Московский государственный университет им. Н. Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»

Лабораторная работа №5 по курсу:

«Разработка интернет-приложений»

Шаблонизация

Исполнитель:
Студентка группы ИУ5-51

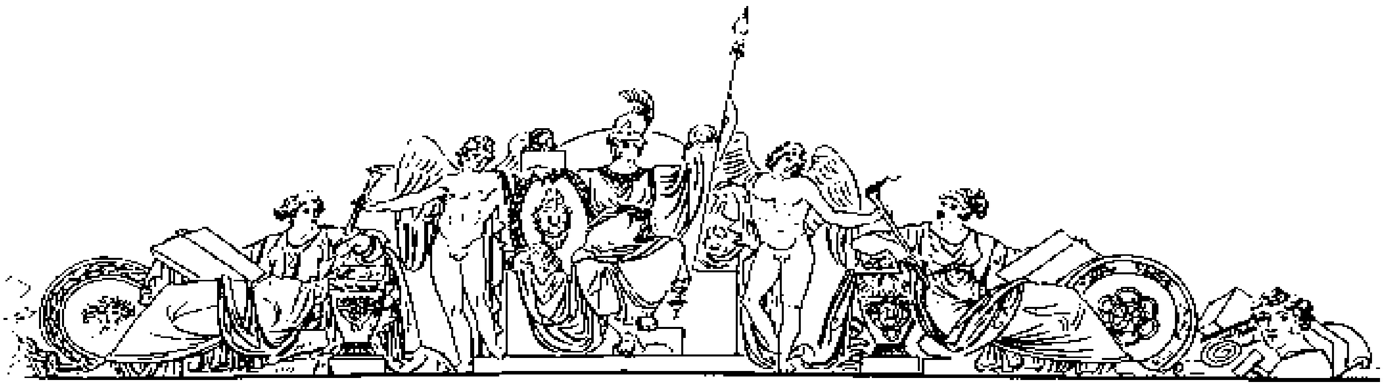
Коледова Н.А.

Преподаватель:

Гапанюк Ю. Е.

«___» _________________

____



Москва 2017г.

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 нажатии на элемент списка должна открываться страница информации об
элементе
● Для верстки необходимо использовать Bootstrap

Листинг:

5

manage.py

```
#!/usr/bin/env python
import os
import sys

if __name__ == "__main__":
    os.environ.setdefault("DJANGO_SETTINGS_MODULE", "lab5.settings")
    try:
        from django.core.management import execute_from_command_line
    except ImportError:
        # The above import may fail for some other reason. Ensure that the
        # issue is really that Django is missing to avoid masking other
        # exceptions on Python 2.
        try:
            import django
        except ImportError:
            raise ImportError(
                "Couldn't import Django. Are you sure it's installed and "
                "available on your PYTHONPATH environment variable? Did 
you "
                "forget to activate a virtual environment?"
            )
        raise
    execute_from_command_line(sys.argv)
```

5\\lab5\

__init__.py

settings.py

```python
"""
Django settings for lab5 project.

Generated by 'django-admin startproject' using Django 1.11.5.

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 = '!wkycvoqdz9p_i3)jn#7xmnv48=oh-91c_sd*!c1tah9*j=si*'

# 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',
    'polls'
]

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, 'polls')],
        '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.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}


# 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 = 'Europe/Moscow'

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 include, url
from django.contrib import admin

urlpatterns = [
    url(r'^', include('polls.urls')),
    url(r'^articles/', include('polls.urls')),
    url(r'^admin/', admin.site.urls)
]
```

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()
```

5\polls\

__init__.py

admin.py

```
from django.contrib import admin

# Register your models here.
```

apps.py

```
from django.apps import AppConfig


class PollsConfig(AppConfig):
    name = 'polls'
```

articles.html

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

{% block content %}

    {% for article in articles %}
        <div class="article-body">
                <h2><a href="/articles/{{article.id}}"
target="_blank">{{article.title}}</a></h2>
                <p>{{article.snippet}}</p>
        </div>
    {% endfor %}
{% endblock %}
```

base.html

```
<!DOCTYPE html>
<html lang="en">

<head>
    <title>Electronic University</title>

    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-
beta/css/bootstrap.min.css" integrity="sha384-
/Y6pD6FV/Vv2HJnA6t+vslU6fwYXjCFtcEpHbNJ0lyAFsXTsjBbfaDjzALeQsN6M"
        crossorigin="anonymous">
    {% load static %}
    <link rel="stylesheet" type="text/css" href="{% static 'css/style.css'
%}">
</head>

<body>
    <div class="container-fluid menu">
        <div class="row">
            <h1 class="col-md-5"> Электронный университет </h1>
            <div class="col-md-5"></div>
            <div class="col-md-2 menu-button">
                <a href="/articles/"><p>ARTICLES</p></a>
            </div>
            <div class="col-md-2 menu-button">
                <a href="#"><p>AUTHORS</p></a>
            </div>
            </div>
    </div>

    <div class="content">
        {% block content %} {% endblock %}
    </div>


    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
        crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.11.0/umd/popper.mi
n.js" integrity="sha384-
b/U6ypiBEHpOf/4+1nzFpr53nxSS+GLCkfwBdFNTxtclqqenISfwAzpKaMNFNmj4"
        crossorigin="anonymous"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-
beta/js/bootstrap.min.js" integrity="sha384-
h0AbiXch4ZDo7tp9hKZ4TsHbi047NrKGLO3SEJAg45jXxnGIfYzk4Si90RDIqNm1"
        crossorigin="anonymous"></script>
</body>

</html>
```

models.py

```
from django.db import models


# Create your models here.


class Article(models.Model):
    title = models.CharField(max_length=100)
    author = models.ForeignKey("Author")
    content = models.TextField()
    snippet = models.CharField(max_length=500)
    pubdate = models.DateTimeField()
    likes = models.PositiveIntegerField()

    def __unicode__(self):
        return self.title
```

```python
class Author(models.Model):
    firstName = models.CharField(max_length=50)
    lastName = models.CharField(max_length=50)
    about = models.TextField()
    email = models.EmailField()
    phone_number = models.CharField(max_length=12)
    pass
```

single.html

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

{% block content %}

        <div class="article-body">
                <h2>{{article.title}}</h2>
                <p>{{article.content}}</p>
        </div>

{% endblock %}
```

tests.py

```python
from django.test import TestCase

# Create your tests here.
```

urls.py

```python
from django.conf.urls import url

from . import views

urlpatterns = [
    url(r'^$', views.index, name='index'),
    url(r'^(?P<id>(\d+))$', views.article, name='article')
]
```

views.py

```python
from django.shortcuts import render
from django.shortcuts import render_to_response
from django.template.loader import render_to_string
from django.http import HttpResponse
from django.template import RequestContext, loader
from .models import Article

# Create your views here..


def index(request):
    articles = Article.objects.all()
    content = {
        'articles': articles
    }
    return render(request, 'articles.html', content)


def article(request, id):
    art = Article.objects.get(id=int(id))
    content = {
        'article' : art
    }
    return render(request, 'single.html', content)
    pass
```

5\polls\static\css\

style.css

```
body{
    background-color: lightsteelblue;
}

.content{
    display: flex;
    flex-direction: column;
}

.menu{
    background-color: dodgerblue;
    -webkit-box-shadow: 0px 2px 21px 0px rgba(0,0,0,0.75);
    -moz-box-shadow: 0px 2px 21px 0px rgba(0,0,0,0.75);
    box-shadow: 0px 2px 21px 0px rgba(0,0,0,0.75);
    position: relative;
    float: top;
}

.menu h1{
    margin: 10px;
}

.menu-button{
    display: block;
    color: #000000;
}

.menu-button p{
    margin: 0;
    text-align: center;
    align-self: center;
}

.menu-button a{
    display: flex;
    text-decoration: none;
    flex-direction: row;
    justify-content: center;
    width: 100%;
    height: 100%;
    color: #ffffff;
    transition: box-shadow 0.3s ease-in-out;
}

.menu-button a:hover{
    -webkit-box-shadow: 0px 1px 10px 0px rgba(0,0,0,0.75);
    -moz-box-shadow: 0px 1px 10px 0px rgba(0,0,0,0.75);
    box-shadow: 0px 1px 10px 0px rgba(0,5,0,0.75);
}

.small-margin{
    margin: 10px;
    display: inline-block;
}

.article-body{
    align-self: center;
    width: 70%;
    background-color: #ffff;
    -webkit-box-shadow: 0px 2px 21px 0px rgba(0,0,0,0.75);
    -moz-box-shadow: 0px 2px 21px 0px rgba(0,0,0,0.75);
    box-shadow: 0px 2px 21px 0px rgba(0,0,0,0.75);
    margin: 10px;
    padding: 20px;
    border-radius: 3px;
}

.article-body a{
    text-decoration: none;
    color: #000;
}
```

5\polls\migrations

__init__.py

0001_initial.py

```
# -*- coding: utf-8 -*-
# Generated by Django 1.11.5 on 2017-09-20 08:56
from __future__ import unicode_literals

from django.db import migrations, models
import django.db.models.deletion


class Migration(migrations.Migration):

    initial = True

    dependencies = [
    ]

    operations = [
        migrations.CreateModel(
            name='Article',
            fields=[
                ('id', models.AutoField(auto_created=True,
primary_key=True, serialize=False, verbose_name='ID')),
                ('title', models.CharField(max_length=100)),
                ('content', models.TextField()),
                ('pubdate', models.DateTimeField()),
                ('likes', models.PositiveIntegerField()),
            ],
        ),
        migrations.CreateModel(
            name='Author',
            fields=[
                ('id', models.AutoField(auto_created=True,
primary_key=True, serialize=False, verbose_name='ID')),
                ('firstName', models.CharField(max_length=50)),
                ('lastName', models.CharField(max_length=50)),
                ('about', models.TextField()),
                ('email', models.EmailField(max_length=254)),
                ('phone_number', models.CharField(max_length=12)),
            ],
        ),
        migrations.AddField(
            model_name='article',
            name='author',

field=models.ForeignKey(on_delete=django.db.models.deletion.CASCADE,
to='polls.Author'),
        ),
    ]
```

0002_article_snipped.py

```
# -*- coding: utf-8 -*-
# Generated by Django 1.11.5 on 2017-09-21 12:58
from __future__ import unicode_literals

from django.db import migrations, models


class Migration(migrations.Migration):

    dependencies = [
        ('polls', '0001_initial'),
    ]

    operations = [
        migrations.AddField(
            model_name='article',
            name='snippet',
            field=models.CharField(default='Snippet', max_length=500),
            preserve_default=False,
        ),
    ]
```