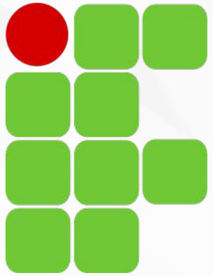


Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte
Diretoria de Gestão de Tecnologia da Informação
Coordenação de SIstemas de Informação

suap
sistema unificado de administração pública

Desenvolvimento Python e Django voltado para o SUAP

Parte 1 - Fundamentos do Python e Django

Allyson Barros - allyson.barros@ifrn.edu.br

Salvador, Setembro / 2018

Apresentação

- Mestre em Eng. de Software - UFRN
- Especialista em Arquitetura de Nuvem - UFRN
- Tecnólogo em Análise e Desenvolvimento de Sistemas - IFRN
- Analista de Tecnologia da Informação - IFRN
- Pesquisador - Laboratório de Inovação Tecnológica em Saúde (LAIS) - UFRN
- Pesquisador - Núcleo Avançado de Inovação Tecnológica (NAVI) - IFRN

Agenda

- 2 - Fundamentos do Django
  - Introdução ao Django
  - Administração do Django
  - Modelo MTV (Model, Template, View)
  - Arquivos estáticos, views genéricas e testes
  - Deploy da aplicação

Introdução ao Django

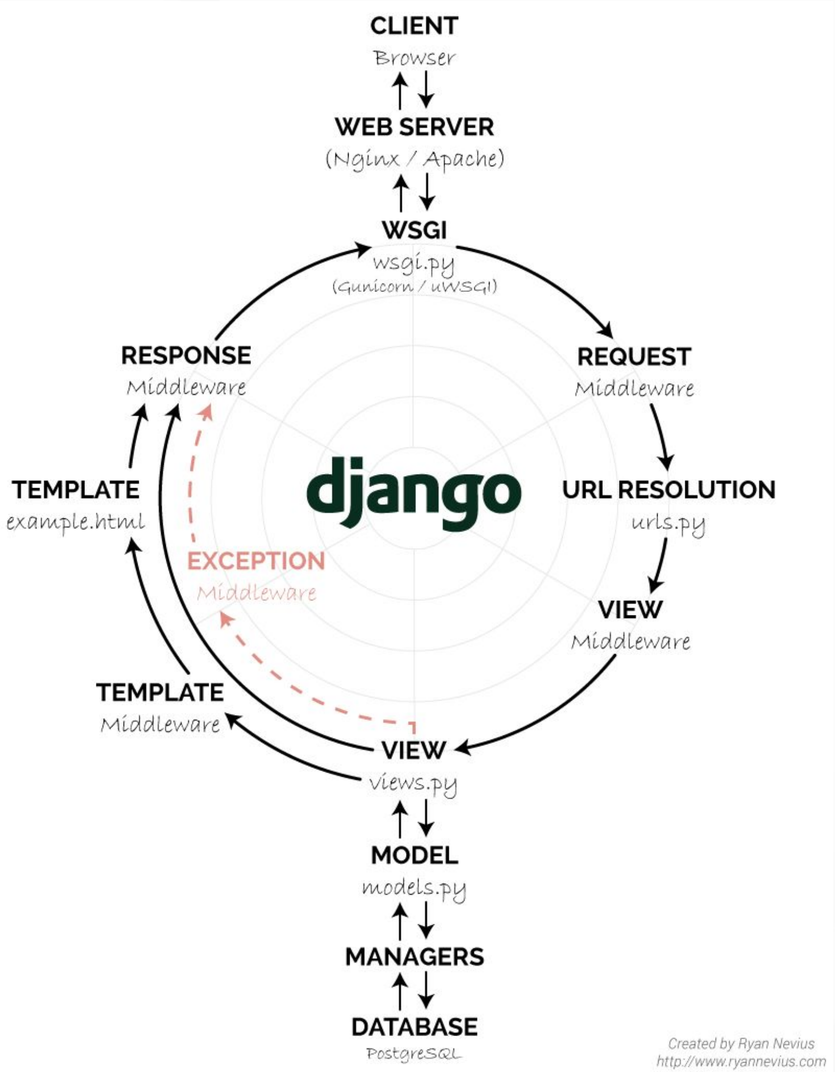
Histórico

- Os primeiros sistemas para internet não possuíam tantos recursos e técnicas;
- Desenvolvedores identificaram tarefas repetitivas entre sistemas;
- As tarefas repetitivas começaram a ser organizadas e compartilhadas através de frameworks;
- O Django cresceu a partir de aplicações reais escritas para produzir e manter diversas aplicações em um ambiente ditado por prazos jornalísticos;
- Iniciado em 2003 para suprir demandas de funcionalidades e aplicações inteiras em dias ou até horas;
- Em 2005, o Django foi disponibilizado como software livre (licença BSD);
- Nomeado como homenagem a Django Reinhardt, um guitarrista de jazz;

Introdução ao Django

Conceitos

- Dividido em três partes conceituais, Model, View e Template (MVT);
- Model define a estrutura do banco de dados;
- View seleciona e filtra quais dados serão apresentados;
- Template cuida da apresentação dos dados;
- Em comparação com o padrão MVC, a equipe responsável pelo Django considera que a camada de Controller está distribuído por várias partes do framework;



Introdução ao Django

Organização do Django

- O desenvolvimento é realizado em um projeto
- O projeto é separado por aplicações
- As aplicações possuem views, templates e models
- Um sistema de URLs direciona as requisições para views que utilizam templates para apresentar dados definidos nos models, usando o ORM do Django

Introdução ao Django

Ambiente de Desenvolvimento

- Editor de texto simples;
- Ambiente virtual da linguagem Python;
  - Virtualenv cria uma instalação de Python isolada
  - Permite que bibliotecas sejam instaladas em ambientes virtuais de acordo com os projetos sem que entrem em conflito
- Banco de Dados;
  - Assim como o Python, o Django permite a conexão com diversos SGDBs como Sqlite3, Postgres, SQL Server.

Introdução ao Django

Ambiente de Desenvolvimento

- Editor de texto simples;
- Ambiente virtual da linguagem Python;
  - Virtualenv cria uma instalação de Python isolada
  - Permite que bibliotecas sejam instaladas em ambientes virtuais de acordo com os projetos sem que entrem em conflito
- Banco de Dados;
  - Assim como o Python, o Django permite a conexão com diversos SGDBs como Sqlite3, Postgres, SQL Server.

Introdução ao Django

Sobre o Projeto a ser desenvolvido

- Projeto baseado no tutorial do Django;
- Sistema de reservas
  - Gerenciamento de Clientes
  - Gerenciamento das Reservas de Clientes

Introdução ao Django

Instalando o Django

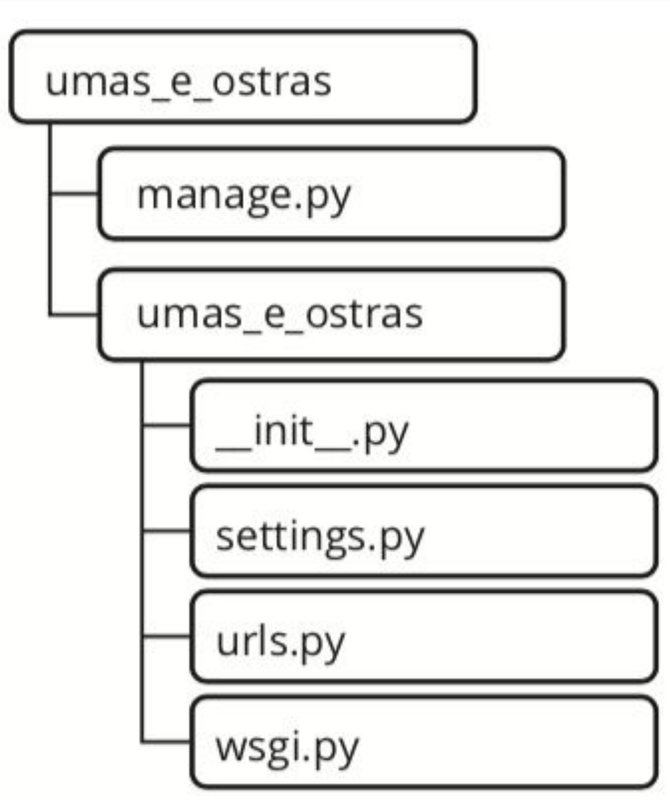
- Instalando o Ambiente Virtual do Python
  - pip install virtualenv virtualenvwrapper
  - export WORKON_HOME=~/Envs
  - mkdir -p $WORKON_HOME
  - source /usr/local/bin/virtualenvwrapper.sh

- Criando um novo Ambiente Virtual do Python
  - mkvirtualenv curso_python
  - pip install django==1.11

Introdução ao Django

Criando o projeto

- Para criar o projeto:
  - mkvirtualenv curso_python
  - django-admin startapp umas_e_ostras
- Estrutura do projeto:
  - Diretório raiz umas_e_ostras: contém o projeto, o nome não faz diferença;
  - manage.py: utilitário para interagir com o projeto;
  - Diretório interno umas_e_ostras: pacote do projeto;
  - umas_e_ostras/__init__.py: arquivo que indica que o diretório é um pacote;
  - umas_e_ostras/settings.py: arquivo de configuração;
  - umas_e_ostras/urls.py: declaração de URLs para o sistema de URLs do Django;
  - umas_e_ostras/wsgi.py: ponto de entrada para servidores web WSGI hospedarem o site.



Introdução ao Django

Configurando o banco de dados

- No arquivo umas_e_ostras/settings.py, o banco é configurado;
- Por padrão, a configuração usa o sqlite3 e um banco chamado db.sqlite3;
- Existem muitas opções de banco, entre elas:
  - ‘django.db.backends.postgresql_psycopg2’
  - ‘django.db.backends.mysql’.
- É necessário adicionar outras opções para esses bancos:
  - USER
  - PASSWORD
  - HOST

Introdução ao Django

Aplicações

- Projetos Django são divididos em aplicações;
- Aplicações são independentes entre si e podem ser compartilhadas entre projetos;
- Django possui aplicações padrão que são instaladas junto com o framework;
- Novas aplicações são criadas para resolver tarefas específicas de acordo com os requisitos do sistema;
- É utilizada a variável INSTALLED_APPS para configurar as aplicações a serem usadas no projeto;
- Por padrão, são habilitadas algumas aplicações pré-instaladas.

Introdução ao Django

Aplicações padrão do Django

- django.contrib.admin
  - Sistema Administrativo;
- django.contrib.auth
  - Sistema de autenticação;
- django.contrib.contenttypes
  - Framework para tipos de conteúdos;
- django.contrib.sessions
  - Framework de sessões;
- django.contrib.messages
  - Framework de mensagens;
- django.contrib.staticfiles
  - Framework para gerenciamento de arquivos estáticos;

Introdução ao Django

Configurando o projeto “umas_e_ostras”

- Antes de ser iniciado o desenvolvimento do projeto, algumas alterações podem ser feitas para adequá-lo às características da região
  - Editar o arquivo umas_e_ostras/settings.py
    - Alterar a variável do código de linguagem
      - LANGUAGE_CODE = ‘pt-BR’
    - Alterar a variável do fuso horário:
      - TIME_ZONE = ‘America/Sao_Paulo’
  - Executar o comando:
    - python3 manage.py migrate

Introdução ao Django

Executando o servidor de desenvolvimento

- Servidor simples e leve.
- Não deve ser usado em produção.
- O servidor de desenvolvimento é reiniciado automaticamente quando mudanças são salvas, mas precisa ser manualmente reiniciado quando novos arquivos são adicionados.
- Iniciando o servidor:
  - python manage.py runserver
- Com IPV6 ::1 porta 8000:
  - python3 manage.py runserver -6
- Mudando a porta padrão:
  - python3 manage.py runserver 8080
- Permitindo acesso na mesma rede:
  - python3 manage.py runserver 0.0.0.0:8000

Introdução ao Django

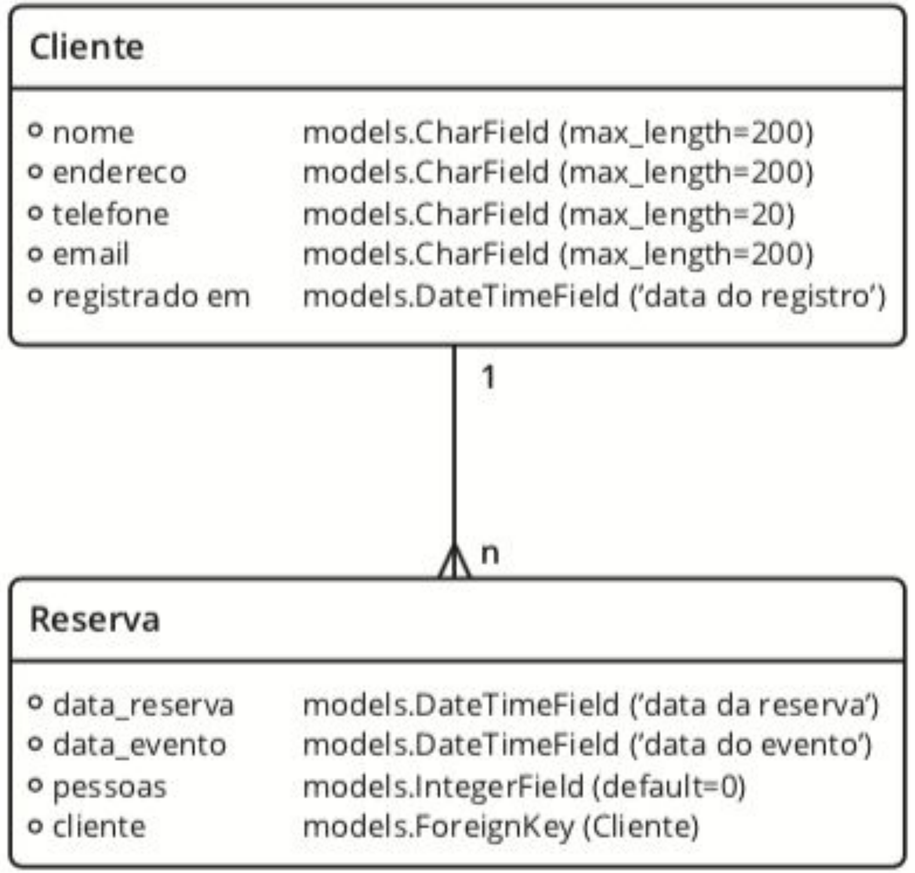
Criando os modelos

- A filosofia dos modelos é definir o modelo de dados em um lugar e derivar outras coisas a partir desse lugar;
- As informações do modelo são utilizadas para várias tarefas do Django;
- Uma das tarefas é criar o esquema do banco de dados (comandos CREATE TABLE);
- Com esses dados é criada uma API para acessar os objetos de Cliente e Reserva.
- Para isso é necessário habilitar a aplicação reservas.

Introdução ao Django

Criando os modelos

- A aplicação de reservas terá duas tabelas:
  - A tabela de clientes armazenará o nome, endereço, telefone, e-mail e a data de registro para cada um deles.
  - A tabela de reservas armaze- nará a data em que a reserva foi feita, a data do evento, quantas pessoas vão ao evento e de qual cliente é a reserva.



Introdução ao Django

Habilitando a aplicação

- De forma análoga às aplicações padrão que já vem instaladas com o Django, é necessário habilitar a aplicação no arquivo de configuração settings.py
  - Na variável INSTALLED_APPS, deve ser adicionada uma nova linha com o nome da classe que configura a aplicação “reservas.apps.ReservasConfig”.

Introdução ao Django

Aplicando as mudanças

- Para informar ao Django que foram feitas mudanças no modelo, rodamos o utilitário de gerenciamento do Django com a opção makemigrations:
  - python manage.py makemigrations reservas
- Para ver as mudanças que serão aplicadas ao banco, é usado o comando sqlmigrate:
  - python manage.py sqlmigrate reservas 0001
- Para aplicar as mudanças no banco, basta rodar o makemigrations e em seguida o comando migrate:
  - python manage.py migrate

Introdução ao Django

Utilizando a API

- É possível usar a API do Django em um shell:
  - python manage.py shell
- Esse comando já prepara o ambiente configurado pelo arquivo settings.py
- Dessa forma, a conexão ao banco, fuso horário e aplicações são configuradas. E os comandos executados no shell utilizam essas configurações

Referências