

# ESPECIFICAÇÃO DE REQUISITOS DE SOFTWARE (ERS)

## 1. Introdução

Este documento detalha os requisitos do sistema **E-Commerce Web**, especificando comportamentos esperados, fluxos de uso, validações, estrutura de dados e modelos UML (descritos em texto).

Serve como base técnica para desenvolvedores e testadores responsáveis pela implementação e validação do sistema.

O sistema **E-Commerce Web** tem como objetivo disponibilizar uma plataforma digital de vendas que permita aos usuários navegar por produtos, adicioná-los ao carrinho, realizar compras e acompanhar pedidos.

Além disso, o sistema possibilita que administradores gerenciem produtos, categorias, estoques e pedidos de forma centralizada.

## 2. Arquitetura Geral do Sistema

### 2.1 Arquitetura de Camadas

O sistema será dividido em:

- **Camada de Apresentação (Front-end):**

Responsável pela interface de interação com o usuário. Desenvolvida utilizando HTML5, CSS3 e JavaScript, garantindo uma navegação intuitiva, responsiva e compatível com diferentes dispositivos.

Essa camada consumirá os serviços disponibilizados pela API do back-end, exibindo produtos, carrinho, pedidos e status de compras em tempo real.

- **Camada de Aplicação (Back-end):** Responsável pelas regras de negócio e controle dos fluxos do sistema. Implementada em Python, utilizando o framework Django.

Essa camada gerencia as funcionalidades de autenticação, cadastro de usuários, controle de produtos, processamento de pedidos e registro de pagamentos.

- **Camada de Persistência:** Responsável pelo armazenamento e recuperação dos dados. Será utilizado o banco de dados relacional PostgreSQL,

garantindo integridade referencial e consistência das informações do sistema.

- Camada de Segurança: Implementada com autenticação e autorização via JWT (JSON Web Token), criptografia de senhas utilizando bcrypt, e controle de acesso baseado em perfis de usuário (Administrador e Cliente).

### 3. Casos de Uso Detalhados

#### UC01 – Cadastrar Usuário

- Ator Primário: Visitante
- Pré-condição: Usuário não autenticado.
- Fluxo Principal:
  1. O visitante acessa tela de cadastro
  2. Informa: Nome completo, e-mail, senha e confirmação de senha
  3. Clica em “Cadastrar”
  4. O sistema valida os dados e cria o registro do usuário no banco de dados.
  5. O sistema exibe mensagem de sucesso e redireciona o usuário para a tela de login.
- Fluxo Alternativo (E-mail já cadastrado):
  - Sistema exibe erro: “E-mail já cadastrado no sistema.”
- Pós-condição: Usuário registrado com perfil padrão “Cliente”

#### UC02 – Realizar login

- Ator Primário: Usuário (Cliente ou Administrador)
- Pré-condição: Usuário deve possuir cadastro ativo
- Fluxo Principal:
  1. Usuário acessa tela de login
  2. Informa e-mail e senha
  3. O sistema valida as credenciais
  4. Confirma
  5. Caso estejam corretas, o sistema gera um token JWT e redireciona o usuário para a página inicial.

#### UC03 – Cadastrar Produto

- Ator Primário: Administrador
- Pré-condição: Usuário autenticado com perfil de administrador
- Fluxo Principal:

1. O Administrador acessa o painel administrativo
2. Seleciona a opção “Cadastrar Produto”
3. Informa nome, descrição, preço, categoria, quantidade em estoque e imagem.
4. Clica em “Salvar”.
5. O sistema valida e salva as informações no banco de dados
6. O sistema exibe a mensagem: “Produto cadastrado com sucesso”

Fluxo alternativo (Campos obrigatórios ausentes)

- O sistema exibe erro: “Erro, preencha campos obrigatórios.”

Pós-condição: Produto registrado e disponível no catálogo

#### UC04 – Adicionar produto ao carrinho

- Ator primário: Cliente
- Pré-condição: Usuário autenticado e produto disponível em estoque
- Fluxo Principal:
  1. O cliente acessa a página de um produto.
  2. Clica no botão “Adicionar ao carrinho”.
  3. O sistema adiciona o item ao carrinho do usuário
  4. O carrinho é atualizado com o novo produto e valor total

Fluxo alternativo: (Produto sem estoque)

- O sistema exibe a mensagem “Produto indisponível no momento”

Pós-condição: Produto adicionado ao carrinho do usuário.

#### UC05 – Finalizar compra

Ator primário: Cliente

Pré-condição: Usuário autenticado com produtos no carrinho

Fluxo principal:

1. O cliente acessa o carrinho e revisa os itens.
2. Clica em “Finalizar compra”
3. Informa endereço de entrega e forma de pagamento
4. O sistema processa o pedido e registra o pagamento
5. O sistema atualiza o estoque e exibe mensagem: “Pedido realizado com sucesso.”

Fluxo alternativo: (Pagamento não autorizado):

- O sistema exibe “Erro Pagamento recusado, tente novamente.”

Pós-condição: Pedido registrado com status “Em processamento.”

UC06 – Gerenciar pedidos

Ator primário: Administrador

Pré-condição: Usuário autenticado com perfil de administrador.

Fluxo principal:

1. O administrador acessa o painel de controle.
2. Visualiza a lista de pedidos realizados.
3. Pode atualizar o status do pedido (Em processamento, Enviado, Entregue).
4. O sistema salva as alterações e atualiza o histórico do cliente.

Pós-condição: Pedido atualizado com novo status.

UC07 – Consultar Histórico de Pedidos

Ator-primário: Cliente

Pré-condição: Usuário autenticado com pedidos registrados.

Fluxo-principal:

1. O cliente acessa seu perfil.
2. Seleciona “Meus pedidos”.
3. O sistema exibe a lista de compras anteriores com datas, valores e status.

Pós-condição: Histórico exibido corretamente para o usuário.

#### 4. Modelos UML (descritos em texto)

##### 4.1 Diagrama de Casos de Uso (simplificado – textual)

Ator: Visitante

→ Criar conta

→ Visualizar Produtos

→ Pesquisar Produtos

Ator: Cliente

→ Realizar Login

→ Adicionar Produtos no carrinho

- Finalizar Compras
- Consultar Histórico de pedidos
- Editar Perfil

Ator: Administrador

- Cadastrar Produto
- Editar Produto
- Excluir Produto
- Gerenciar Categorias
- Gerenciar Pedidos
- Consultar Relatórios de vendas

#### 4.2 Diagrama de Classes (descrição)

Classe: Usuário

id: Long

nome: String

E-mail: String

telefone: String

Tipo: Enum (CLIENTE, ADMINISTRADOR)

data\_cadastro: Date

Classe: Produto

id: Long

descricao:String

Preco: Double

Estoque: int

Imagem\_url: String

categoria: Categoria

Classe: Categoria

id: Long

nome: String

Descricao: String

Classe: Carrinho

id: Long

usuario: Usuário

objetivo: String

itens: List<ItemCarrinho>

Valor\_total: Double

Classe: ItemCarrinho

id: Long

produto: Produto

quantidade: int

subtotal: Double

Classe: Pedido

Id: Long

usuario: Usuário

data\_pedido: Date

status: Enum (EM\_PROCESSAMENTO, ENVIADO, ENTREGUE, CANCELADO)

valor\_total: Double

itens: List<ItemPedido>

Classe: ItemPedido

id: Long

produto: Produto  
quantidade: int  
preco\_unitario: Double  
subtotal: Double

Classe: Pagamento  
id: Long  
pedido: Pedido  
valor: Double  
forma\_pagamento: Enum (CARTAO, PIX, BOLETO)  
status: Enum (PAGO, PENDENTE, RECUSADO)  
data\_pagamento: Date

## 5. Estrutura do Banco de Dados

### TABELA USUARIO

- id (PK)
- nome
- email
- senha
- tipo (CLIENTE, ADMINISTRADOR)
- data\_cadastro

### TABELA CATEGORIA

- id (PK)
- nome
- descricao

### TABELA PRODUTO

- id (PK)
- nome
- descricao
- preco
- estoque
- imagem\_url
- categoria\_id (FK → CATEGORIA .id)

#### TABELA CARRINHO

- id (PK)
- usuario\_id(FK → CATEGORIA.id)
- valor\_total

#### TABELA ITEM\_CARRINHO

- id (PK)
- carrinho\_id(FK → CARRINHO.id)
- produto\_id (FK → PRODUTO.id)
- quantidade
- subtotal

#### TABELA PEDIDO

- id (PK)
- usuario\_id(FK → USUARIO.id)
- data\_pedido
- status (EM\_PROCESSAMENTO, ENVIADO, ENTREGUE, CANCELADO)
- valor\_total

#### TABELA ITEM\_PEDIDO



- id (PK)
- pedido\_id(FK → PEDIDO.id)
- pedido\_id(FK → PRODUTO.id)
- quantidade
- precounitario
- subtotal

#### TABELA DE PAGAMENTO





- id (PK)
- pedido\_id(FK → PEDIDO.id)
- valor
- forma\_pagamento (CARTAO, PIX, BOLETO)
- status (PAGO, PENDENTE, RECUSADO)
- data\_pagamento

#### 6. Regras de Validação

- E-mail deve ser único e válido (formato padrão: [usuario@dominio.com](mailto:usuario@dominio.com)).
- Senha deve conter no mínimo 8 caracteres, incluindo letras e números.
- Produto deve ter preço maior que zero e quantidade em estoque igual ou superior a zero.
- Categoria deve possuir nome único.
- Usuário só poderá acessar áreas restritas se estiver autenticado via token JWT.
- Carrinho não pode estar vazio para finalizar uma compra.
- Pedido só pode ser criado se o pagamento for iniciado com sucesso.
- Estoque deve ser automaticamente reduzido após a confirmação do pedido.
- Pagamento deve ter valor igual ao total do pedido.
- Pedidos cancelados devem restaurar o estoque dos produtos envolvidos.
- Data de pagamento e data de pedido não podem ser retroativas.
- Campos obrigatórios (nome, e-mail, senha, produto, preço, etc.) devem ser validados antes do envio ao banco de dados.

- Administrador é o único perfil autorizado a cadastrar, editar ou excluir produtos e categorias.

## 7. Requisitos de Interface

- O sistema deve possuir um design moderno e responsivo, adaptando-se a diferentes tamanhos de tela (desktop, tablet e mobile).
- O menu principal deve conter as opções: *Início*, *Produtos*, *Carrinho*, *Meus Pedidos* e *Perfil*.
- As cores predominantes devem seguir uma paleta leve e contrastante, garantindo boa legibilidade e aparência profissional.
- Ícones intuitivos devem representar ações como:
  -  Adicionar ao carrinho
  -  Visualizar produto
  -  Editar (área administrativa)
  -  Excluir (área administrativa)
- Cada ação deve gerar feedback visual (mensagens de sucesso, erro ou alerta) exibido em tempo real na interface.
- As telas devem possuir componentes padronizados, como botões, campos de texto e modais, com comportamento consistente em todas as páginas.
- A página inicial deve exibir uma listagem de produtos com imagem, nome, preço e botão “Adicionar ao Carrinho”.
- A página de produto deve exibir detalhes completos (descrição, estoque disponível, preço e imagem ampliada).
- O carrinho de compras deve apresentar a lista de produtos adicionados, quantidades, subtotais e valor total.
- A tela de checkout deve permitir informar endereço e forma de pagamento (cartão, boleto ou PIX).
- A área administrativa deve ter um painel com cards que exibam:
  - Total de produtos cadastrados
  - Pedidos pendentes
  - Pedidos entregues
  - Faturamento total
- O rodapé deve conter informações institucionais e de contato, além de links para redes sociais.

## 8. Requisitos de API REST

### 8.1 Endpoints Principais

## Usuários

Post/api/usuario

Cadastro de novo usuário (cliente)

json

```
{  
  "nome": "João Silva",  
  "email": "joao@email.com",  
  "senha": "senha123"  
}
```

POST/api/login

Autenticação de usuário.

Json

```
{  
  "email": "joao@email.com",  
  "senha": "senha123"  
}
```

Retorno: token JWT para autenticação nas demais rotas

## Produtos

GET/api/produtos

Retorna lista de produtos disponíveis.

POST/api/produtos (somente para administrador)

Cadastra novo produto.

Json

```
{  
  "nome": "Camisa Polo",  
  "descricao": "Camisa 100% algodão",  
  "preco": 79.90,  
  "estoque": 50,
```

```
"categoriaId": 2,  
"imagem_url": "https://site.com/imagens/camisa.jpg"  
}
```

PUT/api/produtos/{id} (somente administrador)

Atualiza informações de um produto.

DELETE/api/produtos/{id} (somente administrador)

Remove um produto do catálogo

Carrinho

POST/api/carrinho/adicionar

Adiciona um produto ao carrinho de cliente.

Json

```
{  
  "produtoId": 5,  
  "quantidade": 2  
}
```

GET/api/carrinho

Retorna os itens do carrinho atual do cliente autenticado.

DELETE/api/carrinho/remover/{itemId}

Remove um item do carrinho.

Pedidos

POST/api/pedidos

Finaliza a compra e cria um pedido.

Json

```
{  
  "endereco_entrega": "Rua das Flores, 123",
```

```
"forma_pagamento": "CARTAO"
```

```
}
```

GET/api/pedidos

Retorna o histórico do pedido do cliente autenticado.

PUT/api/pedidos/{id}/status (somente administrador)

Atualiza um status de um pedido (ex: ENVIADO, ENTREGUE, CANCELADO)

Pagamentos

POST/api/pagamentos

Registra pagamentos de um pedido.

Json

```
{
```

```
"pedidoid": 12,
```

```
"valor": 159.80,
```

```
"forma_pagamento": "PIX"
```

```
}
```

GET/api/pagamentos/{id}

Consulta o status do pagamento de um pedido.

## 8.2 Autenticação

- Todas as rotas protegidas exigem envio de token JWT no cabeçalho da requisição:

Autorization : Bearer<token>

## 8.3 Documentação automática

- A API será documentada automaticamente com o Django REST Framework (DRF) e poderá ser visualizada em:

/api/docs/