Московский Государственный Технический Университет им. Н. Э. Баумана
Факультет «Информатика и системы управления»

Отчет по лабораторной работе № 6
Курс «Разработка Интернет-приложений»

Выполнил:

студент группы ИУ5-51 Марков А.Д.

Москва, МГТУ – 2016 г.

Описание задания лабораторной работы

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 программы

Settings.py

```
"""
Django settings for lab6 project.
Generated by 'django-admin startproject' using Django 1.10.3.
For more information on this file, see
https://docs.djangoproject.com/en/1.10/topics/settings/
For the full list of settings and their values, see
https://docs.djangoproject.com/en/1.10/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/1.10/howto/deployment/checklist/
# SECURITY WARNING: keep the secret key used in production secret!
SECRET_KEY = '&14-rmry%jq3w%d)78)rud8uiki=l8m=ycn%c)1k0dl26d=ncd'
# 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',
 'users',
]
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 = 'lab6.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/1.10/ref/settings/#databases
DATABASES = {
 'default': {
 'ENGINE': 'django.db.backends.mysql',
 'NAME': 'first_db1',
 'USER': 'admin',
 'PASSWORD': '54321',
 'HOST': 'localhost',
 'PORT': '3306',
 'OPTIONS': {'charset': 'utf8'},
 'TEST_CHARSET': 'utf8',
 }
}
# Password validation
# https://docs.djangoproject.com/en/1.10/ref/settings/#auth-passwordvalidators
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/1.10/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/1.10/howto/static-files/
STATIC_URL = '/static/'
```

Models.py

```
from django.db import models
```

```
class User(models.Model):
    idUser = models.IntegerField(unique=True)
    email = models.EmailField(max_length=255, unique=True, null=False)
    bill = models.IntegerField()
    first_name = models.CharField(max_length=255, null=False)
    last_name = models.CharField(max_length=255, null=False)


class Comment(models.Model):
    idComment = models.IntegerField(unique=True)
    Course = models.TextField(max_length=255, null=False)
    Text = models.TextField(max_length=255, null=False)


class Course(models.Model):
    idCourse = models.IntegerField(unique=True)
    Name = models.TextField(max_length=255, null=False)
    Duration = models.TextField(max_length=255, null=False)
    Teacher = models.TextField(max_length=255, null=False)
    Value = models.TextField(max_length=255, null=False)
```

views.py

```
from django.shortcuts import render
from users.Models import Course,Comment
from django.views.generic import TemplateView
from django.views import View


def index(request):
    return render(request, 'index.html')


class Courses(TemplateView):
    template_name = 'index1.html'

    def get_context_data(self, **kwargs):
        Courses = Course.objects.all()
        context = dict(Courses=Courses)
        return context


class Comments(TemplateView):
    template_name = 'index2.html'

    def get_context_data(self, **kwargs):
        Comments = Comment.objects.all()
        context = dict(Comments=Comments)
        return context
```

urls.py

```
"""lab6 URL Configuration
The `urlpatterns` list routes URLs to views. For more information please see:
 https://docs.djangoproject.com/en/1.10/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'^$', Home.as_view(), name='home')
Including another URLconf
 1. Import the include() function: from django.conf.urls import url,
include
 2. Add a URL to urlpatterns: url(r'^blog/', include('blog.urls'))
"""
from django.conf.urls import url
from django.contrib import admin
from users.views import Courses,index,Comments
urlpatterns = [
 url(r'^admin/', admin.site.urls),
 url(r'^$', index),
 url(r'^course/$', Courses.as_view()),
 url(r'^course/comment/$', Comments.as_view()),
]
```

Admin.py

```
from django.contrib import admin
from users.Models import *
admin.site.register(User)
admin.site.register(Comment)
admin.site.register(Course)
```

apps.py

```
from django.apps import AppConfig
  class
AdminConfig(AppConfig):
    name = 'users'
```

FirstScript.py

```
import pymysql

pymysql.install_as_MySQLdb()
db = pymysql.connect(
    host="localhost",
    user="admin",
    passwd="54321",
    db="first_db1",
    charset="utf8"
)
cursor = db.cursor()
cursor.execute("""INSERT INTO Books(name, description)VALUES(%s, %s),(%s,%s)""",
               ("One", "Two",
                "Three", "Four")
               )
db.commit()
# cursor.execute("DELETE FROM Books WHERE id>1")
# db.commit()
cursor.execute("SELECT * FROM Books;")
books = cursor.fetchall()
for book in books:
    print(book)
cursor.close()
db.close()
```

connections.py

```
try:
    import pymysql
pymysql.install_as_MySQLdb() except
ImportError:
    pass
```

SecondScript.py

```
import pymysql as MySQLdb


class Connection:
    def __init__(self, user, password, db, host='localhost'):
        self.user = user
        self.host = host
        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):
        if not self._connection:
            self._connection = MySQLdb.connect(
                host=self.host,
                user=self.user,
                passwd=self.password,
                db=self.db
            )

    def disconnect(self):
        if self._connection:
            self._connection.close()


class Book:
    def __init__(self, db_connection, name, description):
        self.db_connection = db_connection.connection
        self.name = name
        self.description = description

    def save(self):
        c = self.db_connection.cursor()
        c.execute("INSERT INTO books (name, description) VALUES (%s, %s);",
                  (self.name, self.description))
        self.db_connection.commit()
        c.close()


con = Connection("admin", "54321", "first_db1")
with con:
    book = Book(con, 'New book', 'Description new book')
    book.save()
```

wsgi.py

```
"""
WSGI config for lab6 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/1.10/howto/deployment/wsgi/
"""
import os

from django.core.wsgi import get_wsgi_application
 os.environ.setdefault("DJANGO_SETTINGS_MODULE", "lab6-
master.settings")

application = get_wsgi_application()
```

0001_inital.py

```
# -*- coding: utf-8 -*-
# Generated by Django 1.10.4 on 2016-12-29 10:45
from __future__ import unicode_literals

from django.db import migrations, models


class Migration(migrations.Migration):

    initial = True

    dependencies = [
    ]

    operations = [
        migrations.CreateModel(
            name='Comment',
            fields=[
                ('id', models.AutoField(auto_created=True, primary_key=True,
serialize=False, verbose_name='ID')),
                ('idComment', models.IntegerField(unique=True)),
                ('Course', models.TextField(max_length=255)),
                ('Text', models.TextField(max_length=255)),
            ],
        ),
        migrations.CreateModel(
            name='Course',
            fields=[
                ('id', models.AutoField(auto_created=True, primary_key=True,
serialize=False, verbose_name='ID')),
                ('idCourse', models.IntegerField(unique=True)),
                ('Name', models.TextField(max_length=255)),
                ('Duration', models.TextField(max_length=255)),
                ('Teacher', models.TextField(max_length=255)),
                ('Value', models.TextField(max_length=255)),
            ],
        ),
        migrations.CreateModel(
            name='User',
            fields=[
                ('id', models.AutoField(auto_created=True, primary_key=True,
serialize=False, verbose_name='ID')),
                ('idUser', models.IntegerField(unique=True)),
                ('email', models.EmailField(max_length=255, unique=True)),
                ('bill', models.IntegerField()),
                ('first_name', models.CharField(max_length=255)),
                ('last_name', models.CharField(max_length=255)),
            ],
        ),
    ]
```

Manage.py

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

if __name__ == "__main__":
    os.environ.setdefault("DJANGO_SETTINGS_MODULE", "lab6.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)
```

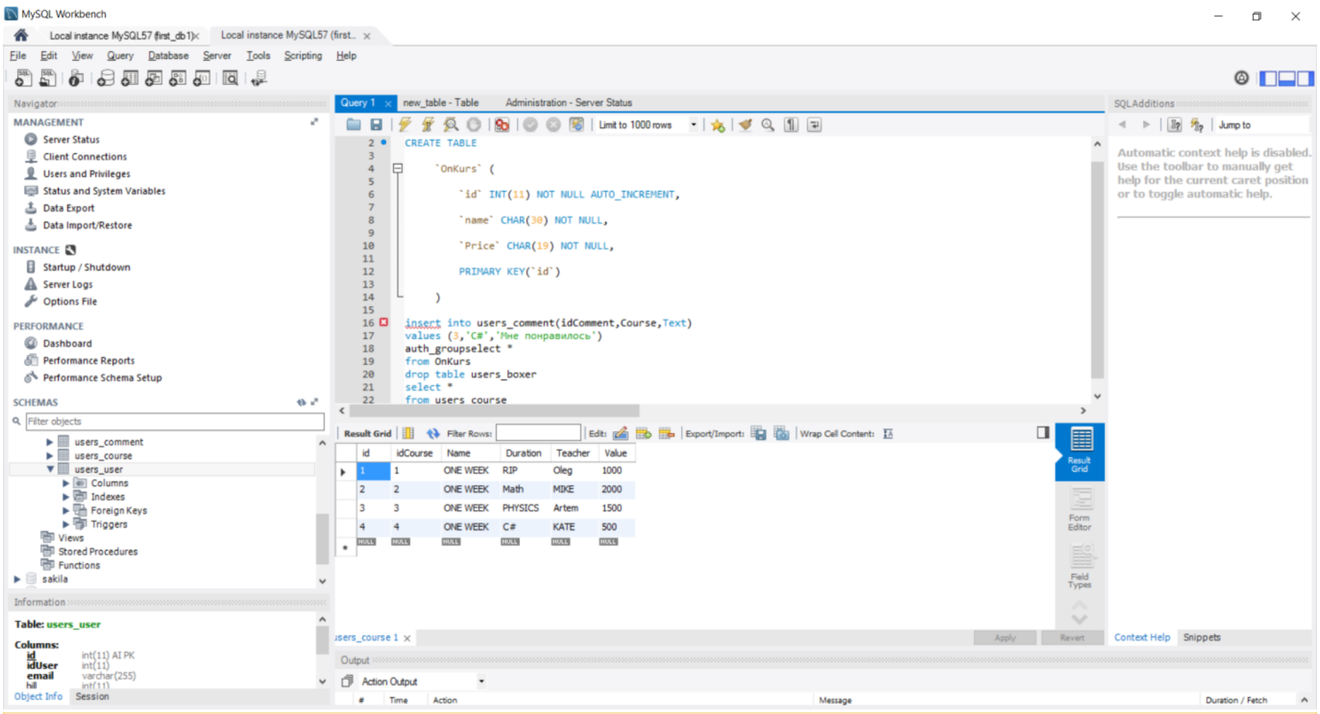
Index1.html

```
{% load static %}
<html>
<body>

 {% for Course in Courses %}
 <p>Название курса-{{ Course.Duration }}<br>
    Длительность курса-{{ Course.Name }}<br>
    Преподаватель-{{ Course.Teacher }}<br>
    Стоимость курса-{{ Course.Value }}<br>
 </p>
 <hr/>
 {% endfor %}
<html>
 <body>
  <form action="http://127.0.0.1:8000/course/comment">
   <button>
       <img src="{% static 'css/feedback.jpg' %}" width="25" height="25" alt=""
style="">
       Comments
   </button></p>
  </form>
 </body>
</html>
</body>
</html>
```

Index2.html

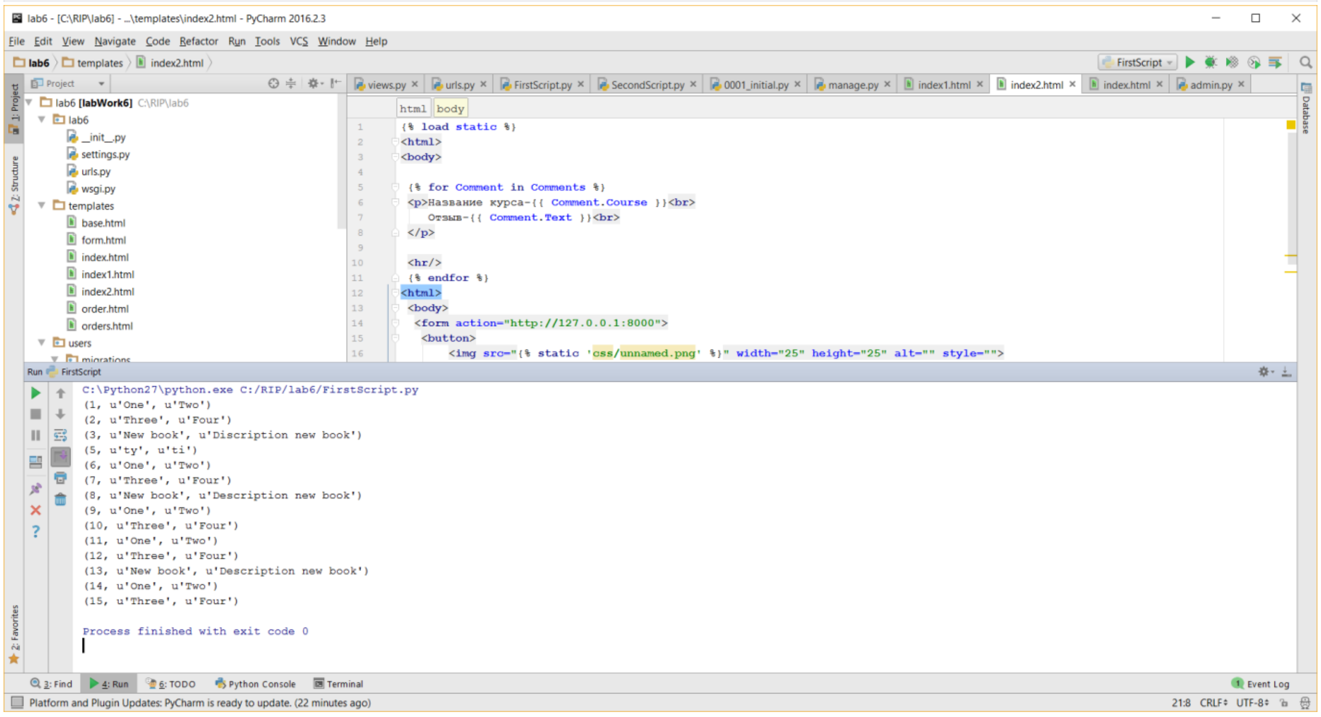
```
{% load static %}
<html>
<body>

 {% for Comment in Comments %}
 <p>Название курса-{{ Comment.Course }}<br>
    Отзыв-{{ Comment.Text }}<br>
 </p>

 <hr/>
 {% endfor %}
<html>
 <body>
  <form action="http://127.0.0.1:8000">
   <button>
       <img src="{% static 'css/unnamed.png' %}" width="25" height="25" alt=""
style="">
       Главная страница
   </button></p>
  </form>
 </body>
</html>
</body>
</html>
```

Результаты работы программы:



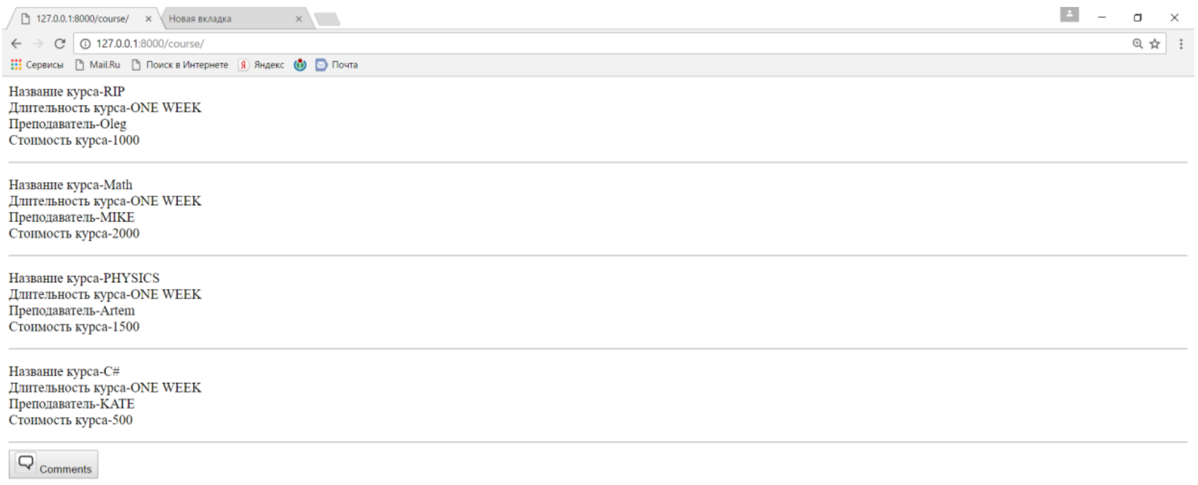
```sql
CREATE TABLE

    `OnKurs` (

        `id` INT(11) NOT NULL AUTO_INCREMENT,

        `name` CHAR(30) NOT NULL,

        `Price` CHAR(19) NOT NULL,

        PRIMARY KEY(`id`)

    )

insert into users_comment(idComment,Course,Text)
values (3,'C#','Мне понравилось')
auth_groupselect *
from OnKurs
drop table users_boxer
select *
from users_course
```

id	idCourse	Name	Duration	Teacher	Value
1	1	ONE WEEK	RIP	Oleg	1000
2	2	ONE WEEK	Math	MIKE	2000
3	3	ONE WEEK	PHYSICS	Artem	1500
4	4	ONE WEEK	C#	KATE	500
NULL	NULL	NULL	NULL	NULL	NULL



```html
{% load static %}
<html>
<body>

 {% for Comment in Comments %}
 <p>Название курса-{{ Comment.Course }}<br>
    Отзыв-{{ Comment.Text }}<br>
 </p>

 <hr/>
 {% endfor %}
<html>
 <body>
  <form action="http://127.0.0.1:8000">
   <button>
       <img src="{% static 'css/unnamed.png' %}" width="25" height="25" alt="" style="">
```

```
C:\Python27\python.exe C:/RIP/lab6/FirstScript.py
(1, u'One', u'Two')
(2, u'Three', u'Four')
(3, u'New book', u'Discription new book')
(5, u'ty', u'ti')
(6, u'One', u'Two')
(7, u'Three', u'Four')
(8, u'New book', u'Description new book')
(9, u'One', u'Two')
(10, u'Three', u'Four')
(11, u'One', u'Two')
(12, u'Three', u'Four')
(13, u'New book', u'Description new book')
(14, u'One', u'Two')
(15, u'Three', u'Four')

Process finished with exit code 0
```

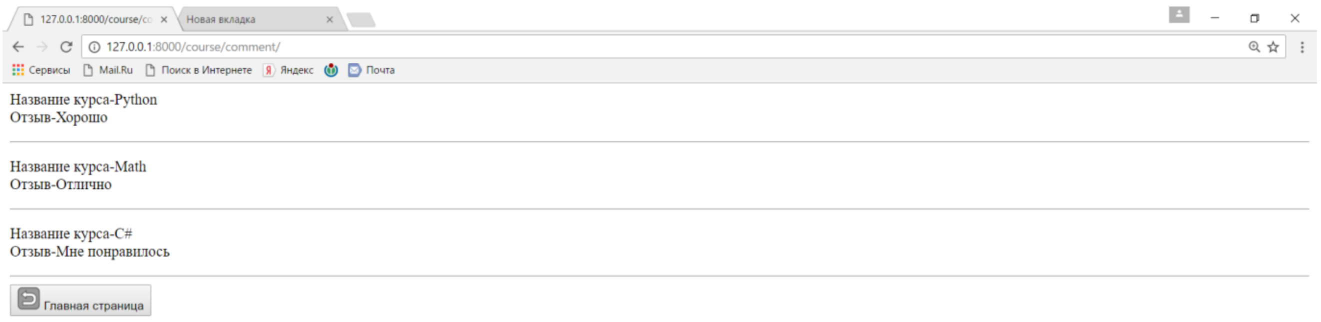


Название курса-RIP
Длительность курса-ONE WEEK
Преподаватель-Oleg
Стоимость курса-1000

Название курса-Math
Длительность курса-ONE WEEK
Преподаватель-MIKE
Стоимость курса-2000

Название курса-PHYSICS
Длительность курса-ONE WEEK
Преподаватель-Artem
Стоимость курса-1500

Название курса-C#
Длительность курса-ONE WEEK
Преподаватель-KATE
Стоимость курса-500



Название курса-Python
Отзыв-Хорошо

Название курса-Math
Отзыв-Отлично

Название курса-C#
Отзыв-Мне понравилось