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»

Отчет по домашнему заданию

Исполнитель:

Студент группы ИУ5-51

Калугина Д.А.

Преподаватель:

Гапанюк Ю.Е.

«___» _______________

____

Москва 2017г.

Задание:

Разработать веб-сервис на базе технологий: Python, Django, JS, MySQL.

Код: Views.py
```
from django.shortcuts import render
from django.http import HttpResponseRedirect, HttpResponse, JsonResponse
from django import forms
from django.contrib.auth import authenticate, login, logout
from django.contrib.auth.decorators import login_required
from django.views.generic import ListView, View
from django.views.generic import DetailView
import datetime
from .models import *


# Create your views here.
def home(request):
    parameters = {
        'header': "Содержимое"
    }
    return render(request, 'home.html', context=parameters)


class GroupsView(ListView):
    model = Group
    template_name = 'home.html'
    context_object_name = 'group_list'


class PersonsView(ListView):
    model = Person
    template_name = 'home.html'
    context_object_name = 'persons_list'


# форма регистрации
```

```
class RegistrationForm(forms.Form):
    username =
forms.CharField(min_length=5,
label='Логин')
    password =
forms.CharField(min_length=8,
widget=forms.PasswordInput,
label='Пароль')
    password2 =
forms.CharField(min_length=8,
widget=forms.PasswordInput,
label='Повторите ввод')
    email =
forms.EmailField(label='Email')
    last_name =
forms.CharField(label='Фамилия')
    first_name =
forms.CharField(label='Имя')


class AuthorizationForm(forms.Form):
    username =
forms.CharField(label='Логин')
    password =
forms.CharField(widget=forms.PasswordIn
put, label='Пароль')


class GroupForm(forms.ModelForm):
    class Meta(object):
        model = Group
        fields = ['name', 'genre',
'description', 'pic']

    def save(self):
        group = Group()

        group.name =
self.cleaned_data.get('name')

        group.genre =
self.cleaned_data.get('genre')

        group.description =
self.cleaned_data.get('description')
        group.pic =
self.cleaned_data.get('pic')
        group.save()


def add(request):
    if request.method == 'POST':
        name1 =
request.POST.get('name')
        genre1 =
request.POST.get('genre')
        member =
request.POST.get('member')
        date = request.POST.get('date')
        print(name1)
        print(member)
        description1 =
request.POST.get('description')
        pic1 = request.FILES.get('pic')
        group1 = Group(name=name1,
genre=genre1, description=description1,
                       pic=pic1)

        group1.save()

        return
HttpResponseRedirect('/item-' +
str(group1.id))

    return render(request, 'add.html',
locals())


# регистрация
def registration(request):
    if request.method == 'POST':
        form =
RegistrationForm(request.POST)
        is_val = form.is_valid()
        data = form.cleaned_data
        if data['password'] !=
data['password2']:
            is_val = False
            form.add_error('password2',
['Пароли должны совпадать'])
        if
User.objects.filter(username=data['user
name']).exists():
            form.add_error('username',
['Такой логин уже существует'])
            is_val = False

        if is_val:
            data = form.cleaned_data
            user =
User.objects.create_user(data['username
'], data['email'], data['password'])
            pers = Person()
            pers.user = user
            pers.first_name =
data['first_name']
            pers.last_name =
data['last_name']

            pers.save()
            return
HttpResponseRedirect('/authorization')
    else:
        form = RegistrationForm()

    return render(request,
'registration.html', {'form': form})


# авторизация django
def authorization(request):
    if request.method == 'POST':
        form =
AuthorizationForm(request.POST)
        print(form)
        data = form.cleaned_data

        if form.is_valid():
            user =
authenticate(request,
username=data['username'],
password=data['password'])
```

```
            # user =
authenticate(request,
username='petrov',password='12345678')
            if user is not None:
                login(request, user)
                return
HttpResponseRedirect('/success_authoriz
ation')
            else:

form.add_error('username', ['Неверный
логин или пароль'])
                # raise
forms.ValidationError('Имя пользователя
и пароль не подходят')

    else:
        form = AuthorizationForm()

    return render(request,
'authorization.html', {'form': form})


# успешная авторизация django
@login_required(login_url='/authorizati
on')
def success_authorization(request):
    return HttpResponseRedirect('/')



# выход
def logout_view(request):
    logout(request)
    return HttpResponseRedirect('/')



class OneItem(DetailView):
    model = Group
    context_object_name = 'group'
    template_name = 'object.html'

    def get_context_data(self,
**kwargs):
        context = super(OneItem, 
self).get_context_data(**kwargs)

        relation = 
Membership.objects.filter(group=self.kw
args['pk'])
        # print(relation)
        customers_list = []
        for rel in relation:
            group = 
Group.objects.get(id=rel.group_id)
            # print(group)
            member = 
Person.objects.get(id=rel.person_id)
            # print(member)
            if member not in 
customers_list:
                # print(member.user)

customers_list.append(member.user)
        # print(customers_list)
        context['customers_list'] = 
customers_list
        context['group_id'] = 
self.kwargs['pk']

        return context


def enter(request):
    if request.method == "GET":
        user = 
User.objects.filter(username=request.GE
T['user_name'])
        pers = 
Person.objects.get(user=user)
        group = 
Group.objects.get(id=request.GET['group
_id'])
        mem = 
Membership.objects.create(person=pers, 
group=group,
date_joined=datetime.datetime.now().dat
e())

    return HttpResponse("ok")
```

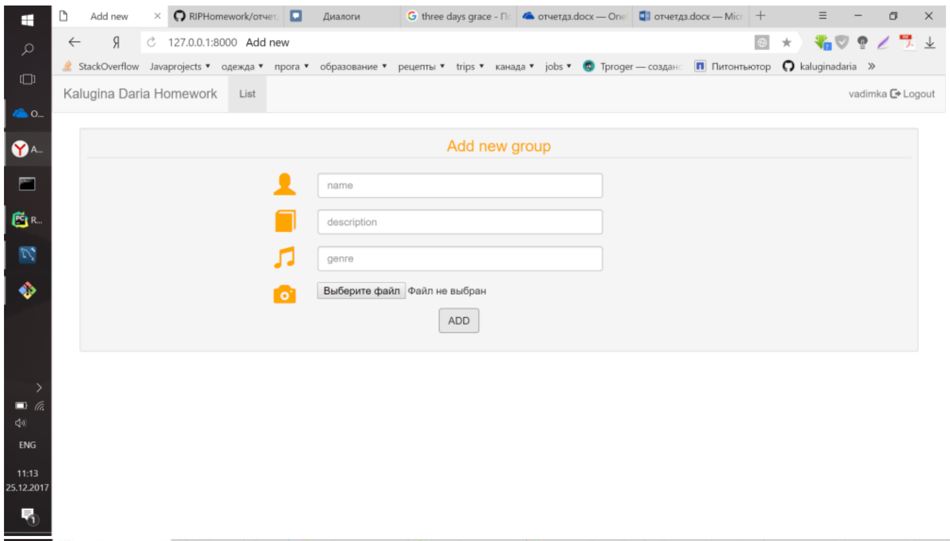
Models.py

```
from django.db
import models

from
django.contrib.au
th.models import
User, UserManager
from
django.contrib
import admin
from django.utils
import timezone
# Create your
models here.

from django.db
import models


class
Person(models.Mod
el):
    user =
models.OneToOneFi
eld(User,
on_delete=models.
CASCADE)
    email =
models.CharField(
max_length=40)
    first_name =
models.CharField(
max_length=40)
    last_name =
models.CharField(
max_length=40)
```

```
    def
__str__(self):
        return
self.user.usernam
e


class
Group(models.Mode
l):
    name =
models.CharField(
max_length=128,
blank=False,
null=False)
    members =
models.ManyToMany
Field(Person,
through='Membersh
ip')
    genre =
models.CharField(
max_length=100)
    description =
models.TextField(
max_length=500,
default='No
description yet')
    pic =
models.ImageField
(upload_to="hw/",
null=True,
blank=True,
max_length=1000)

    def
__str__(self):
        return
self.name


class
Membership(models
.Model):
    person =
models.ForeignKey
(Person)
    group =
models.ForeignKey
(Group)
    date_joined =
models.DateField(
)

    def
__str__(self):
        return
str(self.person)+
" in
"+str(self.group)
```

urls.py
```
from django.db import models

from django.contrib.auth.models import User, UserManager
from django.contrib import admin
from django.utils import timezone
# Create your models here.

from django.db import models


class Person(models.Model):
    user = models.OneToOneField(User,
on_delete=models.CASCADE)
    email = models.CharField(max_length=40)
    first_name = models.CharField(max_length=40)
    last_name = models.CharField(max_length=40)

    def __str__(self):
        return self.user.username


class Group(models.Model):
    name = models.CharField(max_length=128, blank=False,
null=False)
```

```
    members = models.ManyToManyField(Person,
through='Membership')
    genre = models.CharField(max_length=100)
    description = models.TextField(max_length=500,
default='No description yet')
    pic = models.ImageField(upload_to="hw/",  null=True,
blank=True, max_length=1000)

    def __str__(self):
        return self.name


class Membership(models.Model):
    person = models.ForeignKey(Person)
    group = models.ForeignKey(Group)
    date_joined = models.DateField()

    def __str__(self):
        return str(self.person)+" in "+str(self.group)
```



Kalugina Daria Homework List

vadimka Logout

Add new group

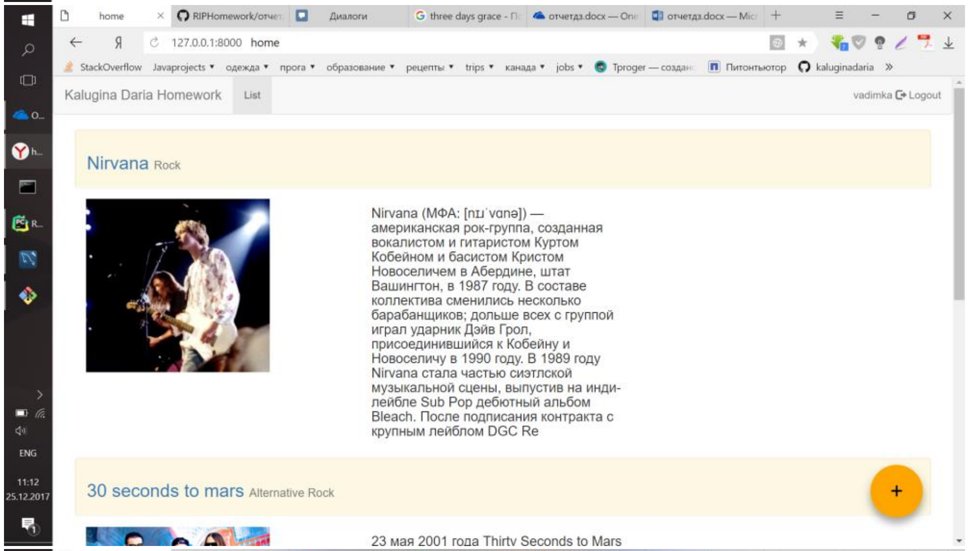
name

description

genre

Выберите файл Файл не выбран

ADD



Kalugina Daria Homework List

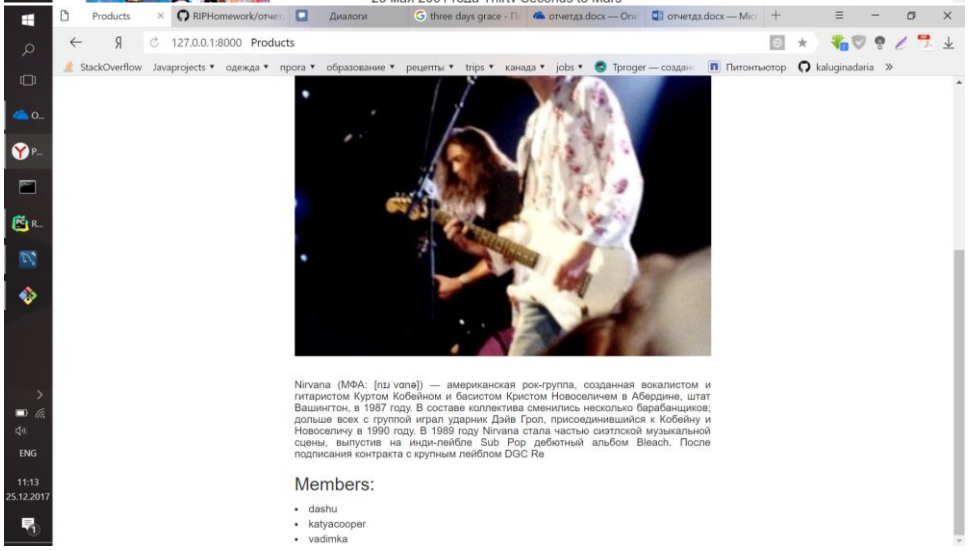
vadimka Logout

Nirvana Rock

Nirvana (МФА: [nɪɪˈvɑnə]) — американская рок-группа, созданная вокалистом и гитаристом Куртом Кобейном и басистом Кристом Новоселичем в Абердине, штат Вашингтон, в 1987 году. В составе коллектива сменились несколько барабанщиков; дольше всех с группой играл ударник Дэйв Грол, присоединившийся к Кобейну и Новоселичу в 1990 году. В 1989 году Nirvana стала частью сиэтлской музыкальной сцены, выпустив на инди-лейбле Sub Pop дебютный альбом Bleach. После подписания контракта с крупным лейблом DGC Re

30 seconds to mars Alternative Rock

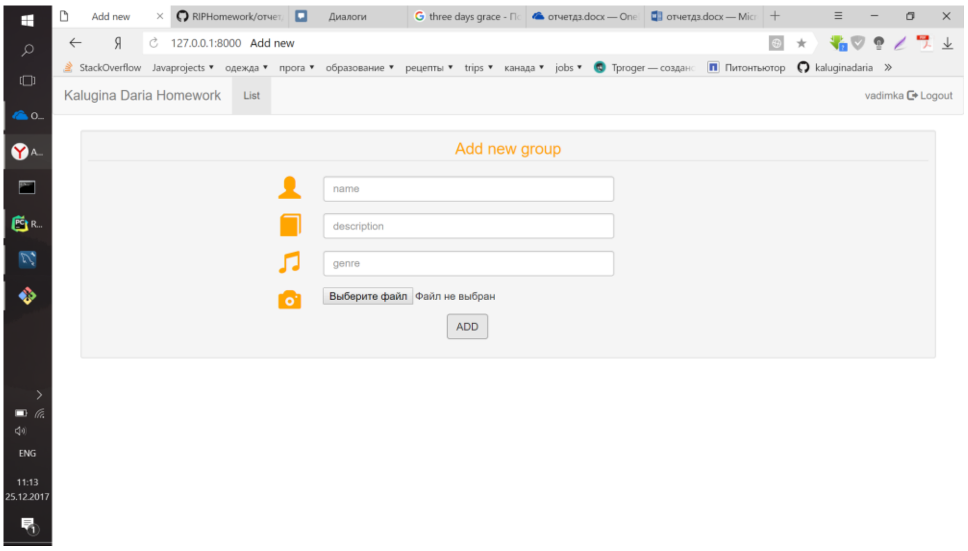
23 мая 2001 года Thirty Seconds to Mars



Nirvana (МФА: [nɪɪˈvɑnə]) — американская рок-группа, созданная вокалистом и гитаристом Куртом Кобейном и басистом Кристом Новоселичем в Абердине, штат Вашингтон, в 1987 году. В составе коллектива сменились несколько барабанщиков; дольше всех с группой играл ударник Дэйв Грол, присоединившийся к Кобейну и Новоселичу в 1990 году. В 1989 году Nirvana стала частью сиэтлской музыкальной сцены, выпустив на инди-лейбле Sub Pop дебютный альбом Bleach. После подписания контракта с крупным лейблом DGC Re

Members:

- dashu
- katyacooper
- vadimka


Add new
RIPHomework/отчет
Диалоги
three days grace
отчетдз.docx — One
отчетдз.docx — Mic
127.0.0.1:8000 Add new
StackOverflow
Javaprojects
одежда
прога
образование
рецепты
trips
канада
jobs
Tproger — создано
Питонтьютор
kaluginadaria
Kalugina Daria Homework
List
vadimka
Logout
Add new group
name
description
genre
Выберите файл
Файл не выбран
ADD
ENG
11:13
25.12.2017