Django Framework do Zero

Como Criar um Aplicativo Web

*Canal **youtube.com/@VamosCodar** - Wanderson Reis*

09 - Área Administrativa

Requisitos

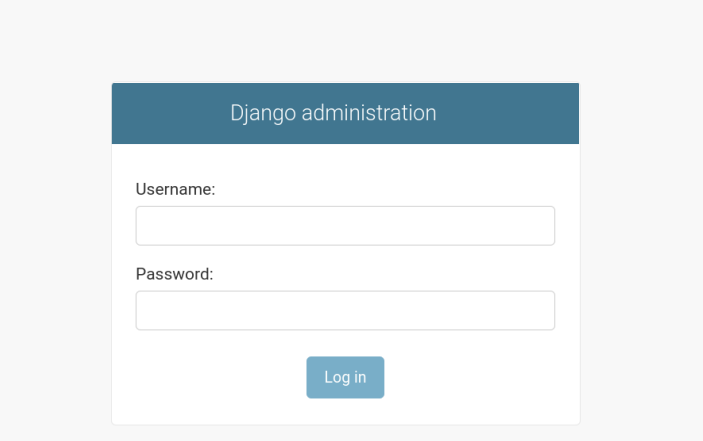
1) Aulas 01 a 08 - Django Framework do Zero - Playlist: https://www.youtube.com/playlist?list=PLFOqHo8oIjzewcT23HCxJV0xWO451CTJe

Introdução

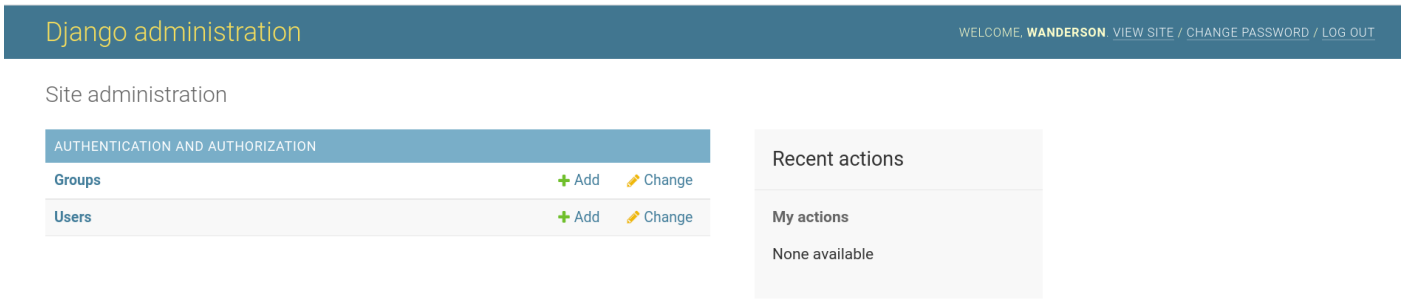
Um dos recursos mais interessantes do Django é uma área administrativa gerada automaticamente para cada novo projeto. A partir dos modelos, uma interface para gerenciamento completo do conteúdo fica disponível para os usuários administrativos. Os usuários criados usando o model User do Django podem ter o perfil de **staff** o que garante o acesso a área administrativa. Outro perfil disponível é o **superuser**, este além de ter acesso a área administrativa também terá todas as permissões para gerenciamento do conteúdo. Nesta aula vamos aprender a usar a área administrativa para acelerar o desenvolvimento.

Nesta aula:

1) Criar um usuário administrativo
2) Acessar a área administrativa
3) Explorar a área administrativa
4) Incluir o Model Question na área administrativa
5) Model CustomUser na área administrativa



Tela de login



Área administrativa - primeiro acesso

A documentação completa sobre área administrativa do Django pode ser lida em:
https://docs.djangoproject.com/en/4.2/ref/contrib/admin/

IMPORTANTE: Lembre-se de ativar o ambiente virtual Python (venv) antes de executar quaisquer comandos no terminal.

Abra um novo terminal pelo menu "Terminal" -> "New Terminal" (Command Prompt), e execute os comandos:

.\venv\Scripts\activate.bat **(ENTER)**

Se estiver no Linux:
source venv/bin/activate **(ENTER)**

Passo 1: Criar um usuário administrativo

Igualmente as demais rotinas de gestão do projeto, a criação de usuário também pode ser realizada pela linha de comando. Vamos criar um usuário com perfil **superuser**.

Execute o comando abaixo e siga as instruções (informando login, e-mail, senha e confirmação da senha):

python manage.py createsuperuser (ENTER)

Veja abaixo um exemplo de criação de um superusuário. Observe que o utilitário informa sobre a segurança da senha, mas permite confirmar o cadastro mesmo com uma senha "fraca" ou "comum".

```
└─ wanderson >> python manage.py createsuperuser
Usuário: wanderson
Endereço de email: wasare@gmail.com
Password:
Password (again):
Esta senha é muito comum.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
```

Outro recurso relacionado ao controle de acesso é o comando para alteração de senha, especialmente útil em casos de esquecimento e desde que tenha acesso ao terminal / servidor onde a aplicação estiver instalada. Para modificar a senha bastaria executar o comando (no terminal:

python manage.py changepassword login-usuário (ENTER)

Passo 2: Acessar a área administrativa

Depois de criado o usuário, o acesso a área administrativa depende da aplicação estar online, desta forma é necessário subir o servidor local:

python manage.py runserver (ENTER)

Agora no navegador acesse http://127.0.0.1:8000/admin

/admin é a rota padrão da área administrativa.

Passo 3: Explorando a área administrativa

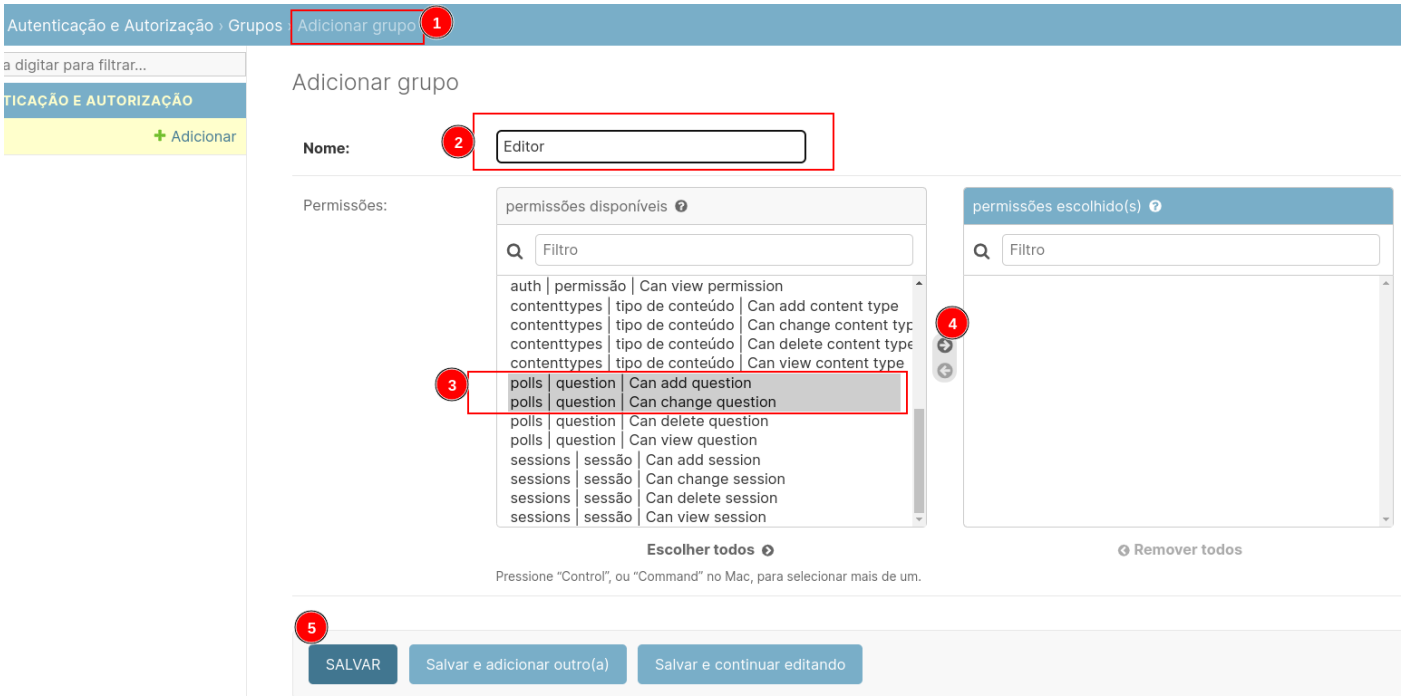
A área administrativa em geral fica organizada por módulo / contexto. Inicialmente apenas os models que permitem gerenciar usuários e grupos ficam habilitados. É possível habilitar outros models como veremos em outro passo mais à frente. Os models quando ativados na área administrativa já contam automaticamente com um CRUD básico (CREATE, READ, UPDATE e DELETE) automático para a gestão do conteúdo deste models. Além disso o Django já cria todo uma estrutura de permissões para cada Model considerando as operações básicas de Add, Change, Delete e View, respetivamente Create, Update, Delete e Read). Ou seja, são as permissões do CRUD com palavras sinônimas.

Agora clique em "**Grupos**" e observe que abre uma lista vazia pois não temos nenhum grupo criado. Se você fez a aula para incluir novos campos no registro do usuário o model **User** será personalizado e por isso não aparece na lista do App "Auth" (módulo Autenticação e Autorização padrão do Django).

Observe também que o painel administrativo já está traduzido (devido a configuração de idioma que fizemos em outra aula).

Agora na tela que lista os grupos clique no botão "**ADICIONAR GRUPO +**" e crie um grupo chamado "**Editor**" e atribua as permissões "**Can add question**" e "**Can change question**"

Siga os passos demonstrados na próxima imagem.



Criar um Grupo Editor com as permissões Add e Change Question

Passo 4: Incluir o Model Question na área administrativa

Considerando que o objetivo de um Framework é permitir o reuso e o desenvolvimento rápido de recursos usando padrões, o Django implementa a capacidade de estender através de herança toda a capacidade da área administrativa para qualquer model da aplicação.

Para habilitar os models da sua aplicação é necessário criar um novo arquivo chamado **admin.py** dentro da pasta do módulo/app. No nosso caso vamos habilitar o model Post do app blog na área administrativa.

Edite o arquivo **django-do-zero/polls/admin.py** substitua com o conteúdo abaixo (**atenção apenas ao conteúdo novo**):

```
# importa o decorator admin que habilita características de admin no nosso model
from django.contrib import admin
from polls.models import Question # importa o model que vamos habilitar no admin site

@admin.register(Question) # registar o model Question e habilita
class QuestionAdmin(admin.ModelAdmin): # configura o model Question no admin site.
    list_display = ('id', 'question_text', 'pub_date') # campos para listar.
    list_filter = ('pub_date',) # campos que poderão ser filtrados.
```

(**salve o arquivo**).

Retorne a área administrativa logada e observe a nova seção **Polls** (correspondente ao app/módulo) e o Model **Questions** habilitado com o CRUD básico gerado automaticamente.

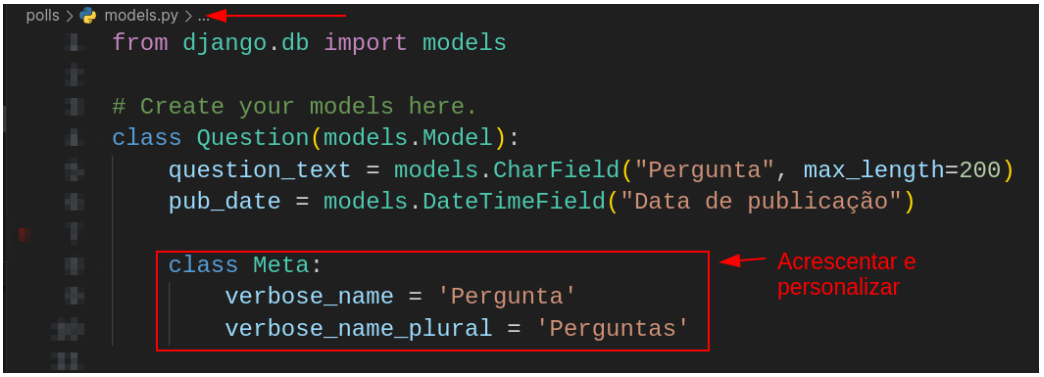
Clicando em Questions é possível fazer a gestão completa (Criar, Visualizar, Atualizar e Excluir) as perguntas das enquetes como pode ser observado na imagem abaixo.

Para complementar vamos aprender a personalizar o nomes do nosso App e dos modelos, assim podemos deixar mais amigável a apresentação.

A alteração do nome do App (para exibição) pode ser feita editando o arquivo **app.py** dentro da pasta módulo/app desejada. É necessário incluir um novo atributo chamado "**verbose_name**" e configurar o nome conforme o desejado. Veja o exemplo na imagem.

(**salve o arquivo**).

O model pode ser renomeado (para exibição) no arquivo models.py. É possível configurar um nome personalizado no singular e no plural. Veja na imagem a seguir.



(**salve o arquivo**).

Feito isso, retorne a área administrativa, atualize a página e veja o resultado.

Passo 5: Model CustomUser na área administrativa

Se você fez a aula que incluiu os novos campos Data de Nascimento e CPF no registro de usuários precisa executar este passo para conseguir administrar os usuários usando a área administrativa.

Para habilitar o model CustomUser da aplicação é necessário criar/editar o arquivo chamado **admin.py** dentro da pasta do **django-do-zero/accounts/**. Basta uma configuração indicando os models, os campos a exibir e pronto, automaticamente o novo Model fica acessível na área administrativa.

Editar o arquivo **django-do-zero/accounts/admin.py** substitua com o conteúdo abaixo (**atenção apenas ao conteúdo novo**):

```
# importa o decorator admin que habilita características de admin no nosso model
from django.contrib import admin
from accounts.models import CustomUser # importa o model para habilitar no admin site

@admin.register(CustomUser) # registar o model CustomUser e habilita
class UserAdmin(admin.ModelAdmin):
    list_display = ('username', 'is_active', 'is_staff', 'is_superuser') # campos para listar.
    list_filter = ('username', 'cpf')  # campos que poderão ser filtrados.
```

(**salve o arquivo**).

Retorne a área administrativa logada e observe a nova seção **Accounts** (correspondente ao app/módulo) e o Model "**Usuários"** (que é o **CustomUser** que já foi traduzido automaticamente). Está habilitado com o CRUD básico gerado automaticamente.

Agora clique em "**Usuários**" e na lista de usuários que exibe os campos definidos no arquivo **admin.py**, clique no nome do usuário criado no Passo 1. O formulário de edição do usuário será carregado. Neste formulário é possível acrescentar ou modificar os campos permitidos e que serão manipulados adequadamente conforme o tipo de dado. Contudo observando o formulário de edição do usuário pela área administrativa o campo senha ficou em texto puro. E neste caso se precisar alterar a senha por este caminho não vai funcionar.

Problema na edição do usuário por causa do Model User que é personalizado.

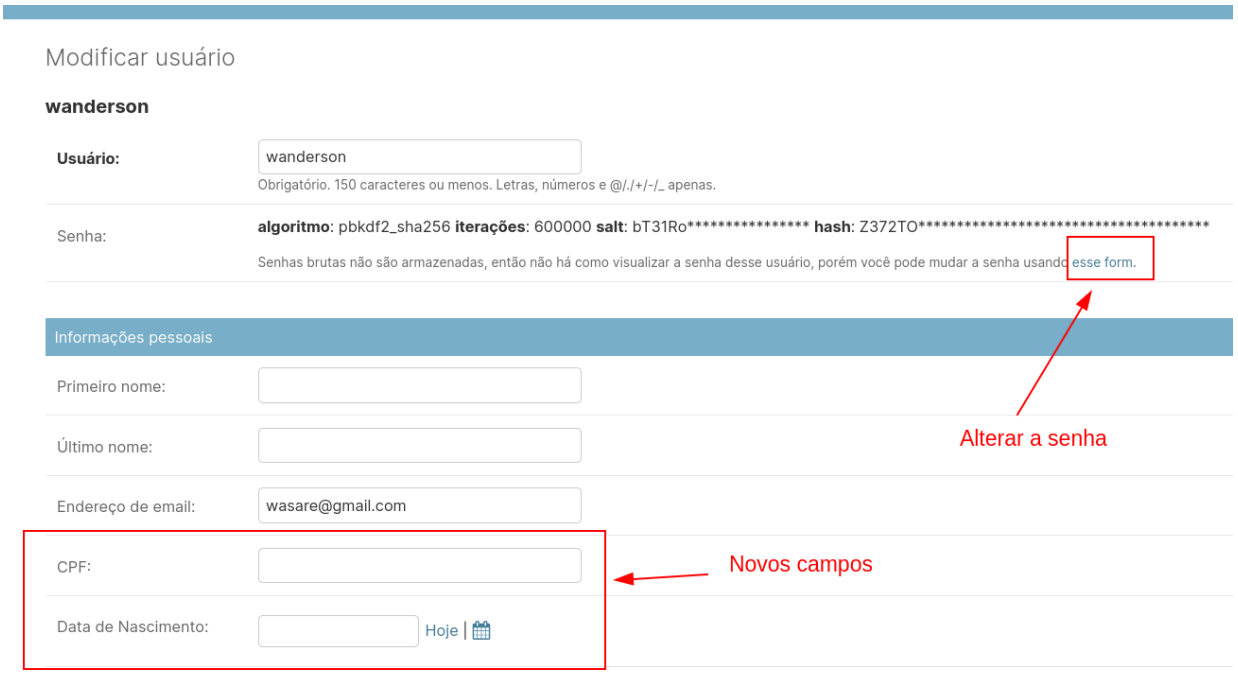
Para resolver esta questão vamos usar as configurações padrão do Django para tratar o Model User na área administrativa. Basicamente vamos copiar a estrutura do código do arquivo **admin.py** do App auth (nativo do Django). Este arquivo fica dentro de **django.contrib.auth** vamos copiar e adaptar para o nosso cenário (usando o Model **CustomUser**).

No repositório do Django no GitHub o arquivo é este:
https://github.com/django/django/blob/main/django/contrib/auth/admin.py

O código já adaptado para o nosso caso (Model CustomUser e novos campos CPF e Data de Nascimento) está disponível neste Gist (copie e substitua o seu **accounts/admin.py**:
https://gist.github.com/wasare/81b929cb4f4934033ce2b3630aafb89e

Modifique ou acrescente alguma informação do usuário e clique em salvar.

Feita a modificação ficará como na imagem abaixo (detalhe) de alterar a senha do usuário e campos novos (além dos demais campos, e a estrutura padrão de grupos e permissões).



Edição de usuário adaptado - no detalhe o link para modificar a senha e novo campos

É importante observar que os models **Group e CustomUser** (este herdou tudo de AbstractUser) já implementam uma estrutura completa para gestão de usuários, grupos e permissões que pode ser facilmente integrada no seu aplicativo feito em Django.