Fuctura

Desenvolvimento WEB com Django

Módulo 2 - Aula 3

Prof. Jéssica Andrade

Bora Revisar?

DJANGO

1°) Oque é um Section?

2°) Oque é HTML?

3°) Oque é uma div?

4°) Oque é CSS?

5°) Qual a estrutura de um arquivo HTML ?

6°) O que é uma tag <head> e qual a sua finalidade em um documento HTML?

7°) Como você cria um link (hyperlink) em HTML?

8°) O que é a tag <meta> e onde ela deve ser colocada dentro do HTML?

9°) Qual a diferença entre as tags <ul> e <ol> em HTML?

10°) O que faz a tag <meta charset="UTF-8"> no HTML e por que ela é importante?

11°) O que significa "semântica" no contexto de HTML e por que ela é importante?

12°) O que significa "sintaxe" no contexto de HTML e por que ela é importante?

DJANGO

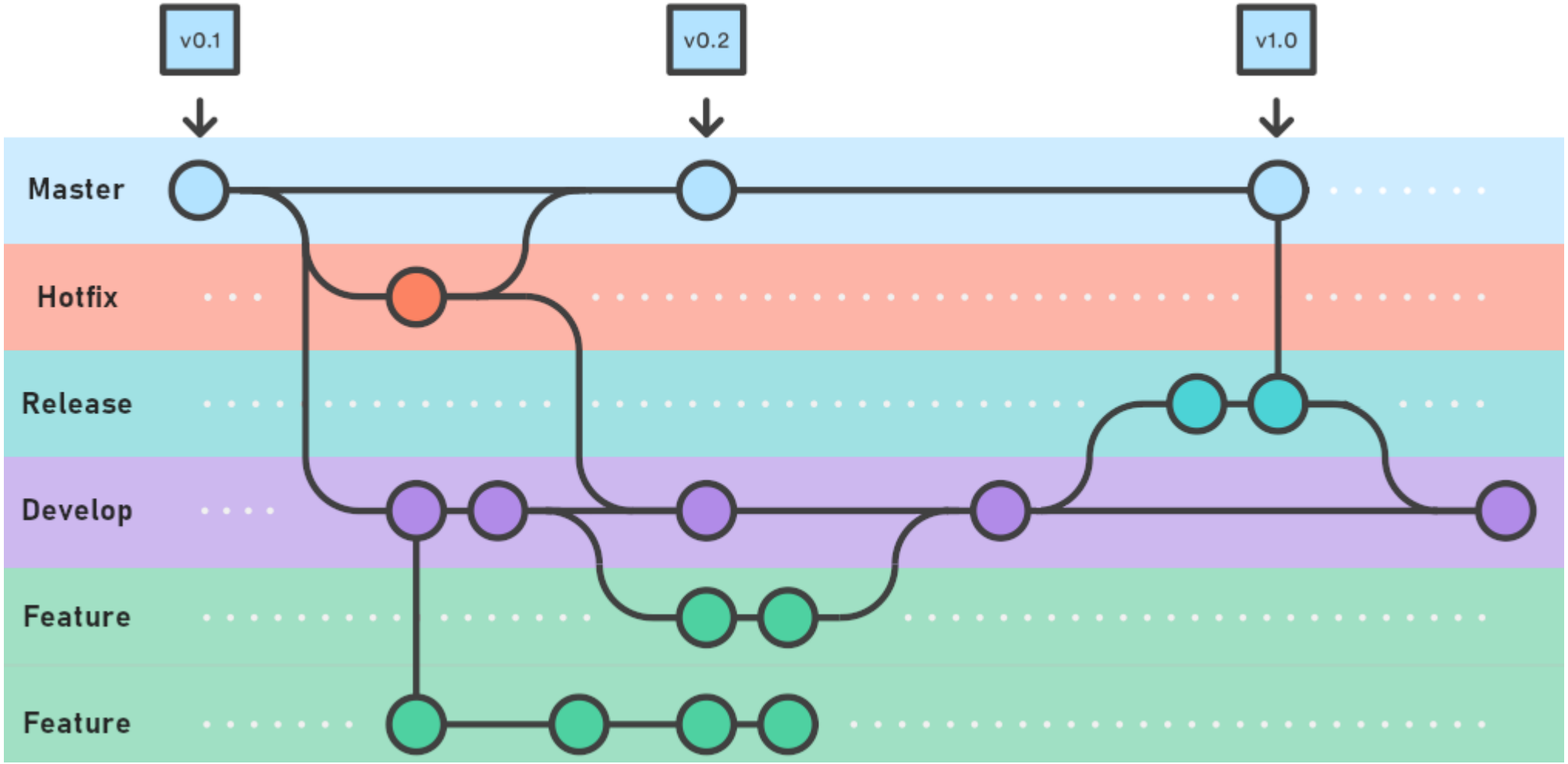
Já criou o seu perfil no github?

Compartilhem no nosso whatsapp o perfil de vocês para que vocês e seus amigos possam se ajudar.
Adicione o seu github no seu curriculo.

Todas as atividades daqui pra frente usarão o git, é importante e NECESSÁRIO vocês entenderem a funcionalidade da ferramenta e entreguem tudo através do git.

Vamos revisar os principais comandos git?

DJANGO



Bora revisar orientação a objetos?

DJANGO

Classe: Um plano para criar objetos.

Objeto: Uma instância de uma classe.

Atributos: Características do objeto (ex: cor, modelo, velocidade).

Métodos: Ações que o objeto pode realizar (ex: acelerar, parar).

__init__(self, atributos...): Esse é o **método inicializador** (ou **construtor**). Inicializar os atributos de um objeto quando ele é instanciado.self refere-se à instância do objeto que está sendo criado.O __init__ **não retorna** nada. Seu único propósito é inicializar o objeto.

Ele é **opcional**, mas muito usado para garantir que cada instância de uma classe tenha valores iniciais definidos.

DJANGO


Orientação a objetos
Encapsulamento
Herança
Composição
Polimorfismo
Princípio da abstração

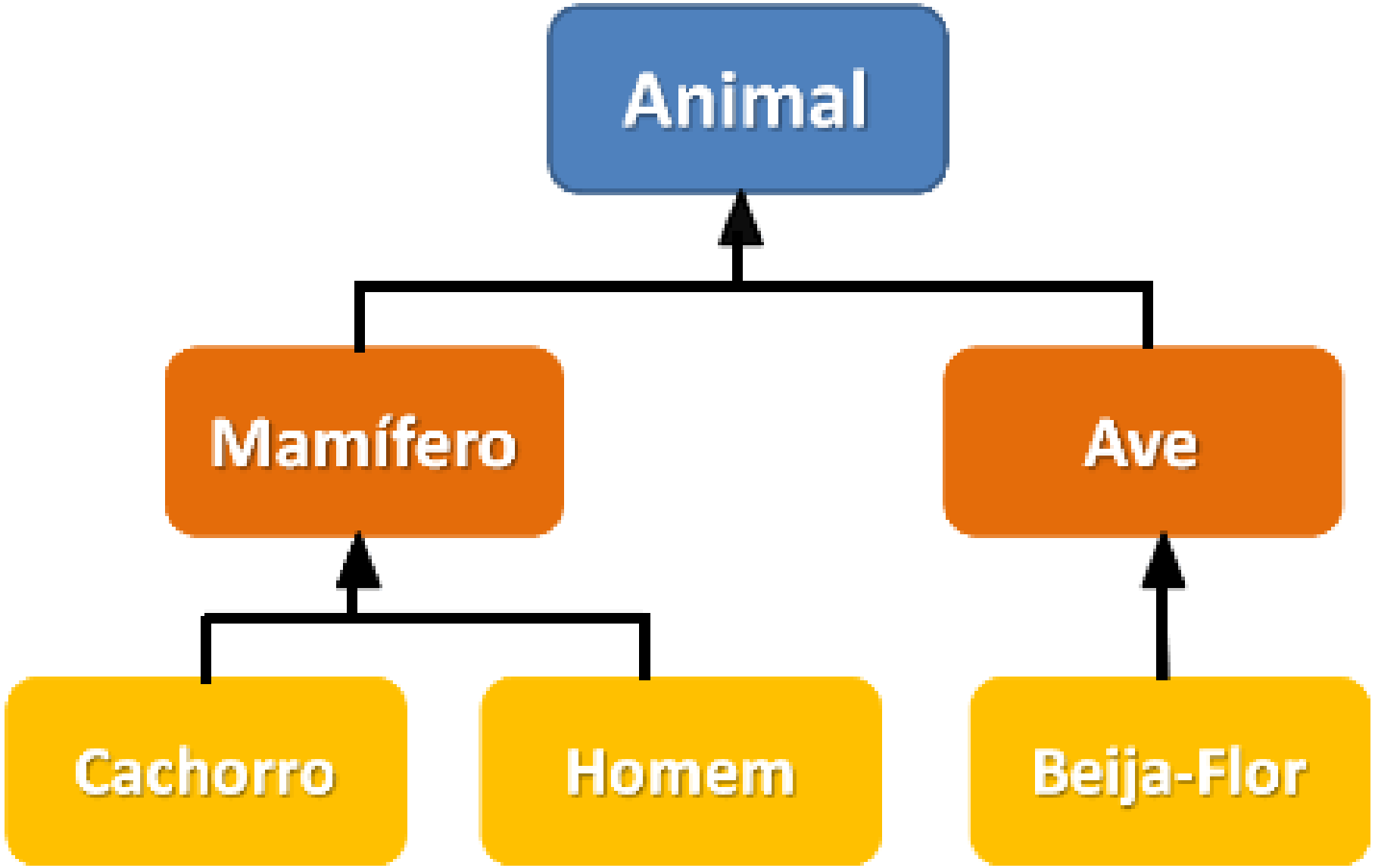
DJANGO

Encapsulamento: Esconde a complexidade e controla o acesso a atributos e métodos.

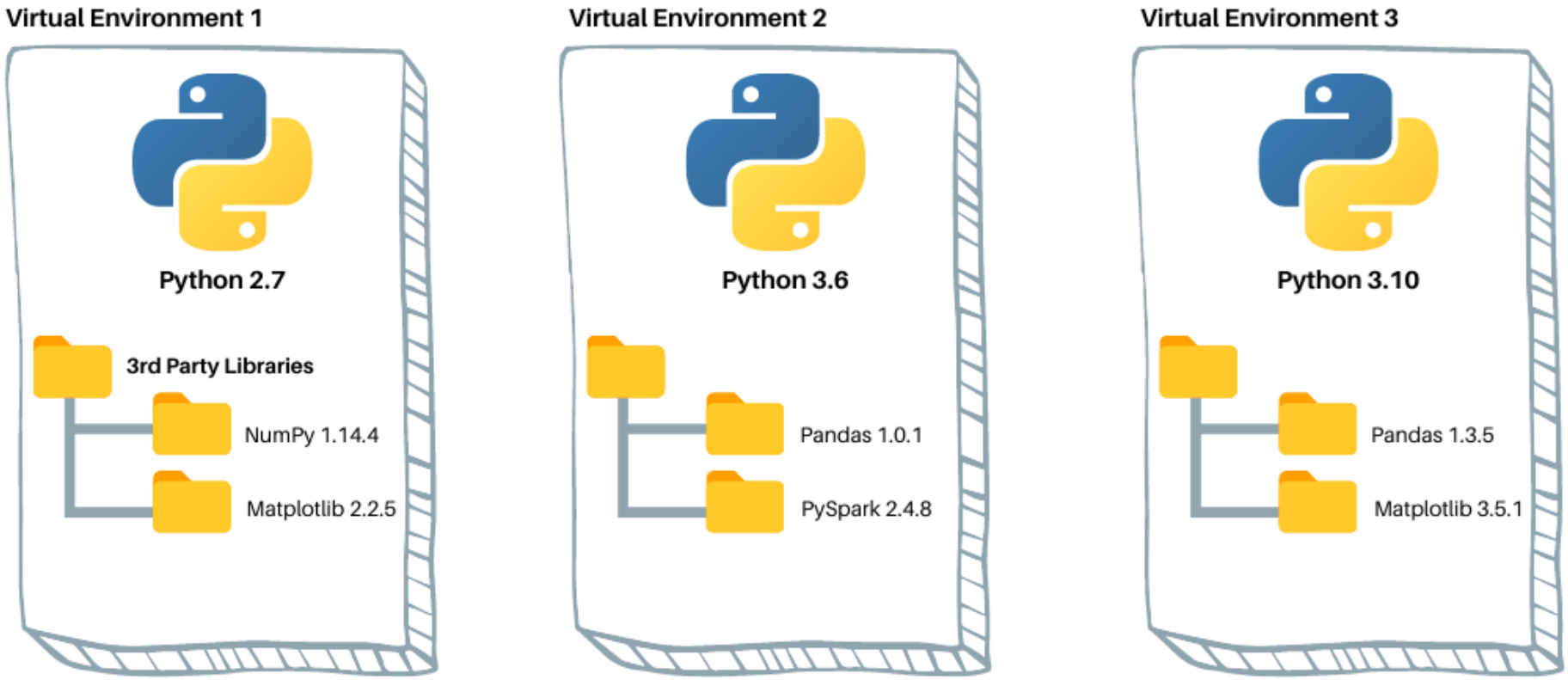
Herança: Cria uma nova classe baseada em uma classe existente, reutilizando e estendendo o código.

Composição/Abstração: Composição é quando um objeto contém outros objetos; Abstração é esconder detalhes complexos e mostrar apenas a interface essencial.

Polimorfismo: Permite que objetos diferentes respondam de maneira diferente ao mesmo método.


Animal
Mamífero
Ave
Cachorro
Homem
Beija-Flor

DJANGO


Virtual Environment 1
Python 2.7
3rd Party Libraries
NumPy 1.14.4
Matplotlib 2.2.5
Virtual Environment 2
Python 3.6
Pandas 1.0.1
PySpark 2.4.8
Virtual Environment 3
Python 3.10
Pandas 1.3.5
Matplotlib 3.5.1

DJANGO

Ambientes Virtuais:

- **Isolamento**: Uma "pasta" isolada para o projeto, sem afetar o sistema.
- **Dependências separadas**: Ferramentas e bibliotecas ficam no ambiente virtual.
- **Módulo venv**: Usado para criar o ambiente com dependências exclusivas.
- **Gestão de versões**: Permite usar versões diferentes de ferramentas sem conflitos.
- **Replicação**: Facilita copiar o ambiente em outros computadores.

DJANGO

Passo 1: Instalando virtualenv e criando o ambiente virtual.
Deve ser executado no cmd(windows), no terminal linux ou no bash do git

Windows

- pip install virtualenv
- python -m venv venv
- cd venv/Scripts

  – O comando abaixo vai depender do seu windows
- source activate ou activate ou ./activate

  – Este pode ser executado em qualquer versão
- cd ../..

Linux

- pip install virtualenv
- virtualenv venv
- source venv/bin/activate
- cd ../..

Vamos discutir sobre o nosso projetinho?

DJANGO

SmartDjangoPDV

DJANGO

- Nosso projeto será a criação de um sistema de pdv(ponto de venda) WEB
- Terá como models – Clientes, Serviços e Agendamentos
- Baseado nele vocês vão criar o projeto de vocês para este desafio
- Capricho no Frontend e no backend – ME SURPREENDAM
- Os melhores projetos receberam o troféu “DevDoMódulo” na última aula, serão 3 lugares(Uma barra de chocolate, entregue pela professora apenas na forma presencial na última aula, dê seus pulos pra vim pegar)
- Convidaremos os demais professores para serem os jurados, democracia sempre!(Se for possível é claro, se não, eu serei a jurada!)
- Pense fora da caixinha, ouse, agora é a sua hora!
- Você tem 5 segundos para cativar seu usuário!
- Fazer os desafios vão te dar pontos no final.
- Pense nisso para o seu porfólio!
- CleanCode PELO AMOR DE DEUS!

Bora começar a codar?

DJANGO

Crie um novo projeto Django chamado **barbearia_pdv,** seguindo o passo a baixo abaixo e executando os comandos em **NEGRITO** abaixo no CMD, Terminal Linux ou BASH:

1° - pip install django = Vamos instalar o django dentro da venv

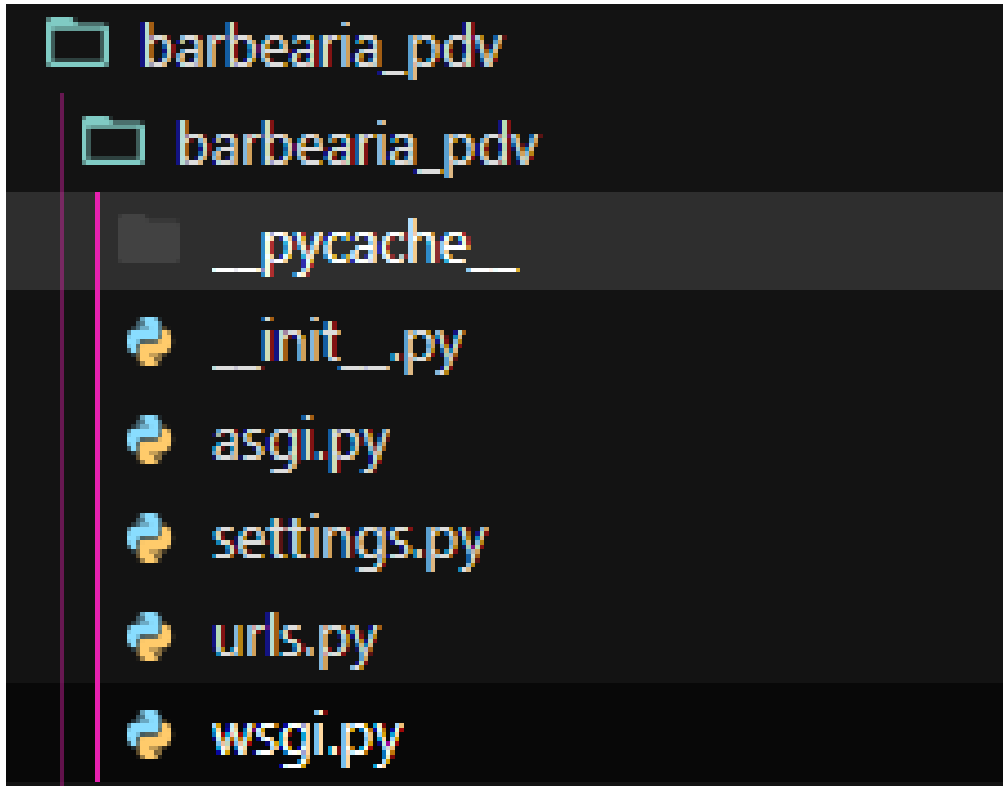
2° - django-admin startproject barbearia_pdv = Cria o projeto barbearia_pdv

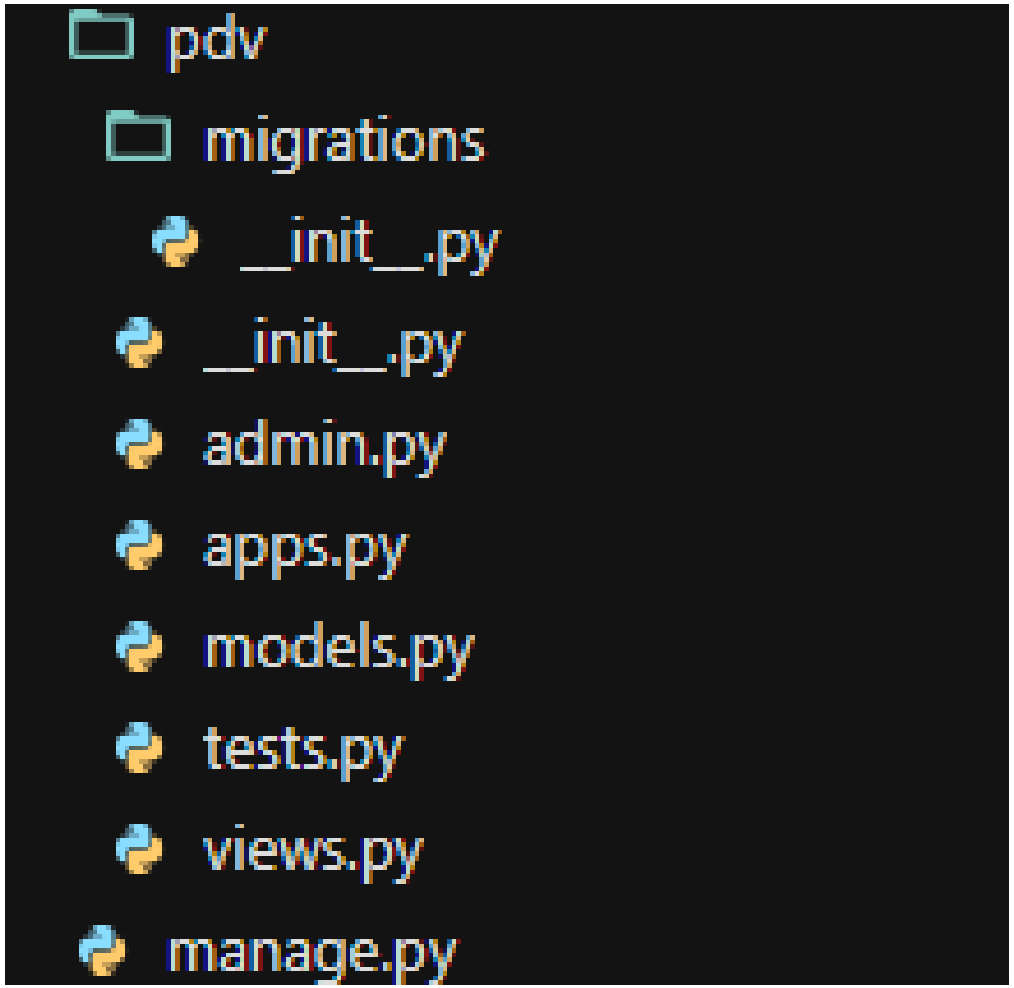
3° - cd barbearia_pdv = Vai para o diretorio de barbearia_pdv

4° - python manage.py startapp pdv = Cria um microsserviço app chamado pdv

DJANGO

Agora temos algo semelhante aos diretórios abaixo:





DJANGO 1

Agora, dentro do arquivo **settings.py**, que esta na pasta principal do projeto, chamada **barbearia_pdv,** insira as duas linhas abaixo:

```
'django.contrib.sites',
'pdv'
```

Agora temos algo semelhante ao exemplo da foto ao lado.

NÃO ESQUEÇA DE SALVAR COM CTRL + S

settings.py

barbearia_pdv > barbearia_pdv > settings.py > ...

```
32
33  INSTALLED_APPS = [
34      'django.contrib.admin',
35      'django.contrib.auth',
36      'django.contrib.contenttypes',
37      'django.contrib.sessions',
38      'django.contrib.messages',
39      'django.contrib.staticfiles',
40      'django.contrib.sites',
41      'pdv',
42  ]
43
```

DJANGO

Para que serve alguns arquivos criados automaticamente e que são importantes?

wsgi.py e asgi.py

- **Conexão servidor-aplicativo**: Gerenciam como o aplicativo lida com requisições do servidor.
- **Django 3+**: Inclui o asgi.py para suportar novas requisições assíncronas.

settings.py

- **Configurações do Django**: Arquivo com todas as configurações do projeto.
- **Módulo Python**: Define variáveis para personalizar o funcionamento do framework.

URLconf

- **Mapeamento de URLs**: Define padrões de URLs para encontrar a resposta correta.

manage.py

- **Gestão do projeto**: Script para gerenciar o site, como iniciar o servidor local.

DJANGO

- Seguindo o conceito de microsserviços, você pode ter quantos projetos/entidades quiser e precisar dentro do seu projeto django. Vamos começar com PDV, e logo depois vamos evoluir para as demais.

- **Sempre** que criarmos um novo app ou instalarmos uma ferramenta, é necessário registrar em INSTALLED_APPS.

DJANGO

- O comando abaixo inicia o servidor, e seve ser executado no CMD, Terminal Linux ou BASH. **Vá em frente e EXECUTE!**
- Lembre-se: **sempre** que formos testar o app, devemos rodar este comando para subir o servidor.

- **python manage.py runserver**

Vá no seu navegador e adicione a url abaixo para visualizar o seu site, ou clique abaixo para ser encaminhado. Esa URL será sempre a mesma sempre que o servidor estiver de pé.

http://127.0.0.1:8000/admin/

DJANGO

The install worked successfully! Congratulations!

View release notes for Django 5.1

You are seeing this page because DEBUG=True is in your settings file and you have not configured any URLs.

django

Se chegamos aqui, tudo está funcionando perfeitamente!

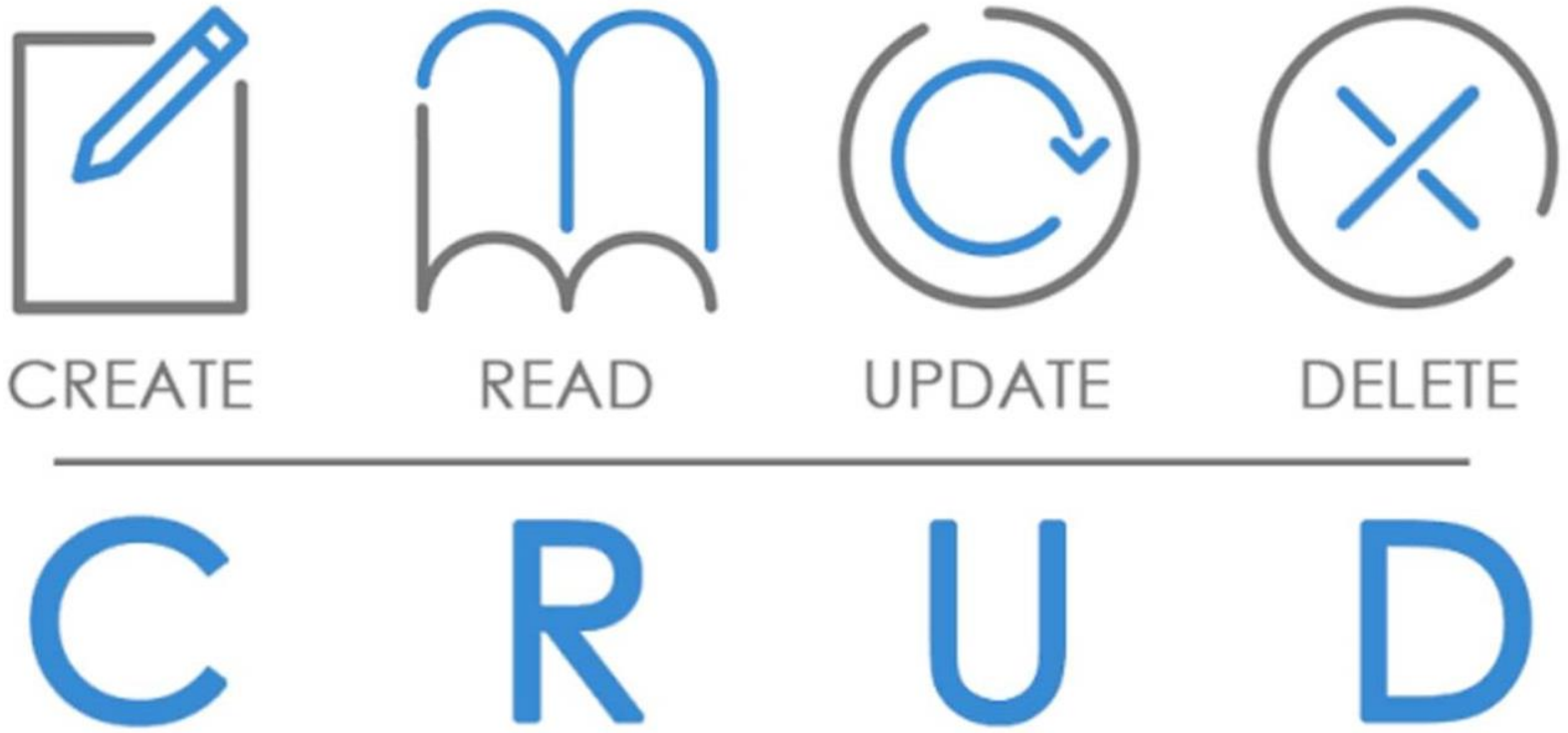
Bora começar a codar?

DJANGO

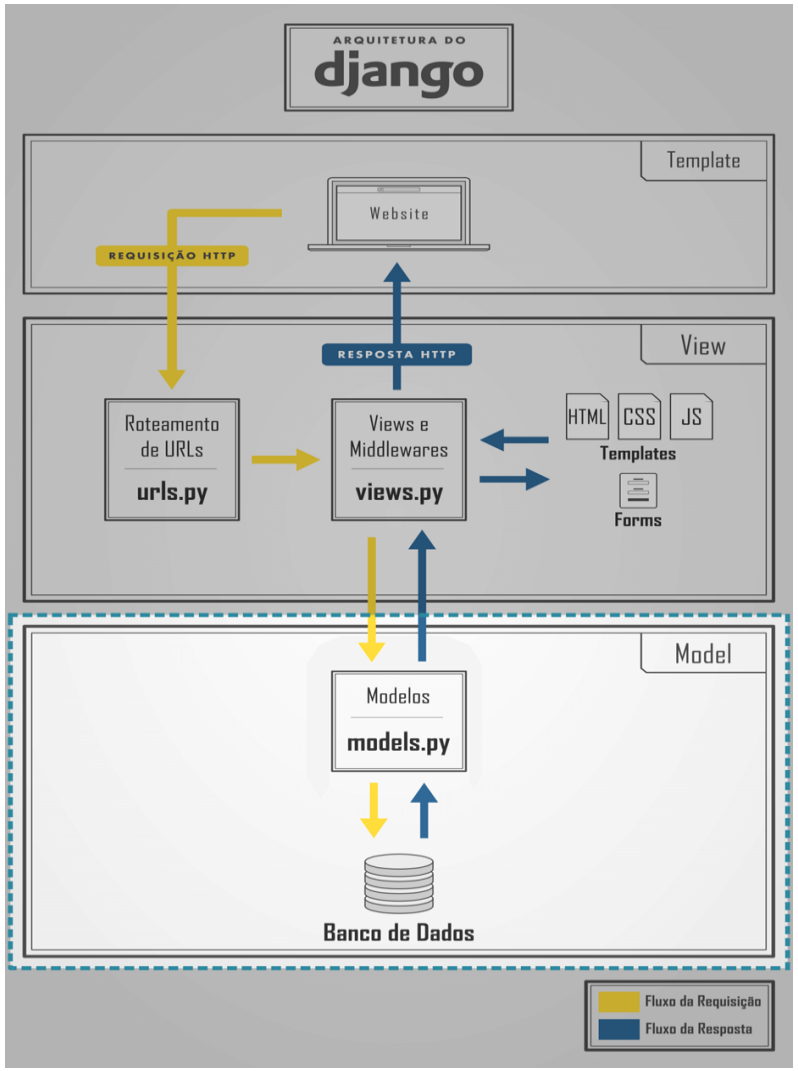
- Sempre criem ambientes virtuais para seus projetos.
- Sempre verifiquem se os ambientes estão ativos. vai aparecer escrito como (venv) encima do nome do seu diretorio no terminal, CMD ou BASH
- Sejam organizados com seus projetos.
- No início desse material, temos um passo a passo que pode ser seguido para iniciar projetos em Django.

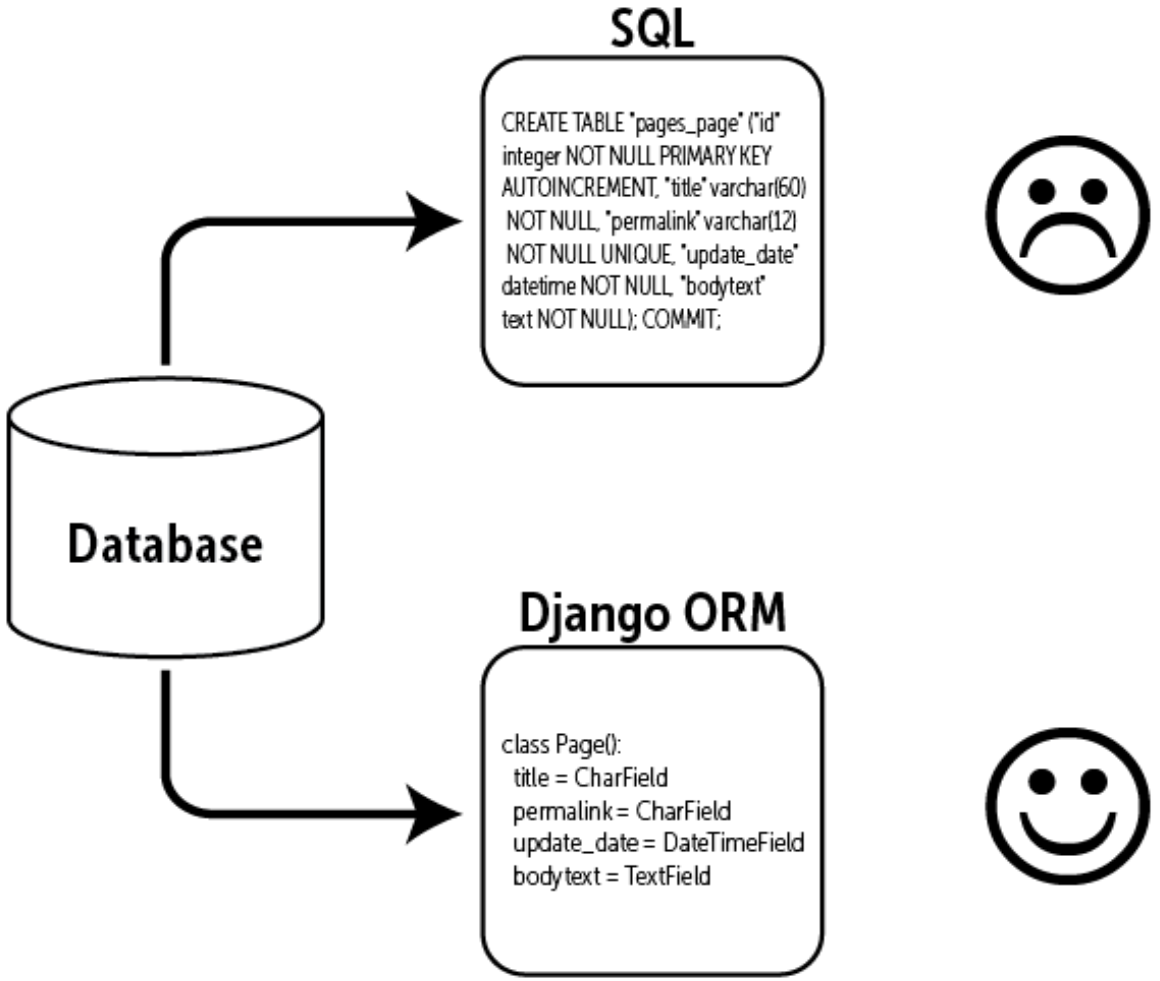
A partir do próximo slide, vamos aprender a fazer um CRUD com o Django.

DJANGO



DJANGO - MODEL





DJANGO

Insira o código ao lado dentro do arquivo **models.py**, que esta dentro da nossa pasta **pdv.**

Estamos criando nossas classes, estas irão referenciar tabelas do banco de dados que serão criadas pelo django futuramente.

NÃO ESQUEÇA DE SALVAR COM CTRL + S

models.py

barbearia_pdv > pdv > models.py > ...

```python
from django.db import models
from django.contrib.auth.models import User

class Cliente(models.Model):
    nome = models.CharField(max_length=100)
    telefone = models.CharField(max_length=15)
    email = models.EmailField()

    def __str__(self):
        return self.nome

class Servico(models.Model):
    nome = models.CharField(max_length=100)
    preco = models.DecimalField(max_digits=10, decimal_places=2)

    def __str__(self):
        return self.nome

class Agendamento(models.Model):
    cliente = models.ForeignKey(Cliente, on_delete=models.CASCADE)
    servico = models.ForeignKey(Servico, on_delete=models.CASCADE)
    data_hora = models.DateTimeField()

    def __str__(self):
        return f"{self.cliente} - {self.servico} em {self.data_hora}"
```

– Execute o comando abaixo no CMD, Terminal ou BASH

python manage.py makemigrations

Esse comando irá criar as instruções SQL de forma correta e injetar elas no banco de dados, se tudo estiver certo, vocês vão ver algo parecido com esta saída

```
$ python manage.py makemigrations
Migrations for 'pdv':
  pdv\migrations\0001_initial.py
    + Create model Cliente
    + Create model Servico
    + Create model Agendamento
(venv)
```

DJANGO

– Execute o comando abaixo no CMD, Terminal ou BASH

python manage.py migrate

Este comando irá inserir corretamente e salvar as operações no banco de dados. Se tudo estiver certo, vocês receberão uma saida parecida com esta.

```
$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, pdv, sessions, sites
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying pdv.0001_initial... OK
  Applying sessions.0001_initial... OK
  Applying sites.0001_initial... OK
  Applying sites.0002_alter_domain_unique... OK
(venv)
```

Mas assim fica difícil de visualizar nossos esquemas, tabelas e colunas criadas para nosso banco de dados. Como podemos melhorar e usar oque o Django já tem de pronto?

– Execute os comandos abaixo um após o outro, no CMD, Terminal ou BASH

python manage.py runserver

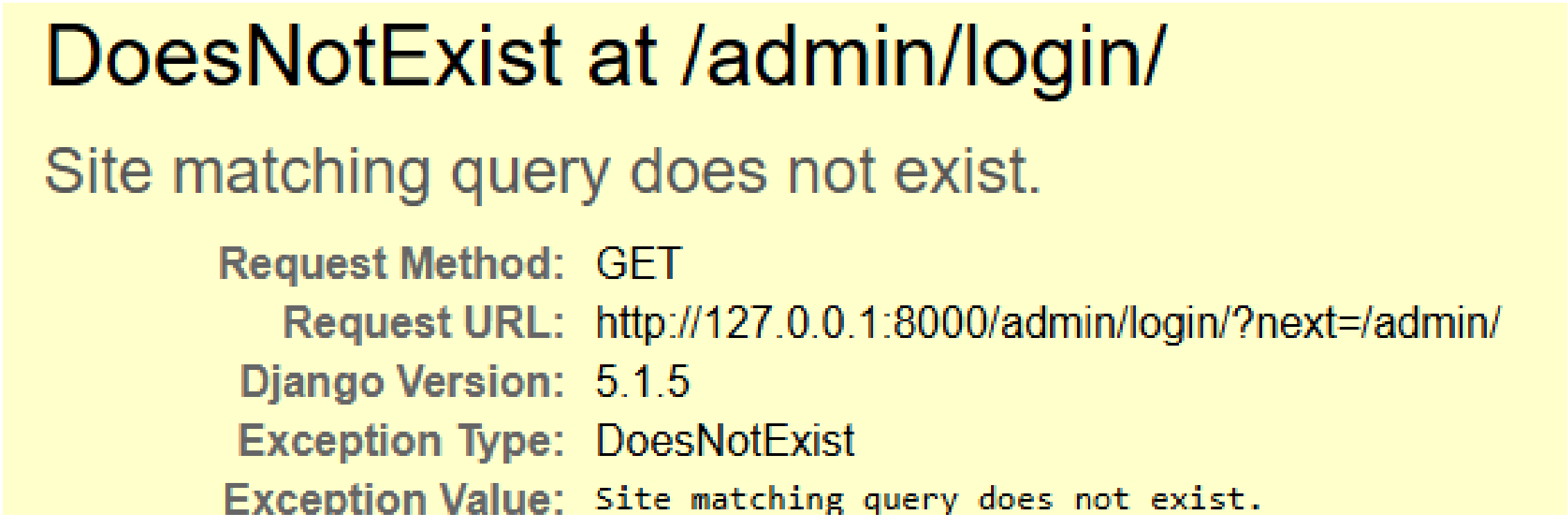
Acesse a URL abaixo no seu navegador

http://127.0.0.1:8000/admin/

Oque nós vamos receber?

DJANGO

Vamos ser encaminhados para uma rota que não existe, precisamos corrigir isso!!!



– Execute os comandos abaixo um após o outro, no CMD, Terminal ou BASH

```
python manage.py shell
```

```
from django.contrib.sites.models import Site
```

```
site = Site.objects.create(domain='127.0.0.1:8000', name='Localhost')
```

```
Site.objects.all()
```

– Apenas uma explicação do que cada comando faz

Este comando irá abrir o Shell do django

python manage.py shell

Vamos criar um novo site apenas para o painel admin

from django.contrib.sites.models import Site

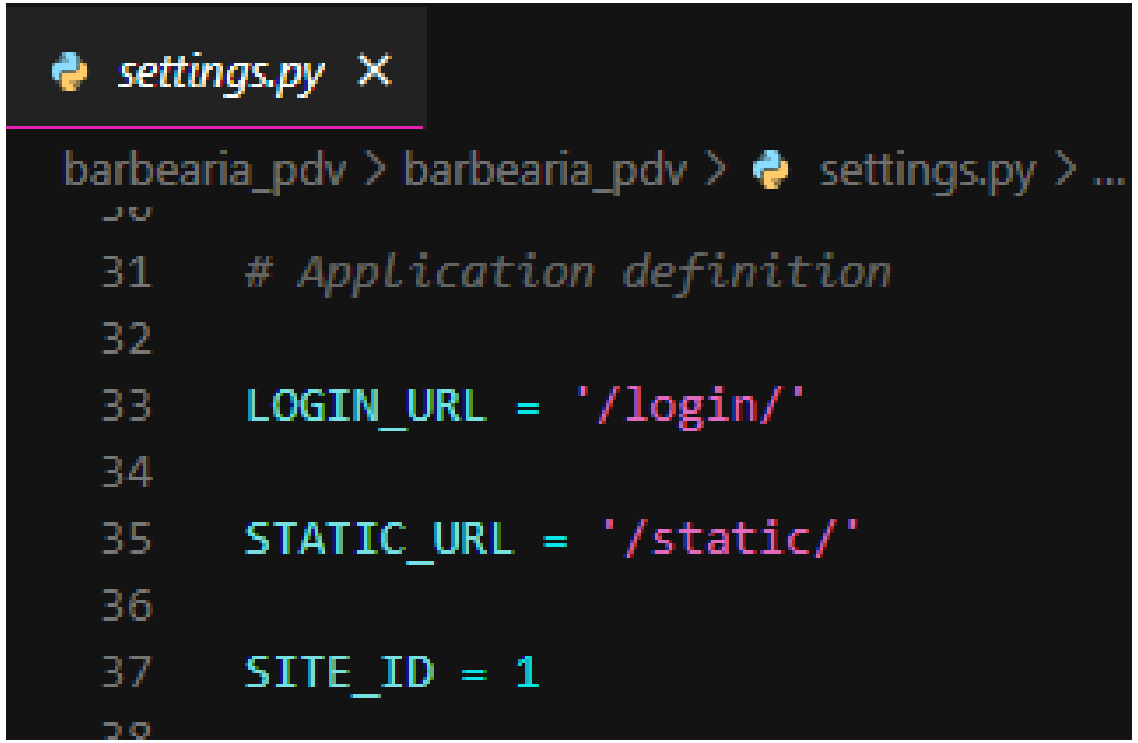
Crie um novo Site

site = Site.objects.create(domain='127.0.0.1:8000', name='Localhost')

#Este comando mostra todos os sites ativos

Site.objects.all()

DJANGO



Vamos validar se isso existe dentro do nosso arquivo de **settings.py** que esta no nosso projeto principal **dentro da pasta barbearia_pdv.** Se não houver, adicione!

NÃO ESQUEÇA DE SALVAR COM CTRL + S

DJANGO

```python
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    #path('', include('pdv.urls')),
]
```

Por fim, precisamos configurar nossa url de admin, dentro do arquivo **urls.py**, que esta dentro da nossa pasta principal **barbearia_pdv**, para que sigam este padrão, mantenha a linha com o include comentada por enquanto

➡ **NÃO ESQUEÇA DE SALVAR COM CTRL + S**

– Execute os comandos abaixo um após o outro, no CMD, Terminal ou BASH

python manage.py createsuperuser

```
$ python manage.py createsuperuser
Username: teste
Email address: teste@email.com
Password:
Password (again):
The password is too similar to the username.
This password is too short. It must contain at least 8 characters.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.
(venv)
```

MUITO CUIDADO NA HORA DA CRIAÇÃO, USE UM EMAIL E UMA SENHA QUE VOCÊ CONHEÇA E QUE SEJAM FACEIS PARA VOCÊ LEMBRAR.

Este comando deve te gerar a saida abaixo, você vai criar um super usuario para o painel admin do django

– Execute os comandos abaixo um após o outro, no CMD, Terminal ou BASH

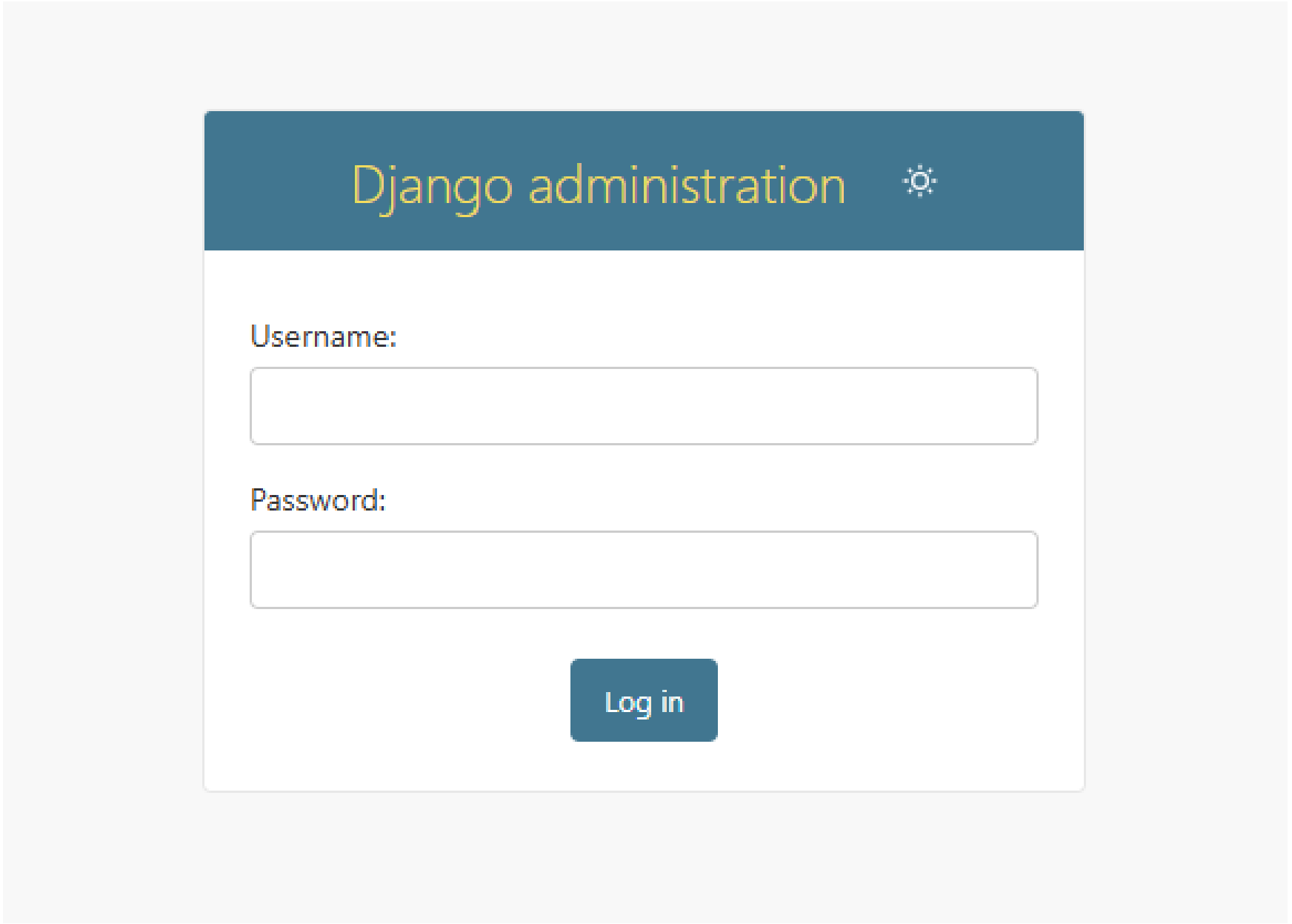
python manage.py runserver

Acesse a URL abaixo no seu navegador

http://127.0.0.1:8000/admin/

E agora oque nós vamos receber?

DJANGO



Oba, agora temos um dashboard para gerenciar nossos modelos do banco de dados, use o super usuario que criou antes para acessar!

DJANGO

Por que eu ainda não consigo visualizar meus modelos de Agendamentos, clientes e Serviços?

Precisamos registrar eles para que o painel de administradores consiga gerenciar eles corretamente

DJANGO

Adicione dentro do arquivo **admin.py**, que esta na pasta **pdv** as linhas abaixo, desta forma vamos registrar nossos modelos para serem acessados no dashboard admin

➡ NÃO ESQUEÇA DE SALVAR COM CTRL + S

admin.py

barbearia_pdv > pdv > admin.py

```python
from django.contrib import admin
from .models import Cliente, Servico, Agendamento

# Registre os modelos para que apareçam no painel de administração
admin.site.register(Cliente)
admin.site.register(Servico)
admin.site.register(Agendamento)
```

– Execute os comandos abaixo um após o outro, no CMD, Terminal ou BASH

python manage.py runserver

Acesse a URL abaixo no seu navegador

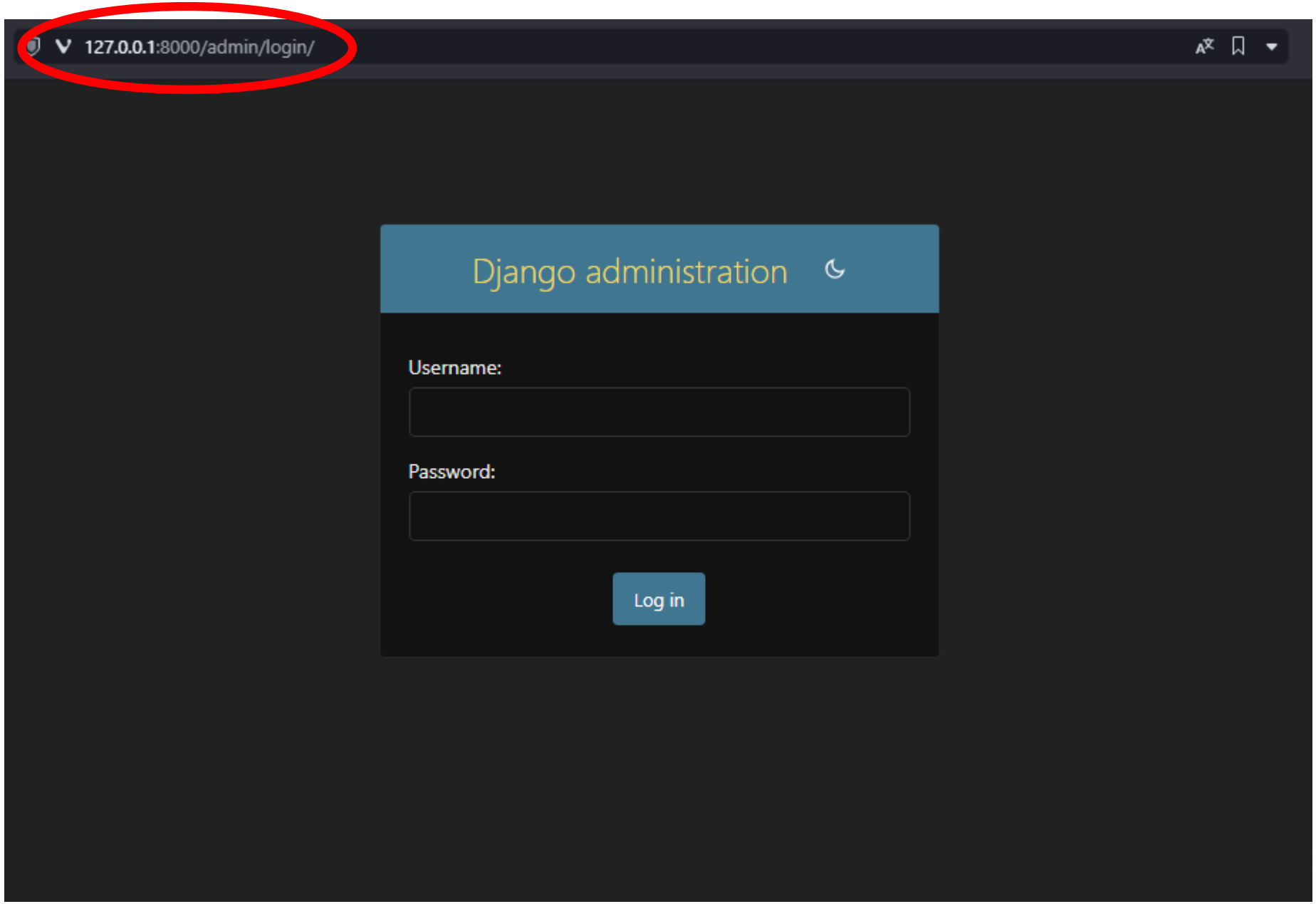
http://127.0.0.1:8000/admin/

DJANGO

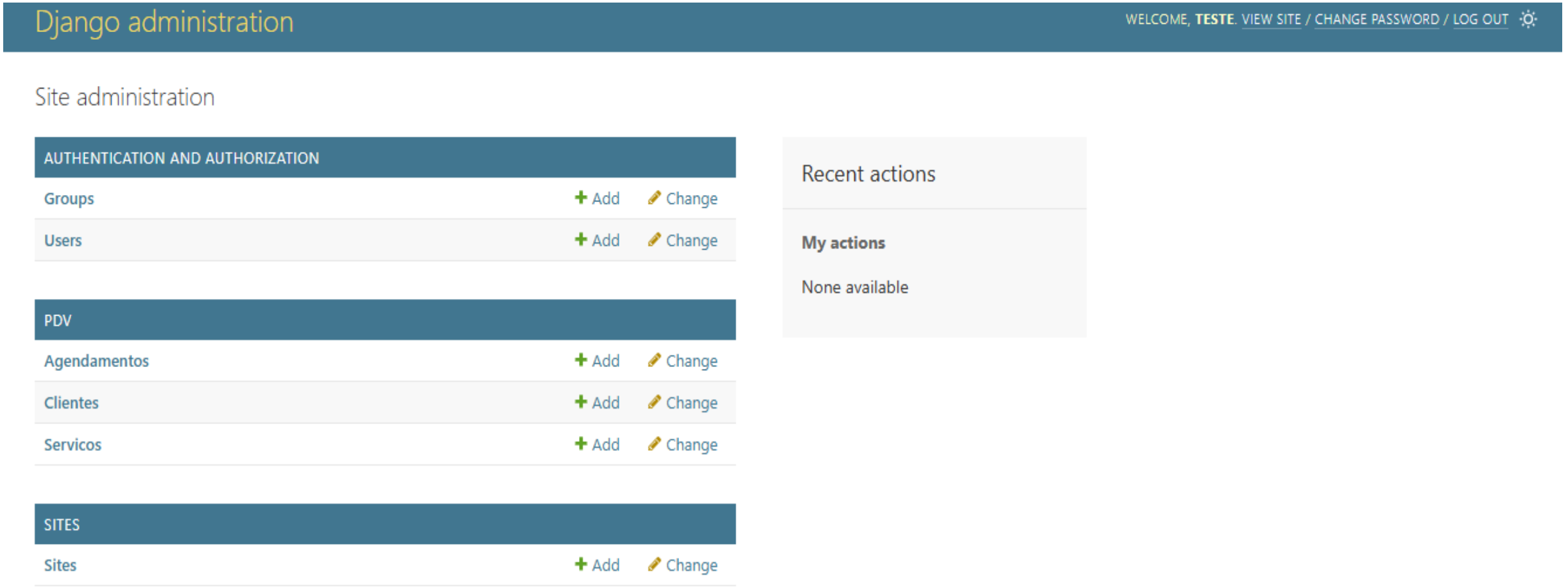
Ele entrega um template completo com uma integração de acesso robusta e com um crud completo para os nossos modelos do banco de dados, de forma simples, segura, “humana” e intuitiva

DJANGO

Após isso, basta entrar com o usuário criado no passo do superuser.



DJANGO


Django administration
WELCOME, TESTE. VIEW SITE / CHANGE PASSWORD / LOG OUT
Site administration
AUTHENTICATION AND AUTHORIZATION
Groups + Add Change
Users + Add Change
PDV
Agendamentos + Add Change
Clientes + Add Change
Servicos + Add Change
SITES
Sites + Add Change
Recent actions
My actions
None available

Sua hora de brilhar!

DJANGO

Os exercícios daqui pra frente vão ser complementares, ou seja, conforme teremos nossas aulas, vocês devem iniciar um projeto a parte, e usar oque aprendemos em aula neste projeto.

Incrementem com oque aprendemos na aula de hoje!

Subam este projeto no git e me enviem o repositório!

Todos os nossos códigos devem ser armazenados e entregues pelo git.

DJANGO

EXERCICIO PARA SER FEITO BASEADO NO QUE CRIAMOS EM AULA!

Sistema de Reservas de Hotel
Descrição: Sistema para gerenciar reservas em um hotel, alocando clientes em quartos disponíveis. **Escolha o nome que quiser para o seu hotel**

Classes - modelos de banco de dados:
- Quarto: número, tipo, preço.
- Cliente: nome, cpf, telefone.
- Reserva: cliente, quarto, datas de entrada e saída.
- Despesa: descrição, valor, data.
- Receita: descrição, valor, data.

USEM A CRIATIVIDADE E OS EXEMPLOS VISTOS EM AULA!!

Desafios para treinar e melhorar seu portfólio!

OPCIONAIS

DJANGO

Criação de um Blog Simples:

- Crie um aplicativo com um modelo Post que tenha título, conteúdo e data de criação.
- Configure URLs e views para listar, visualizar e criar posts.
- Adicione templates HTML para cada página (lista, detalhe e formulário de criação).

Sistema de Cadastro de Usuários:

- Implemente um sistema de registro e autenticação para novos usuários.
- Crie páginas de login e logout.
- Proteja algumas rotas para que apenas usuários logados possam acessá-las.

Aplicativo de To-Do List:

- Crie um modelo Tarefa com atributos como título, descrição e status de conclusão.
- Configure views para criar, visualizar, atualizar e excluir tarefas.
- Adicione uma página para listar todas as tarefas com links para editar e deletar cada uma.

DJANGO

Agenda de Contatos:

- Crie um aplicativo onde cada contato tem um nome, telefone e e-mail.
- Configure views e templates para listar todos os contatos, além de adicionar e remover.
- Implemente uma barra de pesquisa para filtrar contatos pelo nome.

Sistema de E-commerce Básico:

- Crie modelos para Produto (nome, descrição, preço) e Categoria (com nome e descrição).
- Configure uma página inicial que exiba produtos e suas respectivas categorias.
- Adicione uma página de detalhes para cada produto.

NÃO ESQUEÇA


@PROGRAMADORESDEPRE
OS ROBÔS NÃO VÃO TIRAR SEU EMPREGO SE VOCÊ FOR O PROGRAMADOR.