ROTEIRO DO DESAFIO 1

Site de Histórico de Matérias

Uma vez que já temos as dependências instaladas (python3, pip3 e django) vamos criar a nossa aplicação de visualização do histórico de matérias.

Parte 1 - Criando o esqueleto do projeto

1) Vamos começar executando o comando **django-admin** para iniciar o nosso projeto. Depois, vamos entrar na pasta do projeto, recém-criada
(DICA: use o comando **ls** dentro da pasta para verificar os arquivos que foram criados!)

```
$ django-admin startproject meuhistoricoufba
$ cd meuhistoricoufba
```

2) Vamos subir o web server de desenvolvimento para testar se tudo está executando corretamente. Para inicializar o servidor, vamos utilizar o script de gestão do projeto, o arquivo **manage.py**. Você perceberá que mais tarde utilizaremos esse arquivo novamente.

```
$ python3 manage.py runserver
```

3) Uma vez tendo executado o comando sem erros, coloque no seu browser o endereço: **http://127.0.0.1:8000/**

4) Agora vamos criar um aplicativo que fará parte do nosso projeto. Para cada módulo do seu website (ex: blog, fórum, catálogo…) é interessante que você crie um aplicativo diferente, já que um aplicativo terá as suas próprias models e views, dentre outros arquivos.
(DICA: depois de criar o aplicativo, execute **ls** na pasta **historico/**, para verificar os novos arquivos que foram criados!)

```
$ python3 manage.py startapp historico
```

5) Apesar do nosso aplicativo ter sido criado, ele ainda não foi registrado no nosso projeto. Para corrigir esse problema, vamos editar o arquivo global de configuração do projeto (**meuhistoricoufba/settings.py**) para incluir a linha do nosso app em **INSTALLED_APPS**.
(DICA: aproveite para ler o conteúdo pré-existente em INSTALLED_APPS! Temos aplicações como admin, auth e sessions. Reconheceu alguma delas?)

```
INSTALLED_APPS = [
    <linhas de codigo...>
    'historico',
]
```

6) Na pasta do projeto existe um arquivo **meuhistoricoufba/urls.py**. É interessante que você conecte esse arquivo de forma que você possa fazer o mapeamento de URLs diretamente no aplicativo associado, **historico**. Abra o arquivo no editor de texto e adicione no final, dentro da lista **urlpatterns** (PS: lembre de fazer também o import no include):

```
from django.conf.urls import include

urlpatterns += [
    path('historico/', include('historico.urls')),
]
```

7) Crie o arquivo **historico/urls.py** e preencha com o conteúdo a seguir. Por enquanto, a lista de urls está vazia. Daqui a pouco iremos inserir uma url nessa lista e criar a nossa primeira rota!

```
from django.urls import path
from historico import views

urlpatterns = [

]
```

8) Antes de testarmos, precisamos executar uma migração no banco de dados para atualizar nosso banco sobre as aplicações instaladas. Execute os comandos a seguir (**makemigrations** cria as migrações mas não aplica e **migrate** aplica as migrações)
(DICA: você precisará executar esses comandos sempre que fizer alguma alteração nas suas models!)

```
$ python3 manage.py makemigrations
$ python3 manage.py migrate
```

Parte 2 - Criando uma funcionalidade completa

Agora vamos criar a nossa primeira funcionalidade completa - vamos criar uma **model** para as matérias do nosso histórico, um **template** para uma página de visualização de todas as matérias cadastradas no sistema, uma **view** para fazer as requisições HTTP e uma **url** para mapear nossa funcionalidade.
Como faremos somente a parte READ das operações do CRUD, é interessante utilizarmos o Django admin para fazer a operação CREATE, a fim de verificar que os objetos realmente estão sendo inseridos no banco.

1) Criando a Model - modelos são definidos no arquivo models.py de uma aplicação. Vamos abrir o arquivo **historico/models.py** e inserir as seguintes linhas no final do arquivo:

```
class Materia(models.Model):
      # Campos
      codigo = models.CharField(max_length=10, help_text="Coloque o
código da matéria, como consta no SIAC")
      nome = models.CharField(max_length=100, help_text="Coloque o
nome da matéria, como consta no SIAC")
      carga_horaria = models.IntegerField(help_text="Coloque a carga
horária da matéria, como consta no SIAC")

      OPCOES_SEMESTRE = (
      ('2018.1', '2018.1'),
      ('2018.2', '2018.2'),
      ('2019.1', '2019.1'),
      ('2019.2', '2019.2'),
      )

      semestre = models.CharField(
      max_length = 6,
      choices=OPCOES_SEMESTRE,
      help_text='Selecione o semestre que você cursou essa matéria',
      )

      # Metadados
      class Meta:
      ordering = ['semestre']

      # Metodos
      def __str__(self):
      return f'{self.codigo} - {self.nome}'
```

2) Lembre-se de executar novamente as migrações:

```
$ python3 manage.py makemigrations
$ python3 manage.py migrate
```

3) Criando a View - para simplificar nossa tarefa, vamos criar uma view baseada ńa classe genérica **ListView**. Nossa classe herdará dela, já que ela já implementa a maior parte das funcionalidades que precisamos e segue as boas práticas do Django.
Abra o arquivo **historico/views.py** e copie o código no final do arquivo:

```
from django.shortcuts import render

from historico.models import Materia

def lista_materias(request):
    materias = Materia.objects.all()
    return render(request, 'historico/materia_list.html', context =
{'materia_list': materias})
```

4) Criando o Template - primeiro, vamos precisar criar o caminho padrão para os templates da aplicação historico. Execute o comando:

```
$ mkdir templates/historico
```

5) Ainda dentro da pasta raiz, crie o arquivo **templates/historico/materia_list.html** e copie o conteúdo a seguir:

```
<h1>Histórico de Matérias</h1>

  {% if materia_list %}
  <table style="width:100%;text-align: left;">
      <tr>
      <th>Semestre</th>
      <th>Código</th>
      <th>Nome</th>
      <th>Carga Horária</th>
      </tr>
```

```
      {% for materia in materia_list %}
      <tr>
            <td> {{ materia.semestre }} </td>
            <td> {{ materia.codigo }} </td>
            <td> {{ materia.nome }} </td>
            <td> {{ materia.carga_horaria }} </td>
      </tr>
      {% endfor %}
  </table>

  {% else %}
      <p>Nenhuma matéria foi cadastrada ainda.</p>
  {% endif %}
```

6) Para registrar o diretório dos templates, vá no arquivo **settings.py** e insira as seguinte linha e modifique a outra com o seguinte código.

```
TEMPLATES_DIR = os.path.join(BASE_DIR, 'templates')

TEMPLATES = [
    {
        ....
        'DIRS': [TEMPLATES_DIR]
        ....
    },
]
```

7) Criando a rota - abra o arquivo **historico/urls.py** e acrescente

```
urlpatterns = [
  path('materias/', views.lista_materias, name='materias'),
]
```

8) E agora, será que acabou? Execute novamente o comando para subir o servidor (no diretório **meuhistoricoufba/**) e acesse o endereço no seu navegador http://127.0.0.1:8000/historico/materias/

```
$ python3 manage.py runserver
```

Se tudo ocorreu como planejado, **parabéns!** Seria mais interessante se a página exibisse matérias cadastradas, né? Como não iremos implementar todas as funcionalidades do CRUD aqui, vamos criar matérias utilizando o Django admin.

9) O aplicativo de administração do Django utiliza os models da sua aplicação para criar, automaticamente, uma área que você pode utilizar para fazer CREATE, READ, UPDATE e DELETE de registros. Abra o arquivo **historico/admin.py** e adicione as seguintes linhas no final, para registrar a nossa model Materia no Django admin:

```
from historico.models import Materia

admin.site.register(Materia)
```

10) Agora, precisamos criar um superusuário para acessar a interface de administração. Execute o seguinte comando, no mesmo diretório que **manage.py**, para criar o superusuário. Você será solicitado a digitar um nome de usuário, endereço de e-mail e senha forte.
(DICA: Não esqueça de reiniciar o servidor de desenvolvimento após a criação do super usuário!)

```
$ python3 manage.py createsuperuser
```

11) Com o servidor de desenvolvimento executando, entre no endereço **http://127.0.0.1:8000/admin/** e informe suas credenciais de superusuário.

12) Uma vez dentro do admin, adicione quantas matérias quiser.
(DICA: Quando estiver fazendo isso, compare com os campos que foram inseridos no models.py! Notou qual o papel do **help_text**? Percebeu como a escolha de semestre é diferente dos outros campos?)

13) Acesse novamente **http://127.0.0.1:8000/historico/materias/** para ver o resultado do seu trabalho! :)