|  |  |
| --- | --- |
|  | **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*

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:***

**Разработка Web-приложения, содержащего информацию о фильмах, с использованием баз данных.**

Студент ИУ7-61Б  А.Д. Сушина

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсовой работы   **М**.В. Филиппов

(Подпись, дата) (И.О.Фамилия)

Консультант  \_\_\_\_\_\_\_\_\_\_\_\_\_\_

***(Подпись, дата) (И.О.Фамилия)***

*2020 г*

**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)**

УТВЕРЖДАЮ

Заведующий кафедрой ИУ7

(Индекс)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.В.Рудаков

(И.О.Фамилия)

« \_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

**ЗАДАНИЕ**

**на выполнение курсового проекта**

по дисциплине \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Базы данных\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент группы ИУ7-61б

Сушина Анастасия Дмитриевна

(Фамилия, имя, отчество)

Тема курсового проекта Разработка Web-приложения, содержащего информацию о фильмах, с использованием баз данных.

Направленность КП (учебный, исследовательский, практический, производственный, др.)

\_\_Учебный\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Источник тематики (кафедра, предприятие, НИР) \_\_кафедра\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

График выполнения проекта: 25% к 4 нед., 50% к 7 нед., 75% к 11 нед., 100% к 14 нед.

***Задание***

Разработать Web-приложение, содержащее информацию о фильмах, с использованием баз данных. В базе данных должна храниться информация о фильмах и пользователях. Должна быть создана возможность просмотреть информацию о фильмах, а также найти фильм в базе данных по каким-либо критериям: по году, жанру, стране и так далее. Пользователь должен иметь возможность зарегистрироваться или войти в приложение. Зарегистрированный пользователь может оценить фильм или оставить комментарий к фильму, также может настроить свой профиль, например, изменить изображение профиля.

***Оформление курсового проекта:***

1. Расчетно-пояснительная записка на 25-30 листах формата А4.

Расчетно-пояснительная записка должна содержать постановку задачи, введение, аналитическую часть, конструкторскую часть, технологическую часть, экспериментально-исследовательский раздел, заключение, список литературы, приложения.

2. Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

На защиту проекта должна быть представлена презентация, состоящая из 10-15 слайдов. На слайдах должны быть отражены: постановка задачи, использованные методы и алгоритмы, расчетные соотношения, структура комплекса программ, диаграмма классов, интерфейс, характеристики разработанного ПО, результаты проведенных исследований.

Дата выдачи задания « \_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_ г.

**Руководитель курсового проекта** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Филиппов М.В.

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** Сушина А.Д.

(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

Оглавление

[Введение 5](#__RefHeading___Toc4499_488842657)

[1 Аналитический раздел 6](#__RefHeading___Toc4501_488842657)

[1.1 Постановка задачи 6](#__RefHeading___Toc4503_488842657)

[1.2 Базы данных и СУБД 6](#__RefHeading___Toc4505_488842657)

[1.3 Классификация СУБД 7](#__RefHeading___Toc4507_488842657)

[1.3.1 Иерархическая модель данных 8](#__RefHeading___Toc4509_488842657)

[1.3.2 Сетевая модель данных 8](#__RefHeading___Toc4511_488842657)

[1.3.3 Реляционная модель данных 9](#__RefHeading___Toc4513_488842657)

[1.3.4 NoSQL базы данных 11](#__RefHeading___Toc4515_488842657)

[1.4 Выбор СУБД 12](#__RefHeading___Toc4517_488842657)

[1.5 Вывод 12](#__RefHeading___Toc4519_488842657)

[2 Конструкторский раздел 13](#__RefHeading___Toc4521_488842657)

[2.1 IDEF0 диаграмма 13](#__RefHeading___Toc4922_1259275439)

[2.2 Диаграмма использования 16](#__RefHeading___Toc4523_488842657)

[2.3 Диаграмма сущность-связь 17](#__RefHeading___Toc4525_488842657)

[2.4 Проектирование базы данных 17](#__RefHeading___Toc4527_488842657)

[2.5 Диаграмма компонентов 19](#__RefHeading___Toc4924_1259275439)

[2.6 Выводы 20](#__RefHeading___Toc4926_1259275439)

[3 Технологический раздел 21](#__RefHeading___Toc4529_488842657)

[3.1 Выбор средств реализации и среды разработки 21](#__RefHeading___Toc5055_3497386710)

[3.2 Выбор СУБД 21](#__RefHeading___Toc4928_1259275439)

[3.3 Требования к программному обеспечению 22](#__RefHeading___Toc5057_3497386710)

[3.4 Структура программы 22](#__RefHeading___Toc5059_3497386710)

[3.4.1 Компонент доступа к данным 22](#__RefHeading___Toc4930_1259275439)

[3.4.2 Компонент бизнес-логики 22](#__RefHeading___Toc4932_1259275439)

[3.4.3 Реализация взаимодействия между бизнес-логикой и GUI. 23](#__RefHeading___Toc4934_1259275439)

[3.5 Описание интерфейса программы 24](#__RefHeading___Toc5065_3497386710)

[3.6 Выводы 29](#__RefHeading___Toc4936_1259275439)

[Заключение 30](#__RefHeading___Toc4531_488842657)

[Список используемой литературы 31](#__RefHeading___Toc4533_488842657)

# Введение

Более чем за сто лет существования, фильмы стали неотъемлемой частью нашей жизни. Каждый год выходит огромное количество новых картин. С появлениям новых технологий, многие фильмы стали доступны для просмотра в интернете. Однако найти определенный фильм по своему вкусу достаточно сложно. Для решения этой проблемы этой существуют специальные сервисы, которые помогают с поиском, дают возможность по каким-то определенным критериям подобрать фильмы, которые могут быть вам интересны. Также использование таких сервисов дает преимущество — всегда можно прочитать отзывы людей, которые уже посмотрели этот фильм и сделать вывод о том, стоит его смотреть или нет. Таким образом, можно найти что-то себе по душе, не тратя на это слишком много времени и ресурсов.

Целью данной курсового проекта является создание web-приложения «MoviesToWatch», которое предоставляет возможность просмотра информации о различных фильмах. Пользователь может зарегистрироваться или войти в приложение, а также изменить информацию о себе. Зарегистрированный пользователь может оставить комментарий о фильме или оценить его. Приложение также позволяет осуществлять поиск фильмов по определенным критериям.

Для достижения цели были поставлены следующие задачи:

1. Описать задачу и определить необходимый функционал;
2. Изучить стек технологий, необходимых для создания web-приложения;
3. Спроектировать базу данных, необходимую для хранения и структурирования данных;
4. Реализовать спроектированную базу данных с использованием выбранной СУБД;
5. Реализовать web-приложение, решающее задачи, поставленные в техническом задании, и поддерживающее взаимодействие с реализованной базой данных.

# Аналитический раздел

В аналитическом разделе будет дано описание выбранной предметной области, будут предъявлены требования к функционалу, а также будут рассмотрены возможные существующие методы реализации.

## Постановка задачи

Необходимо разработать web-приложение, реализующее доступ к информации о фильмах. Можно выделить несколько задач, которые должно решать это приложение.

Прежде всего, должна быть реализована система регистрации и авторизации пользователя в системе. А также возможность для пользователя изменить информацию о себе. Должна быть реализована система поиска фильмов по различным критериям. Также должна быть реализована возможность оставить комментарий и оценить фильм у пользователя.

Все работа приложения основывается на работе с базой данных, которая содержит информацию о фильмах и пользователях приложения. Кроме того, необходимо реализовать клиент-серверное приложение, где в роли клиента будет выступать браузер, а в роли сервера — разработанное приложение. Для реализации такого взаимодействия нужно использовать стек технологий, позволяющий осуществлять взаимодействие по протоколу HTTP.

## Базы данных и СУБД

База данных — это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляется системой управления базами данных (СУБД). Данные вместе с СУБД, а также приложения, которые с ними связаны, называются системой баз данных, или, для краткости, просто базой данных.

К базе данных предъявляются следующие требования:

1. целостность базы данных – требование полноты и непротиворечивости данных;
2. многократное использование данных;
3. быстрый поиск и получение информации по запросам пользователей;
4. простота обновления данных;
5. минимизация избыточности данных;
6. безопасность данных - защита данных от несанкционированного доступа, искажения и уничтожения.
7. Восстановление данных после программных и аппаратных сбоев;
8. независимость данных от прикладных программ.

СУБД должна выполнять следующие функции:

1. Непосредственное управление данными во внешней памяти
2. Управление буферами оперативной памяти
3. Управление транзакциями
4. Журнализация
5. Поддержка языков БД

## Классификация СУБД

СУБД можно классифицировать по:

1. по модели данных;
   1. Иерархическая СУБД;
   2. Сетевая СУБД;
   3. Реляционная СУБД;
   4. Объектно-ориентированная СУБД;
   5. Объектно-реляционная СУБД.
2. по степени распределённости;
   1. Локальные СУБД (все части локальной СУБД размещаются на одном компьютере);
   2. Распределённые СУБД (части СУБД могут размещаться на двух и более компьютерах).
3. по способу доступа к БД;
   1. Файл-серверные;
   2. Клиент-серверные;
   3. Встраиваемые СУБД;
   4. прочие.

### Иерархическая модель данных

Иерархическая модель данных - это модель данных, где используется представление базы данных в виде древовидной (иерархической) структуры, состоящей из объектов (данных) различных уровней. Между объектами существуют связи, каждый объект может включать в себя несколько объектов более низкого уровня. Такие объекты находятся в отношении предка (объект более близкий к корню) к потомку (объект более низкого уровня), при этом возможна ситуация, когда объект-предок не имеет потомков или имеет их несколько, тогда как у объекта-потомка обязательно только один предок.

Иерархической базой данных является файловая система, состоящая из корневого каталога, в котором имеется иерархия подкаталогов и файлов. [1]

Достоинства:

* Простота организации;
* Наиболее быстрый доступ к информации.

Недостатки:

* Избыточность - нельзя ссылаться на одно и то же, необходимо дублировать информацию.
* Не любая предметная область может быть представлена такой структурой.
* При изменении структуры модели данных требуется изменение программного обеспечения и программных средств или создание нового. [2]

|  |
| --- |
| Рис 1. Иерархическая модель базы данных |

### Сетевая модель данных

К основным понятиям сетевой модели базы данных относятся: уровень, элемент (узел), связь.

Узел - это совокупность атрибутов данных, описывающих некоторый объект. На схеме иерархического дерева узлы представляются вершинами графа. В сетевой структуре каждый элемент может быть связан с любым другим элементом.

Сетевые базы данных подобны иерархическим, за исключением того, что в них имеются указатели в обоих направлениях, которые соединяют родственную информацию.

Достоинства:

* быстрый доступ к информации бд;
* простота реализации.

Недостатки:

* При изменении информации требуется изменение программного обеспечения.

|  |
| --- |
| Рис 2. Сетевая модель данных |

### Реляционная модель данных

Реляционная модель - совокупность данных, состоящая из набора двумерных таблиц. В теории множеств таблице соответствует термин отношение (relation), физическим представлением которого является таблица, отсюда и название модели – реляционная. В сравнении с иерархической и сетевой моделью данных, реляционная модель отличается более высоким уровнем абстракции данных. На реляционной модели данных строятся реляционные базы данных.

При табличной организации данных отсутствует иерархия элементов. Любая таблица в реляционной базе состоит из строк, которые называют записями, и столбцов, которые называют полями. На пересечении строк и столбцов находятся конкретные значения данных. Для каждого поля определяется множество его значений. В реляционной алгебре поименованный столбец отношения называется атрибутом, а множество всех возможных значений конкретного атрибута – доменом. Строки таблицы со значениями разных атрибутов называют кортежами. Атрибут, значение которого однозначно идентифицирует кортежи, называется ключевым. Так ключевое поле – это такое поле, значения которого в данной таблице не повторяется. В отличие от иерархической и сетевой моделей данных в реляционной отсутствует понятие группового отношения. Для отражения ассоциаций между кортежами разных отношений используется дублирование их ключей. Индекс – это поле или совокупность полей, чьи значения имеются в нескольких таблицах и которое является первичным ключом в одной из них. С помощью индексов и ключей устанавливаются связи между таблицами. [3]

Согласно Дейту реляционная модель состоит из трех частей, описывающих разные аспекты реляционного подхода: структурной части, манипуляционной части и целостной части.

Структурная часть отвечает за принцип построения структуры реляционной базы данных на нормализированном наборе n-арных отношений, в форме таблиц. Важно что реляционная база данных, структурно может представляться только в виде отношений.

В целостной части реляционной модели данных фиксируются два базовых требования целостности, которые должны поддерживаться в любой реляционной СУБД. Первое требование называется требованием целостности сущностей. Конкретно требование состоит в том, что любое отношение должно обладать первичным ключом. Второе требование называется требованием целостности по ссылкам и является несколько более сложным. Требование целостности по ссылкам, или требование внешнего ключа состоит в том, что для каждого значения внешнего ключа, появляющегося в дочернем отношении, в родительском отношении должен найтись кортеж с таким же значением первичного ключа, либо значение внешнего ключа должно быть неопределенным.

В манипуляционной части модели утверждаются операторы манипулирования отношениями - реляционная алгебра и реляционное исчисление. Первый механизм базируется в основном на классической теории множеств (с некоторыми уточнениями), а второй - на классическом логическом аппарате исчисления предикатов первого порядка. Основной функцией манипуляционной части реляционной модели является обеспечение меры реляционности любого конкретного языка реляционных БД: язык называется реляционным, если он обладает не меньшей выразительностью и мощностью, чем реляционная алгебра или реляционное исчисление.

Достоинства

* Изложение информации в простой и понятной для пользователя форме (таблица).
* Реляционная модель данных основана на строгом математическом аппарате, что позволяет лаконично описывать необходимые операции над данными.
* Независимость данных от изменения в прикладной программе при изменении.
* Позволяет создавать языки манипулирования данными не процедурного типа.
* Для работы с моделью данных нет необходимости полностью знать организацию БД.

Недостатки

* Относительно медленный доступ к данным.
* Трудность в создании БД основанной на реляционной модели.
* Трудность в переводе в таблицу сложных отношений.
* Требуется относительно большой объем памяти.

SQL (Structured Query Language) – основной интерфейс работы с реляционными базами данных. SQL используется для добавления, обновления и удаления строк данных, извлечения наборов данных для обработки транзакций и аналитических приложений, а также для управления всеми аспектами работы базы данных.

### NoSQL базы данных

NoSQL — термин, обозначающий ряд подходов, направленных на реализацию СУБД, имеющих существенные отличия от моделей, используемых в традиционных реляционных СУБД с доступом к данным средствами языка SQL

NoSQL-способ структуризации данных заключается в избавлении от ограничений при хранении и использовании информации. Базы данных NoSQL, используя неструктуризированный подход, предлагают много эффективных способов обработки данных в отдельных случаях.

Реляционные базы данных обеспечивают набор свойств ACID: атомарность, непротиворечивость, изолированность, надежность. Базы данных NoSQL зачастую предлагают компромисс, смягчая жесткие требования свойств ACID ради более гибкой модели данных, которая допускает горизонтальное масштабирование. [4]

Типы баз данных NoSQL:

* БД на основе пар «ключ-значение» - наиболее простой вариант хранилища данных, использующий ключ для доступа к значению в рамках большой хэш-таблицы.
* Документно-ориентированные.  В коде приложения данные часто представлены как объект или документ в формате, подобном JSON, поскольку для разработчиков это эффективная и интуитивная модель данных.
* Графовые БД. Графовые базы данных упрощают разработку и запуск приложений, работающих с наборами сложносвязанных данных.
* БД в памяти. При хранении компьютерных данных система в основном полагается на оперативную память.
* Поисковые бд. Эти БД используют индексы для классификации общих характеристик для поступающих данных и основная цель этих БД – содействовать быстрому поиску.

Достоинства:

* гибкость;
* масштабируемость;
* высокая производительность;
* широкие функциональные возможности;
* возможность работать с разными представлениями информации;
* высокая доступность.

Недостатки:

* ограниченная емкость встроенного языка запросов;
* сложности в поддержке всех ACID-требований;
* сильная привязка приложения к конкретной СУБД;
* недостаток специалистов по noSQL-базам. [5]

## Выбор СУБД

Так как курс основывался на работе с SQL, было принято решение использовать реляционную модель данных для реализации решения задачи.

## Вывод

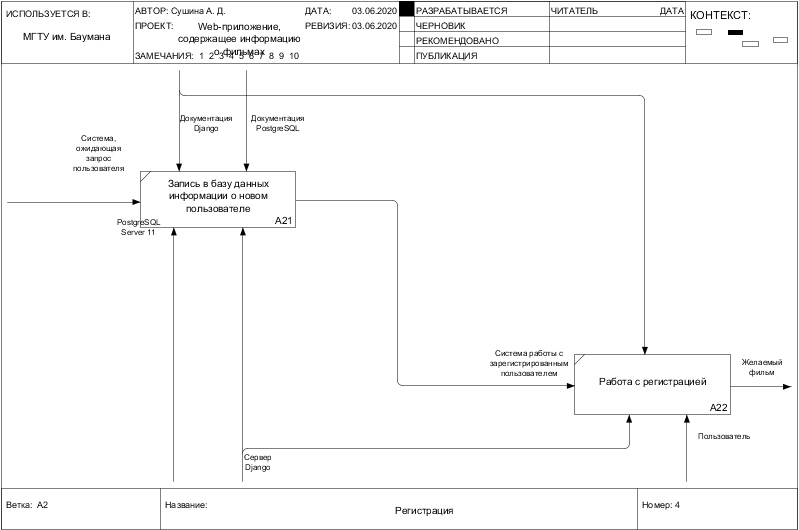
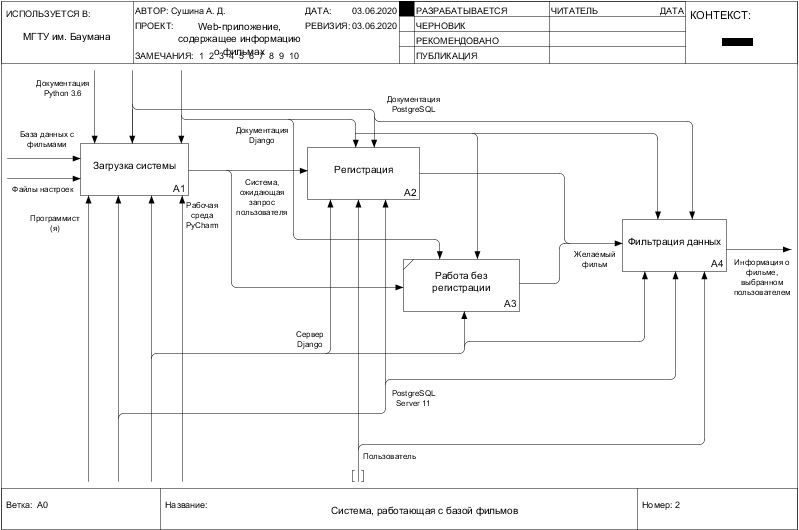
В данном разделе была выполнена постановка задачи, были рассмотрены различные виды СУБД и выбрана подходящий для заданной задачи тип базы данных.

# Конструкторский раздел

В конструкторском разделе будет проведен разбор используемых алгоритмы, будут представлены необходимые диаграммы для понимания сути проекта.

## IDEF0 диаграмма

На рисунках 3-7 представлена IDEF0 диаграмма

Система получает на вход файлы настроек и может общаться с уже заполненной базой данных. Главной задачей приложения является поиск информации о фильмах, поэтому основная информация, получаемая на выходе — информация о фильмах.

|  |
| --- |
| Рис 3. Система. Работающая с базой фильмов |

Сначала происходит запуск приложения. По окончании запуска, приложение открывает порт 8000 и начинает слушать его. Система обрабатывает каждый поступающий запрос. Пользователь отправляет некоторые запросы и получает на них ответы. Таким образом, он может зарегистрироваться в системе или продолжить работу без регистрации. После того как пользователь выбрал фильм, происходит поиск информации о фильме с помощью фильтров.

|  |
| --- |
| Рис 4. Декомпозиция первого уровня |

Загрузка системы включает в себя чтение файла настроек, запуск непосредственно сервера и инициализацию работы с базой данных .

|  |
| --- |
| Рис 5. Декомпозиция загрузки ситемы |

Регистрация пользователя включает в себя создание новой записи о пользователе и дальнейшую работу приложения с зарегистрированным пользователем.

|  |
| --- |
| Рис 6. Декомпозиция регистрации |

Фильтрация данных включает в себя непосредственно использование фильтров или же выбор фильма на главной странице из отдельных подборок. После этих манипуляций происходит поиск информации о самом фильме, а затем сбор всей информации в единый шаблон.

|  |
| --- |
| Рис 7. Декомпозиция фильтрации данных |

## Диаграмма использования

На рисунке 8 представлена use-case диаграмма приложения, описывающая возможные сценарии использования приложения.

|  |
| --- |
| Рис 8 Use-case диаграмма |

## Диаграмма сущность-связь

На рисунке 9 представлена ER модель приложения. На ней выделены основные сущности приложения, определены связи между сущностями, а также описаны атрибуты каждой сущности.

|  |
| --- |
| Рис 9 ER-диаграмма |

## Проектирование базы данных

База данных будет реализована с помощью следующих таблиц:

* таблица фильмов Film;
* таблица стран Country;
* таблица жанров Genre;
* таблица актеров Actor;
* таблица комментариев Comment;
* таблица оценок Like;
* таблица пользователей Profile;

Таблица Country содержит информацию о странах, которые производят фильмы. Содержит следующие поля:

* id — идентификатор страны;
* name — название страны.

Таблица Genre содержит информацию о жанрах, в котрых могут сниматься фильмы. Содержит следующие поля:

* id — идентификатор жанра;
* name — название жанра.

Таблица Actor содержит информацию об актерах, которые снимаются в фильмах. Содержит следующие поля:

* id — идентификатор актера;
* firstName — имя актера;
* lastName — фамилия актера;

Таблица Film содержит информацию о фильме. Содержит следующие поля:

* title — название фильма;
* year — год выпуска;
* description - описание фильма;
* rating — рейтинг фильма на сайте (среднее среди всех оценок);
* image — изображение фильма на сайте (хранится только ссылка на изображение);
* creaded\_at — время и дата создания записи о фильме.

Таблица Profile содержит информацию о пользователе. Содержит следующие поля:

* username — имя пользователя;
* password — пароль пользователя;
* email — email пользователя;
* avatar — изображение профиля(хранится только ссылка на изображение);
* created\_at — дата и время регистрации.

Таблица Comment содержит информацию о комментарях. Содержит следующие поля:

* text — текст комметария;
* film\_id — id фильма, к которому оставлен комментарий;
* author\_id — id пользователя, который оставил комментарий;
* created\_at — дата и время создания комментария.

Таблица Like содержит информацию об оценках. Содержит следующие поля:

* value — значение оценки, целое значение от 1 до 5;
* author\_id - id пользователя, который оставил оценку;
* film\_id — id фильма, на которой поставлена оценка;
* created\_at — дата и время создания оценки.

Для реализации отношения «многие ко многим» также необходимо создать три дополнительные таблицы: film\_actors(поля film\_id, actor\_id), film\_genres(film\_id, genre\_id), film\_countries(film\_id, country\_id).

Схема базы данных представлена на рисунке 10.

|  |
| --- |
| Рис 10. Схема базы данных. |

## Диаграмма компонентов

На рисунке 11 представлена диаграмма компонентов приложения.

* Manage отвечает за запуск сервера и выдачу ответов на запросы.
* Mycourse отвечает за настройки проекта, настройки доступа к базе данных, а также за направление запросов к своим обработчикам.
* Coureser.views отвечает за обработку запросов и связь между бизнес-логикой и интерфейсом пользователя. Отвечает за отдачу web-страниц.
* Coureser.common содержит простые функции, не относящиеся к другим компонентам.
* Coureser.logic содержит бизнес-логику приложения.
* Coureser.form содержит информацию о формах, отвечает за их отрисовку и обработку.
* Coureser.models содержит информацию о моделях, которые представляют собой интерфесы для доступа к таблицам базы данных
* Coureser.managers отвечает за обработку сложных запросов к базе данных.

|  |
| --- |
| Рис 11. Диаграмма компонентов |

## Выводы

В данном разделе была проведена формализация задачи, были представлены диаграмма использования, диаграмма базы данных, диаграмма компонентов.

# Технологический раздел

В технологическом разделе будет рассмотрен выбор средств реализации, а также представлен интерфейс пользователя и результаты тестирования

## Выбор средств реализации и среды разработки

Для разработки проекта выбран язык программирования Python версии 3.6. Это современный язык программирования, он поддерживает объектно-ориентированную парадигму, а также поддерживает фреймворки для разработки web-приложений. В данном проекте используется фреймворк Django[6], так как он предоставляет широкий набор инструментов для создания web-приложений, а также поддерживает ORM технологию, упрощающую работу с базами данных.

Была использована среда разработки PyCharm 2020.1.1 (Professional Edition)[9], так как она предоставляет удобный интерфейс для работы с Python и Django, а также доступна бесплатно для студентов.

Для разработки web-страниц был использован Bootstrap[7], так как этот инструментарий предоставляет доступ к простым шаблонам оформления web-страниц.

## Выбор СУБД

Данный проект реализуется в рамках учебного курса, поэтому основными критериями для выбора СУБД являются опыт работы, бесплатная лицензия, а также наличие ORM. Выбор производился из наиболее популярных баз данных, а именно: SQLite, PostgeSQL, Oracle, MongoDB.

Таблица 1. Обзор систем управления базами данных.

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Бесплатная лицензия | Опыт работы | ORM |
| Oracle | - | - | + |
| SQLite | + | - | + |
| PostgreSQL | + | + | + |
| MongoDB | + | - | + |

В ходе проведенного анализа была выбрана СУБД PostgreSQL.[8] Эта СУБД позволит реализовать все необходимые для приложения действия над базами данных. Эту СУБД можно использовать с ORM-фреймворками, что позволит разработать эффективное приложение, приложив минимум усилий.

## Требования к программному обеспечению

Для корректной работы приложения необходимы следующие условия:

* Интерпетатор Python 3.6 с зависимостями, описанными в файле requirements.txt
* СУБД PostgreSQL

## Структура программы

Далее будет приведена структура программы, в соответствии с модулями, представленными в пункте 2.5

### Компонент доступа к данным

Были реализованны следущие классы:

1. класс Film;
2. класс Profile;
3. класс Like;
4. класс Genre;
5. класс Actor;
6. класс Country;
7. класс Comment
8. класс FilmManager;
9. класс ProfileManager;
10. класс LikeManager;

Классы 1-7 описывают таблицы баз данных, к которым осуществляется доступ. Класс Profile связан с встроенным классом Django User[6], который описывает базу данных пользователей приложения. Классы 8-10 описывают дополнительные действия по доступу к данным (Например, получение набора фильмов по списку параметров.)

### Компонент бизнес-логики

Компонент бизнес-логики представлен следующими классами:

* класс CommentLogic;
* класс LikeLogic;
* класс ProfileLogic.

Все эти классы реализуют логику работу приложения.

### Реализация взаимодействия между бизнес-логикой и GUI.

В Django используется шаблонизатор, который преобразует шаблоны в html-страницы. [6] Этот инструмент позволяет динамически создавать HTML страницы с использованием данных, полученных в приложении.

Были реализованы следующие шаблоны:

* base.html — основная страница, остальные шаблоны наследуются от нее.
* Film.html - страница фильма.
* index.html - главная страница сайта.
* login.html — страница входа.
* register.html — страница регистрации.
* search.html — страница поиска.
* settings.html — страница настроек.
* comment.html — шаблон одного комментария.
* film-big-card.html — шаблон одной большой карточки (расположены на странице поиска).
* film-card.html — шаблон маленькой карточки (расположены на главной странице).
* pagination.html — шаблон элемента перехода между страницами.

Для передачи данных в шаблоны были реализованы следующие классы:

* FilmView;
* IndexView;
* LikeView;
* LoginView;
* LogoutView;
* RegisterView;
* SearchView;
* SettingsView.

Для реализации и обработки форм, заполняемых пользователем на страницах, были реализованы следующие классы:

* CommentForm — форма комментария.
* LikeForm — форма оценки.
* LoginForm — форма входа.
* RegisterForm — форма регистрации.
* SearchForm — форма поиска.
* SettingsForm — форма настроек.

## Описание интерфейса программы

Главная страница приложения показана на рисунке 12. На странице представлены три подборки фильмов: подборка новых фильмов, появившихся на сайте, подборка самых обсуждаемых фильмов и подборка самых рейтинговых фильмов на сайте. В верхней части страницы находится навигационная панель, из которой можно перейти на страницы входа, регистрации, а также на страницу поиска. Пользователь может перейти на страницу фильма, кликнув по карточке фильма.

|  |
| --- |
| Рис 12 Главная страница приложения |

На рисунках 13 и 14 представлены страницы входа и регистрации соответственно. На обеих страницах находятся формы для входа. Пользователь может зарегистрироваться или войти, если у него уже есть аккаунт. Зарегистрированным пользователям доступно больше действий, чем незарегистрированным.

|  |
| --- |
| Рис 13 Страница входа |
| Рис 14 Страница регистрации |

После авторизации верхняя навигационная панель меняет свой вид, в ней появляется информация о пользователе и кнопки «Настройки» и «Выйти», с помощью которых можно перейти к настройкам профиля или выйти из аккаунта. Новые элементы панели показаны на рисунке 15.

|  |
| --- |
| Рис 15 Информация о пользователе в навигационной панели |

На рисунке 16 представлена страница настроек. На этой странице пользователь может изменить свое имя, email или фотографию профиля.

|  |
| --- |
| Рис 16 Страница настроек профиля |

На рисунке 17 представлена страница одного фильма. На этой страницы представлено изображение, относящееся к фильму, а также информация о нем. Год, страна, жанры и актеры являются ссылками, на которым можно найти все фильмы, которые каким-либо образом относятся к этому году, стране, жанру или актеру. Также на странице можно увидеть комментарии, которые оставили другие пользователи к этому фильму. Если комментариев много, то можно посмотреть больше, перейдя на следующую страницу.

Страница фильма выглядит по-разному в зависимости от того, авторизован пользователь или нет. На рисунках 18 и 19 показаны различия на странице. Если пользователь авторизован, то он модет поставить фильму оценку или оставить комментарий. В ином случае появляется ссылка для перехода к странице авторизации.

|  |
| --- |
| Рис 17 Страница одного фильма |
| Рис 18 Функционал для авторизованного пользователя на странице фильма |
| Рис 19 Функционал для неавторизованного пользователя на странице фильма. |

На рисунках 20 и 21 представлена страница поиска. На странице выводятся все фильмы, которые соответствуют запросу. Также выводится краткая информация о них. При нажатии на картинку или название фильма можно перейти на страницу фильма. При нажатии на данные о фильме, можно перейти к списку фильмов, связанных с этими данными. В правой части страницы находится форма для выбора параметров поиска. После ввода всех параметром необходимо нажать кнопку «Применить». В нижней части страницы также находятся кнопки для перелистывания страниц.

|  |
| --- |
| Рис 20 Страница поиска (верхняя часть) |
| Рис 21 Страница поиска (нижняя часть) |

## Выводы

В данном разделе был выбран набор инструментов для реализации поставленной задачи. Была описана структура программы, а также продемонстрирован интерфейс приложения.

# Заключение

В ходе работы были проанализированы существующие СУБД, их преимущества и недостатки.   
 Было разработано web-приложение «MoviesToWatch», которое предоставляет информацию о фильмах. Для хранения данных была разработана база данных, содержащая информацию о фильмах и пользователях.

Реализованная система позволяет просмотреть информацию о фильмах, найти фильмы по каким-либо параметрам. Также существует возможность зарегистрироваться в системе или войти в систему. Зарегистрированный пользователь может оставить комментарий к фильму, поставить фильму оценку и изменить информацию о себе.

# Список используемой литературы

1. Андреев, А.В. КЛАССИФИКАЦИЯ СИСТЕМ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ / А.В. Андреев (дата обращения 02.06.2020)
2. Иерархический и сетевой подходы при построении баз данных, основные понятия, достоинства и недостатки. [Электронный ресурс] URL: <https://sites.google.com/site/gosyvmkss12/bazy-dannyh/06-ierarhiceskij-i-setevoj-podhody-pri-postroenii-baz-dannyh-osnovnye-ponatia-dostoinstva-i-nedostatki> (Дата обращения 02.06.2020)
3. Реляционная модель данных [Электронный ресурс] URL: <https://ru.bmstu.wiki/Реляционная_модель_данных>(Дата обращения 02.06.2020)
4. Что такое NoSQL? [Электронный ресурс] URL: <https://aws.amazon.com/ru/nosql/>(Дата обращения 02.06.2020)
5. Что такое NoSQL СУБД: история, виды, примеры, применение Big Data [Электронный ресурс] URL: <https://www.bigdataschool.ru/wiki/nosql>(Дата обращения 02.06.2020)
6. Документация Django [Электронный ресурс]. URL: <https://djbook.ru/rel3.0/> (Дата обращения 18.05.2020)
7. Bootstrap documentation [Электронный ресурс]. URL: <https://bootstrap-4.ru/> (Дата обращения 19.05.2020)
8. Документация PostgeSQL [Электронный ресурс]. URL: <https://postgrespro.ru/docs/postgresql/12/index.html> (Дата обращения 20.05.2020)
9. PyCharm [Электронный ресурс]. URL: https://www.jetbrains.com/ru-ru/pycharm/ (дата обращения 20.05.2020)