

ANÁLISE E DESENVOLVIMENTO DE SOFTWARE

TOÓPICOS INTREGADORES II

DOCUMENTAÇÃO DO PROJETO DE SOFTWARE CORRETORA DE IMÓVEIS

VERSÃO 1.1

Auto(a): <REBECCA ALMEIDA>

<https://github.com/rebeccaalmeida1995/Projeto-de-software-corretora-de-imoveis.git>

Controle de versão do documento

|  |  |  |
| --- | --- | --- |
| Data | versão | Descrição |
| 21/06/2020 | 1.1 | Exemplo de descrição: Sistema que efetua cadastro e busca de imóveis e anunciantes interessados em alugar ou vender os mesmos. |

# SUMÁRIO**2**

# **APRESENTAÇÃO3**

# MODELO DE VISÃO4

# REQUISITOS E RESTRIÇÕES5

# **MODELAGEM DE DADOS (**MODELO ENTIDADE RELACIONAMENTO(ER))**6**

# MODELO RELACIONAL DE DADOS7

# MODELO LÓGICO DE DADOS9

# DICIONÁRIO DE DADOS (Tabelas existentes)10

# DICIONÁRIO DE DADOS (Tabela Anuncio)11

# DICIONÁRIO DE DADOS (Tabela Imóvel)12

# DICIONÁRIO DE DADOS (Tabela Plano)13

# DICIONÁRIO DE DADOS (Tabela Forma de pagamento)14

# DICIONÁRIO DE DADOS (Tabela Comodo, Client e Busca)15

# DICIONÁRIO DE DADOS (Tabela location)16

# **DEFINIÇÃO DO BANCO DE DADOS17**

# DATA DEFINITION LANGUAGE (SCRIPT SQL DDL)18

# DATA QUERY LANGUAGE (SCRIPT SQL DQL)19

# **ANÁLISE E DESIGN20**

# DIAGRAMA DE CLASSES21

# **ARQUITETURA DE SOFTWARE22**

# PADRÃO DE PROJETO23

# PROTÓTIPO24

# TELAS E CLASSES DA CAMADA VIEW25

# CLASSES DA CAMADA CONTROLL26

# CLASSES DA CAMADA MODEL27

# CLASSES DE CONEXÃO COM O BANCO28

**APRESENTAÇÃO**

Este é um projeto de software com objetivo de aprendizagem acadêmica sem fins lucrativos. Desenvolvido para fixar o conteúdo do curso tecnólogo de análise e desenvolvimento de sistemas, aprendido em sala de aula na faculdade Joaquim Nabuco (Guararapes). Trata-se de um sistema que busca imóveis que estão a venda ou alugando, onde o cliente entra com o seu endereço de e-mail e senha para ter acesso a os imóveis de interesse. O sistema terá vários imóveis cadastrados com diversos perfis classificados, para suprir a necessidade do cliente. O site de busca intermediara entre o interessado no imóvel e o anunciante, facilitando a comunicação entre ambos, para garantir a entrega do serviço. Todo anunciante terá a oportunidade de cadastrar seu imóvel com a contratação de um plano especial de sua preferência e necessidade. Este sistema precisará guardar diversos dados, tal como, documentos e informações pessoais, então será importante manter a segurança e integridade do cliente.

MODELO DE VISÃO

Corretora IMv

No sistema de venda e aluguel de imóveis da corretora IMV será necessário cadastrar os imóveis, o mesmo terá um código, valor de venda ou aluguel, um preço de condomínio, um preço de IPTU, porém não será obrigatório informar condomínio e IPTU, área por m2, diferencial, tipo de negociação que será alugar ou vender, tipo de uso que poderá ser para uso residencial ou comercial e o tipo de imóvel que poderá ser cadastrado ou buscado como por exemplo, apartamento, casa, loft, Studio, casa de condomínio, sala comercial e etc..

Cada imóvel vai conter uma localização com rua, bairro, cidade, descrição onde poderá colocar seus pontos de referência, complemento e andar que serão opcionais e uf. Cada imóvel também tem uma quantidade de cômodos existente como, quantidade de quartos, quantidade de banheiros, suítes(opcional) e quantidade de vagas na garagem(opcional)

O sistema terá a opção de anúncios de imóveis que para cadastro será necessário escolher um tipo de anunciante que poderá ser corretor, imobiliária ou proprietário. adicionar um nome, um e-mail e dois telefones. Porém o sistema terá que salvar a data de anuncio para atualizar os imóveis cadastrado diariamente.

Terá uma opção de classificação da pessoa que quer anunciar, será pessoal física ou jurídica com esses dados necessários, nome completo, razão social, e-mail, telefone fixo, celular, CPF e CNPJ a depender da escolha da pessoa.

O cliente interessado poderá entrar no sistema para efetuar buscas, através de cadastro adicionando e-mail e senha poderá fazer a busca do imóvel por relevância que pode ser pela data de atualização, valor ou área do imóvel.

REQUISITOS DO SISTEMA

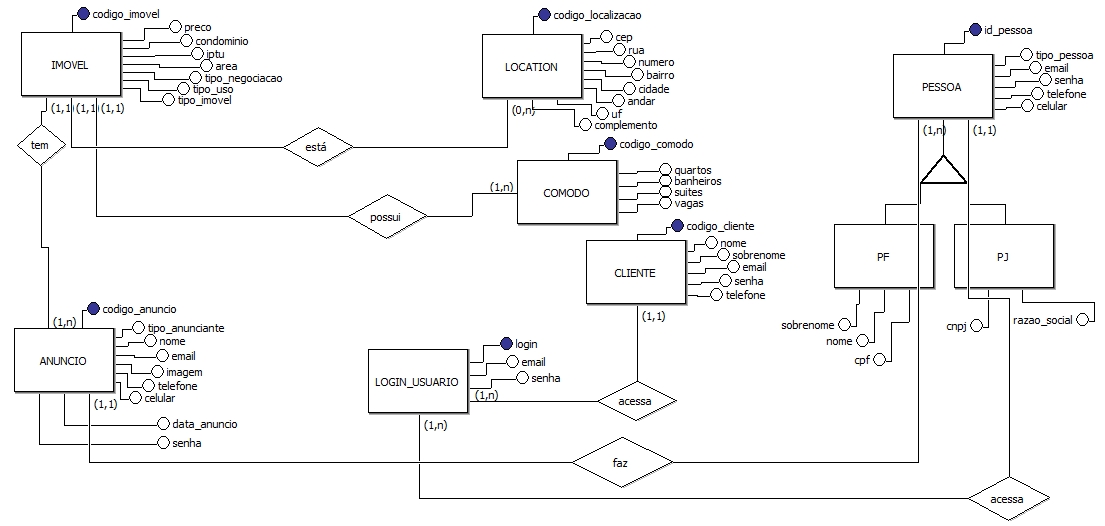
O sistema terá inicialmente a opção de anunciar imóveis, entrar ou cadastrar e central de ajuda como link disponível na tela, uma barra de pesquisa, e suas especificas classificações para busca do imóvel, também mostrara os imóveis e suas respectivas propriedades numa lista de cards. O link de anunciar imóveis levará ao cadastro da pessoa que fará uso do serviço e do plano do sistema, pessoas interessadas em imóveis para comprar ou alugar podem se cadastrar ou fazer login no link, entrar ou cadastrar e receber informações sobre seu interesse no seu e-mail.

Restrições

Para mostrar um ou mais imóveis nos cards é necessário existir o imóvel no sistema já cadastrado. Para anunciar um imóvel será necessário contratar algum plano do nosso sistema. Para contratar um plano ter um anuncio e ter dados válidos para o pagamento.

**MODELAGEM DE DADOS**

*MODELO ENTIDADE RELACINAMENTO(ER)*

****

**MODELO RELACIONAL – LÓGICO DE DADOS**

*SISTEMA CORRETORA DE IMOVEIS*

IMOVEL (código\_imovel, preco, comdominio, iptu, área, tipo\_negociacao, tipo\_uso, diferencial, tipo\_imovel)

codigo\_imovel PRIMARY KEY,

);

LOCATION (codigo\_localizacao, rua, cep, numero, bairro, cidade, uf, complemento, andar, codigo\_imovel)

codigo\_localizacao PRIMARY KEY,

codigo\_imovel FOREIGN KEY references IMOVEL

);

COMODO (código\_comodo, quartos, banheiros, suítes, vagas, código\_imovel)

codigo\_qtdComodos INTEGER PRIMARY KEY,

quartos INTEGER,

banheiros INTEGER,

suites INTEGER,

vagas INTEGER,

codigo\_imovel FOREIGN KEY references IMOVEL

);

ANUNCIO (código\_anuncio, código\_imovel, tipo\_anunciante, nome, email, senha, telefone, celular, imagem, data\_atual)

codigo\_anuncio INTEGER PRIMARY KEY,

codigo\_imovel FOREIGN KEY references IMOVEL

);

PESSOA (id\_pessoa, tipo\_pessoa, email, senha, telefone,celular, nome, sobrenome, cpf, razão\_social, cnpj, codigo\_anuncio, login)

id\_pessoa INTEGER PRIMARY KEY,

LOGIN INT ,

CODIGO\_ANUNCIO INT,

LOGIN FOREING KEY references LOGIN\_USUARIO,

codigo\_anuncio FOREIGN KEY references ANUNCIO,

);

CLIENT (id\_client, nome, sobrenome, email, senha, telefone, login)

id\_client PRIMARY KEY,

LOGIN INT,

LOGIN FOREING KEY references LOGIN\_USUARIO

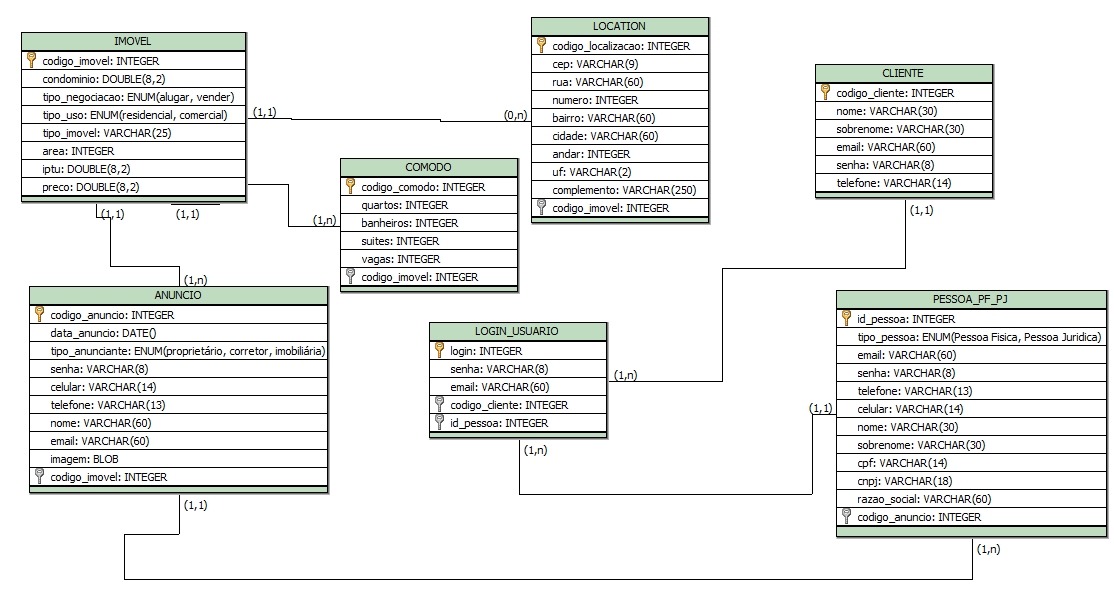
);

LOGIN\_USUARIO (login, código\_anuncio, email, senha, id\_pessoa, id\_cliente)

Login PRIMARY KEY,

);

**MODELO LÓGICO DE DADOS**

** *SISTEMA CORRETORA DE IMÓVEIS*

**DICIONÁRIO DE DADOS** *SISTEMA CORRETORA DE IMÓVEIS*

Tabelas existentes

|  |  |  |  |
| --- | --- | --- | --- |
| **Tabela** | **Relacionamento** | **Nome do Relacionamento** | **Descrição** |
| Tabela\_anuncio | Tabela imóvel | Pertence | Tabela para busca do imóvel |
|
| Tabela\_client | Busca |
|
| **Tabela** | **Relacionamento** | **Nome do Relacionamento** | Descrição |
| Tabela\_loginUsuario | Tabela\_pessoa | entra | Tabela do usuário logar no sistema |
| Tabela\_cliente | entra |
| Tabela imóvel | Tabela\_location | tem | Tabela para cadastro dos imóveis. |
| Tabela Comodo | possui |
| Tabela-pessoa | anuncia |

TABELA ANUNCIO

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ENTIDADE ANUNCIO** | | | | |
| Atributo | Tipo de dado | Compri | Restrições | Descrição |
| codigo\_anuncio | inteiro | 4 bytes | PK, NOT NULL | código de identificação do anuncio. |
| codigo\_imóvel | inteiro | 4 bytes | FK, NOT NULL | código de identificação do imóvel. |
| Tipo\_anunciante | caractere | 1 bit | NOT NULL | classificação de tipo de anunciante se é proprietário, corretor ou imobiliária. |
| data\_atual | date | 10 bytes | NOT NULL | data atual. |

TABELA IMÓVEL

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ENTIDADE IMÓVEL** | | | | |
| Atributo | Tipo de dado | Compri | Restrições | Descrição |
| codigo\_imovel | inteiro | 4 bytes | PK, NOT NULL | código de identificação do imóvel |
| preço | double |  | NOT NULL | valor do aluguel ou da venda do imóvel. |
| condomínio | double |  |  | valor do condomínio do imóvel. |
| IPTU | doubel |  |  | preço do IPTU do imóvel. |
| area | inteiro | 20 bytes | NOT NULL | tamanho do imóvel. |
| tipo\_negociação | enum | 15 bytes | NOT NULL | tipo de negociação que pode ser alugar ou vender. |
| tipo\_uso | enum | 15 bytes | NOT NULL | tipo de uso que pode ser pra residência ou comercio |
| tipo\_imovel | caractere | 60 bytes | NOT NULL | tipo de imóvel que pode ser apartamento ou casa e etc... |
| diferencial | caractere | 250bytes |  | descrição dos pontos de referência e outras atribuições do imóvel |

**DICIONÁRIO DE DADOS**

*SISTEMA CORRETORA DE IMÓVEIS*

TABELA COMODO

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ENTIDADE COMODO** | | | | |
| **Atributo** | **Tipo de dado** | **Compri** | **Restrições** | **Descrição** |
| codigo\_qtdComodos | inteiro | 4 bytes | PK, NOT NULL | código de identificação dos cômodos existentes. |
| codigo\_imovel | inteiro | 4 bytes | FK< NOT NULL | Código de identificação do imóvel relacionado ao comodo |
| quartos | inteiro | 1 bit | NOT, NULL | quantidade de quartos existentes |
| banheiros | inteiro | 1 bit | NOT, NULL | quantidade de banheiros existentes |
| suites | inteiro | 1bit |  | Quantidade de suítes existentes no imóvel |
| vagas | inteiro | 1 bit |  | Quantidade de vagas na garagem existentes no imóvel |

**DICIONÁRIO DE DADOS**

*SISTEMA CORRETORA DE IMÓVEIS*

TABELA LOCATION

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ENTIDADE LOCATION** | | | | |
| **Atributo** | **Tipo de dados** | **Compri** | **Restrições** | **Descrição** |
| codigo\_localizacao | inteiro | 4 bytes | PK, NOT NULL | Código de localização do imóvel |
| codigo\_imovel | inteiro | 4 bytes | FK, NOT NULL | Código de identificação do imóvel |
| rua | caractere | 60 bytes | NOT, NULL | rua |
| bairro | caractere | 60 bytes | NOT, NULL | bairro |
| cidade | caractere | 60 bytes | NOT, NULL | cidade |
| uf | caractere | 2 bytes | NOT, NULL | uf |
| numero | inteiro | 5 bytes | NOT, NULL | Número do imóvel |
| cep | caractere | 9 bytes | NOT, NULL | cep |
| andar | inteiro | 5 bytes |  | Andar do imóvel |
| complemento | caractere | 250 bytes |  | Descrição da localização pontos de referência e etc.... |

TABELA CLIENTE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ENTIDADE CLIENTE** | | | | |
| Atributo | Tipo de dado | Compri | Restrições | Descrição |
| id\_client | inteiro | 4 bytes | PK, NOT NULL | código de identificação do interessado no imóvel. |
| nome | caractere | 40bytes | NOTNULL | nome |
| sobrenome | caractere | 40bytes | NOTNULL | sobrenome |
| telefone | caractere | 14bytes | NOTNULL | telefone |
| email | caractere | 40 bytes | NOT NULL | email do interessado. |
| senha | caractere | 8 bytes | NOT NULL | senha. |
| login | int | 4 bytes | NOT NULL | Codigo de login |

TABELA PESSOA

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ENTIDADE PESSOA** | | | | |
| **Atributo** | **Tipo de dados** | **Compri** | **Restrições** | **Descrição** |
| id\_pessoa | inteiro | 4 bytes | PK, NOT NULL | Código de identificação de pessoa |
| tipo\_pessoa | enum | 20 bytes | NOT, NULL | Tipo de pessoa se é física ou juridica |
| email | caractere | 60 bytes | NOT, NULL | email |
| senha | caractere | 8 bytes | NOT, NULL | senha |
| celular | caractere | 14 bytes | NOT, NULL | celular |
| telefone | caractere | 13 bytes |  | telefone |
| nome | caractere | 30 bytes |  | nome |
| sobrenome | caractere | 30 bytes |  | sobrenome |
| cpf | caractere | 14 bytes |  | cpf |
| razaoSocial | caractere | 60 bytes |  | Razão social |
| cnpj | caractere | 18 |  | cnpj |
| login | int | 4 bytes | NOT NULL | Codigo de login |

TABELA LOGINUSUARIO

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ENTIDADE LOGINUSUARIO** | | | | |
| Atributo | Tipo de dado | Compri | Restrições | Descrição |
| login | inteiro | 4 bytes | PK, NOT NULL | código de identificação do login do usuário. |
| email | caractere | 40bytes | NOTNULL | email |
| senha | caractere | 8bytes | NOTNULL | senha |
|  |  |  |  |  |

**BANCO DE DADOS MYSQL**

O **MySQL** é um sistema de gerenciamento de banco de dados (SGBD), que utiliza a linguagem SQL (Linguagem de Consulta Estruturada, do inglês *Structured Query Language*) como interface. É atualmente um dos sistemas de gerenciamento de bancos de dados mais populares da Oracle Corporation, com mais de 10 milhões de instalações pelo mundo.[[2]](https://pt.wikipedia.org/wiki/MySQL#cite_note-2)

Entre os usuários do banco de dados MySQL estão: [NASA](https://pt.wikipedia.org/wiki/NASA), [Friendster](https://pt.wikipedia.org/wiki/Friendster" \o "Friendster), [Banco Bradesco](https://pt.wikipedia.org/wiki/Banco_Bradesco), [Dataprev](https://pt.wikipedia.org/wiki/Dataprev" \o "Dataprev), [HP](https://pt.wikipedia.org/wiki/Hewlett-Packard), [Nokia](https://pt.wikipedia.org/wiki/Nokia), [Sony](https://pt.wikipedia.org/wiki/Sony), [Lufthansa](https://pt.wikipedia.org/wiki/Lufthansa), U.S. Army, U.S. Federal Reserve Bank, [Associated Press](https://pt.wikipedia.org/wiki/Associated_Press), [Alcatel](https://pt.wikipedia.org/wiki/Alcatel), [Slashdot](https://pt.wikipedia.org/wiki/Slashdot" \o "Slashdot), [Cisco Systems](https://pt.wikipedia.org/wiki/Cisco_Systems), [Google](https://pt.wikipedia.org/wiki/Google), entre outros.

**Caracteristicas Mysql**

* Portabilidade (suporta praticamente qualquer plataforma atual);
* Compatibilidade (existem drivers [ODBC](https://pt.wikipedia.org/wiki/ODBC), [JDBC](https://pt.wikipedia.org/wiki/JDBC) e [.NET](https://pt.wikipedia.org/wiki/.NET) e módulos de interface para diversas linguagens de programação, como [Delphi](https://pt.wikipedia.org/wiki/Delphi), [Java](https://pt.wikipedia.org/wiki/Java_(linguagem_de_programa%C3%A7%C3%A3o)), [C](https://pt.wikipedia.org/wiki/Linguagem_de_programa%C3%A7%C3%A3o_C)/[C++](https://pt.wikipedia.org/wiki/C%2B%2B), [C#](https://pt.wikipedia.org/wiki/C_sharp), [Visual Basic](https://pt.wikipedia.org/wiki/Visual_Basic), [Python](https://pt.wikipedia.org/wiki/Python), [Perl](https://pt.wikipedia.org/wiki/Perl), [PHP](https://pt.wikipedia.org/wiki/PHP), [ASP](https://pt.wikipedia.org/wiki/ASP) e [Ruby](https://pt.wikipedia.org/wiki/Ruby_(linguagem_de_programa%C3%A7%C3%A3o)))
* Excelente desempenho e estabilidade;
* Pouco exigente quanto a recursos de novos [hardware](https://pt.wikipedia.org/wiki/Hardware);
* Facilidade no manuseio;
* É um [Software Livre](https://pt.wikipedia.org/wiki/Software_Livre) com base na GPL (entretanto, se o programa que acessar o Mysql não for GPL, uma licença comercial deverá ser adquirida)[[7]](https://pt.wikipedia.org/wiki/MySQL#cite_note-7);
* Contempla a utilização de vários Storage Engines como MyISAM, InnoDB, Falcon, BDB, Archive, Federated, CSV, Solid…
* Suporta controle transacional;
* Suporta Triggers;
* Support Cursors (Non-Scrollable e Non-Updatable);
* support Stored Procedures e Functions;
* Replicação facilmente configurável;
* Interfaces gráficas ([MySQL Toolkit](https://pt.wikipedia.org/w/index.php?title=MySQL_Toolkit&action=edit&redlink=1)) de fácil utilização cedidos pela MySQL Inc.

**DATA DEFINITION LANGUAGE (DDL)**

*COMANDOS DE CRIAÇÃO SQL SISTEMA DE CORRETORA DE IMÓVEIS*

Antes de executar os comados abaixo deverá ser criado um banco de dados chamado db\_corretoraIMv.

Create database db\_corretoraimv;

Use db\_corretoraimv;

CREATE TABLE IMOVEL (

codigo\_imovel INTEGER auto\_increment PRIMARY KEY,

preco DOUBLE(8,2) not null,

comdominio DOUBLE(8,2),

IPTU DOUBLE(8,2),

area INTEGER,

tipo\_negociacao ENUM('alugar', 'vender') not null,

tipo\_uso ENUM('residencial', 'comercial') not null,

diferencial VARCHAR(250),

tipo\_imovel VARCHAR(60) not null

);

CREATE TABLE LOCATION (

codigo\_localizacao INTEGER auto\_increment PRIMARY KEY,

cidade VARCHAR(60) not null,

CEP VARCHAR(9)not null,

bairro VARCHAR(60) not null,

rua VARCHAR(60) not null,

uf CHAR(2) not null,

complemento VARCHAR(250),

numero INTEGER,

andar INTEGER not null,

codigo\_imovel INTEGER not null,

FOREIGN KEY(codigo\_imovel) REFERENCES IMOVEL (codigo\_imovel)

);

CREATE TABLE ANUNCIO (

codigo\_anuncio INTEGER auto\_increment PRIMARY KEY,

codigo\_imovel INTEGER not null,

tipo\_anunciante ENUM('proprietario', 'corretor', ‘imobiliária’),

data\_atual DATE not null,

imagem BLOB,

FOREIGN KEY(codigo\_imovel) REFERENCES IMOVEL (codigo\_imovel)

);

CREATE TABLE IMAGEM(

codigo\_img int auto\_increment primary key,

imagem blob

);

CREATE TABLE COMODO (

codigo\_qtdComodos INTEGER auto\_increment PRIMARY KEY ,

quartos INTEGER not null,

banheiros INTEGER not null,

suites INTEGER,

vagas INTEGER,

codigo\_imovel INTEGER not null,

FOREIGN KEY(codigo\_imovel) REFERENCES IMOVEL (codigo\_imovel)

);

CREATE TABLE LOGIN\_USUARIO (

login INTEGER auto\_increment PRIMARY KEY,

email VARCHAR(60) NOT NULL,

senha VARCHAR(8) NOT NULL,

id\_pessoa integer not null,

id\_client integer not null,

FOREIGN KEY(id\_pessoa) REFERENCES PESSOA(id\_pessoa),

FOREIGN KEY(id\_client) REFERENCES CLIENTE(id\_client)

);

CREATE TABLE CLIENT (

id\_client INTEGER auto\_increment PRIMARY KEY,

nome VARCHAR(30) NOT NULL,

sobrenome VARCHAR(30) NOT NULL,

email VARCHAR(60) NOT NULL,

telefone CHAR(13),

senha VARCHAR(8) NOT NULL,

);

CREATE TABLE PESSOA (

id\_pessoa INTEGER auto\_increment PRIMARY KEY,

tipo\_pessoa ENUM('PF', 'PJ') NOT NULL,

email VARCHAR(60) NOT NULL,

senha VARCHAR(8) NOT NULL,

telefone CHAR(13),

celular CHAR(14) NOT NULL,

nome VARCHAR(30),

sobrenome VARCHAR(30),

CPF CHAR(14),

razaõ\_social VARCHAR(80),

CNPJ CHAR(18),

codigo\_anuncio INTEGER NOT NULL,

FOREIGN KEY (codigo\_anuncio) REFERENCES ANUNCIO(codigo\_anuncio)

);

**DATA MANIPULATION LANGUAGE (DML)**

*COMANDO DE MANIPULAÇÃO SQL SISTEMA CORRETORA DE IMÓVEIS*

*Use db\_corretoraimv;*

**Exemplos de inserção de imóveis no banco:**

Insert into imovel (preco, condominio, iptu, area, tipo\_negociacao, tipo\_uso, tipo\_imovel, diferencial) values ("3.000", 250.00, 300.00, 80, "ALUGAR", "RESIDENCIAL", "COBERTURA", NULL);

Insert into imovel (preco, condominio, iptu, area, tipo\_negociacao, tipo\_uso, tipo\_imovel, diferencial) values ("750.000", NULL, 300.00, 100, "VENDER", "RESIDENCIAL", "CASA", NULL);

Insert into imovel (preco, condominio, iptu, area, tipo\_negociacao, tipo\_uso, tipo\_imovel, diferencial) values ("2200.00", NULL, 230.00, 45, "ALUGAR", "COMERCIAL", "SALA COMERCIAL", NULL);

Insert into imovel (preco, condominio, iptu, area, tipo\_negociacao, tipo\_uso, tipo\_imovel, diferencial) values ("900.000", 350.00, 400.00, 150, "VENDER", "RESIDENCIAL", "APARTAMENTO", NULL);

**Exemplos de inserção de anuncios no banco:**

insert into anuncio (codigo\_imovel, tipo\_anunciante, data\_atual) values (35, 1, 20200402);

insert into anuncio (codigo\_imovel, tipo\_anunciante, data\_atual) values (38, 1, 20200403);

insert into anuncio (codigo\_imovel, tipo\_anunciante, data\_atual) values (36, 2, 20190404);

insert into anuncio (codigo\_imovel, tipo\_anunciante, data\_atual) values (37, 1, 20190421);

insert into anuncio (codigo\_imovel, tipo\_anunciante, data\_atual) values (39, 1, 20200405);

**Exemplos de inserção de comodos no banco:**

*insert into comodo (quartos, banheiros, suites, vagas, codigo\_imovel) values (4, 3, 2, 2, 101);*

*insert into comodo(quartos, banheiros, suites, vagas, codigo\_imovel) values (5, 2, 2, 2, 102);*

*insert into comodo (quartos, banheiros, suites, vagas, codigo\_imovel) values (3, 2, 1, 3, 103);*

*insert into comodo(quartos, banheiros, suites, vagas, codigo\_imovel) values (4, 2, 2, 2, 104);*

**Exemplos de inserção de location no banco:**

*Insert into location (cep, rua, numero, bairro, cidade, uf, andar, complemento, codigo\_imovel) values (50061-630, "Rua das Groria", 430, "Candeias", "Jaboatão dos Guararapes", "PE", 8, "Proximo a farmácias e lojas", 110);*

*Insert into location (cep, rua, numero, bairro, cidade, uf, codigo\_imovel) values (50070-640, "Rua Itambé", 920, "Centro", "Recife", "PE", 107);*

*Insert into location (cep, rua, numero, bairro, cidade, uf, andar, complemento, codigo\_imovel) values (50060-630, "Rua das Graças", 720, "Boa Viagem", "Recife","PE", 20, "Proximo a padarias e shopping", 108);*

*Insert into location (cep, rua, numero, bairro, cidade, uf, complemento, codigo\_imovel) values (50061-630, "Rua das Groria", 430, "Candeias", "Jaboatão dos Guararapes", "PE", "Proximo a farmácias e lojas", 109);*

*Insert into location (cep, rua, numero, bairro, cidade, uf, codigo\_imovel) values (50070-640, "Rua Itambé", 920, "Centro", "Recife", "PE", 106);*

**Exemplos de inserção de cliente no banco:**

*Insert into client (nome, sobrenome, email, senha, telefone) values ("tiago", "firmino", "firminoTiago@hotmail.com", '12345678', '81988888888');*

*Insert into client (nome, sobrenome, email, senha, telefone) values ("rafaela", "vitoria", "rafaVitoria@gmail.com" , '12345678', '81988888888');*

*Insert into client (nome, sobrenome, email, senha, telefone) values ("Maria", "silva", "mariasilva@hotmail.com", '12345678','81988888888');*

*Insert into client (nome, sobrenome, email, senha, telefone, login) values ("tiago", "firmino", "firminoTiago@hotmail.com", '12345678', '81988888888');*

*Insert into client (nome, sobrenome, email, senha, telefone) values ("rafaela", "vitoria", "rafaVitoria@gmail.com" , '12345678', '81988888888');*

**Exemplos de inserção de login de usuários no banco:**

*Insert into login\_usuario(email, senha, id\_pessoa) values ("mariasilva@hotmail.com", "12345678", 132);*

*Insert into login\_usuario(email, id\_cliente) values ("firminoTiago@hotmail.com", "12345678", 211);*

*Insert into login\_usuario(email, senha, id\_pessoa) values ("rafaVitoria@gmail.com", "12345678", 132);*

*Insert into login\_usuario(email, senha, id\_cliente) values ("josealmeida@hotmail.com", "12345678", 147);*

*Insert into login\_usuario(email, senha, id\_pessoa) values ("mariasilva@hotmail.com", "12345678", 157);*

**Exemplos de inserção de pessoas no banco:**

*Insert into pessoa(tipo\_pessoa, email, senha, telefone, nome, sobrenome, cpf, id\_plano) values (1, "rafaVItoria@hotmail.com", "12345678", '81987777777', "Rafaela vitória", "almeida", '123.456.789-90', 348);*

*Insert into pessoa(tipo\_pessoa, email, senha, telefone, razao\_social, cnpj, id\_plano) values (2, "ImoveisFirmino@hotmail.com", "12345678", '81987777777', "Realizando sonhos", '12.345.678/9902-34', 347);*

*Insert into pessoa(tipo\_pessoa, email, senha, telefone, nome, sobrenome, cpf, id\_plano) values (1, "mariagloria@hotmail.com", "12345678", '81987777777',"maria gloria", "silva", '123.456.789-90', 358);*

*Insert into pessoa(tipo\_pessoa, email, senha, telefone, razao\_social, cnpj, id\_plano) values (2, "lardocelar@hotmail.com", "12345678" '81987777777', "lar doce lar", '12.345.678/9902-34', 359)*

**DATA QUERY LANGUAGE (DQL)**

*COMANDOS DE CONSULTA SQL SISTEMA CORRETORA DE IMÓVEIS*

**RELATÓRIOS**

* Listar os imóveis cadastrados em janeiro de 2019. Informar preço, área, cidade, bairro, ordenado pelo preço:

***SELECT i.preco, i.area, l.bairro, l.cidade, a.data\_atual FROM imovel AS i CROSS JOIN location AS l INNER JOIN anuncio AS a ON (month(data\_atual) = '01')***

* Listar pagamentos realizados para determinado plano de código 350:

**SELECT \*FROM plano INNER JOIN forma\_pagamento WHERE id\_plano = 350;**

* Listar os cômodos do imóvel de código 145:

**SELECT c.quartos, c.banheiros, c.suites, c.vagas from comodo AS c WHERE codigo\_imovel = 145**

* Informar a quantidade de anúncios por tipo anunciante:

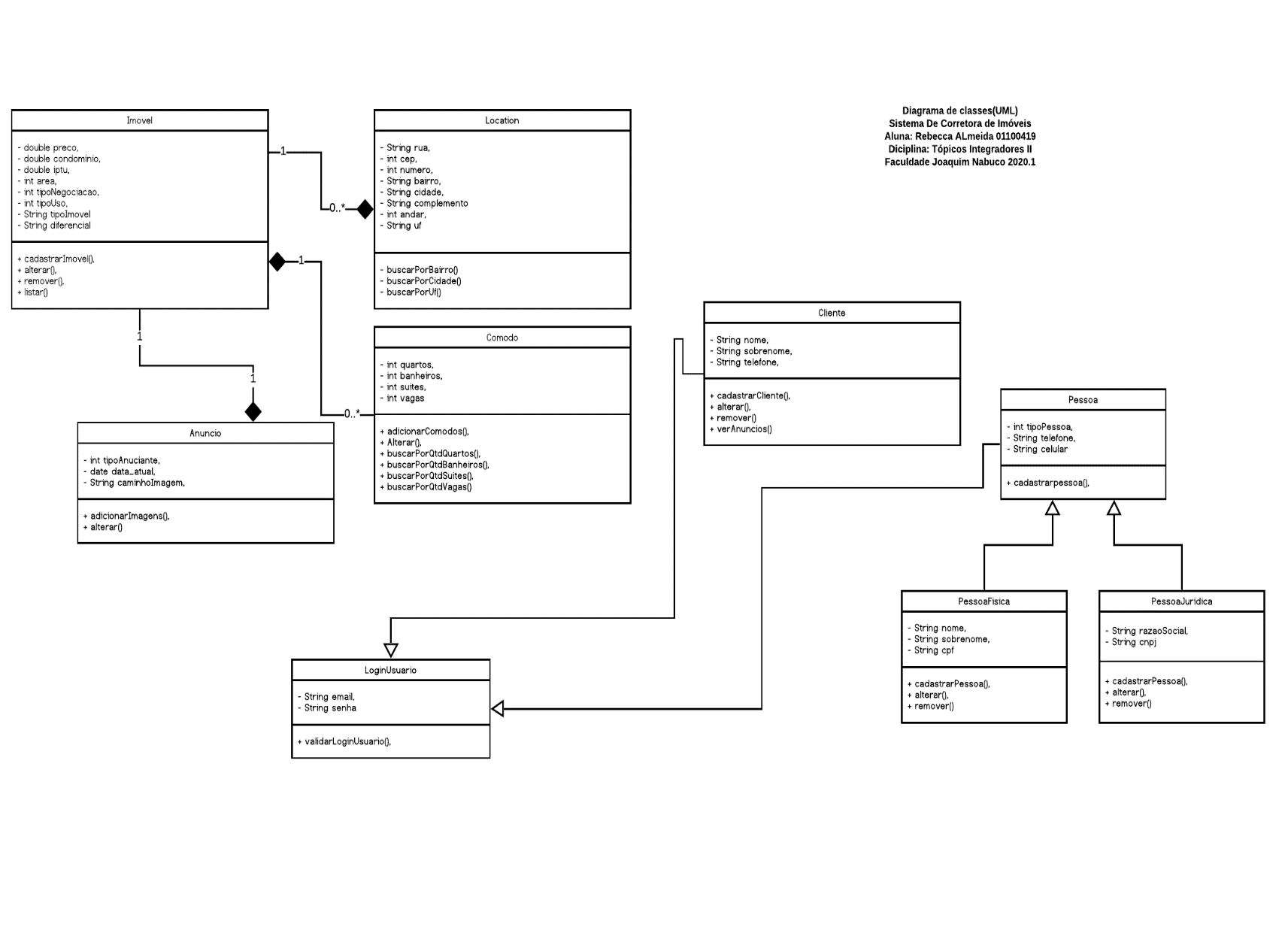
**SELECT COUNT(\*) AS contagem FROM anuncio WHERE tipo\_anunciante = 'proprietario';**

**SELECT COUNT(\*) AS contagem FROM anuncio WHERE tipo\_anunciante = 'corretor'**

* Listar os nomes, email e telefone dos clientes, ordenado pelo nome.

**SELECT nome, email, telefone FROM client**

**ANÁLISE E DESIGN**

 **DIAGRAMA DE CLASSES**

**ARQUITETURA DE SOFTWARE**

**PADRÃO DE PROJETO MODEL-VIEW-CONTROLLER(MVC)**

Normalmente usados para o desenvolvimento de interfaces de usuários que divide uma aplicação partes (camadas/componentes) interconectadas. Isto é feito para separar representações de informação internas dos modos como a informação é apresentada para e aceita pelo usuário, levando ao desenvolvimento paralelo de maneira eficiente.

### Camada de modelo ou da lógica da aplicação (Model)

Está camada vai obter todas as classes do sistema tais como: cliente, anuncio, imóvel, comodo, location, pessoa e loginUsuario e seus atributos. Contendo também todos os métodos a serem implementados e os cruds necessários. Além das regras de negocio que são imprecindiveis.

### Camada de apresentação ou visualização (View)

Está camada vai obter todas as telas de interação com o usuário e conexão com o banco, tais como: login, cadastro de cliente, cadastro e pessoa física e pessoa jurídica, cadastro de imóveis, home, cadastro de endereço e adição de imagens.

### Camada de controle ou controlador (Controller)

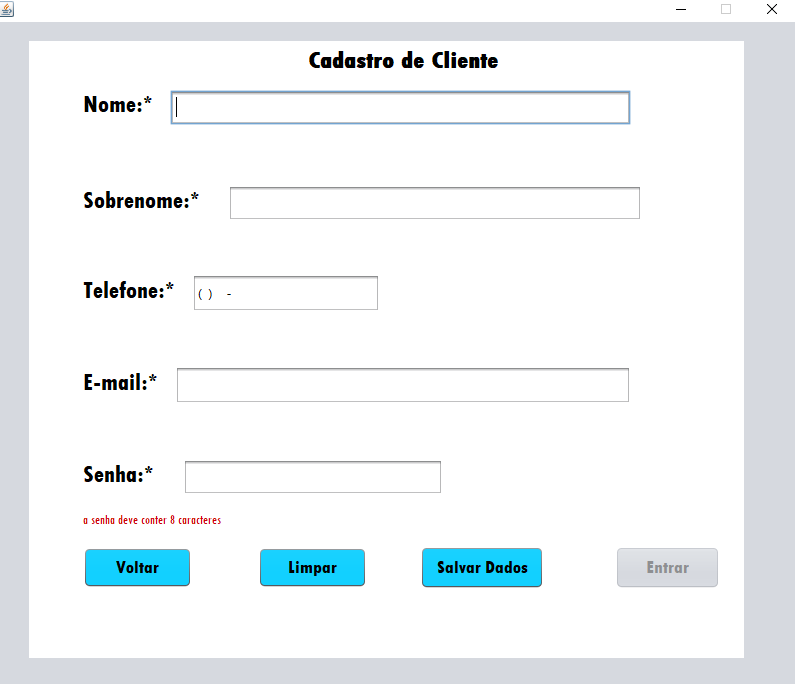
Está camada vai obter todas as classes de controle de dados aquelas que são necessárias para obtenção e inserção de dados no sistema, tais como: imóvel, comodo, location, pessoaFisica, pessoaJuridica, cliente e login.

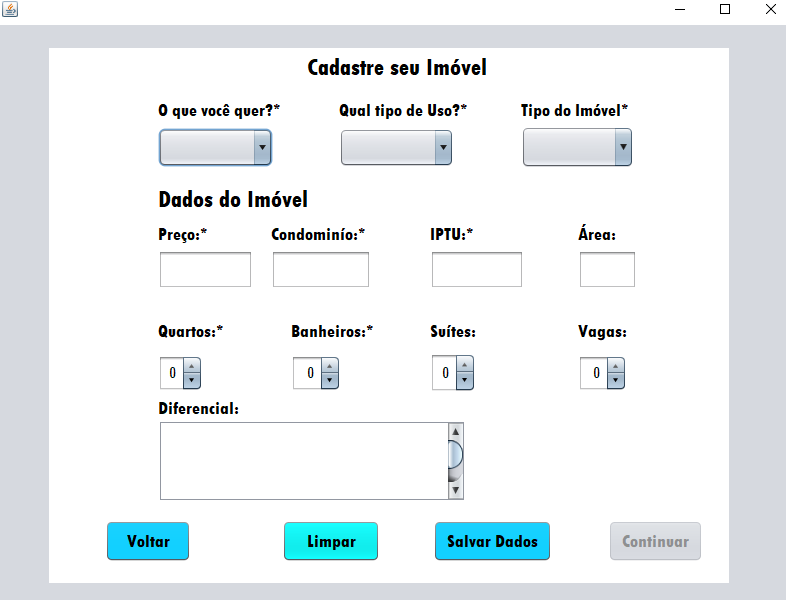
**PRÓTOTIPOS**

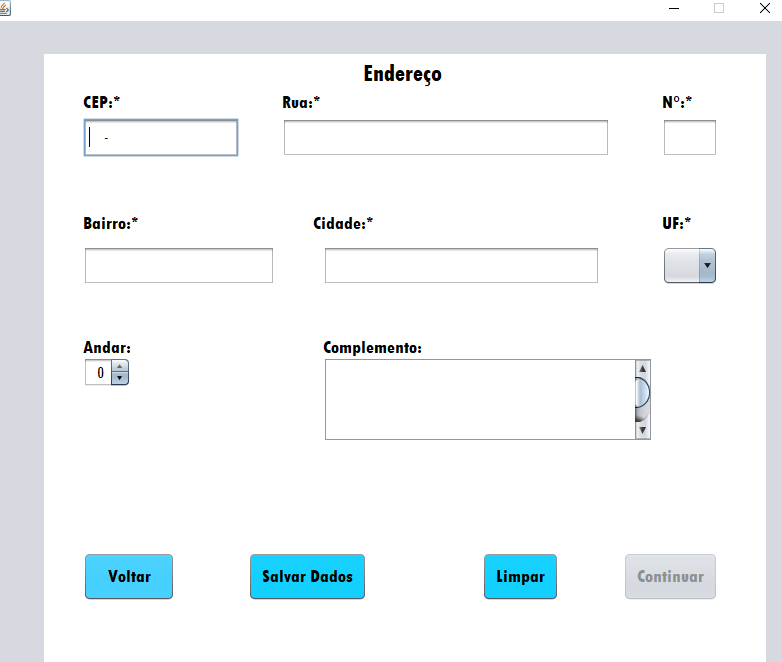
**TELAS E CLASSES DA CAMADA VIEW**

CLASSES DA CAMADA VIEW

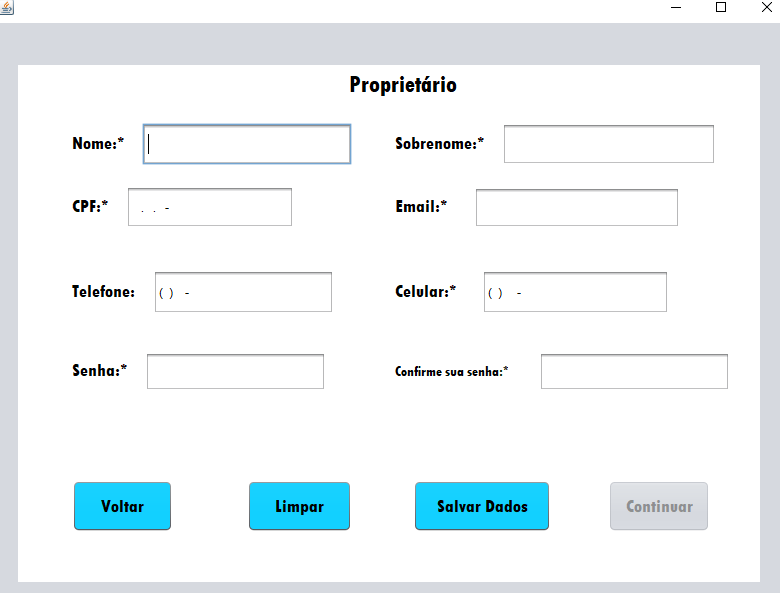
[**https://github.com/rebeccaalmeida1995/camada-view-corretoraimv.git**](https://github.com/rebeccaalmeida1995/camada-view-corretoraimv.git)



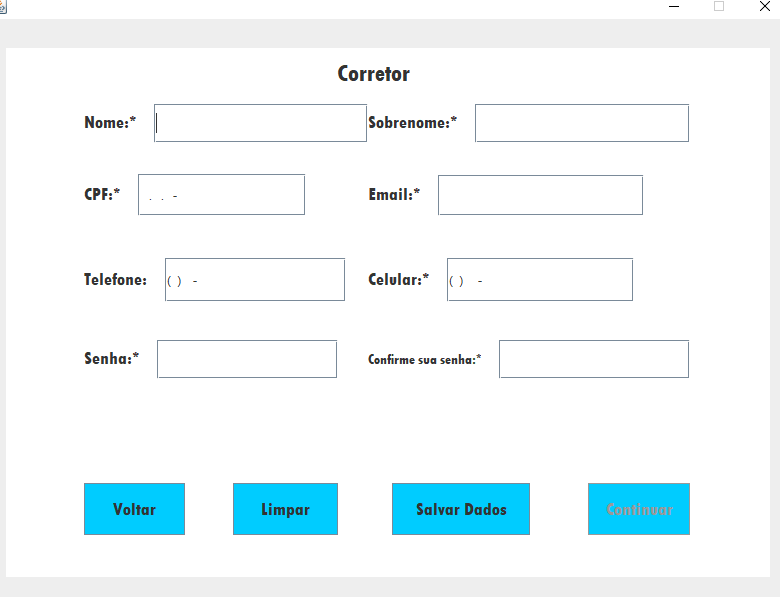
TELA DE CADASTRO DE IMÓVEL

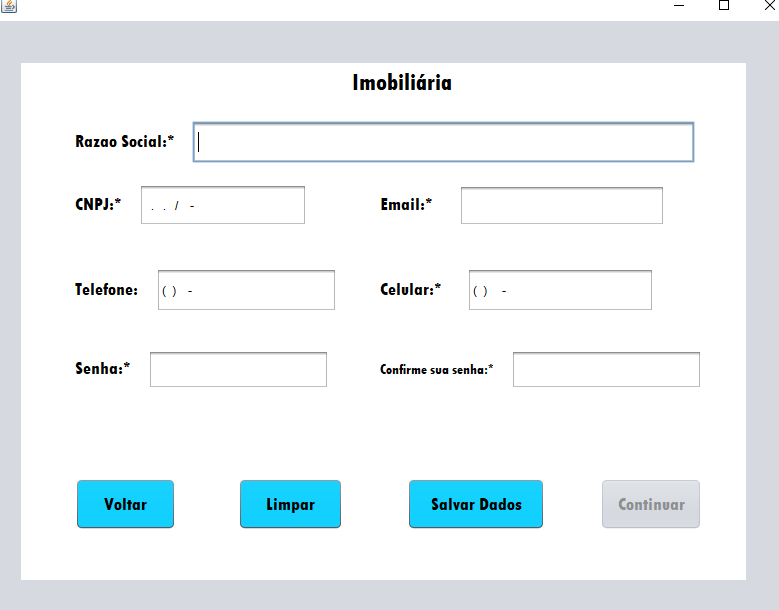
TELA DE CADASTRO DE ENDEREÇO

TELA DE CADASTRO DE PESSOA FISICA PROPRIETRÁRIO

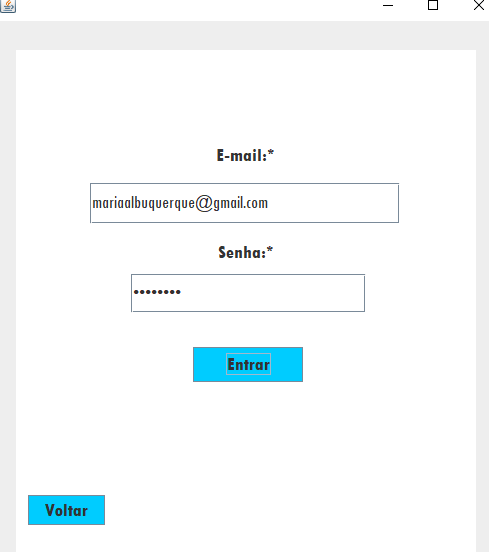


TELA CADASTRO PESSOA FÍSICA CORRETOR



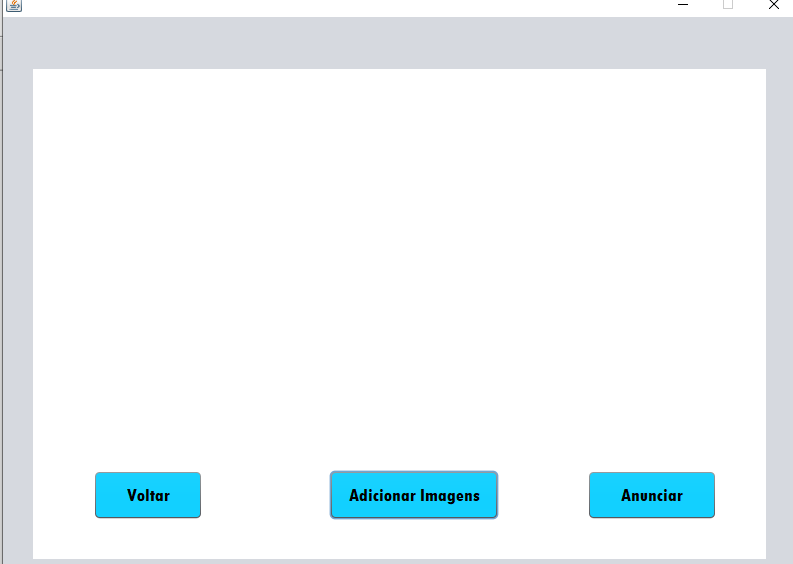
TELA DE CADASTRO PESSOA JURÍDICA IMOBILIÁRIA

TELA DE LOGIN DE USUÁRIO



TELA DE SELEÇÃO DE PESSOA

TELA DE ADIÇÃO DE IMAGEM



**CLASSES DA CAMADA MODEL**

[**https://github.com/rebeccaalmeida1995/camada-view-corretoraimv.git**](https://github.com/rebeccaalmeida1995/camada-view-corretoraimv.git)

**CLASSES DA CAMADA CONTROLLER**

[**https://github.com/rebeccaalmeida1995/camada-controller-corretoraimv.git**](https://github.com/rebeccaalmeida1995/camada-controller-corretoraimv.git)

**CLASSES DE CONEXÃO COM O BANCO**

[**https://github.com/rebeccaalmeida1995/conexao-mysql-corretoraimv.git**](https://github.com/rebeccaalmeida1995/conexao-mysql-corretoraimv.git)

**DOCUMENTAÇÃO DO PROJETO**

[**https://github.com/rebeccaalmeida1995/projeto-corretora-de-imoveis.git**](https://github.com/rebeccaalmeida1995/projeto-corretora-de-imoveis.git)

REFERÊNCIAS

<https://github.com/>

<https://www.youtube.com/watch?v=yXwAIFKuVqc&t=140s>

<https://www.youtube.com/watch?v=faPR87RX2c0>

<https://www.devmedia.com.br/>

[https://app.lucidchart.com/documents/edit/d3d0de4c-9d79-4760-9b8d-4d24c5e52e02/0\_0?beaconFlowId=14FC4330BCF8904E#](https://app.lucidchart.com/documents/edit/d3d0de4c-9d79-4760-9b8d-4d24c5e52e02/0_0?beaconFlowId=14FC4330BCF8904E)

<https://pt.stackoverflow.com/>

<https://www.guj.com.br/>

[www.bosontreinamentos.com.br](http://www.bosontreinamentos.com.br)