**DECLARAÇÃO DE VISÃO DO PROJETO**

**ERIKA ANDRÉ QUEIROZ**

**MATHEUS HENRIQUE DA SILVA MENDES**

**PABLO MARTINS BALIEIRO**

**RENATO DE MORAES FERREIRA**

**DESENVOLVIMENTO DE UM SITE DE VENDAS *ONLINE* DE PULSEIRAS ARTESANAIS & *HIPPIES* – PRAIA DAS PULSEIRAS**

**SÃO PAULO**

**2025**

# **DECLARAÇÃO DE VISÃO DO PROJETO**

## **1 TÍTULO**

Desenvolvimento de um site de vendas *online* de pulseiras artesanais e *hippies* de nome “Praia das Pulseiras”.

# **2 OBJETIVO**

Desenvolver um site de vendas *online*, onde os usuários possam ver e escolher entre modelos e cores de pulseiras feitas à mão e fazer algum tipo de encomenda caso desejem algo personalizado.

Para desenvolver esse projeto, estima-se levar em média uns quatro meses.

# **3 JUSTIFICATIVA**

Atualmente existem muitos sites de vendas de artigos de bijuterias, porém, nenhum focado em artigos feitos 100% à mão de maneira totalmente artesanal e personalizado ao gosto dos clientes.

Sendo assim, esse site resolverá essa questão e daria maior visibilidade para os produtos de quem faz esse tipo de artesanato.

# **4 INTRODUÇÃO**

Nos últimos anos, o comércio eletrônico se tornou uma das principais formas de compra e venda no mundo, transformando a maneira como adquirimos produtos e serviços. A praticidade, a possibilidade de alcançar consumidores em diferentes regiões e a enorme variedade de itens disponíveis são fatores que impulsionaram essa revolução digital. Dentro deste cenário, o artesanato encontrou seu espaço, permitindo que artesãos expandam seus negócios e alcancem um público ainda maior.

Este trabalho tem como proposta a criação de um site de vendas online chamado "Praia das Pulseiras", voltado para a comercialização de pulseiras artesanais e *hippies*. A escolha desse nicho vem do crescente interesse por produtos exclusivos e personalizados, que carregam consigo um significado especial e um toque de autenticidade. Além disso, a estética *hippie*, com suas cores vibrantes e símbolos espirituais, tem se destacado no mundo da moda, conquistando um público diverso e consciente.

Mais do que um simples canal de vendas, o site busca valorizar o trabalho dos artesãos envolvidos, destacando suas habilidades e histórias. Para isso, serão considerados aspectos como um design intuitivo e responsivo, integração com meios de pagamento seguros e estratégias eficazes de marketing digital para atrair e fidelizar clientes.

Ao longo deste TCC, serão apresentadas todas as etapas do desenvolvimento do site, desde a concepção da ideia até a implementação final, abordando as ferramentas e tecnologias utilizadas. O objetivo é não apenas impulsionar o comércio artesanal *online*, mas também inspirar outros empreendedores a explorarem as oportunidades do e-commerce no setor artesanal.

1. **METODOLOGIA**

* Os softwares que melhor atende as demandas deste projeto, serão:
  + *Visual Studio Code;*
  + *Docker;*
  + *Github.*
* Atendimento aos requisitos funcionais e não funcionais, interface gráfica e demais necessidades do sistema e de infraestrutura em relação à plataforma de *software* e *hardware.*
* **Requisitos Funcionais:**

| CÓDIGO | DESCRIÇÃO |
| --- | --- |
| RF001 | O sistema deve permitir o cadastro de clientes. |
| RF002 | O sistema deve permitir o cadastro de formas de pagamento. |
| RF003 | O sistema deve exibir a lista de produtos disponíveis e também uma lista com os produtos esgotados logo depois do último item disponível. |
| RF004 | O sistema deve categorizar os produtos como “Novidades” e “Mais Antigos”. |
| RF005 | O sistema poderá enviar notificações aos clientes quando um produto novo for enviado, com o nome de “Novidades”. |
| RF006 | O sistema deve permitir que o cliente visualize todos os produtos ou visualize por categorias. |

* **Requisitos Não Funcionais:**

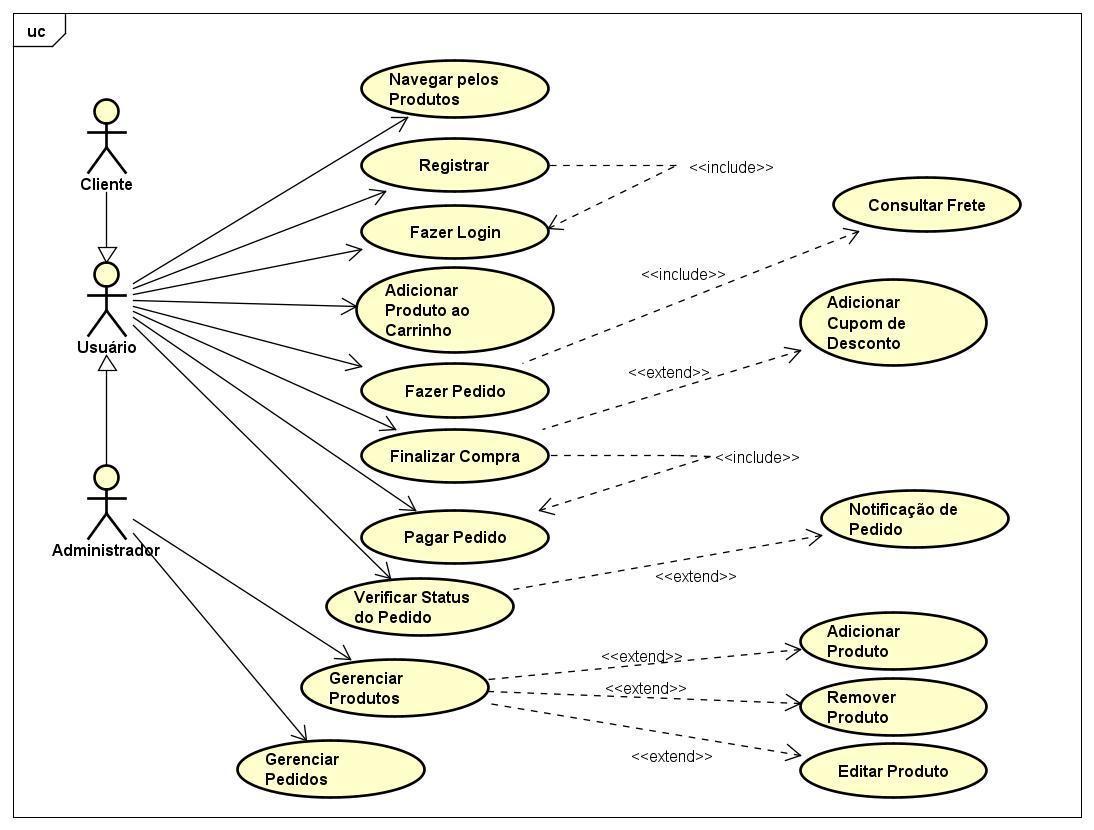
| CÓDIGO | DESCRIÇÃO |
| --- | --- |
| RNF001 | O sistema deve ser responsivo e funcionar em navegadores de dispositivos móveis e desktops. |
| RNF002 | O sistema deve garantir a proteção de dados dos usuários, em conformidade com a LGPD. |
| RNF003 | O tempo de resposta para carregamento das imagens deve ser inferior a 3 segundos. |
| RNF004 | O sistema poderá enviar notificações com, no mínimo, 24 horas depois de o ítem ser postado no sistema. |
| RNF005 | O sistema deve ser compatível com os principais navegadores (Chrome, Firefox, Edge, Safari e Opera) |
| RNF006 | O sistema deve estar disponível 99,9% do tempo (alta disponibilidade) |

* Descrição do software utilizado para atender *à necessidade do projeto.*
  + *Visual Studio Code: editor de código simples, prático e leve para o tamanho do projeto.*
  + *Docker: Configuração do ambiente de desenvolvimento*
  + *Github: Ferramenta de versionamento do código do projeto*
  + *MySQL: Banco de Dados*
* Descrição detalhada da equipe e das partes interessadas:

| NOME | DESCRIÇÃO |
| --- | --- |
| Clientes | Necessidade de fazer suas compras online sem ter a necessidade de ir em alguma loja especializada. |
| Vendedores | Usuários que tem a necessidade de demonstrar seus produtos de maneira rápida e eficaz para o público. |
| Equipe de Desenvolvimento | Responsável pela construção e manutenção do sistema. |
| Administradores do sistema | Equipe de Desenvolvimento. |

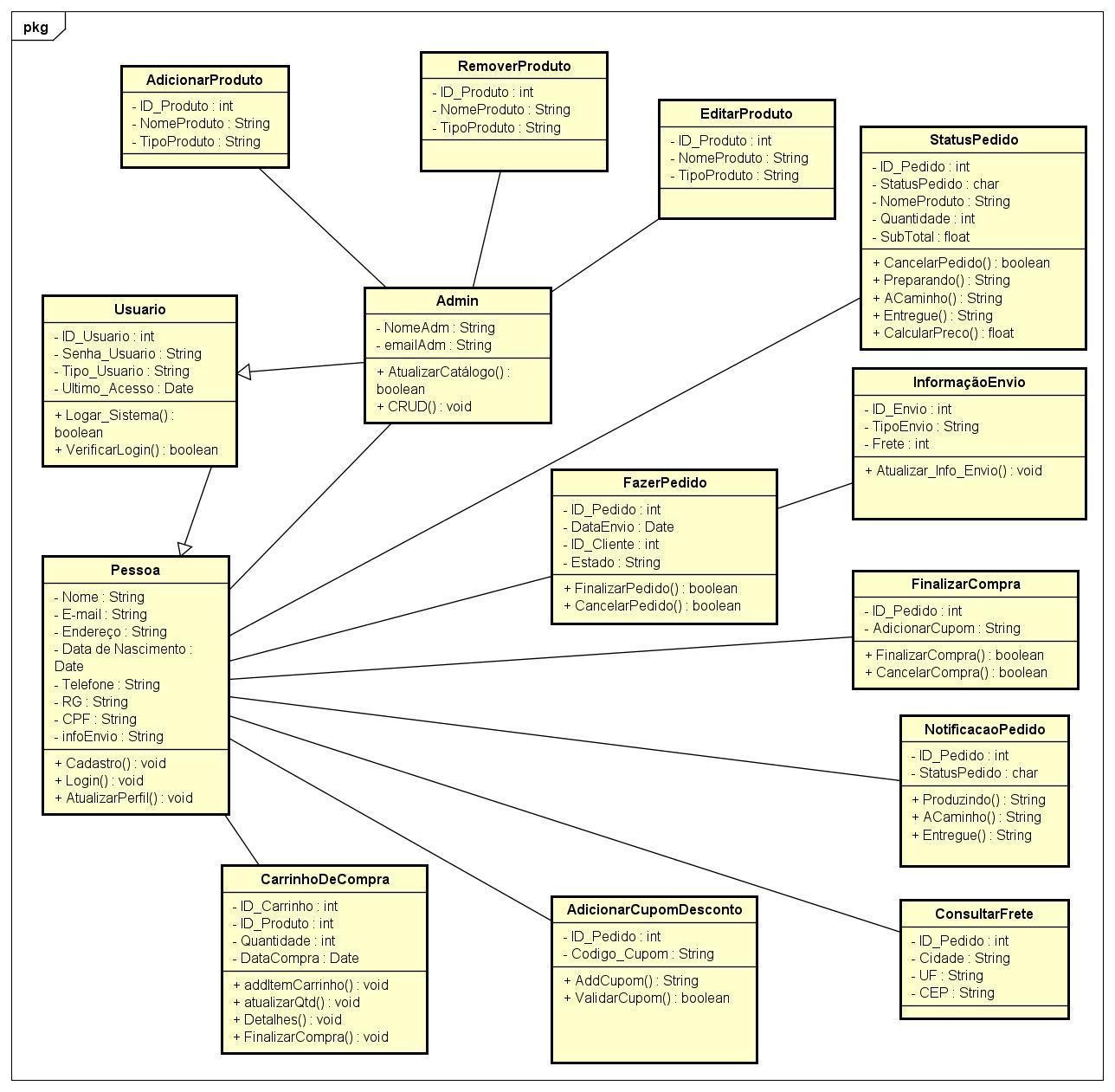
* Descrição dos recursos utilizados para o desenvolvimento do projeto (como: descrição dos equipamentos, técnicas e processos utilizados).
  + Projeto Monolítico;
  + Padrão MVC;
  + Página de apresentação do comercio e um catálogo de produtos;
  + CRUD de produtos para o *keyUser*.
* Nas técnicas e processos utilizados você e seu grupo deverão inserir os **casos de uso** desenvolvidos na agenda 04 de PTCC. Para isso, deverão descrever os casos de uso que desenvolveram, fazerem os ajustes, caso necessário, colarem os gráficos e fazerem as explicações dos processos.

# **Diagrama de Caso de Uso:**



**Fonte: Os autores (2025).**

# **Diagrama de Classe:**



**Fonte: Os autores (2025).**

# **6 DESCRIÇÃO GERAL**

O site permitirá ao usuário a escolha e compra de seus produtos, podendo colocar em um carrinho de compras para finalizar em momento mais oportuno ou então já fazer sua compra de forma imediata.

O site também permitirá ao usuário aplicar cupom de descontos caso haja algum para ser aplicado.

E também o site permitirá que o usuário calcule o frete de seu pedido antes de efetuar sua compra, para ela já ficar ciente do quanto tudo ficará para ele.

O site contará com sistema de pagamento via PIX ou boleto bancário e futuramente com parcelamento em cartão de crédito, facilitando assim as compras dos clientes.

# **7 EQUIPE**

| NOME | FUNÇÃO |
| --- | --- |
| Erika André Queiroz | Team |
| Izadora Bononi Lopes | *Product Owner* |
| Pablo Martins Balieiro | Scrum Master |
| Matheus Henrique da Silva Mendes | Team |
| Renato de Moraes Ferreira | Team |

# **8 PREMISSAS**

O site otimizará o tempo de pessoas que estejam interessadas em adquirir acessórios da moda para usar em seu dia a dia e até mesmo em ocasiões especiais, mas sem ter a necessidade de se deslocar em até alguma loja em shoppings ou feiras de artesanatos locais para tal.

O site também facilitará a divulgação do trabalho de vendedores que fazem esse tipo de trabalho artesanal e que não tem um leque grande de contatos para fazer a divulgação apenas boca-a-boca.

# **9 RESTRIÇÕES**

O sistema será desenvolvido para páginas da Web podendo ser acessado de qualquer dispositivo móvel conectado com a internet.

A comunicação de dados entre cliente e vendedor deve ser segura e atender à LGPD e também proteger os dados bancários dos clientes.

O desenvolvimento deve ser finalizado em um prazo de 120 dias, considerando as iterações de testes.

# **10 ESCOPO EXCLUÍDO**

Desenvolvimento de plataforma *mobile* (Aplicativo) do sistema.

# **11 RISCOS PRELIMINARES**

Perda de membros-chave de qualquer uma das partes interessadas. Dificuldade de incentivar os clientes a se cadastrarem. Riscos associados à privacidade dos clientes e ao cumprimento da LGPD. Possíveis problemas de compatibilidade com diferentes versões de navegadores e dispositivos. O *Product Owner* mudar de ideia e desistir do site.

**12 MODELAGEM DE DADOS**

**12.1 Modelo Conceitual (Diagrama Entidade-Relacionamento)**

## Entidades Principais

### USUÁRIO

* Atributos:
  + id\_usuario (identificador)
  + nome\_usuario
  + email\_usuario
  + senha\_usuario
  + tipo\_usuario (admin, cliente, etc.)
  + status\_usuario (ativo, inativo)
  + datas (criação e atualização)

### CLIENTE

* Atributos:
  + id\_cliente (identificador)
  + nome
  + cpf
  + telefone
  + endereço (descrição + CEP)
  + relacionamento com USUÁRIO

### PRODUTO

* Atributos:
  + id\_produto (identificador)
  + descrição
  + quantidade\_estoque
  + preço
  + data\_atualização
  + relacionamento com USUÁRIO criador (via KEY\_USER)

### 

### PEDIDO

* Atributos:
  + id\_pedido (identificador)
  + data\_pedido
  + status
  + valor\_total
  + relacionamento com CLIENTE

### ITEM\_PEDIDO (entidade associativa)

* Atributos:
  + id\_item (identificador)
  + quantidade
  + preço\_unitário
  + subtotal
  + relacionamentos com PEDIDO e PRODUTO

### KEY\_USER (entidade para controle de acesso)

* Atributos:
  + id\_key
  + data\_criação
  + relacionamento com USUÁRIO

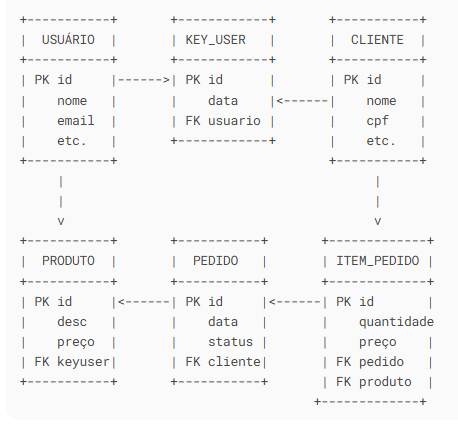
## Relacionamentos

1. USUÁRIO (1) — cria — (N) PRODUTO (via KEY\_USER)
2. USUÁRIO (1) — tem — (1) CLIENTE (1:1 opcional)
3. CLIENTE (1) — faz — (N) PEDIDO
4. PEDIDO (1) — contém — (N) ITEM\_PEDIDO
5. PRODUTO (1) — está em — (N) ITEM\_PEDIDO

## 

## 

**12. 2 Diagrama em formato textual:**



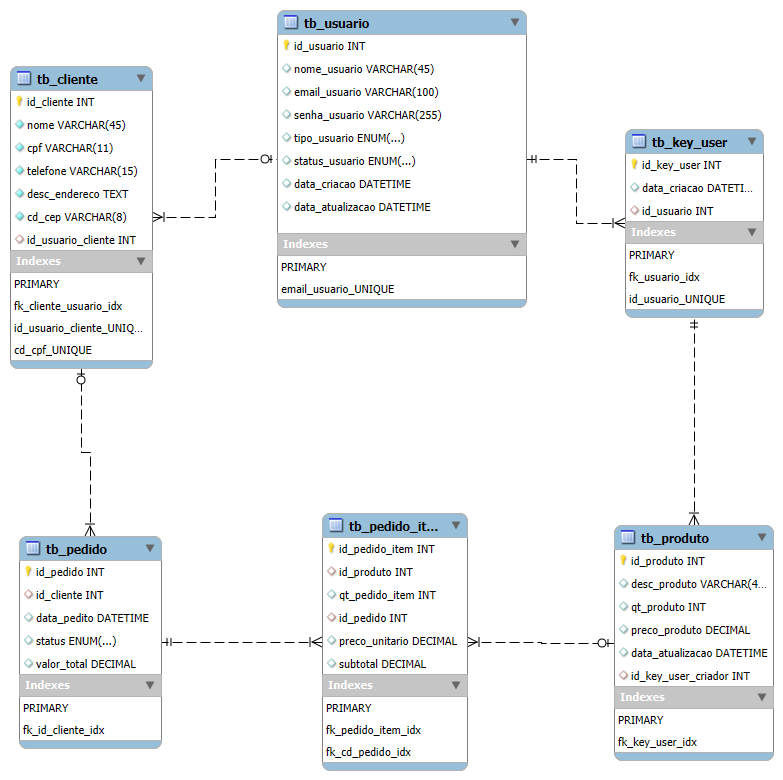
# 

# 

**12.3 Diagrama Modelo Conceitual**

# 

**12.4 Modelo Lógico**



**12.5 Modelo Físico, Script sql básico**

**-- Criação do banco de dados (caso não exista)**

CREATE DATABASE IF NOT EXISTS sistema\_pedidos;

USE sistema\_pedidos;

**-- Tabela de Usuários**

CREATE TABLE tb\_usuario (

id\_usuario INT AUTO\_INCREMENT PRIMARY KEY,

nome\_usuario VARCHAR(45) NOT NULL,

email\_usuario VARCHAR(100) NOT NULL UNIQUE,

senha\_usuario VARCHAR(255) NOT NULL,

tipo\_usuario ENUM('admin', 'vendedor', 'cliente') NOT NULL,

status\_usuario ENUM('ativo', 'inativo', 'pendente') NOT NULL DEFAULT 'pendente',

data\_criacao DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP,

data\_atualizacao DATETIME ON UPDATE CURRENT\_TIMESTAMP

);

**-- Tabela de Chaves de Usuário**

CREATE TABLE tb\_key\_user (

id\_key\_user INT AUTO\_INCREMENT PRIMARY KEY,

data\_criacao DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP,

id\_usuario INT NOT NULL,

UNIQUE KEY (id\_usuario),

CONSTRAINT fk\_key\_user\_usuario FOREIGN KEY (id\_usuario)

REFERENCES tb\_usuario (id\_usuario)

ON DELETE CASCADE

);

**-- Tabela de Clientes**

CREATE TABLE tb\_cliente (

id\_cliente INT AUTO\_INCREMENT PRIMARY KEY,

nome VARCHAR(45) NOT NULL,

cpf VARCHAR(11) NOT NULL UNIQUE,

telefone VARCHAR(15),

desc\_endereco TEXT,

cd\_cep VARCHAR(8),

id\_usuario\_cliente INT NOT NULL UNIQUE,

CONSTRAINT fk\_cliente\_usuario FOREIGN KEY (id\_usuario\_cliente)

REFERENCES tb\_usuario (id\_usuario)

ON DELETE CASCADE

);

**-- Tabela de Produtos**

CREATE TABLE tb\_produto (

id\_produto INT AUTO\_INCREMENT PRIMARY KEY,

desc\_produto VARCHAR(45) NOT NULL,

qt\_produto INT NOT NULL DEFAULT 0,

preco\_produto DECIMAL(10,2) NOT NULL,

data\_atualizacao DATETIME ON UPDATE CURRENT\_TIMESTAMP,

id\_key\_user\_criador INT NOT NULL,

CONSTRAINT fk\_produto\_key\_user FOREIGN KEY (id\_key\_user\_criador)

REFERENCES tb\_key\_user (id\_key\_user)

ON DELETE RESTRICT

);

**-- Tabela de Pedidos**

CREATE TABLE tb\_pedido (

id\_pedido INT AUTO\_INCREMENT PRIMARY KEY,

id\_cliente INT NOT NULL,

data\_pedido DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP,

status ENUM('pendente', 'processando', 'enviado', 'entregue', 'cancelado') NOT NULL DEFAULT 'pendente',

valor\_total DECIMAL(10,2) NOT NULL DEFAULT 0.00,

CONSTRAINT fk\_pedido\_cliente FOREIGN KEY (id\_cliente)

REFERENCES tb\_cliente (id\_cliente)

ON DELETE RESTRICT

);

**-- Tabela de Itens de Pedido**

CREATE TABLE tb\_pedido\_item (

id\_pedido\_item INT AUTO\_INCREMENT PRIMARY KEY,

id\_produto INT NOT NULL,

qt\_pedido\_item INT NOT NULL DEFAULT 1,

id\_pedido INT NOT NULL,

preco\_unitario DECIMAL(10,2) NOT NULL,

subtotal DECIMAL(10,2) NOT NULL,

CONSTRAINT fk\_item\_pedido FOREIGN KEY (id\_pedido)

REFERENCES tb\_pedido (id\_pedido)

ON DELETE CASCADE,

CONSTRAINT fk\_item\_produto FOREIGN KEY (id\_produto)

REFERENCES tb\_produto (id\_produto)

ON DELETE RESTRICT

);

**13 – DESIGN DE INTERFACE**

**13.1 – Layout de Telas do sistema - HomeInterface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.  
  
13.2 – Loja**

**Interface gráfica do usuário, Aplicativo, Site, PowerPoint

O conteúdo gerado por IA pode estar incorreto.**

**13.3 - Contato**

**Interface gráfica do usuário, Site

O conteúdo gerado por IA pode estar incorreto.**

**13.4 – Registre-se**

**Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.**

**13.5 - Login**

**Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.**

**13.6 – Painel de Controle - Cliente**

**Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.**

**13.7 – Página do Produto**

**Interface gráfica do usuário, Texto

O conteúdo gerado por IA pode estar incorreto.**

**13.8 – Checkout de Compras (cliente)**

**Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.**

**14- Arquitetura MVC**

**- 14.1 Camada Models**

**- 14.1.1 Classe Cliente**

<?php

class Cliente {

// Atributos

private $id\_cliente;

private $cpf;

private $telefone;

private $desc\_endereco;

private $cep;

private $id\_pedido;

private $pedidos;

// Construtor

public function \_\_construct($id\_cliente = null, $cpf = null, $telefone = null, $desc\_endereco = null, $cep = null) {

$this->id\_cliente = $id\_cliente;

$this->cpf = $cpf;

$this->telefone = $telefone;

$this->desc\_endereco = $desc\_endereco;

$this->cep = $cep;

$this->pedidos = array();

}

// Getters e Setters

public function getId\_cliente() {

return $this->id\_cliente;

}

public function setId\_cliente($id\_cliente) {

$this->id\_cliente = $id\_cliente;

}

public function getCpf() {

return $this->cpf;

}

public function setCpf($cpf) {

$this->cpf = $cpf;

}

public function getTelefone() {

return $this->telefone;

}

public function setTelefone($telefone) {

$this->telefone = $telefone;

}

public function getDesc\_endereco() {

return $this->desc\_endereco;

}

public function setDesc\_endereco($desc\_endereco) {

$this->desc\_endereco = $desc\_endereco;

}

public function getCep() {

return $this->cep;

}

public function setCep($cep) {

$this->cep = $cep;

}

public function getId\_pedido() {

return $this->id\_pedido;

}

public function setId\_pedido($id\_pedido) {

$this->id\_pedido = $id\_pedido;

}

public function getPedidos() {

return $this->pedidos;

}

public function setPedidos($pedidos) {

$this->pedidos = $pedidos;

}

// Métodos

public function criar\_Pedido($pedido) {

// Implementação será feita posteriormente

}

public function get\_Pedido\_Atual($pedido) {

// Implementação será feita posteriormente

}

public function get\_Pedido\_Array() {

// Implementação será feita posteriormente

return array();

}

}

**- 14.1.2 Classe Admin**

<?php

class Admin {

// Atributos

private $id\_usuario;

private $nome\_usuario;

private $email\_usuario;

private $senha\_usuario;

private $tipo\_usuario;

private $status\_usuario;

private $data\_criacao;

private $data\_atualizacao;

private $fazer\_login;

private $fazer\_logout;

// Construtor

public function \_\_construct($id\_usuario = null, $nome\_usuario = null, $email\_usuario = null,

$senha\_usuario = null, $tipo\_usuario = "KEY\_USER") {

$this->id\_usuario = $id\_usuario;

$this->nome\_usuario = $nome\_usuario;

$this->email\_usuario = $email\_usuario;

$this->senha\_usuario = $senha\_usuario;

$this->tipo\_usuario = $tipo\_usuario;

$this->status\_usuario = "ATIVO";

$this->data\_criacao = date('Y-m-d H:i:s');

$this->data\_atualizacao = date('Y-m-d H:i:s');

$this->fazer\_login = false;

$this->fazer\_logout = false;

}

// Getters e Setters

public function getId\_usuario() {

return $this->id\_usuario;

}

public function setId\_usuario($id\_usuario) {

$this->id\_usuario = $id\_usuario;

}

public function getNome\_usuario() {

return $this->nome\_usuario;

}

public function setNome\_usuario($nome\_usuario) {

$this->nome\_usuario = $nome\_usuario;

}

public function getEmail\_usuario() {

return $this->email\_usuario;

}

public function setEmail\_usuario($email\_usuario) {

$this->email\_usuario = $email\_usuario;

}

public function getSenha\_usuario() {

return $this->senha\_usuario;

}

public function setSenha\_usuario($senha\_usuario) {

$this->senha\_usuario = $senha\_usuario;

}

public function getTipo\_usuario() {

return $this->tipo\_usuario;

}

public function setTipo\_usuario($tipo\_usuario) {

$this->tipo\_usuario = $tipo\_usuario;

}

public function getStatus\_usuario() {

return $this->status\_usuario;

}

public function setStatus\_usuario($status\_usuario) {

$this->status\_usuario = $status\_usuario;

}

public function getData\_criacao() {

return $this->data\_criacao;

}

public function setData\_criacao($data\_criacao) {

$this->data\_criacao = $data\_criacao;

}

public function getData\_atualizacao() {

return $this->data\_atualizacao;

}

public function setData\_atualizacao($data\_atualizacao) {

$this->data\_atualizacao = $data\_atualizacao;

}

public function getFazer\_login() {

return $this->fazer\_login;

}

public function setFazer\_login($fazer\_login) {

$this->fazer\_login = $fazer\_login;

}

public function getFazer\_logout() {

return $this->fazer\_logout;

}

public function setFazer\_logout($fazer\_logout) {

$this->fazer\_logout = $fazer\_logout;

}

// Métodos

public function gerenciarProdutos() {

// Implementação será feita posteriormente

}

public function gerenciarPedidos() {

// Implementação será feita posteriormente

}

public function gerenciarClientes() {

// Implementação será feita posteriormente

}

}

**- 14.1.3 Classe Pedido**

<?php

class Pedido {

// Atributos

private $id;

private $status;

private $valor\_total;

private $data;

private $itensPedido;

// Construtor

public function \_\_construct($id = null, $status = "PENDENTE", $valor\_total = 0.0) {

$this->id = $id;

$this->status = $status;

$this->valor\_total = $valor\_total;

$this->data = date('Y-m-d H:i:s');

$this->itensPedido = array();

}

// Getters e Setters

public function getId() {

return $this->id;

}

public function setId($id) {

$this->id = $id;

}

public function getStatus() {

return $this->status;

}

public function setStatus($status) {

$this->status = $status;

}

public function getValor\_total() {

return $this->valor\_total;

}

public function setValor\_total($valor\_total) {

$this->valor\_total = $valor\_total;

}

public function getData() {

return $this->data;

}

public function setData($data) {

$this->data = $data;

}

public function getItensPedido() {

return $this->itensPedido;

}

public function setItensPedido($itensPedido) {

$this->itensPedido = $itensPedido;

}

// Métodos

public function adicionar\_produto\_pedido($itemPedido) {

// Implementação será feita posteriormente

}

public function remover\_produto\_pedido($itemPedido) {

// Implementação será feita posteriormente

}

public function calcular\_total() {

// Implementação será feita posteriormente

return 0;

}

}

// Classe ItemPedido conforme mostrado no diagrama

class ItemPedido {

// Atributos

private $id\_itemPedido;

private $id\_produto;

private $quantidade;

private $preco\_unitario;

private $subtotal;

// Construtor

public function \_\_construct($id\_itemPedido = null, $id\_produto = null, $quantidade = 0, $preco\_unitario = 0.0) {

$this->id\_itemPedido = $id\_itemPedido;

$this->id\_produto = $id\_produto;

$this->quantidade = $quantidade;

$this->preco\_unitario = $preco\_unitario;

$this->subtotal = $quantidade \* $preco\_unitario;

}

// Getters e Setters

public function getId\_itemPedido() {

return $this->id\_itemPedido;

}

public function setId\_itemPedido($id\_itemPedido) {

$this->id\_itemPedido = $id\_itemPedido;

}

public function getId\_produto() {

return $this->id\_produto;

}

public function setId\_produto($id\_produto) {

$this->id\_produto = $id\_produto;

}

public function getQuantidade() {

return $this->quantidade;

}

public function setQuantidade($quantidade) {

$this->quantidade = $quantidade;

$this->atualizarSubtotal();

}

public function getPreco\_unitario() {

return $this->preco\_unitario;

}

public function setPreco\_unitario($preco\_unitario) {

$this->preco\_unitario = $preco\_unitario;

$this->atualizarSubtotal();

}

public function getSubtotal() {

return $this->subtotal;

}

// Métodos

public function atualizarSubtotal() {

$this->subtotal = $this->quantidade \* $this->preco\_unitario;

}

public function validarEstoque() {

// Implementação será feita posteriormente

return true;

}}

**- 14.1.4 Classe Produto**

<?php

class Produto {

// Atributos

private $id;

private $descricao;

private $id\_produto\_pai;

private $preco;

private $data\_atualizacao;

private $imagem\_url;

private $id\_loja\_pai\_criador;

// Construtor

public function \_\_construct($id = null, $descricao = null, $preco = 0.0, $imagem\_url = null) {

$this->id = $id;

$this->descricao = $descricao;

$this->preco = $preco;

$this->data\_atualizacao = date('Y-m-d H:i:s');

$this->imagem\_url = $imagem\_url;

}

// Getters e Setters

public function getId() {

return $this->id;

}

public function setId($id) {

$this->id = $id;

}

public function getDescricao() {

return $this->descricao;

}

public function setDescricao($descricao) {

$this->descricao = $descricao;

}

public function getId\_produto\_pai() {

return $this->id\_produto\_pai;

}

public function setId\_produto\_pai($id\_produto\_pai) {

$this->id\_produto\_pai = $id\_produto\_pai;

}

public function getPreco() {

return $this->preco;

}

public function setPreco($preco) {

$this->preco = $preco;

}

public function getData\_atualizacao() {

return $this->data\_atualizacao;

}

public function setData\_atualizacao($data\_atualizacao) {

$this->data\_atualizacao = $data\_atualizacao;

}

public function getImagem\_url() {

return $this->imagem\_url;

}

public function setImagem\_url($imagem\_url) {

$this->imagem\_url = $imagem\_url;

}

public function getId\_loja\_pai\_criador() {

return $this->id\_loja\_pai\_criador;

}

public function setId\_loja\_pai\_criador($id\_loja\_pai\_criador) {

$this->id\_loja\_pai\_criador = $id\_loja\_pai\_criador;

}

// Métodos

public function atualizarEstoque() {

// Implementação será feita posteriormente

}

}

// Classe IPersistencia (interface) transformada em classe conforme solicitado

class IPersistencia {

// Atributos

private $field\_type;

private $criar;

private $deletar;

private $atualizar;

private $listar;

// Getters e Setters

public function getField\_type() {

return $this->field\_type;

}

public function setField\_type($field\_type) {

$this->field\_type = $field\_type;

}

public function getCriar() {

return $this->criar;

}

public function setCriar($criar) {

$this->criar = $criar;

}

public function getDeletar() {

return $this->deletar;

}

public function setDeletar($deletar) {

$this->deletar = $deletar;

}

public function getAtualizar() {

return $this->atualizar;

}

public function setAtualizar($atualizar) {

$this->atualizar = $atualizar;

}

public function getListar() {

return $this->listar;

}

public function setListar($listar) {

$this->listar = $listar;

}

}