

Московский государственный технический университет им. Н.Э. Баумана
Факультет «Информатика и системы управления»
Кафедра «Автоматизированные системы обработки информации и управления»



Гапанюк Ю.Е.

**Отчет по Лабораторной работе №6 «Работа с СУБД»
По курсу
“Разработка интернет-приложений”**

Выполнил:
Постникова М.А.
Студент группы ИУ5-54

Москва 2017

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

Исходный код

connection.py

```
import MySQLdb
```

```
class Connection:
```

```
    def __init__(self, host, db_name, db_user, db_pass, db_charset):
```

```
        self.host = host
```

```
        self.db_name = db_name
```

```
        self._connection = None
```

```
        self.db_user = db_user
```

```
        self.db_pass = db_pass
```

```
        self.db_charset = db_charset
```

```
@property
```

```
    def connection(self):
```

```
        return self._connection
```

```
    def __enter__(self):
```

```
        return 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,
```

```
                db=self.db_name,
```

```

        user=self.db_user,
        passwd=self.db_pass,
        charset=self.db_charset,
    )

    return self.connection

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

```

index.html

```

{% extends "base.html" %}
{% block content %}

<div class="container wrap">
    <div class="row">
        <div class="col-md-9 col-sm-6 ">
            {% block question %}
                {% for question in questions %}
                    {% include "question.html" %}
                {% endfor %}
            {% endblock %}
        </div>
        <div class="col-md-3 col-sm-6 right-col">
        </div>
    </div>
<br><br>
</div>

<!-- /.container -->

<!---->
    {% endblock %}

```

ask/urls.py

```

from django.conf.urls import url
from . import views

urlpatterns = [

```

```

url(r'^$', views.index, name='index'),
url(r'^questions/$', views.index, name='index'),
url(r'^questions/(?P<id>(\d+))$', views.question, name='question'),
url(r'^singup/$', views.singup, name='singup'),
url(r'^login/$', views.singin, name='login'),
url(r'^logout/$', views.log_out, name="logout")
]

```

ask/views.py

```

# coding=utf-8
from django.shortcuts import render
from .models import Question
from django.shortcuts import render
from django.shortcuts import render_to_response
from django.template.loader import render_to_string
from django.http import HttpResponseRedirect
from django.http import HttpResponseRedirect
from django.template import RequestContext, loader
from django.contrib.auth.models import User
from django.contrib.auth import authenticate, login, logout
from django.contrib.auth.decorators import login_required
from django.core.paginator import Paginator
from django.http.response import Http404, JsonResponse

# django imports
from django.shortcuts import render, redirect
from .forms import RegisterForm, LoginForm

# Create your views here..
def index(request):
    questions = Question.objects.all()
    content = {
        'questions': questions
    }

```

```

for q in questions:
    print(q)
    return render(request, 'index.html', content)
def question(request, id):
    q = Question.objects.get(id=int(id))
    content = {
        'question' : q
    }
    return render(request, 'questions.html', content)
index.html
{% extends "index.html" %}
{% block question %}
    <div class="question row">
        <div class="col-md-2">
            <a href="#"></a>
            <a href="#" class="like"></a><a href="#" class="likes">{{
question.rating }}</a>
            <a href="#" class="dislike"></a><a href="#" class="likes">0</a>
        </div>
        <div class="col-md-10">
            <div class="q-title"> <a href="#" target="_blank">{{
question.title }}</a></div>
            <div class="q-text">{{ question.text }}
                <br>
                <div class="tags">
                    <a href="#">answer(3)</a>
                    Tags: <a href="#">bender</a> <a href="#">disney</a>
                </div>
            </div>
        </div>
    </div>
{% endblock %}

```

question.html

```
<div class="question row">
  <div class="col-md-2">
    <a href="#"></a>
    <a href="#" class="like"></a><a href="#" class="likes">{{
question.rating }}</a>
    <a href="#" class="dislike"></a><a href="#" class="likes">0</a>
  </div>
  <div class="col-md-10">
    <div class="q-title"> <a href="/questions/{{question.id}}"
target="_blank">{{ question.title }}</a></div>
    <div class="q-text">{{ question.snippet }}
    <a class="learn-more" href="/questions/{{question.id}}"
target="_blank">?????? ??????...</a>
    <br>
    <div class="tags">
      <a href="#">answer(3)</a>
      Tags: <a href="#">bender</a> <a href="#">disney</a>
    </div>
  </div>
</div>
```

ask/models.py

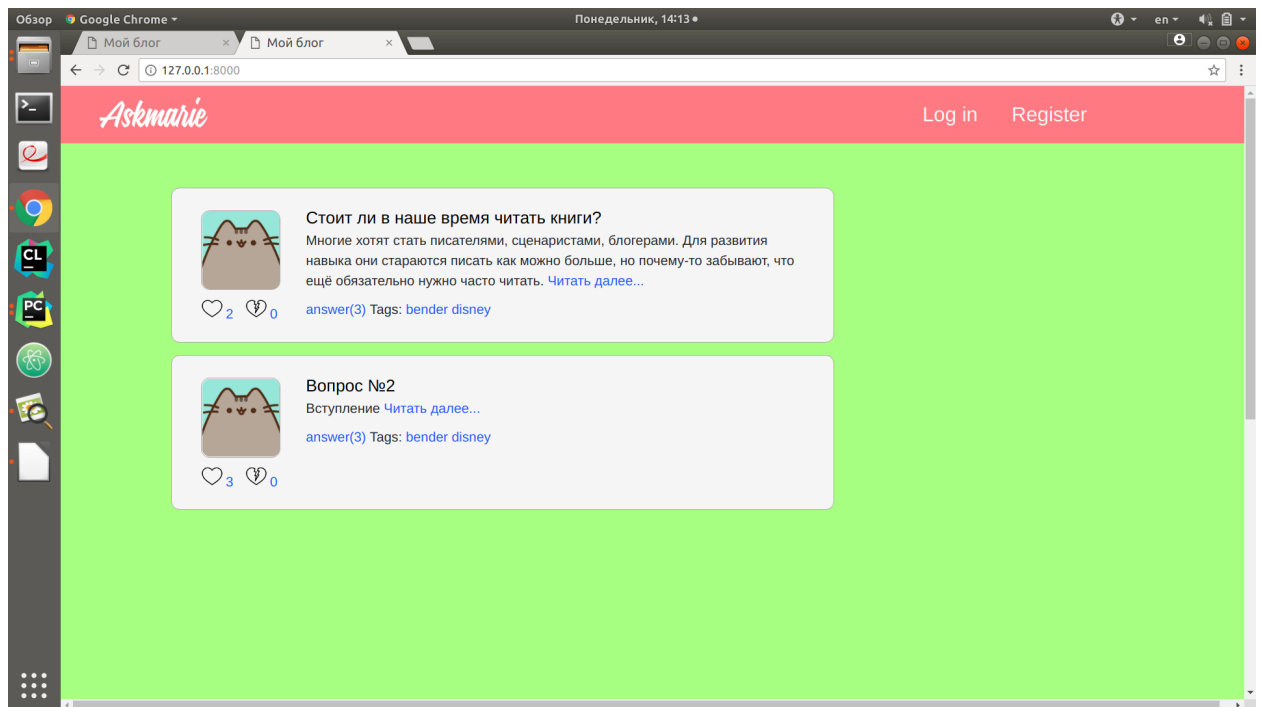
```
from django.db import models
from django.utils import timezone
# Create your models here.
```

```
class Question(models.Model):
    title = models.CharField(max_length=60)
    text = models.TextField()
    snippet = models.TextField()
    rating = models.IntegerField(default=0)
    created_at = models.DateTimeField(default=timezone.now)

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

```
class Answer(models.Model):
    author = models.CharField(max_length=50)
    text = models.TextField()
    created_at = models.DateTimeField(default=timezone.now)
    question = models.ForeignKey(Question)
    rating = models.IntegerField(default=0)
```

Результаты работы программы




ОбзорGoogle ChromeПонедельник, 14:13

Мой блогМой блогМой блог

127.0.0.1:8000/questions/1

AskmarieLog inRegister



20

Стоит ли в наше время читать книги?

Многие хотят стать писателями, сценаристами, блогерами. Для развития навыка они стараются писать как можно больше, но почему-то забывают, что ещё обязательно нужно часто читать. Стивен Кинг в своих советах начинающим писателям сказал, что писателем можно стать, если просто читать и писать. Кто-то с этим не согласится. Казалось бы, зачем будущему сценаристу читать утомительную классику, если можно посмотреть фильмы и учиться на них. Но насколько вообще в наш век электронных статей и визуальных медиа важно читать книги? Конечно, все любители чтения скажут, что очень важно. Но, как бы грустно это ни было, можно понять и тех, кто говорит, что у них просто нет времени и сил читать что-то длиннее статьи. И ведь сейчас в мире столько всего происходит, не лучше ли посвящать время тому, чтобы быть в курсе событий? С одной стороны, это так, мы должны уделять внимание тому, что происходит вокруг нас. Но не стоит забывать, что к средствам массовой информации и новостям всегда стоит относиться с осторожностью. Конечно, книги — это не истина в последней инстанции, и в них бывают ошибки. По сути, они тоже относятся к средствам массовой информации, только совершенно другого типа. Впечатления от чтения книги сильно отличаются от впечатлений при чтении статьи в газете или в Сети.

[answer\(3\)](#) Tags: [bender](#) [disney](#)