Modulo 7

Contents

1 introduction

Listing 1: comandos django

```
django-admin  startproject project .# inicia o projeto django com o manage.py na raiz
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

```
def home (request):
    print("home")
    return HttpResponse("HOME")

def my_view(request):
    return HttpResponse("hello world")


urlpatterns = [
    path('admin/', admin.site.urls),
    path('blog/', my_view),
    path('', home),
]
```

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

comnandos

Listing 3: comando django

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

Aqui ocorre o aninhamento das urls que estarão nos apps

Listing 4: project/urls.py

```
from django.contrib import admin
from django.urls import include ,path
urlpatterns = [
    path('', include('home.urls')),
    path('admin/', admin.site.urls),
    path('blog/', include('blog.urls')),
]
```

Aqui são as responses de cada urls

Listing 5: blog/views.py

```
#from django.shortcuts import render
from django.http import HttpResponse

# Create your views here.
def blog(request):
    return HttpResponse("BLOG")

def example(request):
    return HttpResponse("BLOG/example")
```

Listing 6: blog/urls.py

```
from django.urls import path
from blog.views import blog, example

urlpatterns = [
    path('', blog),
    path('example/', example)
]
```

E a mesma coisa acontece para o home

Listing 7: home/views.py

```
from django.http import HttpResponse
#from django.shortcuts import render

# Create your views here.
def home (request):#function base view
    print("home")
    return HttpResponse("HOME")
```

Listing 8: home/urls.py

```
from django.urls import path
from home.views import home

urlpatterns = [
    path('', home),
]
```

4 Renderizando HTML, render, templates, INSTALLEDAPPS e TemplateDoesNotExist

Listing 9: project/settings.py

```
INSTALLED_APPS = [
'django.contrib.admin',
'django.contrib.auth',
'django.contrib.contenttypes',
'django.contrib.sessions',
'django.contrib.messages',
'django.contrib.staticfiles',
'home',
'blog',
]#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

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <b> BLOG</b>
</html>
```

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

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <strong>Example</strong>
</html>
```

Listing 12: blog/views.py

```python
from django.shortcuts import render

# Create your views here.
def blog(request):
    return render(request,
                'blog.html')

def example(request):
    return render(request,
                'example.html')
```

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

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <strong>HOME</strong>
</html>
```

Listing 14: home/views.py

```python
from django.shortcuts import render

# Create your views here.
def home (request):#function base view
    print("home")
    return render(
        request,
        'home.html'
    )
```

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

Listing 15: project/settings.py

```python
# aqui estao as configuracoes dos templates usados no app
TEMPLATES = [
{
    'BACKEND': 'django.template.backends.django.DjangoTemplates',
    'DIRS': [
        \textcolor{blue}{BASE_DIR / 'base'}
        ],
    'APP_DIRS': True,
    'OPTIONS': {
        'context_processors': [
            'django.template.context_processors.debug',
            'django.template.context_processors.request',
            'django.contrib.auth.context_processors.auth',
            'django.contrib.messages.context_processors.messages',
        ],
    },
},
]
```

Listing 16: base/global/base.html

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

Listing 17: home/views.py

```
from django.shortcuts import render
# Create your views here.
def home (request):#function base view
    print("home")
    return render(
        request,
        'home/home.html'
    )
```

extends significa extender para o html da base

Listing 18: home/templates/home.html

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

6 configurando templates globais

Listing 19: base/global/base.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1> {% block texto  %} BASE {% endblock %}</h1>
</body>
</html>
```

Listing 20: home/templates/home.html

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

Listing 21: blog/templates/blog.html

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

Listing 22: blog/templates/example.html

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

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

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

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

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

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

Listing 25: base/global/base.html

```
    {% include "global/partials/head.html" %}
    <h1> {% block texto  %} BASE {% endblock %}</h1>
    {% include "global/partials/paragrafo.html" %}
    {% include "global/partials/paragrafo.html" %}
    {% include "global/partials/paragrafo.html" %}
    {% include "global/partials/paragrafo.html" %}
</body>
</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

```
    # Static files (CSS, JavaScript, Images)
    # https://docs.djangoproject.com/en/5.0/howto/static-files/

    STATIC_URL = 'static/'
    STATICFILES_DIRS = [
        BASE_DIR / 'base'/ 'static'
]
```

Listing 27: home/css/blue.css

```css
    body{
        background: blue;
    }
```

Listing 28: global/css/red.css

```css
    body{
        background: red;
    }
```

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

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

9 Usando context para enviar dados para dentro do views

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

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

Listing 31: home/views.py

```python
    from django.shortcuts import render

    # Create your views here.
    def home (request):#function base view
        print("home")
        context = {
                'text' : 'estamos aqui'
            }

        return render(
            request,
            'home/home.html',
            context
        )
```

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

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

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

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

Listing 34: blog/views.py

```python
    from django.shortcuts import render

# Create your views here.
def blog(request):
    context = {
        'text' : 'ola aqui do blog'
    }
    return render(request,
                  'blog/blog.html',
                  context
                  )

def example(request):
    context = {
        'text' : 'Example'
    }
    return render(request,
                  'blog/example.html',
                  context
                  )
```

10 trabalhando com urls dinamicas

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

```html
    <nav>
    <ul>
        <li>
            <a href= "{% url 'home:index'%}">Home</a>
        </li>

        <li>
            <a href= "{% url 'blog:home'%}">Blog</a>
        </li>

        <li>
            <a href= "{% url 'blog:example'%}">Example</a>
        </li>
    </ul>
</nav>
```

Adicionando namespace a url

Listing 36: blog/urls.py

```python
from django.urls import path
from blog.views import blog, example

app_name = 'blog'
urlpatterns = [
    path('', blog, name='home'),
    path('example/', example, name='example')
]
```

Listing 37: home/urls.py

```python
from django.urls import path
from home.views import home

app_name = 'home'
urlpatterns = [
    path('', home, name= 'index'),
]
```

11 Movendo todos os arquivos de css para global

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

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

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

```css
    *{
        margin: 0;
        padding: 0;
        box-sizing: border-box;
    }
```

12 Criando o partial postblock.html e usando include

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

```css
    /* Reset */
*,
*:after,
*:before {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

html {
  font-size: 62.5%;
}

body {
  font-size: 1.6rem;
  background: #f1f1f1;
  font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto,
    Oxygen, Ubuntu, Cantarell, 'Open Sans', 'Helvetica Neue', sans-serif;
}
```

Listing 41: base/global/base.html

```html
{% include "global/partials/head.html" %}
{% include "global/partials/menu.html" %}

        <h1> {% block texto  %} BASE {% endblock %}</h1>
        {% include "global/partials/paragrafo.html" %}
        {% include "global/partials/paragrafo.html" %}

        <main class="posts">
            {% include "global/partials/postblock.html" %}
            {% include "global/partials/postblock.html" %}
            {% include "global/partials/postblock.html" %}


        </main>



    </body>
</html>
```

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

```html
    <article>
    <header>
        <h2 class="post__title">
            Lorem ipsum dolor sit amet consectetur adipisicing elit. Natus placeat blanditiis
                ipsam quas est, provident, exercitationem illo inventore molestias iure beatae
                soluta aliquid iusto facere corporis, quaerat aspernatur debitis laudantium?
        </h2>
    </header>
    <div class="post__body">
        Lorem ipsum dolor sit amet consectetur adipisicing elit. Eum laboriosam beatae veritatis
            dolorem natus voluptatibus fugiat sequi eaque exercitationem dolor nobis assumenda
            facere, praesentium aspernatur id odit aliquam ipsa nisi.
    </div>
</article>
```

13 usando block para criar blocos de posts e home

Listing 43: base/global/base.html

```html
    {% include "global/partials/head.html" %}
    {% include "global/partials/menu.html" %}

            <h1> {% block texto  %} BASE {% endblock %}</h1>
            {% include "global/partials/paragrafo.html" %}
            {% include "global/partials/paragrafo.html" %}

            <main class="content">
                {% block posts  %} {% endblock %}
                {% block home  %}{% endblock home  %}

            </main>



        </body>
    </html>
```

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

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

    {% include "global/partials/postblock.html" %}
    {% endblock posts %}

    {% block home %}
    <h1> Blog </h1>
    {% endblock home %}
```

14 Entendendo seu HTML final + adiconando 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

```css

.content {
    display: grid;
    gap: 1.5rem;
    padding: 1.5rem;
  }

  .post{
    background : #fff;
    padding: 1.5rem;
    box-shadow: 5px 2px 5px rgba(0, 0, 0, 0.9)
  }

  @media (min-width: 600px){
    .content{
      grid-template-columns: repeat(auto-fill, minmax(32rem, 1fr));
    }
  }
```

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.htmll

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

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

```
{% extends 'global/base.html' %}

{% block texto %} {{ text }} {% endblock texto %}

{% block posts %}
{% for post  in posts  %}
{% include 'global/partials/postblock.html' %}
{% endfor %}
{% endblock posts %}
```

Listing 48: blog/views.py

```
    def blog(request):
    context = {
        'text' : 'ola aqui do blog',
        'posts' : data.posts
    }
    return render(request,
                  'blog/blog.html',
                  context
                  )
```

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

```
    from django.shortcuts import render
from . import data
# Create your views here.
def blog(request):
    context = {
        'text' : 'ola aqui do blog',
        'posts' : data.posts
    }
    return render(request,
                  'blog/blog.html',
                  context
                  )

def example(request):
    context = {
        'text' : 'Example'
    }
    return render(request,
                  'blog/example.html',
                  context
                  )
```

Listing 50: base/global/base.html

```
    {% include "global/partials/head.html" %}
{% include "global/partials/menu.html" %}
        {% if text %}
        <h1> {% block texto  %} BASE {% endblock %}</h1>
        {% endif %}

        {% include "global/partials/paragrafo.html" %}
        {% include "global/partials/paragrafo.html" %}

        <main class="content">
            {% block posts  %}{% endblock posts %}
            {% block home  %}{% endblock home  %}
        </main>



    </body>
</html>
```

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

Listing 51: blog/urls.py

```python
from django.urls import path
from blog.views import blog, example

app_name = 'blog'
urlpatterns = [
    path('', blog, name='home'),
    path('post/<id>', blog, name='post'),
    path('example/', example, name='example')
]
```

Listing 52: blog/views.py

```python
    from django.shortcuts import render
    from . import data
    # Create your views here.
    def blog(request):
        context = {
            'text' : 'ola aqui do blog',
            'posts' : data.posts
        }
        return render(request,
                      'blog/blog.html',
                      context
                      )

    def post(request, id):
        print('post', id)
        context = {
            #'text' : 'ola aqui do blog',
            'posts' : data.posts
        }
        return render(request,
                      'blog/blog.html',
                      context
                      )


    def example(request):
        context = {
            'text' : 'Example'
        }
        return render(request,
                      'blog/example.html',
                      context
                      )
```

Fazendo o reverse match

Listing 53: Commit

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

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

```
    <article class="post">
    <header>
      <a href="{% url "blog:post" post.id %}">
        {{post.title}}
      </a>
      <h2 class="post__title">Lorem ipsum dolor sit amet.{{post.title}}</h2>
    </header>
    <div class="post__body">{{post.body}}</div>
  </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

```python
from django.urls import path
from blog.views import blog, example, post

app_name = 'blog'
urlpatterns = [

    path('<int:id>/',post, name='post'),
    path('example/', example, name='example'),
    path('', blog, name='home'),
]
```

Listing 56: blog/views.py

```python
    from django.shortcuts import render
    from typing import Any
    from . import data
    from django.http import HttpRequest
    # Create your views here.
    #Django url dispatcher
    def blog(request):
        context = {
            'text' : 'ola aqui do blog',
            'posts' : data.posts
        }
        return render(request,
                      'blog/blog.html',
                      context
                      )

    def post(request: HttpRequest, post_id):
        found_post :  dict[str, Any] | None = None
        for post in data.posts:
            if post['id'] == post_id:
                found_post = post
                break
        if found_post is None:
            raise Exception('post nao existe.')

        print('post', id)
        context = {
            #'text' : 'ola aqui do blog',
            'post' : found_post,
            'title': found_post['title'] + ' - '
        }
        return render(request,
                      'blog/post.html',
                      context
                      )
```

19 Configurando um post unico no Template post.html

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

```html
    {% extends 'global/base.html' %}

    {% block texto %} {{ text }} {% endblock texto %}

    {% block posts %}
    <article class="post single-post">
        <header>
          <a href="{% url "blog:post" post.id %}">
            {{post.title}}
          </a>
          <h2 class="post__title">Lorem ipsum dolor sit amet.{{post.title}}</h2>
        </header>
        <div class="post__body">{{post.body}}</div>
      </article>
    {% endblock posts %}
```

Listing 58: blog/views.py

```python
    def post(request, post_id):
    found_post   = None
    for post in data.posts:
        if post['id'] == post_id:
            found_post = post
            break


    print('post', id)
    context = {
        #'text' : 'ola aqui do blog',
        'post' : found_post,
        'title': found_post['title'] + ' - '
    }
```

configurando a vizualização de um unico post

Listing 59: global/css/style.css

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

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

20 Exibindo o Erro 404

Listing 60: blog/views.py

```
    def post(request: HttpRequest, post_id):
    found_post :  dict[str, Any] | None = None
    for post in data.posts:
        if post['id'] == post_id:
            found_post = post
            break
    if found_post is None:
        raise Http404('post nao existe.')

    print('post', id)
    context = {
        #'text' : 'ola aqui do blog',
        'post' : found_post,
        'title': found_post['title'] + ' - '
    }
```

21 configurando css

Listing 61: global/css/rstyle.css

```
    *,
*:after,
*:before {
  margin: 0.9rem;
  padding: 1.5rem;
  box-sizing: border-box;
}

html {
  font-size: 62.5%;
}
a {
   color:rgb(76, 133, 219);
   text-decoration: none;
}

a :hover {
  color:rgb(76, 133, 219);
  text-decoration: underline;
}

body {
  font-size: 1.6rem;
  background: #f1f1f1;
  font-family: system-ui, -apple-system, BlinkMacSystemFont, 'Segoe UI', Roboto, Oxygen, Ubuntu,
      Cantarell, 'Open Sans', 'Helvetica Neue', sans-serif;
}

.content {
  display: grid;
  gap: 1.5rem;
  padding: 1.5rem;
}

.post{
  background : #fff;
  padding: 1.5rem;
  box-shadow: 5px 2px 5px rgba(0, 0, 0, 0.9)
}

.menu{
  background :rgba(0, 0, 0, 0.9);
  padding: 0 1.5rem;
}

.menu__links a{
  display:block ;
  color: #f1f1f1;
  padding: 1.5rem;

}

.menu__links {
  list-style: None;
  display: flex;
}

```

```css


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