Teste para desenvolvedor fullstack/fullcycle

O teste é composto de três partes. São elas:

1. Questões sobre código e boas práticas
2. Questões sobre infraestrutura/devops
3. Teste prático

Regras:

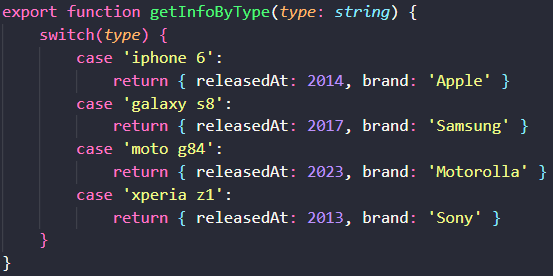
1. Encaminhar as respostas das três primeiras partes do teste via e-mail para giovana@startrh.io, em formato PDF. Para as questões que demandam exemplificações, favor anexar os arquivos de exemplo e/ou adicionar screenshots ao PDF.
2. O teste prático deve ser publicado no github ou similar
3. O repositório deve ser compartilhado com os contatos citados anteriormente.
4. Não utilize chatgpt.

# Parte 1 - Questões sobre código e boas práticas

1. No evento de um desenvolvedor com menos experiência lhe perguntar: “Qual é a melhor maneira de criar um componente no ReactJs? Utilizando funções ou classes”, qual seria a sua resposta e por quê?
2. Explique o benefício de utilizar o useMemo dentro de um componente.
3. De que maneira podemos melhorar o código abaixo?



1. Mostre como fazer uma requisição http no react e explique o benefício de fazer da forma escolhida.
2. Quais pacoted do npm lhe parecem mais apropriados para a criação de uma REST API?
3. Suponha que tenhamos um gargalo por conta de concorrência de uso. Quais atitudes poderíamos tomar para acabar ou amenizar com o gargalo? Exemplifique.
4. Mostre como criar uma rota POST e como criar uma rota GET, onde a roda get deve receber um parâmetro (route param).
5. Mostre como se conectar à um banco de dados utilizando nodejs
6. O que é um ORM?
7. Você possui experiência com microsserviços? Se sim, qual foi a estratégia adotada para comunicação?
8. Cite uma ferramenta de observability recomendada por você e explique o benefício de utilizá-la.
9. Demonstre a aplicação de polimorfismo utilizando nodejs
10. Demonstre a aplicação de um singleton utilizando nodejs
11. Como podemos melhorar o código abaixo?



1. Crie um teste unitário do código acima.

# Parte 2 – Infraestrutura/devops

1. Você tem familiaridade com algum tipo de pipeline?
2. Qual tipo de webserver é recomendado por você?
3. Qual é a sua estratégia para tratamento de chamadas HTTP/HTTPS utilizando o proxy reverso?
4. Mostre como atualizar o repositório de aplicativos local do linux
5. Mostre como criar um link simbólico
6. Num cenário hipotético, gostaríamos de configurar um processo de backups automáticos do banco de dados, todas as noites às 23:30. Qual estratégia você adotaria e por quê?
7. Exemplifique como um aplicativo escrito em nodejs pode ser migrado para a cloud (AWS/Azure/similar)
8. Agora, exemplifique como criar um aplicativo serverless simples na cloud.
9. Para qual tipo de aplicação você optaria por um banco não-relacional e para qual tipo você optaria por um banco relacional?
10. Demonstre uma forma de enviar o código da máquina local para um servidor e rodá-lo através de uma porta. Faça utilizando uma máquina virtual com linux instalado.

# Parte 3 – Teste prático

Construa, da maneira que lhe for mais agradável, um aplicativo MVC que siga as regras abaixo:

1. É necessário ter um frontend
2. É permitido o uso de bibliotecas de CSS
3. É necessário ter um backend
4. É permitido o uso de frameworks
5. É necessário possuir integração com um banco de dados
6. Caso o seu banco não esteja disponível online, por favor nos envie as variáveis de ambiente e o tipo de banco de dados escolhido para que possamos testar localmente
7. Tema: Locadora

Crie um aplicativo que liste os filmes disponíveis em seu banco de dados e permita que um filme possa ser alugado por X dias. Além disso, deve haver uma tela administrativa que liste os filmes alugados e alguma forma de exportar, via csv, um relatório que mostre todos os filmes alugados e todos os filmes disponíveis. Não se esqueça de que cada filme deve ter uma quantidade e um filme já alugado não deve estar disponível para aluguel. Por fim, o projeto deverá conter uma tela de login funcional (somente login, sem cadastro ou recuperação de senha).

Boa sorte e ótimo teste.