Levantamento de Requisitos - E-commerce DealHub

1. Visão Geral

O **DealHub** será um e-commerce inspirado no modelo da Kabum, oferecendo uma experiência fluida para compra de produtos eletrônicos. A plataforma contará com um design moderno, funcional e seguro, garantindo que clientes possam navegar, escolher produtos, adicionar ao carrinho, realizar compras e gerenciar suas contas de forma eficiente.

2. Requisitos Funcionais

2.1. Funcionalidades para Usuários (Clientes)

- **Home**:
    - Apresentação de banners promocionais.
    - Produtos em destaque e recomendados.
    - Links rápidos para categorias populares.
- **Cadastro e Login**:
    - Cadastro de usuários com nome, e-mail, CPF, telefone e senha.
    - Login via e-mail e senha.
    - Recuperação de senha por e-mail.
    - Segurança via Spring Security e criptografia de senhas (BCrypt).
- **Catálogo de Produtos**:
    - Exibição de produtos organizados por categoria.
    - Sistema de filtros (por preço, marca, categoria, avaliações).
    - Pesquisa por nome e características do produto.
    - Paginação dos resultados.
- **Página de Produto**:
    - Imagens do produto.
    - Nome, preço, descrição e especificações técnicas.
    - Avaliações e comentários de usuários.
    - Botão "Adicionar ao Carrinho".
- **Carrinho de Compras**:
    - Listagem de produtos adicionados.
    - Atualização da quantidade de produtos.

    - Cálculo automático do total da compra.
    - Opção para remover itens.
    - Botão "Finalizar Compra".
- **Checkout e Pagamento**:
    - Cadastro de endereços de entrega.
    - Escolha da forma de pagamento (Cartão de Crédito, Boleto, PIX).
    - Integração com gateways de pagamento.
    - Cálculo de frete baseado no CEP.
    - Confirmação do pedido com número de rastreio.
- **Área do Cliente**:
    - Visualização de pedidos e status de entrega.
    - Alteração de dados cadastrais.
    - Gerenciamento de endereços e formas de pagamento salvas.
- **Contato e Suporte**:
    - Formulário de contato para dúvidas e reclamações.
    - Integração com WhatsApp ou chat ao vivo.

2.2. Funcionalidades Administrativas

- **Dashboard**:
    - Gráficos de vendas e métricas do sistema.
- **Gerenciamento de Produtos**:
    - Cadastro, edição e exclusão de produtos.
    - Upload de imagens.
    - Definição de preços, estoque e promoções.
- **Gerenciamento de Categorias**:
    - Cadastro e edição de categorias e subcategorias.
- **Gerenciamento de Pedidos**:
    - Listagem de pedidos com status.
    - Alteração manual do status do pedido (ex: Enviado, Cancelado).
    - Envio de notificações aos clientes sobre mudanças no pedido.
- **Gestão de Usuários**:

- Listagem de clientes cadastrados.
- Definição de permissões para administradores.

---

3. Requisitos Não Funcionais

3.1. Tecnologia e Infraestrutura

- Backend: **Spring Boot (Java 21) + Spring Security + Spring Data JPA**
- Frontend: **Thymeleaf + Bootstrap/Tailwind CSS**
- Banco de Dados: **PostgreSQL**
- Infraestrutura: **Docker**
- Segurança:
  - Autenticação e autorização via Spring Security.
  - Criptografia de senhas (BCrypt).
  - Proteção contra ataques CSRF e XSS.

---

4. Casos de Uso Principais

Aqui estão alguns dos principais fluxos de uso:

4.1. Fluxo de Compra

1. Usuário acessa o site e navega pelos produtos.
2. Adiciona itens ao carrinho.
3. Acessa o carrinho e revisa os produtos.
4. Clica em "Finalizar Compra" e faz login (se necessário).
5. Escolhe a forma de pagamento e insere os dados.
6. Confirma a compra e recebe número de rastreio.

4.2. Fluxo de Cadastro e Login

1. Usuário acessa a página de login.
2. Caso não tenha conta, preenche formulário de cadastro.
3. Sistema valida dados e armazena usuário.
4. Usuário recebe e-mail de confirmação.

5.Modelagem do Banco de Dados (Entidades Principais)

Usuário:

Id long valor único autoincrementado não nulo

Nome string max 100 caracteres não nulo

Email string max 100 caracteres não nulo

Senha string 255 caracteres não nulo

Cpf string 11caracteres

Telefone string 11 caracteres

Role_id long não nulo

Produto:

Id long único valor autoincrementado não nulo

Nome string max 100 caracteres não nulo

Descrição string 255 caracteres

Preço Bigdecimal não nulo

Categoria_id long não nulo

Estoque inteiro não nulo

Pedido:

Id long único valor autoincrementado não nulo

Usuário_id long não nulo

Data_pedido datetime não nulo

Status_pedido string não nulo

Total bigdecimal não nulo

6.Regra de negócios

6.1Fluxo de Navegação do Site

O site do **DealHub** segue o fluxo padrão de um e-commerce:

6.1.1. Página Inicial (Home)

- Acessível a qualquer usuário, sem necessidade de login.
- Exibe:
    - Banner promocional rotativo.
    - Produtos em destaque e recomendados.
    - Categorias em destaque.
    - Link para promoções.
- Possui um menu superior com:
    - **Home**: Retorna à página inicial.
    - **Categorias**: Abre um submenu com as categorias de produtos.
    - **Carrinho**: Redireciona para a tela do carrinho.
    - **Login/Registrar**: Se o usuário não estiver autenticado.
    - **Área do Cliente**: Se o usuário estiver autenticado.
    - **Área Administrativa**: Se o usuário for ADMIN.
- Barra de pesquisa no topo para busca de produtos.

6.1.2. Categorias de Produtos

- Exibe uma listagem de produtos filtrada por categoria.
- Permite ordenação por preço, avaliação e novidades.
- Usuário pode clicar em um produto para visualizar detalhes.

6.1.3. Página do Produto

- Exibe as informações do produto:
    - Nome, preço, imagens, descrição, avaliações, especificações técnicas.
    - Quantidade disponível em estoque.
    - Botão **Adicionar ao Carrinho** (apenas se houver estoque).
- Usuário pode adicionar ao carrinho e ser redirecionado para lá ou continuar comprando.

6.1.4. Carrinho de Compras

- Exibe os produtos adicionados, permitindo:
    - Alteração de quantidade.
    - Remoção de itens.
    - Cálculo do valor total.

    - Cálculo do frete via CEP.
- Botão "Finalizar Compra" leva ao checkout.

6.1.5. Checkout

- Apenas para usuários autenticados.
- Usuário deve fornecer:
    - Endereço de entrega.
    - Método de pagamento (Cartão, PIX ou Boleto).
    - Revisão dos itens e valores.
- Após confirmação, pedido é criado e status definido como "Aguardando Pagamento".

6.1.6. Área do Cliente

- Disponível apenas para usuários logados.
- Permite:
    - Atualizar dados pessoais e senha.
    - Visualizar histórico de pedidos e status.
    - Gerenciar endereços de entrega.

6.1.7. Área Administrativa

- Apenas para usuários com role **ADMIN**.
- Funcionalidades:
    - **Gerenciar Produtos** (criar, editar, excluir).
    - **Gerenciar Categorias**.
    - **Gerenciar Pedidos** (alterar status).
    - **Gerenciar Usuários**.
    - **Dashboard com métricas de vendas**.

6.2. Regras de Acesso e Cadastro

6.2.1. Cadastro de Usuário

- Usuários podem se cadastrar informando:
    - Nome completo, e-mail, CPF, telefone e senha.
- O sistema valida:
    - CPF único e válido.
    - E-mail único e válido.

    - Senha com pelo menos 8 caracteres, incluindo letra e número.
- Após cadastro, recebe um e-mail de confirmação.
- Somente após confirmação pode efetuar login.

6.2.2. Login e Autenticação

- Login feito via e-mail e senha.
- Senha armazenada criptografada (BCrypt).
- Após login:
    - Se for **cliente**, redireciona para **Home**.
    - Se for **admin**, pode acessar a **Área Administrativa**.

6.2.3. Permissões de Acesso

Funcionalidade	**Cliente**	**Administrador**
Home	✅	✅
Ver produtos	✅	✅
Carrinho	✅	✅
Checkout	✅ (logado)	✅
Gerenciar Pedidos	❌	✅
Gerenciar Produtos	❌	✅
Gerenciar Usuários	❌	✅
Dashboard Administrativo	❌	✅

6.3. Regras Específicas do Sistema

6.3.1. Produtos

- Apenas administradores podem cadastrar e editar produtos.
- Produtos sem estoque não podem ser adicionados ao carrinho.

6.3.2. Pedidos

- Após a compra, o pedido recebe status inicial: **"Aguardando Pagamento"**.
- Após pagamento confirmado, status muda para **"Em Processamento"**.
- Após envio, status muda para **"Enviado"** e o cliente recebe código de rastreamento.
- Cliente pode cancelar pedidos apenas se o status for **"Aguardando Pagamento"**.

6.3.3. Carrinho

- O carrinho expira após 24 horas sem ação do usuário.
- Se um produto no carrinho ficar sem estoque, o sistema remove automaticamente.

6.3.4. Avaliações de Produtos

- Apenas clientes que compraram podem avaliar.
- Avaliações podem ser denunciadas e removidas por administradores.

6.4. Regras de Interface e UX

- A barra de pesquisa sempre visível no topo.
- Ícone de carrinho exibe número de itens dentro.
- Notificações visuais quando um item é adicionado ao carrinho.
- Confirmação visual para cada ação importante (ex: remoção de item do carrinho).
- Layout responsivo para desktop e mobile.

Relacionamentos do USUÁRIO

- **USUÁRIO → ENDEREÇO**: Um usuário pode ter vários endereços (1-N)
    - Implementação: @OneToMany na classe Usuario para lista de Endereco
- **USUÁRIO → CARRINHO**: Um usuário tem exatamente um carrinho (1-1)
    - Implementação: @OneToOne bidirecional
- **USUÁRIO → PEDIDO**: Um usuário pode fazer vários pedidos (1-N)
    - Implementação: @OneToMany na classe Usuario
- **USUÁRIO → AVALIAÇÃO**: Um usuário pode fazer várias avaliações (1-N)
    - Implementação: @OneToMany na classe Usuario

Relacionamentos do PRODUTO

- **PRODUTO → CATEGORIA**: Cada produto pertence a exatamente uma categoria (N-1)
    - Implementação: @ManyToOne na classe Produto
- **PRODUTO → SUBCATEGORIA**: Um produto pode (ou não) pertencer a uma subcategoria (N-0/1)
    - Implementação: @ManyToOne(optional=true)
- **PRODUTO → IMAGEM_PRODUTO**: Um produto pode ter várias imagens (1-N)
    - Implementação: @OneToMany na classe Produto

- **PRODUTO → AVALIAÇÃO**: Um produto pode receber várias avaliações (1-N)
    - Implementação: @OneToMany na classe Produto
- **PRODUTO → ITEM_CARRINHO**: Um produto pode estar em vários itens de carrinho (1-N)
    - Implementação: Relacionamento indireto via ItemCarrinho
- **PRODUTO → ITEM_PEDIDO**: Um produto pode ser vendido em vários itens de pedido (1-N)
    - Implementação: Relacionamento indireto via ItemPedido

Relacionamentos do CARRINHO

- **CARRINHO → ITEM_CARRINHO**: Um carrinho contém vários itens (1-N)
    - Implementação: @OneToMany na classe Carrinho

Relacionamentos do PEDIDO

- **PEDIDO → ITEM_PEDIDO**: Um pedido inclui vários itens (1-N)
    - Implementação: @OneToMany na classe Pedido
- **PEDIDO → USUÁRIO**: Cada pedido pertence a exatamente um usuário (N-1)
    - Implementação: @ManyToOne na classe Pedido
- **PEDIDO → ENDEREÇO**: Cada pedido tem um endereço de entrega específico (N-1)
    - Implementação: @ManyToOne na classe Pedido
- **PEDIDO → PAGAMENTO**: Cada pedido tem exatamente um pagamento (1-1)
    - Implementação: @OneToOne bidirecional

Relacionamentos de CATEGORIA/SUBCATEGORIA

- **CATEGORIA → SUBCATEGORIA**: Uma categoria pode ter várias subcategorias (1-N)
    - Implementação: @OneToMany na classe Categoria
- **CATEGORIA → PRODUTO**: Uma categoria agrupa vários produtos (1-N)
    - Implementação: @OneToMany na classe Categoria
- **SUBCATEGORIA → CATEGORIA**: Cada subcategoria pertence a uma categoria pai (N-1)
    - Implementação: @ManyToOne na classe Subcategoria

Relacionamentos dos ITEMS

- **ITEM_CARRINHO → PRODUTO**: Cada item referencia um produto específico (N-1)
    - Implementação: @ManyToOne na classe ItemCarrinho

- **ITEM_CARRINHO → CARRINHO**: Cada item pertence a um carrinho (N-1)
    - Implementação: @ManyToOne na classe ItemCarrinho
- **ITEM_PEDIDO → PRODUTO**: Cada item de pedido referencia um produto (N-1)
    - Implementação: @ManyToOne na classe ItemPedido
- **ITEM_PEDIDO → PEDIDO**: Cada item pertence a um pedido (N-1)
    - Implementação: @ManyToOne na classe ItemPedido

Relacionamentos de AVALIAÇÃO

- **AVALIAÇÃO → PRODUTO**: Cada avaliação é para um produto específico (N-1)
    - Implementação: @ManyToOne na classe Avaliacao
- **AVALIAÇÃO → USUÁRIO**: Cada avaliação é feita por um usuário (N-1)
    - Implementação: @ManyToOne na classe Avaliacao

Relacionamento de PAGAMENTO

- **PAGAMENTO → PEDIDO**: Cada pagamento está vinculado a um pedido (N-1)
    - Implementação: @OneToOne na classe Pagamento com mappedBy no Pedido

src/main/java/com/edgecommerce/

├── config/ # Classes de configuração

├── controller/ # Controladores MVC

|--- enums/

├── dto/ # Objetos de Transferência de Dados

├── exception/ # Tratamento de exceções

├── model/ # Entidades JPA

├── repository/ # Interfaces Spring Data JPA

├── service/ # Lógica de negócio

└── EdgeCommerceApplication.java

Etapas recomendadas:

1. **Definir o domínio do sistema**
   → Quais são as principais entidades? (ex: Usuário, Cliente, Produto, Pedido, etc.)
2. **Criar o banco de dados e suas entidades no Java**
   → Usar @Entity, @Id, @OneToMany, @ManyToOne...

3. **Configurar o application.properties**
   → Para conectar com o MySQL e configurar o JPA.
4. **Criar os repositórios (JpaRepository)**
   → Para fazer as operações básicas sem precisar escrever SQL.
5. **Criar os services (regras de negócio)**
   → Aqui você encapsula a lógica de como os dados devem ser tratados.
6. **Criar os controllers (camada web)**
   → Expor APIs REST (ou páginas Thymeleaf).
7. **Construir a interface (HTML com Thymeleaf)**
   → Se você estiver fazendo front com o próprio Spring.

- **Sistema de Autenticação Completo** (Spring Security + JWT)
    - Configuração de roles (USER, ADMIN)
    - Proteção de endpoints
    - Recuperação de senha
- **Gerenciamento de Endereços**
    - CRUD de endereços para usuários
- **Carrinho de Compras Avançado**
    - Persistência do carrinho
    - Validação de estoque
    - Cálculo de total
- **Checkout e Pedidos**
    - Fluxo completo de finalização
    - Integração com gateway de pagamento (simulado inicialmente)
- **Página de Produto**
    - Detalhes completos
    - Avaliações
    - Adição ao carrinho

1.2. Funcionalidades Administrativas

- **Dashboard Administrativo**
    - Métricas básicas de vendas

- **Gerenciamento Avançado de Produtos**
    - Upload de imagens
    - Controle de estoque
- **Gerenciamento de Pedidos**
    - Alteração de status
    - Filtros