**Московский государственный технический университет им. Н.Э.Баумана**

|  |  |  |
| --- | --- | --- |
|  |  |  |

**Отчет по лабораторной работе №4 по курсу**

**РИП**

«Шаблонизация»

8

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы **ИУ5-52** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| **Румянцев О.** | "19" ноября 2018 г. |

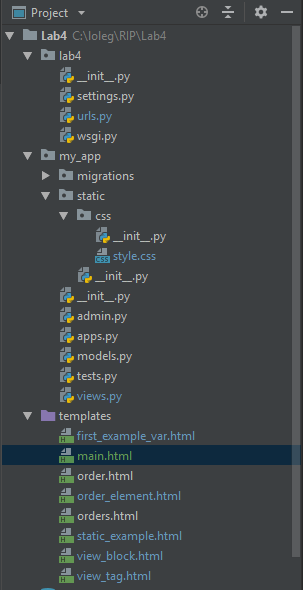
Москва, МГТУ - 2018

**Задание и порядок выполнения**

В этой ЛР вы создадите Django-проект, покажете пользователю статичную страницу, познакомитесь с конструкциями шаблонизаторов: переменные, теги, наследование шаблонов.

1. Создать проект
2. Реализовать view, в которых генерируются html-страницы
3. В шаблонах должны быть использованы рассмотренные конструкции: переменные, вложенные значения, циклы, условия
4. Все шаблоны должны расширять базовый шаблон
5. Для элементов списка использовать тег include
6. По нажатии на элемент списка должна открываться страница информации об элементе
7. Для верстки необходимо использовать Bootstrap

**Структура проекта**



**lab4**

**urls.py**

from django.conf.urls import url  
from django.contrib import admin  
from django.urls import path  
from my\_app.views import \*  
  
urlpatterns = [  
 path('', lambda x: render(x, 'main.html')),  
 path('admin/', admin.site.urls),  
 url(r'^function\_view/', function\_view),  
 url(r'^class\_based\_view/', ExampleClassBased.as\_view()),  
 url(r'^static\_view/', ExampleStaticView.as\_view()),  
 url(r'^var\_view/', var\_view),  
 url(r'^tag/', var\_tag),  
 url(r'^orders/', OrdersView.as\_view()),  
 url(r'^order/(?P<id>\d+)', OrderView.as\_view(), name='order\_url'),  
]

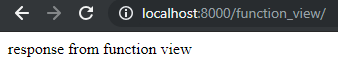
Остальные файлы генерируются автоматически.

**my\_app**

**views.py**

from django.http import HttpResponse  
from django.shortcuts import render  
from django.views import View  
import random  
  
  
def function\_view(request):  
 return HttpResponse('response from function view')  
  
  
class ExampleClassBased(View):  
 def get(self, request):  
 return HttpResponse('response from class based view')  
  
  
class ExampleStaticView(View):  
 def get(self, request):  
 return render(request, 'static\_example.html')  
  
  
def var\_view(request):  
 return render(request, 'first\_example\_var.html', {'my\_variable': 'RIP'})  
  
  
def var\_tag(request):  
 list = [random.randint(0, 10) for \_ in range(10)]  
 return render(request, 'view\_tag.html', {'list': list})  
  
  
class OrdersView(View):  
 def get(self, request):  
 data = {  
 'orders': [  
 {'title': 'First order', 'id': 1},  
 {'title': 'Second order', 'id': 2},  
 {'title': 'Third order', 'id': 3},  
 {'title': 'Fourth order', 'id': 4}  
 ]  
 }  
 return render(request, 'orders.html', data)  
  
  
class OrderView(View):  
 def get(self, request, id):  
 data = {  
 'order':{  
 'id': id  
 }  
 }  
 return render(request, 'order.html', data)

Пример использования View

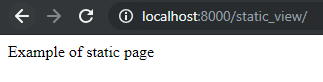


**templates**

**static\_example.html**

Пример использования шаблона

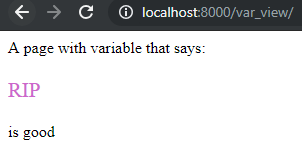
<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Title</title>  
</head>  
<body>  
 Example of static page  
</body>  
</html>



**first\_example\_var.html**

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Title</title>  
</head>  
<body>  
 A page with variable that says:<br>  
 <p style="color: #cd66cc; font-size: 130%;">{{ my\_variable }}</p> is good  
</body>  
</html>

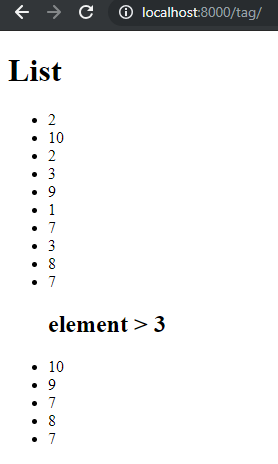
Пример использования переменной



**view\_tag.html**

Пример использования тегов

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Tag</title>  
</head>  
<body>  
 <h1>List</h1>  
 <ul>  
 {% for element **in** list %}  
 <li>{{ element }}</li>  
 {% empty %}  
 empty list  
 {% endfor %}  
 <h2>element > 3</h2>  
 {% for element **in** list %}  
 {% if element > 3 %}  
 <li>{{ element }}</li>  
 {% endif %}  
 {% empty %}  
 empty list  
 {% endfor %}  
 </ul>  
</body>  
</html>



**view\_block.html**

Файл, от которого происходит наследование

<!DOCTYPE html>  
{% load static %}  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>{% block title %}{% endblock %}</title>  
 <link rel="stylesheet" type="text/css" href="{% static 'css/style.css' %}">  
 <style>  
 body{  
 background-color: #ffffff;  
 }  
 div{  
 padding-left: 0px;  
 margin-left: 20px;  
 background-color: lawngreen;  
 font-palette: light;  
 }  
 </style>  
</head>  
<body>  
 <div>Общая часть</div>  
 <br>  
 <div>  
 {% block body %} Переопределяется в наследниках {% endblock %}  
 </div>  
</body>  
</html>

**orders.html**

Наследник от **view\_block.html**, выводящий список заказов

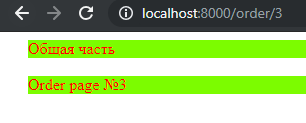
{% extends 'view\_block.html' %}  
{% load static %}  
{% block title %} Orders {% endblock %}  
  
{% block body %}  
 <ul>  
 {% for order **in** orders %}  
 {% include 'order\_element.html' **with** element=order %}  
 {% empty %}  
 <li> Empty list</li>  
 {% endfor %}  
 </ul>  
{% endblock %}



**order.html**

Страница одного из заказов. Наследник **view\_block.html.**

{% extends 'view\_block.html' %}  
  
{% block title %}Order №{{ order.id }}{% endblock %}  
  
{% block body %}Order page №{{ order.id }}{% endblock %}



**order\_element.html**

Пример использования Include

<p style="color: seagreen">  
<li>Заказ № {{ element.id }}<br>  
 <a href="{% url 'order\_url' element.id %}">{{ element.title }}</a>  
</li><br>  
</p>