



Requisitos para os testes funcionais

Os testes funcionais devem ser rodados fora do docker, isso porque é muito complicado ficar alterando horário do container (já que o comportamento padrão não é pra ser assim). Então apenas faça o build do container de dados e rode o projeto manualmente.

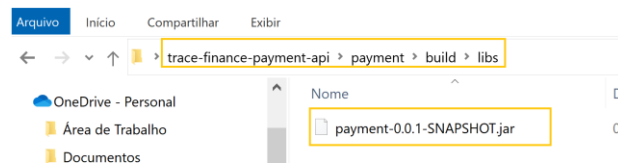
Para rodar o projeto manualmente TENHA CERTEZA DE TER O JAVA NA SUA MÁQUINA, NA VERSÃO 17 OU SUPERIOR!!!!!!

O primeiro passo é garantir que o container do banco de dados esteja funcionando, para fazer isso digite no terminal dentro da raiz do projeto o seguinte comando **“docker-compose up db -d”**:

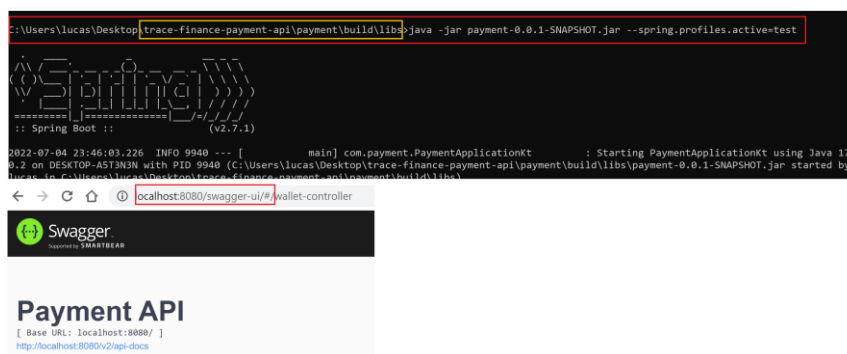
```
Prompt de Comando
C:\Users\lucas\Desktop\trace-finance-payment-api>docker-compose up db -d
[+] Running 1/1
  Container mysql_container Started
C:\Users\lucas\Desktop\trace-finance-payment-api>docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS                    NAMES
8b5b1ed8d334   mysql:8.0   "docker-entrypoint.s..." 7 seconds ago  Up 6 seconds  0.0.0.0:3306->3306/tcp, 33060/tcp  mysql_container
```

Agora faça o package da aplicação, entre no diretório payment e digite **“gradlew clean bootjar”** (se estiver em linux use **“./gradlew clean bootjar”**). Esse comando vai gerar um **.jar dentro de build/libs**:

```
Prompt de Comando
C:\Users\lucas\Desktop\trace-finance-payment-api>cd payment
C:\Users\lucas\Desktop\trace-finance-payment-api\payment>gradlew clean bootjar
BUILD SUCCESSFUL in 3s
5 actionable tasks: 5 executed
```



Agora basta executar esse jar usando o perfil de test, pra fazer navegue até o diretório do jar e digite: **“java -jar payment-0.0.1-SNAPSHOT.jar --spring.profiles.active=test”**:



Depois que a aplicação subir basta acessar **http://localhost:8080/swagger-ui/#/**, se a página do swagger estiver rodando tudo está ok.

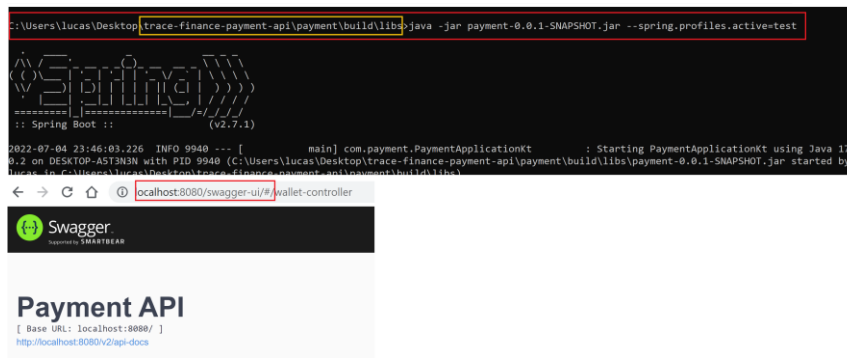
PARA TESTAR AS REGRAS DE PERIODO!!!

A aplicação detecta o período de acordo com o horário atual da máquina que está rodando-a. Então quando for testar os limites de horário lembre-se de mudar a hora da sua máquina E **REINICIAR O JAR DA API**.

Para reiniciar o .jar é simples, apenas digite “ctrl+c” no terminal que estiver rodando a aplicação, isso vai interrompê-la. Depois digite **“java -jar payment-0.0.1-SNAPSHOT.jar --spring.profiles.active=test”** novamente para iniciá-la:

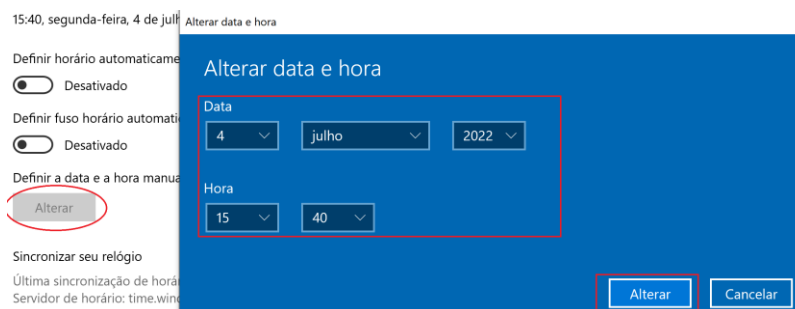
Um fluxo de teste seria mais ou menos assim:

-> **suba a aplicação:**

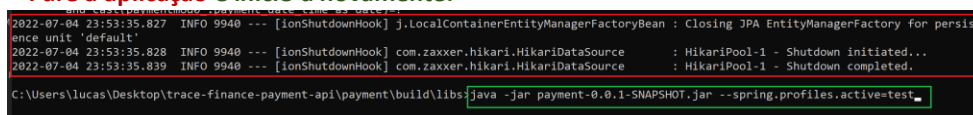


-> **teste os endpoints que precisa:** provavelmente aqui você vai criar uma carteira, fazer pagamentos até atingir o limite do período vigente, olhar o limite.

-> **mude o horário da sua máquina, para alterar os períodos de pagamento você precisa alterar a hora do seu computador, já que a aplicação usa o horário da máquina que está rodando para definir o período:**



-> **Pare a aplicação e inicie-a novamente:**

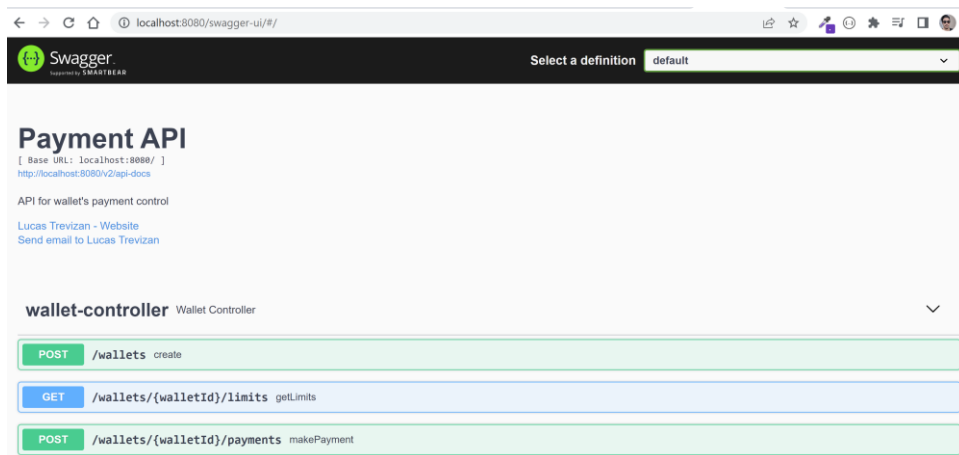


-> **teste o restante dos endpoints que precisa:** provavelmente testar o limite em outro período.

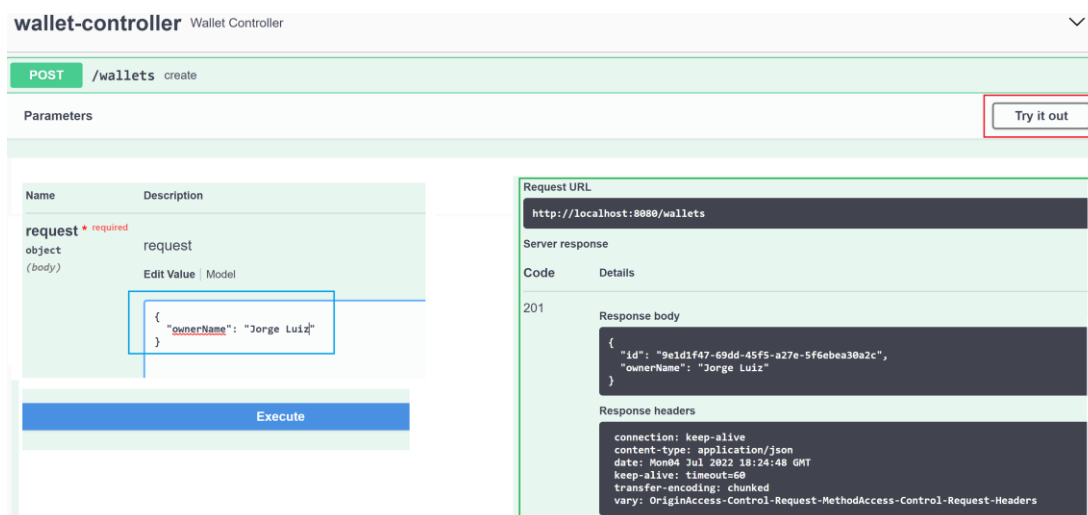
-> **teste do reset de limite:** mude o horário para 23:59, pare a aplicação e suba-a novamente, espere dar 00:00 e faça um request no limite de uma carteira (a mesma que você usou anteriormente)

Vamos aos testes com swagger

Para utilizar o swagger basta acessar `localhost:8080/swagger-ui/#/`: Depois disso clique em “wallet-controller”, vão aparecer os endpoints da aplicação :



Clique no endpoint que deseja testar e depois em **“try out”**, preencha os dados da requisição e depois clique em **“execute”**. Verifique a **resposta da aplicação** logo abaixo do botão de execute:



Você também pode usar o postman ou qualquer outro testador de request se preferir, nessa pasta também tem alguns arquivos úteis como sql's para consultas e a coleção do postman com os endpoints.