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

Умряев Д.Т.

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/'
```

lab6/urls.py

```
"""lab6 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, include
from django.contrib import admin
from tutoring.views import OrdersView, main, prog_lang, db, UniversitiesList, RegionsList, SubjectsList,
TutorsList
from django.conf.urls.static import static
from django.conf import settings

urlpatterns = [
    url(r'^admin/', admin.site.urls),
    url(r'^db/universities', UniversitiesList.as_view(), name='universities_url'),
    url(r'^db/regions', RegionsList.as_view(), name='regions_url'),
    url(r'^db/subjects', SubjectsList.as_view(), name='subjects_url'),
    url(r'^db/tutors', TutorsList.as_view(), name='tutors_url'),
    url(r'^db/', db, name='db_url'),
    url(r'^orders/', include('tutoring.urls')),
    url(r'^orders/', OrdersView.as_view(), name='orders_url'),
    url(r'^main/', main, name='main_url'),
    url(r'^(?P<prog_lang>\w+)/', prog_lang, name='prog_lang_url'),
]
if settings.DEBUG:
    urlpatterns += static(settings.STATIC_URL, document_root=settings.STATIC_URL)
```

views.py

```
from django.shortcuts import render
from django.views import View
from django.views.generic import ListView
from datetime import datetime
from tutoring.models import Education, Subjects, Regions, Tutors
# Create your views here.


class OrdersView(View):
    def get(self, request):
        variable = 'Django'
```

```
    today_date = datetime.now()
    data = {
        'orders': [
            {'title': 'Первый заказ', 'id': 1},
            {'title': 'Второй заказ', 'id': 2},
            {'title': 'Третий заказ', 'id': 3}
        ]
    }
    return render(request, 'orders.html', locals())


class OrderView(View):
    def get(self, request, id):
        variable = 'Django'
        today_date = datetime.now()
        data = {
            'order': {
                'id': id
            }
        }
        return render(request, 'order.html', locals())


def main(request):
    return render(request, 'main.html', locals())


def prog_lang(request, prog_lang):
    name = ['C++', 'Python', 'Java']
    cpp_info = 'C++ — компилируемый, статически типизированный язык программирования общего назначения. Синтаксис C++ унаследован от языка C. Одним из принципов разработки было сохранение совместимости с C. Тем не менее, C++ не является в строгом смысле надмножеством C; множество программ, которые могут одинаково успешно транслироваться как компиляторами C, так и компиляторами C++, довольно велико, но не включает все возможные программы на C.'
    java_info = 'Java — сильно типизированный объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems (в последующем приобретённой компанией Oracle). Приложения Java обычно транслируются в специальный байт-код, поэтому они могут работать на любой компьютерной архитектуре, с помощью виртуальной Java-машины. Дата официального выпуска — 23 мая 1995 года.'
    python_info = 'Python — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций.'
    info = [cpp_info, java_info, python_info]
    data1 = {'lang': {'id': prog_lang}}
    data2 = {'langs': [{'id': 'cpp', 'lang_name': 'C++', 'info': cpp_info},
                       {'id': 'java', 'lang_name': 'Java', 'info': java_info},
                       {'id': 'python', 'lang_name': 'Python', 'info': python_info}]}
```

```
    return render(request, 'prog_lang.html', locals())


def db(request):
    return render(request, 'db.html', locals())


class UniversitiesList(ListView):
    model = Education
    template_name = "universities.html"


class SubjectsList(ListView):
    model = Subjects
    template_name = "subjects.html"


class RegionsList(ListView):
    model = Regions
    template_name = "regions.html"


class TutorsList(ListView):
    model = Tutors
    template_name = "tutors.html"
```

connection.py

```
import MySQLdb


class Connection:
    def __init__(self, user, password, db, host='localhost', charset='utf8'):
        self.user = user
        self.host = host
        self.password = password
        self.db = db
        self._connection = None
        self.charset = charset

    @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,
                charset=self.charset
            )

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


class Education:
    def __init__(self, db_connection, name_university):
        self.db_connection = db_connection.connection
        self.name_university = name_university

    def save(self):
        c = self.db_connection.cursor()
        c.execute("INSERT INTO образование (ВУЗ) VALUES (%s);", (self.name_university,))
        self.db_connection.commit()
        c.close()


class Subjects:
    def __init__(self, db_connection, name_subject):
        self.db_connection = db_connection.connection
        self.name_subject = name_subject

    def save(self):
        c = self.db_connection.cursor()
        c.execute("INSERT INTO предметы (Название_предмета) VALUES (%s);", (self.name_subject,))
        self.db_connection.commit()
        c.close()


class Regions:
    def __init__(self, db_connection, name_region):
        self.db_connection = db_connection.connection
        self.name_region = name_region

    def save(self):
        c = self.db_connection.cursor()
```

```
        c.execute("INSERT INTO `tutoring`.`регионы` (`Регион`) VALUES (%s);", (self.name_region,))
        self.db_connection.commit()
        c.close()


class Tutors:
    def __init__(self, db_connection, name, surname, patronymic, email, tel, birth_date,
date_tutoring_begin, address, id_region):
        self.db_connection = db_connection.connection
        self.name = name
        self.surname = surname
        self.patronymic = patronymic
        self.email = email
        self.tel = tel
        self.birth_date = birth_date
        self.date_tutoring_begin = date_tutoring_begin
        self.address = address
        self.id_region = id_region

    def save(self):
        c = self.db_connection.cursor()
        c.execute("INSERT INTO репетиторы (Имя, Фамилия, Отчество, email, Мобильный_телефон,
Дата_рождения, Дата_начала_преподавания, Адрес, регионы_ID_региона) VALUES (%s, %s, %s,
%s, %s, %s, %s, %s, %s);", (self.name, self.surname, self.patronymic, self.email, self.tel,  self.birth_date,
self.date_tutoring_begin, self.address, self.id_region))
        self.db_connection.commit()
        c.close()


class TutorsEducation:
    def __init__(self, db_connection, id_tutor, id_education):
        self.db_connection = db_connection.connection
        self.id_tutor = id_tutor
        self.id_education = id_education

    def save(self):
        c = self.db_connection.cursor()
        c.execute("INSERT INTO репетиторы_образование (репетиторы_ID_репетитора,
образование_ID_образования) VALUES (%s, %s);", (self.id_tutor, self.id_education))
        self.db_connection.commit()
        c.close()


class TutorsSubjects:
    def __init__(self, db_connection, id_tutor, id_subject):
        self.db_connection = db_connection.connection
        self.id_tutor = id_tutor
        self.id_subject = id_subject
```

```
    def save(self):
        c = self.db_connection.cursor()
        c.execute("INSERT INTO репетиторы_предметы (репетиторы_ID_репетитора,
предметы_ID_предмета) VALUES (%s, %s);", (self.id_tutor, self.id_subject))
        self.db_connection.commit()
        c.close()


con = Connection(user='dbuser', password='123', db='tutoring')

# with con:
    # tutedu = TutorsEducation(con, '1', '1')
    # tutedu.save()
    # sub = Subjects(con, 'Математика')
    # sub.save()
    # tutsub = TutorsSubjects(con, '1', '1')
    # tutsub.save()
    # tut = Tutors(con, 'Петр', 'Петров', 'Петрович', 'ivanov@mail.ru', '89992223344', '1970.01.01',
'1999.01.01', 'Baker St. 22', '1')
    # tut.save()
    # edu = Education(con, 'МГУ')
    # edu.save()
    # reg = Regions(con, 'Москва')
    # reg.save()
```

models.py

```
from django.db import models

# Create your models here.


class Education(models.Model):
    id_university = models.AutoField(primary_key=True)
    name_university = models.CharField(max_length=100)


class Subjects(models.Model):
    id_subject = models.AutoField(primary_key=True)
    name_subject = models.CharField(max_length=100)


class Regions(models.Model):
    id_region = models.AutoField(primary_key=True)
    name_region = models.CharField(max_length=100)
```

```
class Tutors(models.Model):
    id_tutor = models.AutoField(primary_key=True)
    name = models.CharField(max_length=45)
    surname = models.CharField(max_length=45)
    patronymic = models.CharField(max_length=45)
    email = models.EmailField(max_length=254)
    tel = models.CharField(max_length=20)
    birth_date = models.DateField()
    date_tutoring_begin = models.DateField()
    address = models.CharField(max_length=100)
    region = models.ForeignKey(Regions, on_delete=models.CASCADE)
    edu = models.ManyToManyField(Education)
    subjects = models.ManyToManyField(Subjects)
```

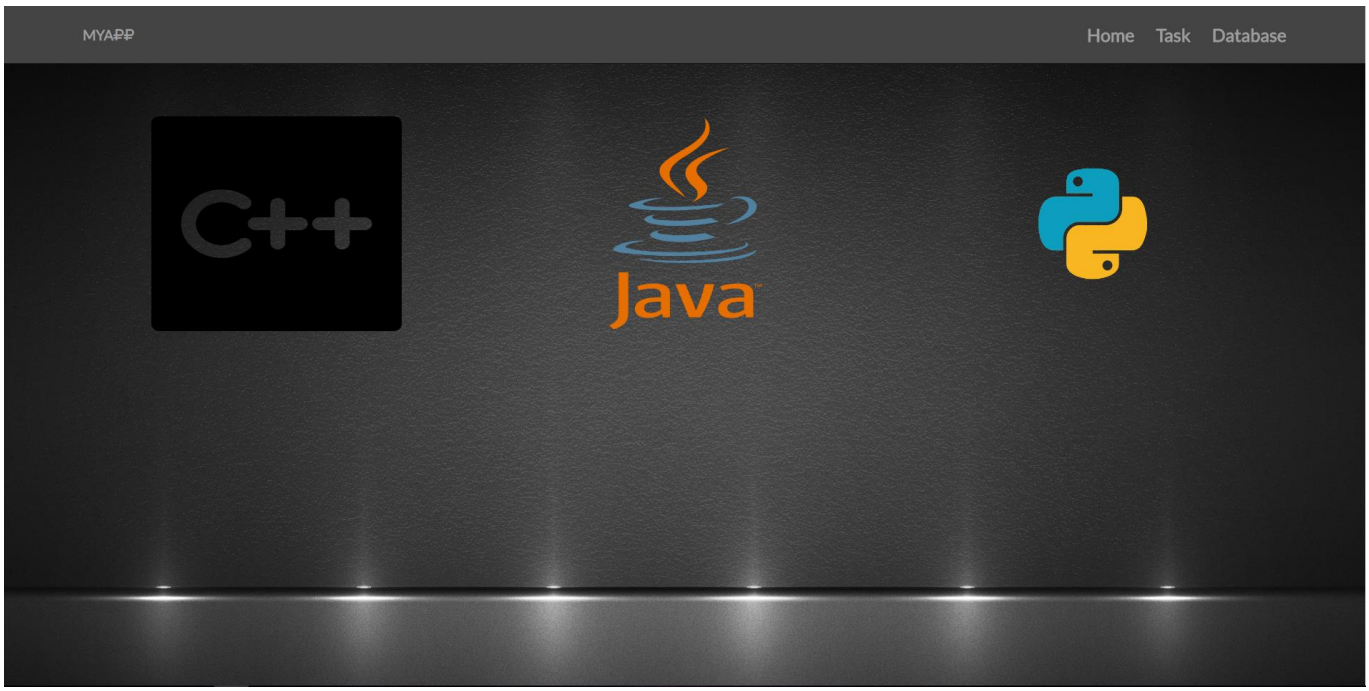
base.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Title</title>
    {% load static %}
    <link rel="stylesheet" href="{% static 'Bootstrap/css/bootstrap.css' %}">
    <link rel="stylesheet" href="{% static 'Bootstrap/css/font-awesome.min.css' %}">
    <link rel="stylesheet" href="{% static 'Bootstrap/css/main.css' %}">
</head>
<body>
    <div class="navbar navbar-inverse navbar-fixed-top">
        <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-
collapse">
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                <a class="navbar-brand" href="#">MYA<i class="fa fa-rub" aria-hidden="true"></i><i
class="fa fa-rub" aria-hidden="true"></i></a>
            </div>
            <div class="navbar-collapse collapse">
                <ul class="nav navbar-nav navbar-right">
                    <li><a href="{% url 'main_url' %}">Home</a></li>
                    <li><a href="{% url 'orders_url' %}">Task</a></li>
                </ul>
            </div>
        </div>
    </div>
    {% block body1 %}{% endblock %}
```

```
</body>
</html>
```

main.html

```
{% extends 'base.html' %}

{% block body1 %}
   <div id="header1">
      <div class="container">
         <div class="row centered">
            <div class="col-lg-8 col-lg-offset-2">
               <h1>Programming Languages</h1>
            </div>
         </div>
      </div>
      <br><br>
      <div class="container">
         <div class="row centered">
            {% load static %}
            <div class="col-lg-4">
               <a href="{% url 'prog_lang_url' 1 %}"><img src="{% static "images/C++.png" %}"
width="350" height="300" alt="C++ image"/></a>
            </div>
            <div class="col-lg-4">
               <a href="{% url 'prog_lang_url' 2 %}"><img src="{% static "images/Java.png" %}"
width="220" height="300" alt="Java image"/></a>
            </div>
            <div class="col-lg-4">
               <a href="{% url 'prog_lang_url' 3 %}"><img src="{% static "images/Python.png" %}"
width="300" height="300" alt="Python image"/></a>
            </div>
         </div>
      </div>
   </div>
{% endblock %}
```



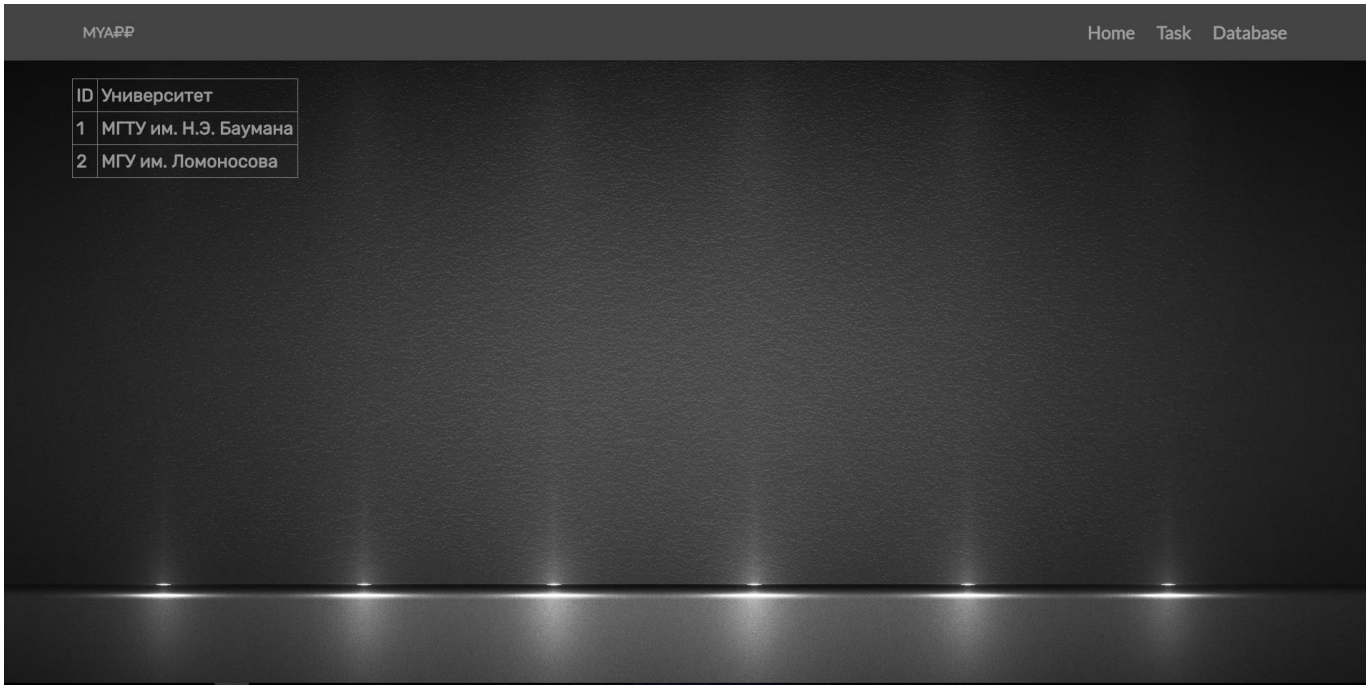
db.html

```
{% extends 'base.html' %}

{% block body1 %}
   <div id="header1">
      <div class="container">
         <div class="row centered">
            <div class="col-lg-8 col-lg-offset-2">
               <h1><br><br>Сущности</h1>
            </div>
         </div>
      </div>
      <div class="container">
         <div class="row centered">
            <li><a href="{% url 'universities_url' %}">Университеты</a></li>
            <li><a href="{% url 'regions_url' %}">Регионы</a></li>
            <li><a href="{% url 'subjects_url' %}">Предметы</a></li>
            <li><a href="{% url 'tutors_url' %}">Репетиторы</a></li>
         </div>
      </div>
   </div>
{% endblock %}
```

universitites.html

```
{% extends 'base.html' %}

{% block body1 %}
   <div id="header1">
      <div class="container">
         <div class="row centered">
            <div class="col-lg-8 col-lg-offset-2">
               <h1></h1>
            </div>
         </div>
      </div>
      <br><br>
      <div class="container">
         <div class="row centered">
         <table>
            <tr>
               <th>ID</th>
               <th>Университет</th>
            </tr>
            {% for unvier in object_list %}
               <tr>
                  <td>{{ unvier.id_university }}</td>
                  <td>{{ unvier.name_university }}</td>
               </tr>
            {% endfor %}
         </table>
         </div>
      </div>
```

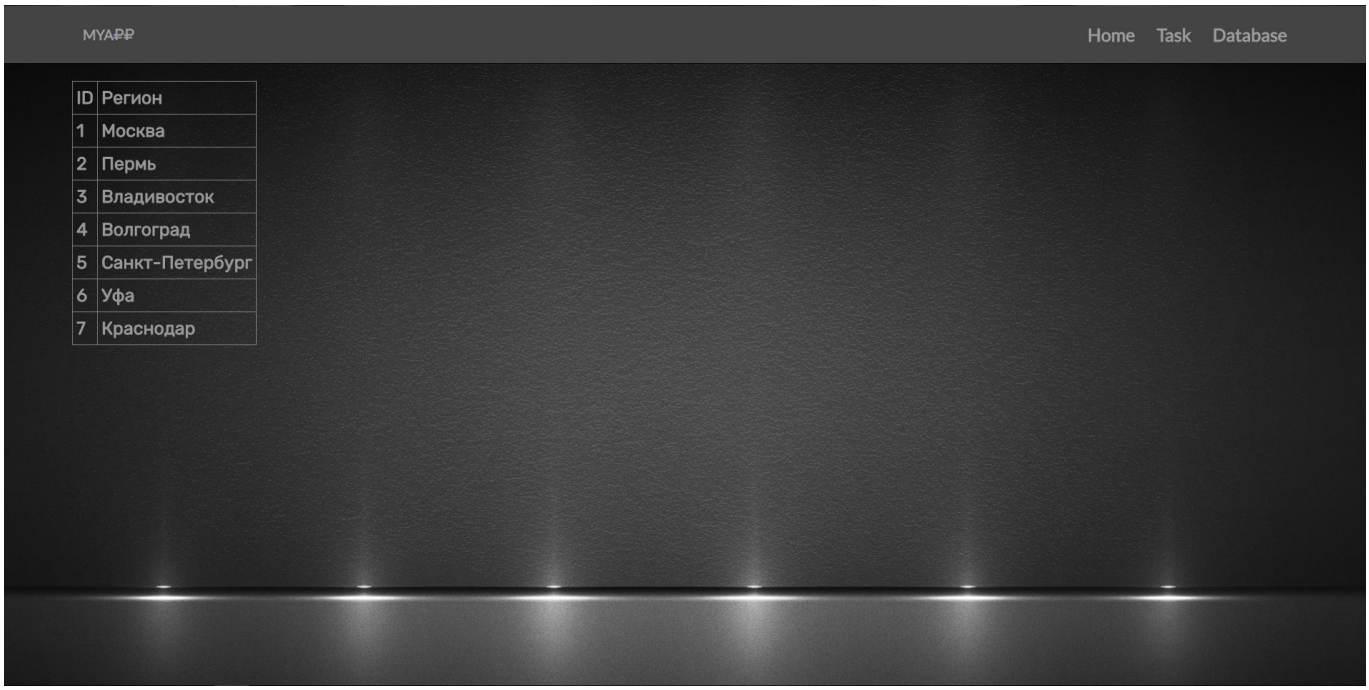
```
  </div>
{% endblock %}
```



regions.html

```
{% extends 'base.html' %}

{% block body1 %}
  <div id="header1">
    <div class="container">
      <div class="row centered">
        <div class="col-lg-8 col-lg-offset-2">
          <h1></h1>
        </div>
      </div>
    </div>
    <br><br>
    <div class="container">
      <div class="row centered">
      <table>
        <tr>
          <th>ID</th>
          <th>Регион</th>
        </tr>
        {% for region in object_list %}
          <tr>
            <td>{{ region.id_region }}</td>
            <td>{{ region.name_region }}</td>
          </tr>
        {% endfor %}
```

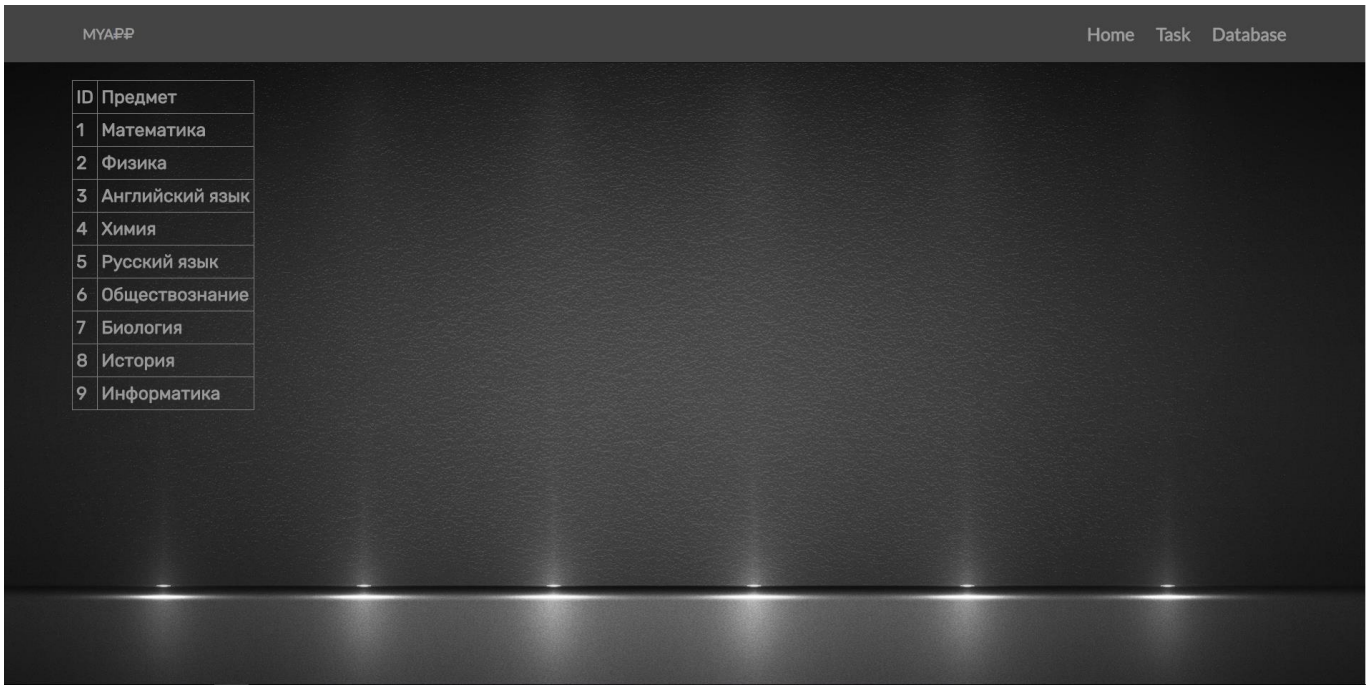
```
        </table>
        </div>
      </div>
   </div>
{% endblock %}
```



subjects.html

```
{% extends 'base.html' %}

{% block body1 %}
   <div id="header1">
      <div class="container">
         <div class="row centered">
            <div class="col-lg-8 col-lg-offset-2">
               <h1></h1>
            </div>
         </div>
      </div>
      <br><br>
      <div class="container">
         <div class="row centered">
         <table>
            <tr>
               <th>ID</th>
               <th>Предмет</th>
            </tr>
            {% for subject in object_list %}
               <tr>
                  <td>{{ subject.id_subject }}</td>
```

```
                <td>{{ subject.name_subject }}</td>
              </tr>
            {% endfor %}
          </table>
          </div>
        </div>
    </div>
{% endblock %}
```



tutors.html

```
{% extends 'base.html' %}

{% block body1 %}
    <div id="header1">
        <div class="container">
            <div class="row centered">
                <div class="col-lg-8 col-lg-offset-2">
                    <h1></h1>
                </div>
            </div>
        </div>
        <br><br>
        <div class="container">
            <div class="row centered">
            <table>
                <tr>
                    <th>ID</th>
                    <th>Имя</th>
                    <th>Фамилия</th>
```

```
            <th>Отчество</th>
            <th>Почта</th>
            <th>Телефон</th>
            <th>Дата рождения</th>
            <th>Дата начала преподавания</th>
            <th>Адрес</th>
            <th>Регион</th>
          </tr>
          {% for tutor in object_list %}
            <tr>
              <td>{{ tutor.id_tutor }}</td>
              <td>{{ tutor.name }}</td>
              <td>{{ tutor.surname }}</td>
              <td>{{ tutor.patronymic }}</td>
              <td>{{ tutor.email }}</td>
              <td>{{ tutor.tel }}</td>
              <td>{{ tutor.birth_date }}</td>
              <td>{{ tutor.date_tutoring_begin }}</td>
              <td>{{ tutor.address }}</td>
              <td>{{ tutor.region.name_region }}</td>
            </tr>
          {% endfor %}
        </table>
        </div>
      </div>
    </div>
{% endblock %}
```