Содержание

[Введение 3](#_Toc132249075)

[1 Аналитический обзор существующих методов и средств автоматизации 5](#_Toc132249076)

[1.1 Анализ предметной области 5](#_Toc132249077)

[1.2 Аналитический обзор существующих аналогов 6](#_Toc132249078)

[1.3 Обзор технологий для реализации программного обеспечения 7](#_Toc132249079)

[1.4 Постановка задачи 9](#_Toc132249080)

[2 Проектирование программного обеспечения 10](#_Toc132249081)

[2.1 Функциональное моделирование 10](#_Toc132249082)

[2.2 Информационное обеспечение приложения 10](#_Toc132249083)

[2.3 Архитектура приложения 10](#_Toc132249084)

[3 Структура и реализация программного обеспечения 11](#_Toc132249085)

[3.1 Описание реализации программного комплекса 11](#_Toc132249086)

[3.2 Описание пользовательского интерфейса 11](#_Toc132249087)

[4 Тестирование программного обеспечения 12](#_Toc132249088)

[4.1 Модульное тестирование 12](#_Toc132249089)

[4.2 Ручное тестирование программного комплекса 12](#_Toc132249090)

[Заключение 13](#_Toc132249091)

[Список использованных источников 14](#_Toc132249092)

# ВВЕДЕНИЕ

Информационные технологии оказывают значительное влияние на человеческую деятельность в различных сферах жизни. Они позволяют людям быстро и эффективно обмениваться информацией, автоматизировать процессы и повышать производительность труда.

В бизнесе информационные технологии используются для управления бухгалтерскими и финансовыми операциями, управления производственными процессами, улучшения маркетинговых стратегий и управления ресурсами. Информационные технологии также позволяют компаниям создавать новые продукты и услуги, которые могут быть доступны для потребителей через интернет.

Информационная система представляет собой совокупность методов и средств, которые используются для хранения и обработки информации. Существует множество типов и вариаций информационных систем.

Наиболее используемым типом информационной системы является клиент-серверная система. Данный тип системы представляет собой взаимодействие структурных компонентов, где структурными компонентами являются сервер и узлы-поставщики определённых сервисов, а также клиенты, которые пользуются данным сервисом. Данный тип системы наиболее часто используется в создании корпоративных баз данных, в которой база данных является главным элементом, а все необходимые операции с базой выполняются сервером. Запросы на получение и изменение информации из базы данных отправляют клиенты. Сервер обрабатывает запросы и возвращает ответ клиенту. Преимуществом такой системы является её достаточно высокий уровень производительности за счёт распределения вычислительной нагрузки между клиентом и сервером, а также непротиворечивость данных за счёт централизованной обработки.

Целью данной курсовой работы является проектирование и создание базы данных в СУБД и разработка веб-приложения, которое обеспечивает отображение, редактирование и обработку информации из разработанной базы данных.

Для решения поставленной задачи в качестве СУБД используется *MS Sql Server*. Данная СУБД обеспечивает поддержку баз данных очень большого объёма и обработку сложных запросов, а также имеет эффективные алгоритмы для работы с памятью и автоматизированным контролем размера файлов баз данных.

В качестве технологии для разработки веб-приложения используется платформа *ASP.NET Core MVC*. Данная платформа является многофункциональной платформой для создания веб-приложений с помощью структуры проектирования *Model-View-Controller* (модель-контроллер-представление). Структура архитектуры *MVC* предполагает разделение приложения на три основных компонента: модель, представление и контроллер. Каждый компонент решает свои задачи и взаимодействует с другими компонентами.

Одной из главных преимуществ *ASP.NET* является его интеграция с *Visual Studio*, интегрированной средой разработки (*IDE*) от *Microsoft*. *Visual Studio* предоставляет множество инструментов для разработки приложений на *ASP.NET*, включая отладчик, инструменты тестирования и управление версиями.

# 1 АНАЛИТИЧЕСКИЙ ОБЗОР СУЩЕСТВУЮЩИХ МЕТОДОВ И СРЕДСТВ АВТОМАТИЗАЦИИ

## **1.1 Анализ предметной области**

Современный мир стремительно развивается в сторону цифровой экономики, в которой цифровой контент играет ключевую роль. Онлайн сервисы распространения цифрового контента предоставляют возможность создания, хранения и распространения цифрового контента в электронном виде. Это позволяет пользователям получать доступ к контенту из любой точки мира, а также авторам быстро и эффективно распространять свои произведения.

Одной из главных причин необходимости онлайн сервисов распространения цифрового контента является удобство использования. Пользователи могут получать доступ к контенту из любой точки мира, без необходимости покупки физических носителей. Это экономит время и деньги.

Второй причиной необходимости онлайн сервисов распространения цифрового контента является возможность масштабирования. Онлайн сервисы позволяют авторам распространять свои произведения на мировом уровне, что делает их доступными для большего числа пользователей. Это в свою очередь позволяет авторам получать больше прибыли от своих произведений.

Третьей причиной необходимости онлайн сервисов распространения цифрового контента является возможность управления контентом. Онлайн сервисы позволяют авторам контролировать распространение своих произведений и защищать их от незаконного использования. Это помогает сохранять авторские права.

Четвертой причиной необходимости онлайн сервисов распространения цифрового контента является экономическая выгода. Онлайн сервисы позволяют сократить затраты на производство, хранение и распространение контента.

В целом, онлайн сервисы распространения цифрового контента являются необходимыми в современном мире. Они обеспечивают удобство использования, возможность масштабирования, управления контентом, и экономическую выгоду. Это делает их привлекательными для пользователей и авторов, что способствует развитию цифровой экономики в целом.

Для разработки онлайн сервиса распространения цифрового контента необходимо четкое понимание предметной области.

В ходе курсового проектирования будет разрабатываться информационная система, которая позволит распространять такой цифровой контент как фильмы.

Обзор темы показал, что для работы с приложением необходимо будет определить ролевую политику, где каждой роли будет соответствовать свой функционал.

К функционалу Менеджера будет относиться:

* добавление, обновление, удаление фильмов;
* добавление, обновление, удаление актеров;
* добавление, обновление, удаление кинотеатров;
* добавление, обновление, удаление продюсеров;
* просмотр всех пользователей в системе;
* просмотр всех заказов в системе.

К функционалу Пользователя будет относиться:

* поиск фильма;
* просмотр подробной информации о фильме;
* добавление фильма в корзину;
* покупка фильмов;
* обновление профиля;
* просмотр совершенных заказов.

## **1.2 Аналитический обзор существующих аналогов**

Аналог должен иметь идентичное назначение и область применения по сравнению с проектируемым объектом и давать качественно одинаковый результат их использования.

В качестве аналога веб-приложения будут рассматриваться кинотеатры.

Кинотеатры и цифровые приложения — это два разных способа потребления кино. В настоящее время все больше людей предпочитают смотреть фильмы через интернет, используя различные приложения и сервисы. Однако, несмотря на это, кинотеатры продолжают оставаться популярными местами для просмотра новых фильмов.

Первое и, для многих, незначительное отличие заключается в том, что кинотеатры предоставляют возможность просмотра фильмов на большом экране в комфортабельных условиях с звуком высокого качества. Это позволяет зрителю полностью погрузиться в атмосферу фильма и насладиться им в полной мере. В то же время, просмотр фильмов через цифровые приложения может быть ограничен размером экрана и качеством звука.

Кроме того, в кинотеатрах обычно идет показ фильмов в определенное время, что позволяет заранее запланировать поход в кино. В то же время, приложения позволяют просматривать фильмы в любое время и в любом месте, где есть доступ к интернету. Но даже при отсутствии интернета, в приложениях есть возможность скачивания контента.

Еще одно отличие заключается в том, что приложения могут предоставлять более широкий выбор фильмов, чем кинотеатры. Благодаря этому пользователь может выбрать любимый жанр и насладиться просмотром фильма, который ему действительно интересен. В то же время, в кинотеатрах выбор фильмов может быть ограничен, и не всегда можно найти нужный фильм в нужное время.

Кроме того, зачастую перед походом в кинотеатр пользователь ищет информацию об интересующем его фильме. Это может быть информация о съемочной группе или простой просмотр трейлера. Во время покупки билета в кинотеатры такая информация не предоставляется. Веб-ресурсы же предоставляют всю необходимую информацию в одном месте, и перед покупкой можно ознакомиться со всей интересующей информацией, даже не выходя из приложения.

Кроме того, цифровые приложения могут предоставлять дополнительные возможности для просмотра фильмов. Например, некоторые приложения позволяют смотреть фильмы с субтитрами или на других языках. Это может быть полезно для тех, кто изучает иностранные языки или просто хочет наслаждаться кино на другом языке.

Однако, несмотря на все преимущества цифровых приложений, кинотеатры остаются популярными местами для просмотра фильмов. Они предоставляют уникальный опыт, который невозможно воспроизвести дома или через приложение.

В заключение можно сказать, что кинотеатры и цифровые приложения — это два разных способа просмотра кино, каждый из которых имеет свои преимущества и недостатки. Кинотеатры предоставляют уникальный опыт просмотра фильмов на большом экране с высококачественным звуком и атмосферой кинотеатра. В то же время, цифровые приложения позволяют просматривать фильмы в любое время и в любом месте и предоставляют более широкий выбор фильмов.

## **1.3 Обзор технологий для реализации программного обеспечения**

Для решения поставленной задачи в качестве СУБД используется *Ms SQL Server*. Данная СУБД обеспечивает поддержку баз данных очень большого объёма и обработку сложных запросов, а также имеет эффективные алгоритмы для работы с памятью и автоматизированным контролем размера файлов баз данных.

В качестве технологии для разработки веб-приложения используется платформа *ASP.NET*.

*ASP.NET* является одной из самых популярных платформ для создания веб-приложений. Она была разработана компанией *Microsoft* и является основой для создания многих веб-сайтов и приложений, которые мы используем каждый день.

*ASP.NET* предоставляет разработчикам мощный инструментарий для создания профессиональных веб-приложений. Он использует языки программирования, такие как *C#* и *Visual Basic*, а также другие технологии *Microsoft*, такие как *SQL Server* и *Windows Server*.

Одной из особенностей *ASP.NET* является его модель программирования на основе событий. В этой модели веб-страницы и компоненты реагируют на события, такие как щелчок мыши или изменение текста в поле ввода, вызывая определенный код. Это позволяет создавать более динамические и интерактивные приложения.

*ASP.NET* также обеспечивает масштабируемость приложений. Система автоматически управляет выделением ресурсов и распределением нагрузки на серверах, что позволяет обеспечивать высокую производительность даже при большом количестве пользователей [1].

Одним из ключевых компонентов *ASP.NET* является его фреймворк *MVC* (*Model-View-Controller*). Он предоставляет разработчикам гибкую архитектуру, которая позволяет разделить приложение на три отдельных слоя: модель, представление и контроллер. Это упрощает разработку и поддержку приложений.

С помощью *ASP.NET* разработчики могут создавать различные типы приложений, включая веб-сайты, веб-службы, веб-*API* и даже мобильные приложения.

Одной из главных преимуществ *ASP.NET* является его интеграция с *Visual* *Studio*, интегрированной средой разработки (*IDE*) от *Microsoft. Visual Studio* предоставляет множество инструментов для разработки приложений на *ASP.NET,* включая отладчик, инструменты тестирования и управление версиями.

*ASP.NET* также поддерживает различные сторонние библиотеки и плагины, которые могут расширить его функциональность. Некоторые из этих библиотек включают в себя *jQuery, AngularJS* и *Bootstrap*.

Для доступа к данным будет использована технология *Entity Framework Core*.

*Entity Framework* (*EF*) — это технология объектно-ориентированного доступа к данным для .*NET* *Framework*. Она предоставляет удобный способ работать с базами данных, используя объекты .*NET*, а не *SQL*-запросы. EF позволяет разработчикам работать с данными на более высоком уровне абстракции, что упрощает процесс разработки и обслуживания приложений.

Одной из ключевых концепций *Entity Framework* является *ORM* (*Object*-*Relational Mapping*). *ORM* — это технология, которая позволяет разработчикам работать с базой данных, используя объекты .*NET. EF* предоставляет *ORM*-функциональность, которая позволяет создавать и обновлять таблицы в базе данных, а также выполнять запросы к данным. Она также позволяет использовать *LINQ (Language Integrated Query)* для выполнения запросов к данным.

*EF* также предоставляет *Code First* подход к разработке приложений. *Code* *First* — это способ создания базы данных, который позволяет разработчикам определять модели данных в коде и затем автоматически создавать базу данных на основе этих моделей. Это упрощает процесс разработки и уменьшает количество кода, необходимого для создания базы данных

## **1.4 Постановка задачи**

Необходимо разработать веб-приложение, которое содержит в себе следующий функционал:

* содержание информации о фильмах;
* содержание информации об актерах;
* содержание информации о кинотеатрах;
* содержание информации о продюсерах;
* содержание информации о всех пользователях;
* содержание информации о всех заказах.

На основании произведённого анализа существующих методов, можно сформировать список требований, для разрабатываемой системы:

* приложение не должно иметь лишнего функционала;
* доступ к приложению должен быть удобным и простым.

Программный продукт должен предоставлять:

* справочники, включающие в себя фильмы, актеры, кинотеатры и продюсеры;
* формирование отчетов, статистики;
* необходимо разделение доступа к функционалу приложения по ролям: Менеджер, Пользователь.

# 2 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## **2.1 Функциональное моделирование**

## **2.2 Информационное обеспечение приложения**

## **2.3 Архитектура приложения**

# 3 СТРУКТУРА И РЕАЛИЗАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## **3.1 Описание реализации программного комплекса**

## **3.2 Описание пользовательского интерфейса**

# 4 ТЕСТИРОВАНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

## **4.1 Модульное тестирование**

## **4.2 Ручное тестирование программного комплекса**

# ЗАКЛЮЧЕНИЕ

# Список использованных источников (не оформленные)

1. [**https://www.onestopdevshop.io/net-framework/top-front-end-and-back-end-asp-net-frameworks/#What\_Is\_ASPNET\_-With\_Example**](https://www.onestopdevshop.io/net-framework/top-front-end-and-back-end-asp-net-frameworks/#What_Is_ASPNET_-With_Example)
2. **Троелсен, Э. (2013). ASP.NET 4.5 на примерах (5-е издание). Питер.**