**Частное Образовательное учреждение высшего образования «Международный Институт Дизайна и Сервиса»**

**(ЧОУВО МИДиС)**

Кафедра математики и информатики

**КУРСОВАЯ РАБОТА**

по дисциплине «Прикладная информатика»

на тему: «АСУ Кинотеатр»

Направление (специальность) разработка мобильных и десктопных приложений

Научный руководитель

Кандидат математических наук

Чеботарёв Сергей Сергеевич

Автор работы

студент группы ПИ-225

Сергеев Павел Дмитриевич

Челябинск 2022

**СОДЕРЖАНИЕ**

**Введение**…………………………………………………………………3

**Глава I**. **Анализ предметной области**

1.1 Актуальность проблемы……………………………………………5

1.2 Словарь предметной области………………………………………6

1.3 Формирование технического задания………………………..……7

**Глава II**. **Проектирование**

2.1 Анализ бизнес-процессов…………..................................................8

2.2 Моделирование базы данных……………………………...............10

2.3 Макеты экранных форм………………………………………...….15

**Глава III**. **Разработка**

3.1 Обоснование выбранных инструментов…………………....….....21

3.2 Архитектура программы........………..............................................22

3.3 Руководство по установке и использованию...………………………………………..........................51

**Заключение.** …………………………………………………………...52

**Список использованных источников**………………………...….…53

**Введение**

В настоящее время в сфере информационных технологий происходит активное развитие и внедрение новых технологий. Сегодня уже во многих областях деятельности человека можно доверить некоторую часть работы компьютеру, ­­­­­­­­­­­­­­­­­­­­­­­ потому что они могут сделать эту работу не хуже человека, а в некоторых­­­­­­ случаях лучше. По скорости работы персональные компьютеры намного быстрее человека могут произвести математические вычисления, поиск информации и решение каких-либо задач. Но без программного обеспечения компьютеры бесполезны. Для каждой области деятельности человека необходимо свое программное обеспечение. Сегодня без этих помощников не обходиться ни одна компания, ни одно предприятие.

На сегодняшний день на рынке программного обеспечения отмечается устойчивый интерес к программным продуктам, способным обеспечить эффективное управление различной области компаний и предприятий. Большинство руководителей понимают, что для успешной организации производственного процесса необходимо применять современные информационные технологии.

В связи с глобальной информатизацией всех областей жизнедеятельности человека, а также с растущим уровнем потребности к программным продуктам, способным обеспечить эффективное управление производственным процессом, предоставляется возможность создания программного средства, решающего поставленную задачу, применяя современные информационные технологии.

Заданием на курсовое проектирование является разработка программного средства для автоматизации работы кассира и администратора в кинотеатре, которое должно обеспечивать ведение базы данных о фильмах и сеансах, билетах и брони на них, залах, персонале и пользователях, осуществлять формирование различных отчётов, выполнять продажу билетов на сеанс.

Автоматизация перечисленных процессов исключит возникновение ошибок при работе с данными, избавит от больших объемов различной документации и ускорит работу.

В результате работы над курсовым проектом будет разработано программное средство и составлена пояснительная записка, которая включает в себя три раздела, содержащих необходимую и достаточную информацию по использованию данного программного средства, подробное описание программы и вычислительной системы.

В заключении про анализируется созданное программное приложение, определится степень соответствия поставленной задачи и выполненной работы.

1. **Анализ предметной области**

**1.1 Актуальность проблемы**

Предметной областью решаемой задачи является работа кассира и администратора в кинотеатре. Такая работа связана с накоплением большого количества информации сеансах, билетах.

Традиционно информация храниться на бумажных носителях. При этом трудно осуществить быстрый поиск и исправление ошибочно указанных данных, а также продажа билетов занимает много времени.

Таким образом, автоматизация работы кассира и администратора в кинотеатре, является актуальной и перспективной.

В функции кассира входят:

* Продажа билетов на сеансы;
* Бронирование билетов на сеансы.

Функции администратора:

* Работа с сеансами;
* Работа с фильмами;
* Работа с залами;
* Работа с персоналом;
* Работа с пользователями;
* Просмотр отчетов.
* Работа с ценами на билеты.

Из всех задач, которые будет решать разрабатываемое программное средство, можно выделить ряд основных:

* хранить данные о сеансах, билетах, фильмах, залах, персонале и пользователях;
* Добавление, редактирование и удаление данных о сеансах, билетах, фильмах, залах, персонале и пользователях;
* осуществлять продажу и бронирование билетов на сеанс;
* формировать отчеты в excel формате;
* сортировку и распечатку данных.

Данное программное средство направлено на повышение эффективности, а также скорости работы кассира и администратора.

В настоящее время существует множество аналогичных программ, предназначенные для составления и ведения баз данных. Однако разработка программы, направленной на решение задачи в определённой сфере, не только облегчает труд кассира и администратора, но и обеспечивает правильность и безопасность хранения и редактирования данных, обрабатываемых этой программой.

**1.2 Словарь предметной области**

Система будет оперировать такими сущностями как:

Кинотеатр – Место просмотра фильмов.

Пользователь – Человек, который бронирует билет на сеанс.

Кассир – Человек, который продает билеты.

Администратор – Человек, который управляет сеансами, пользователями, фильмами, залами, ценами за билеты.

Сеанс – Демонстрация фильма зрителям.

Билет – Документ, дающий право пройти на конкретный сеанс.

Бронь – Заказ места в кинотеатре заранее.

Залы – Помещение, в котором люди смотрят фильм.

Фильм – Совокупность движущихся изображений, связанных единым сюжетом.

**1.3 Формирование технического задания**

Для данной системы планируется сделать:

* Добавление, редактирование и удаление билетов, брони, залов, пользователей, сеансов и фильмы.
* Экспорт данных в excel.
* Распечатку для билетов и данных.
* Авторизацию и регистрацию новых пользователей.

1. **Проектирование**

**2.1 Анализ бизнес-процессов**

Разрабатываемое приложение будет защищено от несанкционированного доступа паролем для того, чтобы другой пользователь не смог получить доступ к базе данных.

Доступ к структуре базы данных должен быть ограничен для неопытных пользователей в целях сохранения структуры базы данных.

Разрабатываемое программное средство должно иметь понятный и удобный в использовании интерфейс, чтобы взаимодействие между программой и пользователем было максимально упрощено.

Все записи протокола должны исключать возможность ввода некорректных данных.

Кроме этого, при разработке форм необходимо соблюдать определённые требования: формы в приложении должны иметь стандартные элементы управления данными, не должны быть перегружены информацией, понятны простому пользователю. При конструировании форм в необходимых случаях нужно предусмотреть возможность защиты данных от изменения, установить ограничения на корректировку записей, а также обработку на отсутствие данных при выполнении отбора записей в базе данных.

Из всех задач, которые будет решать разрабатываемое программное средство, можно выделить ряд основных требований:

* Продажа билетов;
* Бронирование билетов
* Работа с базой данных.

Исходя из исследования предметной области задачи принято решение об организации программного средства в виде базы данных MS SQL SERVER. В программе предполагается создание навигационного меню для администратора, состоящего из пунктов «Фильмы», «Залы», «Сеансы», «Цена», «Пользователи», «Общие сведения».

Пункт меню «Фильмы» будет вызывать окно с фильмами, где пользователь сможет осуществить просмотр, добавить, удалить и редактировать фильмы, а также произвести экспорт данных в excel.

Пункт меню «Залы» будет вызывать окно с залами, где пользователь сможет осуществить редактирование, добавление и удаление залов.

Пункт меню «Сеансы» будет вызывать окно с сеансами, где пользователь сможет добавить, редактировать и удалить сеансы.

Пункт меню «Цена» будет вызывать окно с ценами на билеты, где пользователь сможет осуществить просмотр и редактировать цену билетов.

Пункт меню «Пользователи» будет вызывать окно со всеми пользователями, где администратор сможет осуществить просмотр, добавить и редактировать логины и уровень доступа пользователя.

Пункт меню «Общие сведения» будет вызывать окно, на котором находятся все данные, у которых есть возможность распечатки.

Также в программе предполагается создание навигационного меню для кассира, состоящего из пунктов «Билеты», «Информация о сеансах», «Информация о бронировании».

Пункт меню «Билеты» будет вызывать окно с продажей билетов, где кассир сможет осуществить продажу, а также просмотр, редактирование, удаление и распечатку билетов.

Пункт меню «Информация о сеансах» будет вызывать окно с информацией о сеансах.

Пункт меню «Информация о бронировании» будет вызывать окно с информацией о бронировании билетов.

**2.2 Моделирование базы данных**

Организация данных подразумевает создание модели данных, главными элементами которой являются сущности и их связи.

Реляционная модель основана на математическом понятии отношения, представлением которого является таблица. В реляционной модели отношения используются для хранения информации об объектах, представленных в базе данных. Отношение имеет вид двухмерной таблицы, в которой строки соответствуют записям, а столбцы - атрибутам. Каждая запись должна однозначно характеризоваться в таблице. Для этого используют первичные и вторичные ключи. Достоинством реляционной модели является простота и удобство физической реализации.

Реляционная модель базы данных подразумевает нормализацию всех таблиц данных. Нормализация — это формальный метод анализа отношений на основе их первичного ключа и функциональных зависимостей, существующих между их атрибутами.

Структура базы данных разрабатываемого программного средства включает Восемь таблиц. Их структура, и краткое описание приводится в таблицах 2.1-2.8.

Таблица «Билеты» предназначена для хранения информации о билетах.

Таблица 2.1 – Структура таблицы «Билеты»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип  поля | Описание поля |
| ID | int | Код билета |
| IDСеанса | int | Код сеансы |
| IDЗала | int | Код зала |
| Ряд | int | Ряд |
| Место | int | Место |
| Бронь | bit | Бронировали или нет |

Таблица «Бронь» предназначена для хранения информации о забронированных билетах.

Таблица 2.2 – Структура таблицы «Бронь»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип  поля | Описание поля |
| ID | int | Код |
| IDБилета | int | Код билета |
| ФИО | nvarchar | ФИО бронирующего |
| Телефон | nvarchar | Телефон бронирующего |

Таблица «Залы» предназначена для хранения информации о залах.

Таблица 2.3 – Структура таблицы «Залы»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип  поля | Описание поля |
| ID | int | Код зала |
| НомерЗала | int | Номер зала |
| IDРазмера | int | Номер его размера |

Таблица «Пользователи» предназначена для хранения информации о персонале.

Таблица 2.4 – Структура таблицы «Пользователи»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип  поля | Описание поля |
| ID | int | Код пользователя |
| Логин | varchar | Логин пользователя |
| Пароль | varchar | Пароль пользователя |
| УровеньДоступа | varchar | Уровень доступа пользователя |

Таблица «РазмерыЗалов» предназначена для хранения информации о размерах залов.

Таблица 2.5 – Структура таблицы «РазмерыЗалов»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип  поля | Описание поля |
| ID | int | Код |
| Наименование | varchar | Название зала |
| Количество рядов | varchar | Количество рядов в зале |

Таблица «Сеансы» предназначена для хранения информации о сеансах.

Таблица 2.6 – Структура таблицы «Сеансы»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип  поля | Описание поля |
| ID | int | Код сеанса |
| IDФильма | int | Номер фильма |
| IDЗала | int | Номер зала |
| Дата | nvarchar | Дата сеанса |
| Время | nvarchar | Время сеанса |
| Премьера | nvarchar | Премьера это или нет |

Таблица «СтоимостьБилетов» предназначена для хранения информации о стоимости билетов.

Таблица 2.7 – Структура таблицы «СтоимостьБилетов»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип  поля | Описание поля |
| ID | int | Код |
| IDСеанса | int | Номер сеанса |
| Стоимость | int | Стоимость билета |

Таблица «Фильмы» предназначена для хранения информации о фильмах.

Таблица 2.8 – Структура таблицы «Фильмы»

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип  поля | Описание поля |
| ID | int | Код |
| Название | nvarchar | Название |
| Жанр | nvarchar | Жанр |
| Длительность | nvarchar | Длительность |
| Год | int | Год |
| Страна | nvarchar | Страна |
| Авторы | nvarchar | Авторы |
| Описание | nvarchar | Описание |

Структура базы данных представлена в схеме на рисунке 2.9:

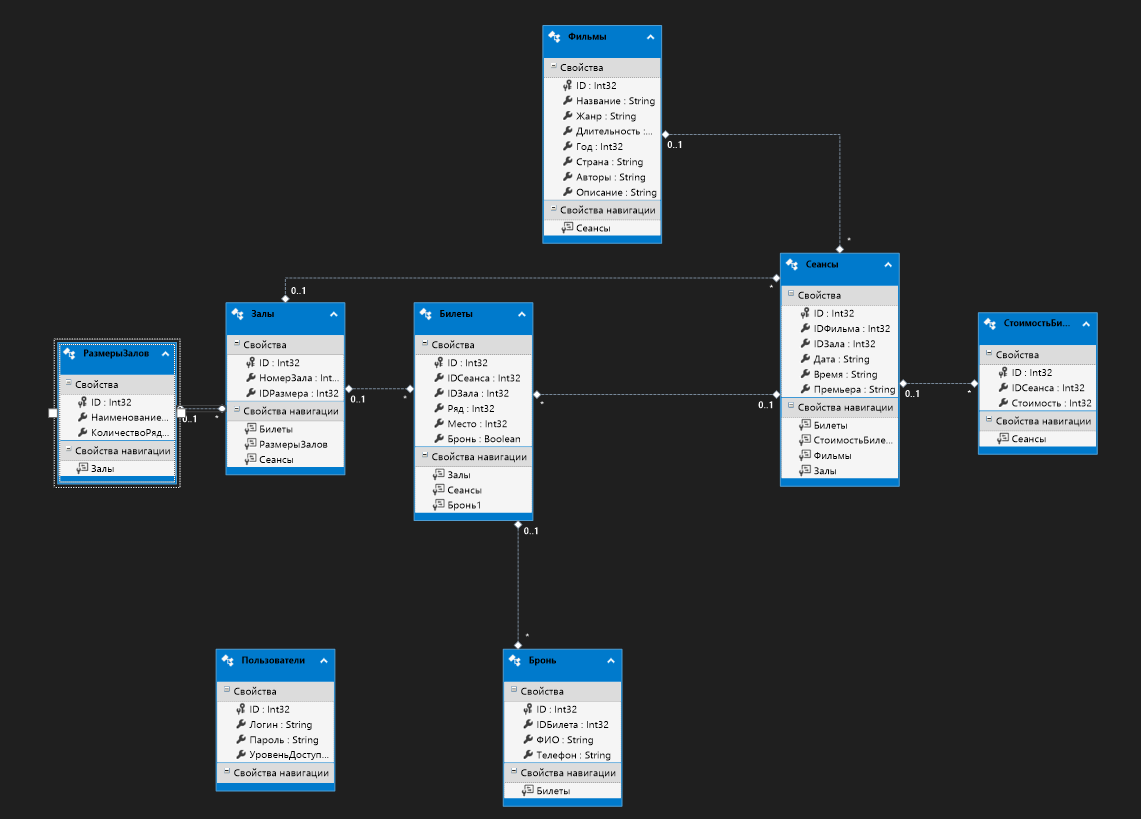


Рисунок 2.9 – Схема данных

**2.3 Макеты экранных форм**

Почти все программы построены из окон. Они необходимы для удобства управления программой. Не будь их, обычному пользователю было бы намного труднее пользоваться программой или вообще невозможно.

Далее будет предоставлен макет каждой экранной формы данного проекта.

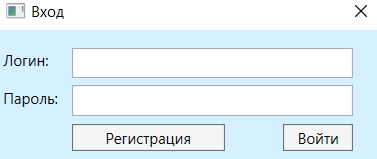


Рисунок 1 – Авторизация

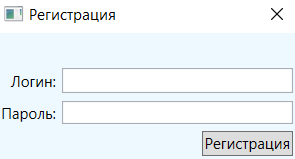


Рисунок 2 – Регистрация нового пользователя

Окна администрации:

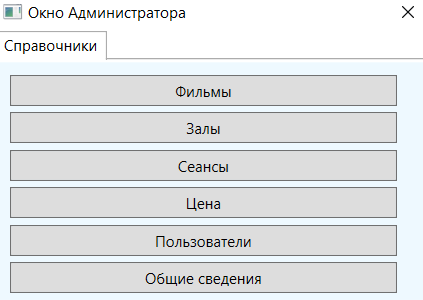


Рисунок 3 – Окно администратора кинотеатра

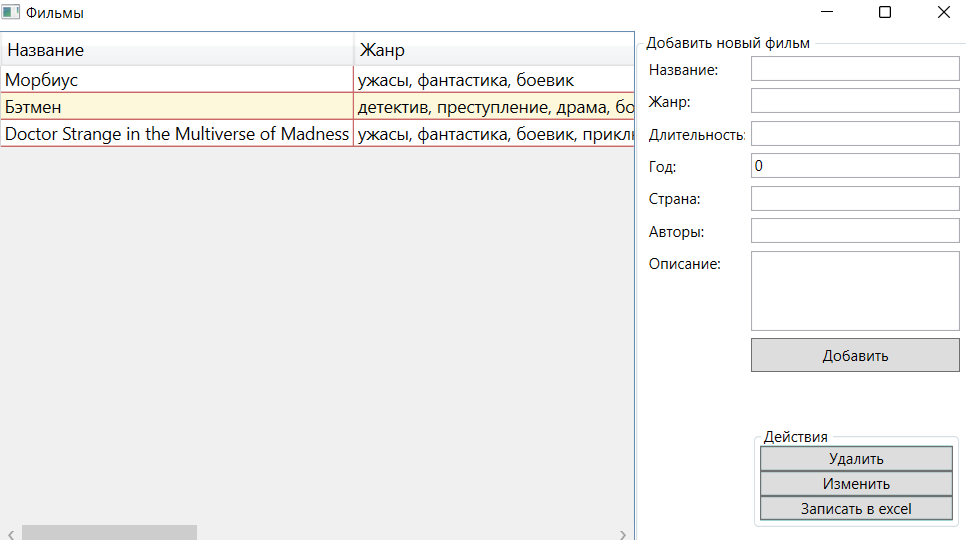


Рисунок 4 – Окно фильмов

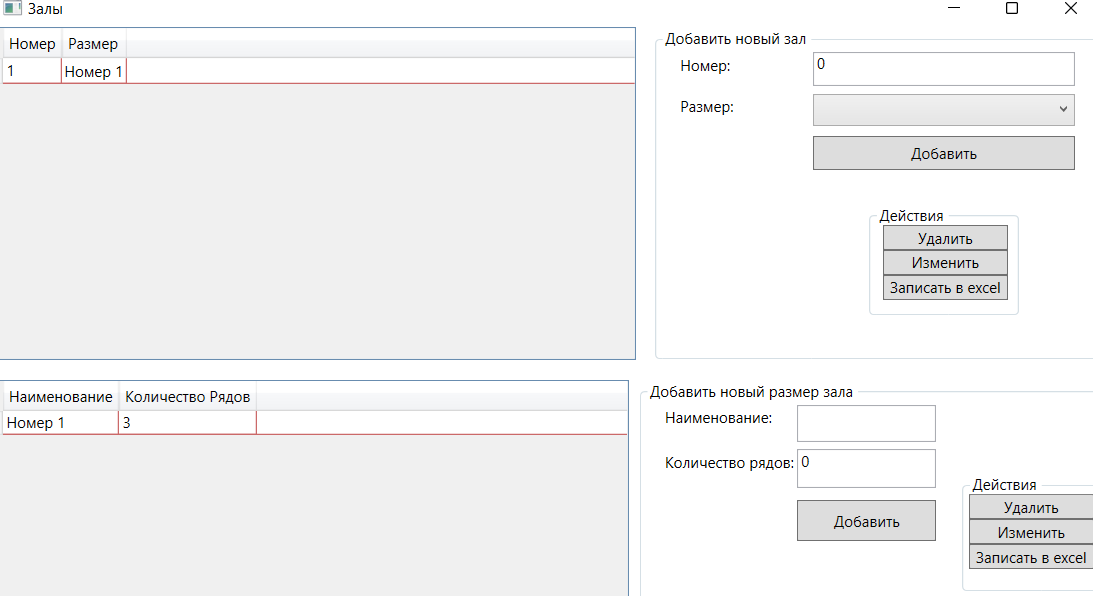


Рисунок 5 – Окно залов

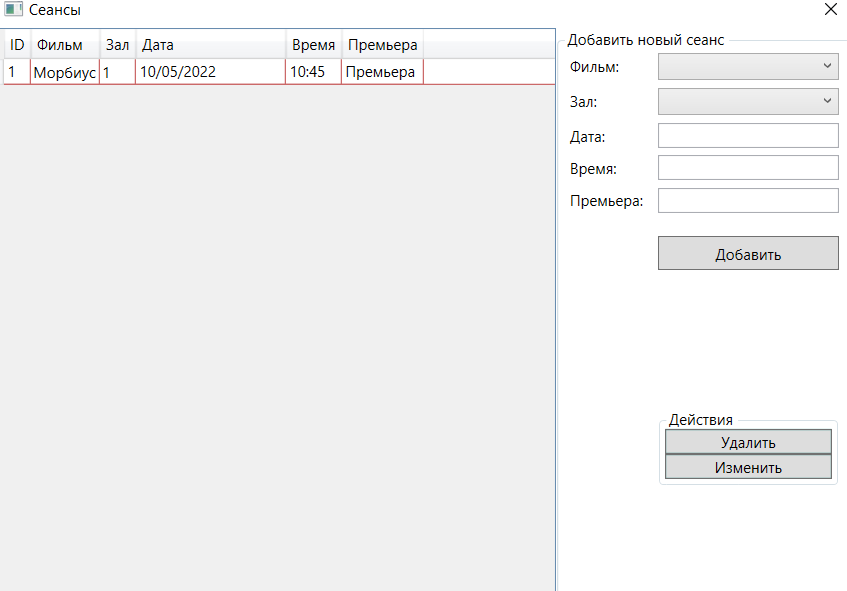


Рисунок 6 – Окно сеансов

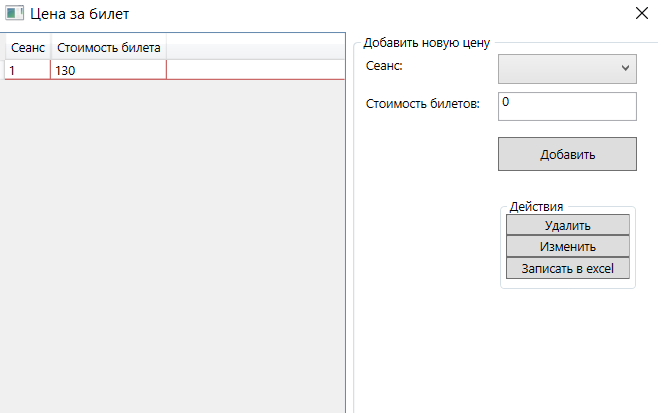


Рисунок 7 – Окно цен на билеты

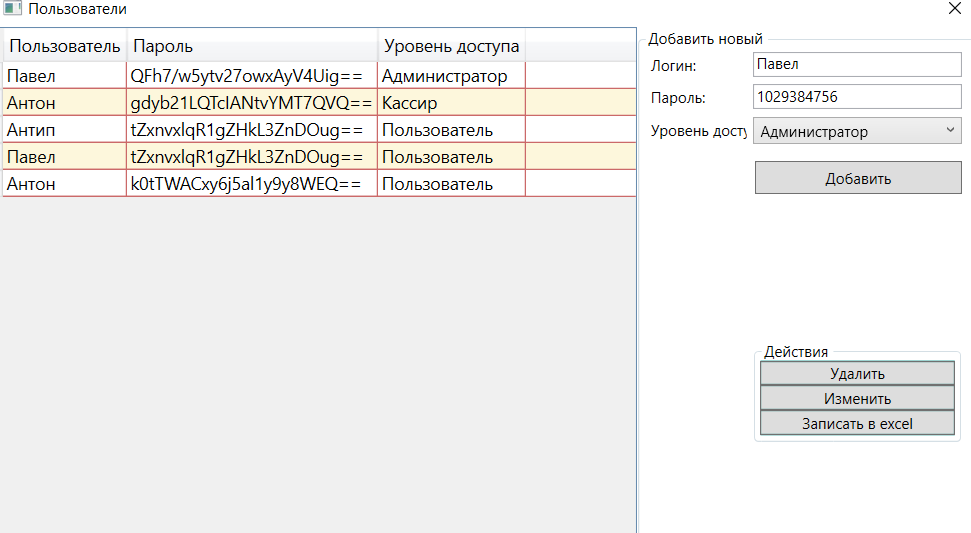


Рисунок 8 – Окно пользователей

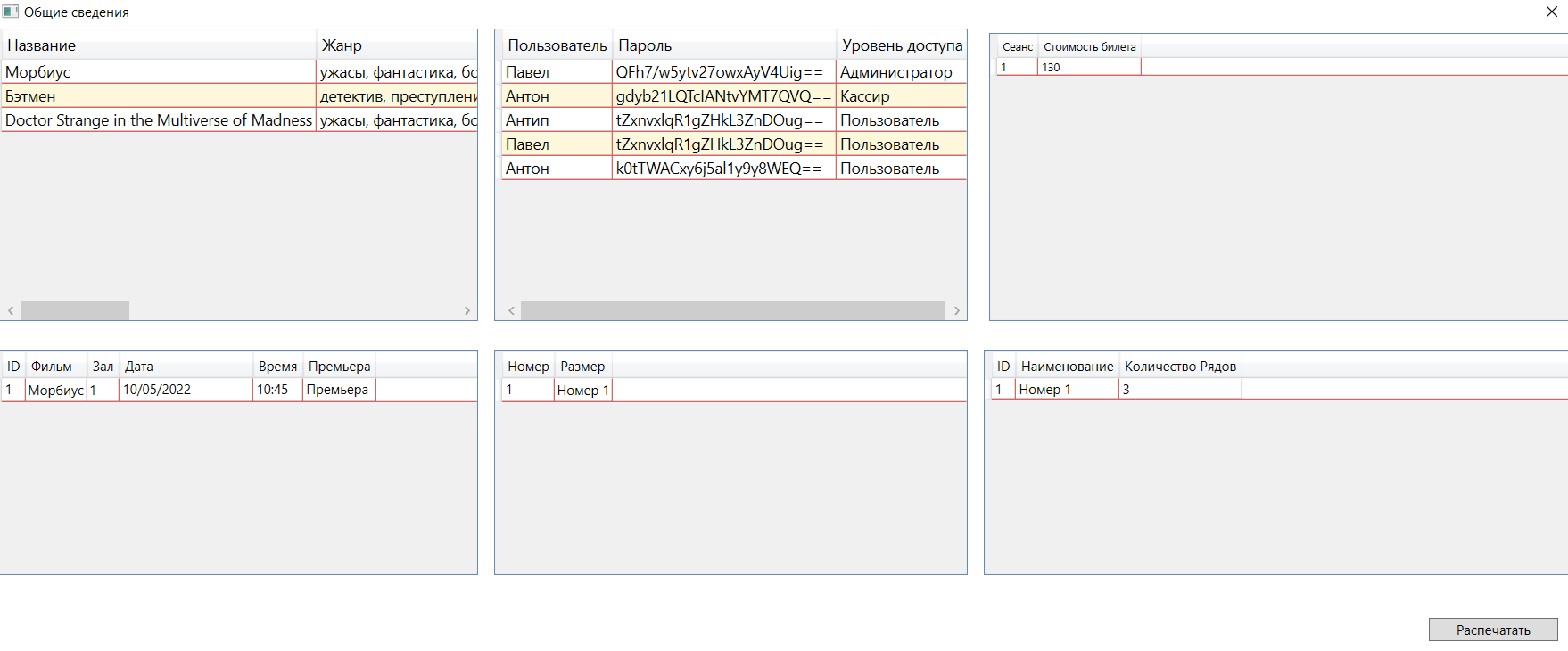


Рисунок 1.9 – Окно общих сведений обо всех данных администратора

Окна кассира:

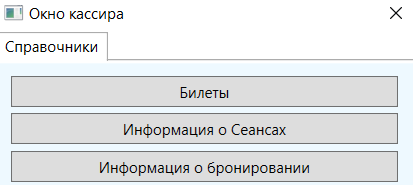


Рисунок 10 – Окно кассира

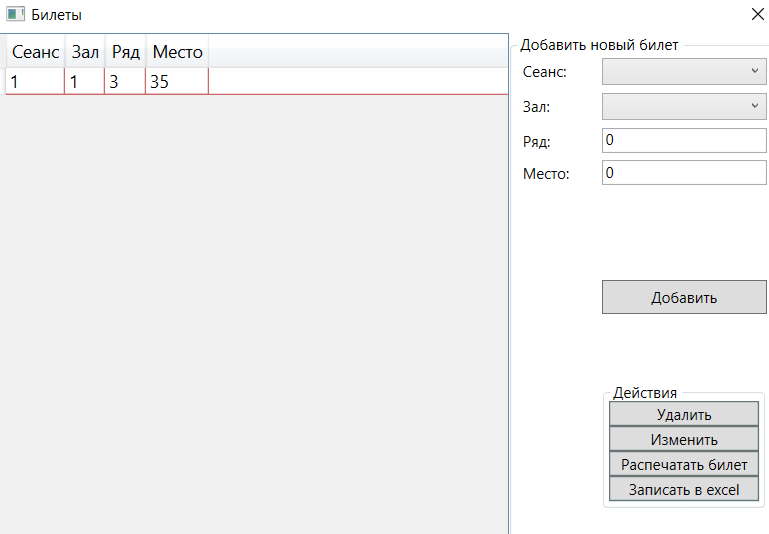


Рисунок 11 – Окно продажи билетов

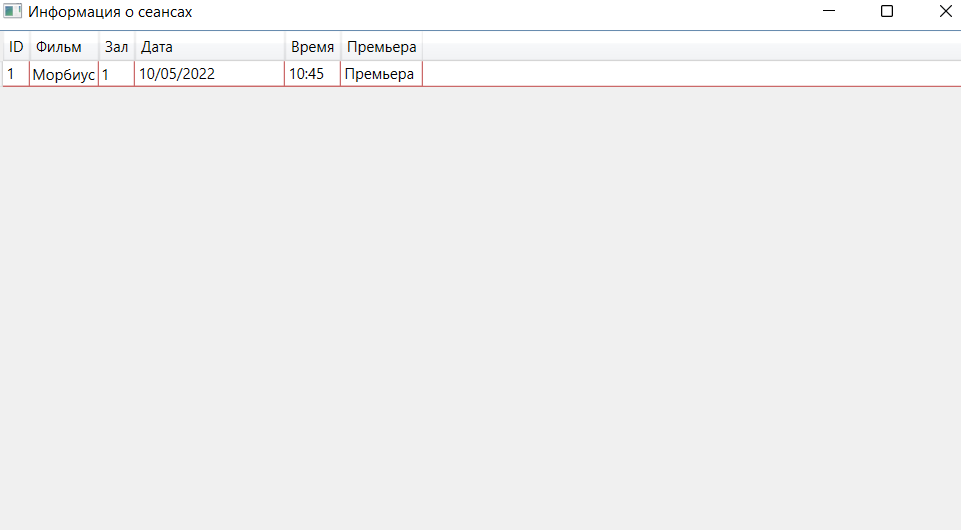


Рисунок 12 – Информация о сеансах

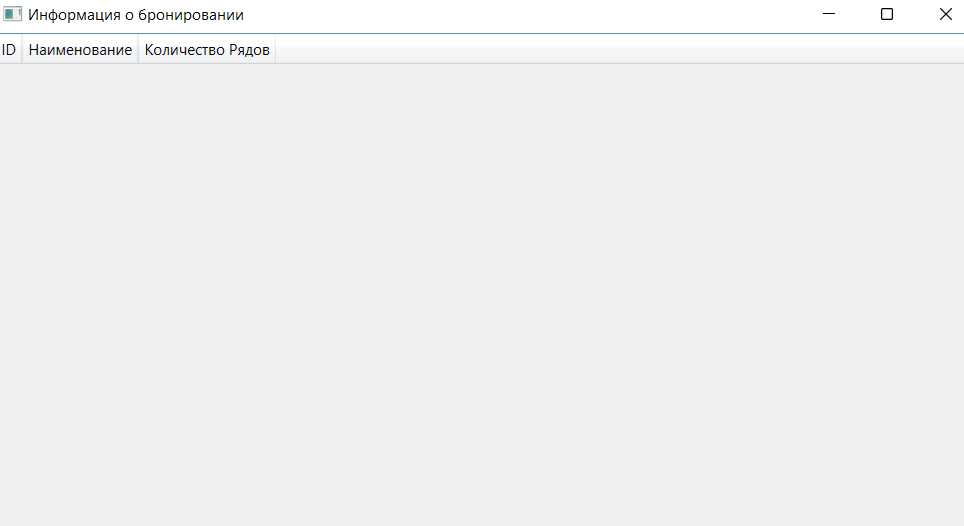


Рисунок 13 – Информация о бронировании билетов

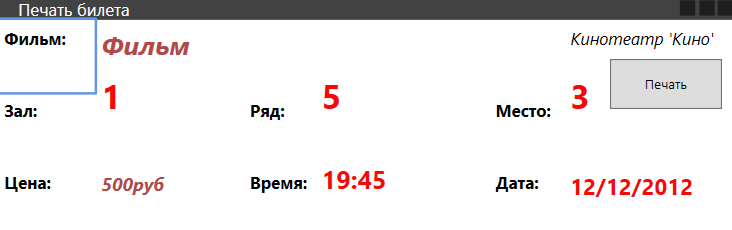


Рисунок 14 – Билет

1. **Разработка**

**3.1 Обоснование выбранных инструментов**

Для реализации курсового проекта выбрана интегрированная среда разработки Microsoft Visual Studio 2022. В целом среда имеет много достоинств:

* обеспечение интуитивно понятной, расширяемой, унифицированной среды для языка, конструкторов и инструментальных средств.
* предоставление разработчикам набора модернизированных взаимодействующих элементов, соответствующих имеющимся у разработчиков навыкам.
* предоставление высокопроизводительных инструментальных средств для всех этапов жизненного цикла разработки - от определения требований и планирования до последующего сопровождения продукта.

В качестве языка программирования был выбран C#.

Преимущества языка программирования C#:

* Подлинная объектная ориентированность (всякая языковая сущность претендует на то, чтобы быть объектом)
* Компонентно-ориентированное программирование
* Безопасный (по сравнению с языками C и C++) код
* Унифицированная система типизации
* Поддержка событийно-ориентированного программирования
* «Родной» язык для создания приложений в среде .NET
* Объединение лучших идей современных языков программирования: Java, C++, Visual Basic и др.

**3.2** **Архитектура программы**

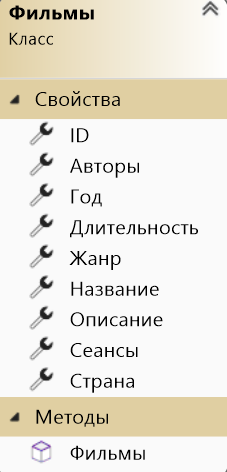
Каждая объектно ориентированная программа состоит из классов. Класс – это шаблон для создания объектов, обеспечивающий начальные значения состоянии: инициализация полей-переменных и реализация поведения функции или методов.



Диаграмма классов

На данной диаграмме находятся классы, из которых состоит программа.

Далее будет приведен рисунок и описание каждого класса.

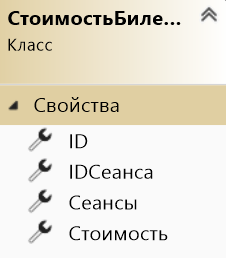
Класс «Фильмы»:

В данном классе расположен объект «фильмы». Он состоит из таких переменных как: ID, Название, Жанр, Длительность, Год, Страна, Авторы, Описание.

* ID – числовая переменная, в которую записывается номер фильма
* Название – строка, в которую записывается название фильма
* Жанр – строка, в которую записывается жанр фильма
* Длительность – строка, в которую записывается длительность фильма
* Год – числовая переменная, в которую записывается год фильма
* Страна – строка, в которую записывается страна-производитель фильма
* Авторы – строка, в которую записывается авторы фильма.
* Описание – строка, в которой находится описание фильма.

Данный класс соединяется с таблицей под названием «фильмы» в базе данных.

Класс «Стоимостьбилетов»:

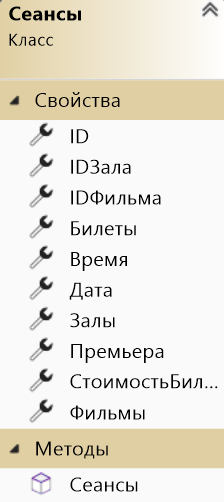


В данном классе находится объект «Стоимостьбилетов». Он состоит из таких переменных как: ID, IDСеанса, Стоимость, Сеансы.

* ID – числовая переменная, в которую записывается номер цены билета.
* IDСеанса – числовая переменная, в которую записывается номер сеанса.
* Стоимость – Числовая переменная, в которую записывается стоимость билета.
* Сеансы – Переменная для связи с сеансами.

Данный класс соединяется с таблице под названием «Стоимостьбилетов» в базе данных.

Класс «Сеансы»:

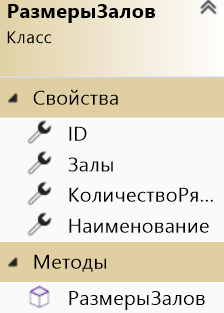


В данном классе находится объект «Сеансы». Он состоит из таких переменных как: ID, IDЗала, IDФильма, Билеты, Время, Дата, Залы, Премьера, СтоимостьБилетов, Фильмы.

* ID – числовая переменная, в которую записывается номер сеанса.
* IDЗала – числовая переменная, в которую записывается номер зала.
* IDФильма – числовая переменная, в которую записывается номер фильма.
* Билеты – переменная, хранящая в себе данные билетов
* Время – строка, в которую записывается время начала сеанса
* Дата – строка, в которую записывается дата сеанса
* Залы – переменная, хранящая в себе данные залов
* Премьера – строка, в которой записывается является ли сеанс премьерой или нет
* СтоимостьБилетов – переменная, хранящее в себе данные о стоимости билетов
* Фильмы – переменная, хранящая в себе данные о фильмах

Данный класс соединяется с таблице под названием «Сеансы» в базе данных.

Класс «РазмерыЗалов»:

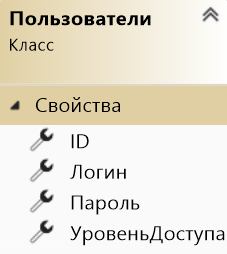


В данном классе находится объект «РазмерыЗалов». Он состоит из таких переменных как: ID, Залы, КоличествоРядов, Наименование.

* ID – числовая переменная, в которую записывается номер размера.
* Залы –переменная, хранящая данные о залах.
* КоличествоРядов – числовая переменная, в которую записывается количество рядов.
* Наименование – строка, в которой записывается название зала

Данный класс соединяется с таблице под названием «РазмерыЗалов» в базе данных.

Класс «Пользователи»:

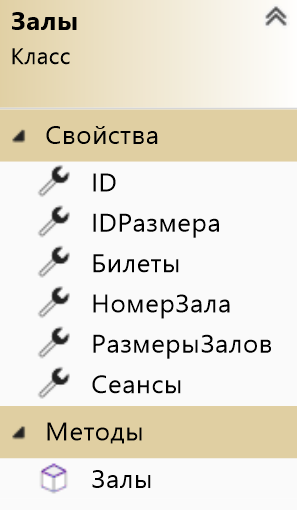


В данном классе находится объект «Пользователи». Он состоит из таких переменных как: ID, Логин, Пароль, УровеньДоступа.

* ID – числовая переменная, в которую записывается номер пользователя
* Логин – строка, в которой записывается логин пользователя
* Пароль – числовая переменная, в которую записывается пароль пользователя в зашифрованном виде
* УровеньДоступа – строка, в которой записывается уровень доступа пользователя

Данный класс соединяется с таблице под названием «Пользователи» в базе данных.

Класс «Залы»:

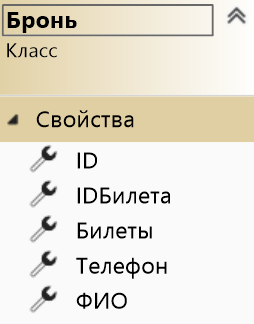


В данном классе находится объект «Залы». Он состоит из таких переменных как: ID, IDРазмера, Билеты, НомерЗала, РазмерыЗалов, Сеансы.

* ID – числовая переменная, в которую записывается номер зала
* IDРазмера – числовая переменная, в которую записывается номер размера
* Билеты – переменная, хранящая данные о билетах
* НомерЗала – числовая переменная, в которую записывается номер зала
* РазмерыЗалов – переменная, хранящая данные о размерах залов
* Сеансы – переменная, хранящая данные о сеансах

Данный класс соединяется с таблицей под названием «Залы» в базе данных.

Класс «Бронь»:

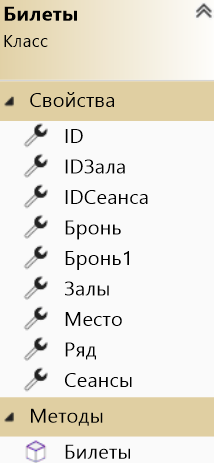


В данном классе находится объект «Бронь». Он состоит из таких переменных как: ID, IDБилета, Билеты, Телефон, ФИО.

* ID – числовая переменная, в которую записывается номер зала
* IDБилета – числовая переменная, в которую записывается номер билета
* Билеты – переменная, хранящая данные о билетах
* Телефон – строка, в которую записывается телефон
* ФИО – строка, в которую записывается фамилия, имя и отчество

Данный класс соединяется с таблицей под названием «Бронь» в базе данных.

Класс «Билеты»:

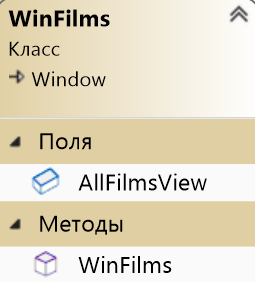


В данном классе находится объект «Билеты». Он состоит из таких переменных как: ID, IDЗала, IDСеанса, Бронь, Бронь1, Залы, Место, Ряд, Сеансы.

* ID – числовая переменная, в которую записывается номер билета
* IDЗала – числовая переменная, в которую записывается номер зала
* IDСеанса – числовая переменная, в которую записывается номер сеанса
* Бронь – переменная, в которую записывается является ли фильм премьерой или нет
* Бронь1 – переменная, хранящая данные о брони
* Залы – переменная, хранящая данные о залах
* Место – числовая переменная, в которую записывается номер места в зале
* Ряд – числовая переменная, в которую записывается ряд в зале
* Сеансы – переменная, хранящая данные о сеансах.

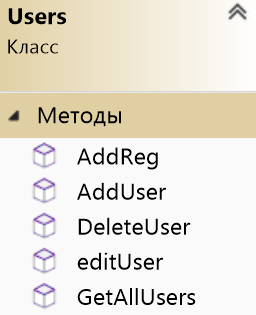
Данный класс соединяется с таблицей под названием «Билеты» в базе данных.

Класс «WinFilms»:



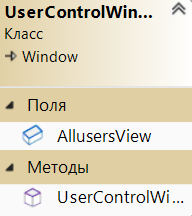
Данный класс является окном фильмов. Он необходим для просмотра, добавления, удаления, редактирования фильмов. В методе WinFilms находится соединение с классом FilmViewModel() и ExcelViewModel(). А переменная AllFilmsView нужно для соединения с таблицей в окне для использования в других классах.

Класс «Users»:



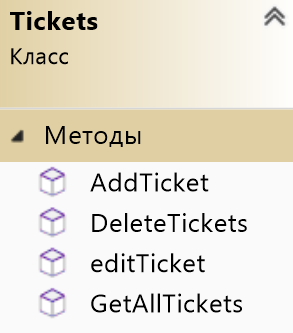
В данном классе находятся методы, необходимые для работы добавления пользователя(AddUser), регистрации пользователя(AddReg), удаления пользователя(DeleteUser), редактирования пользователя(editUser) и список для просмотра пользователей в базе данных(GetAllUsers).

Класс «UserControlWindow»:



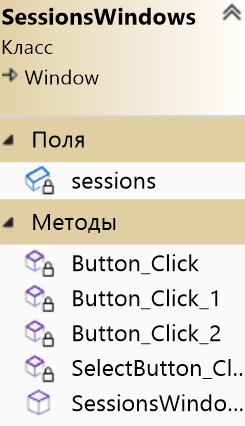
Данный класс является окном пользователей. Он необходим для просмотра, добавления, удаления, редактирования пользователей. В методе UserControlWindow находится соединение с классом LoginViewModel() и ExcelViewModel(). А переменная AllusersView нужно для соединения с контекстом таблицы в окне для использования в других классах.

Класс «Tickets»:



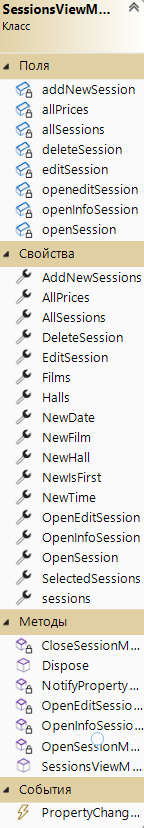
В данном классе находятся методы, необходимые для работы добавления билета(AddTicket), удаления билета(DeleteTickets), редактирования билета(editTicket) и список для просмотра контекста билетов в базе данных(GetAllTickets).

Класс «SessionsWindows»:



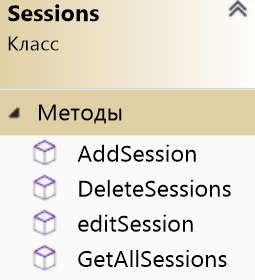
Данный класс является окном сеансов. Он необходим для просмотра, добавления, удаления, редактирования сеансов. В методе SessionsWindows находится соединение с классом SessionViewModel() и ExcelViewModel().

Класс «SessionsViewModel»:



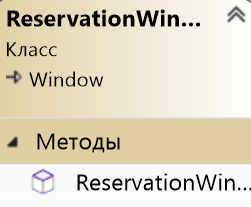
Данный класс является прослойкой между базой данных и окном сеансов. В классе находятся методы взаимодействия базы данных с окном сеансов. Метод Dispose нужен для очистки оперативной памяти от контекста данных сеансов. Метод NotifyPropertyChanged необходим для того, чтобы данные методы и свойства могли работать с окном сеансов. Остальные же методы и свойства нужны для открытия и закрытия окна, а также для добавления, удаления и редактирования сеансов.

Класс «Session»:



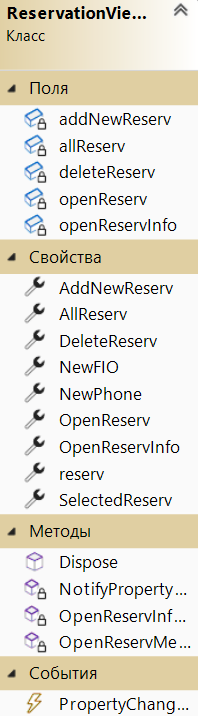
В данном классе находятся методы, необходимые для работы добавления сеансов(AddSession), удаления сеансов(DeleteSessions), редактирования сеансов(editSession) и список для просмотра контекста сеансов в базе данных(GetAllSessions).

Класс «ReservationWindow»:



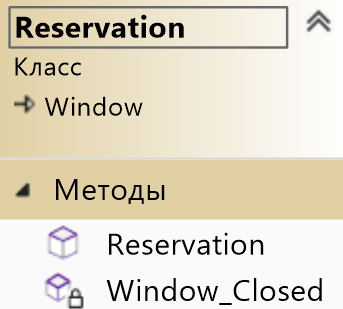
Данный класс является окном просмотра забронированных мест. Нужен для кассира. Соединен с классом ReservationViewModel.

Класс «ReservationViewModel»:



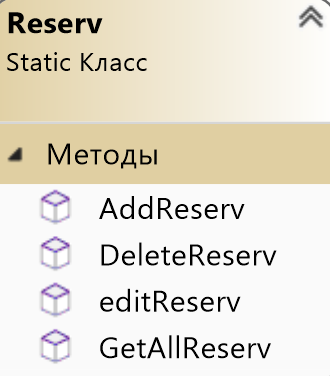
Данный класс является прослойкой между базой данных и окном бронирования. В классе находятся методы взаимодействия базы данных с окном бронирования. Метод Dispose нужен для очистки оперативной памяти от контекста данных бронирования. Метод NotifyPropertyChanged необходим для того, чтобы данные методы и свойства могли работать с окном бронирования. Остальные же методы и свойства нужны для открытия окна, а также для добавления, удаления и редактирования брони.

Класс «Reservation»:



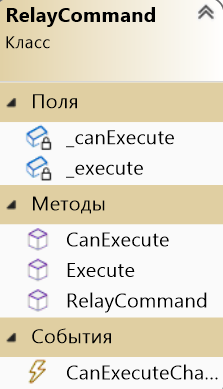
Данный класс является окном бронирования. Он необходим для бронирования билетов. Соединен с классом ReservationViewModel.

Класс «Reserv»:



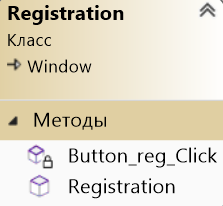
В данном классе находятся методы, необходимые для работы добавления бронирования билетов(AddReserv), удаления бронирования(DeleteReserv), редактирования бронирования(editReserv) и список для просмотра контекста бронирования в базе данных(GetAllReserv).

Класс «RelayCommand»:



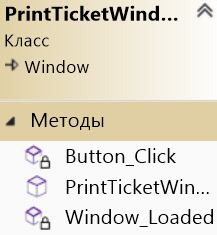
Данный класс необходим для работы интерфейса ICommand.

Класс «Registration»:



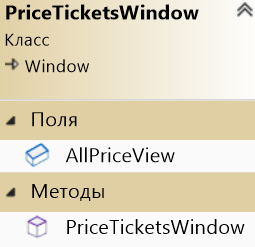
Данный класс является окном регистрации нового пользователя. Соединен с классом LoginViewModel().

Класс «PrintTicketWindow»:



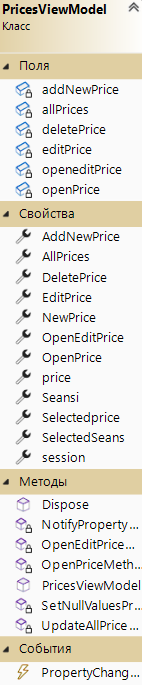
Данный класс является окном распечатки билета. В это окно подгружают все последние данные и показываются в виде билета.

Класс «PricesTicketsWindow»:



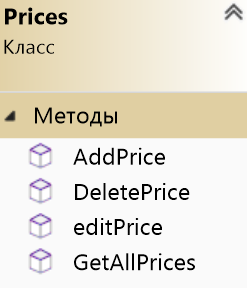
Данный класс является окном цен на билеты. В это окно подгружаются данные о ценах. В окне можно добавлять, редактировать и удалять цены.

Класс «PricesViewModel»:



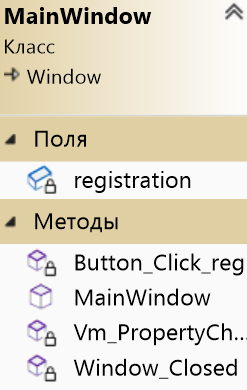
Данный класс является прослойкой между базой данных и окном цен на билеты. В классе находятся методы взаимодействия базы данных с окном цен. Метод Dispose нужен для очистки оперативной памяти от контекста данных цены. Метод NotifyPropertyChanged необходим для того, чтобы данные методы и свойства могли работать с окном цен на билеты. Остальные же методы и свойства нужны для открытия окна, а также для добавления, удаления и редактирования цены.

Класс «Prices»:



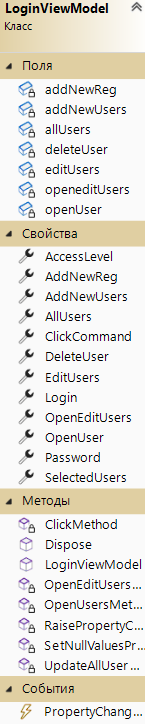
В данном классе находятся методы, необходимые для работы добавления цен на билеты(AddPrice), удаления цен на билеты(DeletePrice), редактирования цен(editPrice) и список для просмотра контекста цен в базе данных(GetAllPrices).

Класс «MainWindow»:



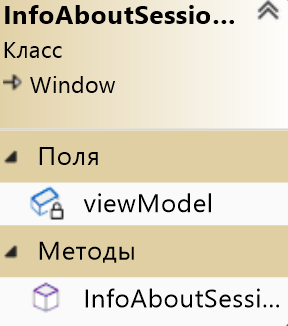
Данный класс является окном авторизации. В это окно подгружаются данные о пользователях.

Класс «LoginViewModel»:



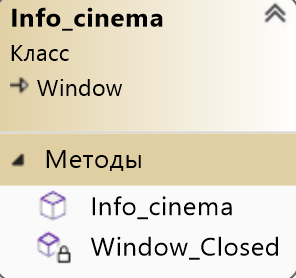
Данный класс является прослойкой между базой данных и окном авторизации. В классе находятся методы взаимодействия базы данных с окном авторизации. Метод Dispose нужен для очистки оперативной памяти от контекста данных пользователей. Метод NotifyPropertyChanged необходим для того, чтобы данные методы и свойства могли работать с окном авторизации. Остальные же методы и свойства нужны для открытия окна, а также для добавления, удаления и редактирования пользователей.

Класс «InfoAboutSession»:



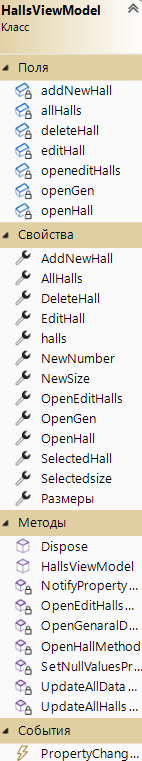
Данный класс является окном информации о сеансах. В это окно подгружаются данные сеансов.

Класс «Info\_cinema»:



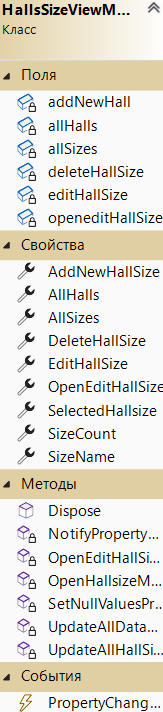
Данный класс является окном администратора для управления кинотеатром.

Класс «HallsViewModel»:



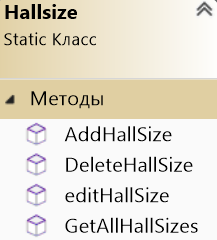
Данный класс является прослойкой между базой данных и окном залов. В классе находятся методы взаимодействия базы данных с окном залов. Метод Dispose нужен для очистки оперативной памяти от контекста данных залов. Метод NotifyPropertyChanged необходим для того, чтобы данные методы и свойства могли работать с окном залов. Остальные же методы и свойства нужны для открытия окна, а также для добавления, удаления и редактирования залов.

Класс «HallsSizeViewModel»:



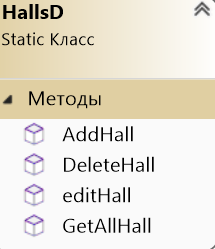
Данный класс является прослойкой между базой данных и окном размера залов. В классе находятся методы взаимодействия базы данных с окном размера залов. Метод Dispose нужен для очистки оперативной памяти от контекста данных размера залов. Метод NotifyPropertyChanged необходим для того, чтобы данные методы и свойства могли работать с окном размера залов. Остальные же методы и свойства нужны для открытия окна, а также для добавления, удаления и редактирования размера залов.

Класс «Hallsize»:

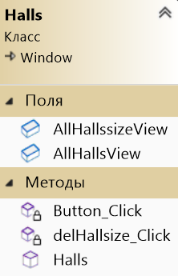


В данном классе находятся методы, необходимые для работы добавления размера залов(AddHallSize), удаления размера залов(DeleteHallSize), редактирования размера залов(editHallSize) и список для просмотра контекста размера залов в базе данных(GetAllHallSize).

Класс «HallsD»:

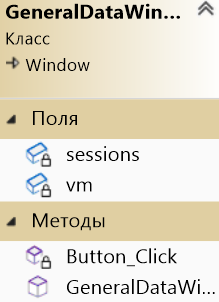


В данном классе находятся методы, необходимые для работы добавления залов(AddHall), удаления залов(DeleteHall), редактирования залов(editHall) и список для просмотра контекста залов в базе данных(GetAllHall).

Класс «Halls»:

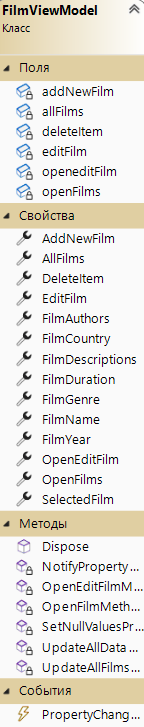
Данный класс является окном залов. Окно нужно для добавления, удаления и редактирования залов.

Класс «GeneralDataWindow»:



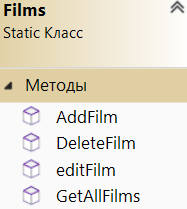
Данный класс является окном всех данных кинотеатра. Окно нужно для быстрого просмотра фильмов, сеансов, залов, пользователей, цен и размера залов.

Класс «FilmViewModel»:



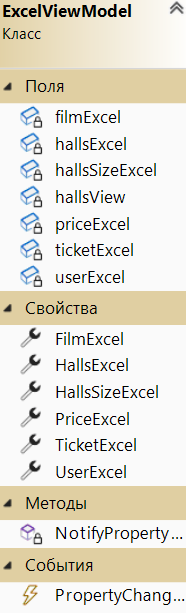
Данный класс является прослойкой между базой данных и окном фильмов. В классе находятся методы взаимодействия базы данных с окном фильмов. Метод Dispose нужен для очистки оперативной памяти от контекста данных фильмов. Метод NotifyPropertyChanged необходим для того, чтобы данные методы и свойства могли работать с окном фильмов. Остальные же методы и свойства нужны для открытия окна, а также для добавления, удаления и редактирования фильмов.

Класс «Films»:



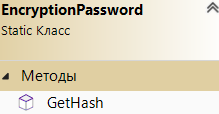
В данном классе находятся методы, необходимые для работы добавления фильмов(AddFilm), удаления фильмов (DeleteFilm), редактирования фильмов (editFilm) и список для просмотра контекста фильмов в базе данных(GetAllFilms).

Класс «ExcelViewModel»:



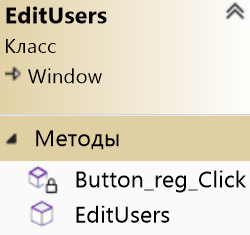
В данном классе прописана работа экспорта данных в excel.

Класс «EncryptionPassword»:



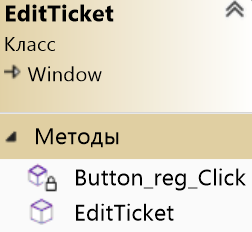
Данный класс нужен для хеширования паролей чтобы защитить их от взлома.

Класс «EditUsers»:



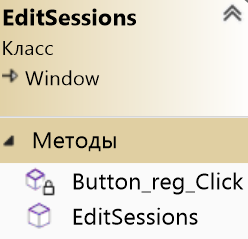
Данный класс является окном редактирования пользователей.

Класс «EditTicket»:



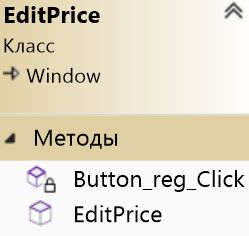
Данный класс является окном редактирования билетов.

Класс «EditSessions»:



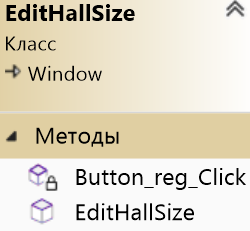
Данный класс является окном редактирования сеансов.

Класс «EditPrice»:



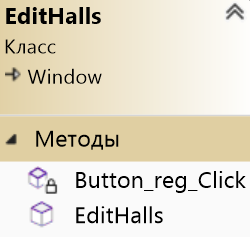
Данный класс является окном редактирования цен.

Класс «EditHallSize»:



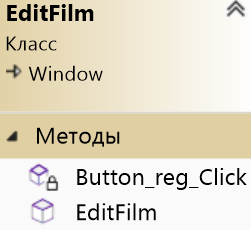
Данный класс является окном редактирования размера залов.

Класс «EditHalls»:



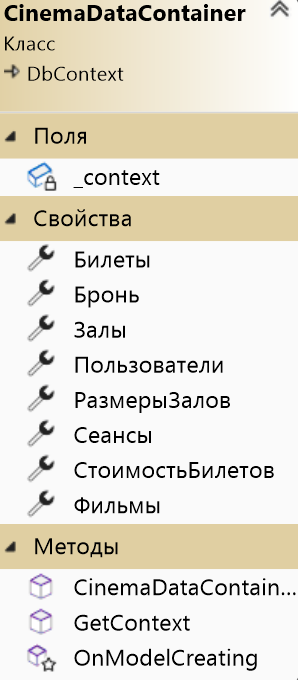
Данный класс является окном редактирования залов.

Класс «EditFilm»:



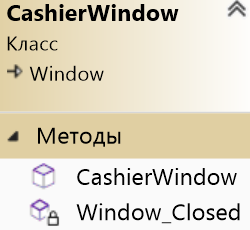
Данный класс является окном редактирования фильмов.

Класс «CinemaDataContainer»:



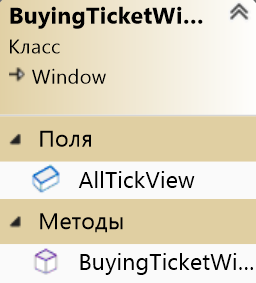
Данный класс является контекстом базы данных для использования в коде.

Класс «CashierWindow»:



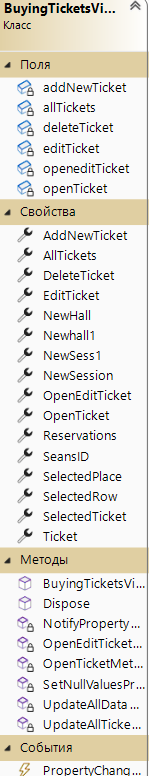
Данный класс является окном кассира для продажи билетов и просмотра данных о бронировании, о сеансах.

Класс «BuyingTicketWindow»:



Данный класс является окном продажи билетов. Он необходим для просмотра, добавления, удаления, редактирования билетов.

Класс «BuyingTicketsViewModel»:



Данный класс является прослойкой между базой данных и окном билетов. В классе находятся методы взаимодействия базы данных с окном билетов. Метод Dispose нужен для очистки оперативной памяти от контекста данных билетов. Метод NotifyPropertyChanged необходим для того, чтобы данные методы и свойства могли работать с окном билетов. Остальные же методы и свойства нужны для открытия окна, а также для добавления, удаления и редактирования билетов.

**3.3** **Руководство по установке и использованию**

Для применения данного программного средства необходимы следующие технические требования:

* процессор Pentium Gold G6400 или выше;
* минимальный объем оперативной памяти 1024 Мб;
* операционная система Windows XP и выше;
* пакет обновлений Framework v4.0;
* рекомендуется монитор типа VGA или с лучшей разрешающей способностью;
* клавиатура;
* мышь.

Компьютер должен работать под управлением операционной системы, начиная с Windows XP и выше. Наиболее удобной операционной системой для проведения испытаний является Windows 10, так как она ориентирована на максимальное использование всех возможностей ПК, сетевых ресурсов и обеспечение комфортных условий работы.

Программа имеет небольшой размер, оптимизирована для быстрого запуска и минимальной загрузки системы. Доступный дружественный интерфейс обеспечивает наглядность простоты в управлении программы и не содержит сложных настроек.

Для начала работы с программным средством необходимо запустить файл «AutoCinema.exe», размер которого составляет 169КБ. Для функционирования программного средства необходимо установить платформу .NET Framework 4.0. Программное средство не нуждается в установке.

**Заключение**

В рамках курсового проектирования было разработано «Программное средство для автоматизации работы кассира и администратора в кинотеатре».

Программное средство имеет ряд достоинств: простой и понятный интерфейс, небольшой объем памяти, занимаемый приложением на различных носителях информации, удобно для быстрой обработки информации.

В программе реализованы такие задачи, как: редактирование данных, экспорт данных, продажа и бронирование билетов на сеансы.

Программа реализована в полном объеме и в соответствии с заданными требованиями. Полностью отлажена и протестирована. Поставленные задачи выполнены.

В программном средстве, благодаря среде разработки Microsoft Visual Studio 2022, был разработан удобный, понятный и простой в использовании интерфейс.

Программное средство готово к практическому использованию. Данное приложение может быть дополнено и модернизировано.

**Список источников**

1 ГОСТ 19.301-2000 ЕСПД. Программа и методика испытаний

2 ГОСТ 19.401-2000 ЕСПД. Текст программы

3 ГОСТ 19.402-2000 ЕСПД. Описание программы

4 ГОСТ2.105-95 ЕСКД. Общие требования к текстовым документам

5 [C# и WPF | Паттерн MVVM (metanit.com)](https://metanit.com/sharp/wpf/22.1.php) – Паттерн MVVM

6 [Developer tools, technical documentation and coding examples | Microsoft Docs](https://docs.microsoft.com/en-us/)

7 Дж. Рихтер. CLR via C#

8 Стивен С. Скиена. Алгоритмы. Руководство по разработке

9 Сергей Тепляков. Паттерны проектирования на платформе .NET

10 LINQ. Карманный справочник. Джозеф Албахари, Бен Албахари

11 Microsoft Visual C#. Подробное руководство. Джон Шарп

12 Леонов В. Простой и понятный самоучитель Word и Excel

13 Адитья Бхаргава. Грокаем алгоритмы

14 Мюррей А. Эффективная работа в Microsoft Excel

15 C# на примерах. П.В. Евдокимов