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:

Студент группы РТ5-51

Макаров А.В.

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 test5_full project.

Generated by 'django-admin startproject' using Django 1.11.7.

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 = '+glayd^@rojiipq)-qy33nk8i+2un!kj1oug(r9178(f+uw6!4'

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = ['192.168.1.10','127.0.0.1']


# 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 = 'test5_full.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 = 'test5_full.wsgi.application'


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

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'userbronkoncert',
        'USER': 'root',
        'PASSWORD': '1111',
        'HOST': 'localhost',
        'PORT': 3306, #Стандартный порт Mysql
        'OPTIONS': {
          'autocommit': True,
        },
        '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.UserAttributeSimilarityValidato
r',
    },
    {
        '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/'
```

```
from django.conf.urls import url
from Myapp.views import *

urlpatterns = [
    url(r'^mine/$', MyView.as_view(), name='my-view'),
    url(r'^users/$', UserList.as_view(), name='my-view2'),
    url(r'^usersList/$', UserList.as_view()),
]
```

connection.py

```
import MySQLdb

class Connection:
    def __init__(self, user, password, db, host='localhost'):
        self.host= host
        self.user= user
        self.password=password
        self.db=db
        self._connection = None

    @property
    def connection(self):
        return self._connection

    def __enter__ (self):
        self.connect()

    def __exit__ (self, exc_type, exc_val, exc_tb):
        self.disconnect()

    def connect(self):
        #открытие соединения
        if not self._connection:
            self._connection= MySQLdb.connect(
                host=self.host,
                user=self.user,
                passwd=self.password,
                db=self.db
            )

    def disconnect(self):
        #закрытие соединения
        if self._connection:
            self._connection.close()


class User:
    def __init__(self, db_connection, id, name, idconcert):
        self.db_connection = db_connection.connection
        self.id = id
        self.name = name
        self.idconcert = idconcert

    def save(self):
        c = self.db_connection.cursor()
        #c.execute("insert into user (id, name, idconcert) values (%s, %s,
%s);",
         #          (self.id, self.name, self.idconcert))
        c.execute("update user set name = %s, idconcert = %s Where id =
%s;",
                  (self.name, self.idconcert, self.id))
        self.db_connection.commit()
        c.close()

class SelAll:
    def __init__(self, db_connection):
        self.db_connection = db_connection.connection
    def save(self):
        c = self.db_connection.cursor()
        c.execute("Select * from USER ")
        entries = c.fetchall()
        c.close()
        #for e in entries:
        #    print(e)
        return('<br>'.join(map(str, entries)))

con = Connection("root", "1111", "userbronkoncert")
with con:
    user = User(con, '10', 's3rpyn', '3')
    user.save()


con = Connection("root", "1111", "userbronkoncert")
with con:
    user = SelAll(con)
    user.save()
```

models.py

```
from django.db import models


class User(models.Model):
    name = models.CharField(max_length=30)
    idconcert = models.harField(max_length=3)

    class Meta:
        ordering = ["-name"]


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

views.py

```
from django.http import HttpResponse
from django.views.generic import View
from Myapp.Connection import *
from django.views.generic import ListView

class UserList(ListView):
    model = User

class UserList(View):
    def get(self, request, *args, **kwargs):
        con = Connection("root", "1111", "userbronkoncert")
        with con:
            user = SelAll(con)
            f=user.save()
        return HttpResponse(f)
```