

Sistema de Gerenciamento de Condomínios

Rute Alves Fernandes - 232009549

Mariana Simion dos Santos - 232005227

O nosso projeto tem como tema o desenvolvimento de um Sistema de Gerenciamento de Condomínios, com o intuito de aprimorar a gestão condominial e oferecer maior controle aos gestores sobre as informações do seu condomínio. A proposta busca modernizar os processos internos, permitindo que cada administrador visualize e gerencie dados relacionados aos moradores, áreas comuns, residências disponíveis e demais informações que atualmente são tratadas de forma manual e descentralizada.

A partir de um banco de dados relacional, o projeto garantirá integridade, segurança e acessibilidade das informações, permitindo a realização de operações de CRUD completas para apoio à administração.

Além de promover eficiência, transparência e organização na administração condominial, o projeto também irá contribuir para a melhoria da gestão local nas comunidades urbanas, atendendo à proposta da disciplina ao oferecer uma solução tecnológica que facilita a vida dos administradores e também da população local, promovendo uma gestão mais moderna, acessível e integrada aos contextos urbanos atuais.

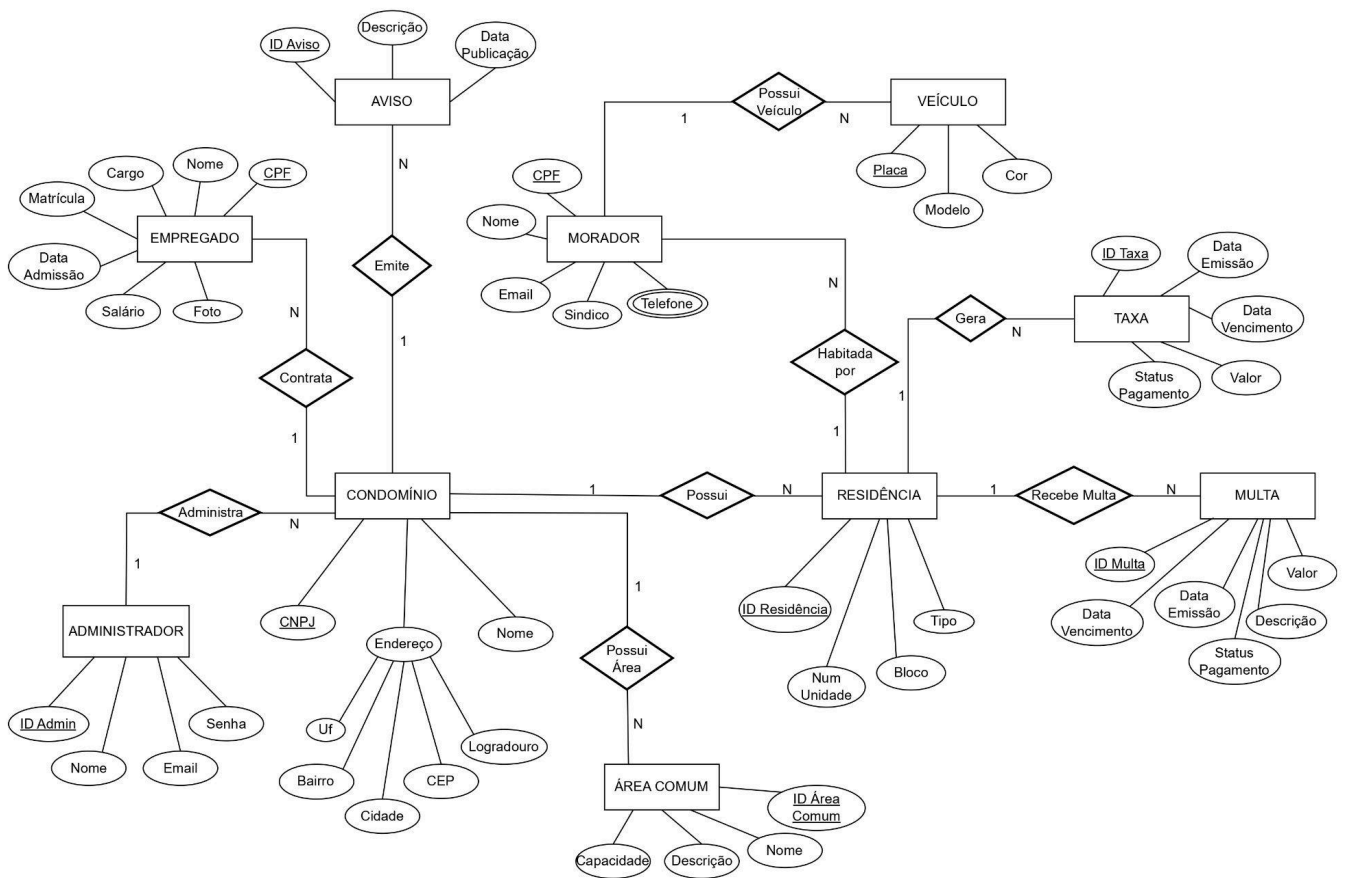
Github do projeto:

<https://github.com/marianassimion/GerenciadorCondominios>

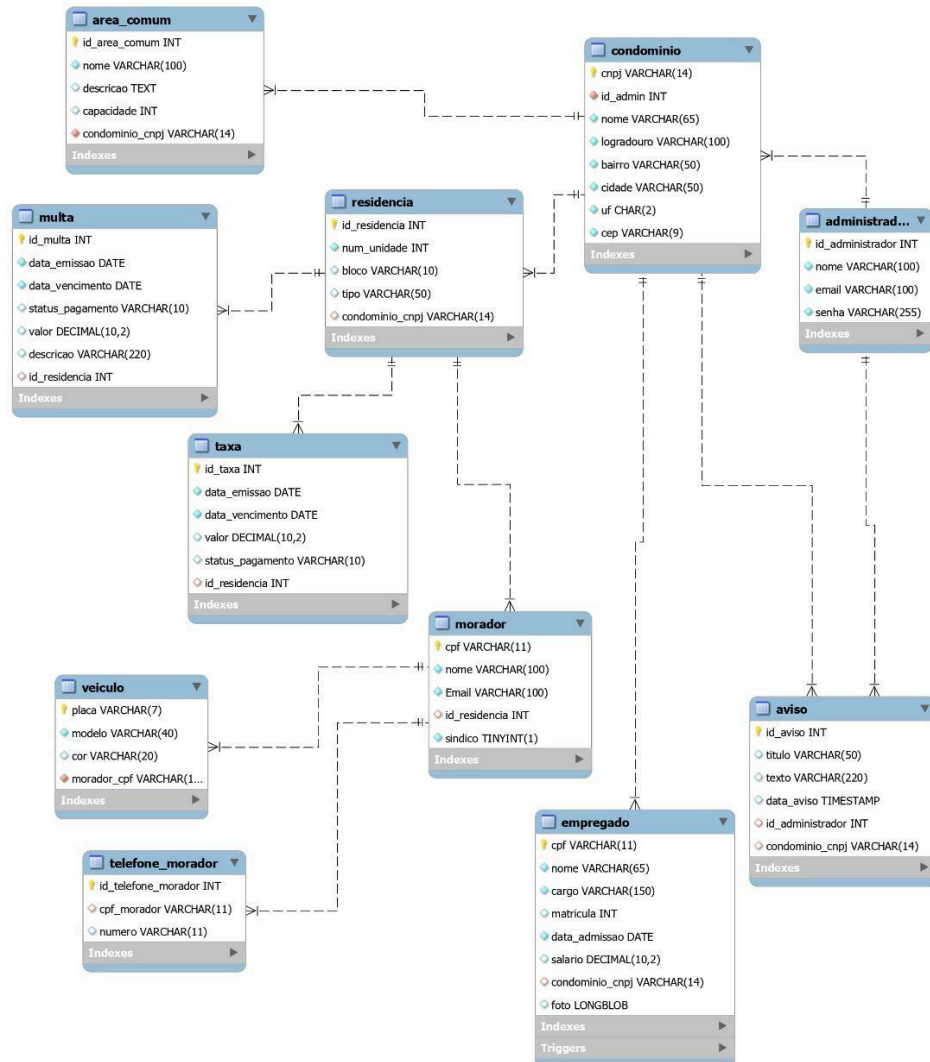
Inteligência Artificial

Durante o desenvolvimento do projeto, utilizamos ferramentas de inteligência artificial como apoio técnico, especialmente para a correção de erros, esclarecimento de dúvidas sobre bibliotecas e consulta à documentação mais recente do Streamlit. Entre as IAs utilizadas, destacam-se o ChatGPT, o Gemini e o Notebook LM, que contribuíram na elaboração de um script para a população do banco de dados, tornando o processo mais eficiente, seguro e alinhado às boas práticas de desenvolvimento de software.

Modelo de Entidade Relacionamento



Modelo Relacional



Script do Banco de Dados

/*SISTEMA DE GERENCIAMENTO DE CONDOMINIOS*/

```
CREATE DATABASE IF NOT EXISTS condominio;
USE condominio;

DROP TABLE IF EXISTS TELEFONE_MORADOR;
DROP TABLE IF EXISTS VEICULO;
DROP TABLE IF EXISTS TAXA;
DROP TABLE IF EXISTS MULTA;
DROP TABLE IF EXISTS AVISO;
DROP TABLE IF EXISTS EMPREGADO;
DROP TABLE IF EXISTS AREA_COMUM;
DROP TABLE IF EXISTS MORADOR;
DROP TABLE IF EXISTS RESIDENCIA;
DROP TABLE IF EXISTS CONDOMINIO;
DROP TABLE IF EXISTS ADMINISTRADOR;

CREATE TABLE ADMINISTRADOR (
    id_administrador integer AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    senha VARCHAR(255) NOT NULL
);

CREATE TABLE CONDOMINIO (
    cnpj          varchar(14) PRIMARY KEY,
    id_admin       INTEGER NOT NULL,
    nome           varchar(65) NOT NULL,
    logradouro     varchar(100) NOT NULL,
    bairro         varchar(50) NOT NULL,
    cidade         varchar(50) NOT NULL,
    uf             char(2) NOT NULL,
    cep            varchar(9) NOT NULL,

    CONSTRAINT FK_condominio_admin
        FOREIGN KEY(id_admin) REFERENCES ADMINISTRADOR(id_administrador)
);

CREATE TABLE RESIDENCIA (
    id_residencia INT PRIMARY KEY AUTO_INCREMENT,
    num_unidade  INTEGER unsigned NOT NULL,
    bloco        VARCHAR(10),
    tipo         VARCHAR(50),
    condominio_cnpj VARCHAR(14),

    CONSTRAINT FK_residencia_condominio
        FOREIGN KEY (condominio_cnpj) REFERENCES CONDOMINIO(cnpj)
);
```

```

CREATE TABLE MORADOR (
    cpf VARCHAR(11) PRIMARY KEY,
    nome VARCHAR(100) NOT NULL,
    Email VARCHAR(100) NOT NULL UNIQUE,
    id_residencia INTEGER,
    sindico BOOLEAN NOT NULL DEFAULT FALSE,

    CONSTRAINT FK_morador_residencia
        FOREIGN KEY (id_residencia) REFERENCES RESIDENCIA(id_residencia)
);

```

```

CREATE TABLE TELEFONE_MORADOR (
    id_telefone_morador INT PRIMARY KEY auto_increment,
    cpf_morador VARCHAR(11),
    numero VARCHAR(11),

    CONSTRAINT FK_telefone_morador
        FOREIGN KEY (cpf_morador) REFERENCES MORADOR(cpf)
        ON DELETE CASCADE
);

```

```

CREATE TABLE EMPREGADO (
    cpf varchar(11) PRIMARY KEY,
    nome varchar(65) NOT NULL,
    cargo varchar(150) NOT NULL,
    matricula integer unsigned UNIQUE,
    data_admissao date NOT NULL,
    salario decimal(10,2),
    condominio_cnpj varchar(14),
    foto LONGBLOB,

    CONSTRAINT FK_empregado_condominio
        FOREIGN KEY (condominio_cnpj) REFERENCES CONDOMINIO(cnpj)
);

```

```

CREATE TABLE VEICULO (
    placa VARCHAR(7) PRIMARY KEY,
    modelo VARCHAR(40) NOT NULL,
    cor VARCHAR(20),
    morador_cpf VARCHAR(11) NOT NULL,

    CONSTRAINT veiculo_morador
        FOREIGN KEY (morador_cpf) REFERENCES MORADOR(cpf)
);

```

```

CREATE TABLE TAXA (
    id_taxa INTEGER AUTO_INCREMENT PRIMARY KEY,
    data_emissao date NOT NULL,
    data_vencimento date NOT NULL,

```

```

    valor decimal(10,2),
    status_pagamento varchar(10) DEFAULT 'Pendente',
    id_residencia integer,

    CONSTRAINT FK_Taxa_Residencia
        FOREIGN KEY(id_residencia) REFERENCES RESIDENCIA(id_residencia)
);

CREATE TABLE MULTA (
    id_multa INTEGER AUTO_INCREMENT PRIMARY KEY,
    data_emissao date NOT NULL,
    data_vencimento date NOT NULL,
    status_pagamento varchar(10) DEFAULT 'Pendente',
    valor DECIMAL(10, 2),
    descricao VARCHAR(220),
    id_residencia integer,

    CONSTRAINT FK_Multa_Residencia
        FOREIGN KEY(id_residencia) REFERENCES RESIDENCIA(id_residencia)
);

CREATE TABLE AVISO (
    id_aviso INTEGER AUTO_INCREMENT PRIMARY KEY,
    titulo VARCHAR(50),
    texto VARCHAR(220),
    data_aviso TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    id_administrador integer,
    condominio_cnpj varchar(14),

    CONSTRAINT FK_aviso_administrador
        FOREIGN KEY(id_administrador) REFERENCES ADMINISTRADOR(id_administrador),

    CONSTRAINT FK_aviso_condominio
        FOREIGN KEY(condominio_cnpj) REFERENCES CONDOMINIO(cnpj)
);

CREATE TABLE AREA_COMUM (
    id_area_comum INTEGER PRIMARY KEY AUTO_INCREMENT,
    nome VARCHAR(100) NOT NULL,
    descricao TEXT,
    capacidade INT,
    condominio_cnpj VARCHAR(14) NOT NULL,

    CONSTRAINT FK_area_comum_condominio
        FOREIGN KEY (condominio_cnpj) REFERENCES CONDOMINIO(cnpj)
);

```

Diagrama de Acesso a Camada de Persistência

Para a comunicação entre a interface gráfica e o banco de dados, foi utilizada uma camada de persistência denominada `db_functions`, responsável por concentrar todas as operações de acesso aos dados, incluindo consultas (*selects*), inserções, atualizações e remoções. A interface realiza o acesso ao banco de dados exclusivamente por meio dessas funções, garantindo maior organização, segurança e desacoplamento entre as camadas do sistema.

