

Защищено:

Гапанюк Ю. Е.

"__" _____ 2016 г.

Демонстрация:

Гапанюк Ю. Е.

"__" _____ 2016 г.

**Отчет по лабораторной работе №7
по курсу «Разработка интернет приложений»**

18

(количество листов)

ИСПОЛНИТЕЛЬ:

Студент группы ИУ5-54 _____

Повираева М. Л.

(подпись)

"__" _____ 2016 г.

Оглавление

Описание задания лабораторной работы.....	3
Реализация поставленной задачи	4
Листинг программы admin.py.....	4
Листинг программы forms.py	4
Листинг программы models.py	5
Листинг программы lab7\urls.py	5
Листинг программы views.py.....	6
Листинг программы manage.py	7
Листинг программы settings.py	8
Листинг программы urls.py.....	10
Листинг программы base.html.....	10
Листинг программы lesson.html	11
Листинг программы lessons.html	11
Листинг программы login.html	12
Листинг программы logon.html	12
Листинг программы logout.html.....	14
Листинг программы signup.html.....	14
Листинг программы success.html	15
Результаты работы программы.....	15

Описание задания лабораторной работы

Требуется создать:

1. Создайте view, которая возвращает форму для регистрации.

Поля формы:

- Логин
- Пароль
- Повторный ввод пароля
- Email
- Фамилия
- Имя

2. Создайте view, которая возвращает форму для авторизации.

Поля формы:

- Логин
- Пароль

3. При отправке формы регистрации во view проверять каждый параметр по правилам валидации, если валидация всех полей пройдена, то создавать пользователя и делать перенаправление на страницу логина, а ошибки, если они есть, выводить над формой.

Правила валидации:

- Логин не меньше 5 символов
- Пароль не меньше 8 символов
- Пароли должны совпадать
- Все поля должны быть заполнены
- Логин – уникален для каждого пользователя

4. При возникновении ошибок в момент отправки формы, введенные значения в полях ввода, кроме пароля, не должны исчезать.

5. Переписать view регистрации с использованием Django Form, правила валидации удалить из view, использовать встроенный механизм валидации полей.

6. Во view авторизации реализовать логин при POST запросе. При успешной авторизации должен происходить переход на страницу успешной авторизации.

7. Страница успешной авторизации должна проверять, что пользователь авторизован. Иначе делать перенаправление на страницу авторизации.

8. Реализовать view для выхода из аккаунта.

9. Заменить проверку на авторизацию на декоратор login_required.

10. Добавить superuser'а через команду manage.py.

11. Подключить django.contrib.admin и войти в панель администрирования.

12. Зарегистрировать все свои модели в django.contrib.admin.

13. Для выбранной модели настроить страницу администрирования:

- Настроить вывод необходимых полей в списке
- Добавить фильтры
- Добавить поиск
- Добавить дополнительное поле в список

Реализация поставленной задачи

Листинг программы admin.py

```
from django.contrib import admin

# Register your models here.

from lab7.models import Lesson, MyUser

class LessonAdmin(admin.ModelAdmin):
    list_display = ('name', 'description', 'datetime', 'desc_len')
    list_filter = ['datetime']
    search_fields = ('id', 'name')

    def desc_len(self, obj):
        return len(obj.description)
    desc_len.short_description = "Длина сообщения"

admin.site.register(Lesson, LessonAdmin)

class MyUserAdmin(admin.ModelAdmin):
    pass

admin.site.register(MyUser, MyUserAdmin)
```

Листинг программы forms.py

```
# __author__ = 'Work'
from django import forms
from django.core.exceptions import ValidationError
from django.core.validators import validate_email
from django.contrib.auth.models import User
from django.contrib.auth import get_user_model

class LoginForm(forms.Form):
    login = forms.CharField(label='Логин')
    password = forms.CharField(label='Пароль', widget=forms.PasswordInput)

class RegistrationForm(forms.Form):
    login = forms.CharField(label='Логин', min_length=5)
    password = forms.CharField(label='Пароль', min_length=8,
    widget=forms.PasswordInput)
    repeat_password = forms.CharField(label='Повторите пароль',
    widget=forms.PasswordInput)
    email = forms.CharField(label='Адрес электронной Почты')
    first_name = forms.CharField(label='Имя')
    last_name = forms.CharField(label='Фамилия')

    def clean_login(self):
        user_model = get_user_model()
        login = self.cleaned_data['login']
        if user_model.objects.filter(username=login):
            raise ValidationError('Этот login уже занят')
        return login

    def clean_email(self):
        user_model = get_user_model()
        email = self.cleaned_data['email']
        validate_email(self.cleaned_data['email'])
        if user_model.objects.filter(email=email):
```

```

        raise ValidationError('Этот email уже зарегистрирован')
    return self.cleaned_data['email']

    def clean(self):
        cleaned_data = super(RegistrationForm, self).clean()
        if self.cleaned_data.get('password') and
self.cleaned_data.get('repeat_password'):
            if self.cleaned_data['password'] !=
self.cleaned_data['repeat_password']:
                raise ValidationError('Пароли не совпадают')
            return cleaned_data

    def save(self):
        user_model = get_user_model()
        user =
user_model.objects.create_user(username=self.cleaned_data['login'],
                                email=self.cleaned_data['email'],

password=self.cleaned_data['password'],

first_name=self.cleaned_data['first_name'],

last_name=self.cleaned_data['last_name'],
                                )

        return user

```

Листинг программы models.py

```

from django.db import models
from django.contrib.auth.models import AbstractUser

# Create your models here.

class MyUser(AbstractUser):
    pass

class Lesson(models.Model):
    name = models.CharField(max_length=30)
    description = models.CharField(max_length=255)
    datetime = models.DateTimeField(auto_now=True)

```

Листинг программы lab7\urls.py

```

"""Templating URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/1.9/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. Add an import:  from blog import urls as blog_urls
    2. Import the include() function: from django.conf.urls import url,
include
    3. Add a URL to urlpatterns:  url(r'^blog/', include(blog_urls))
"""
from django.conf.urls import url

from . import views

```

```
urlpatterns = [
    url(r'^$', views.ExampleView.as_view()),
    url(r'^lessons/', views.LessonsView.as_view()),
    url(r'^lesson/(?P<id>\d+)', views.LessonView.as_view(),
name='lesson_url'),
    url(r'^signupold/', views.registration_old, name='signupNOFORM'),
    url(r'^signup/', views.registration, name='signup'),
    url(r'^login/', views.authorization, name='login'),
    url(r'^logout/', views.exit, name='logout'),
    url(r'^success/', views.SuccessView.as_view(), name='success'),
]
```

Листинг программы views.py

```
from django.shortcuts import render, redirect
from django.http import HttpResponse, HttpResponseRedirect
from django.views import View
from django.views.generic import ListView
from django.views.generic import TemplateView
from django.contrib.auth.decorators import login_required

from lab7.models import *
from lab7.forms import *
from django import forms
from django.contrib.auth.hashers import make_password
from django.contrib.auth import authenticate, logout
from django.contrib import auth

# Create your views here.

class ExampleView(View):
    def get(self, request):
        return render(request, 'base.html')

class SuccessView(View):
    def get(self, request):
        return render(request, 'success.html')

class LessonsView(ListView):
    model = Lesson
    context_object_name = 'lessons'
    template_name = 'lessons.html'

    def get_queryset(self):
        qs = Lesson.objects.all().order_by('id')
        return qs

class LessonView(View):
    def get(self, request, id):
        data = Lesson.objects.get(id__exact=id)
        return render(request, 'lesson.html', {'lesson':data})

def registration_old(request):
    errors = []
    if request.method == 'POST':
        username = request.POST.get('username')
        if not username:
            errors.append('Login required')
        elif len(username)<5:
            errors.append('Login should be 5 or more symbols')

        password = request.POST.get('password')
        if not password:
```

```

        errors.append('Password required')
    elif len(password)<8:
        errors.append('Password length should be 8 or more')

    password_repeat = request.POST.get('password2')

    if password != password_repeat:
        errors.append('Password should be similar')
    #print(errors)
    if not errors:
        user_model = get_user_model()
        user = user_model.objects.create_user(username=username,
                                                email=request.POST.get('email'),
                                                password=password,

first_name=request.POST.get('first_name'),

last_name=request.POST.get('last_name'),
                                )
        return HttpResponseRedirect('/lab7/success')
    return render(request, 'login.html', {'errors': errors})

def registration(request):
    if request.method == 'POST':
        form = RegistrationForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('/lab7/')
        return render(request, 'signup.html', {'form': form})
    else:
        form = RegistrationForm()
    return render(request, 'signup.html', {'form': form})

def authorization(request):
    redirect_url = '/lab7/success'
    if request.method == 'POST':
        form = LoginForm(request.POST)
        if form.is_valid():
            user = auth.authenticate(username=form.cleaned_data['login'],
                                    password=form.cleaned_data['password'])

            if user is not None:
                auth.login(request, user)
                return HttpResponseRedirect(redirect_url)
            else:
                form.add_error(None, 'Wrong login or password')
        else:
            form = LoginForm()
    return render(request, 'login.html', {'form':form, 'continue':
redirect_url})

@login_required
def exit(request):
    logout(request)
    return render(request, 'logout.html')

```

Листинг программы manage.py

```

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

if __name__ == "__main__":
    os.environ.setdefault("DJANGO_SETTINGS_MODULE", "lab7mary.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)

```

Листинг программы settings.py

```

"""
Django settings for lab7mary project.

Generated by 'django-admin startproject' using Django 1.10.4.

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 = '^hc)3#x3a^!$as5u0tqa7#yz0#k8odi3ujy7prk$z2dxdhrfh'

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

ALLOWED_HOSTS = []

AUTH_USER_MODEL = 'lab7.MyUser'

# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'lab7',
]

```



```

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 = 'lab7mary.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [],
        '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 = 'lab7mary.wsgi.application'

# Database
# https://docs.djangoproject.com/en/1.10/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'lab7',
        'USER': 'mary',
        'PASSWORD': '123mary123',
        'OPTIONS': {'charset': 'utf8'},
        'TEST_CHARSET': 'utf8',
    }
}

# Password validation
# https://docs.djangoproject.com/en/1.10/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/1.10/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/'
```

Листинг программы urls.py

```

"""lab7mary 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 include, url
from django.contrib import admin

urlpatterns = [
    url(r'^admin/', admin.site.urls),
    url(r'^lab7/', include('lab7.urls')),
]

```

Листинг программы base.html

```

{% load static %}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <link rel="stylesheet" type="text/css" href="{% static
'css/bootstrap.css' %}">
    <title>{% block title %}ТЕСТОВЫЙ ВЫВОД{% endblock %}</title>
</head>

```

```

<body>
<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">ЛАБА #7</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">
        <li><a href="/lab7/">Базовый класс</a></li>
        <li><a href="/lab7/lessons/">Уроки</a></li>
      </ul>
      <ul class="nav navbar-nav navbar-right">
        {% if user.is_authenticated %}
          <li><div style="padding-top:15px"><i><b>Здравствуйте, {{
user.get_short_name }}</b></i></div></li>
          <li><a href="{% url 'logout' %}">Выйти из {{ user.get_username
}}</a></li>
        {% else %}
          <li><div style="padding-top:15px"><i><b>Вы вошли как
Гость</b></i></div></li>
          <li><a href="{% url 'signup' %}">Зарегистрироваться</a></li>
          <li><a href="{% url 'login' %}">Войти</a></li>
        {% endif %}
      </ul>
    </div><!-- /.navbar-collapse -->
  </div><!-- /.container-fluid -->
</nav>

<div>
  {% block body %}Redefined by spawnlings{% endblock %}
</div>
</body>
</html>

```

Листинг программы lesson.html

```

{% extends 'base.html' %}

{% block title %}Урок №{{ lesson.id }}{% endblock %}

{% block body %}
<h1>Урок №{{ lesson.id }} &ndash; {{ lesson.name }}</h1>
<p>{{ lesson.description }}</p>
{% endblock %}

```

Листинг программы lessons.html

```

{% extends 'base.html' %}

{% block title %}Уроки{% endblock %}

{% block body %}
<ul class="list-group">

```

```

        {% for lesson in lessons %}
            <li class="list-group-item"><a href="{% url 'lesson_url'
lesson.id %}">{{ lesson.id }} &ndash; {{ lesson.name }}</a></li>
        {% empty %}
            <li class="list-group-item disabled">empty list</li>
        {% endfor %}
    </ul>
{% endblock %}

```

Листинг программы login.html

```

{% extends 'base.html' %}

{% block title %}Вход{% endblock %}

{% block body %}

    {{ form.non_field_errors }}
    <form action="/lab7/login/" method="post" class="form-horizontal">
        {% csrf_token %}
        {% for i in form %}
            <div class="form-group">
                <label class="col-sm-2">{{ i.label }}</label>
                <div class="col-sm-10">
                    <div>
                        {{ i }}
                    </div>
                </div>
            </div>
        {% endfor %}

        <div class="form-group">
            <div class="col-sm-offset-2 col-sm-10">
                <input type="submit" value="Войти" class="btn
btn-default"/>
            </div>
        </div>
    </form>
{% endblock %}

```

Листинг программы logon.html

```

{% extends 'base.html' %}

{% block title %}Вход{% endblock %}

{% block body %}

    <form action="/lab7/signupold/" method="post" class="form-horizontal">
        {% csrf_token %}
        <div class="form-group">
            <label class="col-sm-4">ЛОГИН</label>
            <div class="col-sm-8">
                <div>
                    <input id="id_login" minlength="5" name="username"
type="text" required />
                </div>
            </div>
        </div>
    </form>

```

```

<div class="form-group">
  <label class="col-sm-4">Пароль</label>
  <div class="col-sm-8">
    <div>
      <input id="id_password" minlength="8" name="password"
type="password" required />
    </div>
    <div>
      </div>
    </div>
  </div>
</div>

<div class="form-group">
  <label class="col-sm-4">Повторите пароль</label>
  <div class="col-sm-8">
    <div>
      <input id="id_repeat_password" name="password2"
type="password" required />
    </div>
    <div>
      </div>
    </div>
  </div>
</div>

<div class="form-group">
  <label class="col-sm-4">Адрес электронной Почты</label>
  <div class="col-sm-8">
    <div>
      <input id="id_email" name="email" type="text" required />
    </div>
    <div>
      </div>
    </div>
  </div>
</div>

<div class="form-group">
  <label class="col-sm-4">Имя</label>
  <div class="col-sm-8">
    <div>
      <input id="id_first_name" name="first_name" type="text"
required />
    </div>
    <div>
      </div>
    </div>
  </div>
</div>

<div class="form-group">
  <label class="col-sm-4">Фамилия</label>
  <div class="col-sm-8">
    <div>
      <input id="id_last_name" name="last_name" type="text"
required />
    </div>
    <div>
      </div>
    </div>
  </div>
</div>

```

```

</div>

        <div class="form-group">
            <div class="col-sm-offset-2 col-sm-10">
                <input type="submit" value="Зарегистрироваться"
class="btn btn-default"/>
            </div>
        </div>
</form>
{% endblock %}

```

Листинг программы logout.html

```

{% extends 'base.html' %}

{% block title %}Выход{% endblock %}

{% block body %}
<p>Выход произошёл успешно.</p>
{% endblock %}

```

Листинг программы signup.html

```

{% extends 'base.html' %}

{% block title %}Регистрация{% endblock %}

{% block body %}

{{ form.non_field_errors }}
<form action="/lab7/signup/" method="post" class="form-horizontal"
enctype="multipart/form-data">
    {% csrf_token %}
    {% for i in form %}
        <div class="form-group">
            <label class="col-sm-4">{{ i.label }}</label>
            <div class="col-sm-8">
                <div>
                    {{ i }}
                </div>
                <div>
                    {{ i.errors }}
                </div>
            </div>
        </div>
    {% endfor %}

    <div class="form-group">
        <div class="col-sm-offset-4 col-sm-8">
            <input type="submit" value="Зарегистрировать меня" class="btn
btn-default"/>
        </div>
    </div>
</form>
{% endblock %}

```

Листинг программы success.html

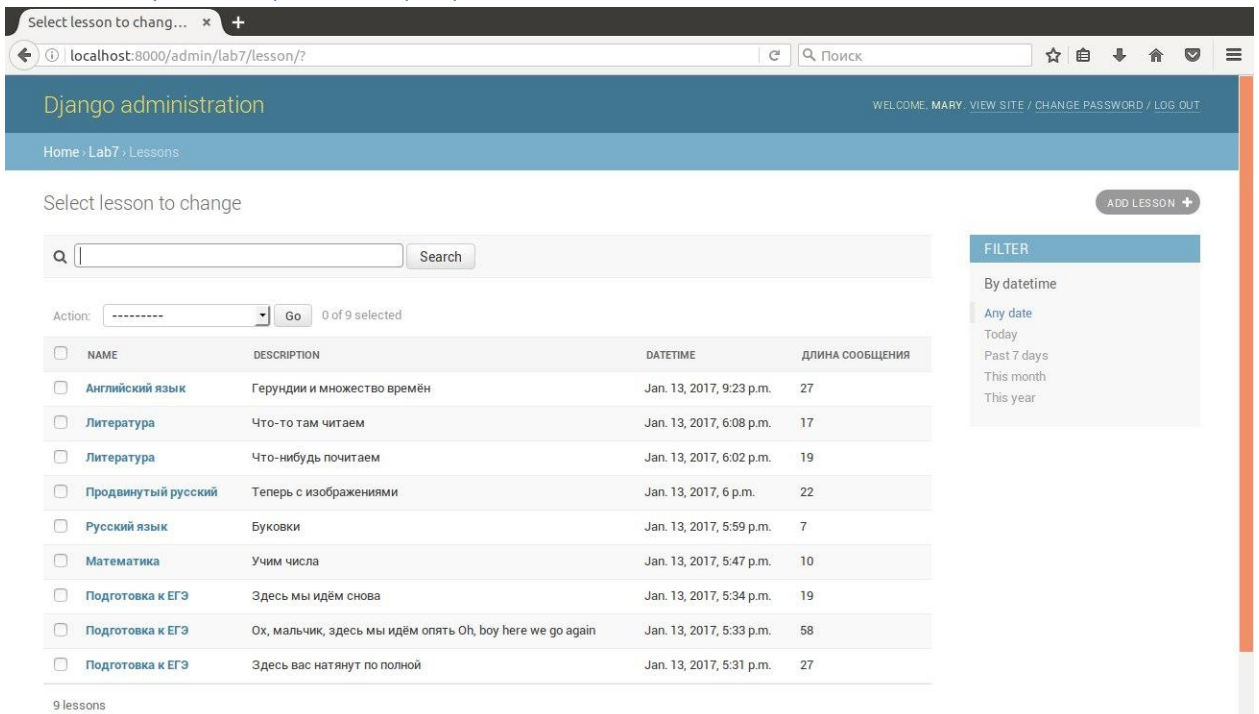
```
{% extends 'base.html' %}

{% block title %}АЛЛИЛУЯ!{% endblock %}

{% block body %}

{% if user.is_authenticated %}
<p>Пользователь {{ user.get_username }} успешно вошёл :)</p>
{% else %}
<p>Авторизации не было</p>
{% endif %}
{% endblock %}
```

Результаты работы программы



The screenshot shows the Django administration interface for a site named 'Django administration'. The user is logged in as 'MARY'. The breadcrumb trail is 'Home > Lab7 > Lessons'. The main heading is 'Select lesson to change'. There is a search bar and a table of lessons. The table has columns: NAME, DESCRIPTION, DATETIME, and ДЛИНА СООБЩЕНИЯ. There are 9 lessons listed. A filter sidebar on the right shows 'By datetime' with options: Any date, Today, Past 7 days, This month, and This year.

9 lessons

<input type="checkbox"/>	NAME	DESCRIPTION	DATETIME	ДЛИНА СООБЩЕНИЯ
<input type="checkbox"/>	Английский язык	Герундии и множество времён	Jan. 13, 2017, 9:23 p.m.	27
<input type="checkbox"/>	Литература	Что-то там читаем	Jan. 13, 2017, 6:08 p.m.	17
<input type="checkbox"/>	Литература	Что-нибудь почитаем	Jan. 13, 2017, 6:02 p.m.	19
<input type="checkbox"/>	Продвинутый русский	Теперь с изображениями	Jan. 13, 2017, 6 p.m.	22
<input type="checkbox"/>	Русский язык	Буковки	Jan. 13, 2017, 5:59 p.m.	7
<input type="checkbox"/>	Математика	Учим числа	Jan. 13, 2017, 5:47 p.m.	10
<input type="checkbox"/>	Подготовка к ЕГЭ	Здесь мы идём снова	Jan. 13, 2017, 5:34 p.m.	19
<input type="checkbox"/>	Подготовка к ЕГЭ	Ох, мальчик, здесь мы идём опять Oh, boy here we go again	Jan. 13, 2017, 5:33 p.m.	58
<input type="checkbox"/>	Подготовка к ЕГЭ	Здесь вас натянут по полной	Jan. 13, 2017, 5:31 p.m.	27

Site administration | Dj...

localhost:8000/admin/Поиск

Django administration

WELCOME, MARY. [VIEW SITE](#) / [CHANGE PASSWORD](#) / [LOG OUT](#)

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups

+ Add

Change

LAB7

Lessons

+ Add

Change

Users

+ Add

Change

Recent actions

My actions

+ Lesson object

Lesson

Выход

localhost:8000/lab7/logout/Поиск

ЛАБА #7

Базовый класс

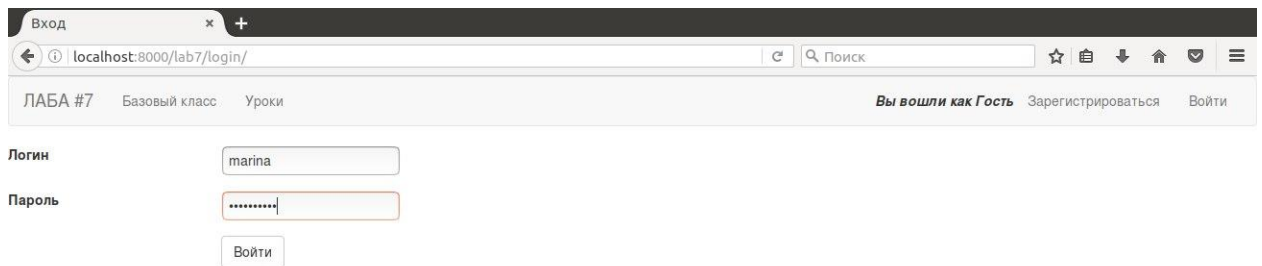
Уроки

Вы вошли как Гость

Зарегистрироваться

Войти

Выход произошёл успешно.



Регистрация

localhost:8000/lab7/signup/Поиск

ЛАБА #7Базовый классУроки

Вы вошли как ГостьЗарегистрироватьсяВойти

Логин

mary

• Ensure this value has at least 5 characters (it has 4).

Пароль

Повторите пароль

Адрес электронной Почты

marynight@test.com

Имя

Мор

Фамилия

Пех

Зарегистрировать меня