

МИНИСТЕРСТВО ОБРАЗОВАНИЯ КИРОВСКОЙ ОБЛАСТИ  
Кировское областное государственное профессиональное образовательное  
бюджетное учреждение

"Слободской колледж педагогики и социальных отношений"

## **КУРСОВОЙ ПРОЕКТ**

по ПМ 01 «Разработка программных модулей» на тему:  
**РАЗРАБОТКА ПРОГРАММНОГО МОДУЛЯ ДЛЯ АВТОМАТИЗАЦИИ  
ПРОДАЖИ БИЛЕТОВ В КИНОТЕАТРЕ**

Выполнил: Платунов Павел  
Андреевич

Специальность 09.02.07  
Информационные системы и  
программирование

Группа 21П-1  
Форма обучения: очная

Руководитель: Калинин  
Арсений Олегович

Дата защиты курсового проекта:

Председатель ПЦК:

Оценка за защиту курсового проекта:

Слободской

2024

## ОГЛАВЛЕНИЕ

Введение .....	3
1. Анализ предметной области.....	5
2. Разработка технического задания.....	12
3. Описание алгоритмов и схема функционирования программного модуля ....	16
4. Тестирование программного модуля .....	22
Заключение .....	25
Список литературы .....	26
Приложения .....	28

## ВВЕДЕНИЕ

В настоящее время достаточно популярным местом отдыха являются кинотеатры. Каждый день множество людей посещают различные кинотеатры для просмотра как недавно, так и давно вышедших фильмов, мультфильмов и других кинокартин. Из-за большого количества клиентов могут возникать очереди при продаже билетов на фильмы. Все это зависит от способности кассира быстро и эффективно работать, а также от наличия современных систем автоматизации.

Раньше продажа билетов производилась только в физическом виде, что занимало хоть и мало времени на заполнение одного билета, но при наплыве клиентов могло уходить очень много времени на обслуживание всей толпы. Тоже самое можно сказать и про сбор и анализ данных при работе кинотеатра, когда нужно было записывать количество проданных билетов и их цену на разные фильмы для того, чтобы в конце рабочего дня подвести итоги продаж и решить является ли показываемый фильм коммерчески выгодным. Сейчас же мы имеем возможность автоматизировать данные бизнес-процессы для повышения эффективности работы кассира и администрации кинотеатра.

Среди основных преимуществ такой автоматизации можно отметить:

- Сокращение времени обслуживания клиентов за счет автоматизации регистрации на сеанс по выбранному фильму и автоматизации печати билетов на выбранный клиентом фильм;
- Увеличение количества обрабатываемых клиентов, которое происходит за счет повышения эффективности работы кассира и сокращения времени обслуживания клиента;
- Повышение удобства клиента за счет уменьшения очередей из людей, которые хотят посмотреть фильм;
- Снижение затрат на содержание персонала за счет уменьшения количества кассиров, ведь при введении систем автоматизации эффективность

работы кассиров увеличиться, а время, затрачиваемое на обслуживание одного клиента уменьшиться;

- Сбор и аналитика данных также будут гораздо быстрее, эффективней, точней, а что самое важное удобней, ведь с введением автоматизации теперь можно отслеживать проданные билеты как на фильмы, так и на отдельный сеанс данного фильма для составления вывода о успешности показываемой кинокартины.

Объект исследования – процесс продажи и учета проданных билетов в кинотеатре.

Предмет исследования – автоматизация процесса продажи и учета проданных билетов в кинотеатре.

Цель курсового проекта – разработка программного обеспечения для автоматизации продажи билетов на фильмы в кинотеатре.

Задачи исследования:

- Описать предметную область;
- Разработать технического задание на создание программного продукта;
- Описать архитектуру программы;
- Описать алгоритмы и функционирование программы;
- Провести тестирование и опытную эксплуатацию;
- Разработать руководство оператора.

Методы исследования: автоматизация процесса продажи и учета проданных билетов в кинотеатре.

Информационную систему исследования составили официальные нормативно-правовые источники, данные об использовании современных информационных систем. Структура работы состоит из введения, трех глав, заключения, списка используемой литературы и приложений.

## 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Кинотеатр — общественное здание или его часть с оборудованием для публичной демонстрации кинофильмов. И для его успешного функционирования необходимо выполнять множество различных функций таких как добавление новых фильмов, составление сеансов по фильмам, продажа билетов, подготовка и показ кинолент, и анализ данных для определения успешности фильма. Успешное функционирование системы зависит от четкой организации работы персонала и эффективного взаимодействия между его различными ролями.

Разрабатываемый программный модуль должен обладать возможностями выбора фильма и конкретного сеанса, на который посетитель хочет сходить, также обеспечить возможность выбора места в зале и печать билета после бронирования места.

Для выполнения всех этих немало важных функций необходим обученный персонал:

- Администратор — управляет системой, работает с базой данных, настраивает конфигурацию зала, управляет персоналом кинотеатра;
- Директор — управляет персоналом, отслеживает продажи билетов на фильмы;
- Букер (менеджер по кинопрокату) — добавляет новые фильмы, составляет и корректирует расписание сеансов, отслеживает продажи билетов на фильмы;
- Кассир — работает с расчетно-кассовым оборудованием, продает билеты на фильмы.

Для более четкого понятия рабочих задач персонала была составлена диаграмма вариантов использования (Рисунок 1).

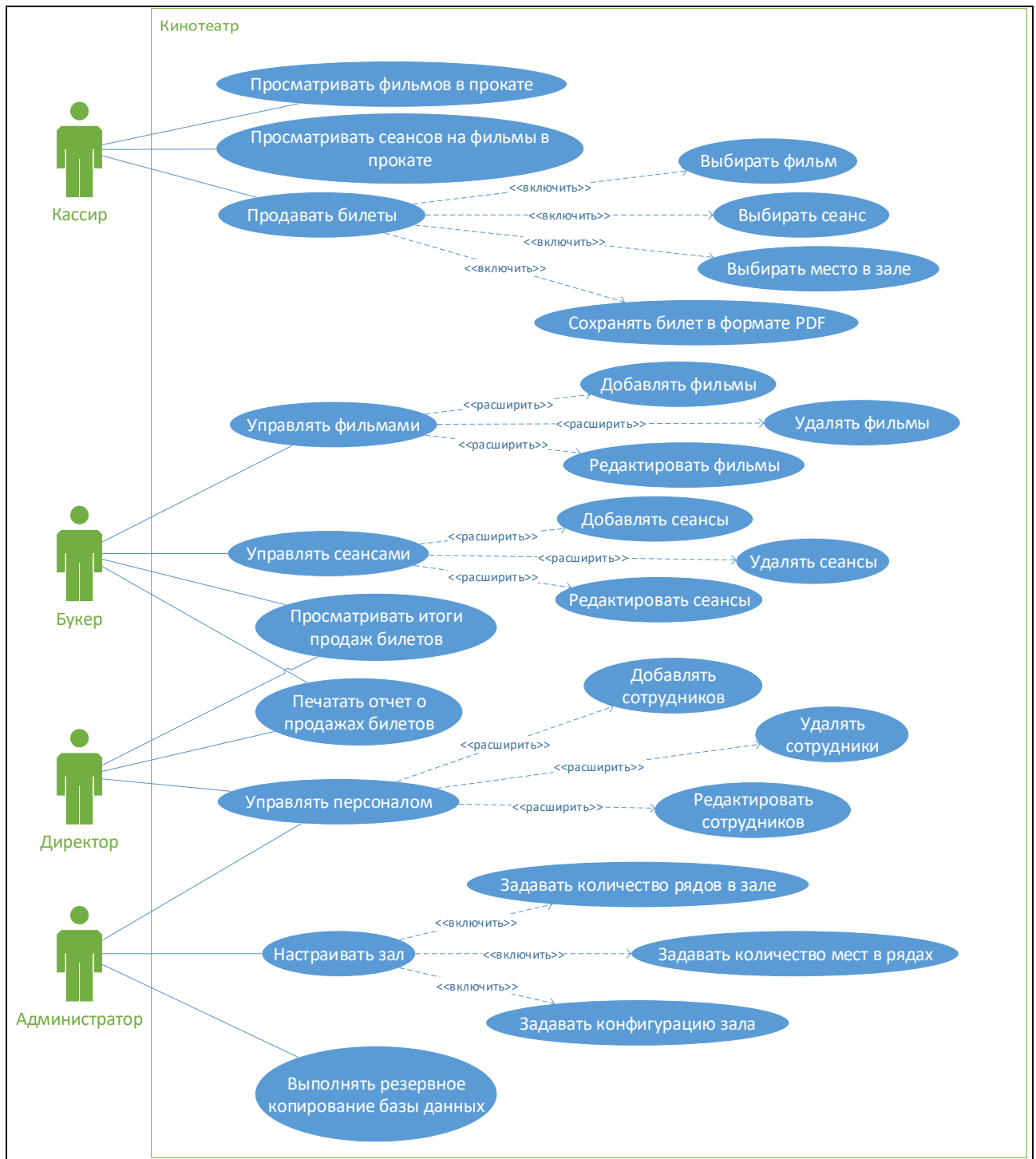


Рисунок 1 - диаграмма вариантов использования

По выделенным объектам и атрибутам была составлена ER-диаграмма показывающая структура создаваемой базы данных (Рисунок 2).

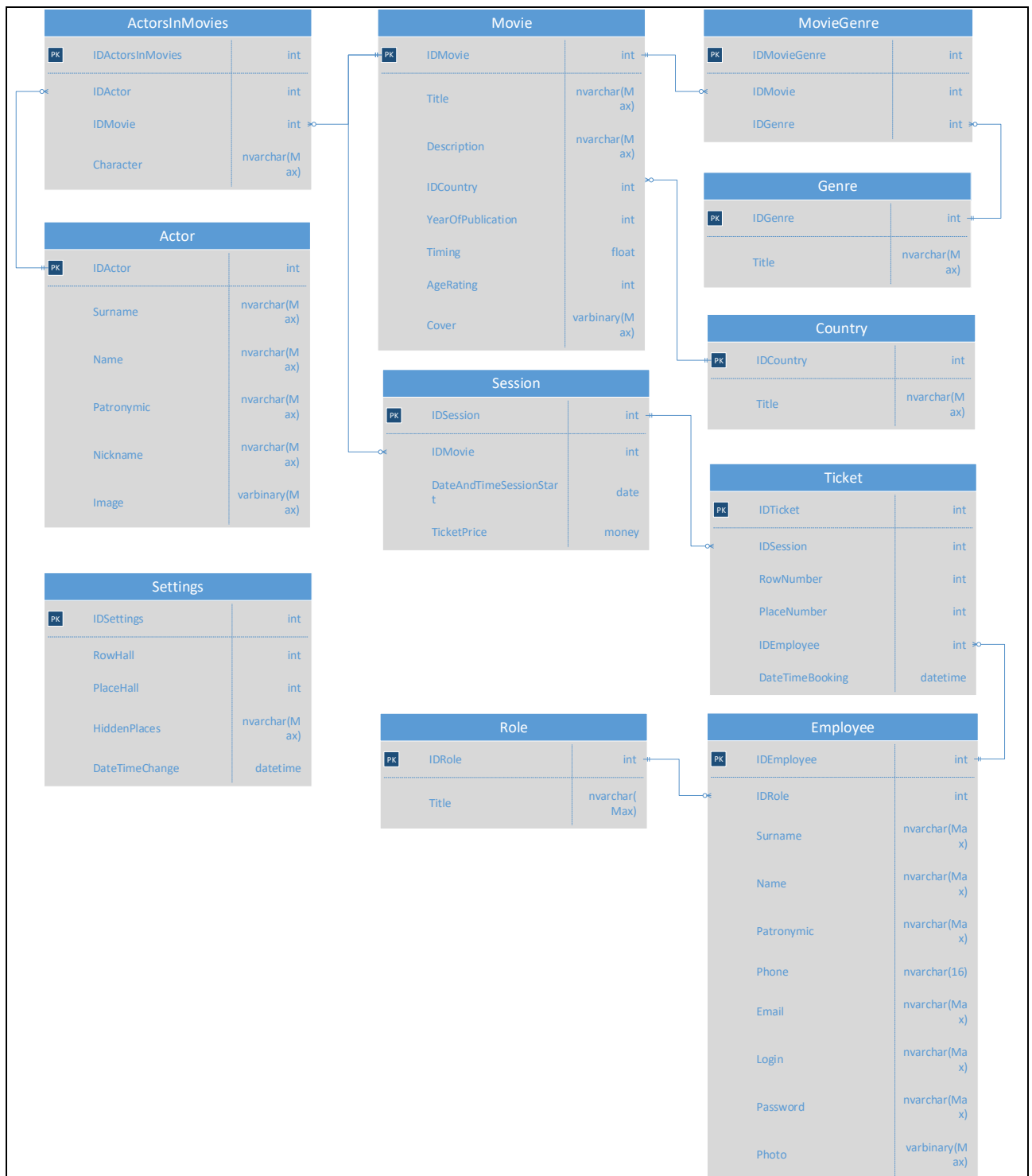


Рисунок 2 – ER-диаграмма

С помощью построенной ER диаграммы, а также выделенным объектам и атрибутам была создана база данных (Рисунок 3).

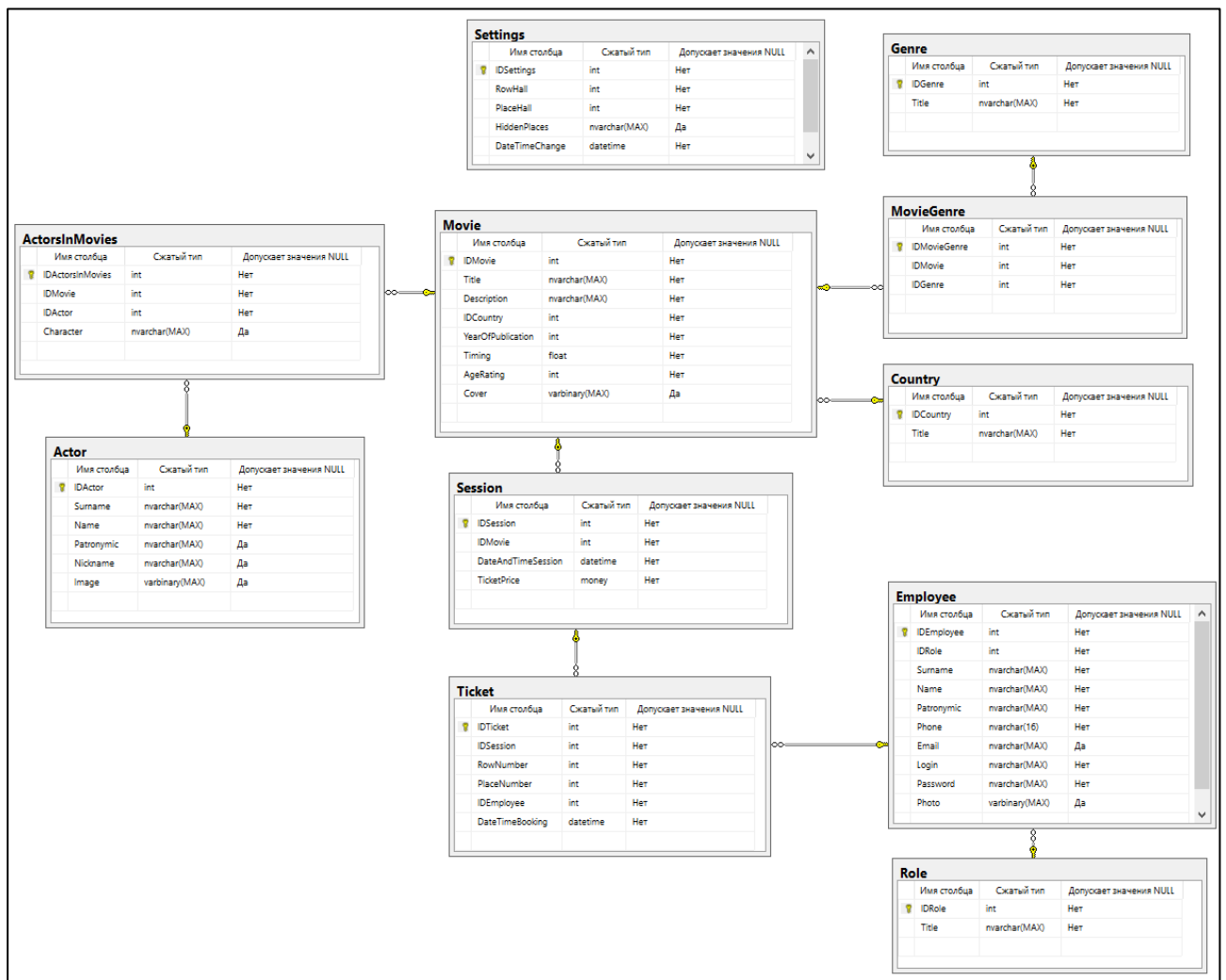


Рисунок 3 - Диаграмма базы данных

Программы аналоги:

InTickets - это платформа для продажи билетов онлайн. Она предоставляет инструменты для создания мероприятий, управления билетами, продажи, обработки платежей и анализа данных. InTickets ориентирована на организаторов событий всех размеров, от небольших концертов до крупных фестивалей (Рисунок 4).



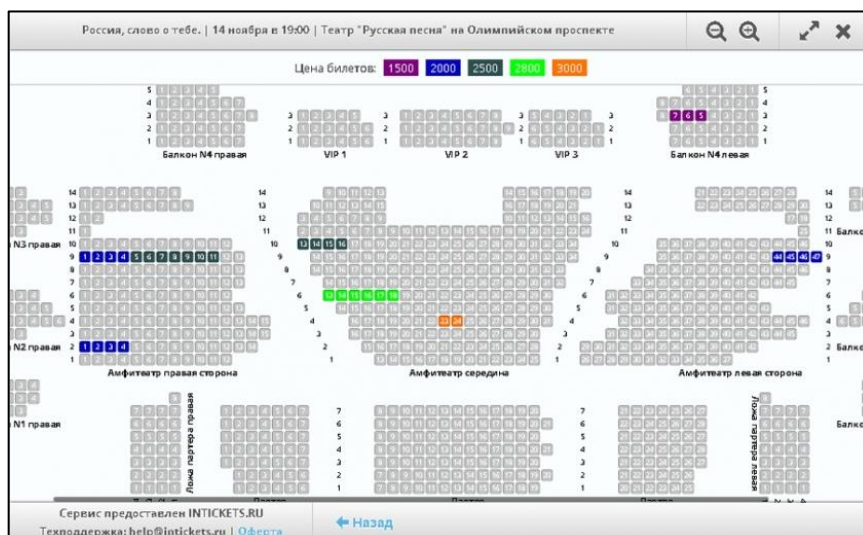


Рисунок 4 – InTickets

#### Преимущества:

- Есть мобильное приложение;
- Гибкая настройка приложения позволяет адаптировать систему под специфику кинотеатра;
- Интеграция с различными системами такими как 1С, Bitrix;
- Стабильная работа;
- Быстрая и качественная техническая поддержка.

#### Недостатки:

- Высокая цена покупки с предварительной договоренностью с поставщиком;
- Сложный в понимание интерфейс;
- Ограниченное количество функций в бесплатной версии.

TicketTool.net - это онлайн-сервис для создания и продажи билетов на события. Он предлагает простой и интуитивный интерфейс для создания мероприятий, установки цен на билеты, настройки дизайна и интеграции с различными платежными системами. TicketTool.net подходит для организаторов небольших мероприятий, таких как концерты, спектакли, семинары и т. д. (Рисунок 5).

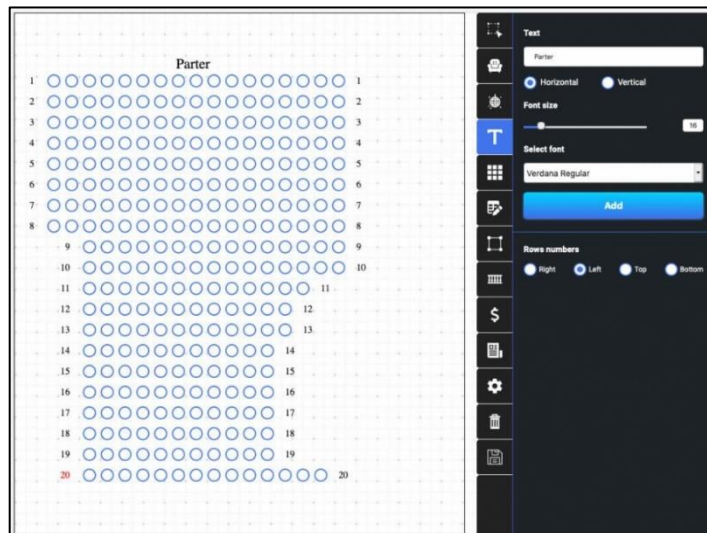


Рисунок 5 - TicketTool.net

#### Преимущества:

- Низкая стоимость программы 2,5% с продаж;
- Простой интерфейс;
- Удобный функционал позволяет легко продавать билеты, управлять расписанием сеансов и отслеживать продажи;
- Мобильная версия;
- Бесплатная пробная демо версия.

#### Недостатки:

- Ограниченные возможности настройки;
- Меньше интеграций чем у InTicket;
- Техническая поддержка своим качеством может варьироваться в зависимости от купленного плана.

Ticket Tech - это компания, которая разрабатывает и предлагает программное обеспечение для управления билетами. Их программное обеспечение позволяет организаторам событий продавать билеты онлайн, на месте и через мобильные приложения. Ticket Tech фокусируется на обеспечении комплексного решения для управления билетами, которое включает в себя функции, такие как управление местами, управление запасами, обработка платежей и отчетность (Рисунок 6).

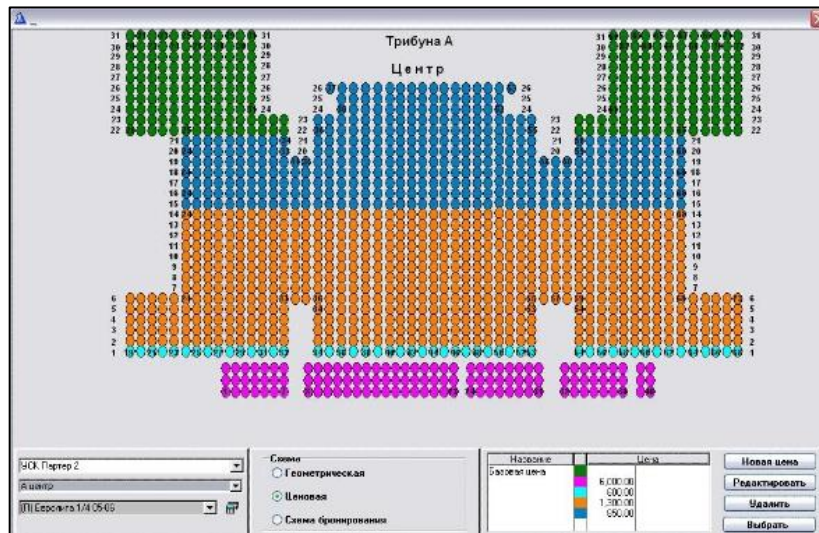


Рисунок 6 - Ticket Tech

#### Преимущества:

- Предлагает передовые решения, например, автоматизированный маркетинг и анализ данных;
- Высокая скорость обработки данных;
- Позволяет легко продвигать кинотеатр в социальных сетях;
- Обеспечивает высокую степень защиты информации;
- Многоязычная поддержка.

#### Недостатки:

- Может быть очень дорогой до 12% с продаж;
- Сложность настройки;
- Ограниченное количество интеграций;
- Отсутствие бесплатной пробной версии.

## 2. РАЗРАБОТКА ТЕХНИЧЕСКОГО ЗАДАНИЯ

При разработке технического задания мы руководствовались требованиями ГОСТ [1; 2].

Наименование программы – «Cinema». Программа предназначена для автоматизации работы кинотеатра.

Разработка программы ведется на основании учебного плана и перечня тем утвержденных на заседании предметно цикловой комиссии информатики и программирования.

Функциональным назначением программы является автоматизация продажи билетов в кинотеатре.

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

- Управление сотрудниками;
- Управление фильмами;
- Управление сеансами;
- Настройка конфигурации зала;
- Расчет количества проданных билетов на фильмы;
- Расчет количества проданных билетов на сеансы выбранного фильма;
- Сохранение отчета по количеству проданных билетов на фильмы в файл формата pdf;
- Сохранение отчета по количеству проданных билетов на сеансы выбранного фильма в файл формата pdf;
- Продажа билетов на выбранный сеанс фильма;
- Открытие pdf файла проданного билета для его печати или сохранения.

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:

- организация бесперебойного питания технических средств;
- использование лицензионного программного обеспечения;
- отсутствие вредоносного программного обеспечения, наличие антивирусной программы;
- соблюдение правил и требований по эксплуатации технических средств.

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать 5 минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

Отказы программы возможны вследствие некорректных действий оператора (пользователя) при взаимодействии с операционной системой. Во избежание возникновения отказов программы по указанной выше причине следует обеспечить работу пользователя без предоставления ему административных привилегий.

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации.

В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ), включающий себя:

- процессор с тактовой частотой, 1 ГГц, не менее;
- оперативную память объемом 512 Мб, не менее;
- жесткий диск со свободным местом 500 Мб, не менее;
- монитор, с разрешением экрана 1366\*768, не менее;
- оптический привод;
- компьютерная мышь;

- клавиатура.

Исходные коды программы должны быть реализованы на языке C#. В качестве интегрированной среды разработки программы должна быть использована среда программирования Microsoft Visual Studio 2022 и Microsoft SQL Server 2014 Management Studio.

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Windows 7/8/10/11.

Программное обеспечение поставляется в виде изделия на CD диске либо USB накопителя.

Упаковка программного изделия должна осуществляться в упаковочную тару предприятия-изготовителя компакт диска или USB накопителя.

Требования к транспортировке и хранению должны соответствовать условиям эксплуатации носителей, на которых находится программный продукт.

Программа должна обеспечивать взаимодействие с пользователем посредством графического пользовательского интерфейса.

Предварительный состав программной документации включает в себя следующие документы:

- техническое задание;
- руководство оператора.

Разработка должна быть проведена в следующие стадии и этапы:

1. Анализ требований:

На стадии анализ требований формулируются цели и задачи проекта. Создается основа для дальнейшего проектирования

2. Проектирование:

На стадии проектирование должны быть выполнены перечисленные ниже этапы работ:

- разработка программной документации;

На этапе разработка программной документации должна быть выполнена разработка технического задания.

При разработке технического задания должны быть выполнены перечисленные работы: постановка задачи, определение и уточнение требований к техническим средствам, определение требований к программе, определение стадий, этапов и сроков разработки программы и документации на нее, выбор языков программирования.

- разработка алгоритма программы;

На этапе разработки алгоритма программы должен быть разработан алгоритм работы программы.

- кодирование;

На стадии кодирования происходит реализация алгоритмов в среде программирования.

- тестирование и отладка.

На стадии тестирования и отладки происходит проверка алгоритмов, реализованных в программе на работоспособность в различных ситуациях. Исправление выявленных ошибок, повторное тестирование.

Приемо-сдаточные испытания должны проводиться при использовании технических средств. Приемка программы заключается в проверке работоспособности программы путем ввода реальных или демонстрационных данных.

Во время приемки работы разработчик предоставляет программу и документацию, которая к ней прилагается. Проводятся испытания программы, при успешных испытаниях программа вводится в эксплуатацию. При ошибках, недопустимых для успешной работы программного продукта – отправляется на доработку.

Было описано техническое задание, содержащее в себе информацию о программном продукте, его функциях, эксплуатации и требования, которые должны учитываться при создании программы и документации к ней.

### 3. ОПИСАНИЕ АЛГОРИТМОВ И СХЕМА ФУНКЦИОНИРОВАНИЯ ПРОГРАММНОГО МОДУЛЯ

Наименование программы – «Сinema». Программа предназначена для автоматизации продажи билетов на сеансы фильмов.

Функциональным назначением программы является автоматизация процесса продажи билетов.

Перед продажей билета на сеанс фильма кассир должен выбрать место в зале, после выбора места в зале информация о нем отобразиться на информационной панели “Бронирование билета” (Рисунок 7).

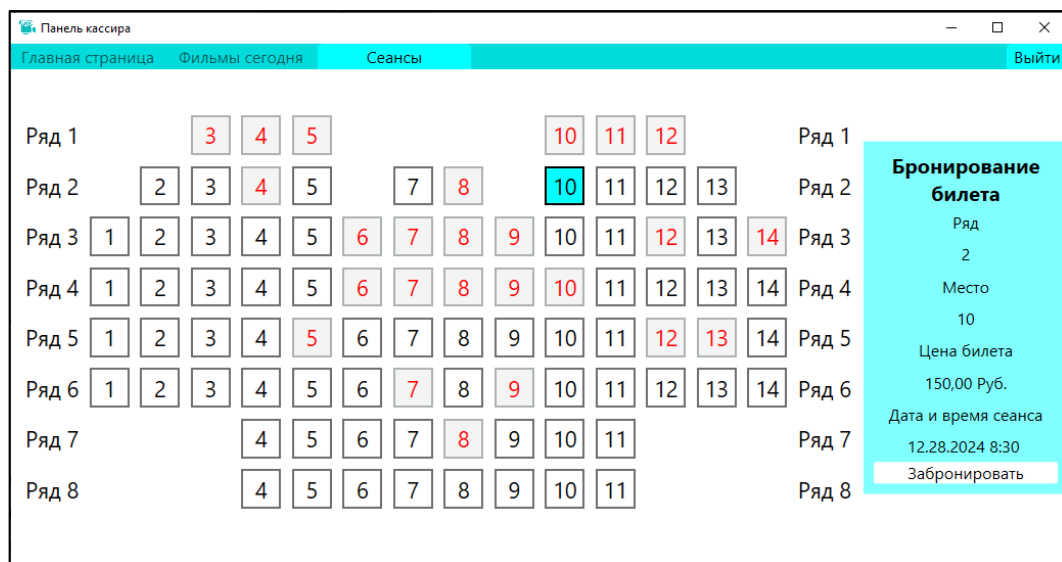


Рисунок 7 – Окно продажи билета на выбранный сеанс

Для отображения выбранного места в зале используется метод «HallButton\_Click» данный метод определяет нажатую кнопку меняет её стилистику и выводит данные о выбранном месте на информационную панель (Рисунок 8, Рисунок 9).

```
private void HallButton_Click(object sender, RoutedEventArgs e)
{
    try
    {
        Button clickedButton = sender as Button;
        if (clickedButton != null)
        {
```

Рисунок 8 – Код метода HallButton\_Click



```

if (selectedRowPlaceButton != null)
{
    selectedRowPlaceButton.Background = Brushes.White;
    selectedRowPlaceButton.Foreground = Brushes.Black;
    selectedRowPlaceButton.BorderBrush = (Brush)(new BrushConverter().ConvertFrom("#FF707070"));
}

Match match = Regex.Match(clickedButton.Name, @"Row(\d+)Place(\d+)");

if (match.Success)
{
    selectedRowNumber = int.Parse(match.Groups[1].Value) + 1;
    selectedPlaceNumber = int.Parse(match.Groups[2].Value);
    clickedButton.Background = Brushes.Aqua;
    clickedButton.Foreground = Brushes.Black;
    clickedButton.BorderBrush = Brushes.Black;
    selectedRowPlaceButton = clickedButton;
}

SelectedPlaceChange();
}
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);
}
}

```

Рисунок 9 – Продолжение метода HallButton\_Click

Поле выбора места в зале кассир нажимает на кнопку “Забронировать” и подтверждает свой выбор (Рисунок 10).

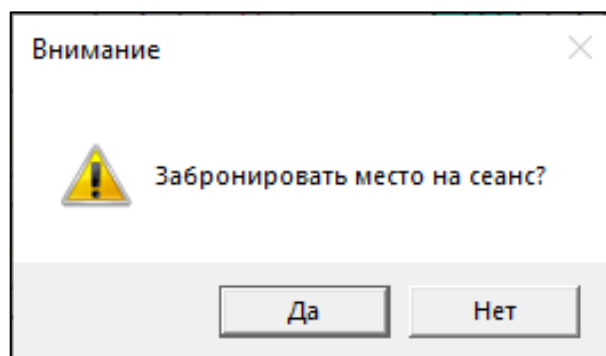


Рисунок 10 – Окно подтверждения бронирования

В результате перед кассиром откроется документ формата PDF с готовым к печати или сохранению билетом (Рисунок 11).

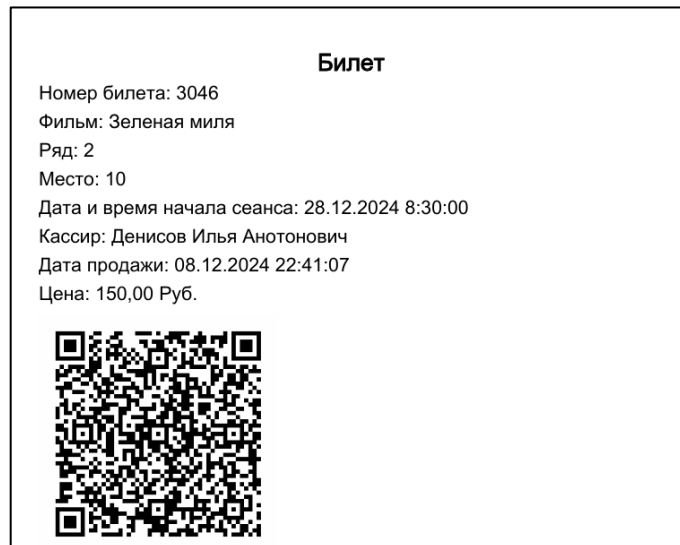


Рисунок 11 – Билет на выбранный сеанс фильма

Алгоритм работы модуля продажи билетов приведен ниже (Рисунок 12).

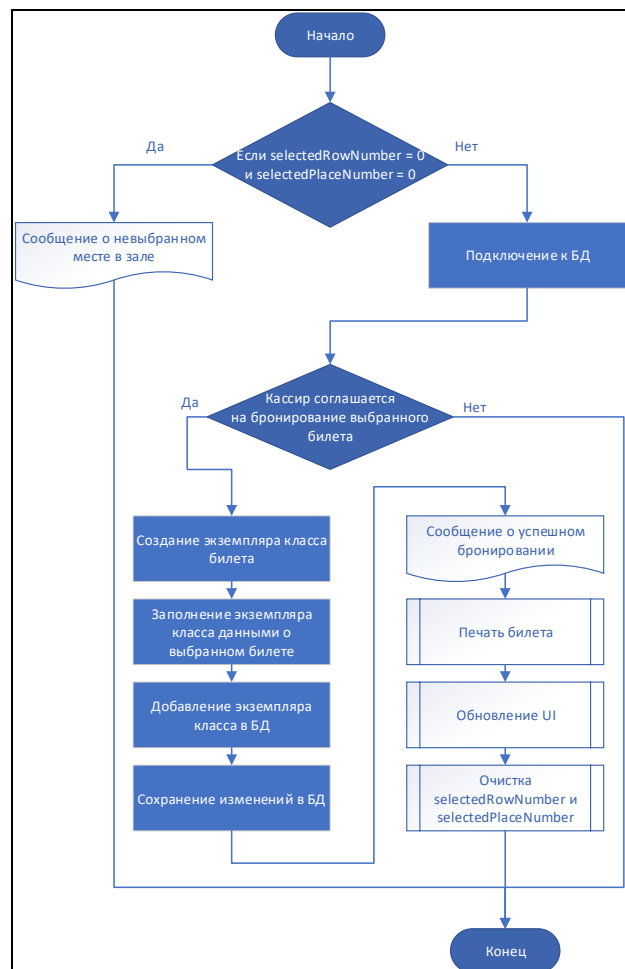


Рисунок 12 - Блок схема функции по продаже билетов на выбранный сеанс фильма

Для сохранения информации о забронированном месте на выбранный сеанс кино используется метод «BookingTicket\_Click», данный метод проверяет выбрал ли кассир место в зале, если нет то выводит соответствующее сообщение, в противном случае выводит сообщение пользователю для подтверждения бронирования билета в случае согласия кассира метод создает экземпляр класса Ticket в который записывает выбранное место и ряд в зале, сотрудника осуществляющего данную операцию, а также дату и время бронирования билета (Рисунок 13, Рисунок 14).

```
private void BookingTicket_Click(object sender, RoutedEventArgs e)
{
    try
    {
        if (selectedRowNumber == 0 && selectedPlaceNumber == 0)
        {
            MessageBox.Show("Место в зале не выбранно", "Внимание", MessageBoxButton.OK,
            MessageBoxImage.Warning);
            return;
        }

        using (var dataBase = new CinemaEntities())
        {
            if (MessageBox.Show("Забронировать место на сеанс?", "Внимание", MessageBoxButton.YesNo,
            MessageBoxImage.Warning) == MessageBoxResult.Yes)
            {
                var newTicket = new Ticket();

                newTicket.IDSession = TransmittedData.idSelectedCashierSession;
                newTicket.RowNumber = selectedRowNumber;
                newTicket.PlaceNumber = selectedPlaceNumber;
                newTicket.IDEmployee = TransmittedData.idEmployee;
                newTicket.DateTimeBooking = DateTime.Now;

                dataBase.Ticket.Add(newTicket);
                dataBase.SaveChanges();

                MessageBox.Show("Место на сеанс зарезервировано", "Готово", MessageBoxButton.OK,
                MessageBoxImage.Information);

                Task.Run(() => PrintTicket(newTicket.IDTicket));

                selectedRowPlaceButton.Background = (Brush)(new BrushConverter().ConvertFrom("#FFDDDDDD"));
                selectedRowPlaceButton.BorderBrush = (Brush)(new BrushConverter().ConvertFrom("#FF707070"));
                selectedRowPlaceButton.Foreground = Brushes.Red;
                selectedRowPlaceButton.IsEnabled = false;

                ClearPlaceChange();
            }
        }
    }
    catch (Exception ex)
```

Рисунок 13 – Код метода BookingTicket\_Click

```

{
    MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);
}
}

```

Рисунок 14 – Продолжение метода BookingTicket\_Click

После сохранения информации в базе данных запускается метод «PrintTicket» (Рисунок 15, Рисунок 16). Данный метод предназначен для вывода билета в PDF формате кассиру. Данный метод использует библиотеку «ITextSharp» для формирования PDF файла и библиотеку «QR Coder» для формирования QR кода.

```

private async Task PrintTicket(int selectedTicket)
{
    try
    {
        using (var dataBase = new CinemaEntities())
        {
            var ticketData = (from ticket in dataBase.Ticket
                             join
                             session in dataBase.Session on ticket.IDSession equals session.IDSession into sessionGroup
                             from session in sessionGroup.DefaultIfEmpty()
                             join
                             movie in dataBase.Movie on session.IDMovie equals movie.IDMovie into movieGroup
                             from movie in movieGroup.DefaultIfEmpty()
                             join
                             employee in dataBase.Employee on ticket.IDEmployee equals employee.IDEmployee into
employeeGroup
                             from employee in employeeGroup.DefaultIfEmpty()
                             where (ticket.IDTicket == selectedTicket)
                             select new
                             {
                                 ticket.IDTicket,
                                 ticket.RowNumber,
                                 ticket.PlaceNumber,
                                 ticket.DateTimeBooking,
                                 movie = movie.Title,
                                 session.DateAndTimeSession,
                                 employee.Surname,
                                 employee.Name,
                                 employee.Patronymic,
                                 session.TicketPrice
                             }).FirstOrDefault();

            string tempFilePath = Path.GetTempFileName() + ".pdf";

            string qrCodeText =
$"{{ticketData.IDTicket}}|{{ticketData.movie}}|{{ticketData.DateAndTimeSession}}|{{ticketData.Surname}}
{{ticketData.Name}} {{ticketData.Patronymic}}|{{Math.Round(ticketData.TicketPrice,2)}}";
            QRCodeGenerator qrGenerator = new QRCodeGenerator();
            QRCodeData qrCodeData = qrGenerator.CreateQrCode(qrCodeText, QRCodeGenerator.ECCLLevel.Q);
            QRCode qrCode = new QRCode(qrCodeData);
            using (System.Drawing.Bitmap qrCodeBitmap = qrCode.GetGraphic(20, System.Drawing.Color.Black,
System.Drawing.Color.White, null))

```

Рисунок 15 – Код метода PrintTicket

```

{
    using (MemoryStream ms = new MemoryStream())
    {
        qrCodeBitmap.Save(ms, System.Drawing.Imaging.ImageFormat.Png); // Сохраняем в PNG
        byte[] qrCodeBytes = ms.ToArray();

        Document doc = new Document();
        PdfWriter writer = PdfWriter.GetInstance(doc, new FileStream($"@\"{tempFilePath}\"", FileMode.Create));

        BaseFont baseFont = BaseFont.CreateFont("C:\\Windows\\Fonts\\arial.ttf", BaseFont.IDENTITY_H,
        BaseFont.NOT_EMBEDDED);
        Font font = new Font(baseFont, 16);
        BaseFont baseFontHead = BaseFont.CreateFont("C:\\Windows\\Fonts\\arial.ttf",
        BaseFont.IDENTITY_H, BaseFont.NOT_EMBEDDED);
        Font fontHead = new Font(baseFont, 20, Font.BOLD);

        doc.Open();
        Paragraph mainParagraph = new Paragraph("Билет", fontHead);
        mainParagraph.Alignment = Element.ALIGN_CENTER;
        Paragraph paragraph = new Paragraph("Номер билета: " + ticketData.IDTicket, font);
        Paragraph paragraph1 = new Paragraph("Фильм: " + ticketData.movie, font);
        Paragraph paragraph2 = new Paragraph("Ряд: " + ticketData.RowNumber, font);
        Paragraph paragraph3 = new Paragraph("Место: " + ticketData.PlaceNumber, font);
        Paragraph paragraph4 = new Paragraph("Дата и время начала сеанса: " +
        ticketData.DateAndTimeSession, font);
        Paragraph paragraph5 = new Paragraph("Кассир: " + ticketData.Surname + " " + ticketData.Name + " "
        + ticketData.Patronymic, font);
        Paragraph paragraph6 = new Paragraph("Дата продажи: " + ticketData.DateTimeBooking, font);
        Paragraph paragraph7 = new Paragraph("Цена: " +
        ticketData.TicketPrice.ToString().Remove(ticketData.TicketPrice.ToString().Length - 2, 2) + " Руб.", font);

        doc.Add(mainParagraph);
        doc.Add(paragraph);
        doc.Add(paragraph1);
        doc.Add(paragraph2);
        doc.Add(paragraph3);
        doc.Add(paragraph4);
        doc.Add(paragraph5);
        doc.Add(paragraph6);
        doc.Add(paragraph7);
        iTextSharp.text.Image image = iTextSharp.text.Image.GetInstance(qrCodeBytes);
        image.ScaleToFit(200, 200);
        doc.Add(image);
        doc.Close();
    }
}
Process.Start(tempFilePath);
}
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message, "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);
}
}

```

Рисунок 16 – Продолжение метода PrintTicket

С полным кодом программы можно ознакомиться в Приложении 1.

#### 4. ТЕСТИРОВАНИЕ ПРОГРАММНОГО МОДУЛЯ

Для проведения тестирования программы мною было произведено базовое тестирование во время разработки программы. При тестировании был выявлен ряд ошибок, которые возникли в ходе выполнения программы.

- Попытка бронирования билета без выбора места в зале  
Без выбора места в зале нажатие на кнопку “Забронировать”

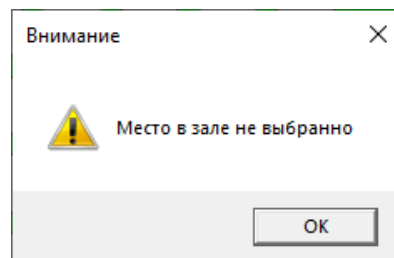


Рисунок 17 - Ошибка бронирования места

Ожидаемый результат: Сообщение о невозможности забронировать билет так как не выбрано место в зале.

Фактический результат: Сообщение о невозможности забронировать билет так как не выбрано место в зале (Рисунок 17).

Решение проблемы: для вывода сообщения о невыбранном месте в зале реализована проверка значений выбранного ряда и места в зале.

- Попытка бронирования билета на занятое место

Попытка выбора места в зале, на который уже забронирован билет

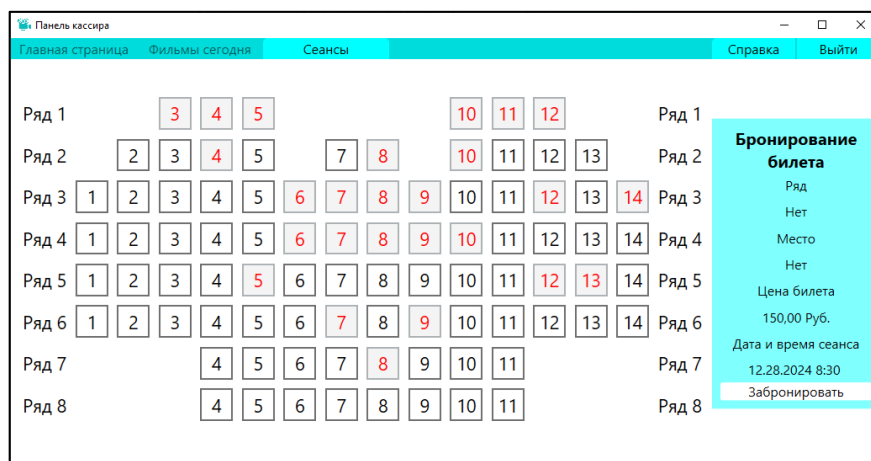


Рисунок 18 - Невозможность бронирования билета на занятое место

Ожидаемый результат: Невозможность выбора ряда и места в зале, на который уже куплен билет.

Фактический результат: Невозможность выбора ряда и места в зале, на который уже куплен билет (Рисунок 18).

Решение проблемы: для решения данной проблемы места, на которые были забронированные билеты выделяются красным цветом, а также отключается взаимодействие с ними.

- Попытка забронировать билет повторно после бронирования

После бронирования билета на свободное место нажать на кнопку бронирования 2-й раз.

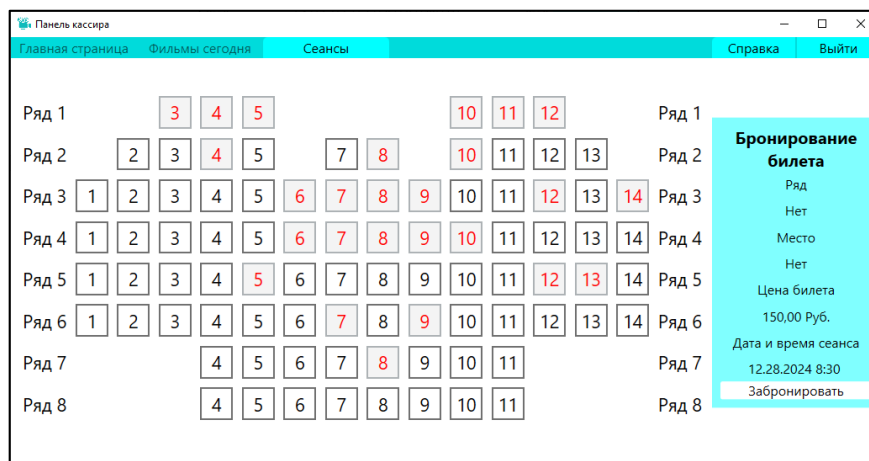


Рисунок 19 – Блокировка повторного бронирования билета

Ожидаемый результат: Значения ряда и места для бронирования на них билета сбрасываются, а в пользовательском интерфейсе место меняет цвет на красный, а также отключается взаимодействие с ним.

Фактический результат: Значения ряда и места для бронирования на них билета сбрасываются, а в пользовательском интерфейсе место меняет цвет на красный, а также отключается взаимодействие с ним (Рисунок 19).

Решение проблемы: Сброс значений ряда и места и блокировка места в интерфейсе пользователя.

- Изменение размера интерфейса в зависимости от размера окна

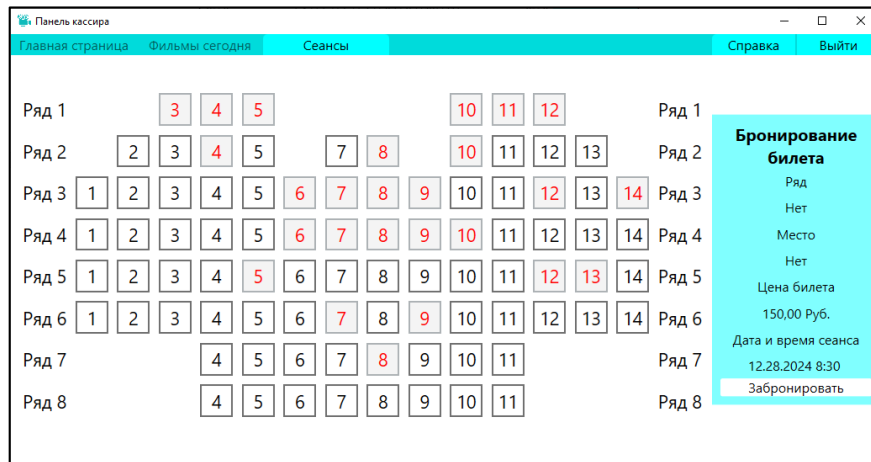


Рисунок 20 - Приложение при разрешении окна 800x1000

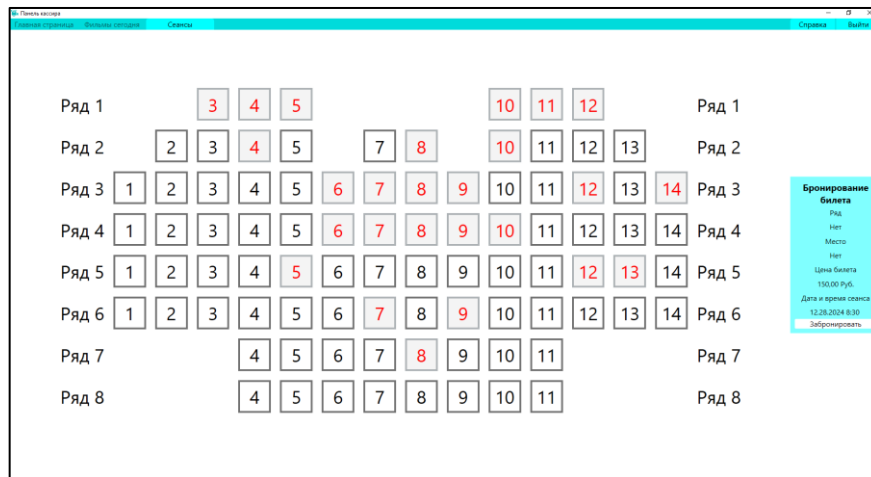


Рисунок 21 - Приложение при разрешении окна 1920x1080

Ожидаемый результат: Интерфейс автоматически изменяет размер по размер окна.

Фактический результат: Интерфейс автоматически изменяет размер по размер окна (Рисунок 20, Рисунок 21).

Решение проблемы: Использование события `SizeChange` у окна для получения его размеров и исходя из них подгонки элементов при помощи `RenderTransform`.



## **ЗАКЛЮЧЕНИЕ**

В результате выполнения курсового проекта был создан программный модуль для автоматизации продажи билетов в кинотеатре. Данный программный модуль позволяет кинотеатрам сократить времени обслуживания клиентов, увеличить пропускную способность кинотеатра, повысить удобство посетителей и снизить затраты на персонал. Также программный модуль позволяет собирать и анализировать данные коммерческой успешности фильма для планирования дальнейшей работы. Для того чтобы помочь пользователю в освоение программы было разработано руководство оператора (Приложение 2). Программный продукт поставляется на USB накопителе (Приложение 3).

Преимущества программного модуля по сравнению с аналогами состоит в простом и интуитивно понятном интерфейсе для любого сотрудника кинотеатра, а также в возможности сбора и анализа данных для определения коммерчески успешного фильма.

В перспективах развития программного модуля можно отметить расширение функциональности таких как интеграция с системами внешних платежей (различные банковские карты, электронные кошельки, мобильные платежи), персонализация интерфейса для каждого пользователя программного модуля, использование данного модуля для реализации онлайн бронирования билета на сайте кинотеатра, встраивание программного модуля в состав десктопного приложения для большего охвата рабочих задач кинотеатра и улучшение производительности.

Таким образом, разработанное приложение представляет собой эффективное решение для оптимизации работы кинотеатра и повышения уровня сервиса для клиентов.

## СПИСОК ЛИТЕРАТУРЫ

1. ГОСТ 19.101-77. Единая система программной документации. Виды программ и программных документов, введ. 01.01.1978. – г. Москва: Изд-во стандартов, 1980. – 4 с.
2. ГОСТ 19.201-78. Единая система программной документации. Техническое задание. Требования к содержанию и оформлению, введ. 01.01.1980. – М.: Изд-во стандартов, 1988. – 3 с.
3. Алекс Дэвис Асинхронное программирование в C# 5.0. / Пер. с англ. Слинкин А. А. – М.: ДМК Пресс, 2013. – 120 с.: ил.
4. Введение в ADO.NET, 2015 [Электронный ресурс] URL: <https://metanit.com/sharp/adonet/1.1.php?ysclid=m2xsepz8p6557122232> (дата обращения 15.10.2024)
5. Джон П. Смит Entity Framework Core в действии / пер. с англ. Д. А. Беликова. – М.: ДМК Пресс, 2022. – 690 с.: ил.
6. Обзор Entity Framework Core — EF Core | Microsoft Learn, 2023 [Электронный ресурс] URL: <https://learn.microsoft.com/ru-ru/ef/core/> (дата обращения: 25.09.2024)
7. Работа с LINQ – C# | Microsoft Learn, 2023 [Электронный ресурс] URL: <https://learn.microsoft.com/ru-ru/dotnet/csharp/tutorials/working-with-linq> (дата обращения: 2.10.2024)
8. Трунин В. Путь программиста T-SQL. Теория и практика – М.: Info-comp.ru, 2020 – 204 с.
9. ADO.NET | Microsoft Learn, 2023 [Электронный ресурс] URL: <https://learn.microsoft.com/ru-ru/dotnet/framework/data/adonet/> (дата обращения 10.10.2024)
10. C# и .NET | LINQ, 2022 [Электронный ресурс] URL: <https://metanit.com/sharp/tutorial/15.1.php?ysclid=m2xsgnx83x118354100> (дата обращения 3.10.2024)

11. C# и .NET | Асинхронные методы, async и await, 2022 [Электронный ресурс] URL: <https://metanit.com/sharp/tutorial/13.3.php?ysclid=m2xsjn72a4147828932> (дата обращения 16.11.2024)
12. Metanit C# | Запись таблиц в PDF, 2012 [Электронный ресурс] URL: <https://metanit.com/sharp/articles/25.php?ysclid=m2xqyr4e12641032739> (дата обращения: 25.09.2024)
13. MS SQL Server в Entity Framework Core и C#, провайдер Microsoft.EntityFrameworkCore.SqlServer, метод UseSqlServer, 2021 [Электронный ресурс] URL: <https://metanit.com/sharp/efcore/7.1.php?ysclid=m2xs59czu1292960590> (дата обращения 10.10.2024)
14. Windows Presentation Foundation - WPF .NET Framework | Microsoft Learn, 2023 [Электронный ресурс] URL: <https://learn.microsoft.com/ru-ru/dotnet/desktop/wpf/?view=netframeworkdesktop-4.8> (дата обращения 27.09.2024)
15. WPF и C# | Полное руководство, 2023 [Электронный ресурс] URL: <https://metanit.com/sharp/wpf> (дата обращения: 26.09.2024)

## **ПРИЛОЖЕНИЯ**

## КОД ПРОГРАММНОГО МОДУЛЯ ДЛЯ БРОНИРОВАНИЯ БИЛЕТОВ

**PlacesCashierControls.xaml**

```

<Page x:Class="Cinema.PlacesCashierControls"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:local="clr-namespace:Cinema"
    mc:Ignorable="d"
    d:DesignHeight="550" d:DesignWidth="1050"
    Title="PlacesCashierControls"
    Background="White" Loaded="Page_Loaded"
    SizeChanged="Page_SizeChanged">

    <Grid Style="{DynamicResource MainColorStyle}">
        <StackPanel x:Name="Hall" Margin="0,0,198,0"
            VerticalAlignment="Center"
            HorizontalAlignment="Center"
            SizeChanged="Hall_SizeChanged">
            <Grid VerticalAlignment="Center"
                HorizontalAlignment="Right" Margin="10,0,0,0"
                Width="200">
                <Rectangle Style="{DynamicResource SecondColorRectanglePlaceStyle}" Opacity="0.5"/>
                <StackPanel>
                    <TextBlock Text="Бронирование билета"
                        Margin="0,10,0,0" TextWrapping="Wrap"
                        Style="{DynamicResource TitleTextBlockStyle}" />
                    <StackPanel>
                        <Label Content="Ряд"
                            HorizontalAlignment="Center"
                            Style="{DynamicResource MainLableStyle}" />
                        <Label x:Name="SelectedRow"
                            Content="Her" HorizontalAlignment="Center"
                            Style="{DynamicResource MainLableStyle}" />
                    </StackPanel>
                    <StackPanel>
                        <Label Content="Место"
                            HorizontalAlignment="Center"
                            Style="{DynamicResource MainLableStyle}" />
                        <Label x:Name="SelectedPlace"
                            Content="Her" HorizontalAlignment="Center"
                            Style="{DynamicResource MainLableStyle}" />
                    </StackPanel>
                </Grid>
            </StackPanel>
        </Grid>
    </Page>

```

```

</StackPanel>
<StackPanel>
    <Label Content="Цена билета"
        HorizontalAlignment="Center"
        Style="{DynamicResource MainLableStyle}" />
    <Label x:Name="TicketPrice" Content=""
        HorizontalAlignment="Center"
        Style="{DynamicResource MainLableStyle}" />
</StackPanel>
<StackPanel>
    <Label Content="Дата и время сеанса"
        HorizontalAlignment="Center"
        Style="{DynamicResource MainLableStyle}" />
    <Label x:Name="DateTimeSession"
        Content="" HorizontalAlignment="Center"
        Style="{DynamicResource MainLableStyle}" />
</StackPanel>
<Button x:Name="BookingTicket"
    Margin="10,0,10,10" Content="Забронировать"
    Style="{DynamicResource TabControlButtonStyle}"
    Click="BookingTicket_Click" />
</StackPanel>
</Grid>
</Grid>
</Page>

```

**PlacesCashierControls.xaml.cs**

```

using iTextSharp.text;
using iTextSharp.text.pdf;
using QRCode;
using System;
using System.Data.Entity.Core.Common.CommandTrees.ExpressionBuilder;
using System.Diagnostics;
using System.IO;
using System.Linq;
using System.Text.RegularExpressions;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Media;
using System.Xml.Linq;
using static Cinema.Authorization;

namespace Cinema
{
    /// <summary>
    /// Логика взаимодействия для
    PlacesCashierControls.xaml
    /// </summary>

```

```

public partial class PlacesCashierControls : Page
{
    public PlacesCashierControls()
    {
        InitializeComponent();
    }

    public int selectedRowNumber;
    public int selectedPlaceNumber;
    public Button selectedRowPlaceButton;

    private void Page_Loaded(object sender,
RoutedEventArgs e)
    {
        LoadData();
    }

    private void LoadData()
    {
        try
        {
            using (var dataBase = new CinemaEntities())
            {
                var ticketData = dataBase.Ticket.Where(w
=> w.IDSession ==
TransmittedData.idSelectedCashierSession).ToList();

                var settingsData =
dataBase.Settings.OrderByDescending(o =>
o.DateTimeChange).FirstOrDefault();

                if (settingsData != null)
                {
                    Hall.Children.Clear();

                    for (int row = 0; row <
settingsData.RowHall; row++)
                    {
                        WrapPanel wrapPanel = new
WrapPanel();

                        TextBlock textBlockBegin = new
TextBlock();
                        textBlockBegin.Text = $"Ряд {row +
1}";
                        textBlockBegin.FontSize = 10;
                        textBlockBegin.Margin = new
Thickness(0, 0, 5, 0);
                        textBlockBegin.VerticalAlignment =
VerticalAlignment.Center;

                        TextBlock textBlockEnd = new
TextBlock();
                        textBlockEnd.Text = $"Ряд {row +
1}";
                        textBlockEnd.FontSize = 10;
                        textBlockEnd.VerticalAlignment =
VerticalAlignment.Center;

```

```
wrapPanel.Children.Add(textBlockBegin);
    for (int place = 1; place <=
settingsData.PlaceHall; place++)
    {
        Button button = new Button();
        button.Content = place;
        button.Name =
$"Row{row}Place{place}";
        button.Width = 18;
        button.Height = 18;
        button.FontSize = 10;
        button.Background =
Brushes.White;
        button.Margin = new Thickness(0,
0, 5, 0);
        button.Click += HallButton_Click;

        if (settingsData.HiddenPlaces !=
null)
        {
            string[] hidePlaceTemp =
settingsData.HiddenPlaces.Split('|');
            foreach (var hidePlaceLine in
hidePlaceTemp)
            {
                Match match =
Regex.Match(hidePlaceLine,
@"Row(\d+)Place(\d+)");

                if (match.Success)
                {
                    if (row + 1 ==
int.Parse(match.Groups[1].Value) + 1 && place ==
int.Parse(match.Groups[2].Value))
                    {
                        button.Click -=
HallButton_Click;

                        button.Opacity = 0;
                        button.IsEnabled = false;
                    }
                }
            }
        }

        foreach (var ticketLine in
ticketData)
        {
            if (row + 1 ==
ticketLine.RowNumber && place ==
ticketLine.PlaceNumber)
            {
                button.Foreground =
Brushes.Red;

                button.IsEnabled = false;
            }
        }

        wrapPanel.Children.Add(button);
    }
}
```

```

        }

wrapPanel.Children.Add(textBlockEnd);

        StackPanel stackPanel = new
StackPanel();
        stackPanel.Margin = new
Thickness(0, 0, 0, 5);
        stackPanel.HorizontalAlignment =
HorizontalAlignment.Center;
        stackPanel.Children.Add(wrapPanel);

        Hall.Children.Add(stackPanel);
    }
    else
    {
        Label label = new Label();
        label.FontSize = 26;
        label.Foreground = Brushes.Red;
        label.Content = "Зал не размечен
администратором.\rОбратитесь с системному
администратору.";
        label.HorizontalAlignment =
HorizontalAlignment.Center;
        label.HorizontalContentAlignment =
HorizontalAlignment.Center;

        Hall.Children.Add(label);
    }

    var sessionData =
dataBase.Session.Where(w => w.IDSession ==
TransmittedData.idSelectedCashierSession).FirstOrDefault();
    TicketPrice.Content =
sessionData.TicketPrice.ToString().Remove(sessionDa
ta.TicketPrice.ToString().Length - 2, 2) + " руб.";
    DateTimeSession.Content =
sessionData.DateAndTimeSession;
    }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Ошибка",
MessageBoxButton.OK, MessageBoxImage.Error);
    }
}

private void Page_SizeChanged(object sender,
SizeChangedEventArgs e)
{
    ResizeWindows();
}

private void Hall_SizeChanged(object sender,
SizeChangedEventArgs e)
{
    ResizeWindows();
}

```

```

    }

    private void ResizeWindows()
    {
        try
        {
            double maxWidth = this.ActualWidth - 10;
            double maxHeight = this.ActualHeight - 10;

            double scale = this.ActualWidth /
(Hall.ActualWidth * 1.28);
            double centerX = Hall.ActualWidth / 2;
            double centerY = Hall.ActualHeight / 2;

            double newWidth = Hall.ActualWidth *
scale;
            double newHeight = Hall.ActualHeight *
scale;

            if (newWidth > maxWidth)
            {
                scale = maxWidth / Hall.ActualWidth;
            }
            if (newHeight > maxHeight)
            {
                scale = Math.Min(scale, maxHeight /
Hall.ActualHeight);
            }

            ScaleTransform scaleTransform = new
ScaleTransform(scale, scale, centerX, centerY);
            Hall.RenderTransform = scaleTransform;
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message, "Ошибка",
MessageBoxButton.OK, MessageBoxImage.Error);
        }
    }

    private void HallButton_Click(object sender,
RoutedEventArgs e)
    {
        try
        {
            Button clickedButton = sender as Button;

            if (clickedButton != null)
            {
                if (selectedRowPlaceButton != null)
                {
                    selectedRowPlaceButton.Background =
Brushes.White;
                    selectedRowPlaceButton.Foreground =
Brushes.Black;
                    selectedRowPlaceButton.BorderBrush =
(Brush)(new
BrushConverter().ConvertFrom("#FF707070"));
                }
            }
        }
    }

```

```

        Match match =
Regex.Match(clickedButton.Name,
@"Row(\d+)Place(\d+)");

        if (match.Success)
        {
            selectedRowNumber =
int.Parse(match.Groups[1].Value) + 1;
            selectedPlaceNumber =
int.Parse(match.Groups[2].Value);

            clickedButton.Background =
Brushes.Aqua;
            clickedButton.Foreground =
Brushes.Black;
            clickedButton.BorderBrush =
Brushes.Black;
            selectedRowPlaceButton =
clickedButton;
        }

        SelectedPlaceChange();
    }
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message, "Ошибка",
MessageBoxButton.OK, MessageBoxImage.Error);
}
}

private void SelectedPlaceChange()
{
    try
    {
        if (selectedRowNumber != 0)
        {
            SelectedRow.Content =
selectedRowNumber;
        }
        else
        {
            SelectedRow.Content = "Her";
        }

        if (selectedPlaceNumber != 0)
        {
            SelectedPlace.Content =
selectedPlaceNumber;
        }
        else
        {
            SelectedPlace.Content = "Her";
        }
    }
    catch (Exception ex)
    {

```

```

        MessageBox.Show(ex.Message, "Ошибка",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

private void ClearPlaceChange()
{
    try
    {
        selectedRowNumber = 0;
        selectedPlaceNumber = 0;
        selectedRowPlaceButton = null;
        SelectedPlaceChange();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Ошибка",
        MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

private async Task PrintTicket(int selectedTicket)
{
    try
    {
        using (var dataBase = new CinemaEntities())
        {
            var ticketData = (from ticket in
dataBase.Ticket
                                join
                                session in dataBase.Session on
ticket.IDSession equals session.IDSession into
sessionGroup
                                from session in
sessionGroup.DefaultIfEmpty()
                                join
                                movie in dataBase.Movie on
session.IDMovie equals movie.IDMovie into
movieGroup
                                from movie in
movieGroup.DefaultIfEmpty()
                                join
                                employee in
dataBase.Employee on ticket.IDEmployee equals
employee.IDEmployee into employeeGroup
                                from employee in
employeeGroup.DefaultIfEmpty()
                                where (ticket.IDTicket ==
selectedTicket)

                                select new
                                {
                                    ticket.IDTicket,
                                    ticket.RowNumber,
                                    ticket.PlaceNumber,
                                    ticket.DateTimeBooking,
                                    movie = movie.Title,
                                    session.DateAndTimeSession,
                                    employee.Surname,
                                }
                            )
        }
    }
}

```



```

        employee.Name,
        employee.Patronymic,
        session.TicketPrice
    }).FirstOrDefault();

    string tempFilePath =
    Path.GetTempFileName() + ".pdf";

    string qrCodeText =
    $"{ticketData.IDTicket}||{ticketData.movie}||{ticketData.
    DateAndTimeSession}||{ticketData.Surname}
    {ticketData.Name}
    {ticketData.Patronymic}||{Math.Round(ticketData.Ticket
    Price,2)}";

    QRCodeGenerator qrGenerator = new
    QRCodeGenerator();
    QRCodeData qrCodeData =
    qrGenerator.CreateQrCode(qrCodeText,
    QRCodeGenerator.ECCLLevel.Q);
    QRCode qrCode = new
    QRCode(qrCodeData);
    using (System.Drawing.Bitmap
    qrCodeBitmap = qrCode.GetGraphic(20,
    System.Drawing.Color.Black,
    System.Drawing.Color.White, null))
    {
        using (MemoryStream ms = new
        MemoryStream())
        {
            qrCodeBitmap.Save(ms,
            System.Drawing.Imaging.ImageFormat.Png); //
            Сохраняем в PNG
            byte[] qrCodeBytes = ms.ToArray();

            Document doc = new Document();
            PdfWriter writer =
            PdfWriter.GetInstance(doc, new
            FileStream($"{tempFilePath}", FileMode.Create));

            BaseFont baseFont =
            BaseFont.CreateFont("C:\\Windows\\Fonts\\arial.ttf",
            BaseFont.IDENTITY_H,
            BaseFont.NOT_EMBEDDED);
            Font font = new Font(baseFont, 16);

            BaseFont baseFontHead =
            BaseFont.CreateFont("C:\\Windows\\Fonts\\arial.ttf",
            BaseFont.IDENTITY_H,
            BaseFont.NOT_EMBEDDED);
            Font fontHead = new Font(baseFont,
            20, Font.BOLD);

            doc.Open();

            Paragraph mainParagraph = new
            Paragraph("Билет", fontHead);
            mainParagraph.Alignment =
            Element.ALIGN_CENTER;

```

```

            Paragraph paragraph = new
            Paragraph("Номер билета: " + ticketData.IDTicket,
            font);

            Paragraph paragraph1 = new
            Paragraph("Фильм: " + ticketData.movie, font);
            Paragraph paragraph2 = new
            Paragraph("Ряд: " + ticketData.RowNumber, font);
            Paragraph paragraph3 = new
            Paragraph("Место: " + ticketData.PlaceNumber, font);
            Paragraph paragraph4 = new
            Paragraph("Дата и время начала сеанса: " +
            ticketData.DateAndTimeSession, font);
            Paragraph paragraph5 = new
            Paragraph("Кассир: " + ticketData.Surname + " " +
            ticketData.Name + " " + ticketData.Patronymic, font);
            Paragraph paragraph6 = new
            Paragraph("Дата продажи: " +
            ticketData.DateTimeBooking, font);
            Paragraph paragraph7 = new
            Paragraph("Цена: " +
            ticketData.TicketPrice.ToString().Remove(ticketData.
            TicketPrice.ToString().Length - 2, 2) + " Руб.", font);

            doc.Add(mainParagraph);
            doc.Add(paragraph);
            doc.Add(paragraph1);
            doc.Add(paragraph2);
            doc.Add(paragraph3);
            doc.Add(paragraph4);
            doc.Add(paragraph5);
            doc.Add(paragraph6);
            doc.Add(paragraph7);

            iTextSharp.text.Image image =
            iTextSharp.text.Image.GetInstance(qrCodeBytes);
            image.ScaleToFit(200, 200);
            doc.Add(image);

            doc.Close();
        }
    }

    Process.Start(tempFilePath);
}
}

catch (Exception ex)
{
    MessageBox.Show(ex.Message, "Ошибка",
    MessageBoxButtons.OK, MessageBoxIcon.Error);
}
}

private void BookingTicket_Click(object sender,
RoutedEventArgs e)
{
    try
    {
        if (selectedRowNumber == 0 &&
        selectedPlaceNumber == 0)

```

```

    {
        MessageBox.Show("Место в зале не
выбранно", "Внимание", MessageBoxButton.OK,
MessageBoxImage.Warning);
        return;
    }

    using (var dataBase = new CinemaEntities())
    {
        if (MessageBox.Show("Забронировать
место на сеанс?", "Внимание",
MessageBoxButton.YesNo,
MessageBoxImage.Warning) ==
MessageBoxResult.Yes)
        {
            var newTicket = new Ticket();

            newTicket.IDSession =
TransmittedData.idSelectedCashierSession;
            newTicket.RowNumber =
selectedRowNumber;
            newTicket.PlaceNumber =
selectedPlaceNumber;
            newTicket.IDEmployee =
TransmittedData.idEmployee;
            newTicket.DateTimeBooking =
DateTime.Now;

            dataBase.Ticket.Add(newTicket);
            dataBase.SaveChanges();

```

```

        MessageBox.Show("Место на сеанс
зарезервировано", "Готово", MessageBoxButton.OK,
MessageBoxImage.Information);

        Task.Run(() =>
PrintTicket(newTicket.IDTicket));

        selectedRowPlaceButton.Background =
(Brush)(new
BrushConverter().ConvertFrom("#FFDDDDDD"));
        selectedRowPlaceButton.BorderBrush =
(Brush)(new
BrushConverter().ConvertFrom("#FF707070"));
        selectedRowPlaceButton.Foreground =
Brushes.Red;
        selectedRowPlaceButton.IsEnabled =
false;

        ClearPlaceChange();
    }
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message, "Ошибка",
MessageBoxButton.OK, MessageBoxImage.Error);
}
}
}

```

Полный исходный код программного продукта поставляется на USB накопителе в Word файле (Приложение 3).

## РУКОВОДСТВО ОПЕРАТОРА

Функциональным назначением программы является автоматизация продажи билетов в кинотеатре.

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

- Управление сотрудниками;
- Управление фильмами;
- Управление сеансами;
- Настройка конфигурации зала;
- Резервное копирование базы данных;
- Продажа билетов на выбранный сеанс фильма;
- Расчет количества проданных билетов на фильмы;
- Расчет количества проданных билетов на сеансы выбранного фильма;
- Сохранение отчета по количеству проданных билетов на фильмы в файл формата pdf;
- Сохранение отчета по количеству проданных билетов на сеансы выбранного фильма в файл формата pdf;
- Открытие pdf файла проданного билета для его печати или сохранения.

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации.

В состав технических средств должен входить IBM-совместимый персональный компьютер (ПЭВМ), включающий себя:

- процессор с тактовой частотой, 1,5 ГГц, не менее;

- оперативную память объемом 1 гб, не менее;
- жесткий диск со свободным местом 500 Мб, не менее;
- монитор, с разрешением экрана 1280\*720, не менее;
- компьютерная мышь;
- клавиатура;

Системные программные средства, используемые программой, должны быть представлены лицензионной локализованной версией операционной системы Windows 7/8/10/11.

Все пользователи должны обладать навыками работы с графическим пользовательским интерфейсом операционной системы.

Для запуска программного продукта запустить исполняемый файл Cinema.exe.

После запуска программа откроется окно авторизации (Рисунок 1). Оно представлено именем базы данных (именем сервера), логином и паролем.

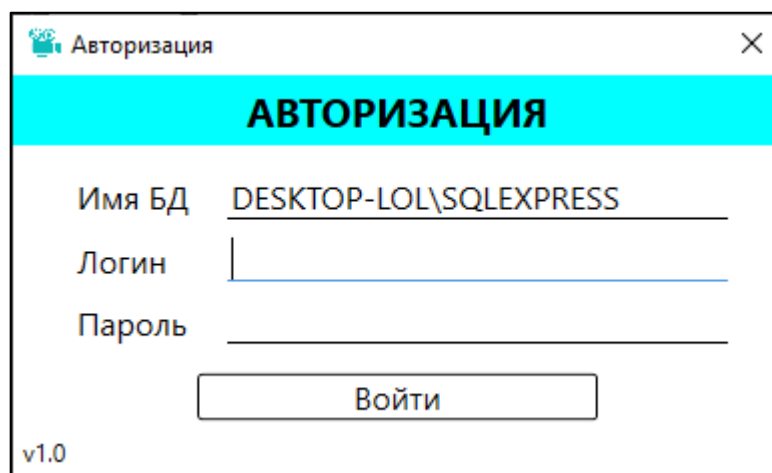


Рисунок 1 – Окно авторизации

Пользователь может авторизоваться по четырем ролям: администратор, кассир, букер(менеджер), директор.

#### **Администратор:**

После авторизации под ролью администратора мы видим панель администратора (Рисунок 2). Она открывается на окне “Сотрудники” здесь мы

можем найти и просмотреть информацию о сотрудниках, а также добавить удалить или редактировать сотрудников.

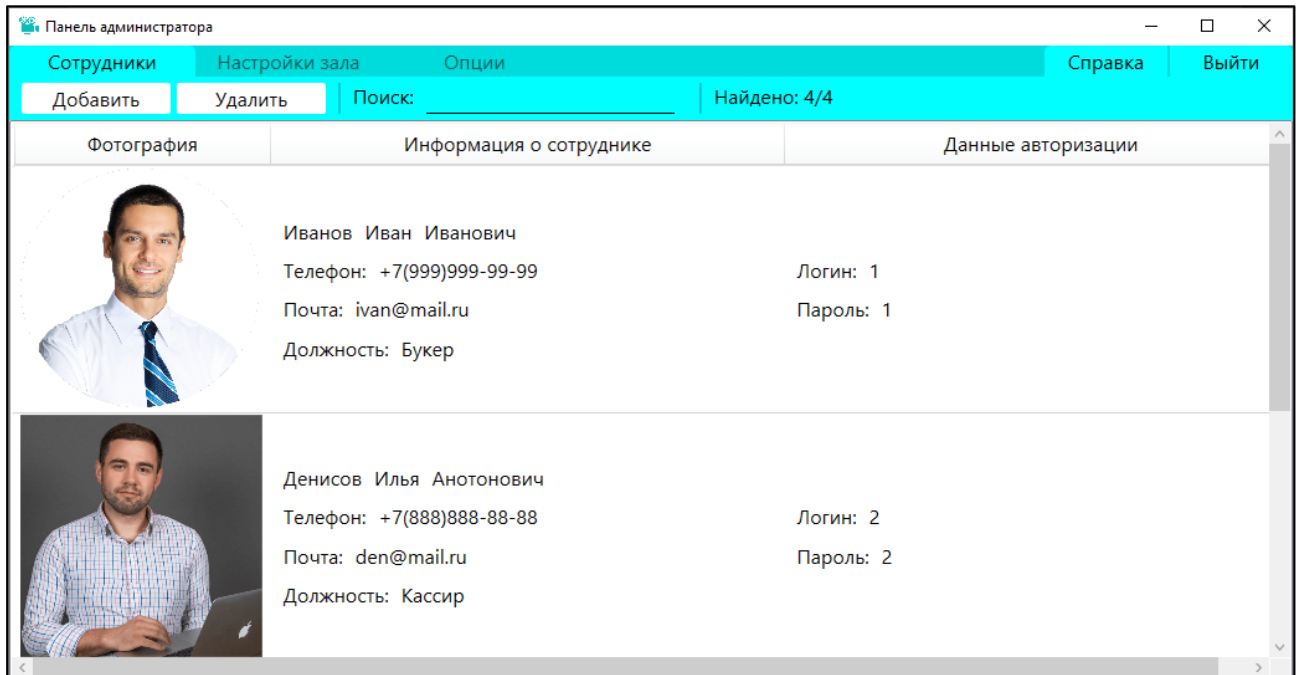


Рисунок 2 – Панель администратора с открытым окном сотрудники

Для того чтобы добавить сотрудника необходимо нажать на кнопку “Добавить” после чего откроется окно добавления сотрудник (Рисунок 3).

The screenshot shows a window titled "Сотрудник" (Employee) with a close button (X) in the top right corner. The window has a blue header with the word "СОТРУДНИК" in white. Below the header, there are several input fields for employee information: "Фамилия:" (Surname), "Имя:" (Name), "Отчество:" (Patronymic), "Телефон:" (Phone), "Эл. Почта:" (Email), "Должность:" (Position) with a dropdown menu showing "Букер", "Логин:" (Login), "Пароль:" (Password), and "Фотография:" (Photo). There is a "Загрузить" (Upload) button next to the photo field. At the bottom of the window, there is a "Сохранить" (Save) button.

Рисунок 3 – Окно добавления сотрудника

После добавления данных о сотруднике приложение отобразит сообщение о успешном сохранение (Рисунок 4).

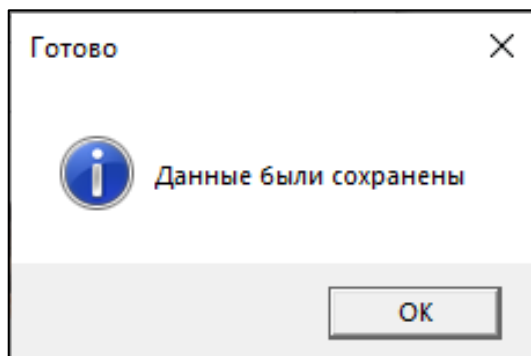


Рисунок 4 – Сообщение о успешном сохранение данных

Для редактирования данных сотрудника необходимо дважды нажать на редактируемого сотрудника, после чего откроется окно редактирования сотрудника (Рисунок 3).

Для удаления сотрудника необходимо выбрать сотрудника и нажать на кнопку удалить после чего отойться сообщение-подтверждение удаления (Рисунок 5).

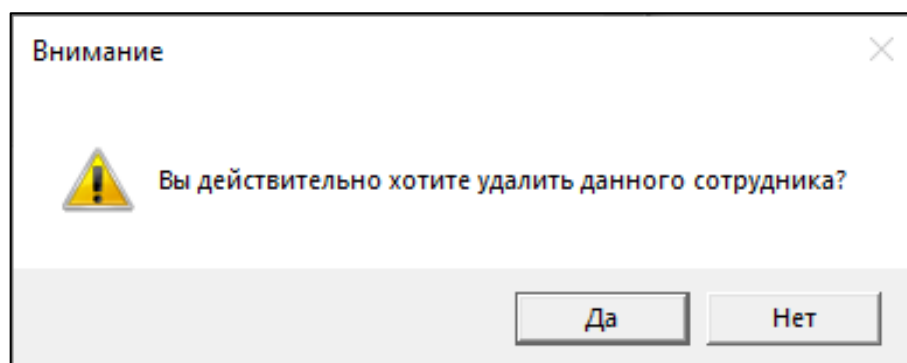


Рисунок 5 – Сообщение подтверждение удаления сотрудника

Также панель администратора обладает настройкой конфигурацией зала (Рисунок 6). В данном окне пользователь может настроить размерность зала выбрав количество рядов и мест, а также убрать лишние места в зале. После настройки зала следует сохранить конфигурацию при помощи кнопки “Сохранить”.

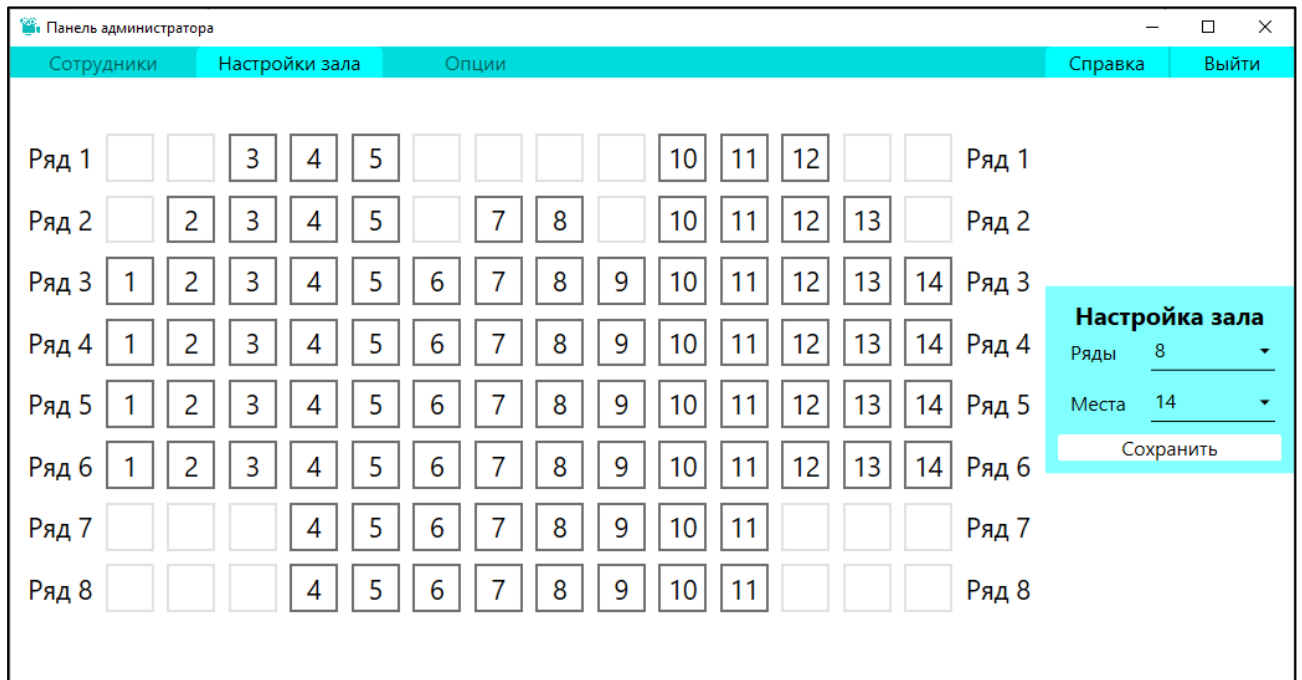


Рисунок 6 – Панель администратора с открытым окном настройки зала

После сохранения конфигурации откроется сообщение о успешном сохранение (Рисунок 7).

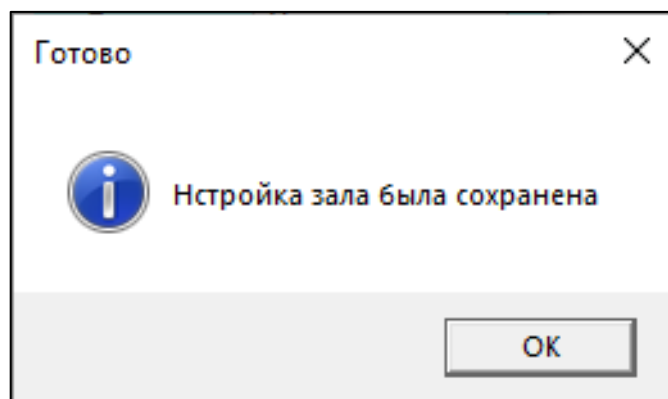


Рисунок 7 – Сообщение о успешном сохранение конфигурации

Панель администратора обладает возможностью резервного копирования, которая находится во вкладке “Опции” (Рисунок 8). В данном окне есть 2 панели для создания полной резервной копии и восстановление из файла резервной копии.

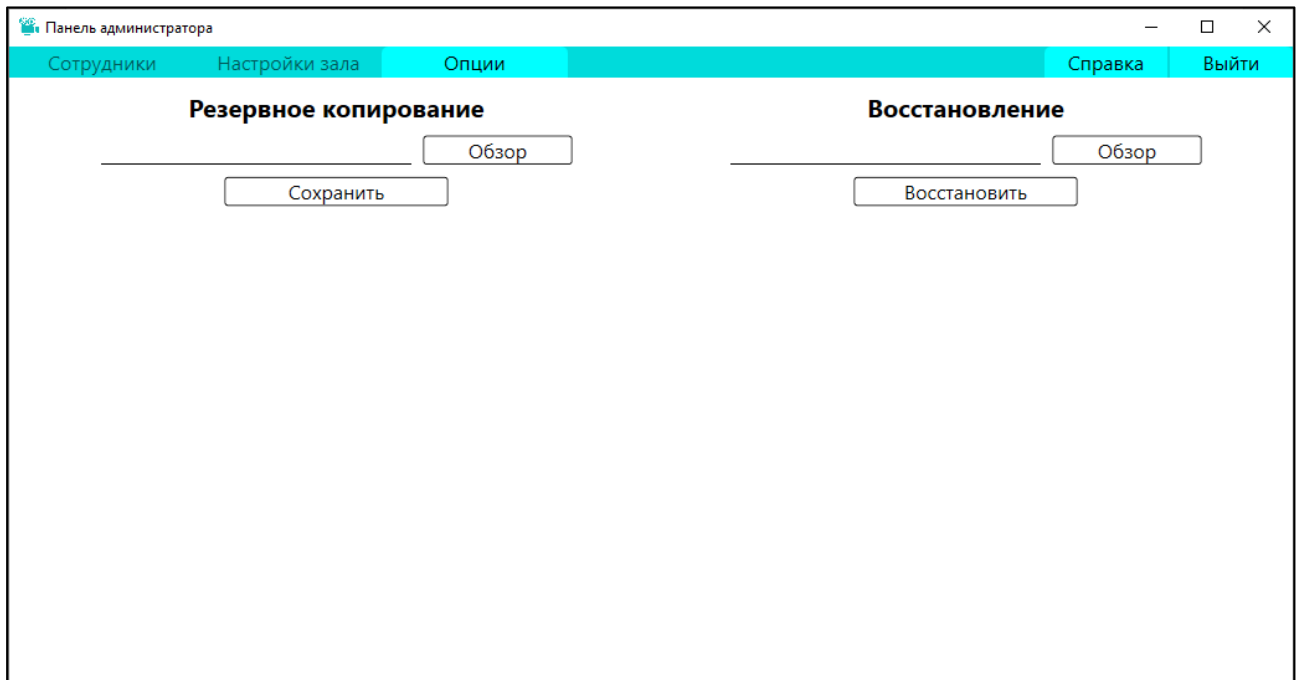


Рисунок 8 – Панель администратора с открытым окном опции

### Букер (менеджер):

После авторизации под ролью букера (менеджера) мы видим панель менеджера (Рисунок 9). Она открывается на окне “Фильмы” здесь мы можем найти и просмотреть информацию о фильмах, а также добавить удалить или редактировать фильмы.

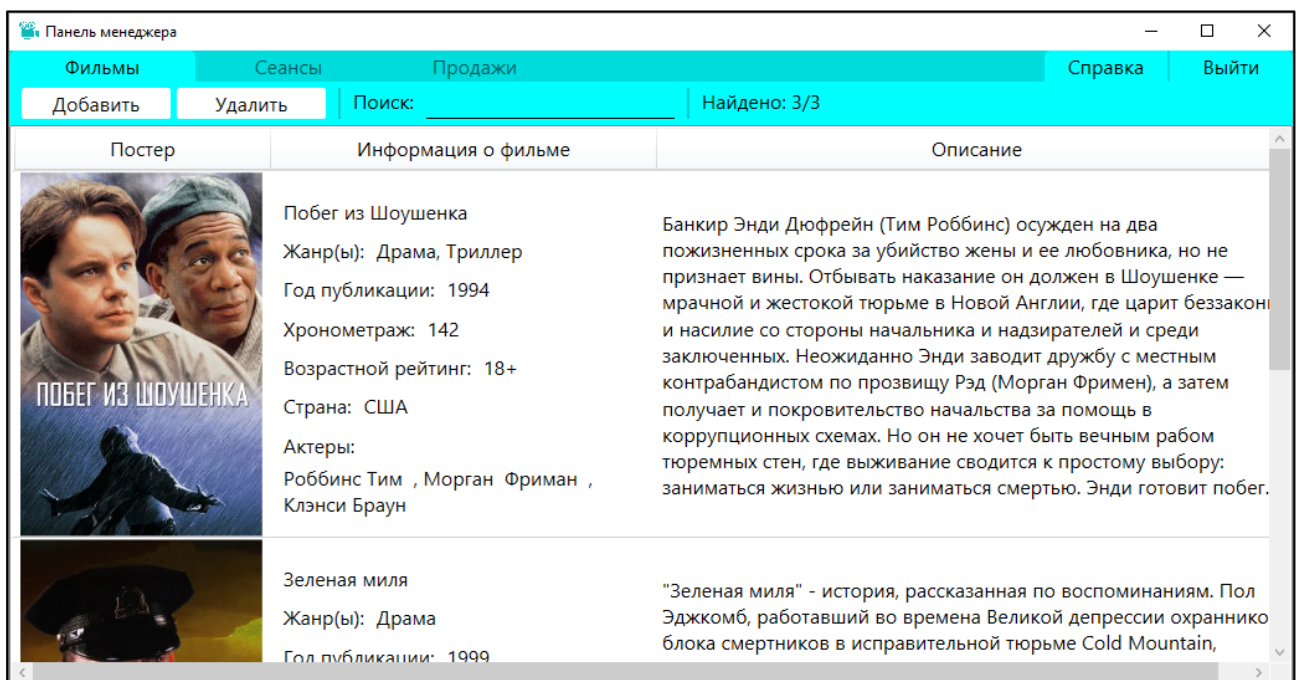


Рисунок 9 – Панель менеджера с открытым окном фильма



Окно добавления фильма поддерживает добавление, изменение и удаление жанров для этого нажать соответствующую кнопку, а в случае редактирования и удаления необходимо предварительно выбрать жанр из списка (Рисунок 10). Для выбора жанра необходимо дважды нажать на нужный жанр.

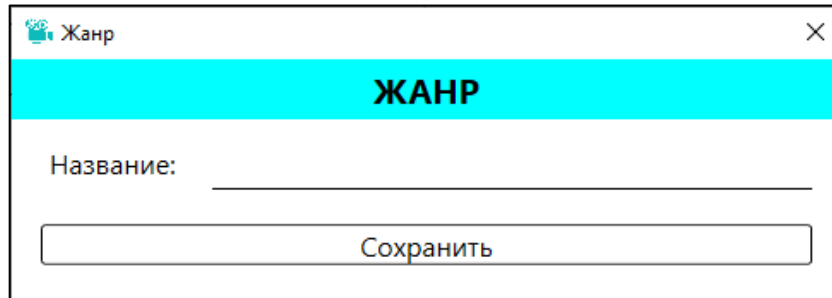


Рисунок 10 – Окно добавления жанра

Окно добавления фильма также поддерживает добавление, изменение и удаление актеров для этого нажать соответствующую кнопку, а в случае редактирования и удаления необходимо предварительно выбрать актера из списка (Рисунок 11).



Рисунок 11 – Окно добавления жанра

Для того чтобы добавить фильм необходимо нажать на кнопку “Добавить” после чего откроется окно добавления фильма (Рисунок 12).

**Фильм**

Название: \_\_\_\_\_

Страна производства: Австралия ▾

Год публикации: \_\_\_\_\_

Хронометраж (мин): \_\_\_\_\_

Возрастной рейтинг: \_\_\_\_\_

Описание: \_\_\_\_\_

Обложка: \_\_\_\_\_

Загрузить

Сохранить

Жанры: \_\_\_\_\_ X

Жанр

Драма
Триллер
Психологический фильм ужасов
Научная фантастика

Добавить Редактировать Удалить

Актеры: \_\_\_\_\_ X

Фамилия	Имя	Отчество	Прозвище
Роббинс	Тим		
Морган	Фриман		
Клэнси	Браун		
Хэнкс	Том		
Пеппер	Барри		

Добавить Редактировать Удалить

Рисунок 12 – Окно добавления фильма

После добавления данных о фильме приложение отобразит сообщение о успешном сохранение (Рисунок 13).

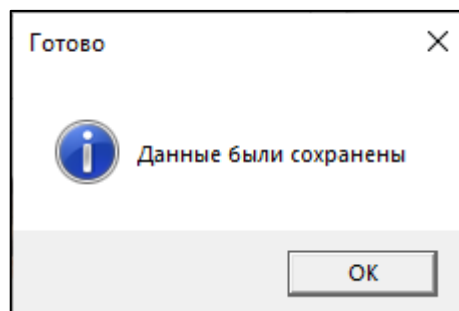


Рисунок 13 – Сообщение о успешном сохранение данных

Для редактирования данных фильма необходимо дважды нажать на редактируемый фильм, после чего откроется окно редактирования фильма (Рисунок 12).

Для удаления фильма необходимо выбрать фильм и нажать на кнопку удалить после чего отойти сообщение-подтверждение удаления (Рисунок 14).

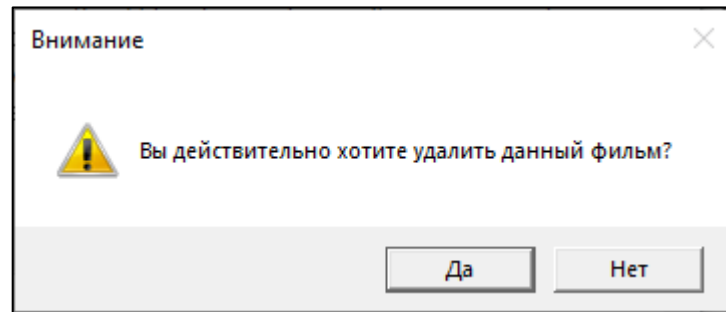


Рисунок 14 – Сообщение подтверждение удаления фильма

Также панель менеджера содержит окно сеансов (Рисунок 15). Здесь мы можем найти и просмотреть информацию о сеансах, а также добавить удалить или редактировать сеанс.

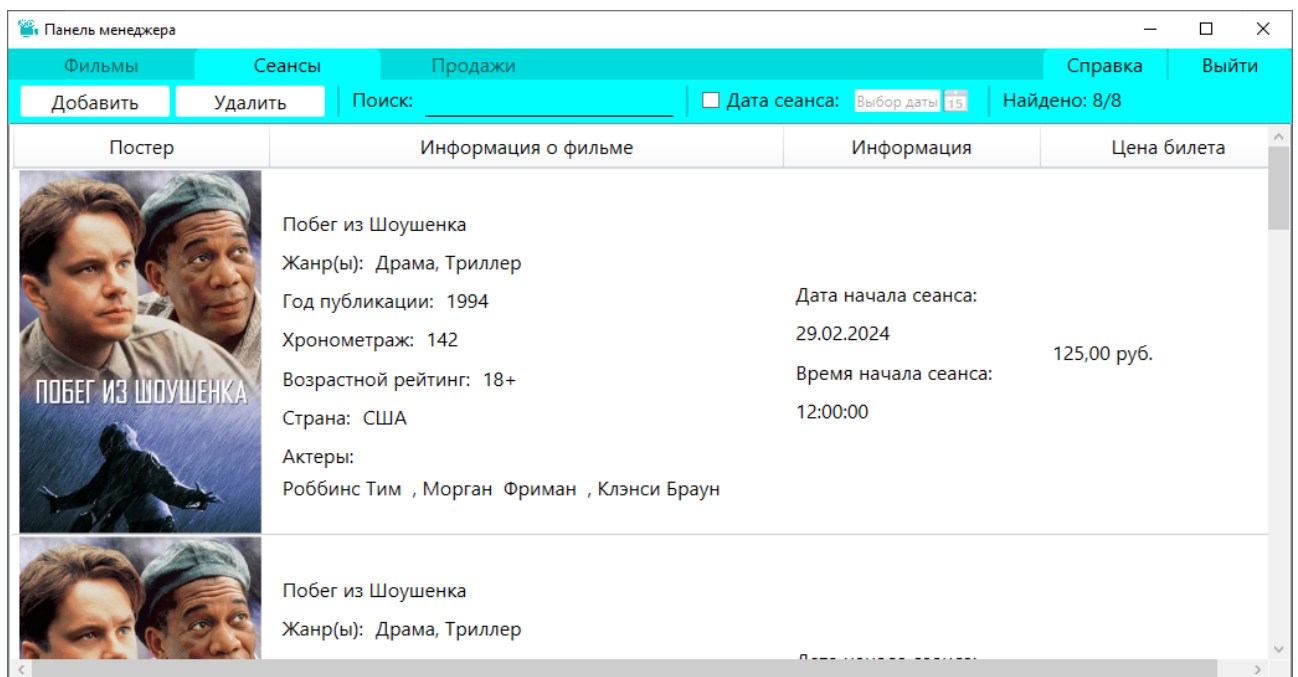



Рисунок 15 – Панель менеджера с открытым окном сеансы

Для того чтобы добавить сеанс необходимо нажать на кнопку “Добавить” после чего откроется окно добавления сеанса (Рисунок 16).



The screenshot shows a window titled 'Сеанс' (Session) with a cyan header bar containing the word 'СЕАНС'. Below the header, there are four input fields: 'Фильм:' (Movie) with a dropdown menu showing '1|Побег из Шоушенка' (1|Escape from Shawshank), 'Дата сеанса:' (Session date) with a date picker showing '15', 'Время сеанса:' (Session time) with an empty text field, and 'Цена билета:' (Ticket price) with an empty text field. At the bottom of the window is a button labeled 'Сохранить' (Save).

Рисунок 16 – Окно добавления сеанса

После добавления данных о сеансе приложение отобразит сообщение о успешном сохранение (Рисунок 17).

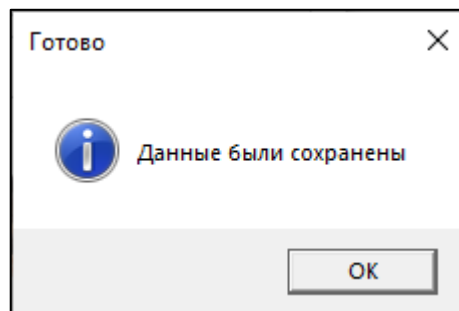


Рисунок 17 – Сообщение о успешном сохранение данных

Для редактирования данных сеанса необходимо дважды нажать на редактируемый сеанс, после чего откроется окно редактирования сеанса (Рисунок 16).

Для удаления сеанса необходимо выбрать сеанс и нажать на кнопку удалить после чего отроиться сообщение-подтверждение удаления (Рисунок 18).

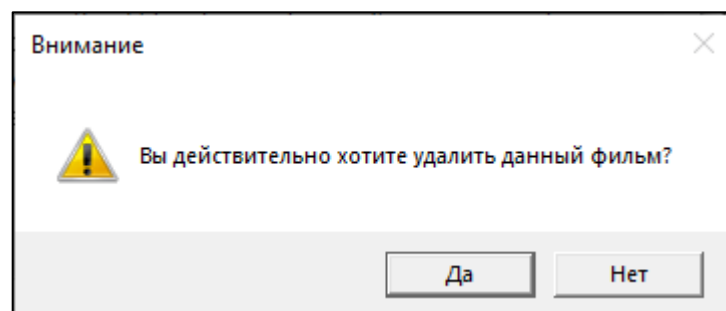


Рисунок 18 – Сообщение подтверждение удаления фильма

Также панель менеджера содержит окно продаж для просмотра и анализа данных по проданным билетам (Рисунок 19). На данном окне можно найти и просмотреть информацию о проданных билетах на фильмы. А также пользователь может сохранить информацию в файл формата PDF.

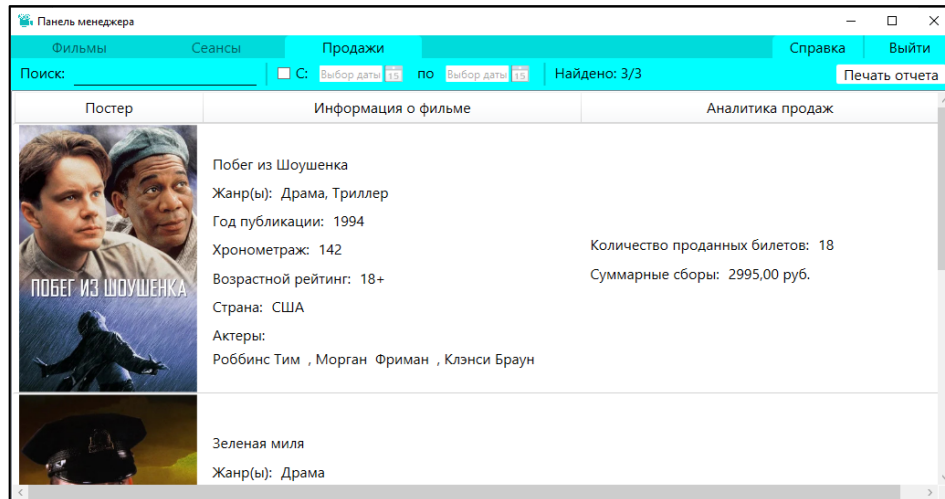


Рисунок 19 – Панель менеджера с открытым окном продаж

Для более подробного просмотра проданных билетов пользователь может нажать на интересующий его фильм и перед ним откроется таблица продаж билетов по сеансам (Рисунок 20). На этом окне также можно найти интересующий вас сеанс, а также сохранить информацию в файл формата PDF.

The screenshot shows the 'Панель менеджера' (Manager Panel) with the 'Продажи' (Sales) tab selected. The interface includes a search bar, a list of movies, and detailed information for 'Побег из Шоушенка' (The Shawshank Redemption). The details include the genre (Drama, Thriller), release year (1994), runtime (142), age rating (18+), country (USA), and cast (Tim Robbins, Morgan Freeman, Clemens Braun). The sales data shows 18 tickets sold for a total of 2995,00 rubles.

Информация	Аналитика продаж
<p>Дата сеанса: 29.02.2024</p> <p>Время сеанса: 12:00:00</p>	<p>Количество проданных билетов: 2</p> <p>Цена билета: 125,00 руб.</p> <p>Суммарные сборы: 250,00 руб.</p>
<p>Дата сеанса: 29.05.2024</p> <p>Время сеанса: 6:00:00</p>	<p>Количество проданных билетов: 1</p> <p>Цена билета: 120,00 руб.</p> <p>Суммарные сборы: 120,00 руб.</p>
<p>Дата сеанса: 28.12.2024</p> <p>Время сеанса: 9:50:00</p>	<p>Количество проданных билетов: 15</p> <p>Цена билета: 175,00 руб.</p> <p>Суммарные сборы: 2625,00 руб.</p>
<p>Дата сеанса: 06.12.2024</p> <p>Время сеанса: 6:30:00</p>	<p>Количество проданных билетов: 0</p> <p>Цена билета: 150,00 руб.</p> <p>Суммарные сборы: 0,00 руб.</p>

Рисунок 20 – Панель менеджера с открытым окном подробного просмотра продаж билетов

## Кассир:

После авторизации под ролью кассира мы видим панель кассира (Рисунок 21). Она открывается на окне “Главная страница” здесь мы можем найти и просмотреть информацию о фильмах.

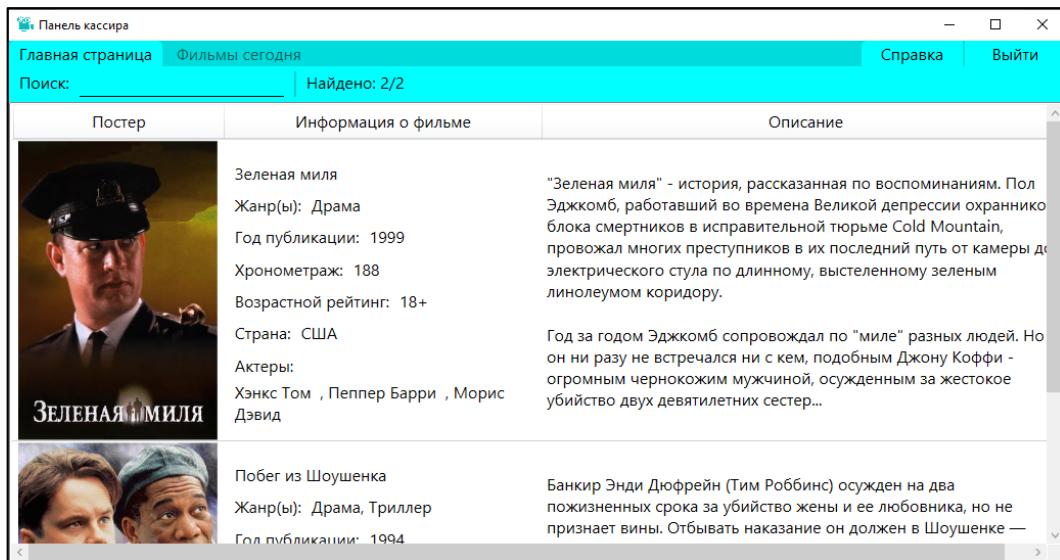


Рисунок 21 – Панель кассира с открытым окном главной страницы

На данном окне пользователь может выбрать фильм, на который нужно забронировать билет. После выбора фильма откроется окно выбора сеансов на выбранный фильм (Рисунок 22). На открытом окне пользователь может выбрать актуальные сеансы.

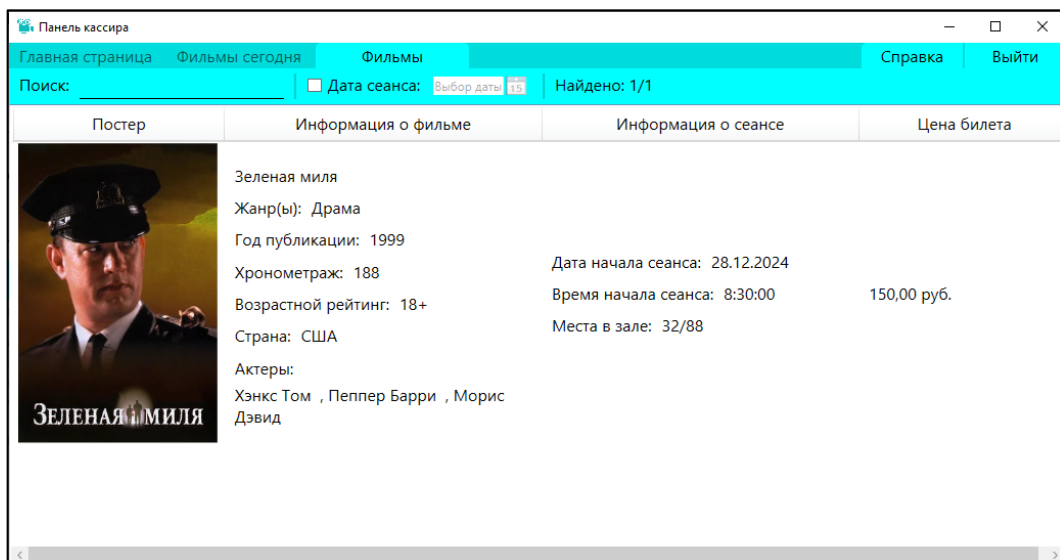


Рисунок 22 – Панель кассира с открытым окном сеансов

После выбора сеанса откроется окно для выбора места в зале (Рисунок 23).  
На данном окне пользователь может выбрать место в зале.

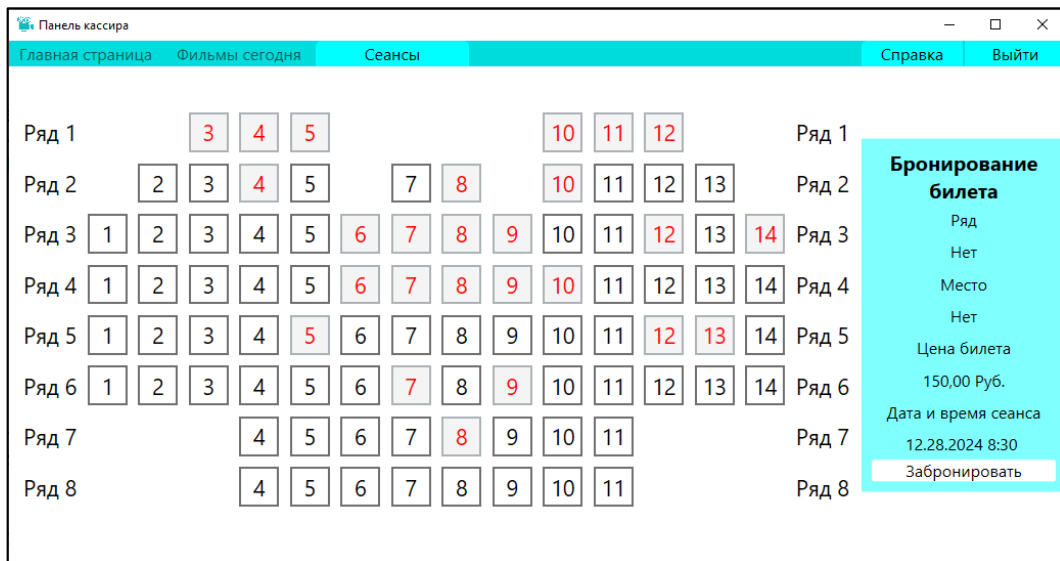


Рисунок 23 – Панель кассира с открытым окном бронирования места в зале

После подтверждения сообщения о бронирование билета (Рисунок 24), он автоматически откроется в PDF редакторе (Рисунок 25).

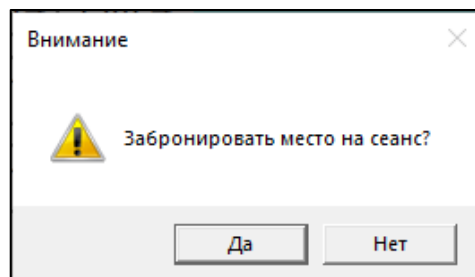


Рисунок 24 – Окно подтверждения бронирования билета

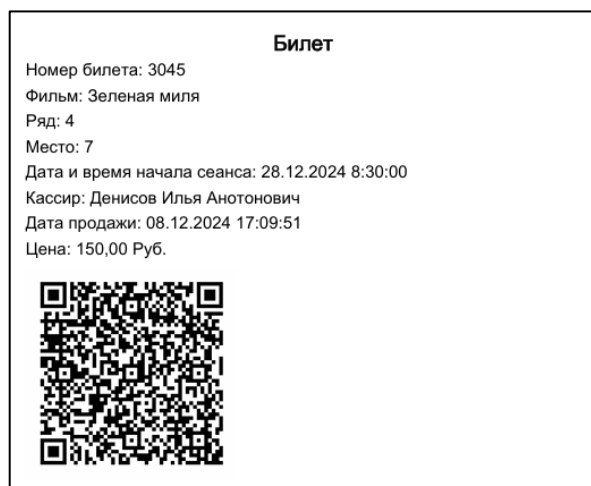


Рисунок 25 – Билет на бронируемый фильм

Для более удобной работы панель кассира обладает возможностью просмотра фильмов, которые буду проходить сегодня (Рисунок 26).

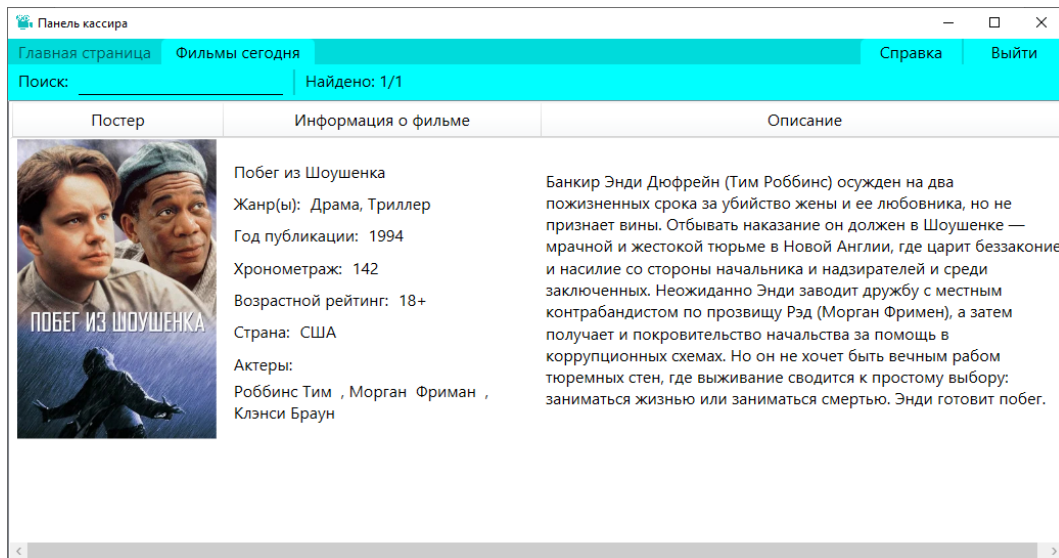


Рисунок 26 – Панель кассира с открытым окном фильма сегодня

### Директор:

После авторизации под ролью директора мы видим панель директора (Рисунок 27). Она открывается на окне “Сотрудники” здесь мы можем найти и просмотреть информацию о сотрудниках, а также добавить удалить или редактировать сотрудников.

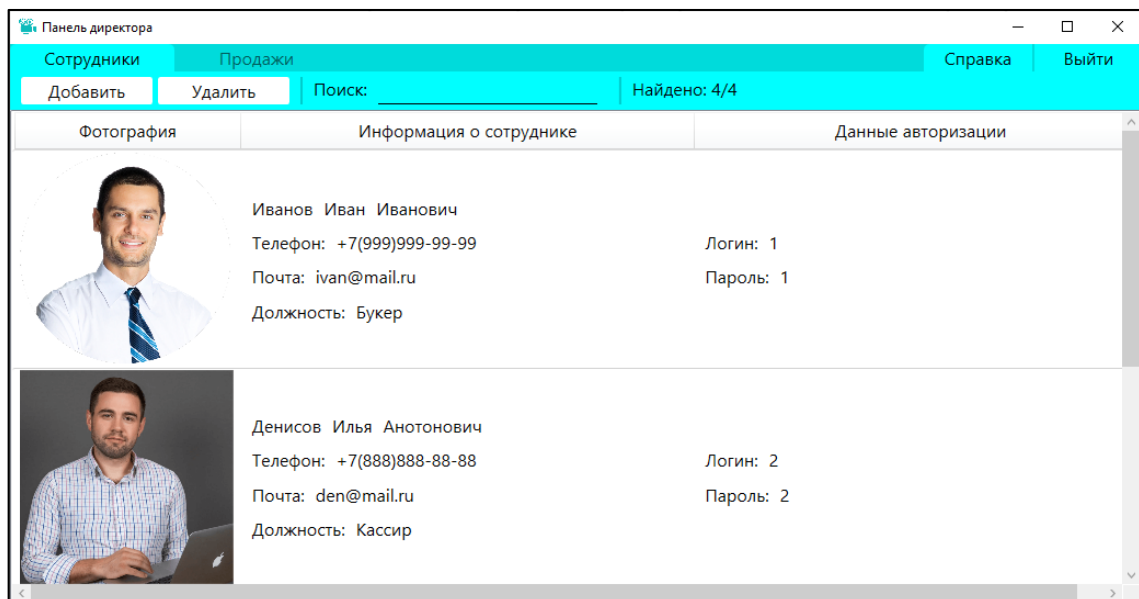


Рисунок 27 – Панель директора с открытым окном сотрудники



Для того чтобы добавить сотрудника необходимо нажать на кнопку “Добавить” после чего откроется окно добавления сотрудника (Рисунок 28).

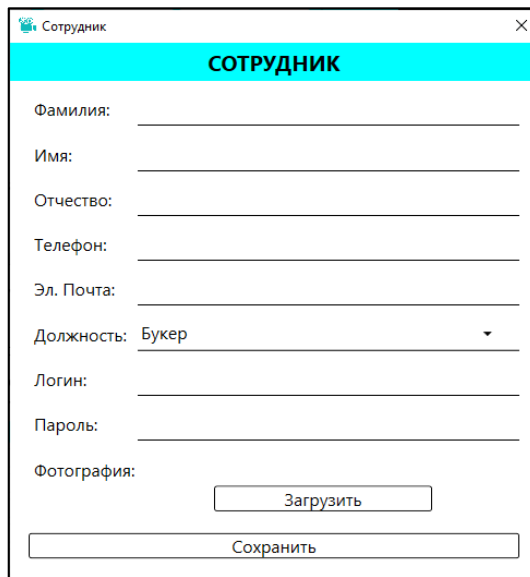


Рисунок 28 – Окно добавления сотрудника

После добавления данных о сотруднике приложение отобразит сообщение о успешном сохранение (Рисунок 29).

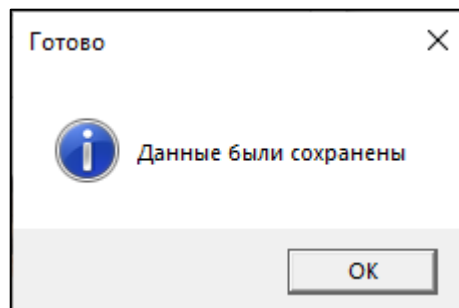


Рисунок 29 – Сообщение о успешном сохранение данных

Для редактирования данных сотрудника необходимо дважды нажать на редактируемого сотрудника, после чего откроется окно редактирования сотрудника (Рисунок 28).

Для удаления сотрудника необходимо выбрать сотрудника и нажать на кнопку удалить после чего отойти сообщение-подтверждение удаления (Рисунок 30).

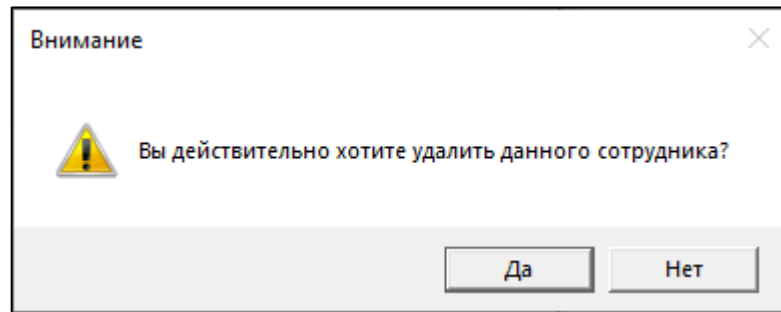


Рисунок 30 – Сообщение подтверждение удаления сотрудника

Также панель директора содержит окно продаж для просмотра и анализа данных по проданным билетам (Рисунок 31). На данном окне можно найти и просмотреть информацию о проданных билетах на фильмы. А также пользователь может сохранить информацию в файл формата PDF.

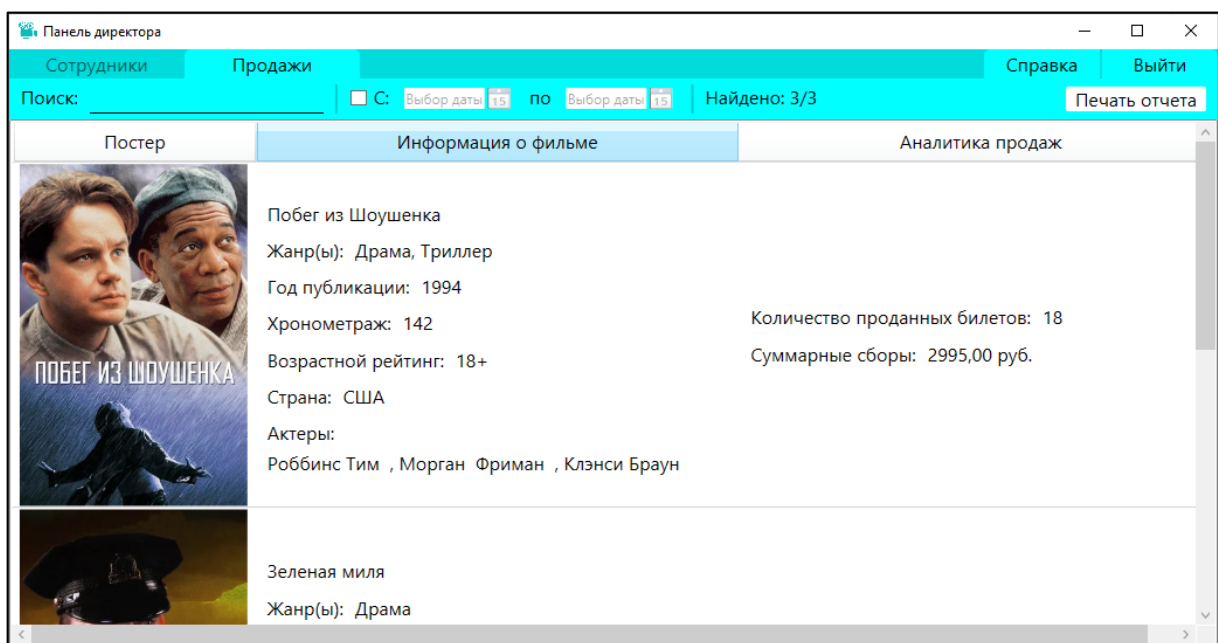


Рисунок 31 – Панель директора с открытым окном продаж

Для более подробного просмотра проданных билетов пользователь может нажать на интересующий его фильм и перед ним откроется таблица продажи билетов по сеансам (Рисунок 32). На этом окне также можно найти интересующий вас сеанс, а также сохранить информацию в файл формата PDF.

Панель директора	
Сотрудники	Продажи
Назад	С: Выбор даты 15 по Выбор даты 15 Найдено: 5/5 Печать отчета
Информация	Аналитика продаж
Дата сеанса: 13.12.2024 Время сеанса: 12:00:00	Количество проданных билетов: 2 Цена билета: 125,00 руб. Суммарные сборы: 250,00 руб.
Дата сеанса: 29.05.2024 Время сеанса: 6:00:00	Количество проданных билетов: 1 Цена билета: 120,00 руб. Суммарные сборы: 120,00 руб.
Дата сеанса: 28.12.2024 Время сеанса: 9:50:00	Количество проданных билетов: 15 Цена билета: 175,00 руб. Суммарные сборы: 2625,00 руб.
Дата сеанса: 06.12.2024 Время сеанса: 6:30:00	Количество проданных билетов: 0 Цена билета: 150,00 руб. Суммарные сборы: 0,00 руб.

Рисунок 32 – Панель менеджера с открытым окном подробного просмотра  
продажи билетов

Программный продукт поставляется на USB накопителе.

На USB накопителя находиться:

- Установщик программы (Cinema installer\setup.exe)
- Проект программного продукта
- Скрипт базы данных без тестовых данных
- Скрипт базы данных с тестовыми данными
- Файл курсового проекта в формате Microsoft Word (.docx)
- Файл исходного кода программного продукта в формате Microsoft

Word (.docx)

- Файл с презентацией курсового проекта

Также программный продукт можно получить, введя ссылку:

- Облако Mail: <https://cloud.mail.ru/public/EksB/woM94C1Zb>
- GitHub: <https://github.com/ALTERRA4546/Cinema>