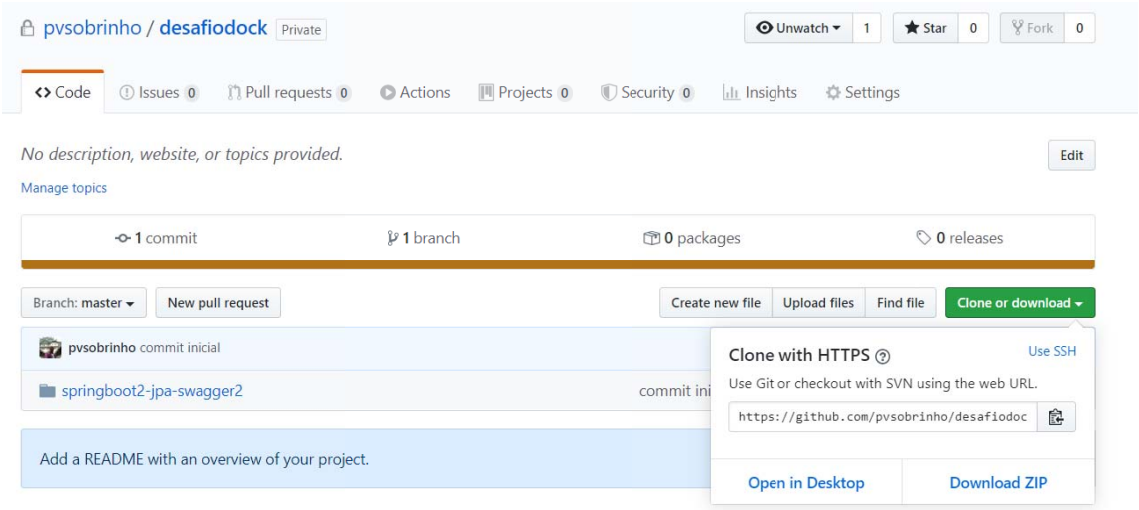
Conteúdo deste Manual

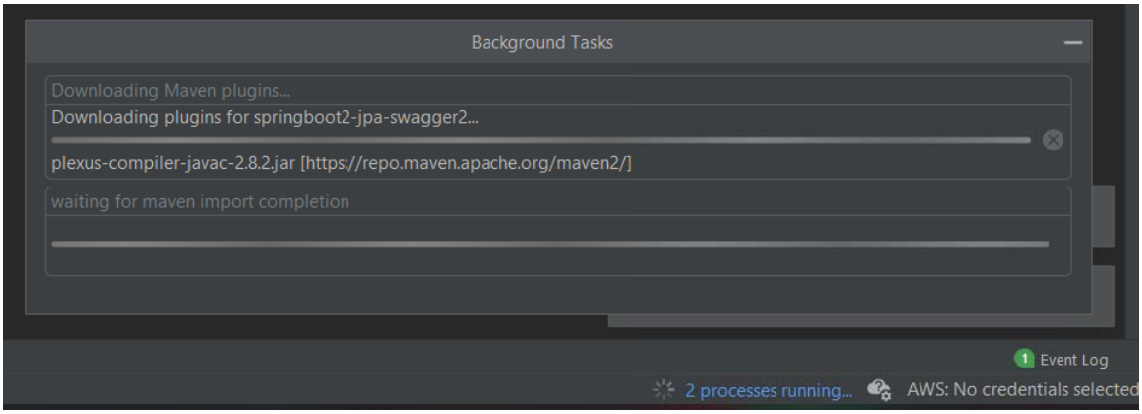
1- Baixar o projeto

Realizar o Clone do Projeto no GitHub em: https://github.com/pvsobrinho/desafiodock.git

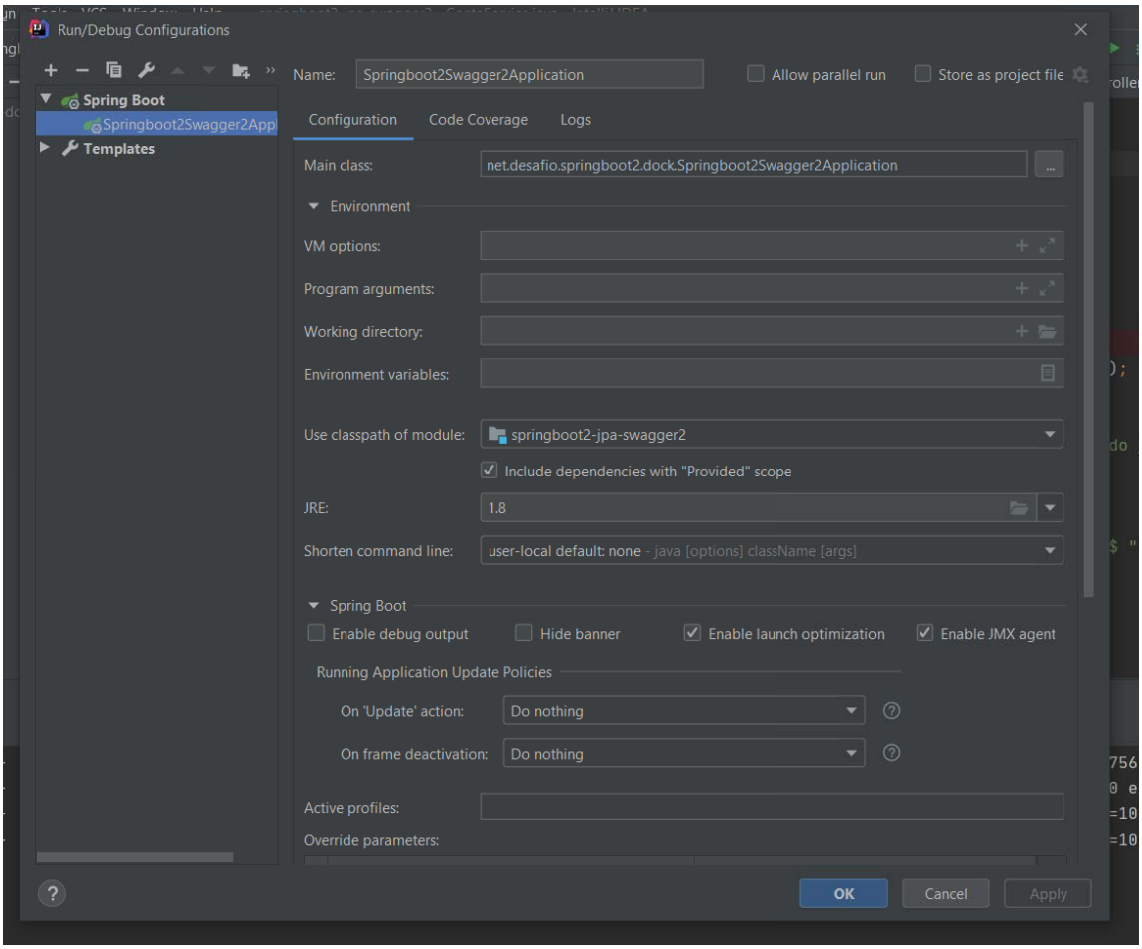


2- Configuração na IDE e JDK

Abrir o projeto preferencialmente no Intelliji a IDE para automatizar operações de downloads de dependências e build e configuração automática do Firewall do seu sistema operacional. Pode utilizar outra IDE de sua preferência caso deseje.

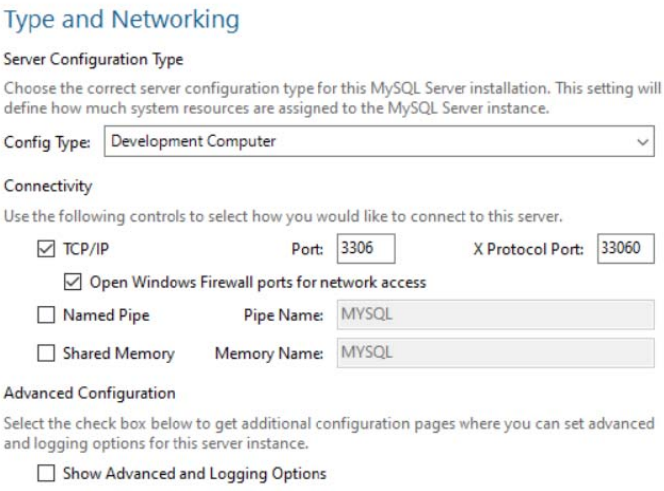


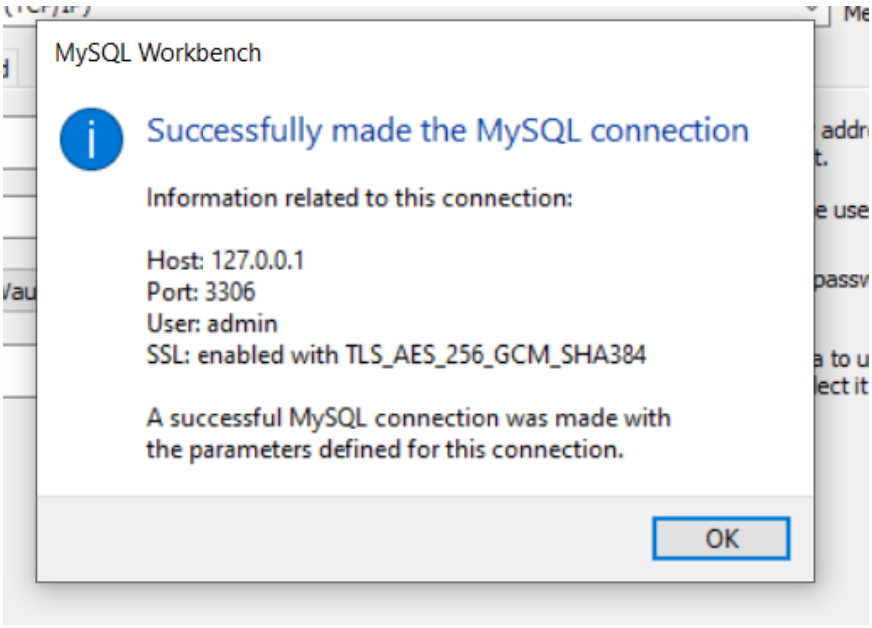
Foi utilizado esta IDE por conta da facilidade de configuração e algumas ações que são feitas automaticamente. Porem é necessário ficar atento a versão do JDK compatível com o projeto. JDK 8.



3- Configurando o Banco de Dados

Configuração do modelo para criação da base de dados:





Na aplicação procurar o arquivo application.properties e mudar a url de conexão e as credenciais para acessar o banco de dados. Para este exemplo foi utilizado a base de dados com nome ***schema = dock.***

```
## Spring DATASOURCE (DataSourceAutoConfiguration & DataSourceProperties)
spring.datasource.url = jdbc:mysql://localhost:3306/dock?useSSL=false
## 127.0.0.1 = localhost
spring.datasource.username = admin
spring.datasource.password = admin


spring.datasource.testWhileIdle = true
spring.datasource.validationQuery = SELECT 1
spring.jpa.show-sql = true


## Hibernate Properties
# The SQL dialect makes Hibernate generate better SQL for the chosen database
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect

# Hibernate ddl auto (create, create-drop, validate, update)
spring.jpa.hibernate.ddl-auto = update

logging.level.root = DEBUG
```

4. Script de Banco de Dados

SCRIPT DE CRIACAO DO BANCO DE DADOS

//CREATE SCHEMA

```
CREATE SCHEMA `dock` ;
```

SCRIPT DE CRIACAO DA TABELA CONTA

O SPRINGBOOT CRIA AUTOMATICAMENTE A TABELA CASO ELA NÃO EXISTA NO BANCO DE DADOS, MAS CASO QUEIRA CRIAR MANUALMENTE SEGUE O SCRIPT

//CONTA

```
SELECT * FROM dock.conta;CREATE TABLE `conta` (
  `id_conta` bigint NOT NULL,
  `data_criacao` datetime NOT NULL,
  `flag_ativo` bigint NOT NULL,
  `id_pessoa` bigint NOT NULL,
  `limite_saque_diario` decimal(19,2) NOT NULL,
  `saldo` decimal(19,2) NOT NULL,
  `tipo_conta` bigint NOT NULL,
  PRIMARY KEY (`id_conta`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

// PESSOA

```
CREATE TABLE `pessoa` (
  `id_pessoa` bigint NOT NULL,
  `cpf` varchar(255) NOT NULL,
  `data_nascimento` datetime NOT NULL,
  `nome` varchar(255) NOT NULL,
```

```
  PRIMARY KEY (`id_pessoa`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

//TRANSACAO

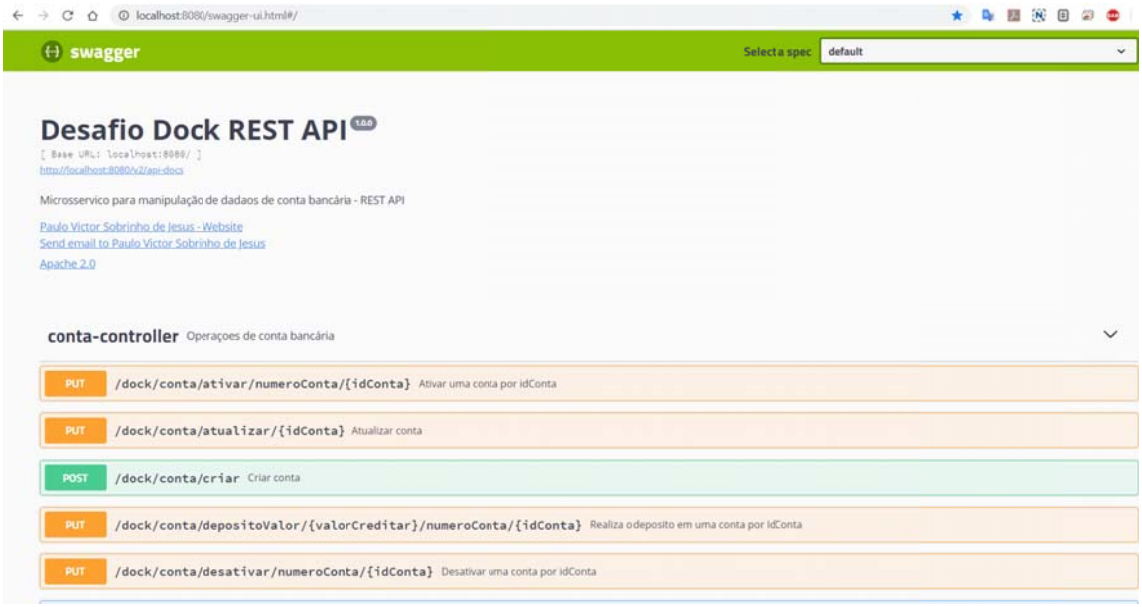
```
CREATE TABLE `transacao` (
  `id_transacao` bigint NOT NULL,
  `data_transacao` datetime NOT NULL,
  `descricao` varchar(255) NOT NULL,
  `id_conta` bigint NOT NULL,
  `valor` decimal(19,2) NOT NULL,
  PRIMARY KEY (`id_transacao`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

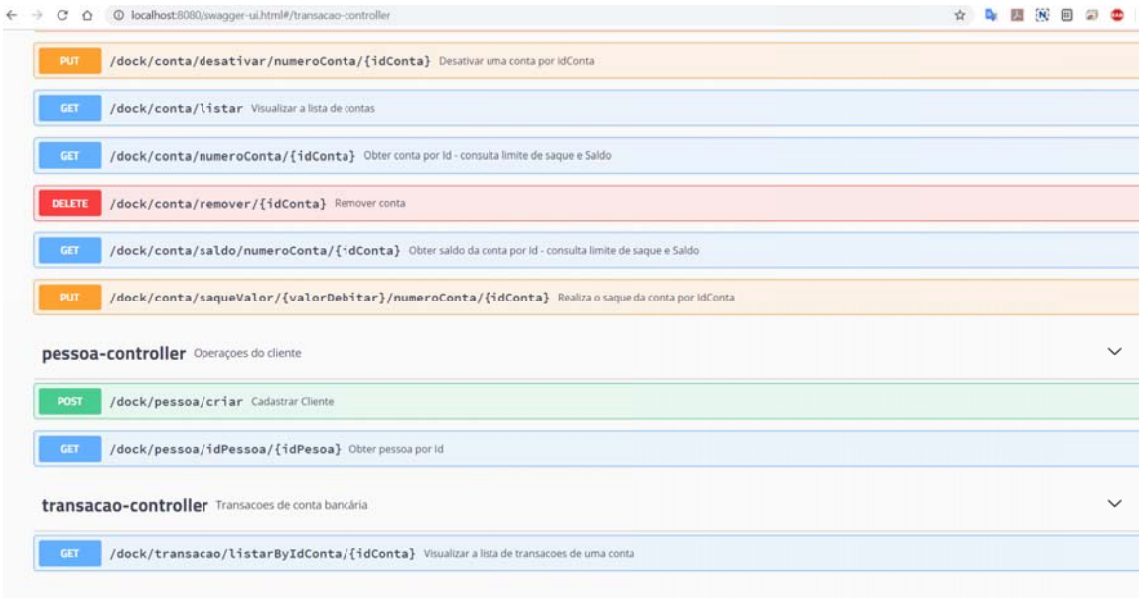
// SCRIPT CRIACAO DE REGISTRO NA TABELA PESSOA

```
INSERT INTO `dock`.`pessoa` (`id_pessoa`, `cpf`, `data_nascimento`, `nome`) VALUES ('14',
'124.456.678-45', '2020-05-20 23:58:01', 'Maria');
```

5. Teste da aplicação com Swagger

ACESSAR O SWAGGER EM : http://localhost:8080/swagger-ui.html#/


localhost:8080/swagger-ui.html#/
swagger
Select a spec
default
Desafio Dock REST API
1.0.0
[Base URL: localhost:8080/]
http://localhost:8080/v2/api-docs
Microsservico para manipulação de dadaos de conta bancária - REST API
Paulo Victor Sobrinho de Jesus - Website
Send email to Paulo Victor Sobrinho de Jesus
Apache 2.0
conta-controller Operaçoes de conta bancária
PUT /dock/conta/ativar/numeroConta/{idConta} Ativar uma conta por idConta
PUT /dock/conta/atualizar/{idConta} Atualizar conta
POST /dock/conta/criar Criar conta
PUT /dock/conta/depositoValor/{valorCreditar}/numeroConta/{idConta} Realiza odeposito em uma conta por IdConta
PUT /dock/conta/desativar/numeroConta/{idConta} Desativar uma conta por idConta


localhost:8080/swagger-ui.html#/transacao-controller
PUT /dock/conta/desativar/numeroConta/{idConta} Desativar uma conta por idConta
GET /dock/conta/listar Visualizar a lista de contas
GET /dock/conta/numeroConta/{idConta} Obter conta por Id - consulta limite de saque e Saldo
DELETE /dock/conta/remover/{idConta} Remover conta
GET /dock/conta/saldo/numeroConta/{idConta} Obter saldo da conta por Id - consulta limite de saque e Saldo
PUT /dock/conta/saqueValor/{valorDebitar}/numeroConta/{idConta} Realiza o saque da conta por IdConta
pessoa-controller Operaçoes do cliente
POST /dock/pessoa/criar Cadastrar Cliente
GET /dock/pessoa/idPessoa/{idPesoa} Obter pessoa por Id
transacao-controller Transacoes de conta bancária
GET /dock/transacao/listarByIdConta/{idConta} Visualizar a lista de transacoes de uma conta