HANDS ON WEEK 4

A rede social Parrot é um sistema white label (ou seja, um sistema modelo criado por uma empresa que pode ser reutilizado por outras, apenas modificando informações como logo e marca) do qual condomínios podem contratar para incentivar a interação entre os moradores.
A plataforma permite que os usuários façam publicações que ficam visíveis para toda comunidade.

O desafio será desenvolvido por times de 3-4 pessoas, incluindo front-end e back-end. Lembre-se que comunicação é tudo em um projeto, estejam sempre alinhados sobre as demandas e prazos.

FRONT-END

Os desenvolvedores do front-end serão responsáveis por criar as páginas com base no layout utilizando Bootstrap, Styled-components, React e Redux.

1. Login
    a. Dados: email e senha
2. Cadastro do Usuário
    a. Dados: Nome, email, senha, unidade/apartamento
3. Feed (criar post, listas todos os posts)
4. Página do usuario (posts do usuário)

É muito importante que vocês mantenham a comunicação com os devs do back-end para garantir que vão receber as informações necessárias para tornar as páginas dinâmicas e funcionais.

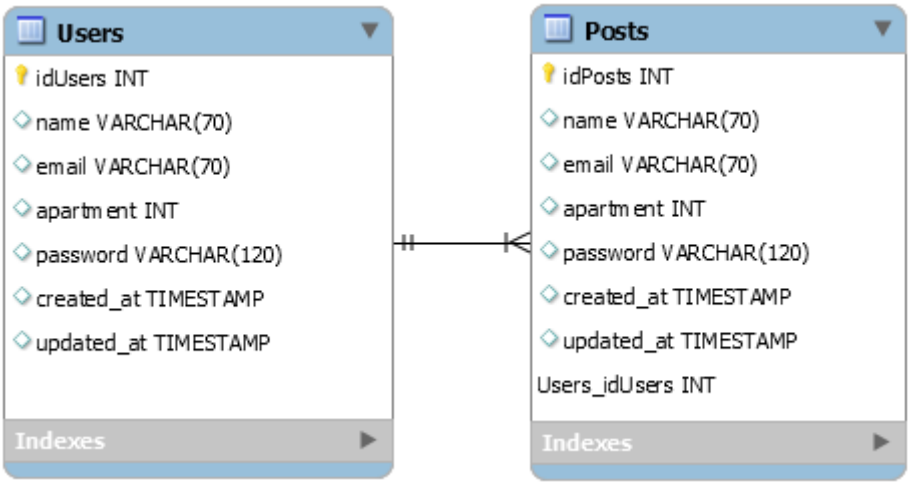
BACK-END

Os desenvolvedores do back-end serão responsáveis por construir a API utilizando Node.js, Express, Jest, Sequelize, MySQL e Arquitetura Limpa.

1. Criação de banco de dados com base no DER (vide última página)
    a. via migrations
    b. utilizar seeders para popular o banco caso necessário
2. API Rest com CRUD para usuários e publicações
3. Autenticação de usuário
4. Implementar Princípios SOLID na arquitetura
5. Testes automatizados (unitários e integração), cobrir ao menos:
    a. controllers
    b. endpoint (casos de erro também)

Garanta que a API está de acordo com os dados esperados na aplicação front-end.

Modelagem de dados:



FUNCIONALIDADES OBRIGATÓRIAS

1. Cadastro e autenticação do usuário
2. Autenticação do administrador
3. Perfil do usuário com lista de posts
4. Criar Post
5. Listar posts (Feed)
6. Edição do Usuário

FUNCIONALIDADE OPCIONAL

1. Like
2. Comentário em um post
3. Utilização de Loading e Página de erros.

CRITÉRIOS DE AVALIAÇÃO

- Responsividade da aplicação
- Validação dos dados de response da API (retorno)
- Validação dos dados de request da API (enviados para a API)
- Feedback de erro na API (uso de código http condizente com o tipo de erro)
- Uso de testes automatizados
- Uso de migrations
- Migrations devem refletir as configurações propostas no documento DER
- Estrutura do código.
- Validação de formulário.
- Validação de erros na requisição.
- Feedback de erros para os usuários
- Divisão de tarefas entre os membros da equipe seguindo os princípios da Metodologia Ágil Scrum
- Utilizar boas práticas de versionamento de código com Git

Entrega:

Deverá ser enviado o link do Github pela plataforma da Gama Academy até à data informada no slack no envio deste documento.