Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

**PROJETO**

Desenvolver o Sistema para uma Loja On-Line

Diagramas DER e Mer

Banco de Dados

Back-end

Front-end

Mobile

**Professores:**  
Rennye Lima

Welligton Fabio

**Alunos:**

Roberto D. Pereira Dias

Rafael Fernandes de Faria

Matheus Belarmino Pignata

Victoria Rosa dos Santos

**Turma:** 2DES

**Grupo:**  2 B

Sumario:

Cronograma de desenvolvimento do projeto3

Divisão de tarefas4

Lojas de referência5

Modelo entidade e relacionamentos6

Manual do sistema7

Códigos de status10

Métodos CRUD11

**Cronograma do desenvolvimento do projeto**

**Primeiro:**

Levantamento de duas lojas, como referências, documentar pontos positivos e negativos.

Elaboração e criação do MER, DER do banco de dados.

Elaboração e criação do LAYOUT do FRONTEND.

**Reuniões:**

Reuniões diária e semanais, onde foi pontuado cronograma, melhorias do projeto

**Pautas:**

Nas reuniões definimos as metas e prazos para cada processo do desenvolvimento.

**Divisão das tarefas**

**Equipe:**

Desenvolveu MER e DER, do banco de dados, relacional, conceitual e lógico.

Com base no diagrama foi projetado o banco de dados e back-end.

**Roberto D. Pereira Dias:**

Organizar e distribuir as tarefas entre a equipe.

Acompanhar o andamento das tarefas entre a equipe.

Documentar as tarefas realizadas.

Desenvolver o back-end em JAVA

Auxílio no front-end

**Rafael Fernandes de Faria:**

Desenvolver o banco de dados utilizando MySQL.

Documentar as tarefas realizadas.

Auxílio no back-end.

**Matheus Belarmino Pignata:**

Organizar e distribuir as tarefas entre a equipe.

Acompanhar o andamento das tarefas entre a equipe.

Desenvolvimento de todo o front-end utilizando HTML, CSS e JavaScript.

Criação de imagens originais do site.

**Victoria Rosa dos Santos:**

Desenvolver de todo o mobile.

Desenvolvimento do REACT-NATIVE.

Desenvolvimento HTML, CSS e JavaScript.

**Lojas de Referência**

**Kabum:**

**Pontos positivos:** Layout e exposição dos produtos no site, descrição técnica dos produtos, responsividade do site.

**Pontos negativos:** Falta um pouco de benefícios como VR, VA. Não possuí promoções internas para funcionários. Algumas coisas ainda não são bem pensadas ou planejada.

Empresa familiar, com regime de punição e que vê o empregado como um mal necessário. Poucos benefícios, desorganização, sem política estratégica, comunicação péssima.

**Link loja:** https://www.kabum.com.br/

**Pichau:**

**Pontos positivos:** Layout e responsividade do site.

**Pontos negativos:** Descrição técnica dos produtos muito poluída, RMA, produto entregue não condis com o que foi comprado,

**Link:** https://www.pichau.com.br

**MODELO ENTIDADE RELACIONAMENTO (MER e DER)**

**Diagrama

Descrição gerada automaticamenteMODELO: Conceitual:**

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente**MODELO: Lógico:**

**Manual do sistema**

**Banco de dados**

**Tabelas**

No banco de dados desenvolvido utilizando MySQL, foram feitas 7 tabelas de acordo com as necessidades do programa.

**Estados**

Foi feita uma tabela com todos os 26 estados e o distrito federal do Brasil além de um campo com suas respectivas siglas utilizando o id como chave primária.

**Cidades**

Na tabela “Cidades” foram feitos 3 campos, id\_cidade que é a chave primária, e uma chave estrangeira: id\_estado, foram cadastradas as 5.564 cidades do Brasil.

**Produtos**

A tabela “Produtos” é usada para armazenas e classificar os produtos cadastrados, contendo um campo chave primária id\_produto, imagem para armazenar links de imagens, valor, promoção caso haja uma mudança de preço temporária, marca, nome, categoria entre elas: placa de vídeo, processador e memória RAM, o último campo, descrição que irá contêm a descrição dos produtos que podem variar de pequenos a longos e extensos textos.

**Clientes**

Uma tabela feita para armazenar os cadastros dos clientes, necessitando do cpf, nome completo, e-mail e senha, o campo id\_cliente foi definido como chave primaria, pois será utilizado em outras tabelas para identificar o cliente.

**Enderecos**

Uma tabela feita para armazenar os endereços dos clientes, considerando que alguns deles possam ter mais de um endereço, podem existir várias tabelas Enderecos para um único cliente. Nesta tabela contém os campos id\_cliente como chave estrangeira para identificar os clientes, id\_cidade para identificar a cidade, id\_uf para identificar o estado, cep logradouro e bairro.

**Pedidos**

A tabela “pedidos” irá conter os pedidos realizados pelos clientes, estejam eles em aberto, em andamento, ou finalizado, nesta tabela contém os campos: id\_pedido como chave primária data, valor e status.

**Itens**

Na tabela “Itens” ficarão armazenados os itens da tabela Pedidos, terá: id\_pedido para identificar o pedido, id\_produto para identificar os produtos do pedido pelo seu id e quantidade, haverá várias destas tabelas para cada pedido pois cada uma delas irá armazenar um único item do pedido, sua quantidade, e o preço total.

**Mobile**

O aplicativo, feito em react-native, carrega os produtos do banco de dados, da categoria 'promoção' e exibe na tela para o cliente, quando o botão comprar é pressionado, carrega o navegador onde o cliente é redirecionado para o site na página do produto selecionado para finalizar a compra.

**Front-end**

**Tela login**

O Front-end possui uma tela login que disponibiliza duas opções: login e cadastro, no login pedindo um e-mail e senha já cadastrados, e no cadastro pedindo o nome completo, CPF, e-mail, senha e confirmação da senha, que após o término vincula automaticamente a nova conta.

**Tela home**

Logo após a realização do cadastro ou login, fica disponível a tela “home” que mostra para o usuário todos os produtos cadastrados no banco de dados, disponibilizando sua imagem, nome, preço original e promoção, além disso, no canto esquerdo fica a mostra as opções de filtrar por categoria, acesso ao carrinho, e ao suporte. No canto direito superior fica localizado o nome do usuário junto ao botão “sair” que ao ser clicado irá desvincular a conta do usuário, e “pedidos” que levará para outra tela, além disso no topo da tela foi colocada uma caixa para a pesquisa direta de produtos pelo seu nome.

**Tela produto**

A tela “produto” é carregada ao usuário selecionar um produto na tela home, onde será mostrado o nome, imagem, preço, avaliação e descrição do produto, junto a um botão “comprar” que ao ser clicado adiciona o item para o carrinho que fica disponível a sua remoção, e escolha da quantidade do mesmo.

**Tela carrinho**

Nesta tela ficam disponíveis os itens do pedido ainda em aberto, mostrando também o valor total, opções de pagamento, e uma botão “finalizar compra”, que quando selecionado levará o usuário a tela “pedidos”.

**Tela pedidos**

Aqui ficará disponível todos os pedidos de acordo com o id da conta do usuário, serão mostradas informações como: ID do pedido, total, nome dos itens comprados, e data da realização do pedido.

**Back-end**

Seguindo o padrão MVC para o desenvolvimento do backend foram feitos os pacotes “controller, DAO, model e servlet”.

**Controller:**

Foi feita uma classe para cada uma as tabelas, onde contém os métodos para criar, ler, atualizar e deletar registros do banco de dados.

**Dao**:

Foram feitas as classes que estabelecem a conexão com o banco de dados e manda os comandos SQL para executar as ações definidas.

**Servlet:**

Neste pacote é onde ficam as classes HTTP, com os métodos “doGet, doPost, doPut e doDelete”, que recebe as requisições do front-end e os manda para a classe DAO inserir junto aos comandos SQL.

**Model:**

Na classe model ficam os construtores vazios e cheios, métodos Getters and Setters, hashCode and equals, além da formatação toString e toJSON.

**Códigos de status:**

**400 Bad request**

Este erro significa que as informações mandadas para o servido do site estão incorretas.

**401 Unauthorized**

Este erro ocorre quando o usuário tenta acessar uma página restrita e que ele não tem acesso.

**404 Not found**

Este erro ocorre quando o usuário se comunica com o servidor, mas o servidor não consegue encontrar o que foi pedido, a página pode não existir ou a URL foi inserida de forma incorreta.

**500 Internal server error**

Este erro indica que o servidor está enfrentando algum problema, ou está inativo.

**406 Not acceptable**

Este erro indica que o servidor não conseguiu produzir uma resposta que combine com os valores inseridos pelo usuário, geralmente ocorre quando o usuário tenta criar ou deletar dados.

**/Clientes**

**GET /clientes**

Resposta: 200 ok

[{

“id\_cliente”: 1,

“cpf”: “111.222.333-12”,

“nomeCompleto”: “João Almeida Almeida”,

“email”: “[usuario123@.com](mailto:usuario123@.com)”,

“senha”: “123abc”

}]

**POST /clientes**

Entrada:

{

“cpf”: 111.222.333-12

“nome\_completo”: “João Almeida Almeida”,

“email”: “[usuario123@.com](mailto:usuario123@.com)”,

“senha”: “123abc”

}

Resposta: 201 created

{ "id\_cliente":1}

**PUT /clientes**

{

“id\_cliente”: 1,

“cpf”: “111.222.333-12”,

“nome\_completo”: “Júlio Almeida Diaz”,

“email”: “[usuario123@.com](mailto:usuario123@.com)”,

“senha”: “123abc”

}

Reposta: 410 Gone

**DELETE /clientes**

{

“Id\_cliente”: 1

}

Resposta:

200 ok

/Produtos

**GET /produtos**

Reposta: 200 ok

[{

“id\_produto”: 1,

“Imagem”: <https://foto.jpg>”,

“valor”: 100.00,

“promocao”: 70.00,

“marca”: “intel”,

“nome”: “I5 9400f”,

“categoria”: “processador”,

“descricao”: “Um modelo de nona geração que se destaca por ter um bom...”

}]

**POST /produtos**

Entrada:

{

"imagem": "https://imagem.jpeg",

"valor": 100.00,

"promocao": 75.00,

"marca": "intel",

"nome": "I59400f",

"categoria": "processador",

"descricao": "Um modelo de nona geração que se destaca por ter um bom..."

}

Resposta: 201 created

{“id\_produto: 1”}

**PUT/produtos**

Entrada:{

“id\_produo”: 1,

"imagem": "https://imagem.jpeg",

"valor": 150.00,

"promocao": 100.00,

"marca": "Intel",

"nome": "I5 9400F",

"categoria": "processador",

"descricao": "Um modelo de nona geração que se destaca por ter um bom..."

}

Resposta: 410 gone

**DELETE/produtos**

Entrada:

{

“idProduto”: 1,

}

Reposta:200 ok

/Pedidos

GET/pedidos

Saída: 200 ok

[{

"cliente":{"id\_cliente":1},

"data":"2021-11-09",

"valor":100,

“status”: “em aberto”

}]

POST/pedidos

Entrada:

{

"id\_cliente": 1,

"data": "2021-11-08",

"valor": "2000.0",

"status": "em aberto"

}

Saída: 201 created

{“id\_pedido”: 1}

DELETE/pedidos

Entrada:

{

“Id\_pedido”: 1,

}

Resposta: 410 gone

/Itens

GET/Itens

Resposta: 200 ok

[{

"produto":{"promocao":0,"valor":0},

"pedido":15,

"quantidade":1

}]

/Estados

GET/estados

Resposta:

[{

"uf":"AC",

"nome":"Acre",

"iduf":1

},

{

"uf":"AL",

"nome":"Alagoas",

"iduf":2

}

POST/estados

Entrada:

{

“Nome”: “São Paulo”,

“uf”: “SP”

}

Saída:

201: created

{“id\_estado”: 26}

DELETE/estados

{

“id\_uf”: 1

}

Saída: 200 ok

/CIDADES

GET/cidades

Resposta: 200 ok

{

“idCidade”:26,

“nome”: “Fundão”,

“idUf”: 8

}

POST/cidades

Entrada:

{

“nome”: “fundão”,

“id\_uf”: 26

}

Resposta: 201 created

{ “id\_cidade”: 5564}

DELETE/cidades

Entrada:

{

“id\_cidade”: 5565

}

Resposta: 200 ok