Django: é um framework, abstrai parte web, open source

Estrutura: na pasta app\_django há esses arquivos

\_\_init\_\_.py

Settings.py – configurações

Urls.py – endereços

Na pasta core

Tests.py -> diz a respeito dos testes

Pip – gerenciamento de pacotes do python, vem junto ao python, versão 3.4+

VirtualEnv – Ambiente virtual python, executa de forma isolada

Estrutura do projeto Django (é criada dando o comando Django-admin startproject

Estrutura app comando Django-admin starpapp nome\_app (se cria dentro da pasta do projeto)

Pip install Django (instalação do Django)

Djang-admin startproject nome\_projeto (criação de projeto)

Django-admin startapp nome\_app (se cria novo app)

Na parte de settings.py em installed apps, adicionamos o novo app criado

No views.py, criamos métodos que retornam algo da visualização

**Estrutura básica DJANGO**

Se inicia com o startproject e startapp, a criação da estrutura básica

Django-admin,py Responsável pelos comandos Django de realização de tarefas, dando Django-admin –help é listada todas tarefas disponíveis (pertence ao pacote Django

Manage.py -> Delega as tarefas para o Django-admin,py (é interface), ele também define paths do sistema operacional, automatiza operações

WSGI.py (web services Gateway Interface), funciona como interface entre servidor e aplicação, como um servidor, a exemplo, apache da vida, deve-se trocar o servidor quando for para produção.

Setting.py -> Define configurações DJANGO, que são dependências da aplicação, templates, time-zone, Apps instalados, base de dados (driver),

URLs.py -> Gerencia rotas, nome e direção que irão seguir

Views.py -> Processo e retorno de resposta a clientes

Models.py -> Modelo de dados, abstração de dados do banco (para linguagem python), comandos CRUD

Admin.py Interface administrativa Django, que gera página web que permite alterar esses dados

Diretório STATIC -> Armazena arquivos staticos (CSS, JS, midia)

Diretório TEMPLATEs -> Responsável de costume de guardar os templates de páginas (HTML)

Python manage.py migrate – cria as tabelas padrão do admin

Por padrão é uma database sqlite

python manage.py createsuperuser --username admin (cria usuario)

manage.py makemigrations app\_name -> cria campos na tabela (ID é criado automático)

Quando se modifica classe de model, necessitamos de executar

Sqlmigrate muda as opções de tabela, migrate faz alterações em definitivo

Autenticação, é dada pelo pacote djang.contrib.auth

Função authenticate, login, logout, login\_required,

Decoradores -> funçoes que faz uma extração de código em comum para todas as partes, como a exemplo um validador.