

Modulo 7

Contents

1	introduction	1
2	Primeira URL e function based view + HttpRequest e HttpResponse3	1
3	Movendo as functions base views para os novos Apps no Django	2
4	Renderizando HTML, render, templates, INSTALLEDAPPS e TemplateDoesNotExist	2
5	Configurando templates globais com DIRS + extends para herança de templates	3
6	configurando templates globais	4
7	Arquivos parciais e includes para separar trechos dos templates(partials)	4
8	Arquivos estaticos (static files), STATICURL, STATICFILES_DIRS, load static	4
9	Usando context para enviar dados para dentro do views	5
10	trabalhando com urls dinamicas	5
11	Movendo todos os arquivos de css para global	6
12	Criando o partial postblock.html e usando include	6
13	usando block para criar blocos de posts e home	7
14	Entendendo seu HTML final + adicionando css aos posts	7
15	Criando os dados de posts (data.py) e usando loop for no template	8
16	Usando if, elif, e else dentro do template	8
17	Criando urls dinâmicas no Django URL Dispatcher, view e template	9
18	Usando a mesma url de forma estática e de forma dinâmica	9
19	Configurando um post unico no Template post.html	10
20	Exibindo o Erro 404	11
21	configurando css	11

1 introduction

Listing 1: comandos django

```
1  django-admin startproject project .# inicia o projeto django com o manage.py na raiz
2  python manage.py runserver #sobe o servidor
```

http codes <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Status>

2 Primeira URL e function based view + HttpRequest e HttpResponse3

Listing 2: urls.py

```
1  def home (request):
2      print("home")
3      return HttpResponse("HOME")
4
5  def my_view(request):
6      return HttpResponse("hello world")
7
8
9  urlpatterns = [
10     path('admin/', admin.site.urls),
11     path('blog/', my_view),
12     path('', home),
13 ]
```

3 Movendo as functions base views para os novos Apps no Django

commandos

Listing 3: comando django

```
1 django-admin startapp <nome_app>
```

Aqui ocorre o aninhamento das urls que estarão nos apps

Listing 4: project/urls.py

```
1 from django.contrib import admin
2 from django.urls import include ,path
3 urlpatterns = [
4     path('', include('home.urls')),
5     path('admin/', admin.site.urls),
6     path('blog/', include('blog.urls')),
7 ]
```

Aqui são as responses de cada urls

Listing 5: blog/views.py

```
1 #from django.shortcuts import render
2 from django.http import HttpResponse
3
4 # Create your views here.
5 def blog(request):
6     return HttpResponse("BLOG")
7
8 def example(request):
9     return HttpResponse("BLOG/example")
```

Listing 6: blog/urls.py

```
1 from django.urls import path
2 from blog.views import blog, example
3
4 urlpatterns = [
5     path('', blog),
6     path('example/', example)
7 ]
```

E a mesma coisa acontece para o home

Listing 7: home/views.py

```
1 from django.http import HttpResponse
2 #from django.shortcuts import render
3
4 # Create your views here.
5 def home (request):#function base view
6     print("home")
7     return HttpResponse("HOME")
```

Listing 8: home/urls.py

```
1 from django.urls import path
2 from home.views import home
3
4 urlpatterns = [
5     path('', home),
6 ]
```

4 Renderizando HTML, render, templates, INSTALLEDAPPS e TemplateDoesNotExist

Listing 9: project/settings.py

```
1 INSTALLED_APPS = [
2     'django.contrib.admin',
3     'django.contrib.auth',
4     'django.contrib.contenttypes',
5     'django.contrib.sessions',
6     'django.contrib.messages',
7     'django.contrib.staticfiles',
8     'home',
9     'blog',
10 ]#adicionando configuracoes dos apps
```

um pasta com o nome do app, dentro da pasta templates, garante segurança na importação

Listing 10: blog/templates/blog/blog.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Document</title>
7 </head>
8 <body>
9     <b> BLOG</b>
10 </html>
```

Listing 11: blog/templates/blog/example.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Document</title>
7 </head>
8 <body>
9     <strong>Example</strong>
10 </html>
```

Listing 12: blog/views.py

```
1 from django.shortcuts import render
2
3 # Create your views here.
4 def blog(request):
5     return render(request,
6                   'blog.html')
7
8 def example(request):
9     return render(request,
10                  'example.html')
```

Listing 13: home/templates/home/home.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Document</title>
7 </head>
8 <body>
9     <strong>HOME</strong>
10 </html>
```

Listing 14: home/views.py

```
1 from django.shortcuts import render
2
3 # Create your views here.
4 def home (request): #function base view
5     print("home")
6     return render(
7         request,
8         'home.html'
9     )
```

5 Configurando templates globais com DIRS + extends para herança de templates

Listing 15: project/settings.py

```
1 # aqui estao as configuracoes dos templates usados no app
2 TEMPLATES = [
3 {
4     'BACKEND': 'django.template.backends.django.DjangoTemplates',
5     'DIRS': [
6         \textcolor{blue}{BASE_DIR / 'base'}
7     ],
8     'APP_DIRS': True,
9     'OPTIONS': {
10         'context_processors': [
11             'django.template.context_processors.debug',
12             'django.template.context_processors.request',
13             'django.contrib.auth.context_processors.auth',
14             'django.contrib.messages.context_processors.messages',
15         ],
16     },
17 },
18 ]
```

Listing 16: base/global/base.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Document</title>
7 </head>
8 <body>
9     \textcolor{blue}{<h1> {% block texto %} BASE {% endblock %}</h1>}
10 </body>
11 </html>
```

Listing 17: home/views.py

```
1 from django.shortcuts import render
2 # Create your views here.
3 def home (request):#function base view
4     print("home")
5     return render(
6         request,
7         'home/home.html'
8     )
```

extends significa estender para o html da base

Listing 18: home/templates/home.html

```
1 {% extends "global/base.html" %}
2 {% block texto %} MUDAR O texto{% endblock texto %}
```

6 configurando templates globais

Listing 19: base/global/base.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Document</title>
7 </head>
8 <body>
9     <h1> {% block texto %} BASE {% endblock %}</h1>
10 </body>
11 </html>
```

Listing 20: home/templates/home.html

```
1 {% extends "global/base.html" %}
2 {% block texto %} MUDAR O texto{% endblock texto %}
```

Listing 21: blog/templates/blog.html

```
1 {% extends "global/base.html" %}
2 {% block texto %} bem vindo ao blog {% endblock texto %}
```

Listing 22: blog/templates/example.html

```
1 {% extends "global/base.html" %}
2 {% block texto %} Example {% endblock texto %}
```

7 Arquivos parciais e includes para separar trechos dos templates(partial)

Listing 23: base/global/partials/head.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Document</title>
7 </head>
```

Listing 24: base/global/partials/paragrafo.html

```
1 <p> um texto qualquer </p>
```

Listing 25: base/global/base.html

```
1 {% include "global/partials/head.html" %}
2 <h1> {% block texto %} BASE {% endblock %}</h1>
3 {% include "global/partials/paragrafo.html" %}
4 {% include "global/partials/paragrafo.html" %}
5 {% include "global/partials/paragrafo.html" %}
6 {% include "global/partials/paragrafo.html" %}
7 </body>
8 </html>
```

8 Arquivos estaticos (static files), STATICURL, STATICFILESDIRS, load static

configurando o settings do project para uma nova pasta static em base

Listing 26: project/settings.py

```
1 # Static files (CSS, JavaScript, Images)
2 # https://docs.djangoproject.com/en/5.0/howto/static-files/
3
4 STATIC_URL = 'static/'
5 STATICFILES_DIRS = [
6     BASE_DIR / 'base' / 'static'
7 ]
```

Listing 27: home/css/blue.css

```
1 body{
2     background: blue;
3 }
```

Listing 28: global/css/red.css

```
1 body{
2     background: red;
3 }
```

Listing 29: base/global/partials/head.html

```
1 {% load static %}<!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <title>Document</title>
7     <link rel = "stylesheet" href = "{% static 'home/css/blue.css' %}">
8     <link rel = "stylesheet" href = "{% static 'global/css/red.css' %}">
9 </head>
```

9 Usando context para enviar dados para dentro do views

Listing 30: home/templates/home/home.html

```
1 {% extends "global/base.html" %}
2 {% block texto %}
3 {{ text }}
4 {% endblock texto %}
```

Listing 31: home/views.py

```
1 from django.shortcuts import render
2
3 # Create your views here.
4 def home (request):#function base view
5     print("home")
6     context = {
7         'text' : 'estamos aqui'
8     }
9
10    return render(
11        request,
12        'home/home.html',
13        context
14    )
```

Listing 32: blog/templates/blog/example.html

```
1 {% extends "global/base.html" %}
2 {% block texto %} {{ text }} {% endblock texto %}
```

Listing 33: blog/templates/blog/example.html

```
1 {% extends "global/base.html" %}
2 {% block texto %} {{ text }} {% endblock texto %}
```

Listing 34: blog/views.py

```
1 from django.shortcuts import render
2
3 # Create your views here.
4 def blog(request):
5     context = {
6         'text' : 'ola aqui do blog'
7     }
8     return render(request,
9         'blog/blog.html',
10        context
11    )
12
13 def example(request):
14     context = {
15         'text' : 'Example'
16     }
17     return render(request,
18         'blog/example.html',
19        context
20    )
```

10 trabalhando com urls dinamicas

Listing 35: base/global/partials/menu.html

```
1 <nav>
2 <ul>
3   <li>
4     <a href= "{% url 'home:index' %}">Home</a>
5   </li>
6
7   <li>
8     <a href= "{% url 'blog:home' %}">Blog</a>
9   </li>
10
11   <li>
12     <a href= "{% url 'blog:example' %}">Example</a>
13   </li>
14 </ul>
15 </nav>
```

Adicionando namespace a url

Listing 36: blog/urls.py

```
1 from django.urls import path
2 from blog.views import blog, example
3
4 app_name = 'blog'
5 urlpatterns = [
6     path('', blog, name='home'),
7     path('example/', example, name='example')
8 ]
```

Listing 37: home/urls.py

```
1 from django.urls import path
2 from home.views import home
3
4 app_name = 'home'
5 urlpatterns = [
6     path('', home, name= 'index'),
7 ]
```

11 Movendo todos os arquivos de css para global

Listing 38: base/global/partials/head.html

```
1 {% load static %}<!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>{{title}} Site do Marcos </title>
7   <link rel="stylesheet" href="{% static 'global/css/style.css' %}">
8 </head>
```

Listing 39: base/static/global/css/style.css

```
1 *{
2     margin: 0;
3     padding: 0;
4     box-sizing: border-box;
5 }
```

12 Criando o partial postblock.html e usando include

Listing 40: base/static/global/css/style.css

```
1 /* Reset */
2 *,
3 *:after,
4 *:before {
5     margin: 0;
6     padding: 0;
7     box-sizing: border-box;
8 }
9
10 html {
11     font-size: 62.5%;
12 }
13
14 body {
15     font-size: 1.6rem;
16     background: #f1f1f1;
17     font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto,
18         Oxygen, Ubuntu, Cantarell, 'Open Sans', 'Helvetica Neue', sans-serif;
19 }
```

Listing 41: base/global/base.html

```
1 {% include "global/partials/head.html" %}
2 {% include "global/partials/menu.html" %}
3
4     <h1> {% block texto %} BASE {% endblock %}</h1>
5     {% include "global/partials/paragrafo.html" %}
6     {% include "global/partials/paragrafo.html" %}
7
8     <main class="posts">
9         {% include "global/partials/postblock.html" %}
10        {% include "global/partials/postblock.html" %}
11        {% include "global/partials/postblock.html" %}
12
13
14    </main>
15
16
17
18 </body>
19 </html>
```

Listing 42: base/global/partials/postblock.html

```
1 <article>
2 <header>
3     <h2 class="post__title">
4         Lorem ipsum dolor sit amet consectetur adipisicing elit. Natus placeat blanditiis
5         ipsam quas est, provident, exercitationem illo inventore molestias iure beatae
6         soluta aliquid iusto facere corporis, quaerat aspernatur debitis laudantium?
7
8     </h2>
9 </header>
10 <div class="post__body">
11     Lorem ipsum dolor sit amet consectetur adipisicing elit. Eum laboriosam beatae veritatis
12     dolorem natus voluptatibus fugiat sequi eaque exercitationem dolor nobis assumenda
13     facere, praesentium aspernatur id odit aliquam ipsa nisi.
14
15 </div>
16 </article>
```

13 usando block para criar blocos de posts e home

Listing 43: base/global/base.html

```
1 {% include "global/partials/head.html" %}
2 {% include "global/partials/menu.html" %}
3
4     <h1> {% block texto %} BASE {% endblock %}</h1>
5     {% include "global/partials/paragrafo.html" %}
6     {% include "global/partials/paragrafo.html" %}
7
8     <main class="content">
9         {% block posts %} {% endblock %}
10        {% block home %} {% endblock home %}
11
12    </main>
13
14
15
16 </body>
17 </html>
```

Listing 44: blog/templates/blog/blog.html

```
1 {% extends "global/base.html" %}
2 {% block texto %} {{text}} {% endblock texto %}
3 {% block posts %}
4
5 {% include "global/partials/postblock.html" %}
6 {% endblock posts %}
7
8 {% block home %}
9 <h1> Blog </h1>
10 {% endblock home %}
```

14 Entendendo seu HTML final + adicionando css aos posts

- .content altera os content da pagina html
- .post altera diretamente os posts
- @media altera o formato das caixas de posts

Listing 45: base/static/global/css/style.css

```
1
2 .content {
3     display: grid;
4     gap: 1.5rem;
5     padding: 1.5rem;
6 }
```

```
7
8     .post{
9         background : #fff;
10        padding: 1.5rem;
11        box-shadow: 5px 2px 5px rgba(0, 0, 0, 0.9)
12    }
13
14    @media (min-width: 600px){
15        .content{
16            grid-template-columns: repeat(auto-fill, minmax(32rem, 1fr));
17        }
18    }
```

15 Criando os dados de posts (data.py) e usando loop for no template

O codigo do data.py foi criado apartir de dados de uma api que usa json.
O codigo foi modificado pra que apartir dos dados do data.py, gere na pagina blog, os diversos posts.

Listing 46: base/global/partials/postblock.html

```
1     <article class="post">
2     <header>
3         <h2 class="post__title">Lorem ipsum dolor sit amet.{{post.title}}</h2>
4     </header>
5     <div class="post__body">{{post.body}}</div>
6 </article>
```

Listing 47: blog/templates/blog/blog.html

```
1 {% extends 'global/base.html' %}
2
3 {% block texto %} {{ text }} {% endblock texto %}
4
5 {% block posts %}
6 {% for post in posts %}
7 {% include 'global/partials/postblock.html' %}
8 {% endfor %}
9 {% endblock posts %}
```

Listing 48: blog/views.py

```
1 def blog(request):
2     context = {
3         'text' : 'ola aqui do blog',
4         'posts' : data.posts
5     }
6     return render(request,
7                   'blog/blog.html',
8                   context
9                   )
```

16 Usando if, elif, e else dentro do template

No exemplo passado em aula, o if é usado para checar a existencia da da variavel text, se não existir ela nao aparece na pagina html

Listing 49: blog/views.py

```
1 from django.shortcuts import render
2 from . import data
3 # Create your views here.
4 def blog(request):
5     context = {
6         'text' : 'ola aqui do blog',
7         'posts' : data.posts
8     }
9     return render(request,
10                   'blog/blog.html',
11                   context
12                   )
13
14 def example(request):
15     context = {
16         'text' : 'Example'
17     }
18     return render(request,
19                   'blog/example.html',
20                   context
21                   )
```

Listing 50: base/global/base.html

```
1     {% include "global/partials/head.html" %}
2 {% include "global/partials/menu.html" %}
3     {% if text %}
4         <h1> {% block texto %} BASE {% endblock %}</h1>
5     {% endif %}
6
7     {% include "global/partials/paragrafo.html" %}
```



```
8         {% include "global/partials/paragrafo.html" %}
9
10        <main class="content">
11            {% block posts %}{% endblock posts %}
12            {% block home %}{% endblock home %}
13        </main>
14
15
16
17    </body>
18</html>
```

17 Criando urls dinâmicas no Django URL Dispatcher, view e template

Listing 51: blog/urls.py

```
1 from django.urls import path
2 from blog.views import blog, example
3
4 app_name = 'blog'
5 urlpatterns = [
6     path('', blog, name='home'),
7     path('post/<id>', blog, name='post'),
8     path('example/', example, name='example')
9 ]
```

Listing 52: blog/views.py

```
1 from django.shortcuts import render
2 from . import data
3 # Create your views here.
4 def blog(request):
5     context = {
6         'text' : 'ola aqui do blog',
7         'posts' : data.posts
8     }
9     return render(request,
10                   'blog/blog.html',
11                   context
12                   )
13
14 def post(request, id):
15     print('post', id)
16     context = {
17         #'text' : 'ola aqui do blog',
18         'posts' : data.posts
19     }
20     return render(request,
21                   'blog/blog.html',
22                   context
23                   )
24
25
26 def example(request):
27     context = {
28         'text' : 'Example'
29     }
30     return render(request,
31                   'blog/example.html',
32                   context
33                   )
```

Fazendo o reverse match

Listing 53: Commit

```
1 <a href="{% url 'blog:post' post.id %}">
2     {{post.title}}
3 </a>
```

Listing 54: base/global/partials/postblock.html

```
1 <article class="post">
2 <header>
3     <a href="{% url 'blog:post' post.id %}">
4         {{post.title}}
5     </a>
6     <h2 class="post__title">Lorem ipsum dolor sit amet.{{post.title}}</h2>
7 </header>
8 <div class="post__body">{{post.body}}</div>
9 </article>
```

18 Usando a mesma url de forma estática e de forma dinâmica

É uma boa pratica que as urls sejam listadas das mais especificas para as mais gerais

Listing 55: blog/urls.py

```
1 from django.urls import path
```

```
2 from blog.views import blog, example, post
3
4 app_name = 'blog'
5 urlpatterns = [
6
7     path('<int:id>/', post, name='post'),
8     path('example/', example, name='example'),
9     path('', blog, name='home'),
10 ]
```

Listing 56: blog/views.py

```
1 from django.shortcuts import render
2 from typing import Any
3 from . import data
4 from django.http import HttpRequest
5 # Create your views here.
6 #Django url dispatcher
7 def blog(request):
8     context = {
9         'text' : 'ola aqui do blog',
10        'posts' : data.posts
11    }
12    return render(request,
13                  'blog/blog.html',
14                  context
15                  )
16
17 def post(request: HttpRequest, post_id):
18     found_post : dict[str, Any] | None = None
19     for post in data.posts:
20         if post['id'] == post_id:
21             found_post = post
22             break
23     if found_post is None:
24         raise Exception('post nao existe.')
25
26     print('post', id)
27     context = {
28         #'text' : 'ola aqui do blog',
29         'post' : found_post,
30         'title': found_post['title'] + ' - '
31     }
32     return render(request,
33                   'blog/post.html',
34                   context
35                   )
```

19 Configurando um post unico no Template post.html

Listing 57: blog/templates/blog/post.html

```
1 {% extends 'global/base.html' %}
2
3 {% block texto %} {{ text }} {% endblock texto %}
4
5 {% block posts %}
6 <article class="post single-post">
7     <header>
8         <a href="{% url "blog:post" post.id %}">
9             {{post.title}}
10         </a>
11         <h2 class="post__title">Lorem ipsum dolor sit amet.{{post.title}}</h2>
12     </header>
13     <div class="post__body">{{post.body}}</div>
14 </article>
15 {% endblock posts %}
```

Listing 58: blog/views.py

```
1 def post(request, post_id):
2     found_post = None
3     for post in data.posts:
4         if post['id'] == post_id:
5             found_post = post
6             break
7
8
9     print('post', id)
10    context = {
11        #'text' : 'ola aqui do blog',
12        'post' : found_post,
13        'title': found_post['title'] + ' - '
14    }
```

configurando a vizualização de um unico post

Listing 59: global/css/style.css

```
1 @media (min-width: 600px){
```

```
2     .content:not(:has(.single-post)){
3         grid-template-columns: repeat(auto-fill, minmax(32rem, 1fr));
4     }
5 }
```

20 Exibindo o Erro 404

Listing 60: blog/views.py

```
1     def post(request: HttpRequest, post_id):
2         found_post : dict[str, Any] | None = None
3         for post in data.posts:
4             if post['id'] == post_id:
5                 found_post = post
6                 break
7         if found_post is None:
8             raise Http404('post nao existe.')
9
10        print('post', id)
11        context = {
12            #'text' : 'ola aqui do blog',
13            'post' : found_post,
14            'title': found_post['title'] + ' - '
15        }
```

21 configurando css

Listing 61: global/css/rstyle.css

```
1     *,
2     *:after,
3     *:before {
4         margin: 0.9rem;
5         padding: 1.5rem;
6         box-sizing: border-box;
7     }
8
9     html {
10        font-size: 62.5%;
11    }
12    a {
13        color:rgb(76, 133, 219);
14        text-decoration: none;
15    }
16
17    a :hover {
18        color:rgb(76, 133, 219);
19        text-decoration: underline;
20    }
21
22    body {
23        font-size: 1.6rem;
24        background: #f1f1f1;
25        font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen, Ubuntu,
26            Cantarell, 'Open Sans', 'Helvetica Neue', sans-serif;
27    }
28
29    .content {
30        display: grid;
31        gap: 1.5rem;
32        padding: 1.5rem;
33    }
34
35    .post{
36        background : #fff;
37        padding: 1.5rem;
38        box-shadow: 5px 2px 5px rgba(0, 0, 0, 0.9)
39    }
40
41    .menu{
42        background :rgba(0, 0, 0, 0.9);
43        padding: 0 1.5rem;
44    }
45
46    .menu__links a{
47        display:block ;
48        color: #f1f1f1;
49        padding: 1.5rem;
50    }
51
52    .menu__links {
53        list-style: None;
54        display: flex;
55    }
56
```

```
57
58
59 @media (min-width: 600px){
60   .content:not(:has(.single-post)){
61     grid-template-columns: repeat(auto-fill, minmax(32rem, 1fr));
62   }
63 }
```