CENTRO UNIVERSITARIO SENAC

Bruno Henrique
Carlos Eduardo
Guilherme Ferreira
Heron Ramos
Levy Paz

São Paulo
2022

Bruno Henrique

Carlos Eduardo

Guilherme Ferreira

Heron Ramos

Levy Paz

Trabalho com foco no Projeto Integrador apresentado ao Senac – Unidade São Paulo, como exigência parcial para obtenção do estudo em Java SWING criando um Sistema CRUD.

Mediado por Prof. Fernando Timoteo Fernandes

São Paulo
2022

# Introdução

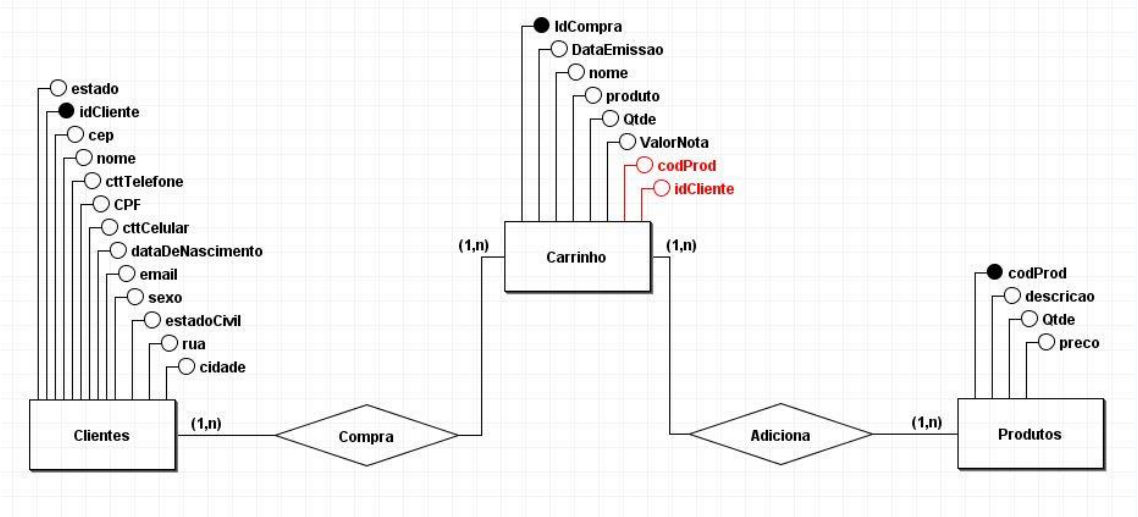
O sistema será composto por dois cadastros principais (cliente e produto), uma tela de venda de produtos e uma tela de “relatório”, que exibirá os resultados da venda dos produtos na própria interface. Os dois cadastros propostos são pré-requisitos para composição da funcionalidade de venda de produto, elemento central do sistema. O primeiro cadastro será composto, basicamente, pelas funcionalidades de manutenção de clientes (CRUD). Deverá ser possível inserir, excluir, alterar e consultar clientes meio de uma pesquisa simples buscando pelo nome ou CPF. Todos os dados principais que normalmente compõem um cadastro de clientes, como nome, cpf, endereço, telefone, e-mail, sexo, estado civil e dados de nascimento devem estar disponíveis na aplicação. Os campos devem ter validação de tipo, tamanho e obrigatoriedade (pelo menos nome, sexo, CPF, e-mail e endereço devem ser inseridos) antes da inserção ou manipulação no banco de dados. Não será permitido cadastrar dois clientes com o mesmo CPF. O segundo cadastro será a manutenção do produto (CRUD) a ser vendido. Deverá ser possível inserir, excluir, alterar e consultar produtos por meio de uma pesquisa simples. Assim como o cliente, o produto também será utilizado para compor a venda. No entanto, terá uma funcionalidade mais complexa em relação ao cadastro de clientes, pois deverá possuir um controle de estoque simples. Ou seja, deve ser possível atualizar o produto com indicadores de estoque, onde as vendas decrementarão este estoque e, da mesma forma, não será possível realizar novas vendas para produtos sem estoque. Os campos necessários para composição da aplicação dependerão do tema escolhido, mas também deve ter validação de tipo, tamanho e obrigatoriedade antes da inserção ou manipulação no banco de dados. Como atividade principal do sistema, está localizado o processo de venda. Uma venda será o registro de saída de estoque de determinados produtos, com acompanhamentos especificados, para determinado cliente num determinado momento no tempo. Durante o processo de venda, o usuário deverá ser capaz de escolher um ou mais produtos a serem vendidos, pois os produtos e qual cliente irá comprá-los e sistema mostrará o valor final da venda e permitirá concretizara-la. Validações de obrigatoriedade (seleção de cliente, produtos e quantidade), tipo, validade e tamanho também devem ser realizados. Por fim, o

relatório de vendas permitirá que o usuário visualize na própria interface do sistema, o resumo das vendas (relatório sintético) em um determinado período (máximo mensal), indicando o valor total das vendas, a data de compra e o cliente. Também será exibido o valor total das vendas do período selecionado no relatório sintético. Além disso, deve ser possível visualizar os detalhes da venda (relatório analítico), com os produtos vendidos em cada venda, bem como suas próprias.

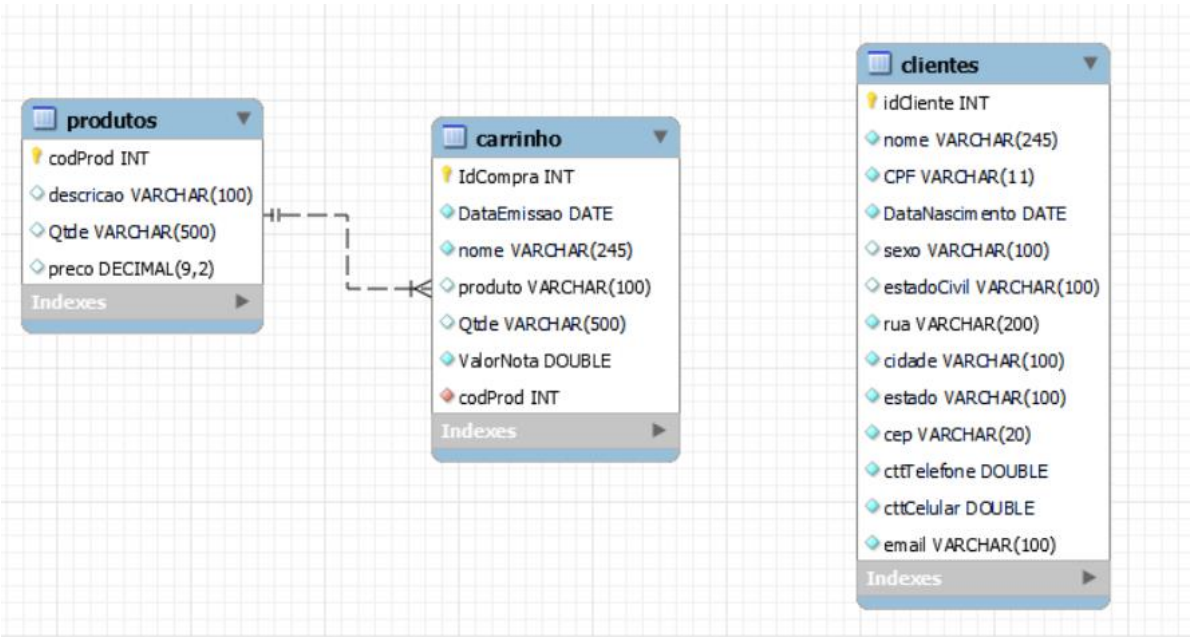
# Banco de Dados

No nosso projeto usamos o MySQL para armazenar as informações dos clientes, produtos e o carrinho de compra onde é armazenada o cliente vinculado a compra dele.

## DER




## Modelo Físico



## Script Banco de dados

```
Create database projetoPi;
use projetoPi;

Create table Produtos (
codProd INT NOT NULL AUTO_INCREMENT,
descricao varchar (100),
Qtde varchar (500),
preco decimal (9,2),
PRIMARY KEY (codProd)
);



select* from Produtos;
Select descricao, preco * Qtde as valor_total
From (Produtos)
group by codProd;

CREATE TABLE Clientes (
idCliente INT NOT NULL AUTO_INCREMENT,
nome varchar (245) Not null,
CPF VARCHAR (11) NOT NULL,
 DataNascimento Date NOT NULL,
sexo varchar (100),
estadoCivil varchar (100),
rua varchar (200) Not null,
cidade varchar (100) Not null,
estado varchar (100) Not null,
```

```
cep varchar (20) Not null,
cttTelefone DOUBLE NOT NULL,
cttCelular DOUBLE NOT NULL,
email varchar (100) Not null,
 PRIMARY KEY (idCliente)
);

select * from Clientes;

CREATE TABLE carrinho (
  IdCompra INT NOT NULL AUTO_INCREMENT,
  DataEmissao Date NOT NULL,
  nome varchar (245) Not null,
  produto varchar (100),
  Qtde varchar (500),
  ValorNota DOUBLE NOT NULL,
  codProd int not null,
  foreign key (codProd) references Produtos(codProd),
  PRIMARY KEY (idCompra)
);

select * from carrinho;

drop table Produtos;
drop table Clientes;
drop table carrinho;

update Produtos inner join carrinho on Produtos.codProd = carrinho. codProd
SET Produtos.Qtde = Produtos.Qtde - carrinho. Qtde
where carrinho. codProd = Produtos.codProd;
```

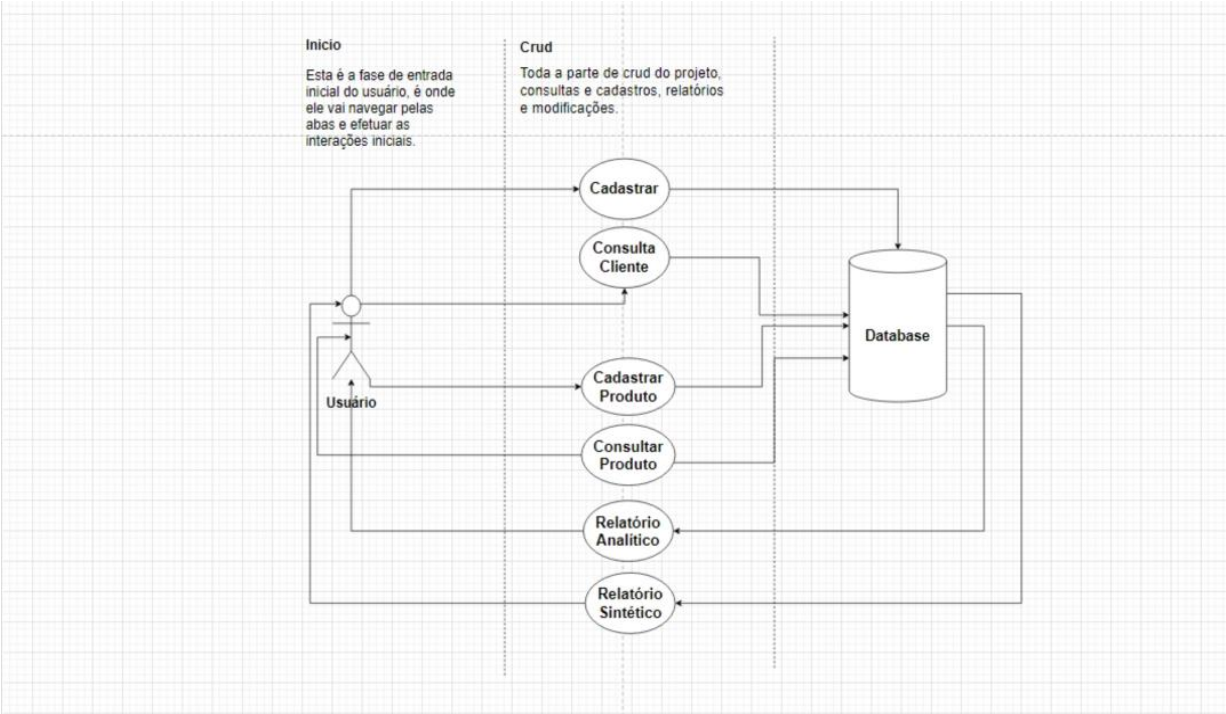
select carrinho. nome, carrinho. produto, carrinho. Qtde, Produtos.preco

from carrinho

inner join Produtos

on carrinho. codProd = carrinho. codProd;

## Diagrama de caso e uso




## Requisitos Funcionais

Dentro da engenharia de uma empresa de software podemos destacar o requisito funcional, onde há a materialização de uma necessidade ou solicitação realizada por um software. Porém, vários Requisitos Funcionais podem ser realizados dentro de uma mesma funcionalidade. São variadas as funções e serviços que um sistema pode fornecer ao seu cliente. Os requisitos funcionais são de extrema importância no desenvolvimento de aplicativos, pois, sem eles não há funcionalidades nos sistemas. Seus modelos devem ser construídos em um nível de entendimento claro e objetivo, além de um código

fonte totalmente aplicável. Conclusão, para se obter requisitos funcionais de qualidade a fábrica de software deve estar atenta a síntese e a semântica deles.

## Requisitos Não-funcionais

Os requisitos não-funcionais estão ligados ao uso da aplicação, em termos de usabilidade, confiabilidade, segurança, facilidade em aplicar manutenções e tecnologias envolvidas

Então, a facilidade de usar a aplicação, o designe simples e de fácil entendimento, é considerado um requisito não-funcional. Nesse caso, focamos em aprimorar a viabilidade do projeto, não só na funcionabilidade, não basta “só funcionar bem”, precisamos levar em conta a experiência do usuário final, que vai utilizar a aplicação em seu dia a dia.

A segurança, como redundâncias e hierarquias de acessos, que evitam que qualquer usuário do sistema apague todas as informações do banco, ou caso insiram informações incorretas, possam alterar e corrigir o erro. São requisitos não funcionais, que tornam o sistema mais seguro.

## Requisitos funcionais no Projeto

**Incluir/Excluir/Alterar:** Nosso projeto temos alguns requisitos funcionais e um deles é poder incluir, excluir ou até mesma altera um cliente ou um produto.

Emissão de relatórios de clientes ou vendas: No projeto também é possível geral um relatório de vendas onde é possível consultar a venda de um cliente, o produto que é comprou, o valor total e o valor de cada item.**Consulta e alterações de dados pessoais de clientes:** No projeto é possível alterar as informações pessoais dos clientes, caso ele mude de telefone, email ou endereço.

**Consulta de saldo ou estoque:** Podemos também buscar em nosso estoque a quantidade de item que ainda tem disponível.

## Requisitos Não-funcionais no Projeto

**Facilidade em aplicar manutenções:** O sistema foi todo desenvolvido orientado à objetos, de forma separada, em pacotes e classes específicos, afim de que no futuro, caso seja necessário atualizar alguma funcionalidade, basta procurar a classe com os métodos que serão aprimorados, ou corrigidos no caso de algum possível bug em uma futura atualização, de forma que se o seu problema foi no cadastro de produtos, quem for dar manutenção no sistema, saberá exatamente onde procurar e corrigir o erro, ou atualizar a função.

**Facilidade de uso:** Focamos em desenvolver um sistema intuitivo, a fim de reduzir o tempo de treinamento dos usuários em uma transição para o nosso sistema, um exemplo disso, é a forma como o sistema foi dividido em telas que são facilmente encontradas nas listas de menus, por exemplo, se você quiser cadastrar um novo produto, basta procurar a lista relacionada, nesse caso seria “Produtos”, e lá você já encontra o acesso a tela de “cadastro de produtos”.

**Segurança na hierarquia de acessos:** Para evitar que qualquer usuário tenha acesso a informações sensíveis, ou a funcionalidades de alta responsabilidade, nós separamos os cadastros de gerentes e vendedores.