DCC060 – Trabalho Prático

Alunos: Márcio Felipe Daniel Gonçalves, Daniel Jorge Reis Caldeira

1. INTRODUÇÃO

O projeto tem como principal objetivo simular o funcionamento de uma loja online. Onde os usuários podem ser clientes, entregadores, gerentes e funcionários, tendo cada um funcionalidades distintas, onde o cliente realiza as compras, o entregador entrega o pedido, o funcionário gerencia os produtos e o gerente gerencia os produtos e os funcionários.

2. REQUISITOS DO PROJETO

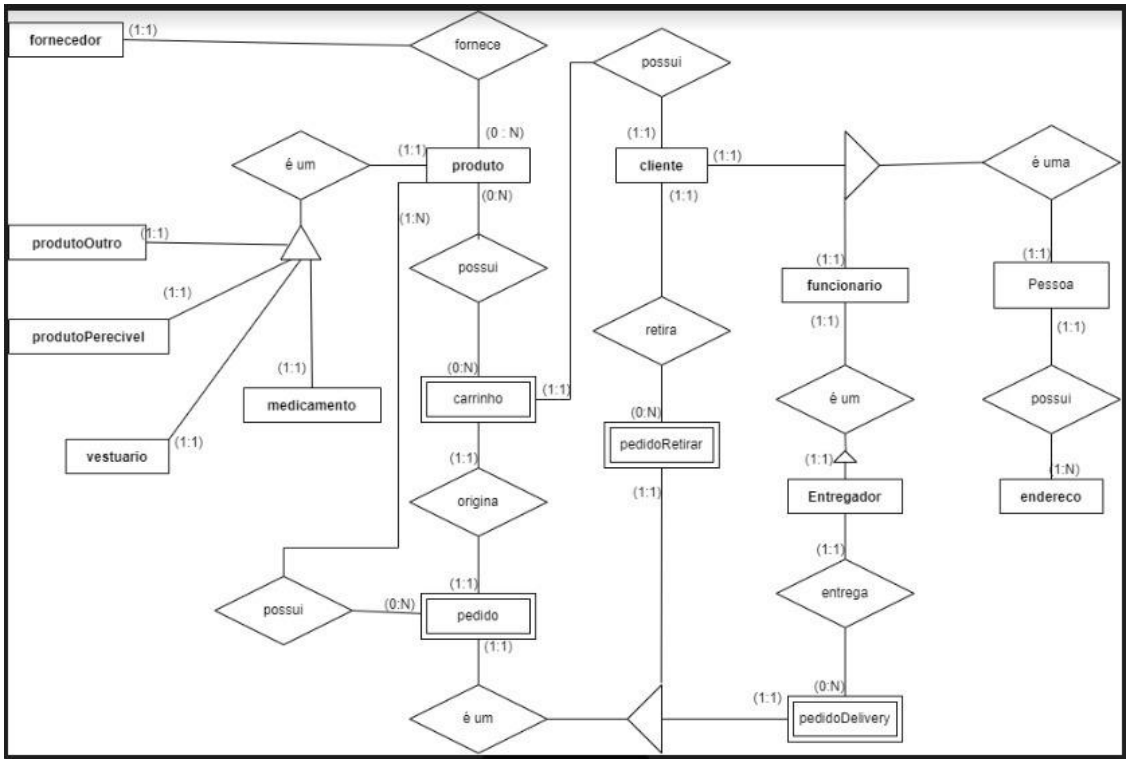
Requisitos do sistema
1. A aplicação deve ter um sistema de login onde define o fluxo de interfaces de acordo com o tipo de usuário. 2. A aplicação deve permitir que novos clientes e funcionários sejam cadastrados. 3. A aplicação deve permitir que os funcionários e gerentes possam cadastrar e excluir produtos. 4. O cliente deve ter acesso a lista de produtos em estoque. 5. O cliente pode adicionar produtos em seu carrinho. 6. O cliente pode remover produtos de seu carrinho. 7. Apenas gerentes e funcionários podem alterar dados dos produtos 8. O entregador deve ter acesso aos seus pedidos para entrega. 9. O gerente pode contratar outros funcionários 10. O cliente pode ver os seus pedidos.

Regras de Negócio
1. É possível ter apenas um gerente no sistema. 2. Todo funcionário deve ter um cargo.

3. O sistema atribui um entregador aleatório para cada pedido delivery.
4. Caso não tenha gerente definido , a conta admin deve ser utilizada para cadastrar um gerente , e seu uso será apenas para esse fim.
5. O cliente pode pagar sua compra à vista ou a prazo.
6. O pedido pode ser entregue por delivery ou retirado na loja pelo cliente.
7. O cliente não poderá realizar uma compra a prazo caso o valor passe de 500 reais.
8. O cliente não poderá realizar uma nova compra a prazo caso tenha alguma conta sem pagar.
9. O sistema deve permitir pedidos a delivery apenas se tiver entregador cadastrado no sistema
10. Apenas o gerente pode editar dados de funcionário.
11. Produtos perecíveis não podem ser cadastrados com o vencimento no mesmo dia de cadastro.

3. MODELAGEM RELACIONAL

3.1 Diagrama Entidade Relacionamento



3.2 Modelagem lógica

endereco(id,cidade, bairro, rua, cep, uf, numero)
o id é incrementado automaticamente no banco de dados

pessoa (cpf, nome,endereco,senha)

cliente(cpf,conta)
cliente.cpf referencia pessoa.cpf

funcionario (id, cpf, cargo)
funcionario.cpf referencia pessoa.cpf
o id é incrementado automaticamente no banco de dados

entregador(id, placaveiculo)
entregador.id referencia funcionario.id

fornecedor(cnpj, nome)

produto(id, fornecedor,preco,quantidade,nome,tipo)
produto.fornecedor referencia fornecedor.cnpj
o id é incrementado automaticamente no banco de dados

produtooutro(id)
produtooutro.id referencia produto.id

produtoperecivel(id, validade)
produtoperecivel.id referencia produto.id

remedio(id, validade)
remedio.id referencia produto.id

vestuario(id, tamanho)
vestuario.id referencia produto.id

carrinho(cliente, produto, quantidade)
carrinho.cliente referencia cliente.cpf
carrinho.produto referencia produto.id

pedido(id, cliente, status)
pedido.cliente referencia cliente.cpf
o id é incrementado automaticamente no banco de dados

pedidodelivery (id, entregador)
pedidodelivery,id referencia pedido.id
pedidodelivery.entregador referencia entregador.id

pedidoretirar(id)
pedidoretrar.id rfrencia pedido.id

produtopedido(pedido, produto, quantidade)
produtopedido.pedido referencia pedido.id
produtopedido.produto referencia produto.id

4 CONSULTAS EM ÁLGEBRA RELACIONAL

Consulta 1: Selecione o idproduto e a quantidade de um "produtoAlvo" caso exista no carrinho do "clienteAlvo"
$\pi_{\{produto,quantidade\}}(\sigma_{produto = produtoAlvo\ and\ cliente = clienteAlvo}(carrinho))$

Consulta 2: Liste todos os produtos que estão no carrinho do "clienteAlvo"
$\sigma_{cliente=clienteAlvo}(carrinho)$

Consulta 3: Selecione a quantidade de um "produtoAlvo" que está no carrinho do "clienteAlvo"
$\pi_{\{,quantidade\}}(\sigma_{produto = produtoAlvo\ and\ cliente = clienteAlvo}(carrinho))$

Consulta 4: Selecione os dados do cliente caso exista um cliente com a senha e cpf informado

$$ClienteAlvo\ \{\ c.cpf, p.nome, p.senha, p.endereco, c.conta\} \leftarrow pessoa\ \otimes\ cliente$$

$$\sigma_{senha = senhaAlvo\ and\ c.cpf = cpfAlvo}(ClienteAlvo)$$

Consulta 5:Selecione os dados do cliente que possui o cpf informado

$$ClienteAlvo\ \{\ c.cpf, p.nome, p.senha, p.endereco, c.conta\} \leftarrow pessoa\ \otimes\ cliente$$

$$\sigma_{c.cpf = cpfAlvo}(ClienteAlvo)$$

Consulta 6: Selecione a placa do veículo do entregador que possui o id informado

$$\pi_{\{placaVeiculo\}}(\sigma_{id = idAlvo}(entregador))$$

Consulta 7: Selecione os dados do entregador que possui o id informado

$$EntregadorAlvo\ \{.id, f.cpf, f.cargo, en.placaveiculo\} \leftarrow funcionario\ \otimes\ entregador$$

$$\sigma_{id = idAlvo}(EntregadorAlvo\ \otimes\ pessoa)$$

Consulta 8: Selecione os dados do funcionário caso exista um funcionário com a senha e id informado

$$FuncionarioAlvo\{p.cpf, p.nome, p.senha, p.endereco, f.id, f.cargo\} \leftarrow pessoa \otimes fu$$

$$\sigma_{senha = senhaAlvo\ and\ f.id = idAlvo}(FuncionarioAlvo)$$

Consulta 9: Selecione o cargo do funcionário que tenha o idAlvo

$$\pi_{\{cargo\}}(\sigma_{id = idAlvo}(funcionario))$$

Consulta 10: Selecione todos os dados de todos os funcionários cadastrados no sistema

$$FuncionarioAlvo\{p.cpf, p.nome, p.senha, p.endereco, f.id, f.cargo\} \leftarrow pessoa \otimes fu$$

$$\pi_{\{p.cpf, p.nome, p.senha, p.endereco, f.id, f.cargo\}}(FuncionarioAlvo)$$

Consulta 11: Selecione o tipo do produto que tenha um id igual ao id idAlvo

$$\pi_{\{tipo\}}(\sigma_{id = idAlvo}(produto))$$

Consulta 12: Selecione o fornecedor do produto que tenha o id igual ao idAlvo
$\pi_{\{fornecedor\}}(\sigma_{id = idAlvo}(produto))$

Consulta 13: Selecione os dados de todos os produtos cadastrados
$\pi_{\{id,nome,preco,tipo\}}(produto)$

Consulta 14: Selecione o nome do fornecedor que tenha o cnpj igual ao cnpj informado
$\pi_{\{nome\}}(\sigma_{cnpj = cnpjAlvo}(fornecedor))$

Consulta 15:Selecione os dados do fornecedor que tenha o cnpj igual ao cnpj informado
$\pi_{\{nome,cnpj\}}(\sigma_{cnpj = cnpjAlvo}(fornecedor))$

5 MODELO FÍSICO (Parte 1) E CARGA DE DADOS

Todos os itens abaixo devem ser descritos no relatório.

5.1 Tabelas

```
CREATE table if not exists endereco(
	id serial primary key,
	uf varchar(2) not null,
	cidade varchar(50) not null,
	bairro varchar(50) not null,
	rua varchar(50) not null,
	numero varchar(10) not null,
	cep varchar(8) not null
);

create table if not exists pessoa(
	cpf varchar(11) primary key,
	nome varchar(50) not null,
	senha varchar(20) not null,
	endereco integer,
	FOREIGN KEY (endereco) REFERENCES public.endereco (id) on DELETE
CASCADE
);

CREATE table if not exists cliente(
	cpf varchar(11) primary key,
	conta real not null default 0,
	FOREIGN KEY (cpf) REFERENCES public.pessoa (cpf) on DELETE CASCADE
);

CREATE table if not exists fornecedor(
	cnpj varchar(15) primary key,
	nome varchar(50) not null
);

CREATE table if not exists funcionario(
	id serial primary key,
	cpf varchar(11) unique not null,
	cargo varchar(11),
	FOREIGN KEY (cpf) REFERENCES public.pessoa (cpf) on DELETE CASCADE
```

```
);

create table if not exists entregador(
	placaVeiculo varchar(7) unique not null,
	id integer primary key,
	FOREIGN KEY (id) REFERENCES public.funcionario (id) on DELETE
CASCADE
);

create table if not exists produto(
	id serial primary key,
	fornecedor varchar(15) not null,
	preco real not null,
	nome varchar(50) not null,
	quantidade real not null,
	tipo varchar(25) not null,
	FOREIGN KEY (fornecedor) REFERENCES public.fornecedor (cnpj) on DELETE
CASCADE
);

create table if not exists vestuario(
	id integer primary key,
	tamanho varchar(5),
	FOREIGN KEY (id) REFERENCES public.produto (id) on DELETE CASCADE
);

create table if not exists remedio(
	id integer primary key,
	precisaReceita bool not null,
	validade varchar(10) not null,
	FOREIGN KEY (id) REFERENCES public.produto (id) on DELETE CASCADE
);

create table if not exists produtoperecivel(
	id integer primary key,
	validade varchar(10) not null,
	FOREIGN KEY (id) REFERENCES public.produto (id) on DELETE CASCADE
);

create table if not exists produtooutro(
```

```
	id integer primary key,
	FOREIGN KEY (id) REFERENCES public.produto (id) on DELETE CASCADE
);

create table if not exists carrinho(
	cliente varchar(11) not null,
	produto integer not null,
	quantidade real not null,
	FOREIGN KEY (cliente) REFERENCES public.cliente (cpf) on DELETE
CASCADE,
	FOREIGN KEY (produto) REFERENCES public.produto (id) on DELETE
CASCADE,
	CONSTRAINT PK_carrinho PRIMARY KEY (cliente, produto)
);

create table if not exists pedido(
	id serial primary key,
	cliente varchar(11) not null,
	status varchar(25) not null default 'Aguardando confirmação',
	FOREIGN KEY (cliente) REFERENCES public.cliente (cpf) on DELETE
CASCADE
);

create table if not exists pedidodelivery(
	entregador integer not null,
	id integer primary key,
	FOREIGN KEY (id) REFERENCES public.pedido (id) on DELETE CASCADE,
	FOREIGN KEY (entregador) REFERENCES public.entregador (id) on DELETE
CASCADE
);

create table if not exists pedidoretirar(
	id integer primary key,
	FOREIGN KEY (id) REFERENCES public.pedido (id) on DELETE CASCADE
);

create table if not exists produtopedido(
	pedido integer not null,
	produto integer not null,
	quantidade real not null,
```

```
	FOREIGN KEY (pedido) REFERENCES public.pedido (id) on DELETE
CASCADE,
	FOREIGN KEY (produto) REFERENCES public.produto (id) on DELETE
CASCADE,
	CONSTRAINT PK_produtopedido PRIMARY KEY (pedido, produto)
);
```

5.2 Verificação

```
INSERT INTO carrinho(cliente, produto, quantidade) VALUES (?,?,?);
UPDATE carrinho SET quantidade = ? WHERE cliente = ? AND produto = ?;
DELETE FROM carrinho WHERE cliente = ? AND produto = ?;
DELETE FROM carrinho WHERE cliente = ?;
UPDATE carrinho SET quantidade = ? WHERE produto = ? and cliente = ?
INSERT INTO cliente (cpf) VALUES (?);
INSERT INTO endereco(uf, cidade, bairro, rua, numero, cep) VALUES (?,?,?,?,?,?);
INSERT INTO pessoa (cpf, nome, senha, endereco) VALUES (?,?,?,?);
INSERT INTO pessoa (cpf, nome, senha) VALUES (?,?,?);
DELETE FROM cliente WHERE cpf = ?;
DELETE FROM pessoa WHERE cpf = ?;
UPDATE cliente SET conta = ? WHERE cpf = ?;
INSERT INTO funcionario (cpf, cargo) VALUES (?,'Entregador');
INSERT INTO entregador (placaveiculo, id) VALUES (?,?);
INSERT INTO endereco(uf, cidade, bairro, rua, numero, cep) VALUES (?,?,?,?,?,?);
INSERT INTO pessoa (cpf, nome, senha, endereco) VALUES (?,?,?,?);
INSERT INTO pessoa (cpf, nome, senha) VALUES (?,?,?);
UPDATE entregador SET placaveiculo = ? WHERE id = ?;
DELETE FROM entregador WHERE id = ?;
DELETE FROM funcionario WHERE id = ?;
UPDATE entregador SET placaveiculo = ? WHERE id = ?;
INSERT INTO fornecedor (cnpj, nome) VALUES (?,?);
UPDATE fornecedor SET nome = ? WHERE cnpj = ?;
DELETE FROM fornecedor WHERE cnpj = ?;
INSERT INTO funcionario (cpf,cargo) VALUES (?,?);
INSERT INTO endereco(uf, cidade, bairro, rua, numero, cep) VALUES (?,?,?,?,?,?);
INSERT INTO pessoa (cpf, nome, senha, endereco) VALUES (?,?,?,?);
UPDATE funcionario SET cargo = ? WHERE id = ?;
INSERT INTO pedido (cliente, pagamento) VALUES (?, ?);
INSERT INTO pedidoretirar (id) VALUES (?);
INSERT INTO pedidodelivery (id,entregador) VALUES (?,?);
UPDATE pedido SET status = ? WHERE id = ;
```

```
UPDATE pessoa SET senha = ? WHERE cpf = ?;
UPDATE pessoa SET nome = ? WHERE cpf = ?;
UPDATE endereco SET uf = ?, cidade = ?, bairro = ?, rua = ?, numero = ?, cep = ?
WHERE id = (SELECT endereco FROM pessoa WHERE cpf = ?);
UPDATE pessoa SET endereco = ? WHERE cpf = ?;
UPDATE produto SET (fornecedor, preco,nome,quantidade) = (?,?,?,?) WHERE id = ?;
UPDATE produto SET quantidade = ? where id = ?;
INSERT INTO produto (fornecedor, nome, preco, quantidade,tipo) VALUES
(?,?,?,?,'Outros');
INSERT INTO produtooutro (id) VALUES (?);
DELETE FROM produtooutro WHERE id = ?;
DELETE FROM produto WHERE id = ?;
UPDATE produtoperecivel SET validade  = ? WHERE id = ?;
INSERT INTO produto (fornecedor, nome, preco, quantidade,tipo) VALUES
(?,?,?,?,'Perecível');
INSERT INTO produtoperecivel (id,validade) VALUES (?,?);
DELETE FROM produtoperecivel WHERE id = ?;
INSERT INTO produto (fornecedor, nome, preco, quantidade,tipo) VALUES
(?,?,?,?,'Remédio');
INSERT INTO remedio (id, validade) VALUES (?, ?);
DELETE FROM remedio WHERE id = ?;
UPDATE remedio SET validade  = ? WHERE id = ?;
INSERT INTO produto (fornecedor, nome, preco, quantidade,tipo) VALUES
(?,?,?,?,'Vestuário');
INSERT INTO vestuario (id, tamanho) VALUES (?,?);
SELECT tamanho FROM vestuario WHERE id = ?;
UPDATE vestuario SET t tamanho  = ? WHERE id = ?;
DELETE FROM vestuario WHERE id = ?;
```

5.3 Carga de Dados

```
insert into endereco (uf,cidade,bairro,rua,numero,cep) values('MG','juiz de
fora','Centro','Rio Branco','12','12345678');
insert into pessoa (cpf,nome,senha,endereco)
values('12345678910','José','jose123',(select Max(id) from endereco));
insert into cliente(cpf,conta) values('12345678910',0);


insert into endereco (uf,cidade,bairro,rua,numero,cep) values('MG','juiz de
fora','Centro','Mister Moo','1230','12347896');
```

```
insert into pessoa (cpf,nome,senha,endereco)
values('12345678789','Henrique','Henrique123',(select Max(id) from endereco));
insert into cliente(cpf,conta) values('12345678789',0);

insert into endereco (uf,cidade,bairro,rua,numero,cep) values('MG','juiz de
fora','Centro','Francisco bernadino','567','12341234');
insert into pessoa (cpf,nome,senha,endereco)
values('12345677984','Pedro','Pedro123',(select Max(id) from endereco));
insert into cliente(cpf,conta) values('12345677984',0);

insert into endereco (uf,cidade,bairro,rua,numero,cep) values('MG','juiz de
fora','Bandeirantes','Rua bandeira ','567','78945610');
insert into pessoa (cpf,nome,senha,endereco)
values('12345674571','Felipe','Felipe123',(select Max(id) from endereco));
insert into funcionario(cpf,cargo) values('12345674571','Gerente');

insert into endereco (uf,cidade,bairro,rua,numero,cep) values('MG','juiz de
fora','Benfica','Rua belfim Moreira ','789','45127403');
insert into pessoa (cpf,nome,senha,endereco)
values('45741029846','Kleiton','Kleiton123',(select Max(id) from endereco));
insert into funcionario(cpf,cargo) values('45741029846','Funcionario');

insert into endereco (uf,cidade,bairro,rua,numero,cep) values('MG','juiz de
fora','Benfica','Rua belfim Moreira ','900','45127403');
insert into pessoa (cpf,nome,senha,endereco)
values('45741029021','Melisa','Melisa123',(select Max(id) from endereco));
insert into funcionario(cpf,cargo) values('45741029021','Entregador');
insert into entregador(placaveiculo,id) values('Abc123',(select MAx(id) from
funcionario));

insert into fornecedor (cnpj,nome) values('12345678910120','Padaria feliz Sorriso');
insert into pedido (cliente,status) values('12345678910','Aguardando Confirmação');
insert into produto(fornecedor,preco,nome,quantidade,tipo)
values('12345678910120',12,'Pao',12,'Perecivel');
insert into produtoperecivel(id,validade) values((select Max(id) from
produto),'12/12/2023');
insert into carrinho(cliente,produto,quantidade) values('12345678910',(Select MAx(id)
from produto),4);
insert into produtopedido(pedido,produto,quantidade) values((Select MAx(id) from
pedido),(Select MAx(id) from produto),4);
```

```
insert into produto(fornecedor,preco,nome,quantidade,tipo)
values('12345678910120',12,'Nelzadina',12,'Remedio');
insert into remedio(id,precisareceita,validade) values((select Max(id) from
produto),false,'12/12/2023');
insert into carrinho(cliente,produto,quantidade) values('12345678910',(Select MAx(id)
from produto),4);
insert into produtopedido(pedido,produto,quantidade) values((Select MAx(id) from
pedido),(Select MAx(id) from produto),4);

insert into produto(fornecedor,preco,nome,quantidade,tipo)
values('12345678910120',12,'Bone',12,'Vestuario');
insert into vestuario(id,tamanho) values((select Max(id) from produto),'M');
insert into carrinho(cliente,produto,quantidade) values('12345678910',(Select MAx(id)
from produto),4);
insert into produtopedido(pedido,produto,quantidade) values((Select MAx(id) from
pedido),(Select MAx(id) from produto),4);

insert into produto(fornecedor,preco,nome,quantidade,tipo)
values('12345678910120',12,'Esqueiro',12,'Outro');
insert into produtooutro(id) values((select Max(id) from produto));
insert into produtopedido(pedido,produto,quantidade) values((Select MAx(id) from
pedido),(Select MAx(id) from produto),4);

insert into pedidodelivery(entregador,id) values((select Max(id) from funcionario),(select
Max(id) from pedido));
```

6. CONSULTAS EM SQL

Consulta 1: Realize uma consulta que retorna o idproduto e a quantidade de um “produtoAlvo” caso exista no carrinho do “clienteAlvo”
SELECT produto, quantidade FROM carrinho where cliente = clienteAlvo and produto = produtoAlvo

Consulta 2: Realize uma consulta que retorne todos os produtos que estão no carrinho do “clienteAlvo”

```
SELECT produto, quantidade FROM carrinho where cliente = clienteAlvo
```

Consulta 3: Realize uma consulta que retorne a quantidade de um “produtoAlvo” que está no carrinho do “clienteAlvo”

```
SELECT quantidade FROM carrinho WHERE cliente = clienteAlvo AND
produto = produtoAlvo
```

Consulta 4: Faça uma consulta que retorne os dados do cliente caso exista um cliente com a senha e cpf informado

```
select  *

from (select c.cpf,p.nome,p.senha,p.endereco,c.conta  from pessoa as p inner
join cliente as c on p.cpf = c.cpf ) as foo

where foo.cpf = cpfAlvo and foo.senha = senhaAlvo
```

Consulta 5: Faça uma consulta que retorne os dados do cliente que possui o cpf informado

```
select  *

from (select c.cpf,p.nome,p.senha,p.endereco,c.conta  from pessoa as p inner
join cliente as c on p.cpf = c.cpf ) as foo

where foo.cpf = cpfAlvo
```

Consulta 6: Faça uma consulta que retorne a placa do veículo do entregador que possui o id informado
SELECT placaVeiculo FROM entregador WHERE id = idAlvo

Consulta 7: Faça uma consulta que retorne os dados do entregador que possui o id informado
select * from (select f.id,f.cpf,f.cargo,p.nome,p.senha,p.endereco,en.placaveiculo from pessoa as p , funcionario as f,entregador as en where f.cpf=p.cpf) as foo where foo.id=idAlvo

Consulta 8: Faça uma consulta que retorne os dados do funcionário caso exista um funcionário com a senha e id informado
select * from (select p.cpf,p.nome,p.senha,p.endereco,f.id,f.cargo from pessoa as p inner join funcionario as f on p.cpf = f.cpf) as foo where foo.id = idAlvo and senha = senhaAlvo

Consulta 9: Faça uma consulta que retorne o cargo do funcionário que tenha o idAlvo
SELECT cargo FROM funcionario WHERE id= **idAlvo**

Consulta 10: Faça uma consulta que retorne todos os dados de todos os funcionários cadastrados no sistema
select * from (select p.nome,p.cpf,p.senha,p.endereco,f.cargo,f.id from pessoa as p, funcionario as f where f.cpf = p.cpf) as foo

Consulta 11: Faça uma consulta que retorne o tipo do produto que tenha um id igual ao id idAlvo
select tipo from produto where id=idAlvo

Consulta 12: Faça uma consulta que retorne o fornecedor do produto que tenha o id igual ao idAlvo
SELECT fornecedor FROM produto WHERE id = idAlvo;

Consulta 13: Faça uma consulta que retorne os dados de todos os produtos cadastrados
select id,nome,preco from produto

Consulta 14: Faça uma consulta que retorne o nome do fornecedor que tenha o cnpj igual ao cnpj informado
SELECT nome FROM fornecedor WHERE cnpj = cnpjAlvo

Consulta 15: Faça uma consulta que retorne os dados do fornecedor que tenha o cnpj igual ao cnpj informado
select f.cnpj,f.nome from fornecedor as f where f.cnpj=cnpjAlvo

7 MODELO FÍSICO (Parte 2)

7.1 Triggers e Funções

7.1.1 Trigger e procedure que remove os produtos vencidos

```
CREATE OR REPLACE FUNCTION removeprodutosvencidos(produtoID integer)
RETURNS trigger AS
        $BODY$
    DECLARE
                dataAtual date;
                dataValidade date;
    BEGIN
                SELECT CONVERT(datetime, produto.validade , 103) INTO
dataValidade;
                SELECT CONVERT (datetime,CURRENT_DATE,103) INTO dataAtual;
                if validade < dataAtual then
                        DELETE FROM produto WHERE produto.id = produtoId;
                end if;
    END;
        $BODY$
    LANGUAGE plpgsql;
```

```
CREATE TRIGGER removecencidos AFTER INSERT OR UPDATE OR DELETE ON
public.produto FOR EACH ROW EXECUTE PROCEDURE
removeprodutosvencidos(produto.id);
```

7.1.2 Trigger e procedure que remove os produtos sem estoque

```
CREATE OR REPLACE FUNCTION removeprodutossemestoque(produtoId integer)
RETURNS trigger AS

        $BODY$

        DECLARE

                quantidadeP double;

    BEGIN

                SELECT quantidade FROM produo WHERE id = produtoId INTO
quantidadeP;

                 if  quantidadeP = 0 then

                        DELETE FROM produto WHERE produto.id = produtoId;

                 end if;

    END;

        $BODY$

    LANGUAGE plpgsql;


CREATE TRIGGER removesemestoque AFTER INSERT OR UPDATE ON
public.produto FOR EACH ROW EXECUTE PROCEDURE
removessemestoque(produto.id);
```

7.2 Verificação

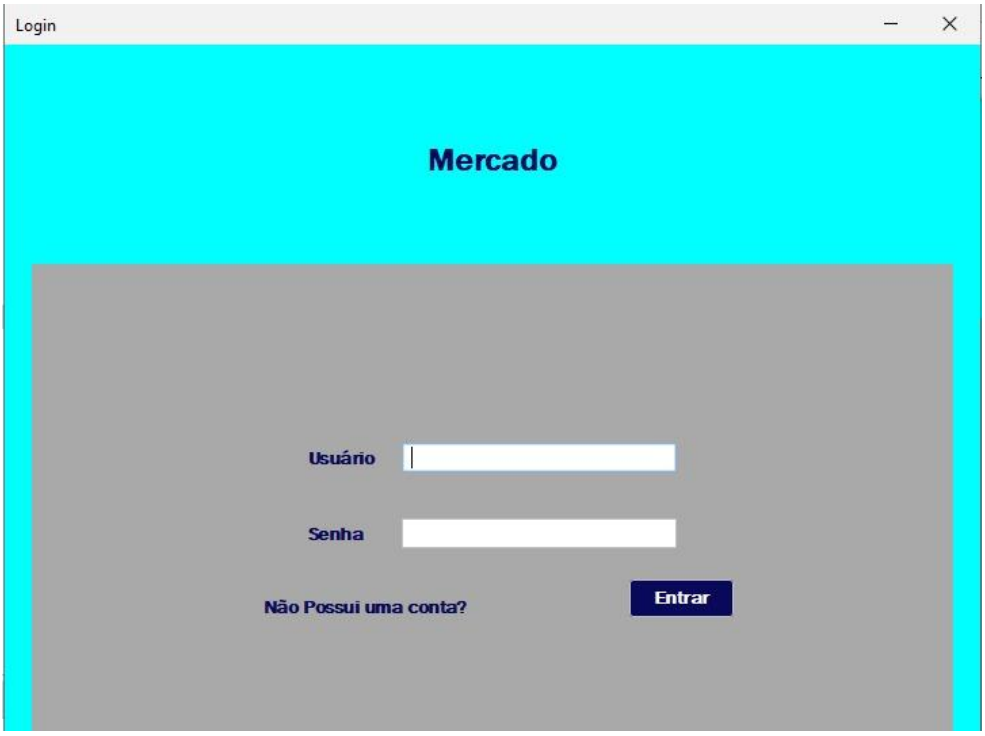
```
SELECT CONVERT(datetime, produto.validade , 103) INTO dataValidade;

SELECT CONVERT (datetime,CURRENT_DATE,103) INTO dataAtual;
```

DELETE FROM produto WHERE produto.id = produtoId;

SELECT quantidade FROM produo WHERE id = produtoId INTO quantidadeP;

8. TELAS DA APLICAÇÃO



RF 1, RF 2

RF 2

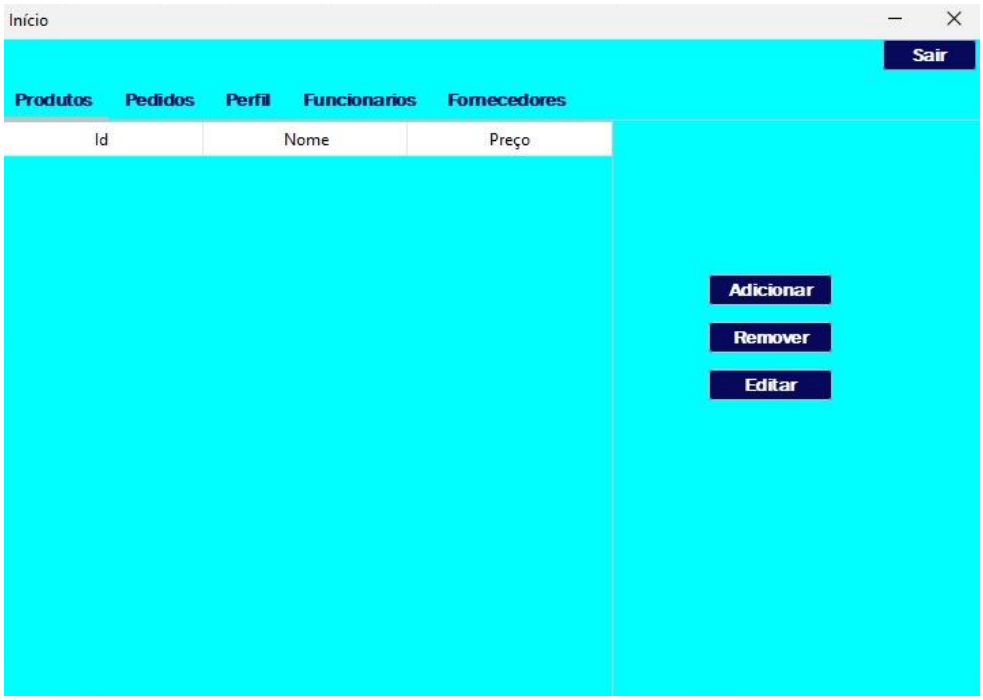
RF 2, RF 9, RN 2, RN 10


Cadastro
Novo Fornecedor
Nome
CNPJ
Voltar
Cadastrar

RF 3


Cadastro
Novo Produto
Nome
Preço
Fornecedor
<Selecione o Fornecedor>
Tipo
<Selecione o Tipo do Produto>
Quantidade
Tamanho
Validade
Voltar
Cadastrar

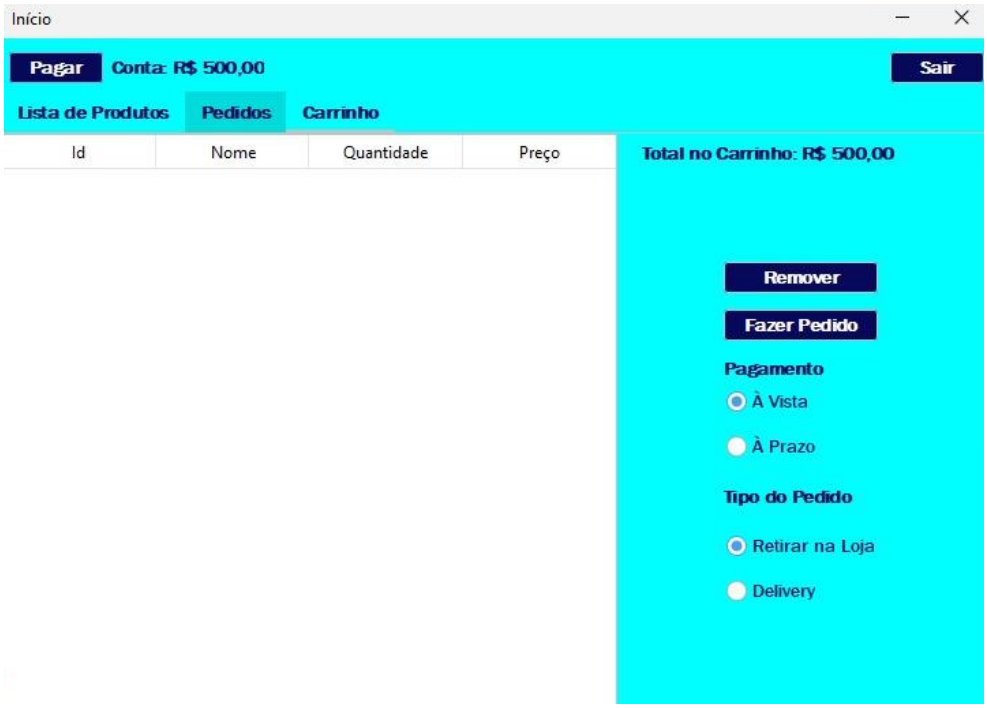
RF 3, RN 11


Início
Sair
Produtos
Pedidos
Perfil
Funcionarios
Fornecedores
Id
Nome
Preço
Adicionar
Remover
Editar

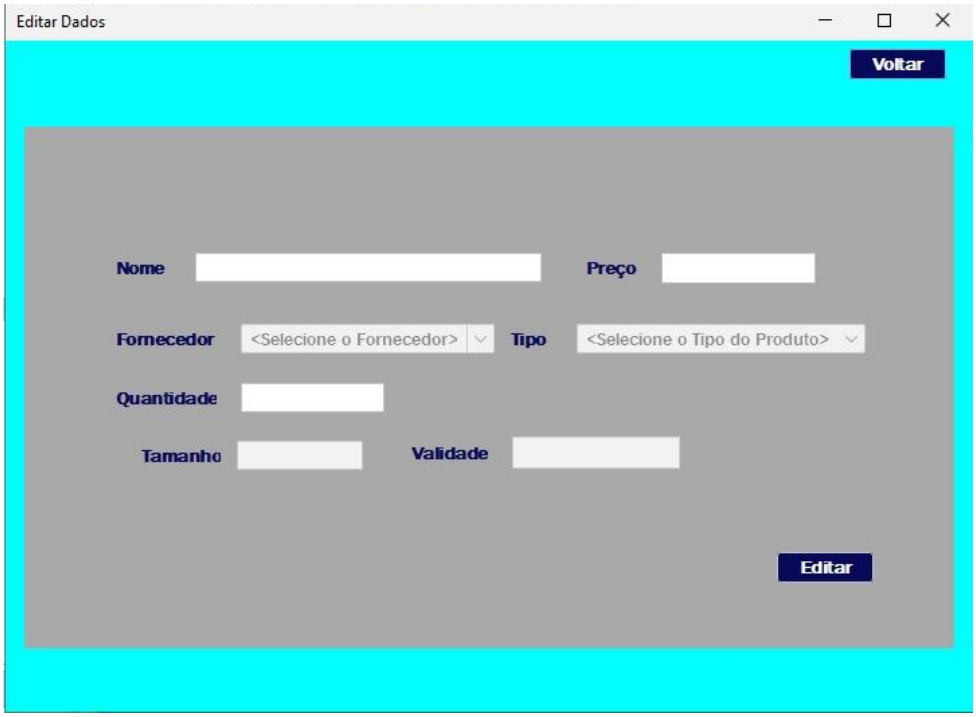
RF 3


Início
Pagar
Conta: R$ 500,00
Sair
Lista de Produtos
Pedidos
Carrinho
Id
Nome
Preço
Quantidade
Adicionar

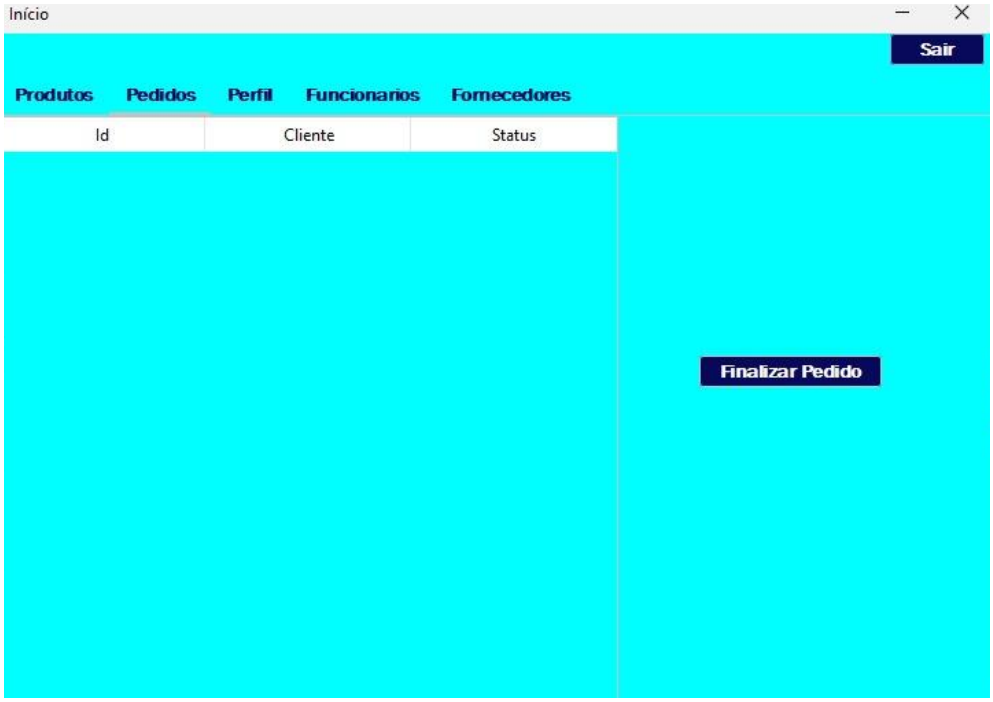
RF 4, RF5



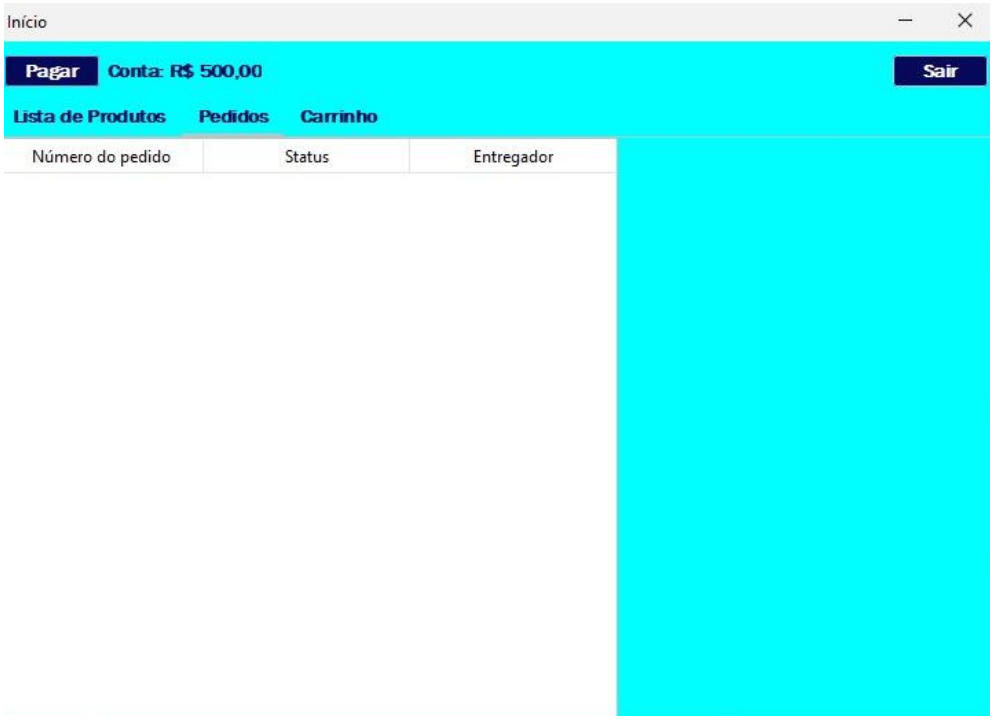
RF 6, RN 5, RN 6, RN 9,



RF 7



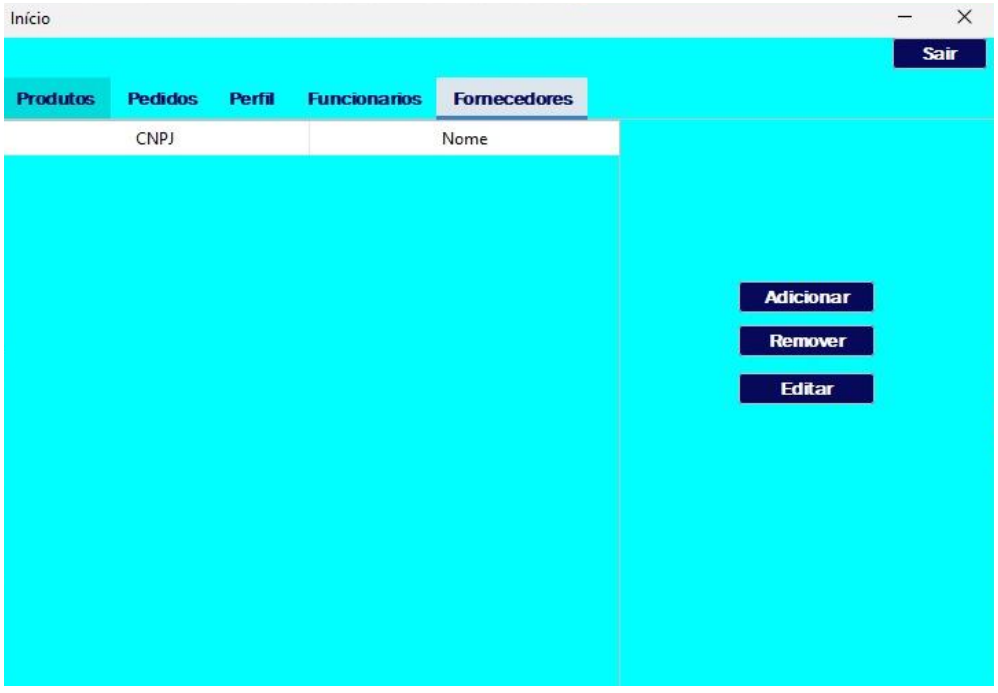
RF 8

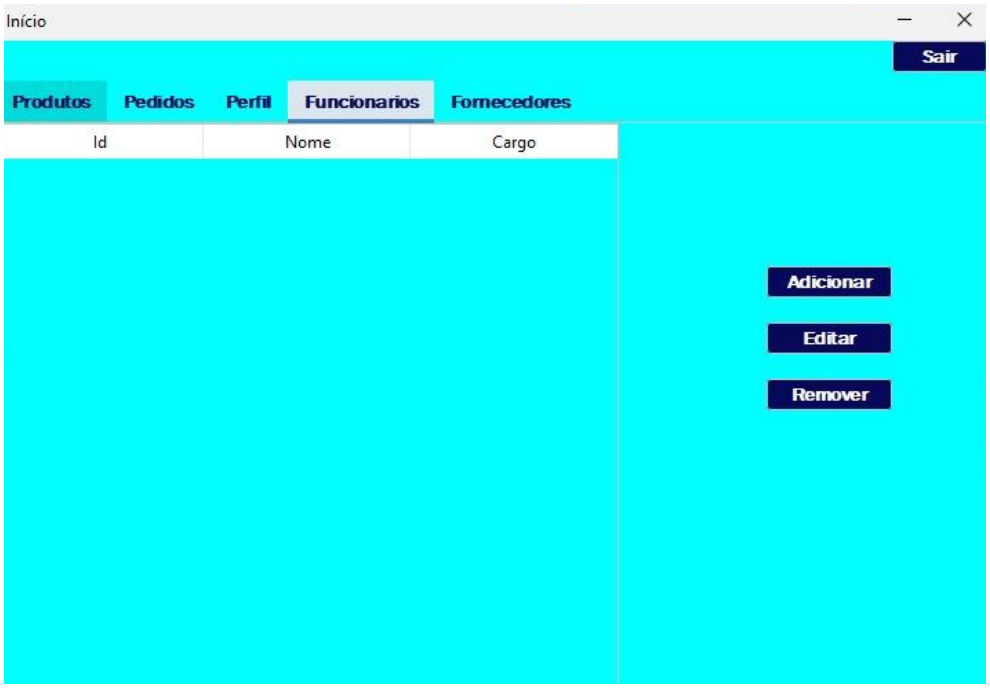


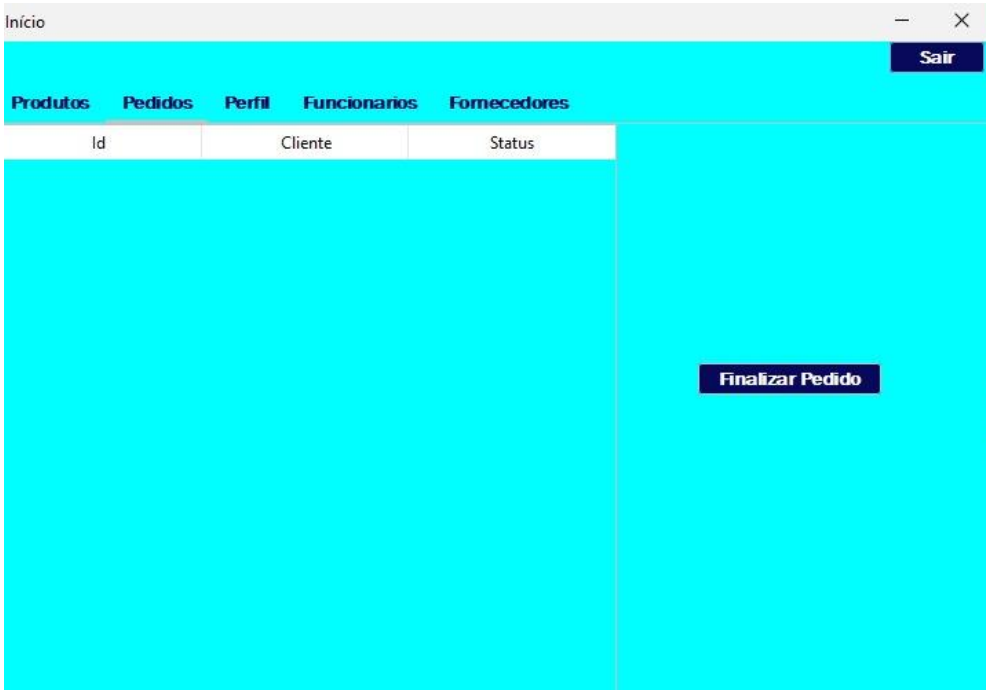
RF 10

Outras telas


Editar Dados
Nome
CNPJ
Voltar
Editar


Início
Sair
Produtos
Pedidos
Perfil
Funcionarios
Fornecedores
CNPJ
Nome
Adicionar
Remover
Editar


Início
Sair
Produtos
Pedidos
Perfil
Funcionarios
Fornecedores
Id
Nome
Cargo
Adicionar
Editar
Remover


Início
Sair
Produtos
Pedidos
Perfil
Funcionarios
Fornecedores
Id
Cliente
Status
Finalizar Pedido