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

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»



Лабораторные работы по курсу:

**«Разработка Интернет Приложений»**

**ЛР6. Работа с СУБД**

Исполнитель:

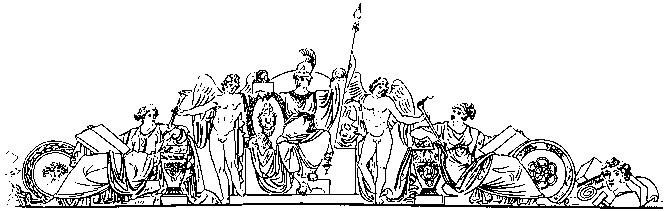
Студентка группы РТ5-51

Карасева А. Д

Преподаватель:

Гапанюк Ю. Е.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



Москва 2017 г.

Задание и порядок выполнения

В этой лабораторной работе необходимо познакомиться с популярной СУБД MySQL, создать свою базу данных. Также нужно будет дополнить свои классы предметной области, связав их с созданной базой. После этого потребуется создать свои модели с помощью Django ORM, отобразить объекты из БД с помощью этих моделей и ClassBasedViews.

Исходный код:

**settings.py**

"""

Django settings for lab6 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 = '49!#b03k-$776(y!yk)1i==$de40779llydo(okzdn3($ca!c7'

# 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',

'my\_app.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 = 'lab6.wsgi.application'

# Database

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

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': 'tutoring\_django',

'USER': 'dbuser',

'PASSWORD': '123',

'HOST': 'localhost',

'PORT': 3306, # Стандартный порт MySQL

'OPTIONS': {'charset': 'utf8'},

'TEST\_CHARSET': 'utf8',

}

}

# 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/'

# kuku/urls.py

"""kuku 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 my\_app import views  
from django.conf import settings  
from django.conf.urls.static import static  
  
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/', views.Example\_template, name='example\_url'),  
 url(r'^variable/', views.Example\_template\_Variable, name='variable\_url'),  
 url(r'^groups/', views.GroupsView.as\_view(), name='groups'),  
 url(r'^group/(?P<id>\d+)', views.GroupView.as\_view(), name='group'),  
]  
if settings.DEBUG:  
 urlpatterns += static(settings.STATIC\_URL, document\_root=settings.STATIC\_URL)

# views.py

**from** django.shortcuts **import** render  
**from** django.http **import** HttpResponse  
**from** django.views **import** View  
  
*# Create your views here.***def** function\_view(request):  
 **return** HttpResponse(**'Response :)'**)  
  
  
**class** ExampleClassBased(View):  
 **def** get(self, request):  
 **return** HttpResponse(**'response from class based view'**)  
  
  
**def** Example\_template(request):  
 print(**'Hello'**)  
 **return** render(request, **'Example.html'**)  
  
**def** Example\_template\_Variable(request):  
 **return** render(request, **'variable.html'**, {**'my\_variable'**: **'Текст'**})  
  
**class** GroupsView(View):  
 **def** get(self, request):  
 data = {  
 **'groups'**: [  
 {**'title'**: **'ИУ5-51'**, **'id'** : **'1'**},  
 {**'title'**: **'ИУ5-52'**, **'id'** : **'2'**}  
 ]  
 }  
 **return** render(request, **'groups.html'**, data)  
  
  
**class** GroupView(View):  
 **def** get(self, request, id):  
 data = {  
 **'group'**: {  
 **'id'**: id  
 }  
 }  
 **return** render(request, **'group.html'**, data)

# base.html

<!DOCTYPE **html**>  
<**html lang="en"**>  
<**head**>  
 <**meta charset="UTF-8"**>  
 <**title**>{% **block title** %}{% **endblock** %}</**title**>  
 {% **load static** %}  
 <**link rel="stylesheet" href="**{% **static 'Bootstrap/css/bootstrap.css'** %}**"**>  
</**head**>  
<**body**>  
 {% **block body** %}{% **endblock** %}  
 {% **load static** %}  
 <**link rel="stylesheet" type="text/css" href="**{% **static 'aaa.css'** %}**"**>  
 <**img src="**{% **static 'images/flowers.jpg'** %}**" alt='bs'**>  
</**body**>  
</**html**>

# groups.html

{% **extends 'base.html'** %}  
  
{% **block title** %}  
 <**h1**>Группы</**h1**>  
{% **endblock** %}  
  
{% **block body** %}  
 <**ul**>  
 {%**for group in groups**%}  
 <**li**><**a href="**{% **url 'group' id**=**group**.**id** %}**"**>{{ **group**.**title** }}</**a**> </**li**>  
 {% **empty** %}  
 <**li**>Пустой список</**li**>  
 {% **endfor** %}  
 </**ul**>  
{% **endblock** %}

# group.html

{% **extends 'base.html'** %}  
  
{% **block title** %}Группа {{ **group**.**id** }}{% **endblock** %}  
  
{% **block body** %}Страница группы {% **endblock** %}

# Example.html

{% **extends 'base.html'** %}  
  
{% **block title** %}Extends{% **endblock** %}  
{% **block body** %}Extended block{% **endblock** %}

# variable.html

{% **extends 'base.html'** %}  
  
{% **block title** %}Переменная{% **endblock** %}  
  
{% **block body** %}  
 <**ul**>  
 {% **if my\_variable** == 2 %}  
 <**li**> Двойка </**li**>  
 {% **else** %}  
 {{ **my\_variable** }}  
 {% **endif** %}  
 </**ul**>  
{% **endblock** %}