# Construindo sites com o framework Django



# Eliane Maciel

github.com/eifmaciel eifmaciel@ucs.br

- → Desenvolvedora web VendaBem Informática
- → Estudante de Ciência da Computação UCS
- → Membro Diretório Acadêmico - DAComp
- → Fundadora do Pyladies Caxias

# O que é Django?

O framework web para perfeccionistas com prazos.

Django é um framework Web Python

- → Open Source.
- → Seguro
- → Escalável



## Django

- MapeamentoObjeto-Relacional (ORM)
- → Admin
- → URLs Amigáveis
- → Sistema de Templates

- → Migrations
- → Class Based Views



# Django

#### **MVT**

- → Model
- → Template
- → View



## Iniciando o Projeto

→ Criando o Projeto → Criando APP
django-admin startproject my\_project python manage.py startapp my\_app

patogenos/settings.py

→ Servidor de Testes

python manage.py runserver



#### Models

```
from django.db import models
class Alerta(models.Model):
    data = models.DateTimeField(
        verbose_name='Data')
    mensagem = models.TextField(
       verbose_name='Nome')
    class Meta:
        verbose_name = 'Alerta'
        verbose_name_plural='Alertas'
```



# Migrations

>> python manage.py makemigrations my\_app

Cria Migrations com as alterações do models.

>> python manage.py migrate

Para todo o Projeto

>> python manage.py migrate my\_app

Apenas uma App

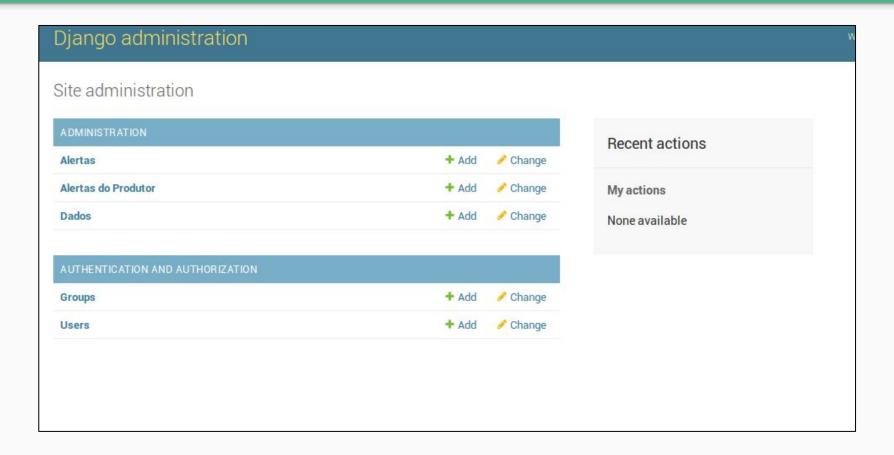


#### Admin

```
from django.contrib import admin
from administration.models import *
class AlertaAdmin(admin.ModelAdmin):
    model = Alerta
    search_fields = ('data', 'mensagem')
class MonitoramentoAdmin(admin.ModelAdmin):
    pass
class AlertaProdutorAdmin(admin.ModelAdmin):
    pass
admin.site.register(Alerta, AlertaAdmin)
admin.site.register(Monitoramento, MonitoramentoAdmin)
```



### Admin



#### Views - Class Based Views

#### Vantagens

- → Organização de código;
- → métodos HTTP específicos (GET, POST, etc.);
- → Técnicas orientadas a objeto;
  - Mixins herança múltipla;
  - Componentes reutilizáveis;



#### Views

- → View
- → TemplateView
- → DetailView
- → ListView
- → CreateView
- → UpdateView

```
from django.views.generic import ListView, CreateView, UpdateView, DetailView
from administration.models import *
from django.contrib.messages.views import SuccessMessageMixin
from django.core.urlresolvers import reverse_lazy
class MonitoramentoCreate(SuccessMessageMixin, CreateView):
    model = Monitoramento
    success_url = reverse_lazy('monitoramento_list')
    template_name = 'form_view.html'
    success_message = u'Successfully Saved!'
    fields = ['data', 'media_temp', 'chuva_soma', 'foliar', 'pontos']
class MonitoramentoList(ListView):
    model = Monitoramento
    template_name = 'monitoramento_list.html'
    queryset = Monitoramento.objects.order_by('data')
```

#### Urls

```
from django.conf.urls import url
from django.contrib import admin
from administration.views import *
from django.contrib.auth.decorators import login_required
urlpatterns = [
    url(r'^admin/', admin.site.urls),
    url(r'^monitoramento/$', login_required(MonitoramentoList.as_view()), name='monitoramento_list'),
    url(r'monitoramento/add$',
        login_required(
            MonitoramentoCreate.as_view()),
        name='monitoramento_add'),
```



## Templates

```
{% load static %}
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"</pre>
    "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html>
<head>
    {% block head %}
        <title>Patogenos</title>
    {% endblock %}
    <link href="{% static 'bootstrap.min.css' %}" rel="stylesheet">
</head>
<body>
    <div class="container">
        {% block content %}
        {% endblock %}
    </div>
</body>
</html>
```



## Templates

```
{% extends "base.html" %}
{% load i18n static %}
{% block content %}
  <h2>Monitoramentos</h2>
  <thead>
        Data
           Soma de Chuva
           Pontos
     </thead>
  {% for monitoramento in object_list %}
     {{ monitoramento.data }}
        {{ monitoramento.chuva_soma }}
        {{ monitoramento.pontos }}
     {% endfor %}
  endblock %}
```



# ObrigAda

# Perguntas

