viagem

6.7. Informações sobre as viagens realizadas

No módulo seguinte, como podemos ver na Figura 17 o mapa mundo contém alguns países selecionados a azul que se referem aos países já visitados por parte do utilizador. Para ser possível obter esta informação foi necessário realizar uma procura na base de dados sobre os respetivos países.

Esta informação é alcançada através do *token* que é passado no pedido ao servidor, o *token* é decodificado e com isso obtém-se o email do utilizador conseguindo, assim, encontrar na base de dados as viagens correspondentes a esse email.

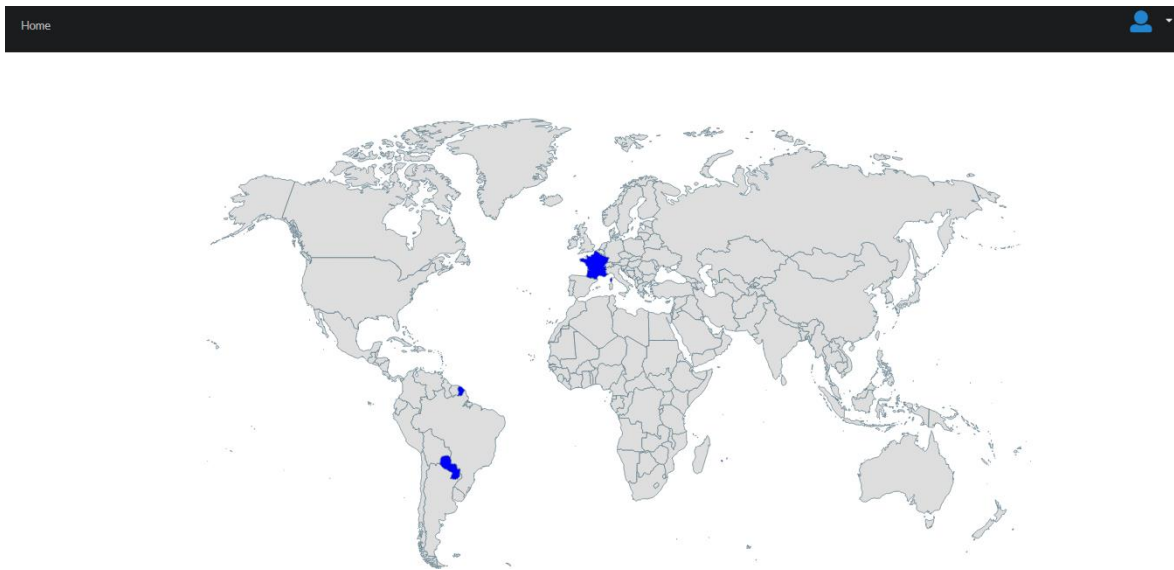


Figura 17 - Mapa Mundo de Passport

Na figura seguinte é apresentado como se conseguiu obter o email de um utilizador permitindo assim guardar a viagem referente ao mesmo.

```
router.get('/', (req, res) => {  
  
  const header = req.headers.authorization;  
  let token;  
  
  if (header) token = header.split(" ")[1];  
  
  const decoded = jwt.decode(token);  
  
  const email = decoded.email;  
  
  Travels.find({ email : [email] }, function(err, travels) {  
    if(err) {  
      res.status(400).json({message: 'Something went wrong'});  
    } else {  
      res.json(travels);  
    }  
  });  
});
```

Figura 18 - Rota para Obter as Viagens Realizadas

6.8. Adicionar Imagem

Este módulo tem como objetivo adicionar uma imagem sempre que é adicionada uma viagem, contudo à data de escrita do presente relatório não é possível a adição de imagens porque falta colmatar alguns pontos na parte do *front-end*. Para alcançar esse objetivo foi utilizada a biblioteca “multer” [20] que permite ajustar alguns parâmetros como, o tamanho do ficheiro que pode ser aceiteado, bem como filtrar qual o formato de imagem (PNG ou JPEG). O *multer* permite guardar o caminho da imagem, alterá-lo se necessário, ou guardá-la localmente, e definir o tipo de dados como guardamos a imagem, binário ou outra forma que se adegue.

```
const storage = multer.diskStorage({
  destination: function (req, file, cb) {
    cb(null, './uploads/');
  },
  filename: function (req, file, cb) {
    cb(null, new Date().toISOString() + file.originalname);
  }
});

const fileFilter = (req, file, cb) => {
  //reject File
  if (file.mimetype === 'image/jpeg' || file.mimetype === 'image/png') {
    cb(null, true); //this will simple ignore the file without store it
  } else {
    cb(null, false);
  }
}

const upload = multer({
  storage: storage,
  limits: {
    fileSize: 1024 * 1024 * 5
  },
  fileFilter : fileFilter
});
```

Figura 19 - Upload de Imagem

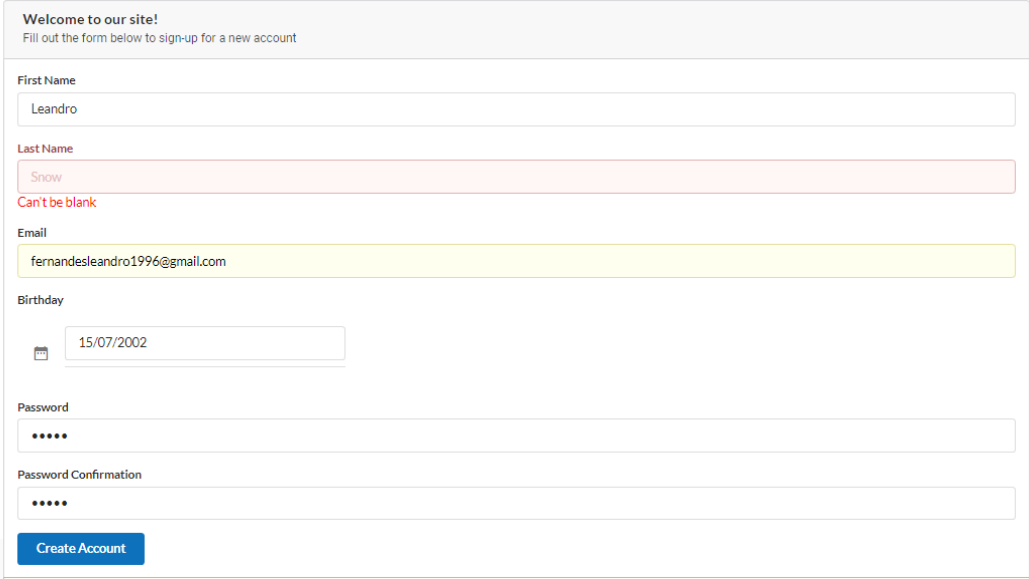
7. Verificação e Validação

Neste capítulo são realizados vários tipos de testes para garantir o bom funcionamento da aplicação *Web*, bem como encontrar falhas para que possam posteriormente ser corrigidas.

Para além dos testes demonstrados de seguida, foram também realizados alguns durante todo o desenvolvimento do projeto com o intuito de validar as funcionalidades desenvolvidas.

Estes testes foram realizados pela equipa de desenvolvimento bem como por possíveis utilizadores da aplicação, de modo a ser possível ter uma vasta e variada ideia do que poderá correr mal.

Na Figura 20 é demonstrado o exemplo de um teste realizado quando o utilizador efetua o registo na aplicação *Passport*. Neste teste valida-se se o utilizador preenche todos os campos, visto que todos eles são obrigatórios.



Welcome to our site!
Fill out the form below to sign-up for a new account

First Name
Leandro

Last Name
Snow
Can't be blank

Email
fernandesleandro1996@gmail.com

Birthday
15/07/2002

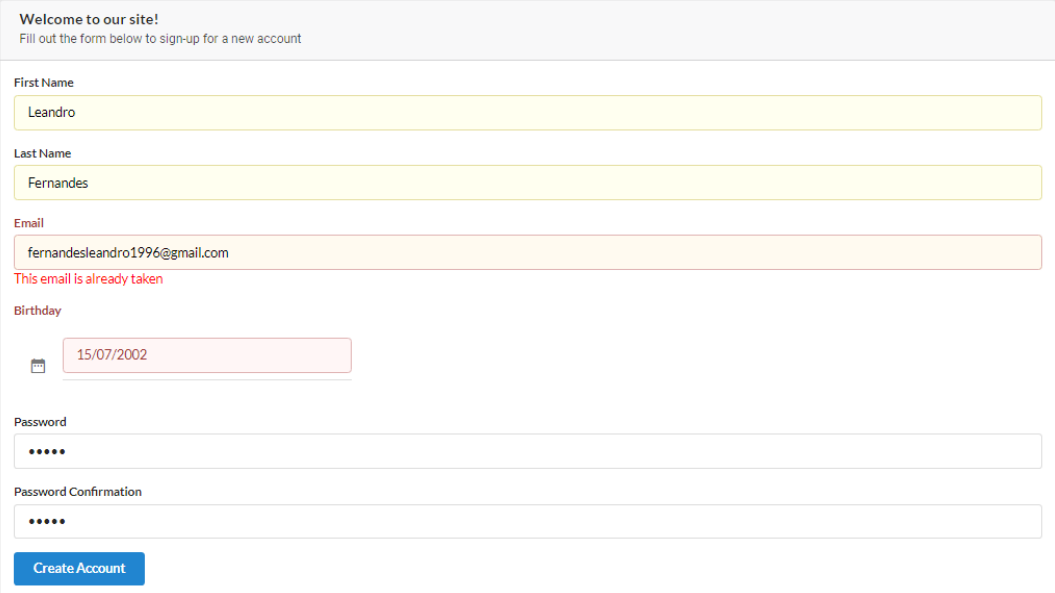
Password

Password Confirmation

Create Account

Figura 20 - Campos Obrigatórios

No teste apresentado na Figura 21, foi feito o registo com um email já previamente registado por outro utilizador.



Welcome to our site!
Fill out the form below to sign-up for a new account

First Name
Leandro

Last Name
Fernandes

Email
fernandesleandro1996@gmail.com
This email is already taken

Birthday
15/07/2002

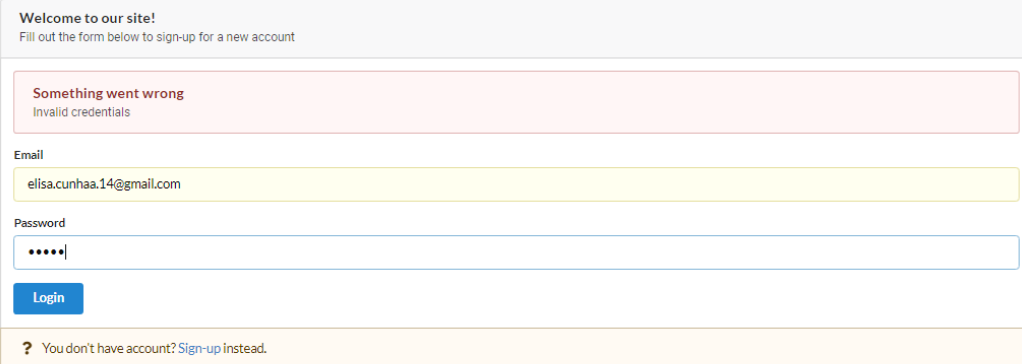
Password
.....

Password Confirmation
.....

Create Account

Figura 21 - Registo com Email Indisponível

No teste mostrado de seguida, Figura 22, foi analisado o *login* de um utilizador com uma *password* incorreta.



Welcome to our site!
Fill out the form below to sign-up for a new account

Something went wrong
Invalid credentials

Email
elisa.cunhaa.14@gmail.com

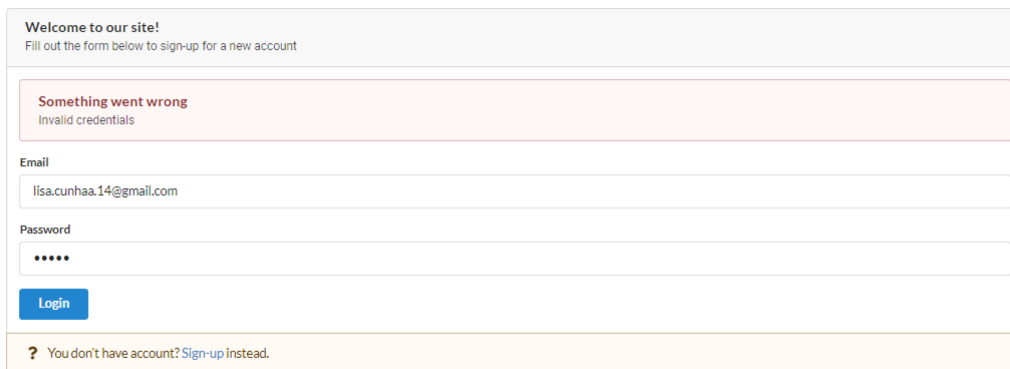
Password
.....

Login

? You don't have account? [Sign-up](#) instead.

Figura 22 - Login com Password Incorreta

Decidiu-se testar também com um email que não existe na base de dados, tal como se pode ver na Figura 23.



Welcome to our site!
Fill out the form below to sign-up for a new account

Something went wrong
Invalid credentials

Email
lisa.cunhaa.14@gmail.com

Password
•••••

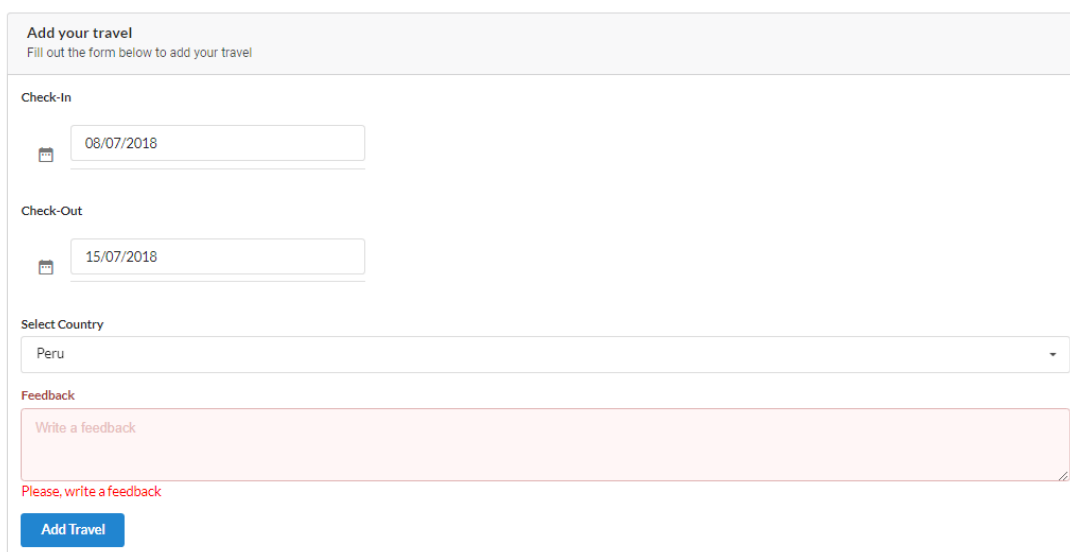
Login

? You don't have account? [Sign-up](#) instead.

Figura 23 - Login com Email Incorreto

Apesar de o teste ser muito idêntico, achamos importante realçá-lo. Visto que consideramos uma medida de segurança muito importante o seguinte, apesar de num dos testes a *password* estar incorreta e no outro estar o email, é crucial notar que o erro fornecido pela nossa aplicação é o mesmo. Isto permite, não fornecer informação extra a um *hacker*, visto que se o nosso erro fosse “invalid password” estávamos a dar a informação de que de facto existe um utilizador na base de dados com este email e isto poderia levar a um “brute force attack”.

Por fim, no último é demonstrado que ao inserir uma viagem é necessário também preencher todos os campos obrigatórios (Figura 24).



Add your travel
Fill out the form below to add your travel

Check-In
08/07/2018

Check-Out
15/07/2018

Select Country
Peru

Feedback
Write a feedback

Please, write a feedback

Add Travel

Figura 24 - Inserção de Viagem sem todos os Campos Obrigatórios Preenchidos

8. Conclusão

A plataforma desenvolvida tem como objetivo se tornar numa rede social para viajantes, onde é possível partilhar as viagens dos mesmos bem como algum *feedback*. Devido á complexidade do projeto foi impossível terminá-lo durante o estágio, com tudo, é um objetivo e será um prazer continuar a desenvolver a aplicação, com o intuito de a melhorar, podendo um dia a partilhar com o mundo.

Algumas das dificuldades com a qual nos deparamos foi como já referido anteriormente, o tempo, a necessidade de aprender todas as novas tecnologias para desenvolver a aplicação que nos ocupou tempo que poderia ter sido utilizado a implementar novas *features* na aplicação *Passport*.

Concluindo este relatório, é de salientar o trabalho desenvolvido, pessoalmente é me satisfatório. Tendo assim a certeza que tudo que foi aprendido neste projeto, será uma mais valia para o meu futuro profissional, mas também pessoal devido às conquistas realizadas.

BIBLIOGRAFIA

- [1] [Online]. Available: <https://www.tripadvisor.pt/Attractions>.
- [2] [Online]. Available: https://play.google.com/store/apps/details?id=com.google.android.apps.travel.onthego&hl=pt_PT.
- [3] R. e. L. (Pplware), *Estado da Arte do Projeto Passport*, 2018.
- [4] [Online]. Available: <http://mern.io/>.
- [5] [Online]. Available: <https://www.w3schools.com/jS/default.asp>.
- [6] [Online]. Available: <https://nodejs.org/en/>.
- [7] [Online]. Available: <https://expressjs.com/>.
- [8] [Online]. Available: <https://www.tutorialspoint.com/mongodb/>.
- [9] [Online]. Available: <http://mean.io/>.
- [10] [Online]. Available: <https://angularjs.org/>.
- [11] “MongoDbCompass,” [Online]. Available: <https://www.mongodb.com/products/compass>.
- [12] “Yarn,” [Online]. Available: <https://yarnpkg.com/lang/en/>.
- [13] “NPM,” [Online]. Available: <https://www.npmjs.com/>.
- [14] “GitHub,” [Online]. Available: <https://github.com/>.
- [15] “Postman,” [Online]. Available: <https://www.getpostman.com/>.
- [16] “Mailtrap,” [Online]. Available: <https://mailtrap.io/>.
- [17] “Bcrypt,” [Online]. Available: <https://github.com/kelektiv/node.bcrypt.js/>.
- [18] “W3Schools-Mailer,” [Online]. Available: https://www.w3schools.com/nodejs/nodejs_email.asp.
- [19] “JsonWebToken,” [Online]. Available: <https://github.com/auth0/node-jsonwebtoken>.
- [20] “Multer,” [Online]. Available: <https://github.com/expressjs/multer>.

Anexos

Visto que o projeto em contexto de estágio foi realizado numa empresa, não é possível divulgar todo o código realizado, contudo nas seguintes figuras estará demonstrado o foi permitido partilhar.

- Travels Schema

```
2
3  const schema = new mongoose.Schema ({
4    checkIn : { type: Date, required: true},
5    checkOut : { type: Date, required: true},
6    country : { type: String, required: true},
7    countryCode : { type: String, required:true},
8    //travelImage : { type: String, required: true},
9    email : { type: String, ref: 'User', required:true},
10   feedback : { type: String, required:true}
11  });
12
13  export default mongoose.model('Travels', schema);|
```

- Pedido HTTP para a confirmação de conta é mostrado na seguinte figura.

```
router.post('/confirmation', (req, res) => {
  const token = req.body.token;
  User.findOneAndUpdate(
    { confirmationToken: token },
    { confirmationToken: "", confirmed: true},
    { new: true}
  ).then(user =>
    user ? res.json({ user: user.toAuthJSON() }) : res.status(400).json({})
  );
});
```

Prints relativos ao funcionamento da aplicação, são demonstrado nas seguintes figuras.

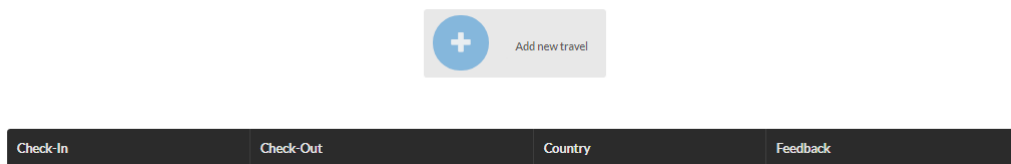
- Confirmação de Conta



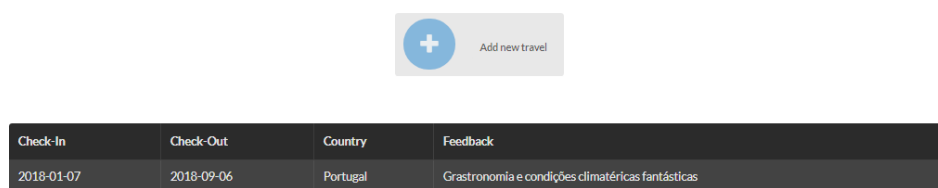
- Confirmação de Conta efetuada com sucesso



- *Dashboard* vazio



- *Dashboard*



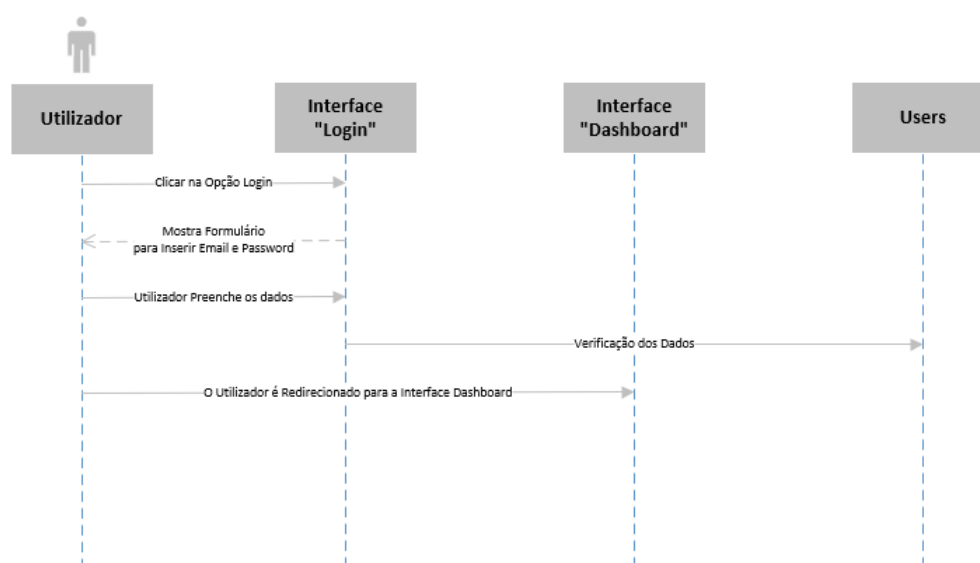
Diagramas Complementares

4.4.2. Descrição do Caso de Uso “Login”

Nome	Login
Descrição	Neste caso de uso o ator efetua o login na aplicação <i>Passport</i>

Pré-Condição	Estar registado
Caminho Principal	<ol style="list-style-type: none"> 1. O ator escolhe a opção “<i>Login</i>”. 2. O sistema mostra um formulário para inserir o email e password com o que ator se registou. 3. O ator preenche todos os campos do formulário (<i>Email, password</i>). 4. O sistema aceita o login e direciona o ator para o <i>Dashboard</i>
Caminho Alternativo	<ol style="list-style-type: none"> 3. O ator não preenche todos os campos. <ol style="list-style-type: none"> a. O ator introduz um email incorreto. b. O ator introduz a <i>password</i> incorreta.
Pós-Condição	--
Suplementos ou Adornos	Realizar um teste em que se verifique que o utilizador introduziu um email e/ou uma password incorreta.

4.5.1. Diagrama de Sequência do Caso de Uso “Login”

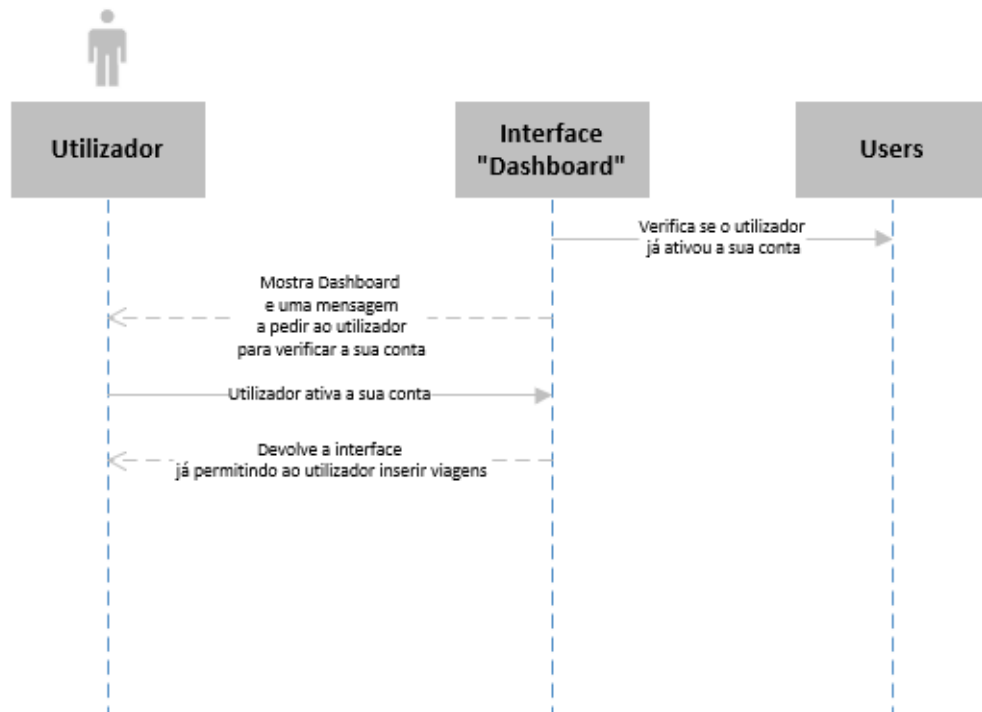


4.4.3. Descrição do Caso de Uso “Email de Confirmação”

Nome	Email de Confirmação
Descrição	Neste caso de uso o ator ativa a sua conta da aplicação <i>Passport</i>
Pré-Condição	O ator ter o login efetuado

Caminho Principal	<ol style="list-style-type: none"> 1. O Dashboard encontra-se bloqueado até à confirmação de conta do utilizador 2. O ator vai ao seu email e ativa a sua conta
Caminho Alternativo	2.1. O utilizador não ter recebido email de confirmação
Pós-Condição	--
Suplementos ou Adornos	Realizar um teste em que se verifique que o utilizador consegue ativar a sua conta através do link que recebeu no email.

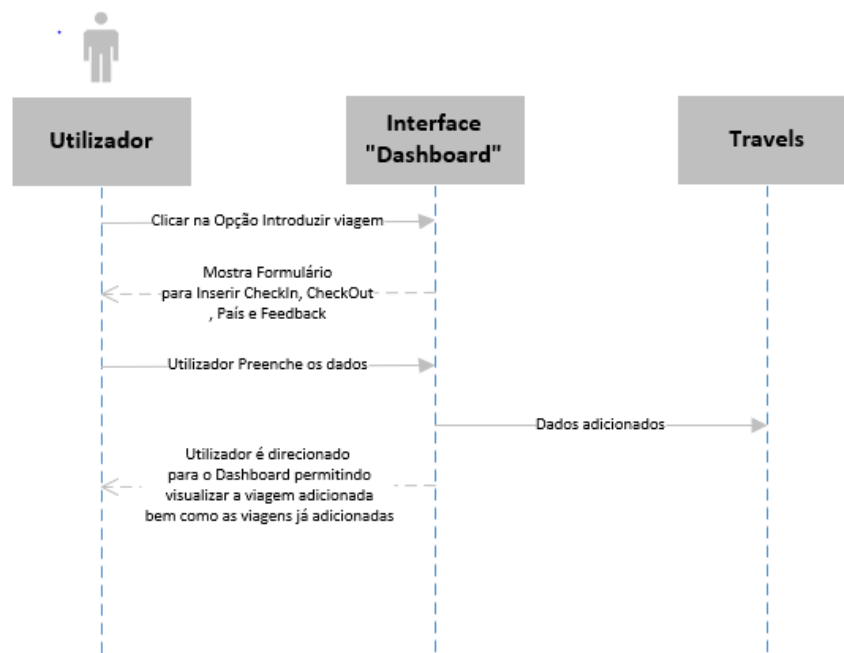
4.5.2. Diagrama de Sequência do Caso de Uso “Confirmação de Email”



4.4.4. Descrição do Caso de Uso “Inserir Países Visitados”

Nome	Inserir Países Visitados
Descrição	Neste caso de uso o ator inserir os países já visitados na aplicação <i>Passport</i>
Pré-Condição	O ator ter o login efetuado e a conta ativada
Caminho Principal	<ol style="list-style-type: none"> 1. O ator pressiona o botão de adicionar viagens na interface Dashboard. 2. O ator preenche todos os campos do formulário (checkIn, checkOut, país visitado e feedback) 3. O sistema aceita o registo e direciona o ator de volta para o Dashboard já com a viagem adicionada
Caminho Alternativo	<ol style="list-style-type: none"> 2. O ator não preenche todos os campos. 3. O sistema não aceita o registo da viagem.
Pós-Condição	--
Suplementos ou Adornos	Realizar um teste em que se verifique que o ator não introduziu todos os campos obrigatórios.

4.5.3. Diagrama de Sequência do Caso de Uso “Inserir Países Visitados”



4.4.5. Descrição do Caso de Uso “Consultar países visitados”

Nome	Consultar Países Visitados
Descrição	Neste caso de uso o ator consulta os países já visitados na aplicação <i>Passport</i>
Pré-Condição	O ator ter o login efetuado e a conta ativada
Caminho Principal	<ol style="list-style-type: none"> 4. O ator pressiona o botão “Dashboard”. 5. O sistema direciona o ator para o dashboard mostrando-lhe as viagens já realizadas bem como o botão para adicionar mais viagens
Caminho Alternativo	<ol style="list-style-type: none"> 4. O sistema não mostrar as viagens já realizadas pelo ator
Pós-Condição	--
Suplementos ou Adornos	Realizar um teste em que se verifique que o sistema consegue sempre mostrar as viagens já efetuadas mesmo quando essa quantidade é enorme.

4.5.4. Diagrama de Sequência do Caso de Uso “Consultar países visitados”

