-- Criação do Banco de Dados

CREATE DATABASE crm\_imobiliaria;

USE crm\_imobiliaria;

-- 🔐 Tabela de Usuários (Login e Segurança)

CREATE TABLE Usuarios (

id INT AUTO\_INCREMENT PRIMARY KEY,

nome VARCHAR(100) NOT NULL,

cargo ENUM('Admin', 'Corretor', 'Secretaria') NOT NULL,

contato VARCHAR(100) NOT NULL,

login VARCHAR(50) UNIQUE NOT NULL,

senha VARCHAR(255) NOT NULL, -- senha armazenada como hash

status ENUM('Ativo', 'Inativo') DEFAULT 'Ativo',

data\_criacao TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- 📞 Tabela de Clientes Compradores

CREATE TABLE ClientesCompradores (

id INT AUTO\_INCREMENT PRIMARY KEY,

nome VARCHAR(100) NOT NULL,

contato VARCHAR(100) NOT NULL,

origem\_lead VARCHAR(100),

status\_lead ENUM('Novo', 'Em negociação', 'Fechado', 'Perdido') DEFAULT 'Novo',

id\_responsavel INT,

FOREIGN KEY (id\_responsavel) REFERENCES Usuarios(id) ON DELETE SET NULL

);

-- 🏠 Tabela de Clientes Proprietários

CREATE TABLE ClientesProprietarios (

id INT AUTO\_INCREMENT PRIMARY KEY,

nome VARCHAR(100) NOT NULL,

contato VARCHAR(100) NOT NULL,

id\_responsavel INT,

FOREIGN KEY (id\_responsavel) REFERENCES Usuarios(id) ON DELETE SET NULL

);

-- 🏢 Tabela de Imóveis

CREATE TABLE Imoveis (

id INT AUTO\_INCREMENT PRIMARY KEY,

id\_proprietario INT,

tipo ENUM('Casa', 'Apartamento', 'Terreno' ) NOT NULL,

localizacao VARCHAR(255) NOT NULL,

valor DECIMAL(15,2) NOT NULL,

status ENUM('Disponível', 'Vendido', 'Reservado') DEFAULT 'Disponível',

data\_cadastro TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

num\_quartos INT,

num\_banheiros INT,

area\_total DECIMAL(10,2),

area\_construida DECIMAL(10,2),

garagem INT,

fotos TEXT,

descricao TEXT,

FOREIGN KEY (id\_proprietario) REFERENCES ClientesProprietarios(id) ON DELETE SET NULL

);

-- 📝 Tabela de Ações de Clientes Compradores

CREATE TABLE AcoesCompradores (

id INT AUTO\_INCREMENT PRIMARY KEY,

id\_cliente INT,

descricao TEXT,

data\_acao TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

resultado VARCHAR(100),

responsavel INT,

FOREIGN KEY (id\_cliente) REFERENCES ClientesCompradores(id) ON DELETE CASCADE,

FOREIGN KEY (responsavel) REFERENCES Usuarios(id) ON DELETE SET NULL

);

-- 🔑 Criação de Usuários de Teste com Senha Criptografada (SHA2)

INSERT INTO Usuarios (nome, cargo, contato, login, senha) VALUES

('Admin', 'Admin', 'admin@email.com', 'admin', SHA2('123456', 256)),

('João Silva', 'Corretor', 'joao@email.com', 'joaosilva', SHA2('joaopass', 256)),

('Maria Santos', 'Secretaria', 'maria@email.com', 'mariasantos', SHA2('mariapass', 256));

-- 🔎 🔥 Procedimento para Autenticação de Login

DELIMITER $$

CREATE PROCEDURE LoginUsuario(

IN p\_login VARCHAR(50),

IN p\_senha VARCHAR(255)

)

BEGIN

DECLARE id\_usuario INT;

-- Verifica se o login e senha estão corretos

SELECT id INTO id\_usuario

FROM Usuarios

WHERE login = p\_login AND senha = SHA2(p\_senha, 256) AND status = 'Ativo';

IF id\_usuario IS NULL THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Login ou senha inválidos';

END IF;

END$$

DELIMITER ;

## 🏗️ ****Plano de Expansão****

Para seguir em frente, vou estruturar o projeto em módulos e criar as **classes principais** para cada entidade do DER, mantendo a arquitetura orientada a objetos. Vou adicionar:  
✔️ Conexão (já temos ✅)  
✔️ Login e autenticação (já temos ✅)  
✔️ CRUD de **Clientes** (já temos ✅)  
✔️ CRUD de **Imóveis**  
✔️ CRUD de **Propostas**  
✔️ CRUD de **Corretores**  
✔️ Logs e permissões (pode entrar na segunda fase)

## 📁 ****Nova Estrutura de Arquivos****

📂 app/

├── 📄 Conexao.php

├── 📄 Auth.php

├── 📄 Cliente.php

├── 📄 Imovel.php

├── 📄 Proposta.php

├── 📄 Corretor.php

├── 📄 Log.php

📂 public/

├── 📄 login.php

├── 📄 cliente/

│ ├── 📄 listar.php

│ ├── 📄 cadastrar.php

│ ├── 📄 atualizar.php

│ ├── 📄 excluir.php

├── 📄 imovel/

│ ├── 📄 listar.php

│ ├── 📄 cadastrar.php

│ ├── 📄 atualizar.php

│ ├── 📄 excluir.php

├── 📄 proposta/

│ ├── 📄 listar.php

│ ├── 📄 cadastrar.php

│ ├── 📄 atualizar.php

│ ├── 📄 excluir.php

## Conexao.php

<?php

class Conexao

{

private static $instance;

private function \_\_construct() {}

public static function getInstance()

{

if (!isset(self::$instance)) {

try {

self::$instance = new PDO(

'mysql:host=localhost;dbname=sistema\_imobiliario;charset=utf8',

'root',

''

);

self::$instance->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

self::$instance->setAttribute(PDO::ATTR\_DEFAULT\_FETCH\_MODE, PDO::FETCH\_ASSOC);

} catch (PDOException $e) {

die("Erro de conexão: " . $e->getMessage());

}

}

return self::$instance;

}

}

## Auth.php

<?php

require\_once 'Conexao.php';

class Auth

{

public static function login($email, $senha)

{

$pdo = Conexao::getInstance();

$sql = "SELECT \* FROM usuarios WHERE email = :email";

$stmt = $pdo->prepare($sql);

$stmt->bindValue(':email', $email);

$stmt->execute();

$usuario = $stmt->fetch();

if ($usuario && password\_verify($senha, $usuario['senha'])) {

session\_start();

$\_SESSION['usuario'] = $usuario['id'];

$\_SESSION['nome'] = $usuario['nome'];

return ['success' => true, 'message' => 'Login realizado com sucesso!'];

} else {

return ['success' => false, 'message' => 'E-mail ou senha inválidos!'];

}

}

public static function logout()

{

session\_start();

session\_destroy();

header("Location: ../public/login.php");

exit();

}

public static function verificarAcesso()

{

session\_start();

if (!isset($\_SESSION['usuario'])) {

header("Location: ../public/login.php");

exit();

}

}

}

if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

$email = $\_POST['email'];

$senha = $\_POST['password'];

$resultado = Auth::login($email, $senha);

echo json\_encode($resultado);

}

## Log.php

<?php

class Log

{

public static function registrar($mensagem)

{

$arquivo = '../logs/app.log';

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

$usuario = $\_SESSION['nome'] ?? 'Sistema';

$conteudo = "[$data] - [$usuario] - $mensagem" . PHP\_EOL;

file\_put\_contents($arquivo, $conteudo, FILE\_APPEND);

}

}

## ****CRUD de Imóveis (Imovel.php)****

### ****app/Imovel.php****

<?php

require\_once 'Conexao.php';

class Imovel

{

public function cadastrar($descricao, $tipo, $valor, $endereco, $status)

{

$pdo = Conexao::getConexao();

$stmt = $pdo->prepare("INSERT INTO Imoveis (descricao, tipo, valor, endereco, status)

VALUES (:descricao, :tipo, :valor, :endereco, :status)");

$stmt->bindParam(':descricao', $descricao);

$stmt->bindParam(':tipo', $tipo);

$stmt->bindParam(':valor', $valor);

$stmt->bindParam(':endereco', $endereco);

$stmt->bindParam(':status', $status);

return $stmt->execute();

}

public function listar()

{

$pdo = Conexao::getConexao();

$stmt = $pdo->prepare("SELECT \* FROM Imoveis");

$stmt->execute();

return $stmt->fetchAll(PDO::FETCH\_ASSOC);

}

public function atualizar($id, $descricao, $tipo, $valor, $endereco, $status)

{

$pdo = Conexao::getConexao();

$stmt = $pdo->prepare("UPDATE Imoveis

SET descricao = :descricao, tipo = :tipo, valor = :valor, endereco = :endereco, status = :status

WHERE id = :id");

$stmt->bindParam(':id', $id);

$stmt->bindParam(':descricao', $descricao);

$stmt->bindParam(':tipo', $tipo);

$stmt->bindParam(':valor', $valor);

$stmt->bindParam(':endereco', $endereco);

$stmt->bindParam(':status', $status);

return $stmt->execute();

}

public function excluir($id)

{

$pdo = Conexao::getConexao();

$stmt = $pdo->prepare("DELETE FROM Imoveis WHERE id = :id");

$stmt->bindParam(':id', $id);

return $stmt->execute();

}

}

### ****Cadastrar Imóvel (cadastrar.php)****

<?php

require\_once '../../app/Imovel.php';

$imovel = new Imovel();

if ($imovel->cadastrar('Apartamento 3 quartos', 'Venda', 350000, 'Rua das Flores, 123', 'Disponível')) {

echo "Imóvel cadastrado com sucesso!";

}

### ****Listar Imóveis (listar.php)****

<?php

require\_once '../../app/Imovel.php';

$imovel = new Imovel();

$imoveis = $imovel->listar();

foreach ($imoveis as $i) {

echo $i['descricao'] . " - R$ " . number\_format($i['valor'], 2, ',', '.') . "<br>";

}

## ****2. CRUD de Propostas (Proposta.php)****

### ****app/Proposta.php****

<?php

require\_once 'Conexao.php';

class Proposta

{

public function cadastrar($id\_cliente, $id\_imovel, $valor\_oferecido, $status)

{

$pdo = Conexao::getConexao();

$stmt = $pdo->prepare("INSERT INTO Propostas (id\_cliente, id\_imovel, valor\_oferecido, status)

VALUES (:id\_cliente, :id\_imovel, :valor\_oferecido, :status)");

$stmt->bindParam(':id\_cliente', $id\_cliente);

$stmt->bindParam(':id\_imovel', $id\_imovel);

$stmt->bindParam(':valor\_oferecido', $valor\_oferecido);

$stmt->bindParam(':status', $status);

return $stmt->execute();

}

public function listar()

{

$pdo = Conexao::getConexao();

$stmt = $pdo->prepare("SELECT \* FROM Propostas");

$stmt->execute();

return $stmt->fetchAll(PDO::FETCH\_ASSOC);

}

}

### ****Cadastrar Proposta (cadastrar.php)****

<?php

require\_once '../../app/Proposta.php';

$proposta = new Proposta();

if ($proposta->cadastrar(1, 1, 340000, 'Em análise')) {

echo "Proposta registrada com sucesso!";

}

## 👥 ****3. CRUD de Corretores (Corretor.php)****

### ****app/Corretor.php****

<?php

require\_once 'Conexao.php';

class Corretor

{

public function cadastrar($nome, $contato)

{

$pdo = Conexao::getConexao();

$stmt = $pdo->prepare("INSERT INTO Corretores (nome, contato)

VALUES (:nome, :contato)");

$stmt->bindParam(':nome', $nome);

$stmt->bindParam(':contato', $contato);

return $stmt->execute();

}

public function listar()

{

$pdo = Conexao::getConexao();

$stmt = $pdo->prepare("SELECT \* FROM Corretores");

$stmt->execute();

return $stmt->fetchAll(PDO::FETCH\_ASSOC);

}

}

### ****Cadastrar Corretor (cadastrar.php)****

<?php

require\_once '../../app/Corretor.php';

$corretor = new Corretor();

if ($corretor->cadastrar('Carlos Silva', '99999-8888')) {

echo "Corretor registrado!";

}

FRONTEND

## ****1. Tela de Login****

### ****public/login.php****

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Login - Sistema Imobiliário</title>

<script src="../assets/scripts.js" defer></script>

<link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">

</head>

<body class="bg-gray-100 flex items-center justify-center h-screen">

<div class="bg-white p-6 rounded-lg shadow-lg w-full max-w-md">

<h2 class="text-2xl font-bold mb-4 text-center">Login</h2>

<form id="loginForm" action="../app/Auth.php" method="POST" class="space-y-4">

<div>

<label for="email" class="block text-sm font-medium text-gray-700">E-mail</label>

<input type="email" id="email" name="email" required

class="w-full px-3 py-2 border rounded-lg focus:ring focus:ring-blue-400">

</div>

<div>

<label for="password" class="block text-sm font-medium text-gray-700">Senha</label>

<input type="password" id="password" name="password" required

class="w-full px-3 py-2 border rounded-lg focus:ring focus:ring-blue-400">

</div>

<button type="submit" class="w-full bg-blue-500 text-white py-2 rounded-lg hover:bg-blue-600">

Entrar

</button>

</form>

</div>

</body>

</html>

### ****Validação do Login (scripts.js)****

document.addEventListener('DOMContentLoaded', () => {

const loginForm = document.getElementById('loginForm');

loginForm.addEventListener('submit', async (e) => {

e.preventDefault();

const formData = new FormData(loginForm);

const response = await fetch('../app/Auth.php', {

method: 'POST',

body: formData

});

const result = await response.json();

if (result.success) {

window.location.href = '../dashboard/index.php';

} else {

alert(result.message);

}

});

});

## 🏡 ****2. Tela Inicial (Dashboard)****

### ****public/dashboard/index.php****

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<title>Dashboard</title>

<link href="https://cdn.jsdelivr.net/npm/tailwindcss@2.2.19/dist/tailwind.min.css" rel="stylesheet">

</head>

<body class="bg-gray-100">

<div class="container mx-auto p-6">

<h1 class="text-3xl font-bold mb-6">Dashboard</h1>

<div class="grid grid-cols-3 gap-6">

<div class="bg-white p-6 rounded-lg shadow-lg">

<h2 class="text-xl font-bold mb-2">Total de Imóveis</h2>

<p id="totalImoveis" class="text-gray-700 text-lg">0</p>

</div>

<div class="bg-white p-6 rounded-lg shadow-lg">

<h2 class="text-xl font-bold mb-2">Total de Clientes</h2>

<p id="totalClientes" class="text-gray-700 text-lg">0</p>

</div>

<div class="bg-white p-6 rounded-lg shadow-lg">

<h2 class="text-xl font-bold mb-2">Total de Propostas</h2>

<p id="totalPropostas" class="text-gray-700 text-lg">0</p>

</div>

</div>

</div>

<script>

document.addEventListener('DOMContentLoaded', async () => {

const response = await fetch('../app/DashboardData.php');

const data = await response.json();

document.getElementById('totalImoveis').textContent = data.totalImoveis;

document.getElementById('totalClientes').textContent = data.totalClientes;

document.getElementById('totalPropostas').textContent = data.totalPropostas;

});

</script>

</body>

</html>

## ****3. CRUD de Imóveis (Frontend)****

### ****public/imovel/listar.php****

<div class="p-4">

<h2 class="text-2xl font-bold mb-4">Lista de Imóveis</h2>

<table class="w-full bg-white rounded-lg shadow-lg">

<thead>

<tr>

<th class="border p-2">Descrição</th>

<th class="border p-2">Valor</th>

<th class="border p-2">Status</th>

<th class="border p-2">Ações</th>

</tr>

</thead>

<tbody id="listaImoveis">

<!-- JS vai preencher aqui -->

</tbody>

</table>

</div>

### ****JS para listar Imóveis****

document.addEventListener('DOMContentLoaded', async () => {

const response = await fetch('../../app/Imovel.php?action=listar');

const data = await response.json();

const lista = document.getElementById('listaImoveis');

lista.innerHTML = data.map(imovel => `

<tr>

<td>${imovel.descricao}</td>

<td>${imovel.valor}</td>

<td>${imovel.status}</td>

<td><button>Editar</button> <button>Excluir</button></td>

</tr>

`).join('');

});