PWA – Trabalho Prático

Programação para a Web Avançada

André Pinto (8200613) / SERGIO Félix (8200615)

2022

Índice

[Introdução 2](#_Toc93968007)

[Implementação 3](#_Toc93968008)

[Pressupostos 3](#_Toc93968009)

[Recursos 3](#_Toc93968010)

[API 3](#_Toc93968011)

[Estrutura 3](#_Toc93968012)

[Configs 3](#_Toc93968013)

[Rotas 4](#_Toc93968014)

[Validação e Sanitização 5](#_Toc93968015)

[Segurança 5](#_Toc93968016)

[Cálculo de Disponibilidade 6](#_Toc93968017)

[Forgot Password 7](#_Toc93968018)

[FrondEnd 8](#_Toc93968019)

[FrontOffice, 8](#_Toc93968020)

[BackOffice (/admin) 9](#_Toc93968021)

[Testes de Software 11](#_Toc93968022)

[StartRating 11](#_Toc93968023)

[HotelFormDrawer 11](#_Toc93968024)

[Melhorias 12](#_Toc93968025)

[Conclusão 13](#_Toc93968026)

[Bibliografia 14](#_Toc93968027)

# Introdução

No âmbito da disciplina de Segurança das Aplicações WEB do 2º ano, 1º semestre do Curso Técnico Superior Profissional em Desenvolvimento para a Web e Dispositivos Moveis, foi pedido o desenvolvimento de um sistema Web que comtemplasse todos os conteúdos teórico-práticos abordados nas aulas, NodeJS e React. Entre os temas apresentados, o selecionado pelo nosso grupo foi “Reservas de Hotel”.

O objetivo principal desta aplicação, será proporcionar ao cliente a criação de reservas no hotel, não esquecendo de toda a gestão por de trás necessária. Para tal está previsto a implementação de um web site, onde o cliente poderá visualizar todas as informações dos hotéis, bem como as ofertas de quartos disponíveis de cada hotel. Posteriormente poderá realizar a sua reserva e efetuar o pagamento, dependendo da disponibilidade do hotel.

Em relação à parte administrativa, a aplicação terá de complementar 3 tipo de utilizadores diferentes. “Administrador”, que tem permissão todas em todos os objetos; “Director”, apenas terá permissões para os hotéis com permissão e por fim “Employee”, com permissões apenas para as suas funções a exercer no hotel, tais como, criar reservas, visualizar disponibilidade, entre mais.

Nesta parte administrativa, será possível gerir todos os dados presente no sistema. Para cada objeto será possível realizar o seu respetivo CRUD de forma fácil e intuitiva, respeitando as permissões atribuídas.

Estre trabalho, será iniciado com a criação da API em NodeJs para possibilitar a interação e manipulação dos dados com a parte FrontEnd, posteriormente desenvolvida em React.

# Implementação

## Pressupostos

* Existem 4 tipos de níveis de acessos da aplicação. “Admin”; “Director”; “Employee” e “Client”.
* Permitir a gestão de vários hotéis.
* Cada hotel possui tipos de quartos de modo a agrupar quartos com características semelhantes exemplo: (“Quarto Duplo”, “Suite”)

## Recursos

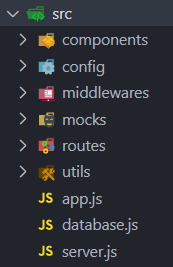
* Express
* ExpressValidator (Ferramenta para validação e sanitização de dados)
* Stripe – (API de pagamentos)
* Tailwind – (CSS Framework)
* Multer - (Upload Framework)
* Antd
* NodeMailer – (Envio de Emails)

## API

### Estrutura

A estrutura utilizada foi a idealizada pelo grupo de forma a ser organizada, compreensível e de boa prática optando assim por dividir por componentes, onde irão ser armazenados por nome contendo o seu modelo(schema), o seu controlador(controller) e o serviço, tornando assim fácil implementação de lógica nas rotas.

### Configs

As configurações estão todas armazenadas na pasta “config” onde são carregadas configurações de variáveis de ambiente (ficheiro. env) dependendo do ambiente de desenvolvimento (produção ou desenvolvimento) caso as mesmas não estejam disponíveis são atribuídos valores padrão. Com esta implementação torna a aplicação de fácil escalabilidade no que refere a configurações.

### Uma imagem com mesa Descrição gerada automaticamenteRotas

Na tabela acima podem ser visualizadas todas as rotas disponíveis na API bem como os devidos parâmetros requisitados pelas mesmas e as roles que contém autorização de acesso em cada rota.

Excluindo as rotas de autenticação (“Auth”) algumas rotas extra nos componentes Hotel, RoomType, e User a maioria são rotas referentes ao CRUD (Create, Read, Update, Delete) de cada componente.

### Validação e Sanitização

Uma imagem com texto

Descrição gerada automaticamenteInicialmente foi estipulado pelo grupo uma validação/sanitização de dados em todas as rotas utilizando a ferramenta ExpressValidator, porém foi apenas implementado em algumas rotas por escassez de tempo.

### Segurança

Uma imagem com texto

Descrição gerada automaticamenteNo desenvolvimento da API, foi necessário a implementação de uma função utilitária que permite em várias rotas verificar se o utilizador que está a tentar executar uma ação para um determinado componente, se o mesmo tem acesso a essa a efetuar a mesma.

### Cálculo de Disponibilidade

É uma funcionalidade essencial, pois é responsável por verificar a disponibilidade dos tipos de quarto para as datas pedidas pelo utilizador.

Esta tarefa foi dificultada porque pressupomos a existência de vários tipos de quarto e para cada tipo de quarto haverá quartos associados. Então o cálculo de disponibilidade é realizado da seguinte forma.

1. Match dos tipos de quarto que possam albergar o número de pessoas para a reserva.
2. Uma imagem com texto

   Descrição gerada automaticamentePara cada um procurar o número de reservas que coincidem com as datas inseridas pelo utilizador
3. Devolver os tipos de quartos que a contagem de reservas encontradas para as datas pedidas pelos utilizadores seja menor que a contagem dos quartos existentes daquele tipo de quarto.

Desta forma obtemos a listagem de todos os tipos de quarto disponíveis que respeitam os pedidos do cliente.

### Forgot Password

Tal como pedido no enunciado, implementamos uma funcionalidade para possibilitar ao cliente alterar a sua password com segurança. Realizamos os seguintes passos para implementação:

1. Verificar existência do email para alterar a senha
2. Criar um token com 15 min de validade com os seguintes dados. UserID, Email e um hash de validação. Este hash é gerado através dos últimos 7 dígitos do hash armazenado da password, para possibilitar a deteção da alteração da password, mesmo que o token ainda não esteja expirado.
3. Enviar um email para o utilizador com o link de recuperação

Após receber o email e introduzir a nova password será efetuado um pedido post para (auth/new-password), e passa pelas seguintes validações.

1. Validação do token
2. Verificar o hash de validação. (Caso seja validado a password ainda não foi alterada senão a password já foi alterada e já não é possível repetir)

## FrondEnd

Para o FrondEnd decidimos a realização de duas áreas diferenciadas. A primeira, seria um website dedicado aos utilizadores e um BackOffice, dedicado a todos os utilizadores com tarefas administrativas nos hotéis.

### FrontOffice,

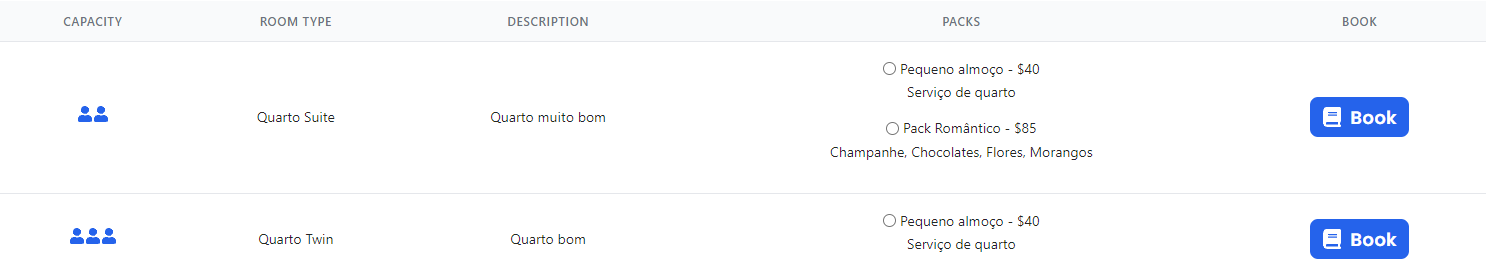
É aqui onde o utilizador comum vai interagir com a aplicação possibilitando-o escolher o hotel que pretende ficar hospedado num determinado tempo.

Uma imagem com texto, céu, exterior, símbolo

Descrição gerada automaticamente

Através da barra de pesquisa localizada na página inicial e na visualização detalhada de um específico hotel, o utilizador consegue definir os seus requisitos (data inicial da reserva, data final da reserva, número de adultos/crianças e hotel), esses mesmos requisitos que vão permitir a pesquisa na API para encontrar disponibilidade de quartos para o hotel especificado com os parâmetros definidos pelo utilizador.

Após isso, se existirem quartos disponíveis os mesmos serão mostrados bem como os packs associados ao tipo de quarto, permitindo assim ao utilizador se desejar concluir a reserva, caso tal se verifique torna assim menos um quarto disponível para as datas pretendidas por esse utilizador.



### BackOffice (/admin)

É responsável por toda a administração da aplicação. Esta, é uma área restrita e apenas é permitido o acesso a Administradores, Diretores e Colaboradores.

Uma imagem com texto

Descrição gerada automaticamenteVisto que a aplicação contempla a criação de vários hotéis, houve a necessidade de identificar o hotel em que o utilizador está a trabalhar. Para tal no canto superior direito é possível selecionar o hotel em que o utilizador tenha permissões. De forma a guardar o “estado” atual em caso de refresh da página, guardamos esses dados na local storage.

Todo BackOffice segue a mesma estrutura em praticamente todas as rotas. Neste, temos a possibilidade de realizar CRUD para Hoteis, RoomTypes, Packs, Rooms e Books. Inicialmente é mostrada uma listagem com a funcionalidade de paginação. Caso o utilizador queira criar um novo item, tem um botão “New” que abrirá um drawer com o formulário necessário.

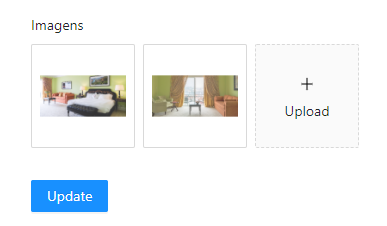
Este mesmo componente, também é reutilizado quando o utilizador pretender editar o objeto. Quando o componente abre em modo de edição, carrega automaticamente todos os dados do objeto para o formulário.

Uma imagem com texto, captura de ecrã, computador, interior

Descrição gerada automaticamente

Caso contrário o formulário será mostrado vazio, com as indicações necessárias para alertar o utilizador que está a realizar uma inserção.

Para a realização dos formulários, utilizamos a biblioteca Antd. Nela utilizamos vários componentes, tais como: Input; Rate; Select e Select múltiplo com pesquisa; TextArea; FormList; InputNumber; DatePicker; RangeDatePicker e Upload (Picture Wall).



# Testes de Software

Como elemento final foi proposto a realização de testes unitários. Para tal, realizamos teste a dois componentes da nossa aplicação. O primeiro foi ao componente StarRating presente na pasta (Frontal-Office/componentes/StarRating) e o segundo foi ao componente HotelFormDrawer (Back-Office/hotel/HotelsFormDrawer):

StartRating

Como o nome indica, este componente é responsável por receber o rating em formato de número e converter para um rating de estrelas. Os testes focaram-se essencialmente no envio de vários valores seguido da validação do seu respetivo output. Criamos testes para enviar a propriedade undifined, null, valores negativos em que o seu output teria de ser zero estrelas. Posteriormente realizamos testes enviando um valor decimal para no final verificara a contagem de estrelas completas e meias estrelas.

Neste componente os testes passaram com sucesso e não foram detetados bugs.

HotelFormDrawer

Este componente, comtempla o formulário para criação e atualização do objeto hotel. Sendo um formulário único, este componente funciona como um formulário de criação quando não é passado nenhum id de hotel, caso contrário funciona como um formulário de atualização. Os testes incidiram na verificação das mensagens apresentadas caso o componente abra em modo de edição ou em modo de inserção.

Nestes testes, foi encontrado um bug. Reparamos que enviávamos todos os dados do hotel como parâmetro, sabendo que posteriormente o componente iria buscar novamente os dados á API. Como as validações necessárias eram feitas aos dados recebidos por parâmetro, não garantia que esse componente existisse o que poderia levar o drawer a abrir em modo de edição e não carregar dados porque o objeto não existia na base de dados.

Apesar de ser um caso remoto, difícil de acontecer, realizamos todas as alterações para melhorar o bom funcionamento do mesmo.

# Melhorias

Nem todas as funcionalidades foram implementadas na realização deste projeto. Assim, foram transferidas para melhorias futuras para o projeto.

Stripe, é uma API de pagamentos. O idealizado seria dar a possibilidades de o utilizador concretizar o pagamento da altura da reserva. Esta funcionalidade foi implementada e testada num projeto separado, no entanto não faltou a integração com o projeto principal.

Redis, armazenamento de estruturas de dados em memória. Estava previsto a implementação de um sistema de cache na API para rotas que fossem bastante requisitadas e os seus dados não sofressem constantes alterações.

Preço do quarto variável. Inicialmente, o preço do quarto era calculado dependendo do mês e do dia da semana. Mas, com o decorrer do desenvolvimento, decidimos deixar para melhoria futura, pois não tivemos tempo para implementar.

# Conclusão

Após o tema proposto do desenvolvimento de um sistema Web em conformidade de um gerenciamento de reservas de hotel os elementos do grupo implementaram para além de funcionalidades básicas, funcionalidades avançadas bem como boas práticas de modo a efetivar o que foi aprendido nas aulas lecionadas para obter o melhor resultado final possível.

Graças á dedicação do grupo, foi possível cumprir a maioria dos objetivos idealizados para a realização deste projeto e solidificar assim ainda mais os nossos conhecimentos sobre a área de desenvolvimento Web, permitindo assim mais estabilidade aquando da entrada no mercado de trabalho.

Conforme o exposto, conclui-se que foi um projeto bem conseguido, que do qual estamos orgulhosos de todo o processo de aprendizagem e do resultado final.

# Bibliografia

* React JS - <https://beta.reactjs.org/>
* AntD - <https://ant.design/components/overview/>
* Tailwind - <https://tailwindui.com/documentation>
* Multer - <https://github.com/expressjs/multer>
* JS W3Scholls - <https://www.w3schools.com/js/>
* Stackoverflow - https://pt.stackoverflow.com/