UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
CAMPUS PAU DOS FERROS
DEPARTAMENTO DE ENGENHARIAS E TECNOLOGIA

ALAN ALMEIDA DA SILVA - 2024011434
BRUNO VICTOR PAIVA DA SILVA - 2020011035
CAIO VINÍCIUS PESSOA GOMES - 2024011364
DALTON FIRMINO CAMPOS - 2024011414
PAULINA JÚLIA COSTA DE OLIVEIRA - 2023023644

QUARTZO - SISTEMA DE GERENCIAMENTO DE IMOBILIÁRIA

PAU DOS FERROS - RN
2025

Introdução

Este documento apresenta a estrutura e as funcionalidades do sistema de gerenciamento imobiliário desenvolvido utilizando o framework Django. Com foco no backend robusto e escalável, o sistema foi projetado para otimizar a administração de aluguéis de imóveis, oferecendo ferramentas que simplificam processos, centralizam informações e garantem uma gestão eficiente e organizada.

1. ## Contrato

   A classe **Contrato** modela os contratos associados ao sistema. Representando os detalhes de um contrato entre um cliente e a administração de um imóvel, abrangendo informações sobre o tipo de contrato, forma de pagamento, período de validade e status.

I. Campos:

- cliente (ForeignKey): Relaciona o contrato a um cliente no sistema.
- imóvel (ForeignKey): Relaciona o contrato a um imóvel registrado.
- tipo_contrato (CharField): Indica o tipo de contrato (Aluguel ou Venda).
- valor (FloatField): Especifica o valor financeiro envolvido
- forma_pagamento (CharField): Define a forma de pagamento utilizada pelo cliente.
- data_inicio (DateField): Data de início do contrato.
- data_fim (DateField): Data de término do contrato.
- status (CharField): Representa o estado atual do contrato.
- observacoes (TextField): Campo para informações adicionais ou notas relevantes ao contrato.
- data_criacao (DateTimeField): Registra a data e a hora em que o contrato foi criado no sistema.
- data_atualizacao (DateTimeField): Registra a última data e hora de atualização do contrato no sistema.

II. Métodos:

- str()
  - Descrição: Retorna uma representação do objeto como string.
- clean()
  - Válida se o contrato pode ser renovado com base nos campos de status e datas.

2. Pagamento

A classe **Pagamento** modela os pagamentos realizados pelos clientes relacionados a contratos. Ela armazena informações sobre valores pagos, datas de pagamento e associações com contratos e clientes, permitindo o gerenciamento detalhado e rastreamento financeiro no sistema.

I. Campos:

- contrato (ForeignKey): Relaciona o pagamento a um contrato específico, permitindo associar transações financeiras ao imóvel e usuário .
- usuário (ForeignKey): Relaciona o pagamento a um usuário do sistema
- valor_pago (FloatField): Indica um valor pago em uma transação específica.
- data_pagamento (DateTimeField): Registra a data em que o pagamento foi efetuado/será.
- data_criacao (DateTimeField): Registra a data e a hora em que o pagamento foi criado no sistema (preenchido automaticamente).
- data_atualizacao (DateTimeField): Registra a última data e hora de atualização do registro.

II. Métodos

- str()
  - Descrição: Retorna uma representação do objeto como string.

- calcular_valor_pendente() Calcula o valor pendente de pagamento em casos de aluguel, soma todos os pagamentos relacionados ao contrato e subtrai do total pago do valor mensal, retornando o pendente.

3. AlterUserEmail

A classe **AlterUserEmail** altera as propriedades do campo de e-mail do modelo User do Django, tornando-o único e obrigatório. Isso evita problemas no login e garante a consistência dos dados, impedindo e-mails duplicados ou vazios.

4. Profile/User

A classe **Profile** Representa informações relacionadas ao usuário do sistema que no projeto usamos o user do django, por isso a relação do profile, com informações pessoais e ações relacionadas à gestão de imóveis e contratos, ele é um complemento do user django.

I. Campos:
- usuário (OneToOneField): Nome relacionado ao usuário.
- telefone(String): Número de telefone do usuário.
- cpf: CPF do usuário.
- superusuario(Booleano) Tipo de usuário(Administrador ou Cliente) já é uma funcionalidade do próprio Django..
- email(Email) Endereço de email do usuário, padrão do django também.
- senha(String): Senha do usuário que é criptografada, padrão do django.

II. Métodos
- str()
  - Descrição: Retorna uma representação do objeto como string.

5. Imóvel

A classe **Imóvel** representa um Imóvel disponível para venda ou aluguel. Ela armazena informações sobre ele, como endereço, tipo, valor, descrição e etc.. É onde podemos realizar o cadastro de imóveis.

I. Campos:
- endereco (CharField): Endereço do imóvel.
- tipo (CharField): Representa se o imovél cadastrado é para aluguel ou venda.
- valor (FloatField): Representa o valor do imóvel cadastrado.
- status (CharField): Status atual do imóvel
- descrição (CharField): Representa uma descrição do mesmo.
- usuario_criacao (ForeignKey): Usuário que cadastrou no sistema
- data_criacao (DateTimeField): Representa a data de cadastro do imovel no sistema, pega de forma automática.
- data_atualizacao (DateTimeField): Representa a Data da última atualização no sistema.

II. Métodos
- save()
  - Descrição: Sobrescrita do método save para realizarmos ações de criação ou atualização automaticamente.
- str()
  - Descrição: Retorna uma representação do objeto como string, mostrando o endereço e tipo do imóvel na lista de imóveis.