



**Desafio Final**

# Chegamos ao Desafio Final após 3 semanas intensas!

Leia atentamente todo este documento e prepare-se para  
o Desafio :)

## Obrigatório

- Utilizar o Projeto Base;
- Utilizar as tecnologias repassadas durante as 3 semanas;
- Só será aceito *push* (entrega do projeto) até 23:59 do dia 31/10/2020

# Informações Úteis

- No dia 31/10/2020, às 18:30h, (horário de Brasília) será liberado acesso ao projeto Base para realização do *fork* no GIT.
- Todo o projeto deverá ser individual e passará por uma análise automatizada a fim de buscar código compartilhado entre os maratonistas.
- Os testes unitários do front-end deverão ser feitos no diretório "desafioFinal/desafiofront/spec" conforme exemplo existente no projeto Base.
- O Back-end necessita da JDK 11.

# Como Iniciar o Desafio às 18:30h

## Clonando o projeto:

1. Logar no GIT Stefanini;
2. Fazer *fork* no projeto base:  
<https://gitmaratonadev.stefanini.com.br/git/maratonadev/DesafioFinal>
3. Verificar o *fork* do projeto que foi feito no seu perfil do GIT utilizando a seguinte estrutura de URL:
  - a. <https://gitmaratonadev.stefanini.com.br/git/SEUUSUARIO/DesafioFinal>
  - b. Substituir “SEUUSUARIO” pelo seu usuário do GIT para verificar se o fork funcionou.
4. Clonar o projeto usando a URL do seu perfil.

Importante: Não façam clone diretamente da URL <https://gitmaratonadev.stefanini.com.br/git/maratonadev/DesafioFinal>

## Iniciando o projeto:

1. Após clonar o projeto acesse o diretório "desafioFinal/desafiofront" e execute os comandos abaixo:
  - a. `npm install` (use o comando para instalar as dependências. Necessário rodar apenas uma vez)
  - b. `npm start` (para executar o projeto localmente)
  - c. `npm run test` (para executar teste de unidade)
2. Após executar os comandos acima, o front-end ficará disponível no endereço <http://localhost:3000>
3. Acesse o diretório "desafioFinal/desafioapi" e rode os comandos abaixo:
  - a. `mvnw quarkus:dev` (para executar o projeto localmente)
  - b. `mvnw clean test` (para rodar teste de unidade)
4. Após executar os comandos do passo anterior, o back-end ficará disponível no endereço <http://localhost:8080/api> e utilize o *swagger* para ver todas as APIs no endereço <http://localhost:8080/swagger-ui/>

# O Desafio

Uma locadora de automóveis deseja continuar o desenvolvimento de um sistema de locação de veículos. A função “listar carros” já está pronta e você foi contratado para desenvolver as demais funções. Ficando a sua escolha o design das telas:

## **Para Cliente:**

- Cadastrar Cliente
- Listar Cliente

## **Para aluguéis de carro:**

A locadora só permite que um cliente alugue um carro por vez e um mesmo carro só poderá ser alugado para outro cliente no momento que ele encontrar-se disponível novamente.

Com base nesta regra, desenvolva a funcionalidade de aluguéis de carros. O relacionamento entre carro e cliente ainda não foi feito e ficará a seu cargo escolher a melhor solução.

# Regras do Sistema

## Regras para cadastro do Cliente [Front-end]

### **Campo Nome:**

- Campo obrigatório;
- Mínimo de 3 caracteres;
- Máximo de 100 caracteres;
- Permite apenas letras e espaços.

### **Campo CPF:**

- Campo obrigatório;
- Permite apenas números.

### **Campo Endereço (CEP, Logradouro, Complemento, bairro, cidade e UF):**

- Obrigatório preenchimento de CEP, bairro, cidade e uf;
- Deve estar integrado com um serviço de consulta de CEP (Sugestão: <https://viacep.com.br>);
- O usuário pode alterar os dados que retornaram do serviço de consulta de CEP.

### **Campo E-mail:**

- Deve ser um e-mail válido.

### **Campo Contato:**

- Deve ser um telefone residencial ou celular válido.

## Regras para cadastro do Cliente [Back-end]

Para cada cliente, aplicar as seguintes regras:

- Não permitir dois clientes com mesmo CPF;
- O nome não deve ter mais que 100 caracteres ou menos que 3.

## Regras para aluguéis de carros [Front-end]

- Não permitir aluguel sem cliente.
- Exibir somente carros disponíveis para aluguel.





## Regras para aluguéis de carros [Back-end]

- Não permitir que um cliente alugue mais de um carro.
- Não permitir que o mesmo carro seja alugado por mais de um cliente simultaneamente.



# Desafios bônus de implementação

Se você conseguiu finalizar tudo que a Locadora solicitou, aproveite seu tempo livre e aprimore o sistema conforme abaixo:

	Teste de unidade no front-end e back-end com 100% de cobertura.
	Evoluir a funcionalidade de “listar carros” para diferenciar carros disponíveis dos carros alugados e exibir o atual locatário.
	Evoluir a funcionalidade de “listar clientes” para indicar se ele tem um carro alugado e qual modelo.
	Criar a funcionalidade “exibir histórico de cliente” e apresentar o histórico de todos os carros que o cliente já alugou.

# Sobre o resultado da Maratona

O resultado final será informado por e-mail até  
13/11/2020.

# Boa sorte!