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 »

Работа с СУБД

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

Преподаватель: **Гапанюк Ю.Е.**

Москва, 2017

Цель работы: знакомство с СУБД MySQL; создание базы данных; дополнение классов предметной области с помощью их связи с созданной базой; создание моделей с помощью Django ORM, отображение объектов из БД с помощью этих моделей и ClassBasedViews.

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'),
]
```

Содержание файла «views.py»:

```
from django.shortcuts import render
from django.views.generic import ListView
from lab6.apps.db_app.models import Actors, Filmmakers, Film_writers,
Producers, Cameramen, Countries, Films

# 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"
```

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)


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


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


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


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


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

```
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()
```

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 «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">
    {% 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 %}
    <br>
    {% 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>
    <div class="container"><a href="{% url 'main_url' %}">Main
page</a></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>
    <div class="container"><a href="{% url 'main_url' %}">Main
page</a></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>
    <div class="container"><a href="{% url 'main_url' %}">Main
page</a></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>
    <div class="container"><a href="{% url 'main_url' %}">Main
page</a></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>
    <div class="container"><a href="{% url 'main_url' %}">Main
page</a></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>
    <div class="container"><a href="{% url 'main_url' %}">Main
page</a></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»:

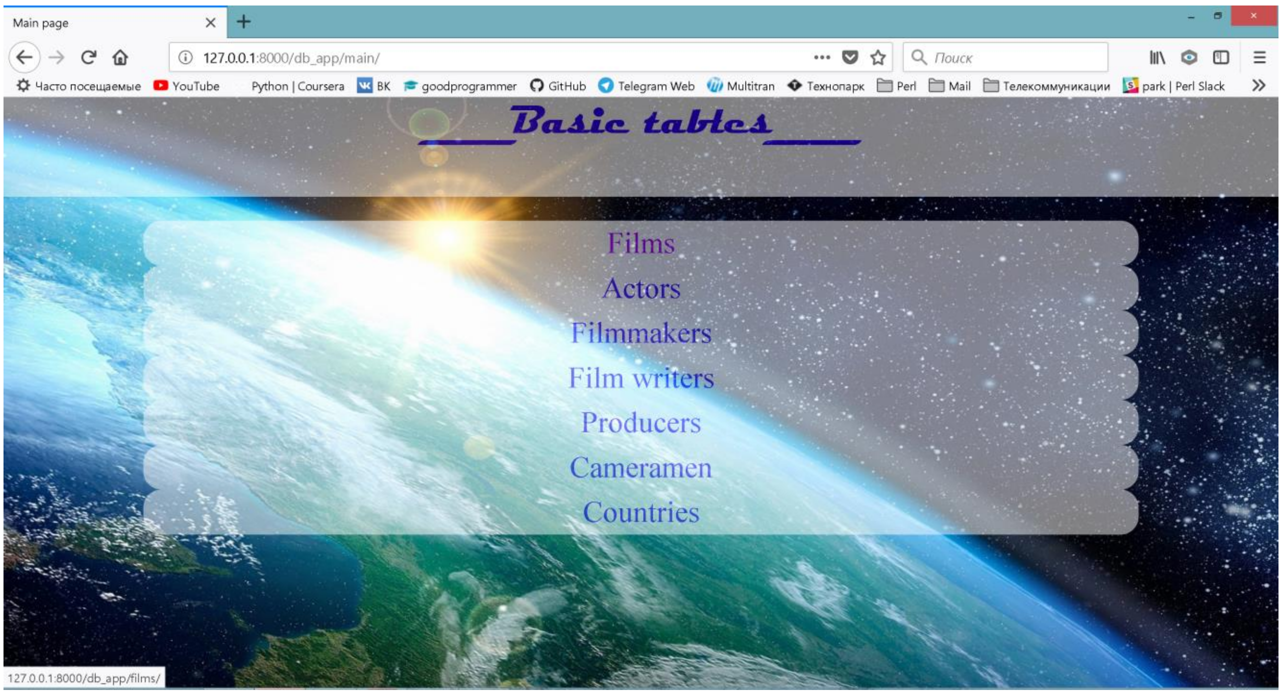
```
{% extends 'main_page.html' %}
{% block title1 %}<title>Films</title>{% endblock %}
{% block title2 %}
    <div id="header"><h1>Films</h1></div>
    <div class="container"><a href="{% url 'main_url' %}">Main
page</a></div>
{% endblock %}
{% block body %}
    <div class="container">
            <div 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 «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: 90%;
    margin: 0 auto;
    text-align:  center;
}

a {
    color: #3100ff;
    text-decoration:none;
    font-family: "Times New Roman";
    font-size: 30px;
}

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

table {
    background-color: white;
```

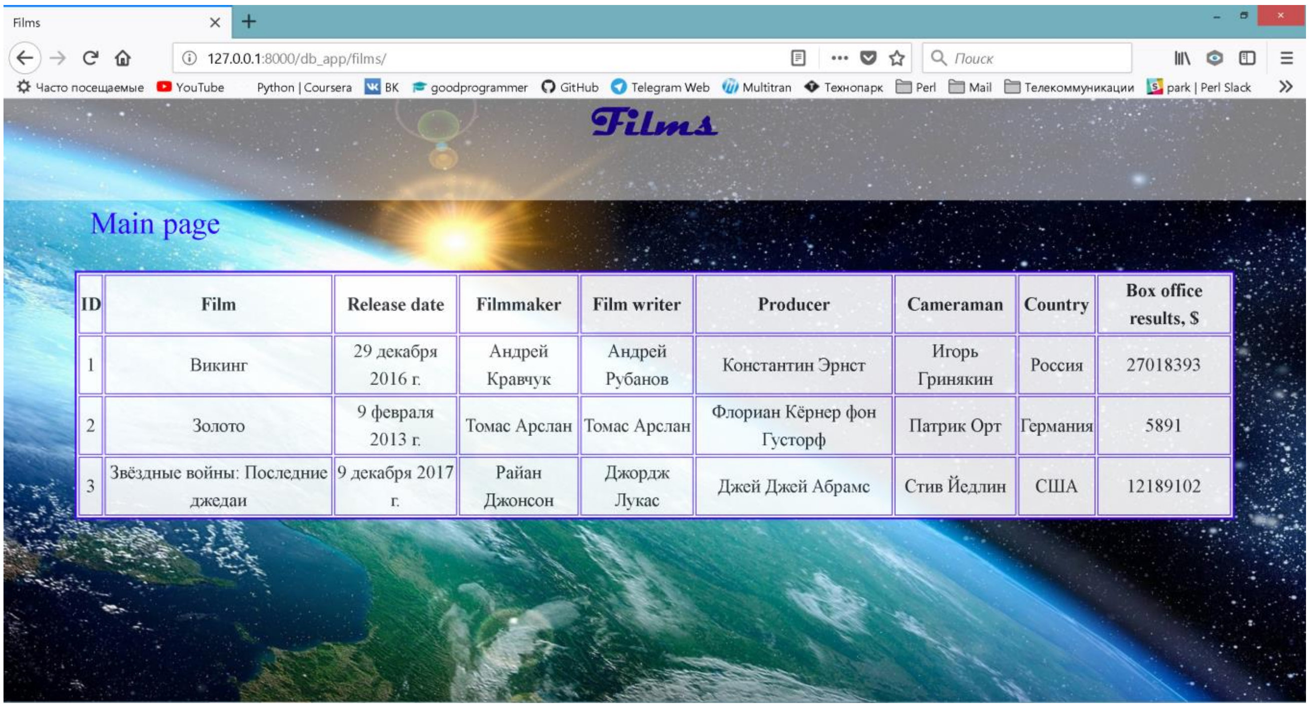
```
        opacity: 0.9;
        border: 2px solid #4500ff;
        width: 100%;
      height:100px;
        font-family: "Times New Roman";
        font-size: 18px;
        text-align: center;
        border-collapse:separate;
}

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

Результат работы программы:



Basic tables

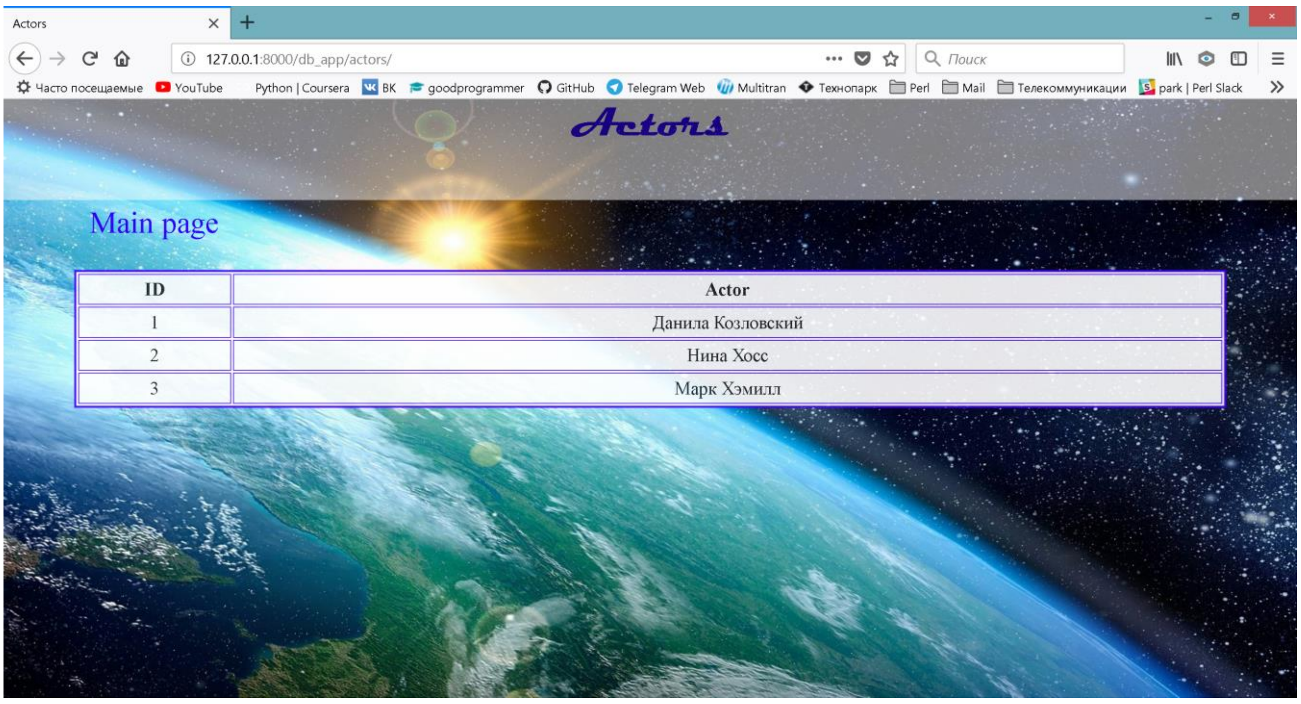
Films
Actors
Filmmakers
Film writers
Producers
Cameramen
Countries



Films

Main page

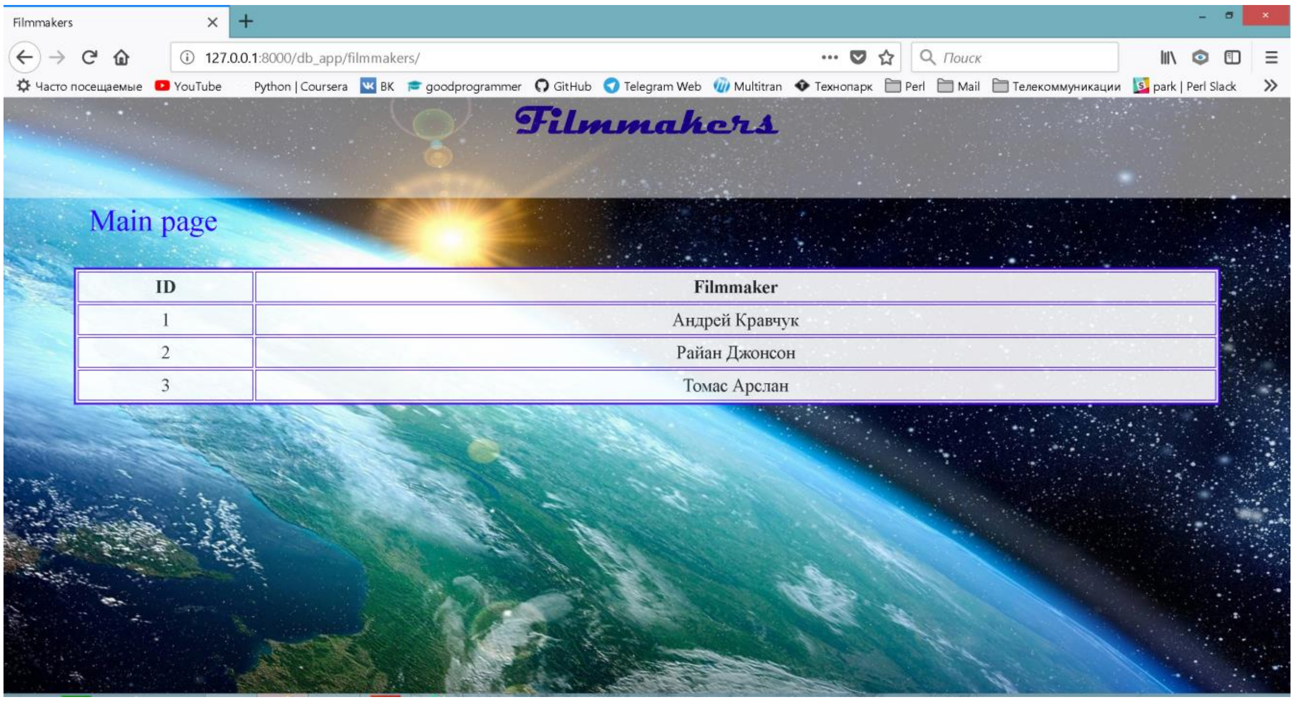
ID	Film	Release date	Filmmaker	Film writer	Producer	Cameraman	Country	Box office results, $
1	Викинг	29 декабря 2016 г.	Андрей Кравчук	Андрей Рубанов	Константин Эрнст	Игорь Гринякин	Россия	27018393
2	Золото	9 февраля 2013 г.	Томас Арслан	Томас Арслан	Флориан Кёрнер фон Густорф	Патрик Орт	Германия	5891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



Actors

Main page

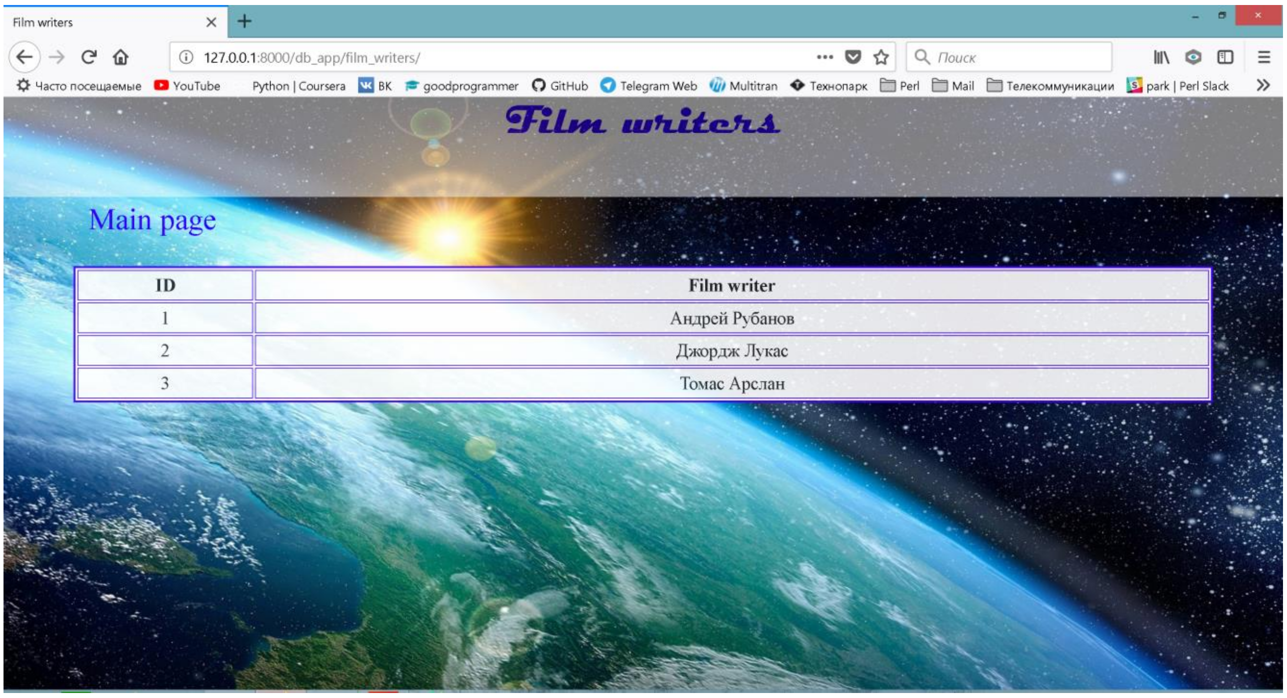
ID	Actor
1	Данила Козловский
2	Нина Хосс
3	Марк Хэмилл



Filmmakers

Main page

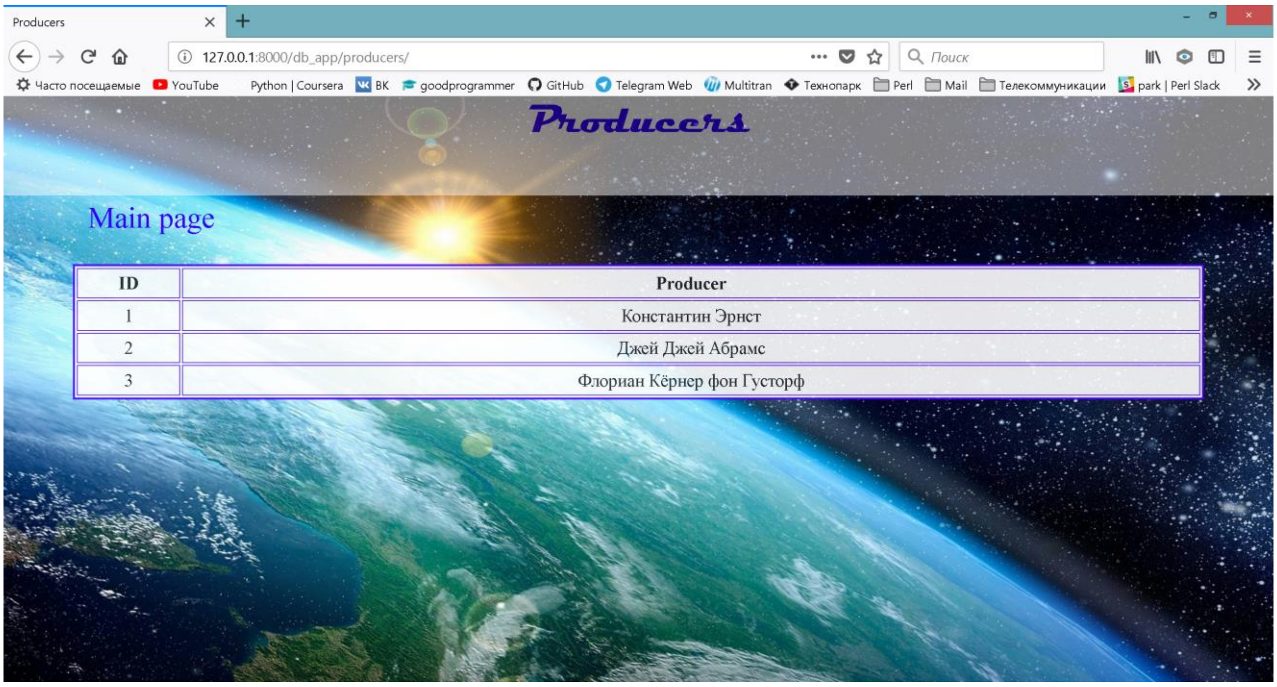
ID	Filmmaker
1	Андрей Кравчук
2	Райан Джонсон
3	Томас Арслан



Film writers

Main page

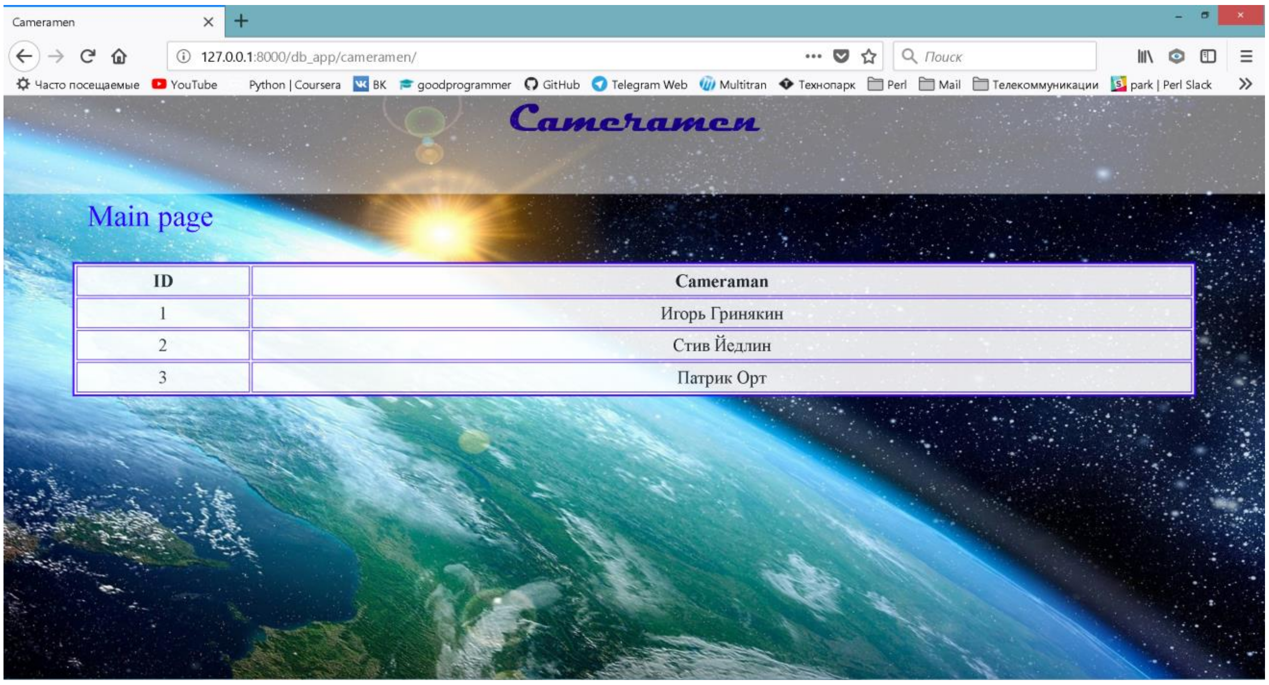
ID	Film writer
1	Андрей Рубанов
2	Джордж Лукас
3	Томас Арслан



Producers

Main page

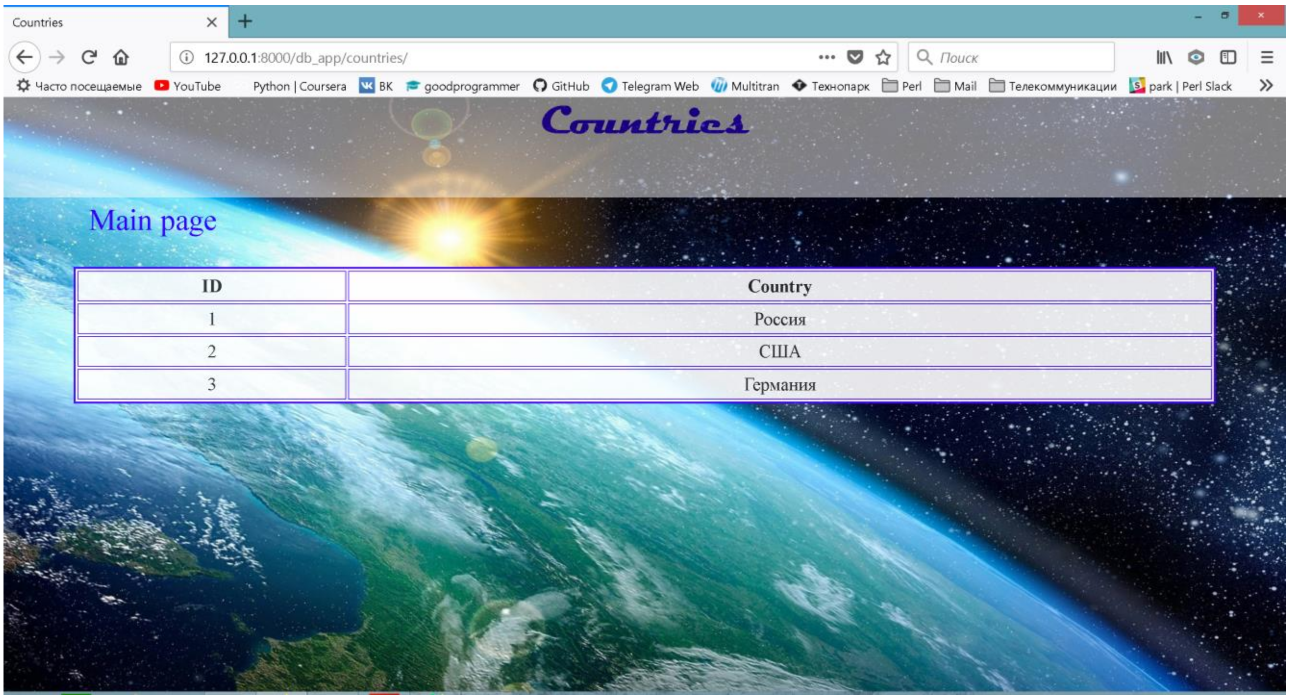
ID	Producer
1	Константин Эрнст
2	Джей Джей Абрамс
3	Флориан Кёрнер фон Густорф



Cameramen

Main page

ID	Cameraman
1	Игорь Гринякин
2	Стив Йедлин
3	Патрик Орт



Countries

Main page

ID	Country
1	Россия
2	США
3	Германия