**Documentação**

* Foi baixo o projeto no spring initializr usando as dependência com Maven. Sendo as dependências: Spring Data JPA, Spring Data Web, Postgresql Drive, Lombok e Redis.
* Entidades: Cliente e Produto.
* Relacionamento: Pedido de Compra.
* Na pasta Documentação contem o Diagram UML das tabelas.
* Foi usado o Redis para fazer cache na consulta da tabela de produto. Pelo fato dos dados ter baixa latência e alta taxa de transferência. Podendo futuramenter ser escalado com mais facilidade.
* Para executar o proto baixar o IntelliJ e Docker.

**Para executar:**

* Abrir o projeto no IntelliJ.
* docker-compose up no arquivo .yml pelo CMD, PowerShell ou Terminal da IDE.
* Execute o projeto. E as tabelas serão criadas (não necessita rodar Script):



* No arquivo application.properties e redis.properties encontram-se todas as configurações do Postgresql e Redis.
* Foi usado a porta 8090 para ser usado no Postgresql com usuário e senha: postgres.
* Foi usado a porta 6379 para ser usado no Redis com usuário Redis e senha: 123456.
* Foi criado uma API REST para fazer os CRUDs do cadastro do Cliente, Produto e Pedido de Compra.
* Para fazer um Post no cadastro de Produdo usar o Postman: <http://localhost:8090/produto/>
* E rodar via JSON*:*

*{*

*"descricao" :"Arroz", "quantidade": "1000", "valor": "30"*

*}*

* O mesmo para Cliente: <http://localhost:8090/cliente/>

*{*

*"nome": "Victor", "sobrenome": "Dias", "cpf": "123", "email": "email@email.com", "telefone": "111", "cep": "00000-111", "logadouro": "Rua A", "numero": "10", "complemento": "Casa a",*

*"bairro": "Santana", "cidade": "São Paulo", "estado": "SP",*

*"id": 1 }*

* Consultando as tabelas:





* O mesmo para Efetuar o Pedido de Compra: <http://localhost:8090/pedido/>
* Nas classe Controller de todas as entidades e relacionamento existem um metodo implementado para consultar todos (Cliente ou Produto), consultar pelo ID um entidade específica, alterar consultando pelo ID e deletar informando o ID.