



Лабораторные работы по курсу:

«Разработка Интернет Приложений»

## Шаблонизация

Исполнитель:

Студент группы РТ5-51

Макаров А.А.

Преподаватель:

Гапанюк Ю.Е.

« \_ » \_\_\_\_\_



### Задание и порядок выполнения

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 нажатию на элемент списка должна открываться страница информации об элементе
- Для верстки необходимо использовать Bootstrap

Листинг

lab5/

lab5/

\_\_init\_\_.py

settings.py

"""

Django settings for lab5 project.

Generated by 'django-admin startproject' using Django 1.11.6.

For more information on this file, see

<https://docs.djangoproject.com/en/1.11/topics/settings/>

For the full list of settings and their values, see

<https://docs.djangoproject.com/en/1.11/ref/settings/>

"""

import os

# Build paths inside the project like this: os.path.join(BASE\_DIR, ...)

BASE\_DIR = os.path.dirname(os.path.dirname(os.path.abspath(\_\_file\_\_)))

# Quick-start development settings - unsuitable for production

# See <https://docs.djangoproject.com/en/1.11/howto/deployment/checklist/>

# SECURITY WARNING: keep the secret key used in production secret!

SECRET\_KEY = 'rs^ygw3gh0&qfo\$9x)yeo\*(1do@0eo6)d8-ht^aff1s9^(\_p\$1'

# SECURITY WARNING: don't run with debug turned on in

production! DEBUG = True

ALLOWED\_HOSTS = []

# Application definition

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

```

'django.contrib.staticfiles',
'myApp.apps.MyappConfig',
]

MIDDLEWARE = [
'django.middleware.security.SecurityMiddleware',
'django.contrib.sessions.middleware.SessionMiddleware',
'django.middleware.common.CommonMiddleware',
'django.middleware.csrf.CsrfViewMiddleware',
'django.contrib.auth.middleware.AuthenticationMiddleware',
'django.contrib.messages.middleware.MessageMiddleware',
] 'django.middleware.clickjacking.XFrameOptionsMiddleware',

ROOT_URLCONF = 'lab5.urls'

TEMPLATES = [
{
'BACKEND': 'django.template.backends.django.DjangoTemplates',
'DIRS': [os.path.join(BASE_DIR, 'templates')]

'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',
},
},
]

WSGI_APPLICATION = 'lab5.wsgi.application'

# Database
# https://docs.djangoproject.com/en/1.11/ref/settings/#databases

DATABASES = {
'default': {
'ENGINE': 'django.db.backends.postgresql_psycopg2',
'NAME': 'django_db',
'USER': 'test_user',
'PASSWORD': 'qwerty',
'HOST': '127.0.0.1',
'PORT': '5432',
}
}

# Password validation
# https://docs.djangoproject.com/en/1.11/ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
{
}, 'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',

{
}, 'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
{
'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',

```

```
}  
{  
    'NAME':
```

```
'django.contrib.auth.password_validation.NumericPasswordValidator', ] },
```

```
# Internationalization
```

```
# https://docs.djangoproject.com/en/1.11/topics/i18n/
```

```
LANGUAGE_CODE = 'en-us'
```

```
TIME_ZONE = 'UTC'
```

```
USE_I18N = True
```

```
USE_L10N = True
```

```
USE_TZ = True
```

```
# Static files (CSS, JavaScript, Images)
```

```
# https://docs.djangoproject.com/en/1.11/howto/static-
```

```
files/ STATIC_URL = '/static/'
```

```
    urls.py
```

```
"""lab5 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 myApp.views import *
```

```
urlpatterns = (  
    url(r'^admin/', admin.site.urls),
```

```
    url(r'^$', IndexBaseClass.as_view(), name="index"),
```

```
    url(r'^goods/', get_goods, name="goods"),
```

```
    url(r'^product/(?P<product_id>[0-9]+)$', product, name='product_url'),
```

```
) url(r'^goods/add/', set_product, name="set_goods"),
```

```
    wsgi.py
```

```
"""
```

WSGI config for lab5 project.

It exposes the WSGI callable as a module-level variable named

`application`. For more information on this file, see

<https://docs.djangoproject.com/en/1.11/howto/deployment/wsgi/>

"""

import os

from django.core.wsgi import get\_wsgi\_application

os.environ.setdefault("DJANGO\_SETTINGS\_MODULE",

"lab5.settings") application = get\_wsgi\_application()

*MyApp/*

*migrations/*

*\_\_init\_\_.py*

o.db import migrations, models

class Migration(migrations.Migration):

initial = True

dependencies = [  
]

operations = [  
migrations.CreateModel(  
name='products',  
fields=[  
(  
'id', models.AutoField(auto\_created=True, primary\_key=True,  
serialize=False, verbose\_name='ID')),  
(  
'name', models.CharField(max\_length=100)),  
(  
'specifications', django.contrib.postgres.fields.jsonb.JSONField()),  
(  
'price', django.contrib.postgres.fields.ranges.FloatField()),  
],  
)],  
)

*MyApp/*

*static/*

*\_\_init\_\_.py*

*views.py*

import json

from django.shortcuts import render

from django.views import View

from .models import \*

# Create your views here.

with open("/home/catmen/Документы/lab2\_repo/lab5/myApp/static/json/goods.json") as  
data\_file:

goods\_list = json.load(data\_file)

class IndexBaseClass(View):

@staticmethod

def get(request):

context = {

} "page\_name": 'Магазин электроники'

return render(request, "index.html", context=context)

```

def product(request, product_id):
    return render(request, "product.html", context={"product" :
Products.objects.get(id=product_id)})

def get_goods(request):
    return render(request, "goods.html", context={"goods" : Products.objects.all()})

def set_product(request):
    pass

Templates/
    base.html

<!DOCTYPE html manifest="{{ STATIC_URL
}}/manifest"> <html lang="ru">
    <head>
        <meta charset="UTF-8"/>
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-
scale=1"> <meta name="description" content="">
        <meta name="author" content="">
        <title>{% block title %}Title{% endblock %}</title>
        <!-- Latest compiled and minified CSS -->
        <link rel="stylesheet"
href="//netdna.bootstrapcdn.com/bootstrap/3.3.2/css/bootstrap.min.css">
        <!-- Optional theme -->
        <link rel="stylesheet"
href="//netdna.bootstrapcdn.com/bootstrap/3.3.2/css/bootstrap-
theme.min.css"> </head>
    <body>
        <div class="navbar navbar-inverse navbar-fixed-top"
role="navigation"> <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse"
data-target=".navbar-collapse">
                    <span class="sr-only">Toggle navigation</span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                <a class="navbar-brand"
href="/">Eshop</a> </div>
                <div class="collapse navbar-collapse">
                    <ul class="nav navbar-nav">
                        <li class="/active"><a href="/">Главная</a></li>
                        <li><a href="/goods">Товары</a></li>
                    </ul>
                </div><!-- /.nav-collapse -->
            </div>
        </div>
        <div class="jumbotron">
            <div class="container">
                {% block body %}Нет данных{% endblock
%} </div><!-- /.container -->
        </div>
        <!-- jQuery first, then Tether, then Bootstrap JS. -->
        <script
src="//ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
        <!-- Latest compiled and minified JavaScript -->
        <script
src="//netdna.bootstrapcdn.com/bootstrap/3.3.2/js/bootstrap.min.js"></script>
    > </body>

```

```
</html>
```

goods.html

```
{% extends 'base.html' %}
{% block title %} Товары {% endblock %}
{% block body %}
    <div class="row">
        {% for product in goods %}
            {% include 'product_privew.html' %}
        {% empty %}
            В данный момент в магазине товары
            отсутствуют. {% endfor %}
    </div>
{% endblock %}
```

index.html

```
{% extends 'base.html' %}
{% block title %} {{ page_name }} {% endblock %}
{% block body %}
    <div class="starter-template">
        <h1>Магазин электронных товаров</h1>
        <p class="lead">В данном магазине очень много
        товаров!!!</p> </div>
{% endblock %}
```

product.html

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

{% block title %} {{ product.name }} {% endblock %}
{% block body %}
    <h1>{{ product.name }}</h1>
    <p class="center">
        {% for key, value in product.specifications.items %}
            {{ key }}: {{ value }}
        {% empty %}
            нет характеристик
        {% endfor %}
        {{ product.price }}
    </p>
{% endblock %}
```

product\_privew.html

```
<div class="col-md-4">

    <h2>{{ product.name }}</h2>
    <!--<p>{{ product.specifications
    }}</p>--> <p>{{ product.price }}
    руб.</p>
    <p><a class="btn btn-default" href="{% url 'product_url' product.id
%}" role="button">Посмотреть описание &raquo;</a></p>
</div>
```