

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии (ИУ7)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.04.04 Программная инженерия

Методические указания к лабораторным работам по курсу

ПРОЕКТИРОВАНИЕ РЕКОМЕНДАТЕЛЬНЫХ СИСТЕМ

Москва 2022

Оглавление

1	Лабораторная работа 1.1. Сборка и запуск учебного веб-сайта	3
1.1	Задание	3
1.2	Учебный веб-сайт	3
1.2.1	Набор данных	3
1.2.2	Серверная часть	3
1.2.3	Клиентская часть	4
1.3	Настройка окружения	4
1.4	Запуск и проверка работоспособности	4
2	Лабораторная работа 1.2. Неперсонализированные рекомендации	5
2.1	Задание	5
2.2	Неперсонализированные рекомендации	5
2.3	Доработка серверной части	5
2.3.1	Отображение новых фильмов	5
2.3.2	Отображение самых популярных фильмов	5
2.4	Доработка клиентской части	6

Введение

Глава 1

Лабораторная работа 1.1. Сборка и запуск учебного веб-сайта

Цель: подготовка рабочего места для лабораторных работ 1.2-1.14.

1.1 Задание

Скачать исходный код учебного веб-сайта из репозитория курса , изучить его, выполнить настройку окружения в соответствии с инструкцией, запустить учебный веб-сайт и проверить работоспособность.

1.2 Учебный веб-сайт

1.2.1 Набор данных

В лабораторных работах будут использованы общедоступные данные веб-сайта MovieLens (<https://movielens.org>), собранные и опубликованные компанией GroupLens Research: <https://grouplens.org/datasets/movielens>. Компания предоставила несколько наборов данных, для учебного веб-сайта выбран набор ml-latest-small.zip (<https://files.grouplens.org/datasets/movielens/ml-latest-small.zip> от 9/2018, содержащий 100 000 оценок и 3 600 тегов, примененных к 9 000 фильмам 600 пользователями.

Набор состоит из следующих файлов:

- movies.csv – фильмы, для каждого указаны идентификатор (movieId), название и год выхода (title), список жанров (genres);
- ratings.csv – оценки, для каждой указаны идентификатор пользователя (userId), идентификатор фильма (movieId), значение от 0.5 до 5 (rating), отметка времени добавления (timestamp);
- tags.csv – теги, для каждого указан идентификатор пользователя (userId), идентификатор фильма (movieId), значение (tag), отметка времени добавления (timestamp);
- links.csv – соответствие идентификаторов фильмов на веб-сайте MovieLens идентификаторам на других ресурсах (<http://www.imdb.com>, <https://www.themoviedb.org>), в лабораторных работах использоваться не будет.

1.2.2 Серверная часть

Серверная часть написана на языке Python с использованием фреймворка Flask и состоит из следующих файлов:

- `model.py` – логика запросов к базе данных (чтение и запись);
- `model_helpers.py` – вспомогательные функции для файла `model.py`, такие как установка и закрытие соединения с базой данных, преобразование объектов `sqlite3.Row` в словари;
- `api.py` – API, каждая функция API вызывает соответствующую функцию из файла `model.py` для чтения или записи данных в базу данных.

В качестве СУБД используется SQLite, реализованная встроенным в Python модулем.

1.2.3 Клиентская часть

Про реакт (?)

1.3 Настройка окружения

1. Установить Python 3 с официального сайта: <https://www.python.org/downloads/>
2. Установить IDE (опционально), например, PyCharm: <https://www.jetbrains.com/pycharm/download/>
3. Установить ???
4. Установить HTTP-клиент для тестирования API, например, Postman: <https://www.postman.com/downloads/>
5. Перейти в директорию с исходным кодом.
6. Установить необходимые пакеты Python (Flask), из IDE или выполнив команду

```
pip install -r requirements.txt
```

7. Запустить скрипт инициализации базы данных `db_init_script.py`, из IDE или выполнив команду

```
python3 db_init_script.py
```

1.4 Запуск и проверка работоспособности

1. Запустить серверную часть `app.py`, из IDE или выполнив команду

```
python3 app.py
```

2. Запустить клиентскую часть ???
3. Открыть стартовую страницу в браузере, если всё настроено верно, её вид должен соответствовать рис. 1.1.
4. В поле "идентификатор пользователя" ввести 1, в открывшейся странице должен отобразиться список фильмов (рис. 1.2)

Глава 2

Лабораторная работа 1.2. Неперсонализированные рекомендации

Цель: Изучение методов формирования неперсонализированных рекомендаций.

2.1 Задание

Доработать серверную и клиентскую часть учебного веб-сайта для отображения неперсонализированных рекомендаций: новых и самых популярных фильмов.

2.2 Неperсонализированные рекомендации

Простейшим видом рекомендаций являются неперсонализированные. Их особенность заключается в том, что, в отличие от персонализированных, они одинаковы для любого пользователя, взаимодействующего с рекомендательной системой. Примерами рекомендацией такого типа являются список наиболее популярных объектов (предположительно, пользователю понравятся те же объекты, которые понравились большинству других) и список новых (последних по дате появления) объектов.

2.3 Доработка серверной части

2.3.1 Отображение новых фильмов

Для отображения новых фильмов требуется:

1. В файле `api.py` создать функцию `get_new_movies`, связанную с URI `/api/movies/new`, для HTTP-метода GET, которая будет вызывать функцию `get_new_movies` из модуля `model.py`
2. В файле `model.py` создать функцию `get_new_movies`, которая будет получать из базы данных топ-20 новых фильмов с помощью SQL-запроса

```
SELECT * FROM movies ORDER BY year DESC LIMIT 20;
```

2.3.2 Отображение самых популярных фильмов

Для отображения самых популярных фильмов требуется:

1. Дополнить таблицу `movies` столбцом `rating`, для этого в файле `db_update_script` реализовать функцию `add_rating_column`, которая будет выполнять следующий SQL-запрос

```
ALTER TABLE movies ADD rating REAL;
```

2. Заполнить столбец `rating` значениями среднего рейтинга фильмов, для этого в файле `db_update_script` реализовать функцию `load_rating`, которая будет выполнять следующий SQL-запрос

```
UPDATE movies SET rating =  
  (SELECT AVG(rating) FROM ratings  
   WHERE ratings.movieId = movies.movieId);
```

3. Модифицировать функцию `row_to_movie` в файле `model_helpers.py`, добавив новое поле `rating`.
4. Модифицировать функцию `get_all_movies` в файле `model.py`, чтобы она возвращала фильмы, упорядоченные по рейтингу, для этого изменить SQL-запрос на следующий

```
SELECT * FROM movies ORDER BY rating DESC;
```

2.4 Доработка клиентской части