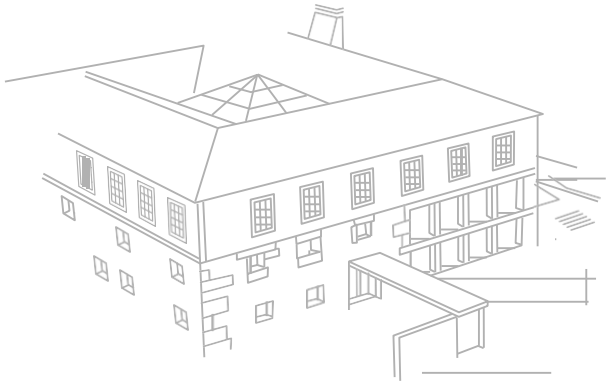
P.PORTO

ESCOLA
SUPERIOR
DE TECNOLOGIA
E GESTÃO



Programação Ambiente Web Restauração

Unidade Curricular: Programação Ambiente Web

Docentes: Fábio Silva, Edgar Esteves, João Ramos, Óscar Oliveira

Elemento: 8170164 – Filipe Oliveira

Licenciatura em Engenharia Informática 2ºAno

Ano letivo: 2019/2020

Felgueiras, 09 de Setembro de 2020

Índice

1. Introdução

O presente trabalho é sobre desenvolvimento de uma aplicação web, mais concretamente uma aplicação web para uma empresa no ramo da restauração que tem em mente a criação de uma plataforma de reserva de mesas para almoços e jantares de modo a organizar o funcionamento do restaurante e diminuir o tempo de espera dos clientes. Para isto irei usar diferentes tecnologias, sendo principalmente elas NodeJS, Express e Angular; com o objetivo de ter 2 partes distintas da minha aplicação sendo uma delas a REST API e outra a parte do cliente. No final espero obter uma plataforma onde utilizadores e admins possam fazer login e possam efetuar todas as ações a que estão autorizados.

2. Identificação e caracterização do projeto

Numa primeira fase realizei uma breve abordagem de todos os endpoints que pudessem ser necessários para a realização deste projeto, esse foi o meu ponto de partida a partir do qual desenvolvi todos os endpoints incluindo models para estar de acordo com o padrão MVC.
De seguida passei para a parte da autenticação, onde do lado do cliente após a autenticação é guardado tanto nas cookies como na localStorage do browser um token que permite à REST validade a sessão do utilizador, para isso usei uma ferramenta chamada JWT para a criação e validação dos tokens e o bcrypt para guardar as passwords dos utilizadores na base de dados de forma encriptada. Foram também implementadas as autorizações em cada endpoint. Com a primeira fase finalizada passei então à realização da aplicação em Angular, onde comecei por definir a estrutura básica de routing, e comecei a configurar os serviços que se iriam ligar à minha REST API. Passei então à construção do HTML e de todo o código em TypeScript que iria controlar as vistas em HTML mais uma vez segundo o padrão MVC.
Por fim realizei algum CSS com a ajuda da framework Bootstrap.
O projeto não foi colocado num repositório git pois tive dificuldades ao realizar push para o repositório.

3. Especificação de requisitos

Requisitos	Descrição	Realizado
CRUD do utilizador	Criar/Editar/Ver perfil de utilizador na plataforma	✔
CRUD das reservas	Criar/Editar/Ver/Remover reservas na Plataforma. No criar, é gerado um código aleatoriamente pelo mongoose, com a informação relevante por parte do user, com o estado da reserva (automaticamente o estado fica como pendente). Sendo que caso ele escolha uma ementa e um numero de pessoas o preço a pagar pelo utilizador leva um desconto de 5%	✔
CRUD das ementas	Criar/Editar/Ver/Remover ementas na plataforma.	✔
CRUD do admin	Criar/Ver/Remover admins na plataforma.	✔
Autenticação	É permitido registar um utilizador na plataforma, fazer login e logout do próprio, desde que este esteja logado.	✔

Autorização	É registada a autorização que cada tipo de utilizador (utilizador, admin) terá nas diversas funcionalidades da aplicaçao	✔
Agendamento automático	Para a visualização da agenda eu utilizei um modulo já existente que encontrei após varias pesquisas na internet (@syncfusion/ej2-angular-schedule) que é basicamente uma agenda de eventos. Sempre que eu vejo a agenda coloca la as reservas como eventos, sendo que verifica se já existe mais de 10 reservas para aquele exato momento. Caso exista a reserva não é adicionada ao evento e o seu estado é automaticamente alterado para cancelada, caso contrário é adicionado o evento à agenda e o seu estado é alterado para confirmada.	✔
Filtros de reservas	Listar reservas por username	✔
Editar password do utilizador	Será possível um utilizador logado alterar a sua password.	✔
Envio de email	Enviar email quando as reservas são confirmadas ou canceladas pelo admin	✘
Gráficos e estatísticas	Realização de gráficos e estatísticas para ver o desempenho da empresa	✘

4. Análise dos principais pontos do trabalho

Neste ponto do relatório iremos apenas abordar opções de desenvolvimento que nada têm a ver com as opções obrigatórias definidas no enunciado do projeto.

1. A coleção dos utilizadores não contém uma variável de permissões sendo que por omissão todos os utilizadores não têm qualquer permissão, foi criada uma coleção de admins, sendo que a única informação que têm é uma referencia para o id de utilizador, escolhi fazer deste modo para permitir uma fácil gestão destes cargos pois estará tudo junto, foi pensado desta maneira pois por exemplo no caso de querer listar todos os admins teríamos de iterar sobre todos os

utilizadores para obter os admins tendo assim um custo de processamento cada vez mais alto baseado no número de utilizadores registados.

2. Optei pelo utilizador nunca saber a password que tem no momento por medidas de segurança embora a tenha no seu token de sessão, deixei a password no token para reduzir processamento do servidor e também porque o token está encriptado e o utilizador dificilmente teria acesso a ele, fazendo assim que mesmo em caso de roubo de token seja muito difícil um utilizador ter a sua password revelada, com base nisto para mudar a password usei também a password contida no token.

3. Usei o username como identificador, no backend para efetuar verificações e etc e para fazer pedidos de alteração relacionados com o utilizador usamos o seu id na base de dados para impedir injeção maliciosa de alterações através do número de cartão de cidadão de algum utilizador. Na parte das reservas, o utilizador anota o seu username, alguma informação relevante, por exemplo data da reserva, número de pessoas, se quer ir jantar ou almoçar, a ementa que deseja . Visto que não tive tempo para colocar a hora de marcação optei pelo campo info da reserva ser onde os utilizadores decidem se querem almoçar ou jantar e caso a reserva seja confirmada o utilizador tem um intrevalo de tempo onde pode ir usufruir da reserva por exemplo, se o utilizador X tiver uma reserva confirmada para o jantar de dia 12 de Setembro de 2020 ele pode passar lá entre as 19h e as 23h que irá ser atendido, se for almoço pode passar lá entre as 11h e as 15h.

4. O admin da aplicação está registado com o username: “admin” e password: “admin”, por alguma razão se esta conta de admin for eliminada quando o servidor de express for iniciado a conta será automaticamente criada.

5. Com o intuito de poupar processamento da REST API, decidi fazer algum processamento de dados localmente no lado do browser, como por exemplo, filtragem de pedidos, visualização de um pedido em concreto etc.

6. Decidi colocar informações que apenas os admins têm acesso na homepage pois seria desnecessário criar um componente apenas para esse efeito.

7. Decidi colocar a alteração de password juntamente com os headers, para que fosse de fácil acesso em qualquer momento, e porque também é algo que poucas vezes será utilizado.

8. Optei por usar Boostrap em vez de usar os Angular Materials.

9. Quanto ao agendamento automático de testes, definimos que testes são agendados para o dia a seguir ao dia atual, mas caso o dia a seguir ainda não cumpra o requisito das 48h mínimas por cada teste, o teste será remarcado para 3 dias a seguir ao dia do 1º teste.

5. Conclusão e Trabalho Futuro

À medida que desenvolvi este projeto, pude aprofundar o meu conhecimento no desenvolvimento de aplicações web. Aprendi bastante sobre uso de frameworks de desenvolvimento web, nomeadamente o Express e o Angular. Aprendi que o padrão MVC é

muito importante no desenvolvimento de aplicações web e que ao aplicá-lo o projeto fica bastante mais robusto e simples de entender. Poss concluir que este projeto me fez aprender muito e que os conhecimentos adquiridos nesta unidade curricular serão imprescindíveis para o meu futuro.

6. Bibliografia

https://ng-bootstrap.github.io/#/home;
https://mongoosejs.com/;
https://angular.io/;
https://expressjs.com/;
https://www.npmjs.com/package/jsonwebtoken;
https://www.npmjs.com/package/bcrypt;
https://ej2.syncfusion.com/angular/documentation/schedule/getting-started/;