Государственное образовательное учреждение высшего профессионального образования

«Московский государственный технический университет имени Н.Э. Баумана»

(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Системы обработки информации и управления

О т ч ё т п о л а б о р а т о р н о й р а б о т е

п о к у р с у

« Р а з р а б о т к а и н т е р н е т - п р и л о ж е н и й »

Авторизация, работа с формами и Django Admin

Исполнитель: студентка группы **РТ5-51**
Галичий Д.А.

Преподаватель: **Гапанюк Ю.Е.**

Москва, 2017

Цель работы: научиться обрабатывать веб-формы на стороне приложения, освоить инструменты, которые предоставляет Django по работе с формами. Также в этой лабораторной работе необходимо освоить инструменты Django по работе с авторизацией и реализовать простейшую авторизацию; познакомиться с инструментом администрирования Django.

Содержание файла «settings.py»:

```
"""
Django settings for lab6 project.

Generated by 'django-admin startproject' using Django 1.11.7.

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 = 'y2a(#+h0c6dui573k5s@9i@=d0f&13#*erc*i3y(@14&=h(*$m'

# 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',
    'lab6.apps.db_app.apps.DbAppConfig',
    #'lab6.apps.db_app',
]

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 = 'lab6.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'lab6/apps/db_app/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': 'films',
        'USER': 'dbuser',
        'PASSWORD': '123',
        'HOST': 'localhost',
        'PORT': 3306,
        '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.UserAttributeSimilarityValida
tor',
    },
    {
        '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 = 'ru-ru'

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/'
```

Содержание файла «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

urlpatterns = [
    url(r'^admin/', admin.site.urls),
    url(r'^db_app/', include('lab6.apps.db_app.urls')),
]
```

Содержание файла «db_app\urls.py»:

```
from django.conf.urls import url
from lab6.apps.db_app import views
from lab6.apps.db_app.views import FilmsList, ActorsList,
FilmmakersList, Film_writersList, ProducersList, CameramenList,
CountriesList


urlpatterns = [
    url(r'^main/', views.main, name='main_url'),
    url(r'^films/', FilmsList.as_view(), name='films_url'),
    url(r'^actors/', ActorsList.as_view(), name='actors_url'),
    url(r'^filmmakers/', FilmmakersList.as_view(),
name='filmmakers_url'),
    url(r'^film_writers/', Film_writersList.as_view(),
name='film_writers_url'),
    url(r'^producers/', ProducersList.as_view(),
name='producers_url'),
    url(r'^cameramen/', CameramenList.as_view(),
name='cameramen_url'),
    url(r'^countries/', CountriesList.as_view(),
name='countries_url'),
    url(r'^registration/', views.registration, name='reg_url'),
    url(r'^registration2/', views.registration2, name='reg2_url'),
    url(r'^login/', views.login, name='login_url'),
    url(r'^account/', views.account, name='account_url'),
    url(r'^logout/', views.logout, name='logout_url'),
]
```

Содержание файла «views.py»:

```
from django.shortcuts import render, HttpResponseRedirect,
render_to_response
from django.views.generic import ListView
from lab6.apps.db_app.models import *
from django.contrib.auth.models import User
from lab6.apps.db_app.registration import *
from django.contrib import auth
from django.contrib.auth import authenticate
from django.contrib.auth.decorators import login_required
# Create your views here.


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


class CountriesList(ListView):
    model = Countries
    template_name = "countries.html"
```

```
class ActorsList(ListView):
    model = Actors
    template_name = "actors.html"


class FilmmakersList(ListView):
    model = Filmmakers
    template_name = "filmmakers.html"


class Film_writersList(ListView):
    model = Film_writers
    template_name = "film_writers.html"


class ProducersList(ListView):
    model = Producers
    template_name = "producers.html"


class CameramenList(ListView):
    model = Cameramen
    template_name = "cameramen.html"


class FilmsList(ListView):
    model = Films
    template_name = "films.html"


def registration(request):
    errors = {'username': '', 'password': '', 'password2': '',
'email': '', 'surname': '', 'firstname': ''}
    error_flag = False
    if request.method == 'POST':
        username = request.POST.get('username')
        if not username:
            errors['username'] = 'Пожалуйста, введите логин.'
            error_flag = True
        elif len(username) < 5:
            errors['username'] = 'Длина логина должна превышать 5
символов.'
            error_flag = True
        elif User.objects.filter(username=username).exists():
            errors['username'] = 'Такой логин уже существует.'
            error_flag = True
        password = request.POST.get('password')
        if not password:
            errors['password'] = 'Пожалуйста, введите пароль.'
            error_flag = True
        elif len(password) < 8:
            errors['password'] = 'Длина пароля должна превышать 8
символов.'
        password_repeat = request.POST.get('password2')
        if password != password_repeat:
            errors['password2'] = 'Пароли должны совпадать.'
            error_flag = True
        email = request.POST.get('email')
        if not email:
            errors['email'] = 'Пожалуйста, введите e-mail.'
        surname = request.POST.get('surname')
        if not surname:
            errors['surname'] = 'Пожалуйста, введите фамилию.'
        firstname = request.POST.get('firstname')
        if not firstname:
            errors['firstname'] = 'Пожалуйста, введите имя.'
        if not error_flag:
            user = User.objects.create_user(username=username,
password=password, email=email,
                                            last_name=surname,
first_name=firstname)
            return HttpResponseRedirect('/db_app/login/')
    return render(request, 'registration.html', locals())


def login(request):
    error = ""
    if request.method == 'POST':
        username = request.POST.get('username')
        password = request.POST.get('password')
        user = authenticate(username=username, password=password)
        if user:
            auth.login(request, user)
            return HttpResponseRedirect('/db_app/account/')
        else:
            error = "Неверный логин или пароль."
    return render(request, 'login.html', locals())


@login_required()
def account(request):
    if not request.user.is_authenticated:
        return HttpResponseRedirect('/db_app/login/')
    else:
        return render(request, 'account.html', locals())


def logout(request):
    auth.logout(request)
    return render_to_response('logout.html')


def registration2(request):
    form = RegistrationForm(request.POST or None)
    if request.method == 'POST':
        if form.is_valid():
            user =
User.objects.create_user(username=request.POST.get('username'),

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

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

last_name=request.POST.get('surname'),
first_name=request.POST.get('firstname'))
            return HttpResponseRedirect('/db_app/login/')
        else:
            form = RegistrationForm()
    return render(request, 'registration2.html', {'form': form})
```

Содержание файла «models.py»:

```
from django.db import models

# Create your models here.


class Actors(models.Model):
    actor_id = models.AutoField(primary_key=True)
    actor_name = models.CharField(max_length=100)

    def __str__(self):
        return self.actor_name

    class Meta:
        verbose_name_plural = "Actors"
        verbose_name = "Actor"


class Filmmakers(models.Model):
    filmmaker_id = models.AutoField(primary_key=True)
    filmmaker_name = models.CharField(max_length=100)

    def __str__(self):
        return self.filmmaker_name

    class Meta:
        verbose_name_plural = "Filmmakers"
        verbose_name = "Filmmaker"


class Film_writers(models.Model):
    film_writer_id = models.AutoField(primary_key=True)
    film_writer_name = models.CharField(max_length=100)

    def __str__(self):
        return self.film_writer_name

    class Meta:
        verbose_name_plural = "Film_writers"
        verbose_name = "Film_writer"


class Producers(models.Model):
    producer_id = models.AutoField(primary_key=True)
    producer_name = models.CharField(max_length=100)

    def __str__(self):
        return self.producer_name

    class Meta:
        verbose_name_plural = "Producers"
        verbose_name = "Producer"


class Cameramen(models.Model):
    cameraman_id = models.AutoField(primary_key=True)
    cameraman_name = models.CharField(max_length=100)

    def __str__(self):
        return self.cameraman_name

    class Meta:
        verbose_name_plural = "Cameramen"
        verbose_name = "Cameraman"


class Countries(models.Model):
    country_id = models.AutoField(primary_key=True)
    country_name = models.CharField(max_length=100)

    def __str__(self):
        return self.country_name

    class Meta:
        verbose_name_plural = "Countries"
        verbose_name = "Country"


class Films(models.Model):
    film_id = models.AutoField(primary_key=True)
    film_name = models.CharField(max_length=100)
    release_date = models.DateField()
    in_the_lead_role = models.ManyToManyField(Actors)
    filmmaker = models.ForeignKey(Filmmakers,
on_delete=models.CASCADE)
    film_writer = models.ForeignKey(Film_writers,
on_delete=models.CASCADE)
    producer = models.ForeignKey(Producers, on_delete=models.CASCADE)
    cameraman = models.ForeignKey(Cameramen, on_delete=models.CASCADE)
    country = models.ForeignKey(Countries, on_delete=models.CASCADE)
    box_office_results = models.IntegerField()

    def __str__(self):
        return self.film_name

    def get_actors(self):
        return "\n".join([i.actor_name for i in
self.in_the_lead_role.all()])
    get_actors.short_description = 'In_the_lead_roles'

    class Meta:
        verbose_name_plural = "Films"
        verbose_name = "Film"
```

Содержание файла «connection.py»:

```
import MySQLdb


class Connection:
    def __init__(self, user, password, db, host='localhost',
charset='utf8'):
        #Параметры соединения
        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):
        #Открытие соединения
        if not self._connection:
            self._connection = MySQLdb.connect(
                host = self.host,
                user = self.user,
                passwd = self.password,
                db = self.db
            )

    def disconnect(self):
        #Закрытие соединения
        if self._connection:
            self._connection.close()


class Countries:

    def __init__(self, db_connection, country_name):
        #Сохранение соединения и данных
        self.db_connection = db_connection.connection
        self.country_name = country_name

    def save(self):
        #Запись данных из объекта в запись БД
        c = self.db_connection.cursor()
        c.execute("INSERT INTO db_app_countries (country_name) VALUES
(%s);", (self.country_name))
        self.db_connection.commit()
        c.close()


class Actors:
    def __init__(self, db_connection, actor_name):
        #Сохранение соединения и данных
        self.db_connection = db_connection.connection
        self.actor_name = actor_name

    def save(self):
        #Запись данных из объекта в запись БД
        c = self.db_connection.cursor()
        c.execute("INSERT INTO db_app_actors (actor_name) VALUES
(%s);", (self.actor_name))
        self.db_connection.commit()
        c.close()


class Filmmakers:

    def __init__(self, db_connection, filmmaker_name):
        #Сохранение соединения и данных
        self.db_connection = db_connection.connection
        self.filmmaker_name = filmmaker_name

    def save(self):
        #Запись данных из объекта в запись БД
        c = self.db_connection.cursor()
        c.execute("INSERT INTO db_app_filmmakers (filmmaker_name)
VALUES (%s);", (self.filmmaker_name))
        self.db_connection.commit()
        c.close()


class Film_writers:

    def __init__(self, db_connection, film_writer_name):
        #Сохранение соединения и данных
        self.db_connection = db_connection.connection
        self.film_writer_name = film_writer_name

    def save(self):
        #Запись данных из объекта в запись БД
        c = self.db_connection.cursor()
        c.execute("INSERT INTO db_app_film_writers (film_writer_name)
VALUES (%s);", (self.film_writer_name))
        self.db_connection.commit()
        c.close()


class Producers:

    def __init__(self, db_connection, producer_name):
        #Сохранение соединения и данных
        self.db_connection = db_connection.connection
        self.producer_name = producer_name

    def save(self):
        #Запись данных из объекта в запись БД
        c = self.db_connection.cursor()
        c.execute("INSERT INTO db_app_producers (producer_name) VALUES
(%s);", (self.producer_name))
```

```
        self.db_connection.commit()
        c.close()


class Cameramen:

    def __init__(self, db_connection, cameraman_name):
        #Сохранение соединения и данных
        self.db_connection = db_connection.connection
        self.cameraman_name = cameraman_name

    def save(self):
        #Запись данных из объекта в запись БД
        c = self.db_connection.cursor()
        c.execute("INSERT INTO db_app_cameramen (cameraman_name)
VALUES (%s);", (self.cameraman_name))
        self.db_connection.commit()
        c.close()


class Films:

    def __init__(self, db_connection, film_name, release_date,
filmmaker_id, film_writer_id, producer_id, cameraman_id, country_id,
box_office_results):
        #Сохранение соединения и данных
        self.db_connection = db_connection.connection
        self.film_name = film_name
        self.release_date = release_date
        self.filmmaker_id = filmmaker_id
        self.film_writer_id = film_writer_id
        self.producer_id = producer_id
        self.cameraman_id = cameraman_id
        self.country_id = country_id
        self.box_office_results = box_office_results

    def save(self):
        #Запись данных из объекта в запись БД
        c = self.db_connection.cursor()
        c.execute("INSERT INTO db_app_films (film_name, release_date,
filmmaker_id, film_writer_id, producer_id, cameraman_id, country_id,
box_office_results) VALUES (%s, %s, %s, %s, %s, %s, %s, %s);",
(self.film_name, self.release_date, self.filmmaker_id,
self.film_writer_id, self.producer_id, self.cameraman_id,
self.country_id, self.box_office_results))
        self.db_connection.commit()
        c.close()


class FilmsActors:
    def __init__(self, db_connection, film_id, actor_id):
        self.db_connection = db_connection.connection
        self.film_id = film_id
        self.actor_id = actor_id

    def save(self):
        c = self.db_connection.cursor()
        c.execute("INSERT INTO db_app_films_in_the_lead_role
(films_id, actors_id) VALUES (%s, %s);", (self.film_id,
self.actor_id))
        self.db_connection.commit()
        c.close()


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

with con:
    country = Countries(con, 'Россия')
    country.save()
    actor = Actors(con, 'Данила Козловский')
    actor.save()
    filmmaker = Filmmakers(con, 'Андрей Кравчук')
    filmmaker.save()
    film_writer = Film_writers(con, 'Андрей Рубанов')
    film_writer.save()
    producer = Producers(con, 'Константин Эрнст')
    producer.save()
    cameraman = Cameramen(con, 'Игорь Гринякин')
    cameraman.save()
    film = Films(con, 'Викинг', '2016.12.29', '1', '1', '1', '1', '1',
'27018393')
    film.save()
    film_actor = FilmsActors(con, '1', '1')
    film_actor.save()
```

Содержание файла «registration.py»:

```
from django import forms


class RegistrationForm(forms.Form):
    username = forms.CharField(min_length=5, label='Логин:')
    password = forms.CharField(min_length=6,
widget=forms.PasswordInput, label='Пароль:')
    password2 = forms.CharField(min_length=6,
widget=forms.PasswordInput, label='Повторите пароль:')
    email = forms.EmailField(widget=forms.EmailInput, label='E-mail:')
    surname = forms.CharField(label='Фамилия:')
    firstname = forms.CharField(label='Имя:')
```

Содержание файла «admin.py»:

```
from django.contrib import admin
from lab6.apps.db_app.models import *

# Register your models here.


class FilmsAdmin(admin.ModelAdmin):
    fields = ('film_name', 'release_date', 'in_the_lead_role',
'filmmaker', 'film_writer', 'producer', 'cameraman',
              'country', 'box_office_results')
    list_filter = ('release_date', 'country', ('in_the_lead_role',
admin.RelatedOnlyFieldListFilter),)
```

```
    list_display = ('film_name', 'release_date', 'get_actors',
'filmmaker', 'film_writer', 'producer',
                    'cameraman', 'country', 'box_office_results')
    search_fields = ('film_name',)
    list_per_page = 10


class ActorsAdmin(admin.ModelAdmin):
    list_per_page = 10


admin.site.register(Actors, ActorsAdmin)
admin.site.register(Films, FilmsAdmin)

admin.site.site_url = '/main/'
admin.site.site_header = 'Django-администрирование'
admin.site.index_title = 'Администрирование'
```

Содержание файла «main_page.html»:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    {% block title1 %}<title>Main page</title>{% endblock %}
    <link rel="stylesheet"
    href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-
beta.2/css/bootstrap.min.css"
    integrity="sha384-
PsH8R72JQ3SOdhVi3uxftmaW6Vc51MKb0q5P2rRUpPvrszuE4W1povHYgTpBfshb"
    crossorigin="anonymous">
    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
    integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
    crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.3/umd/poppe
r.min.js"
    integrity="sha384-
vFJXuSJphROIrBnz7yo7oB41mKfc8JzQZiCq4NCceLEaO4IHwicKwpJf9c9IpFgh"
    crossorigin="anonymous"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-
beta.2/js/bootstrap.min.js"
    integrity="sha384-
alpBpkh1PFOepccYVYDB4do5UnbKysX5WZXm3XxPqe5iKTfUKjNkCk9SaVuEZflJ"
    crossorigin="anonymous"></script>
    {% load static %}
    <link rel="stylesheet" href="{% static 'css/style.css' %}"
type="text/css">
</head>
<body>
<div id="container">
    {% block title2 %}
        <div id="header"><h1>___Basic tables___</h1></div>
    {% endblock %}
    {% block body1 %}
        <nav class="navbar navbar-expand-lg navbar-light">
```

```
            <a class="navbar-brand" href="{% url 'main_url' %}">Main
page</a>
            <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-
label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse"
id="navbarSupportedContent">
                <ul class="navbar-nav mr-auto">
                    <li class="nav-item active">
                        <a class="nav-link" href="{% url 'reg_url'
%}">Registration <span class="sr-only">(current)</span></a>
                    </li>
                    <li class="nav-item active">
                        <a class="nav-link" href="{% url 'reg2_url'
%}">Registration(forms)<span class="sr-only">(current)</span></a>
                    </li>
                    <li class="nav-item active">
                        <a class="nav-link" href="{% url 'login_url'
%}">Login <span class="sr-only">(current)</span></a>
                    </li>
                </ul>
            </div>
        </nav>
    <br>
    {% endblock %}
    {% block body %}
        <div class="container">
            <li><a href="{% url 'films_url' %}">Films</a></li>
            <li><a href="{% url 'actors_url' %}">Actors</a></li>
            <li><a href="{% url 'filmmakers_url'
%}">Filmmakers</a></li>
            <li><a href="{% url 'film_writers_url' %}">Film
writers</a></li>
            <li><a href="{% url 'producers_url' %}">Producers</a></li>
            <li><a href="{% url 'cameramen_url' %}">Cameramen</a></li>
            <li><a href="{% url 'countries_url' %}">Countries</a></li>
        </div>
    {% endblock %}
</div>
</body>
</html>
```

Содержание файла «countries.html»:

```
{% extends 'main_page.html' %}
{% block title1 %}<title>Countries</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Countries</h1></div>
{% endblock %}
{% block body %}
    <div class="container">
            <div class="row centered">
            <table>
                <tr>
```

```
                    <th>ID</th>
                    <th>Country</th>
                </tr>
                {% for country in object_list %}
                    <tr>
                        <td>{{ country.country_id }}</td>
                        <td>{{ country.country_name }}</td>
                    </tr>
                {% endfor %}
            </table>
            </div>
        </div>
{% endblock %}
```

Содержание файла «actors.html»:

```
{% extends 'main_page.html' %}
{% block title1 %}<title>Actors</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Actors</h1></div>
{% endblock %}
{% block body %}
    <div class="container">
            <div class="row centered">
            <table>
                <tr>
                    <th>ID</th>
                    <th>Actor</th>
                </tr>
                {% for actor in object_list %}
                    <tr>
                        <td>{{ actor.actor_id }}</td>
                        <td>{{ actor.actor_name }}</td>
                    </tr>
                {% endfor %}
            </table>
            </div>
        </div>
{% endblock %}
```

Содержание файла «filmmakers.html»:

```
{% extends 'main_page.html' %}
{% block title1 %}<title>Filmmakers</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Filmmakers</h1></div>
{% endblock %}
{% block body %}
    <div class="container">
            <div class="row centered">
            <table>
                <tr>
                    <th>ID</th>
                    <th>Filmmaker</th>
                </tr>
                {% for filmmaker in object_list %}
```

```
                    <tr>
                        <td>{{ filmmaker.filmmaker_id }}</td>
                        <td>{{ filmmaker.filmmaker_name }}</td>
                    </tr>
                {% endfor %}
            </table>
            </div>
          </div>
{% endblock %}
```

Содержание файла «film_writers.html»:

```
{% extends 'main_page.html' %}
{% block title1 %}<title>Film writers</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Film writers</h1></div>
{% endblock %}
{% block body %}
    <div class="container">
            <div class="row centered">
            <table>
                <tr>
                    <th>ID</th>
                    <th>Film writer</th>
                </tr>
                {% for film_writer in object_list %}
                    <tr>
                        <td>{{ film_writer.film_writer_id }}</td>
                        <td>{{ film_writer.film_writer_name }}</td>
                    </tr>
                {% endfor %}
            </table>
            </div>
          </div>
{% endblock %}
```

Содержание файла «producers.html»:

```
{% extends 'main_page.html' %}
{% block title1 %}<title>Producers</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Producers</h1></div>
{% endblock %}
{% block body %}
    <div class="container">
            <div class="row centered">
            <table>
                <tr>
                    <th>ID</th>
                    <th>Producer</th>
                </tr>
                {% for producer in object_list %}
                    <tr>
                        <td>{{ producer.producer_id }}</td>
                        <td>{{ producer.producer_name }}</td>
                    </tr>
                {% endfor %}
            </table>
            </div>
        </div>
{% endblock %}
```

Содержание файла «cameramen.html»:

```
{% extends 'main_page.html' %}
{% block title1 %}<title>Cameramen</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Cameramen</h1></div>
{% endblock %}
{% block body %}
    <div class="container">
            <div class="row centered">
            <table>
                <tr>
                    <th>ID</th>
                    <th>Cameraman</th>
                </tr>
                {% for cameraman in object_list %}
                    <tr>
                        <td>{{ cameraman.cameraman_id }}</td>
                        <td>{{ cameraman.cameraman_name }}</td>
                    </tr>
                {% endfor %}
            </table>
            </div>
        </div>
{% endblock %}
```

Содержание файла «films.html»:

```
{% extends 'main_page.html' %}
{% block title1 %}<title>Films</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Films</h1></div>
{% endblock %}
{% block body %}
    <div class="container">
            <div class="row centered">
            <table>
                <tr>
                    <th>ID</th>
                    <th>Film</th>
                    <th>Release date</th>
                    <th>Filmmaker</th>
                    <th>Film writer</th>
                    <th>Producer</th>
                    <th>Cameraman</th>
                    <th>Country</th>
                    <th>Box office results, $</th>
                </tr>
                {% for film in object_list %}
                    <tr>
```

```
                            <td>{{ film.film_id }}</td>
                            <td>{{ film.film_name }}</td>
                            <td>{{ film.release_date }}</td>
                            <td>{{ film.filmmaker.filmmaker_name }}</td>
                            <td>{{ film.film_writer.film_writer_name
}}</td>
                            <td>{{ film.producer.producer_name }}</td>
                            <td>{{ film.cameraman.cameraman_name }}</td>
                            <td>{{ film.country.country_name }}</td>
                            <td>{{ film.box_office_results }}</td>
                        </tr>
                    {% endfor %}
                </table>
                </div>
            </div>
{% endblock %}
```

Содержание файла «account.html»:

```
{% extends 'main_page.html' %}
{% block title1 %}<title>Account</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Account</h1></div>
{% endblock %}
{% block body %}
    <div class="container">
        <h3>You are now logged in.</h3>
        <p>Click <a href="{% url 'logout_url' %}">here</a> to
logout.</p>
    </div>
    <br>
{% endblock %}
```

Содержание файла «login.html»:

```
{% extends 'main_page.html' %}
{% block title1 %}<title>Login</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Login</h1></div>
{% endblock %}
{% block body %}
    <form method= "POST">
        <div class="auth_block">
            {% csrf_token %}
            <div class="auth_blank">
                <div class="form-group">
                    <label>
                        Логин:
                        <input type= "text" name= "username">
                    </label>
                </div>
            </div>
            <div class="auth_blank">
                <div class="form-group">
                    <label>
                        Пароль:
```

```
                        <input type= "password" name= "password">
                    </label>
                </div>
            </div>
            {% if error != "" %}
                <p>{{ error }}</p>
            {% endif %}
            <button type="submit">Войти</button>
            <br><br><br>
        </div>
    </form>
{% endblock %}
```

Содержание файла «logout.html»:

```
{% extends 'main_page.html' %}
{% block title1 %}<title>Logout</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Logout</h1></div>
{% endblock %}
{% block body %}
    <div class="container">
        <h3>You are now logged out.</h3>
    </div>
    <br>
{% endblock %}
```

Содержание файла «registration.html»:

```
{% extends 'main_page.html' %}
{% block title1 %}<title>Registration</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Registration</h1></div>
{% endblock %}
{% block body %}
    <form method= "POST">
        <div class="auth_block">
            {% csrf_token %}
            <div class="auth_blank">
                <div class="form-group">
                    <label>
                        Логин:
                        <input type= "text" name= "username" value="{{
username }}">
                        {% if errors.username != '' %}
                            <div class="form-error">{{ errors.username
}}</div>
                        {% endif %}
                    </label>
                </div>
            </div>
            <div class="auth_blank">
                <div class="form-group">
                    <label>
                        Пароль:
                        <input type= "password" name= "password">
```

```
                            {% if errors.password != '' %}
                                <div class="form-error">{{ errors.password
}}</div>
                            {% endif %}
                        </label>
                    </div>
                </div>
                <div class="auth_blank">
                    <div class="form-group">
                        <label>
                            Повторите пароль:
                            <input type= "password" name= "password2">
                            {% if errors.password2 != '' %}
                                <div class="form-error">{{
errors.password2 }}</div>
                            {% endif %}
                        </label>
                    </div>
                </div>
                <div class="auth_blank">
                    <div class="form-group">
                        <label>
                            Email:
                            <input type= "email" name= "email" value="{{
email }}">
                            {% if errors.email != '' %}
                                <div class="form-error">{{ errors.email
}}</div>
                            {% endif %}
                        </label>
                    </div>
                </div>
                <div class="auth_blank">
                    <div class="form-group">
                        <label>
                            Фамилия:
                            <input type= "text" name= "surname">
                            {% if errors.surname != '' %}
                                <div class="form-error">{{ errors.surname
}}</div>
                            {% endif %}
                        </label>
                    </div>
                </div>
                <div class="auth_blank">
                    <div class="form-group">
                        <label>
                            Имя:
                            <input type= "text" name= "firstname">
                            {% if errors.firstname != '' %}
                                <div class="form-error">{{
errors.firstname }}</div>
                            {% endif %}
                        </label>
                    </div>
                </div>
                <button type= "submit"> Зарегистрироваться </button>
            <br><br><br>
```

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

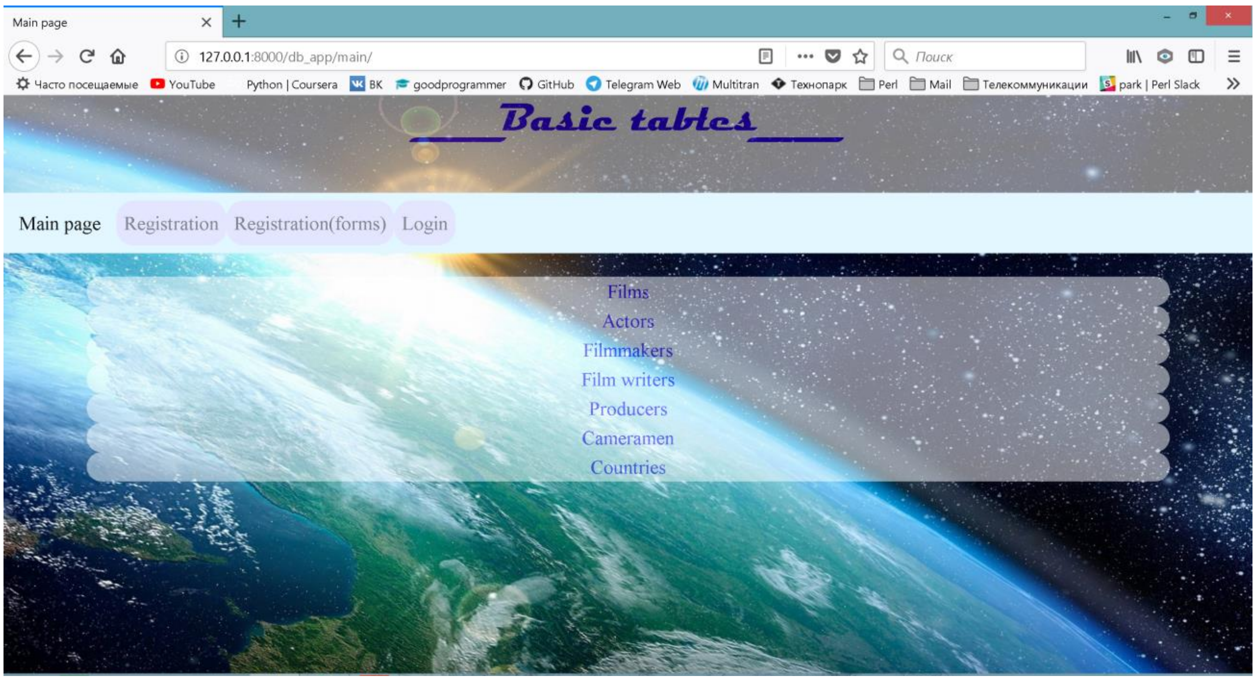
Содержание файла «registration2.html»:

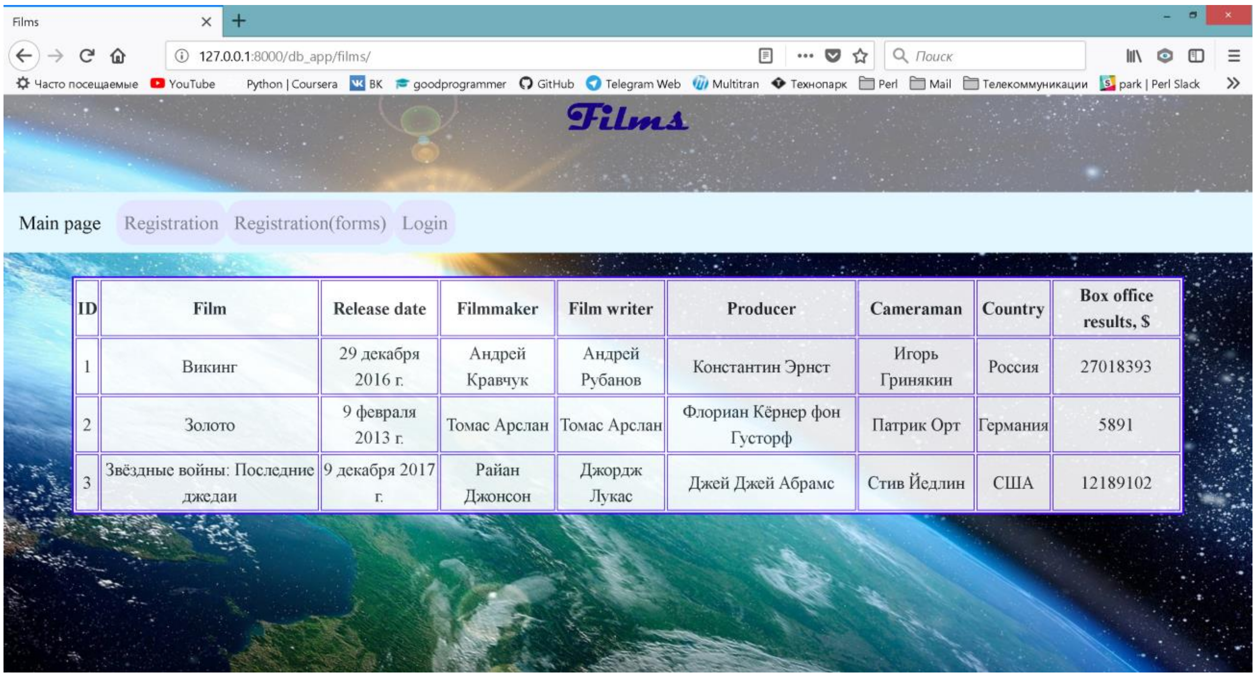
```
{% extends 'main_page.html' %}
{% block title1 %}<title>Registration</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Registration</h1></div>
{% endblock %}
{% block body %}
    <form method= "POST">
        <div class="auth_block">
            {% csrf_token %}
            {{ form.as_p }}
            <button type= "submit"> Зарегистрироваться </button>
        </div>
    <br><br>
    </form>
{% endblock %}
```

Содержание файла «style.css»:

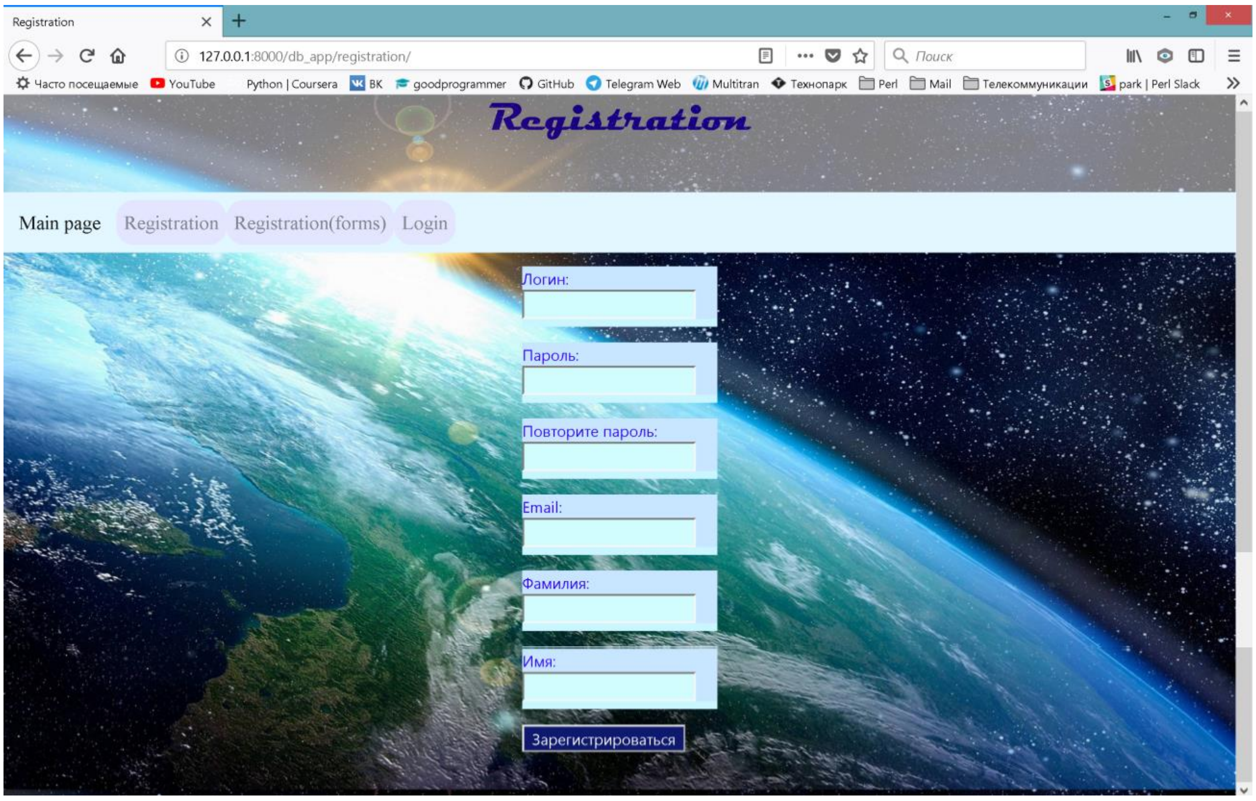
```
body {
    background-image: url("../imgs/bckg.jpg");
    background-size: cover;
}

#header {
    background: white;
    width: 100%;
    height: 100px;
    line-height:100px;
    opacity: 0.5;
    color: #3100ff;
    font-family: Magneto;
    font-size: 38px;
    text-align: center;
}

li {
    background: white;
    opacity: 0.5;
    color: white;
    list-style-type: none;
    link: black;
    border-radius: 18px;
    width: 100%;
    margin: 0 auto;
    text-align:  center;
}

a {
    color: #3100ff;
    text-decoration:none;
    font-family: "Times New Roman";
    font-size: 20px;
}

a:hover {
    color: #ad00ff;
    text-decoration:none
}

table {
    background-color: white;
    opacity: 0.9;
    border: 2px solid #4500ff;
    width: 100%;
   height:100px;
    font-family: "Times New Roman";
    font-size: 18px;
    text-align: center;
    border-collapse:separate;
}

th, td {
    border: 1px solid #4500ff;
}

.navbar {
    background-color: #e2f7ff;
}

.nav-item {
    background-color: #e6d5ff;
}

form {
    position: relative;
    top: 10%;
    left: 50%;
    width: 200px;
    height:100px;
    margin: -10px 0 0 -100px;
}

.form-group {
    background-color: #caf7ff;
}

input {
    background-color: #d1fdff;
}

button {
    background-color: #0f1a6f;
    color: white;
}

label {
    background-color: #c8e6ff;
    font-size: 16px;
    color: #3100ff;
}

h3 {
    color: #0f1a6f;
}
```

Результат работы программы:





ID	Film	Release date	Filmmaker	Film writer	Producer	Cameraman	Country	Box office results, $
1	Викинг	29 декабря 2016 г.	Андрей Кравчук	Андрей Рубанов	Константин Эрнст	Игорь Гринякин	Россия	27018393
2	Золото	9 февраля 2013 г.	Томас Арслан	Томас Арслан	Флориан Кёрнер фон Густорф	Патрик Орт	Германия	5891
3	Звёздные войны: Последние джедаи	9 декабря 2017 г.	Райан Джонсон	Джордж Лукас	Джей Джей Абрамс	Стив Йедлин	США	12189102



Registration

127.0.0.1:8000/db_app/registration/

Registration

Main page Registration Registration(forms) Login

Логин:

Пароль:

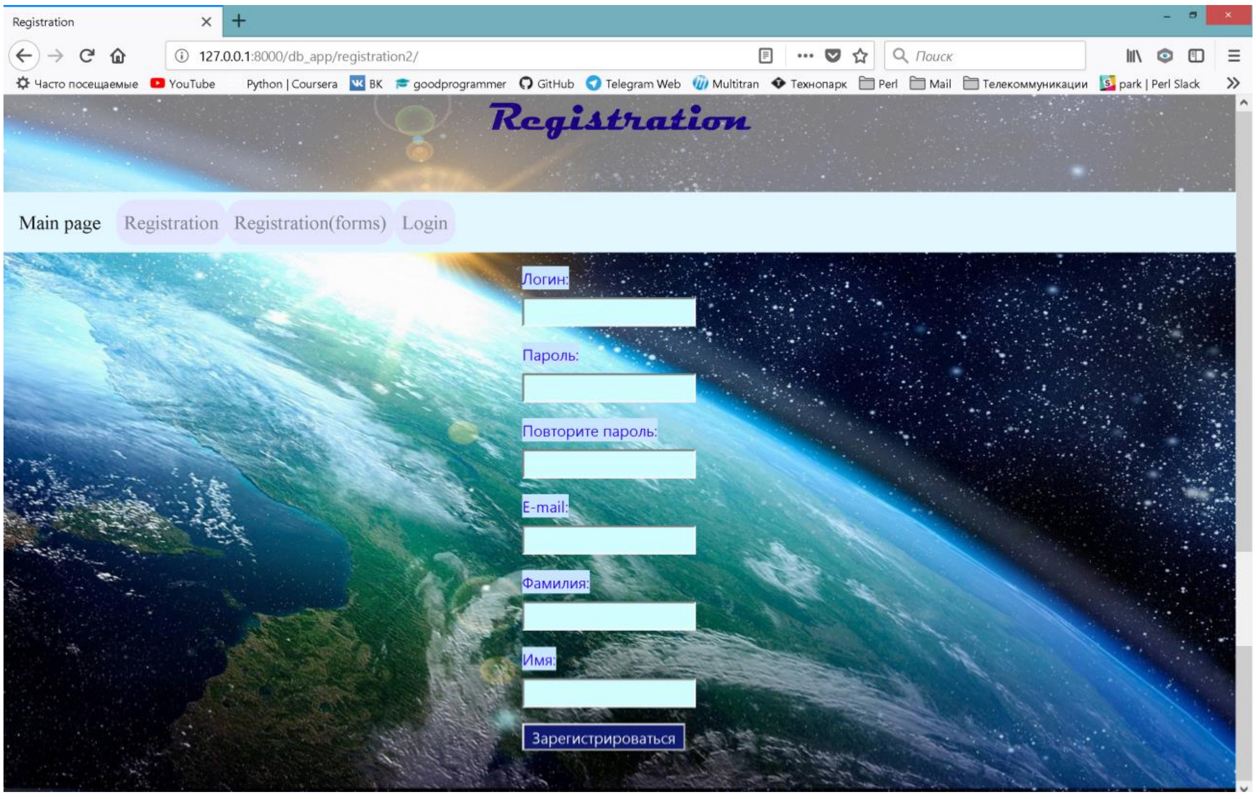
Повторите пароль:

Email:

Фамилия:

Имя:

Зарегистрироваться



Registration

127.0.0.1:8000/db_app/registration2/

Registration

Main page Registration Registration(forms) Login

Логин:

Пароль:

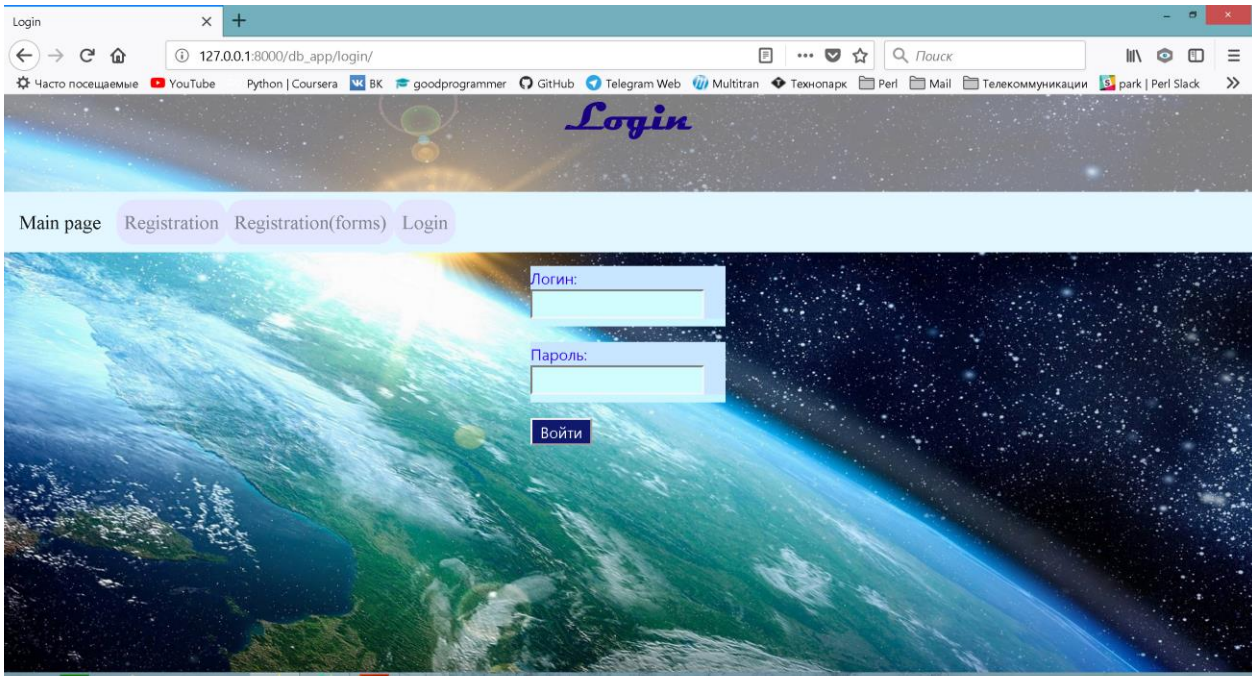
Повторите пароль:

E-mail:

Фамилия:

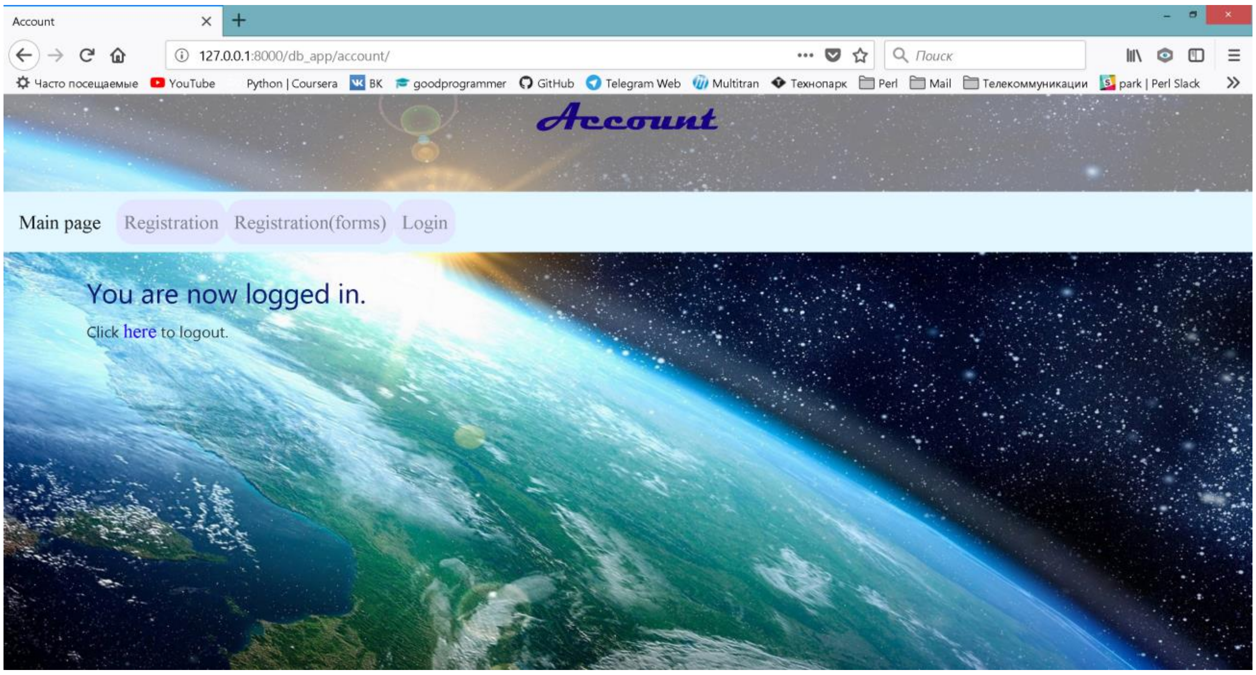
Имя:

Зарегистрироваться



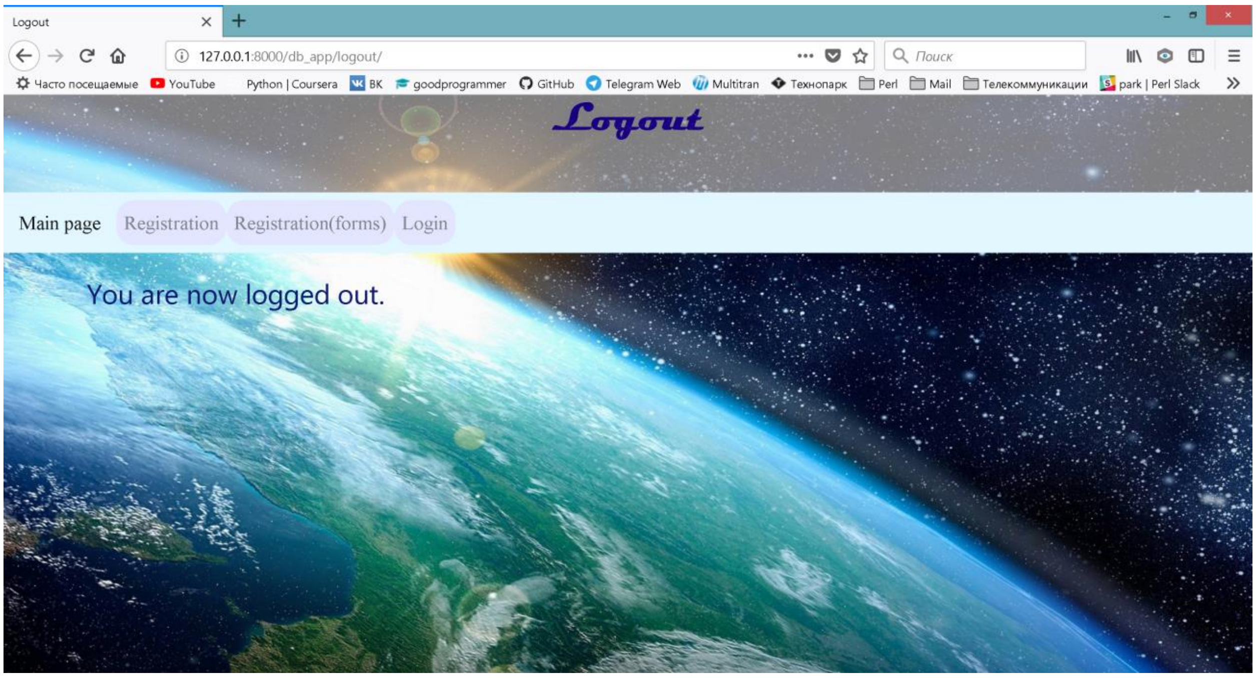
Логин:

Пароль:



You are now logged in.

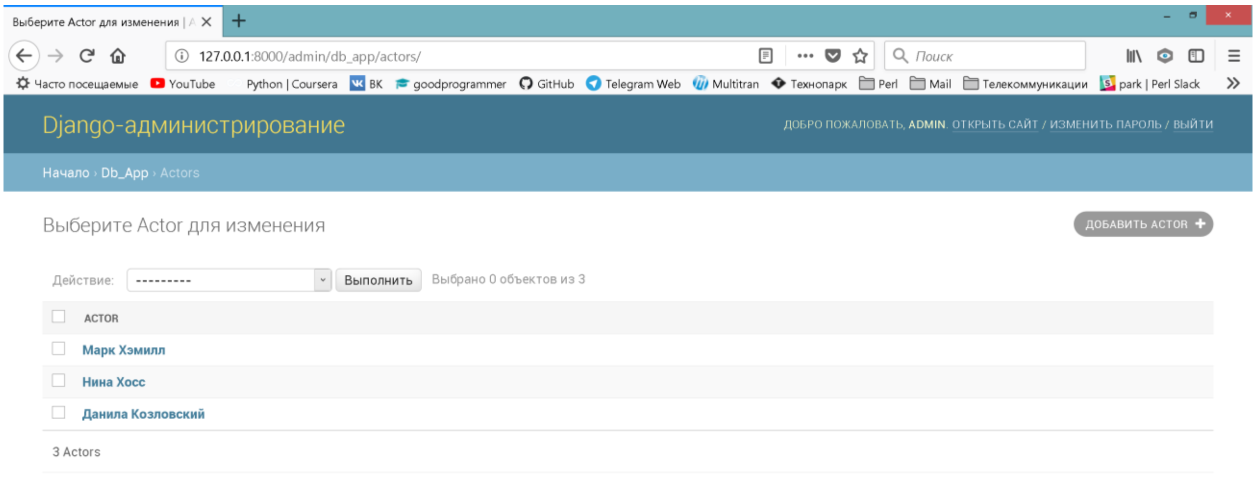
Click here to logout.


Logout

127.0.0.1:8000/db_app/logout/

Main page | Registration | Registration(forms) | Login

You are now logged out.


Выберите Actor для изменения

127.0.0.1:8000/admin/db_app/actors/

Django-администрирование

ДОБРО ПОЖАЛОВАТЬ, ADMIN. ОТКРЫТЬ САЙТ / ИЗМЕНИТЬ ПАРОЛЬ / ВЫЙТИ

Начало › Db_App › Actors

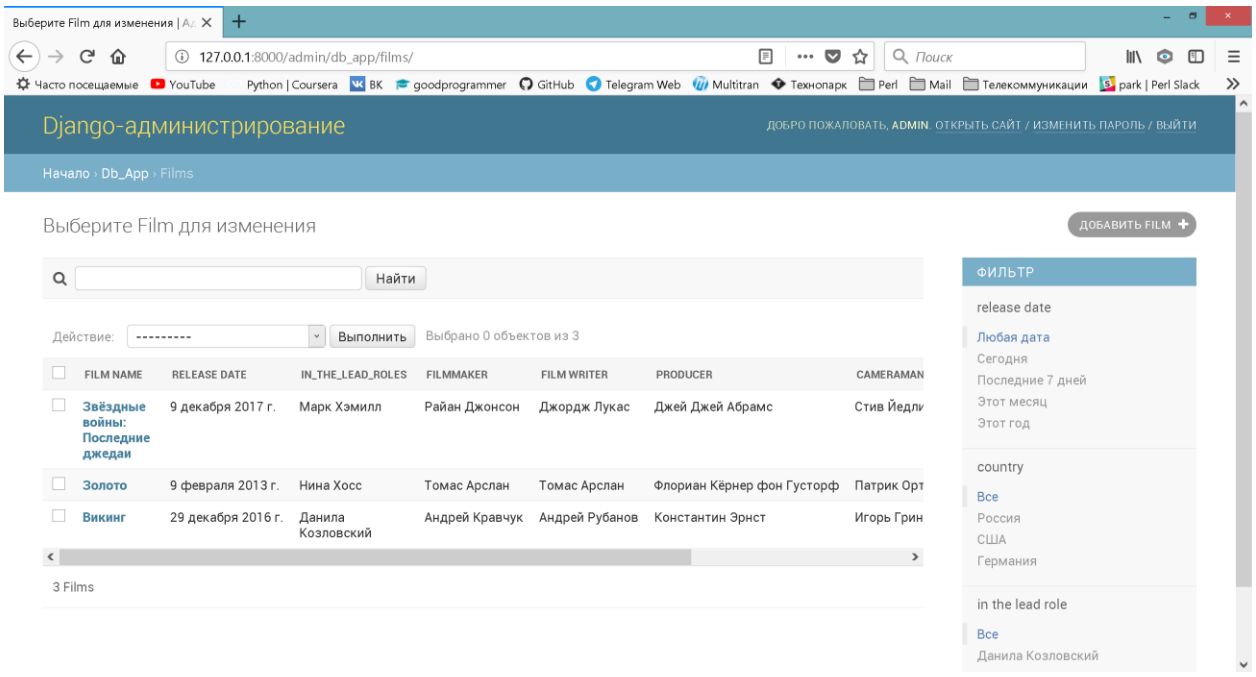
Выберите Actor для изменения

ДОБАВИТЬ ACTOR

Действие: --------- Выполнить Выбрано 0 объектов из 3

ACTOR
Марк Хэмилл
Нина Хосс
Данила Козловский

3 Actors



127.0.0.1:8000/admin/db_app/films/

Django-администрирование

ДОБРО ПОЖАЛОВАТЬ, ADMIN. ОТКРЫТЬ САЙТ / ИЗМЕНИТЬ ПАРОЛЬ / ВЫЙТИ

Начало › Db_App › Films

Выберите Film для изменения

ДОБАВИТЬ FILM

Найти

Действие: --------- Выполнить Выбрано 0 объектов из 3

FILM NAME	RELEASE DATE	IN_THE_LEAD_ROLES	FILMMAKER	FILM WRITER	PRODUCER	CAMERAMAN
Звёздные войны: Последние джедаи	9 декабря 2017 г.	Марк Хэмилл	Райан Джонсон	Джордж Лукас	Джей Джей Абрамс	Стив Йедли
Золото	9 февраля 2013 г.	Нина Хосс	Томас Арслан	Томас Арслан	Флориан Кёрнер фон Густорф	Патрик Орт
Викинг	29 декабря 2016 г.	Данила Козловский	Андрей Кравчук	Андрей Рубанов	Константин Эрнст	Игорь Грин

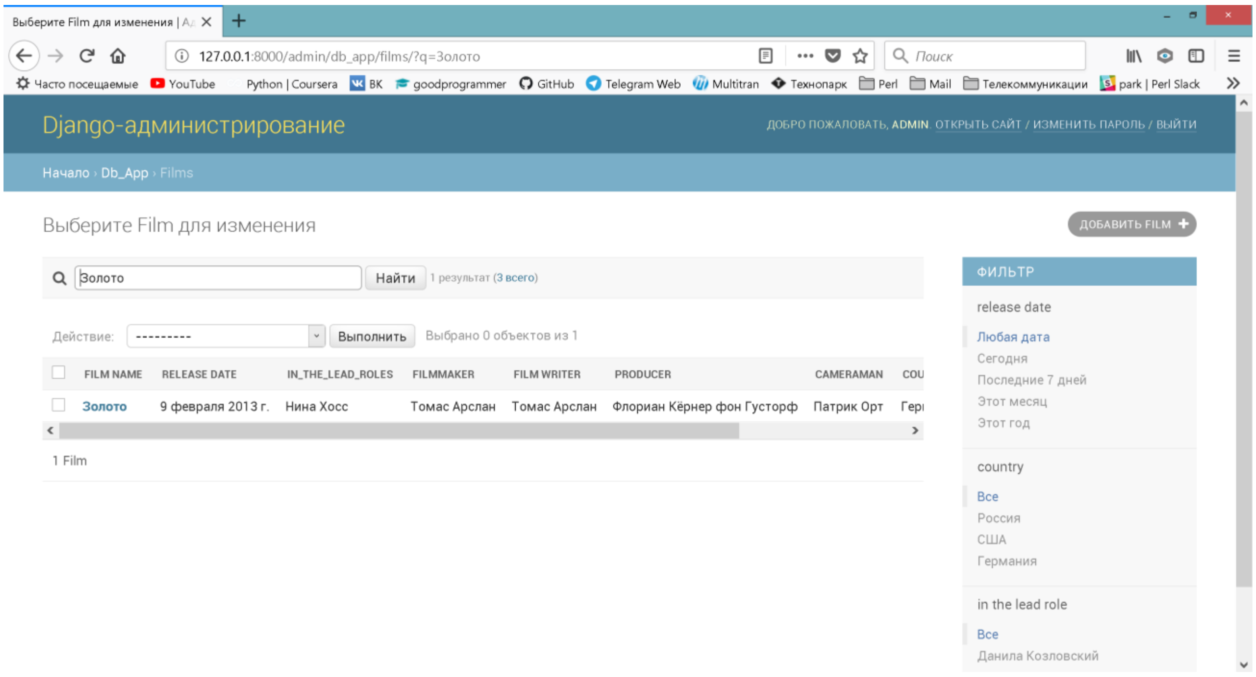
3 Films

ФИЛЬТР

release date
- Любая дата
- Сегодня
- Последние 7 дней
- Этот месяц
- Этот год

country
- Все
- Россия
- США
- Германия

in the lead role
- Все
- Данила Козловский



127.0.0.1:8000/admin/db_app/films/?q=Золото

Django-администрирование

ДОБРО ПОЖАЛОВАТЬ, ADMIN. ОТКРЫТЬ САЙТ / ИЗМЕНИТЬ ПАРОЛЬ / ВЫЙТИ

Начало › Db_App › Films

Выберите Film для изменения

ДОБАВИТЬ FILM

Золото Найти 1 результат (3 всего)

Действие: --------- Выполнить Выбрано 0 объектов из 1

FILM NAME	RELEASE DATE	IN_THE_LEAD_ROLES	FILMMAKER	FILM WRITER	PRODUCER	CAMERAMAN	COU
Золото	9 февраля 2013 г.	Нина Хосс	Томас Арслан	Томас Арслан	Флориан Кёрнер фон Густорф	Патрик Орт	Гер

1 Film

ФИЛЬТР

release date
- Любая дата
- Сегодня
- Последние 7 дней
- Этот месяц
- Этот год

country
- Все
- Россия
- США
- Германия

in the lead role
- Все
- Данила Козловский