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»

Работа с СУБД

Исполнитель:
Студент группы ИУ5-52

Миядин А.А.

Преподаватель:

Гапанюк Ю.Е,

«___» _____

Москва 2017г.

Задание и порядок выполнения

В этой лабораторной работе вы познакомитесь с популярной СУБД MySQL, создадите свою базу данных. Также вам нужно будет дополнить свои классы предметной области, связав их с созданной базой. После этого вы создадите свои модели с помощью Django ORM, отобразите объекты из БД с помощью этих моделей и ClassBasedViews.

Для сдачи вы должны иметь:

1. Скрипт с подключением к БД и несколькими запросами.
2. Набор классов вашей предметной области с привязкой к СУБД (класс должен уметь хотя бы получать нужные записи из БД и преобразовывать их в объекты этого класса)
3. Модели вашей предметной области
4. View для отображения списка ваших сущностей

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
0001_initial.py

```
# -*- coding: utf-8 -*-
# Generated by Django 1.11.6 on 2017-11-10 13:13
from __future__ import unicode_literals

import django.contrib.postgres.fields.jsonb
import django.contrib.postgres.fields.ranges
from django.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.FloatRangeField()),
            ],
        ),
    ]
```

MyApp/
static/
__init__.py
models.py

```
from django.contrib.postgres import fields
from django.db import models

# Create your models here.

class Products(models.Model):
    name = models.CharField(max_length=100)
    specifications = fields.JSONField()
    price = fields.FloatRangeField()
```

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
```

Скриншоты