Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

**«Московский государственный технический университет имени Н.Э. Баумана»   
(МГТУ им. Н.Э. Баумана)**

**Отчет по лабораторной работе № 5**

**по дисциплине «Разработка интернет приложений».**

|  |
| --- |
| ИСПОЛНИТЕЛЬ: |
| студент группы ИУ5-54 |
| Василенко Н.А. |

Москва, 2017

## Задание

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

Код программы

base.html

{% **load static** %}  
<!DOCTYPE **html**>  
<**html lang="en"**>  
 <**head**>  
 <**meta charset="utf-8"**>  
 <**meta http-equiv="X-UA-Compatible" content="IE=edge"**>  
 <**meta name="viewport" content="width=device-width, initial-scale=1"**>  
 *<!-- The above 3 meta tags \*must\* come first in the head; any other head content must come \*after\* these tags -->* <**title**>{% **block title** %}{% **endblock** %}</**title**>  
 <**link href="css/my\_style.css" rel="stylesheet" type="text/css"**>  
 <**link rel="stylesheet" type="text/css" href="**{% **static 'css/my\_style.css'** %}**"**>  
 *<!-- Bootstrap -->* <**link href="css/bootstrap.min.css" rel="stylesheet"**>  
 <**link rel="stylesheet" type="text/css" href="**{% **static 'css/bootstrap.min.css'** %}**"**>  
 *<!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->  
 <!-- WARNING: Respond.js doesn't work if you view the page via file:// -->  
 <!--[if lt IE 9]>  
 <script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>  
 <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>  
 <![endif]-->* </**head**>  
 <**body**>  
 <**div class="razm"**>  
 <**nav class="navbar navbar-default"**>  
 <**div class="container-fluid"**>  
 *<!-- Brand and toggle get grouped for better mobile display -->* <**div class="navbar-header"**>  
 <**button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#bs-example-navbar-collapse-1" aria-expanded="false"**>  
 <**span class="sr-only"**>Toggle navigation</**span**>  
 <**span class="icon-bar"**></**span**>  
 <**span class="icon-bar"**></**span**>  
 <**span class="icon-bar"**></**span**>  
 </**button**>  
 <**a class="navbar-brand" href="#"**>Brand</**a**>  
 </**div**>  
  
 *<!-- Collect the nav links, forms, and other content for toggling -->* <**div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1"**>  
 <**ul class="nav navbar-nav navbar-right"**>  
 <**li class="dropdown"**>  
 <**a href="#" class="dropdown-toggle" data-toggle="dropdown" role="button" aria-haspopup="true" aria-expanded="false"**>Dropdown <**span class="caret"**></**span**></**a**>  
 <**ul class="dropdown-menu"**>  
 <**li**><**a href="#"**>Action</**a**></**li**>  
 <**li**><**a href="#"**>Another action</**a**></**li**>  
 <**li**><**a href="#"**>Something else here</**a**></**li**>  
 <**li role="separator" class="divider"**></**li**>  
 <**li**><**a href="#"**>Separated link</**a**></**li**>  
 </**ul**>  
 </**li**>  
 </**ul**>  
 </**div**>*<!-- /.navbar-collapse -->* </**div**>*<!-- /.container-fluid -->* </**nav**>  
 {% **block body** %}Переопределяется в наследниках{% **endblock** %}  
 </**div**>  
  
  
  
 *<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->* <**script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"**></**script**>  
 *<!-- Include all compiled plugins (below), or include individual files as needed -->* <**script src="static/js/bootstrap.min.js"**></**script**>  
 </**body**>  
</**html**>

Book.html

{% **extends 'base.html'** %}  
{% **load static** %}  
{% **block title** %}Книги{% **endblock** %}  
{% **block body** %}  
 <**div**>  
 {% **for order in orders** %}  
 <**div class="alert alert-success" role="alert"**>  
 <**img src="static/image/book.jpg"**>  
 <**a href="#"**>  
 {{ **order**.**name** }} {{ **order**.**author** }} {{ **order**.**year** }}  
 </**a**>  
 </**div**>  
 {% **empty** %}  
 <**li**>пустой список</**li**>  
 {% **endfor** %}  
 </**div**>  
{% **endblock** %}

My\_style.css

.**razm**{  
 **max-width**: 70%;  
 **margin-left**: **auto**;  
 **margin-right**: **auto**;  
}  
.**form-signin**{  
 **max-width**: 60%;  
 **padding**: 15**px**;  
 **margin**: 0 **auto**;  
}  
.**error**{  
 **color**: **red**;  
}

Views.py

**from** django.shortcuts **import** render  
**from** django.views.generic **import** ListView  
**from** dz\_v1 **import** models  
**from** django.views.generic **import** View  
**from** django.http **import** HttpResponseRedirect  
**from** dz\_v1 **import** forms  
*# Create your views here.***from** django.views.generic.edit **import** FormView  
**from** django.contrib.auth.forms **import** UserCreationForm  
  
  
  
  
**class** book\_view(ListView):  
 model = models.BookModel  
  
 **def** get(self, request):  
 d = dict(orders=self.model.objects.values(**'name'**,**'author'**,**'year'**, **'description'**))  
 **return** render(request, **"book.html"**, d)

urls.py

*"""untitled10 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** dz\_v1 **import** views  
  
urlpatterns = [  
 url(**r'^admin/'**, admin.site.urls),  
 url(**r'^book/'**, views.book\_view.as\_view()),  
 ]

Models.py

**from** django.db **import** models  
  
*# Create your models here.***class** BookModel(models.Model):  
 name = models.CharField(max\_length=30)  
 author = models.CharField(max\_length=50)  
 year = models.CharField(max\_length=6)  
 description = models.CharField(max\_length=255)  
 picture = models.CharField(max\_length=100)  
  
**class** UserModel(models.Model):  
 login = models.CharField(max\_length=30, unique=**True**)  
 password = models.CharField(max\_length=30)  
 email = models.EmailField()  
 surname = models.CharField(max\_length=30)  
 name = models.CharField(max\_length=30)  
 patronymic = models.CharField(max\_length=30)  
 address = models.CharField(max\_length=100)  
 phone\_number = models.CharField(max\_length=10)  
 avatar = models.CharField(max\_length=100)  
  
 **def** validate\_unique(self, \*args, \*\*kwargs):  
 super(UserModel, self).validate\_unique(\*args, \*\*kwargs)  
 print(**"aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa"**)  
 **if not** self.login:  
 print(**"fgngmhjhgbfdghjmhgbfhjm"**)  
 **if** self.\_\_class\_\_.objects.filter(...).exists():  
 print(**"bbbbbbbbbbbbbbbbbbbbbbbbbbb"**)  
 **raise** models.ValidationError(**'Person with same ... already exists.'**)  
  
**class** OrdersModel(models.Model):  
 book\_id = models.ForeignKey(BookModel, on\_delete=models.CASCADE)  
 user\_id = models.ForeignKey(UserModel, on\_delete=models.CASCADE)

Результаты



 