# **ВВЕДЕНИЕ**

За более чем сто лет своего существования, кинематограф стал одним из наиболее популярных и известных видов искусства, но, тем не менее, кино – на рынке всего лишь продукт, требующий, как и все продукты специализированного и научного подхода по популяризации и продвижению в массы. Как и все продукты, он требует грамотного продвижения, тем более учитывая роль рекламной компании в его кассовых сборах и дальнейшей возможности быть узнаваемым.

В настоящее время использование информационных технологий в сервисе оценки кинофильмов не является редкостью. Спектр их применения широк и варьируется от блогов отдельно взятых людей до полноценного сайта со своей системой оценки каждого из фильмов.

Целью данного курсового проекта является создание сервиса оценки кинофильмов, а так же создание удобного и функционального программного обеспечения с понятным пользователю интерфейсом.

Для достижения поставленной цели необходимо решить следующие задачи:

* изучить предметную область сервиса оценки кинофильмов;
* выполнить функциональное моделирование процесса оценки кинофильмов;
* разработать модели представления сервиса оценки кинофильмов на основе UML;
* проанализировать и создать логическую и физическую модель представления данных;
* создать базу данных, хранящую информацию о кинофильмах, о пользователях и объектах, необходимых для выставления оценки кинофильму;
* реализовать возможности регистрации и авторизации;
* разработать программное обеспечение;
* протестировать программный продукт.

Автоматизация сервиса оценки кинофильмов позволит грамотно организовать подсчет и популярность каждого из фильмов, а так же обеспечивает способ своевременного оставления отзывов о просмотренном кинофильме.

**1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ**

Предметной областью данного курсового проекта является сервис оценки кинофильмов.

Задачей курсового проекта является создание приложения, которое позволило бы автоматизировать процессы оценки кинофильмов.

При написании программного кода используется язык Java. Java это объектно-ориентированный язык программирования. Изначально язык программирования назывался Oak (James Gosling) и разрабатывался для бытовой электроники, но впоследствии был переименован в Java и стал использоваться для написания апплетов, приложений и серверного программного обеспечения [3].

У языка Java существует множество преимуществ перед другими языками программирования, что позволяет решать с его помощью практически любые задачи. Так же с помощью Java можно создавать модульные программы, исходный код которых может использоваться многократно.

Одним из основных преимуществ языка Java является возможность переноса программ из одной системы в другую. Поскольку программы на Java не зависят от платформы как на уровне исходного кода, так и на двоичном уровне, их можно запускать в различных системах, что особенно важно для программ, предназначенных для World Wide Web [4].

Работа приложения основывается на том, что клиентская часть реализует пользовательский интерфейс, формирует запросы к серверу и обрабатывает ответы от него, в то время как серверная часть получает запрос от клиента, выполняет вычисления, после чего формирует ответ и отправляет её клиенту. Выполнение всех описанных сетевых функций приложения без проблем реализуется с помощью Java, так как сетевая работа является одной из сильных сторон данного языка программирования.

Огромное количество библиотек классов в Java многократно упрощают работу программиста. Java предоставляет программисту богатый набор классов объектов для ясного абстрагирования многих системных функций, используемых при работе с окнами, сетью и для ввода-вывода. Ключевая черта этих классов заключается в том, что они обеспечивают создание независимых от используемой платформы абстракций для широкого спектра системных интерфейсов [3].

Так же в курсовом проекте используются СУБД. Система управления базами данных - это совокупность языковых и программных средств, которая осуществляет доступ к данным, позволяет их создавать, менять и удалять, обеспечивает безопасность данных и т.д. В общем СУБД - это система, позволяющая создавать базы данных и манипулировать сведениями из них. А осуществляет этот доступ к данным СУБД посредством специального языка - SQL. SQL (Stuctured Query Language ) - язык структурированных запросов, основной задачей которого является предоставление простого способа считывания и записи информации в базу данных. SQL включает много разных типов операторов, разработанных для взаимодействия с базами данных.

Для реализации хранения, обработки и дальнейшего использования информации в данном приложении используется СУБД MySQL.

MySQL является наиболее приспособленной для применения в среде веб системой управления базами данных.

Основные преимущества MySQL:

− многопоточность, поддержка нескольких одновременных запросов;

− оптимизация связей с присоединением многих данных за один проход;

− записи фиксированной и переменной длины;

− гибкая поддержка форматов чисел, строк переменной длины и меток времени;

− быстрая работа, масштабируемость.

 Таким образом, базы данных имеют достаточное количество явно неоспоримых преимуществ, используемых для решения задач данного курсового проекта.

**2 ОПИСАНИЕ ПРОЦЕССА ВЫСТАВЛЕНИЯ ОЦЕНКИ КИНОФИЛЬМУ**

Для получения более полного представления о том, как можно автоматизировать процесс оценки кинофильмов, была создана функциональная модель.

Ключевой процесс в данной теме – оценивание кинофильма, ради чего и будет писаться программное обеспечение. Для создания же функциональной модели был выбран стандарт IDEF0. IDEF0 — методология функционального моделирования и графическая нотация, предназначенная для формализации и описания бизнес-процессов. Отличительной особенностью IDEF0 является её акцент на соподчинённость объектов. В IDEF0 рассматриваются логические отношения между работами, а не их временна́я последовательность[3].

На контекстной диаграмме верхнего уровня (рисунок 2.1) представлена функциональная модель «Выставление оценки кинофильму», а также определены потоки входных и выходных данных, механизмы ограничения и управления данными.

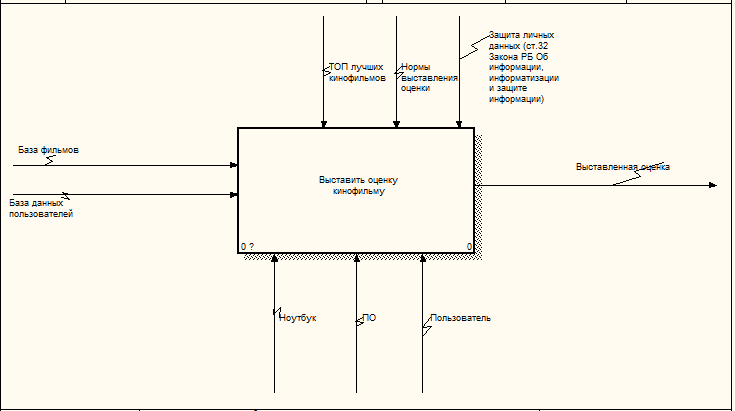


Рисунок 2.1 – Контекстная диаграмма верхнего уровня «Выставить оценку кинофильму»

Входной поток включает в себя базу данных фильмов, а так же базу данных пользователей. После соответствующей обработки в выводном потоке имеем выставленную оценку. В роли управляющих механизмов выступают ТОП лучших кинофильмов, нормы выставления оценок и закон РБ «Об информации, информатизации и защите информации», ст.32. Механизмами являются ноутбук, пользователь и программное обеспечение.

На рисунке 2.2 отображена декомпозиция контекстной диаграммы, состоящая из четырёх блоков: «Найти кинофильм», «Посмотреть кинофильм», «Зайти в аккаунт на сервисе» и «Выставить оценку».

Первый компонент данной декомпозиции («Найти кинофильм») подразумевает нахождение кинофильма с помощью ТОПа лучших кинофильмов.

Декомпозиция данного блока отображена на рисунке 2.3. Она представлена тремя компонентами: «Выбрать жанр», «Выбрать страну производства», «Определиться с фильмом».

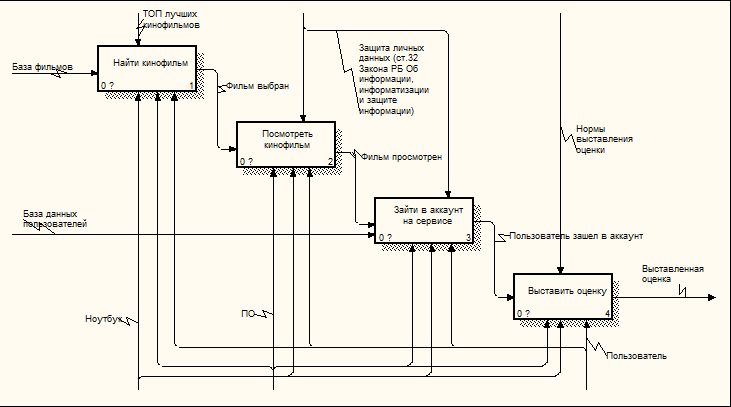


Рисунок 2.2 – Декомпозиция контекстной диаграммы верхнего уровня «Выставить оценку кинофильму»

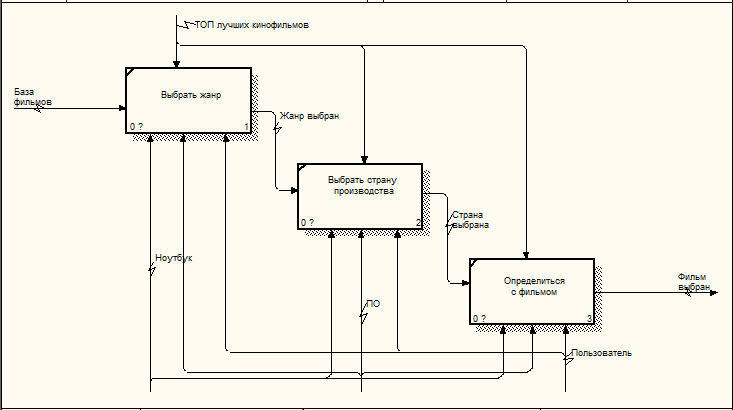


Рисунок 2.3 – Декомпозиция блока «Найти кинофильм»

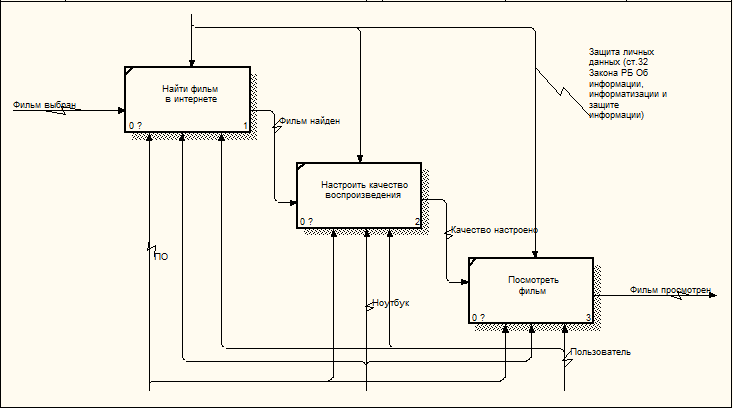


Рисунок 2.4 – Декомпозиция блока «Посмотреть кинофильм»

Второй компонент декомпозиции контекстной диаграммы («Посмотреть кинофильм») включает в себя поиск и просмотр выбранного фильма.

Декомпозиция этого блока отображена на рисунке 2.4. Она представлена тремя компонентами: «Найти фильм в интернете», «Настроить качество воспроизведения» и «Посмотреть фильм».

Третий компонент декомпозиции контекстной диаграммы («Зайти в аккаунт на сервисе») подразумевает собой вход в уже зарегистрированный аккаунт на сервисе. Декомпозиция данного блока отображена на рисунке 2.5. Она представлена тремя компонентами: «Зайти на сайт сервиса», «Ввести логин и пароль от аккаунта» и «Войти в свой аккаунт».

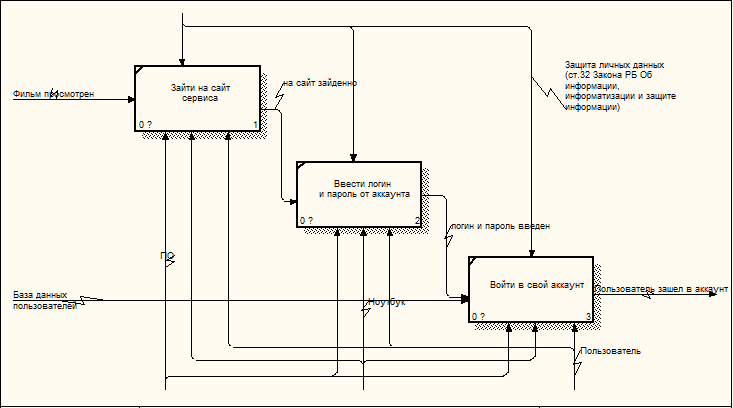


Рисунок 2.5 – Декомпозиция блока «Зайти в аккаунт на сервисе»

Четвертый компонент декомпозиции контекстной диаграммы («Выставить оценку») подразумевает собой выставление оценки на сайте сервиса оценки кинофильмов. Декомпозиция данного блока отображена на рисунке 2.6. Она представлена тремя компонентами: «Найти просмотренный фильм», «Выставить заслуженную оценку» и «Убедиться в верно выставленной оценке».

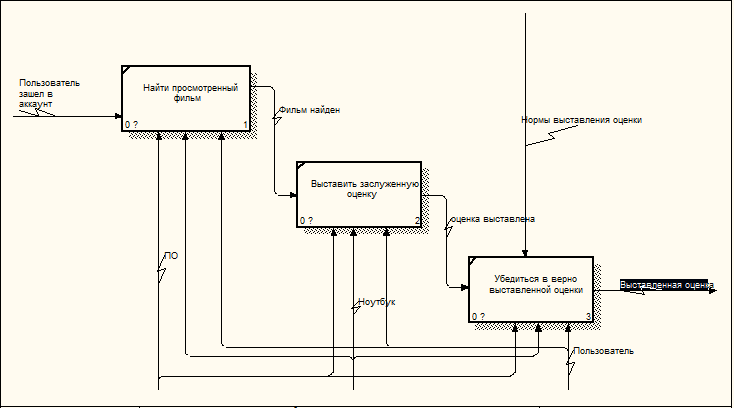


Рисунок 2.5 – Декомпозиция блока «Зайти в аккаунт на сервисе»

Данный продукт открывает некоторые перспективы для упрощения поиска фильма. Пользователи смогут без труда получить информацию о самых интересных фильмах по мнению других пользователей, которые уже посмотрели выбранный фильм.

**3 СПЕЦИФИКАЦИЯ ВАРИАНТОВ ИСПОЛЬЗОВАНИЯ СИСТЕМЫ**

Диаграммы вариантов использования описывают взаимоотношения и зависимости между группами вариантов использования и действующих лиц, участвующими в процессе.

На данной диаграмме представлены 2 актера: пользователь (студент) и администратор. Актеры данной системы имею общие варианты использования, такие как: авторизация, фильтрация данных о студенте, а так же сортировка. Но при этом каждый актер имеет и уникальные варианты использования. Представление диаграммы показано на рисунке 3.1.

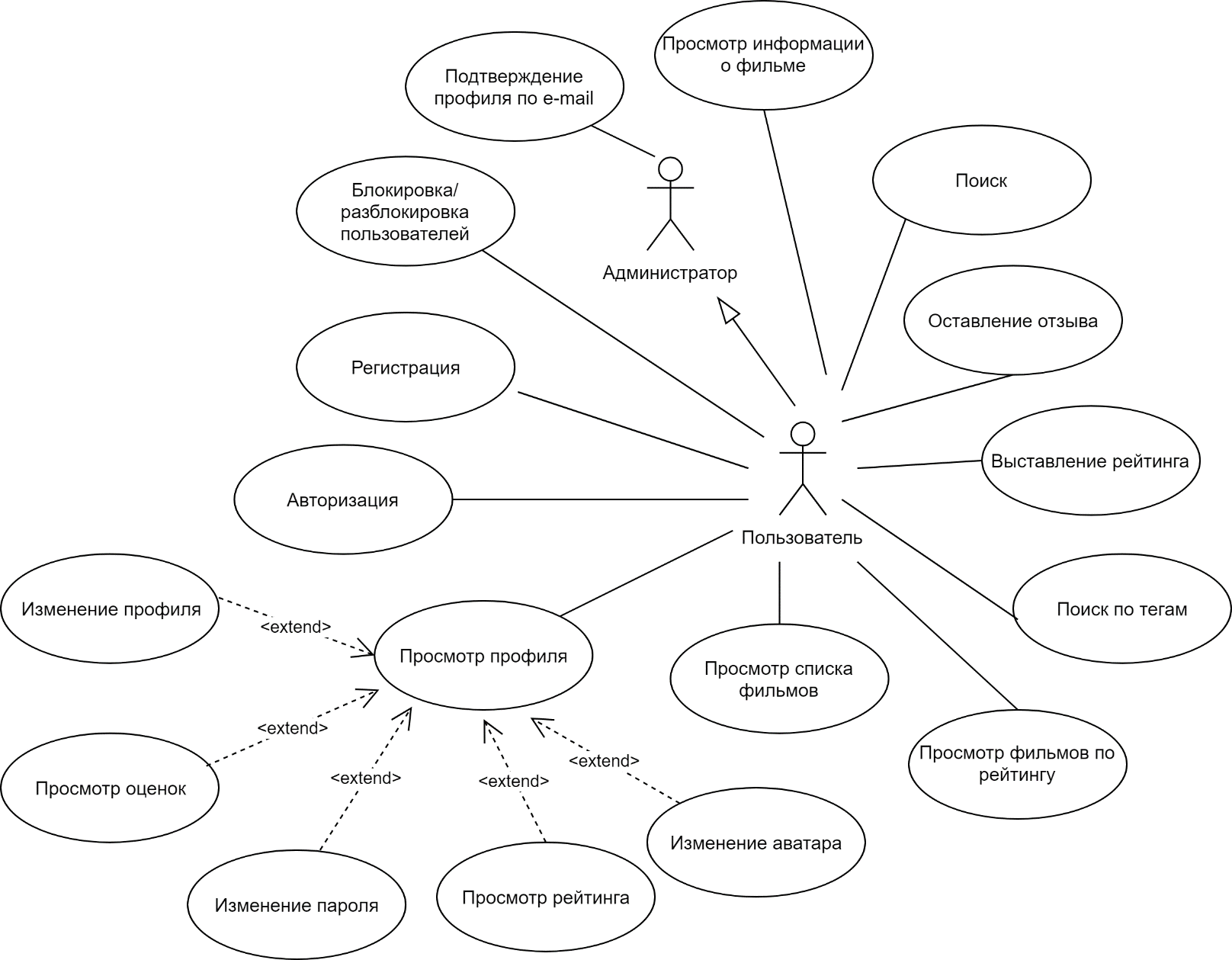


Рисунок 3.1 – Диаграмма вариантов использования

**4 ИНФОРМАЦИОННАЯ МОДЕЛЬ СИСТЕМЫ И ЕЁ ОПИСАНИЕ**

Для качественной работы системы оценки кинофильмов потребовалось выделить и создать шесть сущностей: фильм, жанр, рейтинг, отзыв, режиссер и пользователь.

Информационная модель системы в графическом виде представлена ниже на рисунке 4.1.

Сущность «Пользователь» предназначена для хранения информации о пользователях системы. Данные из этой таблицы используются при авторизации в приложении. Сущность содержит в себе следующие атрибуты:

− iduser – содержит идентификационный номер пользователя приложения(уникальный);

− login – содержит имя пользователя;

− password – содержит пароль, используемый для авторизации в системе;

− email – содержит адрес почты пользователя;

− real\_name – содержит настоящее имя пользователя, отображающееся на сайте;

− date\_of\_registration – дата регистрации на сайте;

− date\_of\_birth – дата рождения пользователя;

− status – содержит статус пользователя, например «заблокирован»;

− avatar – содерит ссылку на аватар пользователя;

− level\_points – количество очков уровня пользователя;

− role – хранит роль пользователя. По умолчанию роль соответствует «пользователю».

Сущность «Рейтинг» предназначена для хранения информации о рейтинге фильма. Сущность содержит в себе следующие атрибуты:

− idrating– содержит идентификационный номер записи(уникальный);

− iduser – содержит идентификационный номер пользователя, которому принадлежит оценка(уникальный);

− idfilm – содержит идентификационный номер фильма, которому предназначена оценка(уникальный);

− is\_seen – содержит информацию о том, просмотрен ли фильм;

− rating\_amount – содержит количественное выражение оценки;

Сущность «Фильм» предназначена для хранения информации о фильме. Сущность содержит в себе следующие атрибуты:

− idfilm– содержит идентификационный номер записи (уникальный);

− name – содержит название фильма;

− release\_year – содержит год выпуска фильма;

− duration – содержит длительность фильма в минутах;

− poster – содержит ссылку на постер;

− iddirector – содержитй идентификационный номер режиссера данного фильма (уникальный);

− description – содержит описание фильма;

Сущность «Отзыв» предназначена для хранения информации об отзыве. Сущность содержит в себе следующие атрибуты:

− idreview– содержит идентификационный номер записи (уникальный);

− iduser - содержит идентификационный номер пользователя, которому принадлежит отзыв (уникальный);

− idfilm - содержит идентификационный номер фильма (уникальный);

− text – содержит текст отзыва;

− date – содержит дату, когда был написан отзыв;

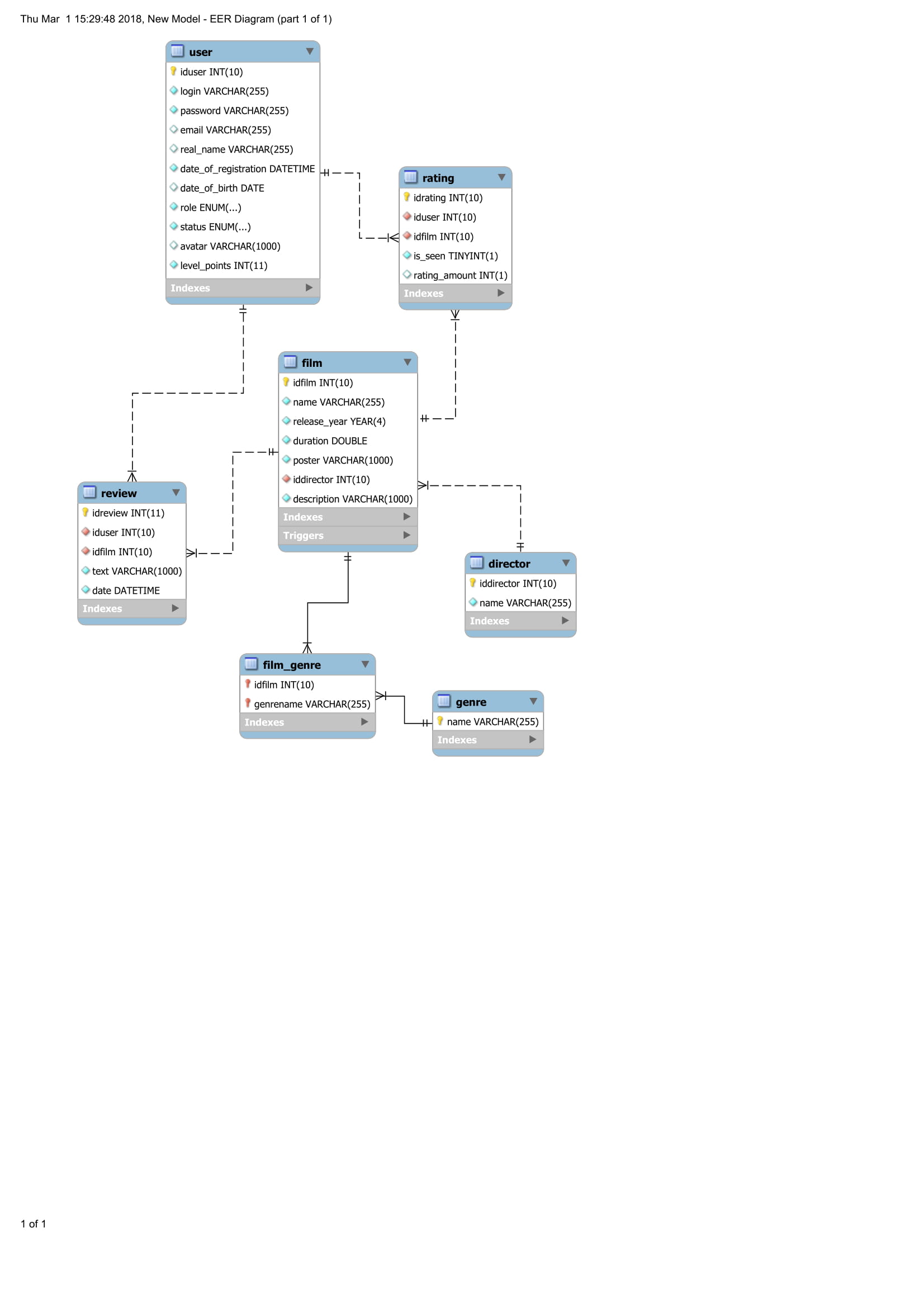


Рисунок 4.1 – Информационная модель системы

Сущность «Жанр» предназначена для хранения информации о жанре. Сущность содержит в себе следующие атрибуты:

− name– содержит название жанра;

Сущность «Режиссер» предназначена для хранения информации о режиссере. Сущность содержит в себе следующие атрибуты:

− iddirector– хранит уникальный идентификационный номер записи;

− name – хранит имя режиссера;

**5 ОБОСНОВАНИЕ ВЫБОРА КОМПОНЕНТОВ И ТЕХНОЛОГИЙ**

Для более подробного изучения и рассмотрения системы программы необходимо рассмотреть такие модели как:

* диаграмма развертывания;
* диаграмма последовательностей;
* диаграмма компонентов.

Система оценки кинофильмов реализована на операционной системе Windows 10 . На диаграмме развертывания (рисунок 5.1), показана конфигурация обрабатывающих узлов, на которых выполняется система, и компонентов, размещенных в этих узлах.

****

Рисунок 5.1 – Диаграмма развёртывания системы оценки кинофильмов

Диаграмма последовательности  — [диаграмма](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B0%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0), на которой для некоторого набора объектов на единой временной оси показан жизненный цикл какого-либо определённого объекта (создание-деятельность-уничтожение некой сущности) и взаимодействие актёров (действующих лиц) в рамках какого-либо определённого прецедента (отправка запросов и получение ответов)[5]. Представление диаграммы показано на рисунке 5.2.

Рисунок 5.2 – Диаграмма последовательностей входа в систему оценки кинофильмов

Диаграмма компонентов – диаграмма физического уровня, которая служит для представления программных компонентов и зависимостей между ними. Диаграмма компонентов разрабатывается для следующих целей: визуализация общей структуры исходного кода программной системы и спецификация исполнимого варианта программной системы[6] (рисунок 5.3).

****

Рисунок 5.3 – Диаграмма компонентов системы оценки кинофильмов

Диаграмма классов используется для визуального изображения отношений между классами и интерфейсами в программе. В приложении А представлена диаграмма классов приложения.

**6 МОДЕЛИ ПРЕДСТАВЛЕНИЯ СИСТЕМЫ И ИХ ОПИСАНИЕ**

**6.1 Диаграмма состояний**

Диаграмма состояний (state diagram) определяют все возможные состояния, в которых может находиться конкретный объект, а также процесс смены состояний объекта в результате влияния некоторых событий. Диаграммы состояний строятся для единственного класса и описывают поведение единственного объекта[4].

Диаграмма состояний представляет собой граф состояний в которых может находиться объект и связей между ними (рисунок 6.1).

**6.2 Алгоритм удаления данных из БД**

Алгоритм удаления данных о пользователях из БД состоит из нескольких шагов.

1. Формирование SQL-запроса в БД на удаление по выбранному id.
2. Выполнение запроса.
3. Выборка данных из БД.
4. Удаление данных.

Данный алгоритм представлен на рисунке 6.2.



Рисунок 6.1 – Диаграмма состояний при регистрации нового пользователя

**7 ОПИСАНИЕ ПРИМЕНЕНИЯ ПАТТЕРНОВ ПРОЕКТИРОВАНИЯ**

Часто в системе могут существовать сущности только в единственном экземпляре, например, система оценки кинофильмов. В таких случаях необходимо уметь создавать единственный экземпляр некоторого типа, предоставлять к нему доступ извне и запрещать создание нескольких экземпляров того же типа.Паттерн Singleton предоставляет такие возможности.

Архитектура паттерна Singleton основана на идее использования глобальной переменной, имеющей следующие важные свойства:

− такая переменная доступна всегда. Время жизни глобальной переменной - от запуска программы до ее завершения;

− предоставляет глобальный доступ, то есть, такая переменная может быть доступна из любой части программы.

Однако, использовать глобальную переменную некоторого типа непосредственно невозможно, так как существует проблема обеспечения единственности экземпляра, а именно, возможно создание нескольких переменных того же самого типа (например, стековых).

Для решения этой проблемы паттерн Singleton возлагает контроль над созданием единственного объекта на сам класс. Доступ к этому объекту осуществляется через статическую функцию-член класса, которая возвращает указатель или ссылку на него. Этот объект будет создан только при первом обращении к методу, а все последующие вызовы просто возвращают его адрес. Для обеспечения уникальности объекта, конструкторы и оператор присваивания объявляются закрытыми. UML-диаграмма классов паттерна Singleton можно увидеть на рисунке 6.1.

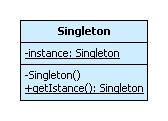


Рисунок 6.1 –Паттерн Синглтон

Достоинства паттерна Singleton:

Класс сам контролирует процесс создания единственного экземпляра, паттерн легко адаптировать для создания нужного числа экземпляров, а также возможность создания объектов классов, производных от Singleton.

Недостатки паттерна Singleton:

В случае использования нескольких взаимозависимых одиночек их реализация может резко усложниться. [7].

Паттерн Command преобразовывает запрос на выполнение действия в отдельный объект-команду. Такая инкапсуляция позволяет передавать эти действия другим функциям и объектам в качестве параметра, приказывая им выполнить запрошенную операцию. Команда – это объект, поэтому над ней допустимы любые операции, что и над объектом.

Интерфейс командного объекта определяется абстрактным базовым классом Command и в самом простом случае имеет единственный метод execute(). Производные классы определяют получателя запроса (указатель на объект-получатель) и необходимую для выполнения операцию (метод этого объекта). Метод execute() подклассов Command просто вызывает нужную операцию получателя. UML-диаграмма классов паттерна Command можно увидеть на рисунке 6.2.

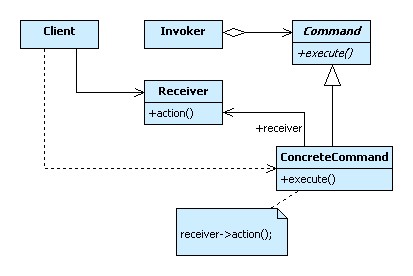


Рисунок 6.2 –Паттерн Команда

Для того, чтобы система оставалась независимой от различных типов объектов, паттерн Factory Method использует механизм полиморфизма - классы всех конечных типов наследуют от одного абстрактного базового класса, предназначенного для полиморфного использования. В этом базовом классе определяется единый интерфейс, через который пользователь будет оперировать объектами конечных типов.

Для обеспечения относительно простого добавления в систему новых типов паттерн Factory Method локализует создание объектов конкретных типов в специальном классе-фабрике. Методы этого класса, посредством которых создаются объекты конкретных классов, называются фабричными. Существуют две разновидности паттерна Factory Method:

Обобщенный конструктор, когда в том же самом полиморфном базовом классе, от которого наследуют производные классы всех создаваемых в системе типов, определяется статический фабричный метод. В качестве параметра в этот метод должен передаваться идентификатор типа создаваемого объекта. UML-диаграмма классов паттерна Factory Method можно увидеть на рисунке 6.3.

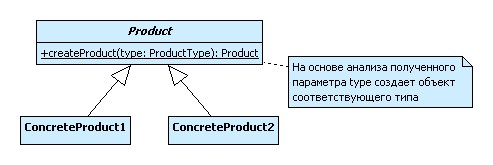


Рисунок 6.3 –Паттерн Фабричный метод

Достоинства паттерна Factory Method:

− создает объекты разных типов, позволяя системе оставаться независимой как от самого процесса создания, так и от типов создаваемых объектов.

Недостатки паттерна Factory Method:

− в случае классического варианта паттерна даже для порождения единственного объекта необходимо создавать соответствующую фабрику.

**8 РУКОВОДСТВО ПО РАЗВЕРТЫВАНИЮ СИСТЕМЫ**

Для того, чтобы развернуть онлайн-сервис оценки кинофильмов необходимо выполнить нижеперечисленные шаги.

Во-первых необходимо осуществить подключение к базе данных. В данной системе используется СУБД MySQL и для правильности взаимодействия сервиса с базой данных необходимо настроить некоторые параметры.

Для начала необходимо скачать и установить MySQL. Для создания базы данных для разрабатываемой системы необходимо выполнить скрипт, находящийся в приложении под названием db\_script.sql . После этого необходимо проверить настройки для доступа к базе в файле project.properties, находящемся в movierating/src/main/resources. В случае несовпадения, исправить на корректные.

Во-вторых необходимо загрузить сервер на хостинг. Для этого необходимо скачать Apache Maven и Apache Tomcat и запустить проект уже с помощью Tomcat.

Из всего вышеперечисленного следует, что для корректной работы и обеспечения полного функционирования системы необходимы такие инструменты, как СУБД MySQL, Apache Maven и Apache Tomcat.

**9 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ И ОЦЕНКА ВЫПОЛНЕНИЯ ЗАДАЧ**

**ЗАКЛЮЧЕНИЕ**

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

**ПРИЛОЖЕНИЕ А**



Рисунок А.1 – Диаграмма классов пакета command

**Продолжение приложения А**



Рисунок А.2 – Диаграмма классов пакета config

**Продолжение приложения А**



Рисунок А.3 – Диаграмма классов пакета controller



Рисунок А.4 – Диаграмма классов пакета dao

**Продолжение приложения А**



Рисунок А.5 – Диаграмма классов пакета connection

**Продолжение приложения А**



Рисунок А.6 – Диаграмма классов пакета entity

**Продолжение приложения А**



Рисунок А.7 – Диаграмма классов пакета entity

**Продолжение приложения А**



Рисунок А.8 – Диаграмма классов пакета exception



Рисунок А.9 – Диаграмма классов пакета filter

**Продолжение приложения А**



Рисунок А.10 – Диаграмма классов пакета listener



Рисунок А.11 – Диаграмма классов пакета util

**Продолжение приложения А**



Рисунок А.12 – Диаграмма классов пакета service