Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

**«Московский государственный технический университет имени Н.Э. Баумана»   
(МГТУ им. Н.Э. Баумана)**

**Отчет по лабораторной работе № 4**

**по дисциплине «Разработка интернет приложений».**

|  |
| --- |
| ИСПОЛНИТЕЛЬ: |
| студент группы ИУ5-54 |
| Никифорова Е.Р. |

Москва, 2017

## Задание

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

* Создать проект
* Реализовать view, в которых генерируются html-страницы
* В шаблонах должны быть использованы рассмотренные конструкции:

переменные, вложенные значения, циклы, условия

* Все шаблоны должны расширять базовый шаблон
* Для элементов списка использовать тег include
* По нажатии на элемент списка должна открываться страница информации об элементе
* Для верстки необходимо использовать Bootstrap

Код программы

Zadanie.html

{% **load static** %}  
<!DOCTYPE **html**>  
<**html lang="en"**>  
 <**head**>  
 <**meta charset="utf-8"**>  
 <**meta http-equiv="X-UA-Compatible" content="IE=edge"**>  
 <**meta name="viewport" content="width=device-width, initial-scale=1"**>  
 *<!-- The above 3 meta tags \*must\* come first in the head; any other head content must come \*after\* these tags -->* <**title**>{% **block title** %}{% **endblock** %}</**title**>  
 <**link href="css/my\_style.css" rel="stylesheet" type="text/css"**>  
 <**link rel="stylesheet" type="text/css" href="**{% **static 'css/my\_style.css'** %}**"**>  
 *<!-- Bootstrap -->* <**link href="css/bootstrap.min.css" rel="stylesheet"**>  
 <**link rel="stylesheet" type="text/css" href="**{% **static 'css/bootstrap.min.css'** %}**"**>  
 *<!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->  
 <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->  
 <!--[if lt IE 9]>  
 <script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>  
 <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>  
 <![endif]-->* </**head**>  
 <**body**>  
 <**div class="razm"**>  
 <**nav class="navbar navbar-default"**>  
 <**div class="container-fluid"**>  
 *<!-- Brand and toggle get grouped for better mobile display -->* <**div class="navbar-header"**>  
 <**button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#bs-example-navbar-collapse-1" aria-expanded="false"**>  
 <**span class="sr-only"**>Toggle navigation</**span**>  
 <**span class="icon-bar"**></**span**>  
 <**span class="icon-bar"**></**span**>  
 <**span class="icon-bar"**></**span**>  
 </**button**>  
 <**a class="navbar-brand" href="#"**>Brand</**a**>  
 </**div**>  
  
 *<!-- Collect the nav links, forms, and other content for toggling -->* <**div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1"**>  
 <**ul class="nav navbar-nav navbar-right"**>  
 <**li class="dropdown"**>  
 <**a href="#" class="dropdown-toggle" data-toggle="dropdown" role="button" aria-haspopup="true" aria-expanded="false"**>Dropdown <**span class="caret"**></**span**></**a**>  
 <**ul class="dropdown-menu"**>  
 <**li**><**a href="#"**>Action</**a**></**li**>  
 <**li**><**a href="#"**>Another action</**a**></**li**>  
 <**li**><**a href="#"**>Something else here</**a**></**li**>  
 <**li role="separator" class="divider"**></**li**>  
 <**li**><**a href="#"**>Separated link</**a**></**li**>  
 </**ul**>  
 </**li**>  
 </**ul**>  
 </**div**>*<!-- /.navbar-collapse -->* </**div**>*<!-- /.container-fluid -->* </**nav**>  
 {% **block body** %}Переопределяется в наследниках{% **endblock** %}  
 </**div**>  
  
  
  
 *<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->* <**script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"**></**script**>  
 *<!-- Include all compiled plugins (below), or include individual files as needed -->* <**script src="static/js/bootstrap.min.js"**></**script**>  
 </**body**>  
</**html**>

Zadanie1.html

{% **extends 'zadanie.html'** %}  
  
{% **block title** %}{% **endblock** %}  
{% **block body** %}  
 {% **for z in orders** %}  
 {% **include 'zadanie2.html' with element**=**z** %}  
 {% **endfor** %}  
{% **endblock** %}

Zadanie2.html

<**div class="alert alert-success" role="alert"**>  
 <**a href="**{% **url 'zadanie1\_url' element**.**id** %}**" class="proba"**>{{ **element**.**title** }}</**a**>  
</**div**>

El\_zadanie.html

{% **extends 'zadanie.html'** %}  
  
{% **block title** %}{% **endblock** %}  
{% **block body** %}  
 <**div class="alert alert-success" role="alert"**>  
 {{ **order**.**id** }}  
 </**div**>  
{% **endblock** %}

My\_style.css

.**razm**{  
 **max-width**: 70%;  
 **margin-left**: **auto**;  
 **margin-right**: **auto**;  
}  
.**form-signin**{  
 **max-width**: 60%;  
 **padding**: 15**px**;  
 **margin**: 0 **auto**;  
}  
.**error**{  
 **color**: **red**;  
}

Views.py

**from** django.shortcuts **import** render  
**from** django.http **import** HttpResponse  
**from** django.views.generic **import** View  
*# Create your views here.***def** function\_view(request):  
 **return** HttpResponse(**'pesponse from function view'**)  
  
**class** ExampleClassBased(View):  
 **def** get(self, request):  
 **return** HttpResponse(**'response from class based view'**)  
  
**class** ExampleView(View):  
 **def** get(self, request):  
 **return** render(request, **'example.html'**, {**'list'**:[**'1'**, **'2'**, **'3'**, 4, 5]})  
  
**class** OrdersView(View):  
 **def** get(self, request):  
 data = {  
 **'orders'**: [  
 {**'title'**: **'Первый заказ'**, **'id'**: 1},  
 {**'title'**: **'Второй заказ'**, **'id'**: 2},  
 {**'title'**: **'Третий заказ'**, **'id'**: 3}  
 ]  
 }  
 **return** render(request, **'orders.html'**, data)  
  
**class** OrderView(View):  
 **def** get(self, request, id):  
 data = {  
 **'order'**: {  
 **'id'**: id  
 }  
 }  
 **return** render(request, **'order.html'**, data)  
  
  
**class** ZadanieView(View):  
 **def** get(self, request):  
 data = {  
 **'orders'**: [  
 {**'title'**: **'Первый заказ'**, **'id'**: 1},  
 {**'title'**: **'Второй заказ'**, **'id'**: 2},  
 {**'title'**: **'Третий заказ'**, **'id'**: 3}  
 ]  
 }  
 **return** render(request, **'zadanie1.html'**, data)  
  
**class** Zadanie1View(View):  
 **def** get(self, request, id1):  
 data = {  
 **'order'**: {  
 **'id'**: id1  
 }  
 }  
 **return** render(request, **'el\_zadanie.html'**, data)

urls.py

*"""untitled2 URL Configuration  
  
The `urlpatterns` list routes URLs to views. For more information please see:  
 https://docs.djangoproject.com/en/1.11/topics/http/urls/  
Examples:  
Function views  
 1. Add an import: from my\_app import views  
 2. Add a URL to urlpatterns: url(r'^$', views.home, name='home')  
Class-based views  
 1. Add an import: from other\_app.views import Home  
 2. Add a URL to urlpatterns: url(r'^$', Home.as\_view(), name='home')  
Including another URLconf  
 1. Import the include() function: from django.conf.urls import url, include  
 2. Add a URL to urlpatterns: url(r'^blog/', include('blog.urls'))  
"""***from** django.conf.urls **import** url  
**from** django.contrib **import** admin  
**from** testApp **import** views  
  
urlpatterns = [  
 url(**r'^admin/'**, admin.site.urls),  
 url(**r'^function\_view/'**, views.function\_view),  
 url(**r'^class\_based\_view/'**, views.ExampleClassBased.as\_view()),  
 url(**r'^example\_view/'**, views.ExampleView.as\_view()),  
 url(**r'^orders/'**, views.OrdersView.as\_view()),  
 url(**r'^order/(?P<id>\d+)'**, views.OrderView.as\_view(), name=**'order\_url'**),  
 url(**r'^zadanie/'**, views.ZadanieView.as\_view()),  
 url(**r'^zadanie1/(?P<id1>\d+)'**, views.Zadanie1View.as\_view(), name=**'zadanie1\_url'**),  
]

Результаты

   