СОДЕРЖАНИЕ

[Введение 4](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233859)

[1 Экономическая характеристика объекта 5](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233860)

[2 Постановка задачи 6](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233861)

[2.1 Экономическое содержание задачи 6](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233862)

[2.2 Входная информация 6](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233863)

[3 Технические и инструментальные средства для решения задачи 7](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233864)

[3.1 Обоснование выбора языка программирования 7](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233865)

[3.2 Технические требования к компьютеру для решения задачи 9](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233866)

[4 Информационное обеспечение задачи 10](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233867)

[4.1 Классификаторы, используемые для решения задачи 10](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233868)

[5 Программное обеспечение задачи 13](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233869)

[5.1 Описание методов 13](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233870)

[5.2 Алгоритм решения задачи 15](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233871)

[5.3 Описание блок – схемы 15](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233872)

[Заключение 19](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233873)

[Список использованных источников 20](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233874)

[Приложение А 21](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233875)

[Приложение Б 22](file:///D:\Office\Kursovaya_Osmanov.docx#_Toc3233876)

**ВВЕДЕНИЕ**

В данной курсовой работе рассматривается проблема отсутствия свободных мест на нужный сеанс во время прихода в кинотеатр. Следствием чего является впустую потраченное время и испорченное настроение. Целью данной курсовой работы является решение данной проблемы путем реализации программного продукта на одном из языков программирования. Задача проектирования состоит в том, чтобы максимально просто добиться результата поставленной задачи. Было предложено реализовать программный продукт, который позволял бы пользователям покупать билеты онлайн – “Система покупки билетов онлайн”.

Для решения данной проблемы были выдвинуты следующие цели:

* пользователи (посетители кинотеатра) могли приобретать билеты онлайн, не выходя из дому;
* пользователи могли просматривать расписание сеансов на неделю вперёд, в том числе занятость зала на интересующий их сеанс;
* администратор (работник кинотеатра) мог бы добавлять или удалять фильмы, сеансы; изменять количество мест в зале, цены билетам; а также продавать билеты на сеансы.

# 1 Экономическая характеристика объекта

**Состав и взаимосвязь помещений**

Кинотеатр представляет собой учреждение, занимающееся организацией кинопоказов и проведением иных развлекательных мероприятий, связанных с кино. Возведение отдельно стоящего однозального кинотеатра стоит примерно 2-3 млн долларов. Эта сумма вполне приемлема при наличии инвестиционных источников и учитывая перспективы будущего бизнеса. Она быстро окупится, и через два года проект начнет приносить значительную прибыль.

Все помещения кинотеатра подразделяют на следующие комплексы и группы: помещения зрительского комплекса;. помещения демонстрационного комплекса — зрительный зал, помещения технологического обеспечения кинопоказа; административно-хозяйственные и производственные помещения; технические помещения.

Каждая из этих групп помещений получает развитие, соответствующее характеру и вместимости здания. Расчет площадей помещений приводится в нормах проектирования культурно-зрелищных учреждений ВСН 45-86. Взаиморасположение групп должно создавать простой и удобный график движения зрителей, оптимальные условия технологического процесса демонстрации фильмов и пожарную безопасность.  
  
**Помещения зрительского комплекса**  
  
Неспециализированных кинотеатров включают: кассовый вестибюль с помещениями касс, входной вестибюль, фойе или распределительные кулуары, буфет с подсобными помещениями, курительную и санитарные узлы. В неспециализированных кинотеатрах для строительства в 1A, 1Б, 1Г климатических подрайонах допускается проектирование выходного вестибюля, гардероба, комнаты для переодевания и независимо от климатических подрайонов — комнаты игровых автоматов и детской комнаты.  
  
Проектирование специализированных кинотеатров ведется по индивидуальным заданиям, в которых определяется состав помещении, отвечающий назначению кинотеатра и конкретным условиям места строительства.  
  
Премьерный кинотеатр — это, как правило, крупнейший кинотеатр города с большим залом универсального назначения, возможно, и с малыми специализированными залами. В кинотеатрах этого типа помимо основного состава помещений зрительского комплекса рекомендуется предусматривать гостиную, аудиторию (просмотровый зал), выставочный зал. Рекомендуется устраивать вестибюль с гардеробом.  
  
Досуговый кинотеатр проектируют с расчетом на посетителей, желающих не только посмотреть кинофильм, но и отдохнуть, потанцевать. Такой кинотеатр может иметь гардероб при вестибюле, развитые фойе и буфет или кооперироваться с кафе и танцзалом, иметь гостиную, бар, зимний сад, зал игровых автоматов. Этот тип кинотеатра может быть двухзальным, причем больший зал рекомендуется делать универсальным, а при эстраде проектировать артистические.  
  
Студийный кинотеатр рассчитан не только на демонстрацию кинофильмов, но на работу по пропаганде киноискусства. Особенностью такого кинотеатра является развитый состав клубных помещений, включающий гостиную, аудиторию, выставочный зал.  
  
Детский кинотеатр имеет развитый состав клубных помещений с расчетом на возраст посетителей. Здесь предусматривают зал игровых автоматов, помещение для настольных игр, выставочный зал, комнаты педагогов, возможно, зимний сад.  
  
Кинотеатры с непрерывным кинопоказом — обычно это кинотеатры хроникально-документального фильма, предназначенные для зрителей, ограниченных временем (у вокзалов, в торговых центрах). В таких кинотеатрах допускается сокращенный состав помещений зрительского комплекса.  
  
Вестибюль во всех типах кинотеатров состоит из двух частей — кассовый и входной вестибюли. Кассовый вестибюль имеет самостоятельный вход и должен быть связан через дверь с входным вестибюлем так, чтобы посетитель, купив билеты, мог в него пройти, не выходя, на улицу. Площадь кассового вестибюля определяется из расчета 0,07 м2 на одно место в зрительном зале. Количество кассовых кабин надо принимать из расчета одна кабина на 400 мест в зале при площади кабины 2,5 м2. Кроме того, необходимо предусмотреть кабину старшего кассира площадью 4 м2 и кабинет дежурного администратора площадью 6 м.

В малых кинотеатрах кассы могут располагаться во входном вестибюле. При этом кассы следует размещать в стороне от путей прохода в зал. Входной вестибюль является местом для постепенного накопления зрителей и кратковременного ожидания. Его площадь определяют из расчета 0,2 м2 на одно место в зрительном зале.  
  
Из входного вестибюля, пройдя пост контроля билетов, зрители попадают в фойе или распределительные кулуары — помещения, в которых зрители накапливаются в ожидании начала сеанса. При фойе в стороне от входов в зрительный зал располагают курительные и санитарные узлы либо на уровне фойе, либо уровнем ниже.  
  
Помещения фойе и кулуаров проектируют в зависимости от типа кинотеатра. В неспециализированных кинотеатрах площадь фойе определяют из расчета 0,4 м2 на место. В кинотеатрах с универсальными залами, где на площади фойе могут быть расположены эстрада и места для зрителей, площадь принимают 0,45 м2 на место, в детских кинотеатрах — 0,6 м2, кинотеатрах с непрерывным кинопоказом — 0,25 м2 на место (включая буфет).  
  
Буфет с подсобными помещениями располагают смежно с фойе. Его площадь с подсобным помещением определяют из расчета 0,25 м2 на одно место в зрительном зале. Подсобная буфета, оборудованная мойкой, непосредственно связана со стойкой буфета (площадь подсобной — не менее 12 м2). При проектировании кинотеатров с универсальными залами и детских кинотеатров площадь буфета принимают 0,28 м2 на расчетное место.  
  
Санитарные узлы проектируют из расчета соотношения мужчин и женщин 1:2. Количество приборов принимают из расчета: один умывальник на 100 человек; в мужских санузлах — один унитаз и два писсуара на 100 человек, в женских — один унитаз на 30 человек.  
  
Помещения демонстрационного комплекса. В кинотеатрах демонстрационный комплекс включает зрительный зал и помещения технологического обеспечения кинопоказа, в кинотеатрах с универсальными залами, кроме того, эстраду и помещения ее технологического обеспечения.  
  
Зрительный зал является основным помещением кинотеатра. Планировка и оборудование зрительного зала должны создавать условия для хорошей видимости со всех мест, нормальной акустики, удобного распределения зрителей по местам и быстрой их эвакуации. Этими основными задачами и определяются соотношения размеров зала, его форма и объем, размеры и расположение экрана, размещение мест и проходов между ними.  
  
Размеры, форма и объем зала зависят в первую очередь от его вместимости. Площадь зала принимают из расчета 1,0 м2 на одного зрителя в кинотеатрах круглогодичного действия и 0,9 м2 — в кинотеатрах сезонного действия (площадь определяют в пределах ограждающих конструкций, включая эстраду).  
  
Основой формообразования зала являются условия кинопроекции и требование расположения зрительских мест в зоне оптимальной видимости. Именно это определяет габариты зала (длину, ширину, высоту) и профиль его пола, размеры и размещение экрана, характер расположения мест.  
  
  
2. Постановка задачи

Предметная область, подлежащая изучению –«кинотеатр». В сферу этой предметной области попадают предприятия, занимающиеся оказанием услуг проведения досуга. Цель функционирования предприятий этой предметной области – оказание услуг, таких как: показ кинофильмов, реклама и анонс будущих фильмов и получение прибыли, извлекаемой из оплаты этих услуг клиентами. Для оказания услуг необходимо наличие квалифицированных специалистов и помещений, удовлетворяющих нормам санитарных и других требований в соответствии с действующим законодательством.

***Целью курсовой работы является*** создание информационно-программной системы для решения следующих задач:

1. автоматизации операций осуществления продажи билетов.
2. централизованного хранения информации о фильмах с целью:
   1. их вторичного показа,
   2. продажи прокатного удостоверения.
3. обеспечения хранения информации о количестве посетителей заведения с целью:
   1. сбора информации о посещаемости фильма,
   2. подсчета вырученных средств от продажи билетов,
   3. возможности увеличения или уменьшения количества сеансов.

К разрабатываемой системе предъявлены следующие требования.

1. наличие централизованного хранилища данных для всех объектов автоматизированного учета,
2. ведение операций информационного учета с компьютеров локальной сети предприятия,
3. создание унифицированного доступа к данным с использованием специализированного программного интерфейса пользователя, который бы отражал специфику этой предметной области.

Для решения поставленных задач планируется выполнение следующих действий:

1. проектирование данных для отражения специфики данной предметной области,
2. разработка и создание базы данных для хранения информации на едином сервере данных под управлением MS SQL Server ,
3. разработка компьютерной программы для выполнения базовых операций над данными предметной области, которая выполняет следующие функции:
   1. добавление информации об основных объектах (фильмах и мультфильмах) редактирование информации по основным объектам,
   2. поиск данных по основным объектам учёта:
      1. фильмов;
      2. посетителей;
      3. сеансов;
   3. получение итоговой информации по отчетным характеристикам системы:
      1. получение суммарной выручки от продажи всех билетов,
      2. наиболее посещаемые фильмы.

## 2.1 Экономическое содержание задачи

Деятельность кинотеатра направлена на удовлетворение развлекательной и культурно-просветительной потребности населения. На основе миссии определяются цели деятельности. Следовательно, целью деятельности данного кинотеатра является максимально большее привлечение кинозрителей к просмотру максимально большего количества фильмов разнообразных жанров и тематик, для развлечения кинозрителей и оказания ни них культурно-научно-просветительского воздействия.

Рассмотрим достоинства и недостатки существующей организационной структуры компании. Управление компанией носит линейный характер, основанный на единоначалии. К основным недостаткам такого метода управления можно отнести следующие:

1. Высокие требования к руководителю, который должен иметь багаж всесторонних знаний, необходимых для управления компанией. Остановимся на этой проблеме управления компанией более подробно. Управление компанией на уровне генерального директора предполагает решение вопросов по позиционированию на рынке реализуемого продукта или услуги, уменьшение затрат на закупку и реализацию, подбор нужных кадров, т.е. объем знаний и умений, в котором сложно быть компетентным одному человеку.
2. Линейная структура лишена необходимых промежуточных звеньев по планированию и подготовке решений, что приводит к отсутствию видения ближайшей перспективы развития компании и негативно отражается на ее будущей конкурентоспособности.
3. Средние уровни управления компанией перегружены лишней организационной информацией, что мешает им решать на должном уровне конкретные необходимые для развития компании задачи.
4. Излишняя перегруженность средних уровней ослабляет связи между ними, что влияет на качество принимаемых на этих уровнях управленческих решений.
5. Так как общее управление компанией организовано по линейному принципу, при отсутствии одного из ключевых руководителей нарушается работа как одного из звеньев компании, так и всей компании целиком. Это способствует дополнительным рискам в работе компании и делает ее менее конкурентоспособной. Ещё одним недостатком в управлении компанией является программная автономия ее отдельных подразделений, занимающихся реализацией билетов.

Для устранения описанных недостатков управления компанией предлагается сделать ее более интегрированной и постепенно перевести механизм управления с линейного на функциональный, используя для этого средства дополнительной автоматизации процесса продажи билетов.

## 2.2 Входная информация

Входной является информация, которую пользователь вносит в файл базы данных, заполняя необходимые поля выбранной таблицы, а также вводя информацию в базу данных с помощью запросов SQL.

Для разрабатываемого программного продукта входной будет служить следующая информация:

- информация о сеансах (время проведения сеанса, стоимость билета на данный сеанс);

- информация о купленных билетах (дата реализации билета, место и ряд в зрительном зале);

- информация об имеющихся фильмах (название фильма, жанр, возрастные ограничения на просмотр данного фильма).

В ИС используется несколько справочников, которые формируют нормативно-справочную, входную и оперативную информацию.

В информационной системе используются следующие справочники:

* Фильмы;
* Сеансы;
* Билеты.

В системе за счет использования запросов сформированы документы:

* Список сеансов;
* Список ID билетов;
* Статистика.

2.3 Выходная информация

Выходная информация - результат выполнения запросов, фильтрации данных, вывод необходимой информации в отчет, печать информации. Информация, которая несет вывод либо обобщает указанные данные в общем виде или по определенному критерию.

Выходной информацией для данного проекта является информация, которая позволяет произвести вывод на печать отчетной формы: список реализованных билетов. Вывод информации о выручке кинотеатра за определенный период.

Количество экранных форм и их содержание спроектировано так, чтобы обеспечить возможность вывода всей результатной информации, а также результатов запросов пользователя к базе данных. Предусмотрены выдача результатов как на экран ПЭВМ, так и на принтер по желанию пользователя. Вся результатная информация проекта автоматизации представляется в виде отчетов и экранных форм. Ниже рассмотрены детали каждого из результатных документов.

Выходными документами разрабатываемой программы являются:

* отчет по продажам;
* отчет по сеансам;
* отчет по возвратам.

# 3 Технические и инструментальные средства для решения задачи

## 3.1 Обоснование выбора языка программирования

C# — современный объектно-ориентированный и типобезопасный язык программирования. C# позволяет разработчикам создавать разные типы безопасных и надежных приложений, выполняющихся в .NET. C# относится к широко известному семейству языков C, и покажется хорошо знакомым любому, кто работал с C, C++, Java или JavaScript. Здесь представлен обзор основных компонентов языка C# 8 и более ранних версий. Если вы хотите изучить язык с помощью интерактивных примеров, рекомендуем поработать с [вводными руководствами по C#](https://docs.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/tutorials/).

C# — ориентированный на компоненты язык программирования. C# предоставляет языковые конструкции для непосредственной поддержки такой концепции работы. Благодаря этому C# подходит для создания и применения программных компонентов. С момента создания язык C# обогатился функциями для поддержки новых рабочих нагрузок и современными рекомендациями по разработке ПО. Вы определяете типы и их поведение.

Вот лишь несколько функций языка C#, которые позволяют создавать надежные и устойчивые приложения. [Сборка мусора](https://docs.microsoft.com/ru-ru/dotnet/standard/garbage-collection/) автоматически освобождает память, занятую недостижимыми неиспользуемыми объектами. [Типы, допускающие значение null,](https://docs.microsoft.com/ru-ru/dotnet/csharp/nullable-references) обеспечивают защиту от переменных, которые не ссылаются на выделенные объекты. [Обработка исключений](https://docs.microsoft.com/ru-ru/dotnet/csharp/fundamentals/exceptions/) предоставляет структурированный и расширяемый подход к обнаружению ошибок и восстановлению после них. [Лямбда-выражения](https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/operators/lambda-expressions) поддерживают приемы функционального программирования. [Синтаксис LINQ](https://docs.microsoft.com/ru-ru/dotnet/csharp/linq/) создает общий шаблон для работы с данными из любого источника. Поддержка языков для [асинхронных операций](https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/concepts/async/) предоставляет синтаксис для создания распределенных систем. В C# имеется [Единая система типов](https://docs.microsoft.com/ru-ru/dotnet/csharp/fundamentals/types/). Все типы C#, включая типы-примитивы, такие как int и double, наследуют от одного корневого типа object. Все типы используют общий набор операций, а значения любого типа можно хранить, передавать и обрабатывать схожим образом. Более того, C# поддерживает как определяемые пользователями [ссылочные типы](https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/builtin-types/reference-types), так и [типы значений](https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/builtin-types/value-types). C# позволяет динамически выделять объекты и хранить упрощенные структуры в стеке. C# поддерживает универсальные методы и типы, обеспечивающие повышенную безопасность типов и производительность. C# предоставляет итераторы, которые позволяют разработчикам классов коллекций определять пользовательские варианты поведения для клиентского кода.

C# подчеркивает Управление версиями , чтобы обеспечить совместимость программ и библиотек с течением времени. Вопросы управления версиями существенно повлияли на такие аспекты разработки C#, как раздельные модификаторы virtual и override, правила разрешения перегрузки методов и поддержка явного объявления членов интерфейса.

Главные возможности среды разработки Visual Studio:

Интегрированная среда разработки (IDE) — это многофункциональная программа, которая поддерживает многие аспекты разработки программного обеспечения. Интегрированная среда разработки Visual Studio — это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Помимо стандартного редактора и отладчика, которые есть в большинстве сред IDE, Visual Studio включает в себя компиляторы, средства автозавершения кода, графические конструкторы и многие другие функции для улучшения процесса разработки.

Такая среда дает возможность править, компилировать, выполнять и отлаживать программы на C#, не покидая эту грамотно организованную среду.

Visual Studio предоставляет не только все необходимые средства для работы с программами, но и помогает правильно организовать их. Она оказывается наиболее эффективной для работы над крупными проектами, хотя может быть с тем же успехом использована и для разработки небольших программ, например, тех, которые приведены в качестве примера в этом курсе.

## 3.2 Технические требования к компьютеру для решения задачи

Рекомендуемые системные требования к компьютеру на которой будет установлена IDE Visual Studio 2022 следующие:

### Операционные системы:

* Windows 11 версии 21H2 или выше: домашняя, Pro, Pro образования, Pro для рабочих станций, Enterprise и образовательных учреждений.
* Windows 10 версии 1909 или более поздних версий: Домашняя, Профессиональная, для образовательных учреждений и Корпоративная.
* Windows Server 2022: Standard и Datacenter.
* Windows Server 2019: Standard и Datacenter.
* Windows Server 2016: Standard и Datacenter
* Windows Server 2022: Standard и Datacenter.
* Windows Server 2019: Standard и Datacenter.
* Windows Server 2016: Standard и Datacenter.

### Оборудование :

64-разрядный процессор 1,8 ГГц или более мощный. Рекомендуется четырехъядерный или с большим количеством ядер. Процессоры ARM не поддерживаются.

* + - Не менее 4 ГБ ОЗУ. На используемые ресурсы влияет множество факторов; для стандартных профессиональных решений рекомендуется 16 ГБ ОЗУ.
    - Windows 365: не менее 2 виртуальных ЦП и 8 ГБ ОЗУ. Рекомендуется 4 виртуальных ЦП и 16 ГБ ОЗУ.
    - Место на жестком диске: от 850 ГБ до 210 ГБ свободного места в зависимости от установленных компонентов, обычно для установки требуется от 20 до 50 ГБ свободного места. Для повышения производительности рекомендуется устанавливать Windows и Visual Studio на твердотельном накопителе (SSD).
    - Видеоадаптер с минимальным разрешением WXGA (1366 на 768 пикселей); для оптимальной работы Visual Studio рекомендуется разрешение 1920 на 1080 пикселей или выше.
    - Минимальное разрешение предполагает масштабирование, параметры DPI и масштабирование текста на уровне 100%. Если не задано значение 100%, необходимо соответствующим образом масштабировать минимальное разрешение. Например, если задать для дисплея Windows значение параметра "Масштаб и макет" на устройстве Surface Book, которое имеет физический дисплей 3000x2000, значение 200%, то для Visual Studio логическое разрешение экрана будет 1500x1000, что соответствует минимальным требованиям в 1366x768.

### Поддерживаемые языки:

Visual Studio доступна на следующих языках: английский, китайский (упрощенное и традиционное письмо), чешский, французский, немецкий, итальянский, японский, корейский, польский, португальский (Бразилия), русский, испанский и турецкий. Язык Visual Studio можно выбрать во время установки. Установщик Visual Studio доступен на тех же четырнадцати языках и будет соответствовать языку Windows (если он доступен).

### Дополнительные требования и инструкции:

Для установки или обновления Visual Studio требуются права администратора.

* Дополнительные рекомендации и инструкции по установке, развертыванию, обновлению и настройке Visual Studio в организации см. в руководстве администратора Visual Studio.
* Для установки Visual Studio необходима платформа .NET Framework 4.5.2 или выше. Для работы Visual Studio требуется .NET Framework 4.8. Если платформа .NET Framework 4.8 еще не установлена, то будет выполнена ее установка.
* Для разработки универсальных приложений для Windows, включая проектирование, редактирование и отладку, требуется Windows 10. Windows Server 2019 и Windows Server 2016 могут использоваться для создания универсальных приложений для Windows из командной строки.
* Для установки Visual Studio необходима среда выполнения WebView2. Если она еще не установлена, то будет выполнена ее установка. Установка Visual Studio завершится сбоем, если не удастся успешно установить WebView2 (из-за ошибки установки или ограничений групповой политики на установку или обновление).
* Для интеграции Team Foundation Server 2019 с Office требуется Office 2016, Office 2013 или Office 2010.

## Требования к системе для инструментов удаленной отладки и изолированного сборщика IntelliTrace для Visual Studio 2022

Для инструментов удаленной отладки и изолированного сборщика IntelliTrace предъявляются те же требования к системе, что и для Visual Studio, со следующими отличиями:

* Также поддерживаются в Windows 7, Windows Server 2008 с пакетом обновления 2 (SP2), Windows Server 2008 R2 с пакетом обновления 1 (SP1), Windows 10 в ARM, Windows 10 Корпоративная LTSC, Windows Server 2012 R2, Windows 8.1 и Windows Server 2012.
* При использовании архитектуры x86 или AMD64/x64 требуется процессор с тактовой частотой не ниже 1,6 ГГц
* Требуется 1 ГБ ОЗУ (1,5 ГБ при запуске на виртуальной машине)
* Требуется 1 ГБ доступного пространства на жестком диске
* Требуется разрешение экрана не ниже 1024 на 768
* Для получения наилучших результатов используйте средства диагностики с самым последним обновлением для вашей версии Visual Studio.

# 4 Информационное обеспечение задачи

## 4.1 Классификаторы, используемые для решения задачи

Классификатор – это сгруппированный, по различным признакам, перечень  наименованных объектов, где каждому из них присвоен уникальный код, в соответствии с их общими признаками или различиями. Классификаторы применяются для упрощения работы по сбору информации и систематизации данных, а также используются для указания кодов в различных унифицированных документах.

Объект – некоторая сущность в цифровом пространстве, обладающая определённым состоянием и поведением в данной предметной области.

Атрибут – элемент, описывающий любую характеристику объекта, важную для данной предметной области. Объектом в данной базе данных будет: информация о фильмах приведенная в таблице 4.1, залы на таблице 4.2 жанры на таблице 4.3, сеансы на таблице 4.4 и информация о выданных билетах в таблице 4.5

Таблица 4.1 – Атрибуты фильма

|  |  |
| --- | --- |
| Объект | Атрибуты |
| Фильмы | Идентификационный код  Фильм  Жанр  Дата сеанса  Год выхода  Зал |

Таблица 4.2 – Атрибуты зала

|  |  |
| --- | --- |
| Объект | Атрибуты |
| Залы | Идентификационный код  Зал |

Таблица 4.3 – Атрибуты жанров

|  |  |
| --- | --- |
| Объект | Атрибуты |
| жанры | Идентификационный код  жанры |

Таблица 4.4 – Атрибуты сеансов

|  |  |
| --- | --- |
| Объект | Атрибуты |
| Сеансы | Идентификационный код  Фильм  Цена  Дата сеанса  Зал | |

Таблица 4.5 – Атрибуты билетов

|  |  |
| --- | --- |
| Объект | Атрибуты |
| Билеты | Идентификационный код  Фильм  Дата сеанса  Зал  Ряд  Место |

Для выполнения проекта была выбрана СУБД SQL Server. Схема БД представлена на рисунке 4.6.

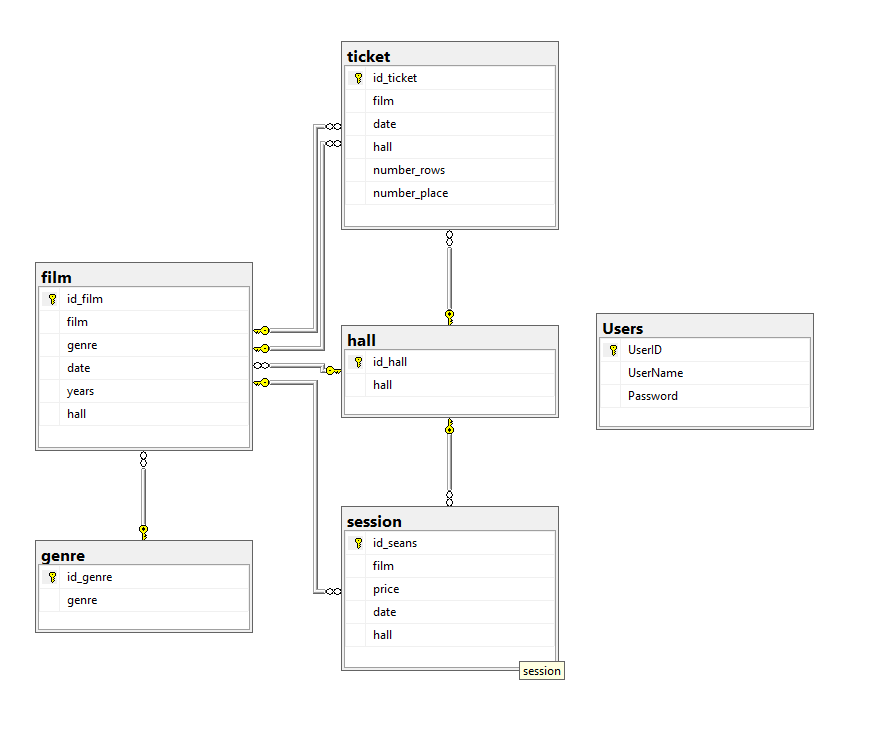


Рисунок 4.6 – Связи таблиц в БД

Структура таблиц базы данных представлена на рисунках 4.7 – 4.11 соответственно.

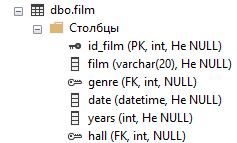


Рисунок 4.7 – Таблица «Фильмы»

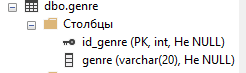


Рисунок 4.8 – Таблица «Жанры»

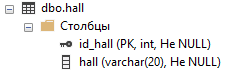


Рисунок 4.9 – Таблица «Залы»

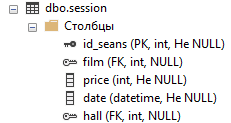


Рисунок 4.10 – Таблица «Сеансы»

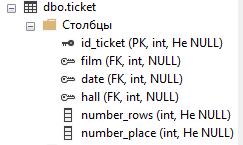


Рисунок 4.11 – Таблица «Билеты»

# 5 Программное обеспечение задачи

## 5.1 Описание методов

Для примера взята часть кода для входа в приложение под своей учетной записью. При нажатии на кнопку вызывается активити, для дальнейшей работы с приложением.

package ru.atomexp.android;

import android.content.Intent;

import android.content.SharedPreferences;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ImageButton;

import android.widget.Toast;

import com.google.gson.Gson;

import com.google.gson.GsonBuilder;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class LoginActivity extends AppCompatActivity {

private Button btnSignIn;

private ImageButton btnQrCode;

private EditText etLogin;

private EditText etPass;

private User user;

private Gson GSON;

public static final String *APP\_PREFERENCES* = "authorizedUser";

public static final String *SAVED\_USER* = "savedUser";

public static SharedPreferences *spUser*;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.*activity\_login*);

*spUser* = getSharedPreferences(*APP\_PREFERENCES*, *MODE\_PRIVATE*);

GSON = new GsonBuilder().create();

btnSignIn = findViewById(R.id.*signInButton*);

btnQrCode = findViewById(R.id.*qrCodeButton*);

etLogin = findViewById(R.id.*loginText*);

etPass = findViewById(R.id.*passwordText*);

}

public void signIn(View view) {

Toast.*makeText*(this, "Нажатие", Toast.*LENGTH\_SHORT*).show();

try {

try {

Class.*forName*("net.sourceforge.jtds.jdbc.Driver");

Toast.*makeText*(this, "Зарегистрирован", Toast.*LENGTH\_LONG*).show();

} catch(Exception ex){

Toast.*makeText*(this, "Не зарегистрирован" + ex.getMessage(), Toast.*LENGTH\_LONG*).show();

}

final String DB\_URL = "jdbc:jtds:sqlserver://192.168.31.80:1433/documents";

final String DB\_USER = "p1onerw0w";

final String DB\_PASSWORD = "4321193";

final Gson GSON = new GsonBuilder().create();

String login = etLogin.getText().toString();

String password = etPass.getText().toString();

String query =

"select \* from users " +

"where (phone = '" + login + "' or email = '" + login + "') " +

"and [password] = '" + password + "'";

try {

Connection connection = DriverManager.*getConnection*(DB\_URL, DB\_USER, DB\_PASSWORD);

Toast.*makeText*(this, "Подключение создано", Toast.*LENGTH\_LONG*).show();

} catch (Exception ex) {

Toast.*makeText*(this, "Не подключен" + ex.getLocalizedMessage(), Toast.*LENGTH\_LONG*).show();

}

if(!*spUser*.contains(*SAVED\_USER*)) {

try {

user = User.*userBuild*(SqlServerService.*getResultSet*(query));

String gsonUser = GSON.toJson(user);

SharedPreferences.Editor editor = *spUser*.edit();

editor.putString(*SAVED\_USER*, gsonUser);

editor.apply();

Intent intent = new Intent(this, MainActivity.class);

intent.putExtra("user", gsonUser);

Toast.*makeText*(this, "Авторизация", Toast.*LENGTH\_SHORT*).show();

startActivity(intent);

} catch(SQLException sqlex) {

Toast.*makeText*(this, "Ошибка SQL", Toast.*LENGTH\_LONG*).show();

} catch(Exception ex) {

Toast.*makeText*(this, "NON-SQL ошибка", Toast.*LENGTH\_LONG*).show();

}

} else {

}

} catch(Exception ex) {

Log.*e*(this.getClass().getSimpleName(), "Получено исключение", ex);

}

}

}

## 5.2 Алгоритм решения задачи

Для проведения каких-либо действий с документами сначала необходимо выбрать раздел: подписать, согласовать, испольнить, ознакомиться или проверить документ.

## 5.3 Описание блок – схемы

Последовательное описание блок – схемы приложения приведено ниже:

Шаг 1. Запуск программы.

Шаг 2. Вопрос: Верно ли указан логин и пароль? Если да, то переходим к следующему шагу, иначе остаемся на 2 шаге.

Шаг 3. Вход в приложение под своей учетной записью.

Шаг 4. Вопрос: Выбрано ли действие? Если да, то переходим к шагу 5, если нет, то переходим к 7 шагу.

Шаг 5. Выполнение

Шаг 6. Возвращаемся к 4 шагу.

Шаг 7. Закрытие приложения.

Блок-схема приведена в приложении А.

Листинг программы приведен в приложении Б.

5.4 Описание интерфейса программы

В активити входа в приложение, представленном на рисунке 5.1, находятся два текстовых поля логин и пароль, кнопка авторизации. Для дальнейшей работы нужно верно указать логин и пароль. Если логин и пароль верны, то откроется основное активити, представленное на рисунке 5.2 с возможностью выбора действия над документами. При нажатии на кнопку происходит открытие нужного нам активити, которая показана на рисунке 5.3.



Рисунок 5.1 – Активити входа

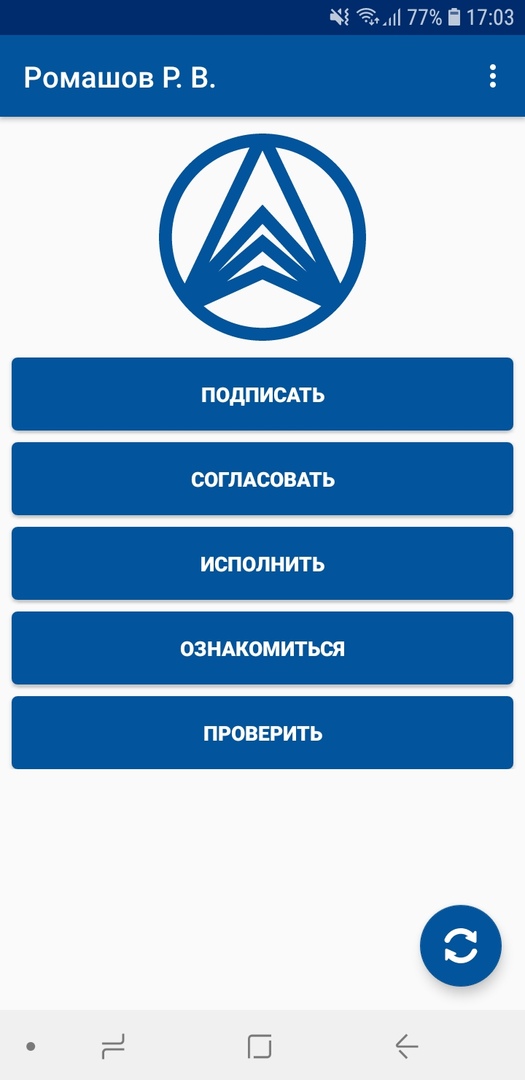


Рисунок 5.2 – Основное активити

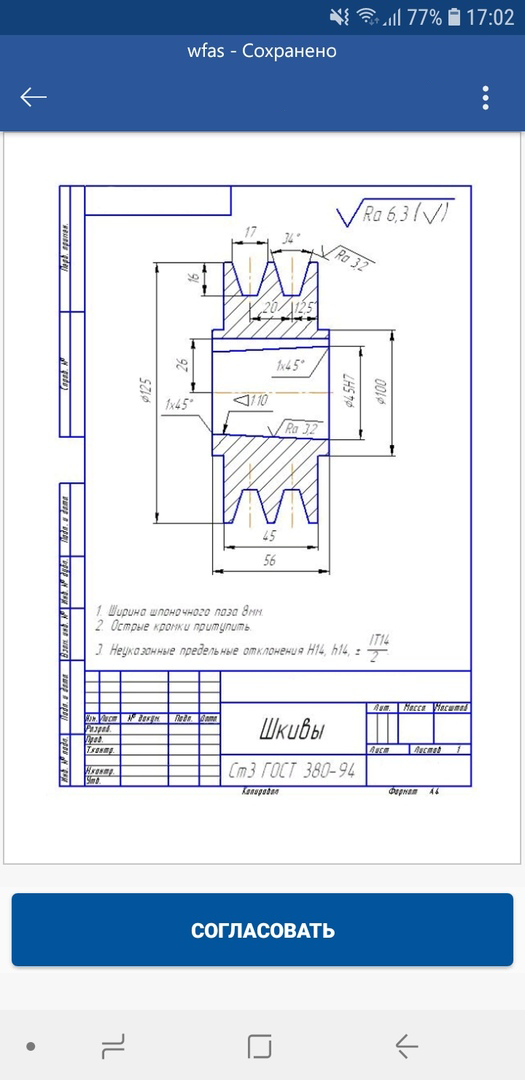


Рисунок 5.3 – Активити согласования

Для успешной работы с программой необходимо:

1. Версия Android 4.0 и выше.
2. Подключение к интернету.
3. Учетная запись.

# Заключение

В процессе работы над программным продуктом автором были выявлены и решены проблемы доступа из основного интерфейса к дополнительным меню и формам программы путем использования оригинального дизайна интерфейса и его эргономичного решения.

В процессе работы над программой была разработана и реализована необходимая система доступа и перехода к основным и дополнительным блокам программного продукта, обеспечивающего точное описание необходимых в работе подразделения ИТ серверных и сетевых механизмов доступа каждого рабочего места сотрудника.

Автором в процессе работы над программным продуктом был произведен необходимый научный и технический анализ для проектирования и реализации программного продукта «Автоматизация продажи билетов».

Цель курсового проекта достигнута. Задачи выполнены.

# Список использованных источников

Интернет ресурсы:

* 1. АО “Атоммашэкспорт”. О компании [Электронный ресурс] – Режим доступа: http://www.atomexp.ru/page/about
  2. Википедия. Язык программирования Java [Электронный ресурс] – Режим доступа: https://ru.wikipedia.org/wiki/Java
  3. Android Developers. Android Studio [Электронный ресурс] – Режим доступа: http://developer.android.com
  4. Habr. Российский блог информационных технологий [Электронный ресурс] – Режим доступа: https://habr.com/ru/

# Приложение А

(обязательное)

Блок – схема программы

Запуск программы

Начало

Введены верные логин и пароль?

Нет

Да

Вход в приложение

Выбрано действие?

Нет

Закрытие

Да

Выполнение

Сохранение документа

Рисунок А.1–Блок схема программы

# Приложение Б

(обязательное)

Листинг программы

create database documents

go

use documents

go

create table employee

(

id int identity primary key,

last\_name nvarchar(20) not null default '',

first\_name nvarchar(20) not null default '',

middle\_name nvarchar(20) not null default '',

position nvarchar(50) not null default '',

phone varchar(15) constraint ck\_users\_phone check (phone like '[8][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),

email varchar(100) constraint ck\_users\_email check (email like '%[@]%'),

[password] varchar(20) not null,

constraint uq\_employee\_phone unique (phone),

constraint uq\_employee\_email unique (email)

)

go

create table document

(

id int identity primary key,

doc\_type nvarchar(50) constraint ck\_document\_doc\_type check (doc\_type IN('Чертеж','Договор','Соглашение','Отчет')),

doc\_path varchar(max) not null,

created\_at date default getdate()

)

go

create table [location]

(

doc\_id int constraint fk\_location\_document foreign key references document(id),

employee\_id int constraint fk\_location\_employee foreign key references employee(id),

[status] nvarchar(50) constraint ck\_location\_status check ([status] IN('Ожидание','В работе','Завершен')),

end\_date\_time datetime null

)

go

package ru.atomexp.android;  
  
import android.content.Intent;  
import android.content.SharedPreferences;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.util.Log;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.ImageButton;  
import android.widget.Toast;  
  
import com.google.gson.Gson;  
import com.google.gson.GsonBuilder;  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.SQLException;  
  
public class LoginActivity extends AppCompatActivity {  
  
 private Button btnSignIn;  
 private ImageButton btnQrCode;  
 private EditText etLogin;  
 private EditText etPass;  
 private User user;  
 private Gson GSON;  
 public static final String *APP\_PREFERENCES* = "authorizedUser";  
 public static final String *SAVED\_USER* = "savedUser";  
 public static SharedPreferences *spUser*;  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_login*);  
 *spUser* = getSharedPreferences(*APP\_PREFERENCES*, *MODE\_PRIVATE*);  
 GSON = new GsonBuilder().create();  
 btnSignIn = findViewById(R.id.*signInButton*);  
 btnQrCode = findViewById(R.id.*qrCodeButton*);  
 etLogin = findViewById(R.id.*loginText*);  
 etPass = findViewById(R.id.*passwordText*);  
 }  
  
 public void signIn(View view) {  
 Toast.*makeText*(this, "Нажатие", Toast.*LENGTH\_SHORT*).show();  
 try {  
 try {  
 Class.*forName*("net.sourceforge.jtds.jdbc.Driver");  
 Toast.*makeText*(this, "Зарегистрирован", Toast.*LENGTH\_LONG*).show();  
 } catch(Exception ex){  
 Toast.*makeText*(this, "Не зарегистрирован" + ex.getMessage(), Toast.*LENGTH\_LONG*).show(); }  
 final String DB\_URL = "jdbc:jtds:sqlserver://192.168.31.80:1433/documents";  
 final String DB\_USER = "p1onerw0w";  
 final String DB\_PASSWORD = "4321193";  
 final Gson GSON = new GsonBuilder().create();  
 String login = etLogin.getText().toString();  
 String password = etPass.getText().toString();  
 String query =  
 "select \* from users " +  
 "where (phone = '" + login + "' or email = '" + login + "') " +  
 "and [password] = '" + password + "'";  
 try {  
 Connection connection = DriverManager.*getConnection*(DB\_URL, DB\_USER, DB\_PASSWORD);  
 Toast.*makeText*(this, "Подключение создано", Toast.*LENGTH\_LONG*).show();  
 } catch (Exception ex) {  
 Toast.*makeText*(this, "Не подключен" + ex.getLocalizedMessage(), Toast.*LENGTH\_LONG*).show();  
 }  
 if(!*spUser*.contains(*SAVED\_USER*)) {  
 try {  
 user = User.*userBuild*(SqlServerService.*getResultSet*(query));  
 String gsonUser = GSON.toJson(user);  
  
 SharedPreferences.Editor editor = *spUser*.edit();  
 editor.putString(*SAVED\_USER*, gsonUser);  
 editor.apply();  
 Intent intent = new Intent(this, MainActivity.class);  
 intent.putExtra("user", gsonUser);  
 Toast.*makeText*(this, "Авторизация", Toast.*LENGTH\_SHORT*).show();  
 startActivity(intent);  
 } catch(SQLException sqlex) {  
 Toast.*makeText*(this, "Ошибка SQL", Toast.*LENGTH\_LONG*).show(); } catch(Exception ex) {  
 Toast.*makeText*(this, "Иная ошибка", Toast.*LENGTH\_LONG*).show();  
 }  
 }   
 } catch(Exception ex) {  
 Toast.*makeText*(this, "Снаружи " + ex.getMessage(), Toast.*LENGTH\_LONG*).show();  
 Log.*e*(this.getClass().getSimpleName(), "Получено исключение", ex);  
 }  
 }  
}

package ru.atomexp.android;  
  
import android.os.Bundle;  
import android.support.design.widget.NavigationView;  
import android.support.v4.view.GravityCompat;  
import android.support.v4.widget.DrawerLayout;  
import android.support.v7.app.ActionBarDrawerToggle;  
import android.support.v7.app.AppCompatActivity;  
import android.support.v7.widget.Toolbar;  
import android.view.Menu;  
import android.view.MenuItem;  
import android.widget.TextView;  
import android.widget.Toast;  
import com.google.gson.Gson;  
import com.google.gson.GsonBuilder;  
import com.google.gson.JsonObject;  
import org.json.JSONObject;  
  
public class MainActivity extends AppCompatActivity  
 implements NavigationView.OnNavigationItemSelectedListener {  
 private TextView navHeaderTitle;  
 private TextView navHeaderSubTitle;  
 private User thisUser;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 Toolbar toolbar = findViewById(R.id.*toolbar*);  
 setSupportActionBar(toolbar);  
 DrawerLayout drawer = findViewById(R.id.*drawer\_layout*);  
 ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(  
 this, drawer, toolbar, R.string.*navigation\_drawer\_open*, R.string.*navigation\_drawer\_close*);  
 drawer.addDrawerListener(toggle);  
 toggle.syncState();  
 NavigationView navigationView = findViewById(R.id.*nav\_view*);  
 navigationView.setNavigationItemSelectedListener(this);  
 if(savedInstanceState == null) { navigationView.setCheckedItem(R.id.*m\_messages*);  
 getSupportActionBar().setTitle(R.string.*messages*);  
 }  
 navHeaderTitle = findViewById(R.id.*headerTitle*);  
 navHeaderSubTitle = findViewById(R.id.*headerSubTitle*);  
 Gson GSON = new GsonBuilder().create();  
 thisUser = GSON.fromJson((String)getIntent().getSerializableExtra("user"), User.class);  
 String headerTitle = thisUser.getLastName() + thisUser.getFirstName();  
 navHeaderTitle.setText(headerTitle);  
 String headerSubTitle = thisUser.getPosition();  
 navHeaderSubTitle.setText(headerSubTitle);  
 }  
  
 @Override  
 public void onBackPressed() {  
 DrawerLayout drawer = findViewById(R.id.*drawer\_layout*);  
 if (drawer.isDrawerOpen(GravityCompat.*START*)) {  
 drawer.closeDrawer(GravityCompat.*START*);  
 } else {  
 *super.onBackPressed();* }  
 }  
  
 @Override  
 public boolean onCreateOptionsMenu(Menu menu) { getMenuInflater().inflate(R.menu.*main*, menu);  
 return true;  
 }  
  
 @Override  
 public boolean onOptionsItemSelected(MenuItem item) { int id = item.getItemId(); if (id == R.id.*action\_settings*) {  
 Toast.*makeText*(MainActivity.this, "Settings", Toast.*LENGTH\_LONG*).show();  
 }  
 return super.onOptionsItemSelected(item);  
 }