**Centro Universitário de João Pessoa UNIPÊ**

Documento de Visão

# OBJETIVO DESTE DOCUMENTO

# Este documento tem como objetivo detalhar todas as especificações, funcionalidades, requisitos e diretrizes para o desenvolvimento, implementação e operação de um sistema de compra de flores online. A plataforma será projetada para oferecer aos clientes uma experiência de compra eficiente, intuitiva e segura, permitindo que eles naveguem, personalizem e adquiram arranjos florais para diversas ocasiões. Este documento serve como um guia abrangente para desenvolvedores, stakeholders e usuários, garantindo um entendimento claro e uniforme do sistema a ser criado.

# HISTÓRICO DE REVISÃO

|  |  |  |  |
| --- | --- | --- | --- |
| ***Histórico de Revisão*** | |  |  |
| **Data** | **Autor** | **Descrição** | **Versão** |
| 25/10 | Davi Nogueira | Criação do Documento | 1.0 |
| 02/07 | Davi Nogueira | Atualização do Documento | 1.1 |
| 04/11 | Davi Nogueira | Atualização do Documento | 1.2 |
| 11/11 | Davi Nogueira | Atualização do Documento | 1.3 |
| 18/11 | Davi Nogueira | Atualização do Documento | 1.4 |
| 20/11 | Davi Nogueira | Finalização do documneto | 1.5 |

# ESCOPO DO PRODUTO

O sistema de compra de flores online será uma plataforma digital que permitirá aos usuários comprar flores de maneira fácil e eficiente. O sistema deve ser acessível via navegador web e dispositivos móveis, proporcionando uma interface amigável que facilite a navegação e a finalização de compras. O principal objetivo é atender à demanda por flores para diversas ocasiões, como aniversários, casamentos, funerais, e outras celebrações.

# NÃO ESCOPO DO PRODUTO

O sistema **NÃO** deverá fornecer os serviços para atender as necessidades dos interessados:

* O sistema não deve conter funcionalidades avançadas.
* O sistema não permitirá a personalização do design da plataforma.
* O sistema não terá versões em aplicativos móveis

REQUISITOS

Os requisitos não funcionais asseguram que o sistema opere de maneira eficiente, segura e escalável:

**Desempenho**: O sistema deve ser capaz de processar múltiplas transações simultaneamente sem comprometer a velocidade e a eficiência.

**Segurança**: Implementação de protocolos de segurança robustos para proteger dados sensíveis dos usuários e garantir transações seguras.

**Usabilidade**: A interface deve ser intuitiva e fácil de usar, proporcionando uma experiência de usuário agradável e sem fricções.

**Escalabilidade**: O sistema deve ser escalável para suportar um aumento no número de usuários e transações, garantindo uma performance consistente.

**Confiabilidade**: O sistema deve ter alta disponibilidade (99.9% uptime) e implementar backups regulares para prevenir perda de dados.

**Manutenibilidade**: O sistema deve ser facilmente mantido e atualizado, com código bem documentado e arquitetura modular.

**Portabilidade**: Compatível com múltiplas plataformas e navegadores, funcionando sem grandes modificações.

**Interoperabilidade**: Capacidade de interagir com outros sistemas através de APIs e padrões abertos.

**Eficiência**: Responder às requisições em menos de 2 segundos e utilizar recursos computacionais eficientemente.

**Escalabilidade Horizontal**: Autoescalabilidade e balanceamento de carga para suportar aumento na carga de trabalho.

**Recuperabilidade**: Recuperar-se rapidamente de falhas, com RPO de menos de 15 minutos e RTO de menos de 30 minutos.

**Experiência do Usuário**: Design responsivo e acessível, garantindo uma interface amigável e usável por pessoas com deficiência.

# Descrição dos Envolvidos

|  |  |  |  |
| --- | --- | --- | --- |
| **Nome** | **Responsabilidades** | **Perfil** |  |
| Administrador | Os administradores gerenciam o sistema, mantendo o catálogo atualizado, monitorando estoques, processando pedidos e oferecendo suporte aos clientes. Eles garantem que o sistema funcione eficientemente e que os clientes estejam satisfeitos. | Utilizador da plataforma como ADM |  |
| Usuário/  Cliente | Os clientes são os usuários finais que utilizam o sistema para comprar flores. Eles acessam a plataforma, navegam pelo catálogo, personalizam arranjos, realizam pagamentos e acompanham os pedidos. Seus objetivos incluem encontrar e comprar flores rapidamente, personalizar arranjos e receber atualizações sobre os pedidos. | Utilizador da plataforma como Comprador |  |

Diagramas

Diagrama de relacionamentos:

Um diagrama de relacionamento, também conhecido como Diagrama de Entidade-Relacionamento (ERD ou DER), é uma ferramenta visual usada para modelar e ilustrar a estrutura de um banco de dados. Ele mostra como diferentes entidades (como tabelas) estão conectadas entre si, representando a relação entre os dados.

Diagrama

Descrição gerada automaticamente

**Tabela "pessoa" (na seção "Cliente")**

* **id\_pessoa**: Um número inteiro que serve como identificador único para cada pessoa. Este campo é a chave primária da tabela e é autoincrementado.
* **nome**: Um campo de texto (VARCHAR) com até 150 caracteres, que armazena o nome da pessoa.
* **cpf**: Um campo de data que armazena o CPF (Cadastro de Pessoas Físicas) do cliente.
* **nascimento**: Um campo de data que armazena a data de nascimento do cliente.
* **email**: Um campo de texto (VARCHAR) com até 150 caracteres, que armazena o endereço de e-mail da pessoa.
* **senha**: Um campo de texto (VARCHAR) com até 30 caracteres, que armazena a senha do cliente.

**Tabela "pedido" (na seção "Loja")**

* **id\_pedido**: Um número inteiro que serve como identificador único para cada pedido. Este campo é a chave primária da tabela e é autoincrementado.
* **id\_pessoa**: Um número inteiro que se refere ao identificador único da pessoa que fez o pedido (chave estrangeira que referencia pessoa.id\_pessoa).
* **id\_produto**: Um número inteiro que se refere ao identificador único do produto solicitado (chave estrangeira que referencia produto.id\_produto).
* **preco\_total\_pedido**: Um campo numérico (FLOAT) que armazena o preço total do pedido.

**Tabela "produto" (na seção "Loja")**

* **id\_produto**: Um número inteiro que serve como identificador único para cada produto. Este campo é a chave primária da tabela e é autoincrementado.
* **nome\_produto**: Um campo de texto (VARCHAR) com até 150 caracteres, que armazena o nome do produto.
* **descricao\_produto**: Um campo de texto que pode armazenar até 1000 caracteres para descrever o produto.
* **preco\_produto**: Um campo numérico (FLOAT) que armazena o preço do produto.
* **quantidade\_produto**: Um campo numérico (INT) que armazena a quantidade disponível do produto.

**Relacionamentos**

* A tabela pedido está relacionada à tabela pessoa por meio da chave estrangeira id\_pessoa, que referencia id\_pessoa na tabela pessoa. Isso indica que cada pedido está associado a uma pessoa específica.
* A tabela pedido também está relacionada à tabela produto por meio da chave estrangeira id\_produto, que referencia id\_produto na tabela produto. Isso indica que cada pedido inclui um produto específico.

Diagrama de caso de uso

Diagrama

Descrição gerada automaticamente

Um diagrama de caso de uso é uma ferramenta visual utilizada em análise e design de sistemas para ilustrar as interações entre os usuários (atores) e o sistema. Ele mostra as funcionalidades que o sistema deve oferecer (casos de uso) e como essas funcionalidades são acessadas pelos diferentes atores.

**Atores**

1. **Cliente**:
   * Representa a pessoa que acessa a loja online para navegar, selecionar e comprar produtos.
2. **Sistema**:
   * Representa o sistema da loja online que processa as ações do cliente, como validação de dados e processamento de compras.

**Casos de Uso**

1. **Navegar pelo Catálogo**:
   * O cliente pode navegar pelo catálogo de produtos disponíveis na loja online.
2. **Colocar Item no Carrinho**:
   * O cliente pode adicionar produtos ao carrinho de compras enquanto navega pelo catálogo.
3. **Informar Endereço**:
   * O cliente deve fornecer o endereço de entrega para onde os produtos comprados serão enviados.
4. **Finalizar Compra**:
   * O cliente pode finalizar a compra dos itens que adicionou ao carrinho. Este caso de uso é composto por outros casos de uso relacionados:
     + **Preencher Dados do Cartão de Crédito**: O cliente deve inserir as informações do cartão de crédito.
     + **Verificar Dados do Cartão de Crédito** (<<extends>>): Este é um caso de uso estendido que ocorre para garantir que os dados do cartão de crédito fornecidos pelo cliente sejam válidos.
     + **Faturar Compra** (<<include>>): O sistema inclui este caso de uso para processar o pagamento e gerar a fatura da compra.
     + **Enviar E-mail de Confirmação** (<<include>>): O sistema envia um e-mail de confirmação da compra ao cliente, informando que o pedido foi realizado com sucesso.

**Relacionamentos**

* **<<extends>>**: Indica que o caso de uso "Verificar Dados do Cartão de Crédito" é uma extensão necessária do caso de uso "Preencher Dados do Cartão de Crédito".
* **<<include>>**: Indica que os casos de uso "Faturar Compra" e "Enviar E-mail de Confirmação" são incluídos como parte do processo de "Finalizar Compra".

Front-End:

Linguagens utulizadas: HTML, CSS, JavaScript

**Código HTML**

Este documento HTML representa a estrutura e o conteúdo de uma página web para uma loja de flores online. Abaixo, explicamos cada parte do código e suas funções:

Interface gráfica do usuário

Descrição gerada automaticamente

O documento HTML representa a estrutura de uma página web para uma loja de flores online chamada "FLOWER STORE".

O documento começa com a declaração do tipo HTML5, informando ao navegador que o código segue esse padrão. Em seguida, a tag <html> define o idioma do documento como português do Brasil com o atributo lang="pt-BR".

Dentro da tag <head>, vários metadados e links são especificados:

* <meta charset="UTF-8">: Define a codificação de caracteres como UTF-8, suportando caracteres especiais.
* <meta name="viewport" content="width=device-width, initial-scale=1.0">: Configura o viewport para tornar o site responsivo, ajustando a largura à tela do dispositivo.
* <link rel="shortcut icon" href="imagens/favicon\_io/favicon.ico" type="image/x-icon">: Define o ícone (favicon) que aparece na aba do navegador.
* <title>FLOWER STORE</title>: Define o título da página que será exibido na aba do navegador.
* <link rel="stylesheet" href="css/main.css">: Linka o arquivo CSS que define o estilo visual da página.

**Corpo do Documento**

A tag <body> contém todo o conteúdo visível da página. A primeira seção, uma div com a classe navegacao, inclui a barra de navegação da loja:

* **Logo da loja**: Representada por uma imagem com a classe um.
* **Nome da loja**: Representado por uma imagem com a classe dois.
* **Campo de busca**: Permite ao usuário pesquisar produtos, identificado por um campo de texto com a classe pesquisa.
* **Ícones de carrinho de compras e login**: Incluem eventos de clique para ações como abrir o modal do carrinho e o login.

A navegação principal, contida na tag <nav class="principal">, inclui links para diferentes categorias de produtos (Arranjos, Buquês, Cestas, Presentes e Flores).

**Carrossel de Imagens**

A seção carrossel exibe um carrossel de imagens que provavelmente mostram promoções ou produtos principais da loja.

Interface gráfica do usuário, Aplicativo, Site

Descrição gerada automaticamente

**Modal do Carrinho de Compras**

A seção de cabeçalho <header> contém o modal do carrinho de compras, ativado quando o ícone do carrinho é clicado. O modal inclui:

Texto

Descrição gerada automaticamente com confiança média

* Ícone para fechar o modal.
* Título do modal.
* Lista de itens no carrinho.
* Total do carrinho.
* Botão para finalizar a compra.

**Listagem de Produtos**

A div container agrupa todos os produtos listados na página, com cada produto sendo representado por uma div com a classe flower. Cada div de produto inclui:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

* **Imagem do produto**: Representa visualmente o item.
* **Título do produto**: Nome do produto, como "Rosas Vermelhas" ou "Girassóis Amarelos".
* **Preço do produto**: Exibe o preço em reais (R$).
* **Informação de parcelamento**: Mostra as opções de parcelamento sem juros.
* **Botão de compra**: Chama uma função JavaScript addToCart() para adicionar o produto ao carrinho de compras.

Os produtos são organizados por categorias, como arranjos, buquês, cestas, presentes e flores, facilitando a navegação e a busca pelos itens desejados.

**Rodapé**

A seção rodape fornece informações de contato e links adicionais, incluindo:



* **Ícones e links para redes sociais**: Como WhatsApp e Instagram da loja.
* **Campo para email**: Permite ao usuário inscrever-se para receber promoções.
* **Links informativos**: Para páginas como "Quem Somos?", "Nossa Loja" e "Horários de Entrega".
* **Formas de pagamento aceitas**: Exibidas por ícones de cartões de crédito e Pix.
* **Mensagem de direitos autorais**: Indica que todos os direitos são reservados pela loja.

**Código Java Script**

**Galeria de Imagens**

**Seleção e contagem de imagens**:

* O código seleciona o elemento HTML com a classe imagens, que contém todas as fotos do carrossel. Ele também conta quantas imagens existem nesse contêiner.

**Função** proximaImagem:

* Esta função é encarregada de avançar para a próxima imagem no carrossel. Toda vez que é chamada, a variável index é aumentada em 1. O operador módulo (%) garante que quando chegamos ao final da lista de imagens, voltamos ao início, criando um looping contínuo. A propriedade style.transform é ajustada para mover o contêiner de imagens horizontalmente, simulando a troca de imagens.

**Intervalo para troca automática**:

* O comando setInterval(proximaImagem, 3000); define que a função proximaImagem será chamada a cada 3 segundos, fazendo com que as imagens mudem automaticamente nesse intervalo.

**Carrinho de Compras**

**Definição da variável** carrinho:

* let carrinho = []; inicializa uma lista vazia que irá armazenar os itens adicionados ao carrinho de compras.

**Função** addToCart:

* Esta função é chamada quando o usuário clica no botão "COMPRAR AGORA". Ela recebe dois parâmetros: nome e preco, que representam o nome e o preço do produto, respectivamente. A função cria um objeto item contendo esses dados e adiciona o objeto à lista carrinho. Logo em seguida, exibe um alerta informando que o item foi adicionado ao carrinho.

**Função** toggleCarrinho:

* Esta função mostra ou oculta o modal do carrinho de compras. O código seleciona o elemento modal usando getElementById e alterna seu estilo display entre "block" e "none". Em seguida, chama a função mostrarCarrinho para atualizar a exibição dos itens no carrinho.

**Função** mostrarCarrinho:

* A função seleciona os elementos HTML que exibem os itens do carrinho e o total do carrinho usando getElementById. Primeiro, ela limpa a lista de itens do carrinho antes de adicionar novos itens. Em seguida, inicializa uma variável total para calcular o valor total dos itens. A função então percorre os itens no carrinho, criando um elemento li para cada item e o adiciona à lista de itens do carrinho. Finalmente, a função atualiza a exibição do total do carrinho com a soma dos preços dos itens.

**Função** finalizarCompra:

* Esta função verifica se o carrinho está vazio. Se estiver, exibe um alerta informando que o carrinho está vazio. Caso haja itens no carrinho, exibe um alerta de agradecimento pela compra. A função então limpa a lista carrinho após a compra ser finalizada e chama a função toggleCarrinho para fechar o modal do carrinho.

Essas funcionalidades tornam a experiência de navegação e compra na loja de flores online mais dinâmica e interativa, permitindo que os usuários naveguem pelas imagens dos produtos automaticamente e gerenciem um carrinho de compras de forma eficiente.

Back-End:

Linguagens: MySQL, PHP

**Operações de Gerenciamento de Banco de Dados**

**1. Exclusão de Tabela**

O código começa removendo uma tabela chamada pessoa. Isso significa que todos os dados armazenados nessa tabela serão permanentemente apagados, e a estrutura da tabela será removida do banco de dados.

**2. Seleção de Dados**

Depois, o código tenta selecionar todos os registros da tabela pessoa. Se a tabela tiver sido removida anteriormente, essa operação resultará em um erro, pois a tabela não existirá mais.

**3. Criação de Tabela de Produtos**

Em seguida, o código cria uma nova tabela chamada produto. Esta tabela é configurada para armazenar informações sobre produtos, incluindo:

* Um identificador único para cada produto (que é gerado automaticamente e aumenta a cada novo registro).
* O nome do produto.
* Uma descrição do produto (opcional).
* O preço do produto.

**4. Criação de Tabela de Pedidos**

O código então cria uma tabela chamada pedido no banco de dados específico mybanco. Esta tabela armazena informações sobre pedidos realizados pelos clientes, incluindo:

* Um identificador único para cada pedido (também gerado automaticamente).
* O identificador do cliente que fez o pedido (referenciando a tabela pessoa).
* O identificador do produto solicitado (referenciando a tabela produto).
* O preço total do pedido.

**5. Seleção de Dados de Produtos**

Depois de criar as tabelas, o código seleciona todos os registros da tabela produto para exibir as informações dos produtos armazenados.

**6. Inserção de Registros na Tabela de Produtos**

Por fim, o código insere múltiplos registros na tabela produto. Cada registro contém o nome e o preço de diferentes tipos de flores. Esta operação popula a tabela com diversos produtos, permitindo que o sistema tenha dados reais para trabalhar.

**Código PHP**

Este código PHP cria uma tabela em um banco de dados MySQL se ela ainda não existir. Aqui está a análise detalhada:

1. **Conexão com o Banco de Dados**
   * $criaTabela = mysqli\_query($conexao, "CREATE TABLE IF NOT EXISTS $database.$nomeTabela(...)"**)**:
     + Utiliza a função mysqli\_query para executar uma consulta SQL.
     + $conexao: A variável que representa a conexão com o banco de dados MySQL.
     + O comando SQL passado para mysqli\_query cria a tabela somente se ela não existir.
2. **Nome da Tabela**
   * $database.$nomeTabela:
     + $database: Nome do banco de dados.
     + $nomeTabela: Nome da tabela a ser criada.
3. **Estrutura da Tabela**
   * A tabela é definida com as seguintes colunas e suas propriedades:
     + id\_pessoa:
       - Tipo: INT
       - Não pode ser nulo
       - Deve ser único
       - Incrementado automaticamente (AUTO\_INCREMENT)
       - Chave primária (PRIMARY KEY)
     + nome:
       - Tipo: VARCHAR(50)
       - Não pode ser nulo
     + cpf:
       - Tipo: INT
       - Não pode ser nulo
     + nascimento:
       - Tipo: DATE
       - Não pode ser nulo
     + email:
       - Tipo: VARCHAR(50)
       - Não pode ser nulo
     + senha:
       - Tipo: VARCHAR(36)
       - Não pode ser nulo

**Resumo do Código**

Este trecho de código PHP cria uma tabela em um banco de dados MySQL, chamada de acordo com os valores de $database e $nomeTabela. A tabela inclui uma chave primária id\_pessoa que é um inteiro autoincrementado e único. Outras colunas incluem nome, cpf, nascimento, email, e senha, todas definidas como não nulas. Isso garante que cada registro na tabela tenha valores definidos para essas colunas.