



Trabalho 1: WhyZap

Entrega



IMPORTANTE



As entregas serão feitas pelo moodle, seguindo os horários estabelecidos no sistema. O envio deverá ser feito enviando um arquivo compactado com todos arquivos utilizados no react na extensão zip com o nome do integrante do trabalho.


Exemplo: "Flavio.zip" (aqui vão todos arquivos do src, colocar o link do surge no readme)

Instruções Gerais

- Lembre-se de iniciar um projeto em React
- Você têm que criar componentes separados, da maneira que achar melhor.
- Vale a pena destacar que o tutorial e a documentação oficiais do ReactJS são muito boas; então, caso tenham alguma dificuldade muito grande, pode recorrer a este arquivo.

React - Uma biblioteca JavaScript para criar interfaces de usuário

React faz com que a criação de UIs interativas seja uma tarefa fácil. Crie views simples para cada estado na sua aplicação, e o React irá atualizar e renderizar de forma eficiente apenas os componentes necessários na

 <https://pt-br.reactjs.org/>



- Enviar **junto com o projeto o link do Surge!**

Escopo do Projeto

O cliente de hoje está tentando fazer uma utopia (você vai reparar que isto é muito comum no mercado): criar um aplicativo de troca de mensagens para bater de frente com os gigantes que existem atualmente, tais como WhatsApp, Telegram, Messenger e muitos outros.

Um projeto tão grande como este, normalmente, começa com a elaboração de um *MVP*. *MVP* significa "*Minimum Viable Product*", em tradução livre, "mínimo produto viável". Isto significa que começaremos implementando as *features* (funcionalidades) que indiquem o objetivo principal do produto do nosso cliente. Desta forma, não há a necessidade de se desenvolver um sistema que envie e receba mensagens de fora - criaremos algo que simplesmente mostra novas mensagens.

O escopo deste *MVP* é:

1. Lista de mensagem:

- Cada mensagem deve possuir um remetente (ou seja, um usuário que enviou) e o conteúdo em si.
- No layout, eles devem ficar assim:

nome do remete: conteúdo

2. Envio de mensagem:

- Abaixo da lista citada, deve existir um local onde o usuário escolhe o nome do remetente (input) ; o conteúdo da mensagem (input) e um botão de enviar;
- No layout, eles devem ficar um ao lado do outro. Sendo o campo do remetente com largura muito menor do que o campo do conteúdo;
- Ao enviar a mensagem, os campos devem ser resetados para ficar em branco novamente

Exemplo do MVP:

Flavio: MTSDAOmdf

eu: ifdis

eu

Mensagem

Enviar

▼ 🏆 Algumas funcionalidades extras

Faça na ordem:

1. Faça com que tanto o botão de enviar, como o botão "enter" envie as mensagens
2. Implemente a funcionalidade de deletar mensagem. Sempre que o usuário clicar duas vezes em cima de uma mensagem ela deve ser apagada.

▼ 💡 Dica

- Crie um próprio componente para mostrar a mensagem e o botão de deletar

3. Agora, vamos melhorar o design do nosso projeto. Queremos deixar mais evidente qual usuário está mandando a mensagem. Para isso, pensamos em mudar um pouco a disposição das mensagens. Sempre que o nome do usuário for "eu", a mensagem deve aparecer à direita e sem mostrar o nome dele; se for outro nome, deve aparecer à esquerda, e deve mostrar o nome do remetente. (Obs.: é uma funcionalidade bem parecida com os grupos do Whatsapp). Tentem seguir o layout do exemplo como guia.

Exemplo das funcionalidades 2 e 3:

