CES26

Apresentação Final do Projeto Exploratório

COMP 23

Professor

Edgar Yano

Equipe

Arthur José
Caio Costa
Rafael Frisch

Instituto Tecnológico de Aeronáutica - ITA

1 Introdução

Este projeto teve por finalidade construir um *chat app*, similar ao *WhatsApp*. As principais tecnologias utilizadas foram React.JS, Node.JS, Express, MongoDB e Mongoose.

Criaram-se dois repositórios: uma para o front-end e outro para o back-end.

2 Front-end

Através da utilização do framework do React, foram criados os componentes do chat, separados por classes. A estrutura da página foi feita através da linguagens de marcação HTML e a respectiva estilização foi feita através de CSS.

O app foi divido em três páginas: tela de cadastro (*sign up*), tela de login e tela de chat, como mostram as Figuras 1, 2 e 3.

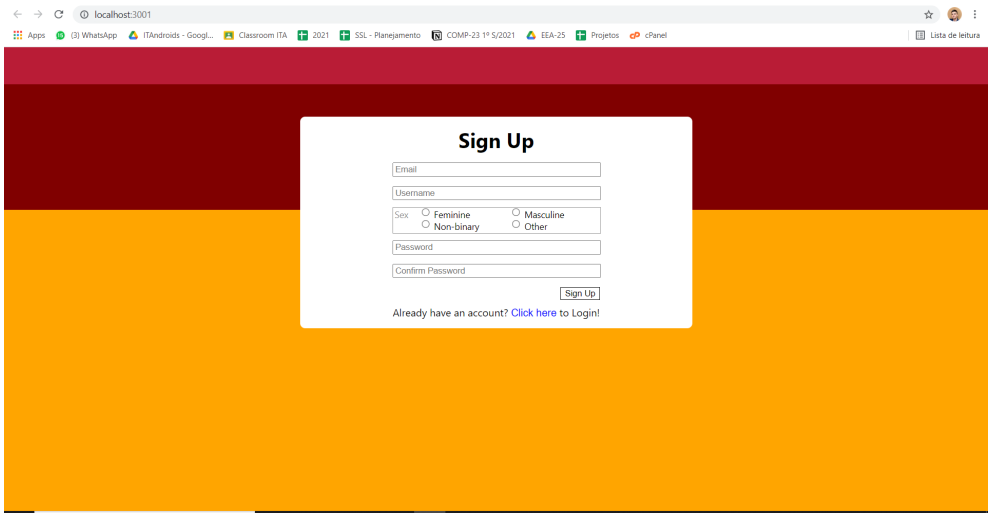

Figura 1: Tela de cadastro de usuário.

Os avatares, que simulam a imagem de perfil de usuário, cuja implementação será pensada para o próximo bimestre, foram obtidas através da DiceBear Avatars API.

Como dito anteriormente, o React permite que a aplicação seja dividida em componetes. Nesse sentido, as partes do chat foram esquematizadas desta forma, em que, por exemplo, o campo de mensagens é um componente e o card de usuário é outro componente. A divisão de componentes pode ser vista na figura 4.

3 Back-end

Atavés da tecnologia Node.JS, criou-se o back-end da aplicação, que é uma API que liga o front-end com o banco de dados, através dos métodos de requisição HTTP. Em relação à tecnologia de banco de dados, utilizou-se MongoDB (banco NoSQL), por intermédio do Mongoose, que é uma biblioteca que facilita o uso do MongoDB. O Workspace deste pode ser visto na Figura 5. O banco de dados está hospedado na cloud no MongoDB e a

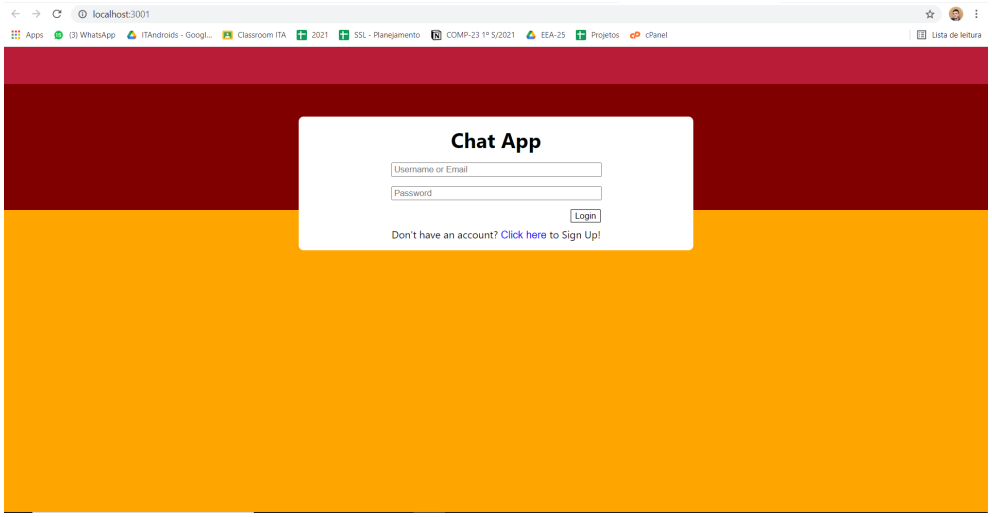

Figura 2: Tela de login de usuário.

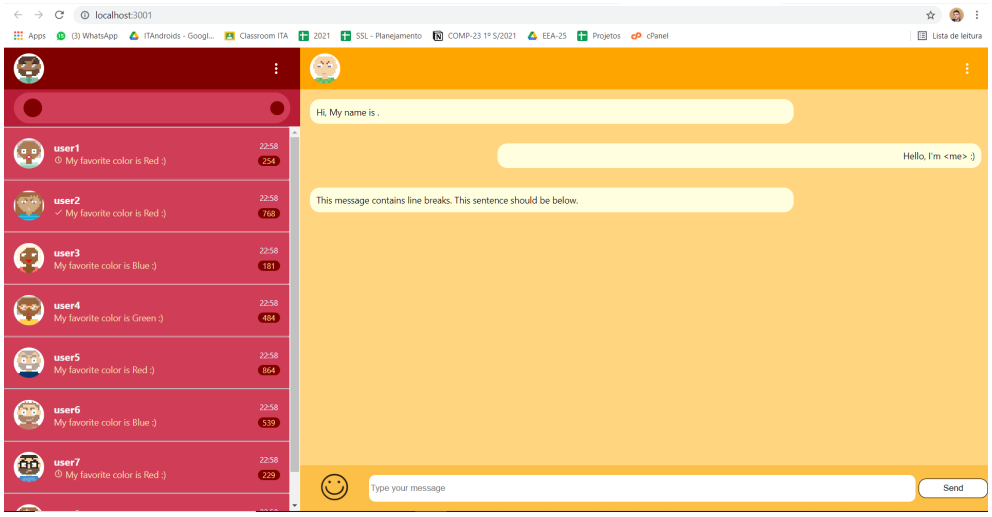

Figura 3: Tela de chat.

aplicação se conecta a esse banco por meio de uma connection string presente no arquivo .env.

Os métodos de requisição da API construída foram estruturadas também no Postman (API Client para tratar requisições), como mostra a Figura 6.

Para iniciar o chat, foi necessário integrar o front com o back. Isso foi possível configurando-se a portas do *server* do Node.JS de forma que o React app se "ouça" a respectiva porta. Além disso, as *routes* da API, os *middlewares* para autenticação de usuário, os *models* para o banco de dados e os *controllers* da conversação foram também estruturados no Node.JS, como pode ser visto na Figura 7.

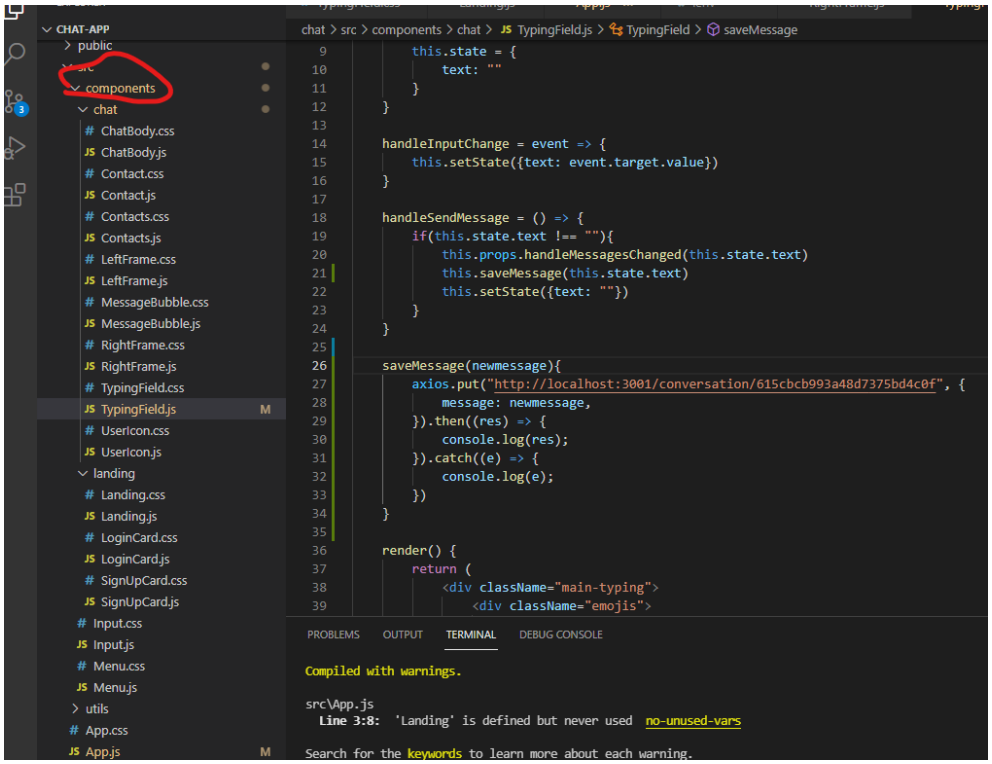

Figura 4: Divisão por componentes.

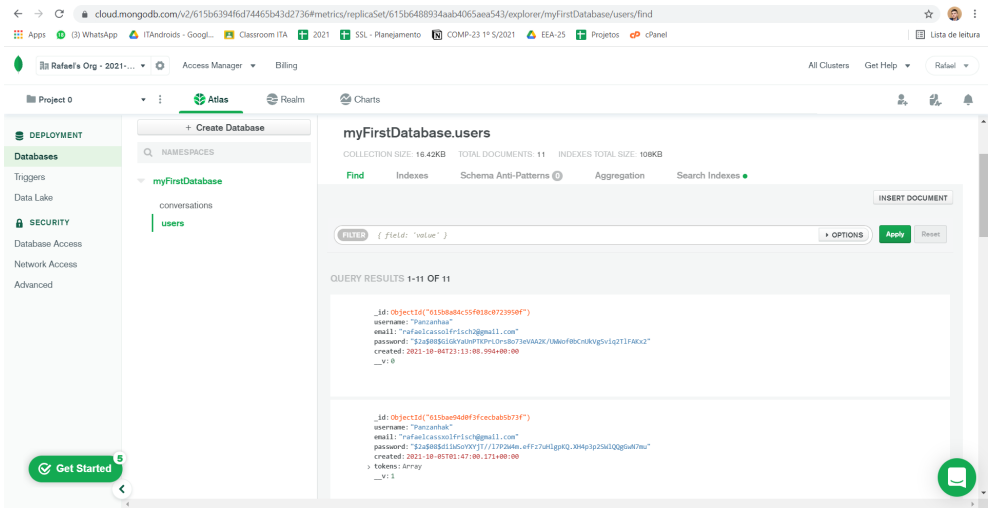

Figura 5: Workspace MongoDB.

4 Desafios

O principal desafio foi aprender diversas tecnologias em apenas um bimestre, de forma que o primeito contato com o desenvolvimento Web (para alguns membros do grupo) foi desafiador, ao mesmo tempo em que se abriu para os membros novos horizontes no desenvolvimento de software.

No front-end, o principal desafio foi atualizar os componentes (classes diferentes) a

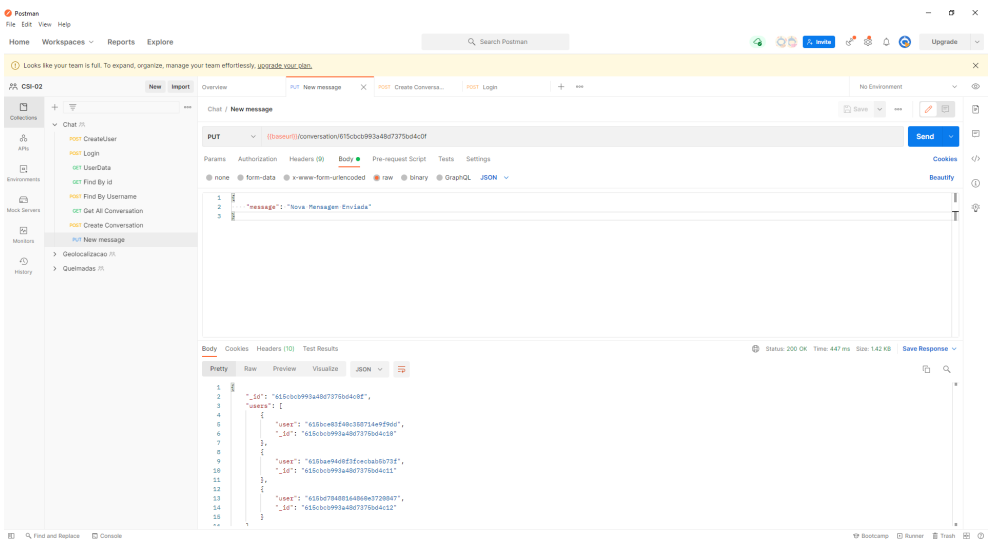

Figura 6: Postman - requisições HTTP.

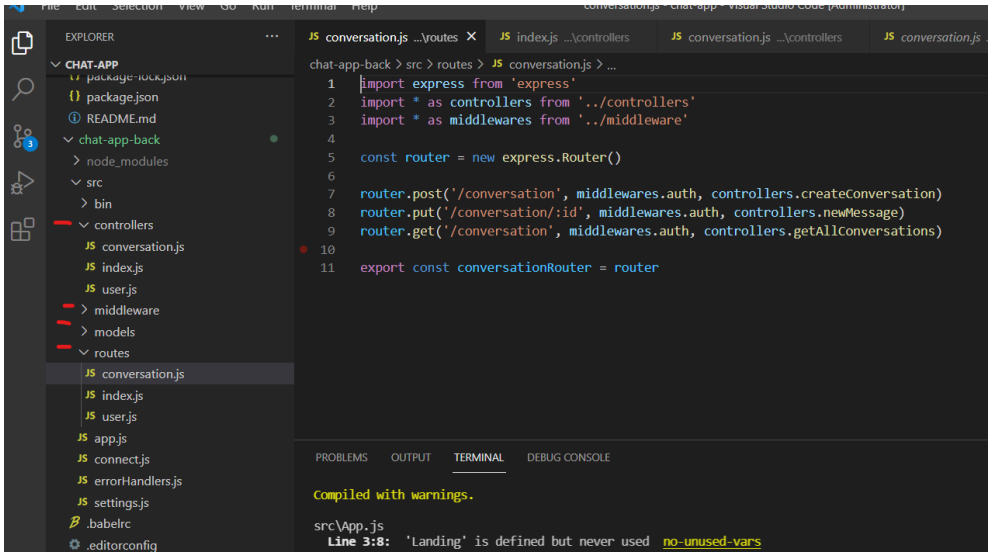

Figura 7: Estruturação do back-end.

partir dos eventos ocorridos em cada um. Por sorte, o React trata essa tarefa de forma simplificada, apesar não ter sido fácil descobrir o método adequado.

No back-end, o principal desafio foi o sistema de autenticação de usuário, visto que demanda certa complexidade. Nesse caso, foi utilizado um middleware de autenticação que é reutilizado em todas as rotas que requerem autenticação.