|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Министерство образования Московской области  Государственное образовательное учреждение высшего образования Московской области  «Государственный гуманитарно-технологический университет»  Промышленно-экономический колледж | | | | | |
| **КУРСОВОЙ ПРОЕКТ** | | | | | |
| **РАЗРАБОТКА СИСТЕМЫ ДЛЯ АВТОМАТИЗАЦИИ ОБРАБОТКИ БИЛЕТОВ ОНЛАЙН-КИНОТЕАТРА** | | | | | |
| дисциплина МДК.01.01. Разработка программных модулей | | | | | |
|  | **Выполнил:**  **Игнатенко Евгений Павлович**  студента группы ИСП.22.2А  по специальности  09.02.07 Информационные системы и программирование  очной формы обучения | | | | |
| **Руководитель:**  **Климов Александр Андреевич** | | | | |
| Оценка | | | | |
|  |  | / |  | / |
|  |  |  | Подпись руководителя | |
| Орехово-Зуево  2025 | | | | | |

Изм.

Лист

№ докум.

Подпись

Дата

Лист

2

КП.09.02.07.ИСП.22.2А.П3

Разраб.

Игнатенко Е.П.

Провер.

Климов А.А.

Реценз.

Н. Контр.

Утверд.

Грачева Н.М.

Разработка системы автоматизации обработки билетов онлайн-кинотеатра

Лит.

Листов

43

ПЭК ГГТУ

# Содержание

[Содержание 2](#_Toc193376402)

[Введение 3](#_Toc193376403)

[Глава 1. Теоретическая глава 6](#_Toc193376404)

[1.1 Общие сведения 6](#_Toc193376405)

[1.2 Основы информационных систем 7](#_Toc193376406)

[1.3 Выбор СУБД 8](#_Toc193376407)

[1.4 Выбор языка программирования 14](#_Toc193376408)

[1.5 Среда разработки 20](#_Toc193376409)

[1.7 Стадии и этапы разработки 26](#_Toc193376410)

[Глава 2. Практическая часть 28](#_Toc193376411)

[2.1 Проектирование и нормализация базы данных 28](#_Toc193376412)

[2.2 Разработка информационной системы онлайн-кинотеатра и описание пользовательского интерфейса 31](#_Toc193376413)

[2.3 Тестирование созданного приложения 37](#_Toc193376414)

# Введение

Современный мир информационных технологий невозможно представить без автоматизации различных процессов, связанных с обработкой данных. В условиях роста объемов информации традиционные методы ее обработки становятся недостаточно эффективными.

Одним из наиболее востребованных направлений является индустрия онлайн-кинотеатров, предоставляющих пользователям возможность просмотра фильмов, сериалов и другого видеоконтента в высоком качестве.

Благодаря таким сервисам зрители могут наслаждаться новинками киноиндустрии без необходимости посещать кинотеатры, что особенно актуально в условиях развития стриминговых технологий и изменяющихся потребительских привычек.

В настоящее время онлайн-кинотеатры становятся все более популярными, предоставляя пользователям удобный доступ к фильмам и сериалам. Однако процесс обработки заказов, учета билетов и взаимодействия с пользователями требует значительных временных затрат и подвержен человеческому фактору. Автоматизация этих процессов позволит значительно повысить эффективность работы системы, улучшить качество обслуживания клиентов и минимизировать ошибки, связанные с обработкой билетов.

Актуальность разработки моей системы обусловлена растущей потребностью в удобных и эффективных инструментах управления билетами в онлайн-кинотеатрах. Разработка системы для автоматизированной обработки билетов обеспечит удобное приложение, учета и контроля билетов.

Целью курсовой работы является разработка системы для автоматизации обработки билетов онлайн-кинотеатра, которая позволит упростить процесс бронирования, и учета билетов.

Объект исследования – система автоматизации обработки билетов онлайн-кинотеатра.

Предмет исследования – процесс проектирования и создания данной системы.

Методы исследования: теоретические, эмпирические и статистические.

Задачи курсовой работы:

1. Определить понятие информационной системы, ее принципы и основные компоненты.
2. Определить область применения создаваемой системы.
3. Провести обзор существующих сред разработки и баз данных.
4. Разработать структуру базы данных для хранения информации о билетах и пользователях.
5. Создать приложение с удобным графическим интерфейсом для пользователей.
6. Провести тестирование системы и оценить ее работоспособность.

В данном курсовом проекте использованы различные литературные источники и интернет-ресурсы, содержащие актуальную информацию по теме разработки системы для автоматизации обработки билетов онлайн-кинотеатра.

Основными источниками стали книги по программированию, а также статьи из сайтов, посвящённых информационным технологиям. Эти материалы помогли сформировать теоретическую базу и определить наиболее эффективные подходы к реализации проекта.

Данный курсовой проект имеет следующую структуру:

1. Введение, в котором обоснована актуальность темы, сформулированы цель и задачи исследования.
2. Теоретическая часть, где рассматриваются основные аспекты разработки информационной системы, анализируются технологии, которые могут быть использованы при создании системы, а также формулируются требования к разрабатываемому программному продукту.
3. Практическая часть, содержащая описание этапов проектирования системы, разработку базы данных, описание функционала приложения и реализацию пользовательского интерфейса.
4. Заключение, в котором подводится итог данного курсового проекта.

Практическая значимость работы заключается в автоматизация обработки билетов онлайн-кинотеатра, которое позволит значительно сократить затраты времени на бронирование и учет билетов, уменьшить количество ошибок, повысить удобство для пользователей и улучшить эффективность работы кинотеатра.

# Глава 1. Теоретическая глава

# 1.1 Общие сведения

Современные онлайн-кинотеатры предоставляют пользователям удобный доступ к просмотру фильмов и сериалов. Однако, с увеличением количества пользователей и просмотров возникает необходимость автоматизации процессов обработки билетов, бронирования мест и управления доступом к контенту. Ручное управление этими процессами не только требует значительных временных затрат, но и увеличивает вероятность ошибок.

Для оптимизации работы онлайн-кинотеатра разрабатывается система, позволяющая автоматизировать обработку билетов.

Основная задача системы - упрощение процесса бронирования билетов, управления заказами. Система должна иметь понятный и простой интерфейс, а пользователи — инструмент для просмотра и управления бронированиями билетов.

Основанием для разработки технического задания является задание к курсовой работе по теме «Разработка системы автоматизации обработки билетов онлайн-кинотеатра».

Наименование темы разработки - «Разработка информационной системы для пользователей онлайн-кинотеатра».

Функциональные требования

1. Просмотр списка фильмов и расписания сеансов.
2. Выбор и бронирование билетов.
3. Получение электронного билета (номер заказа).
4. Отмена брони билета.
5. Просмотр истории бронирований.
6. Окно авторизации при входе.

Система должна соответствовать нефункциональным требованиям:

1. Удобство использования – интуитивный интерфейс, доступный для широкой аудитории.
2. Безопасность – защита персональных данных.
3. Производительность – быстрая обработка запросов и минимальное время ожидания.

# 1.2 Основы информационных систем

Современные информационные системы играют ключевую роль в автоматизации процессов обработки и управления данными. Основной технической основой таких систем является персональный компьютер, серверное оборудование и специализированное программное обеспечение. Однако, несмотря на важность технической составляющей, информационная система не может функционировать без пользователей, которые обрабатывают, анализируют и применяют полученные данные.

Существует важное различие между компьютером и информационной системой. Компьютеры, оснащенные специализированными программными средствами, являются инструментами для работы с информацией, в то время как информационная система представляет собой комплексное решение, включающее технические средства, программное обеспечение, пользователей и обработки данных.

Информационная система обладает следующими основными свойствами:

1. Любая информационная система может быть проанализирована, спроектирована и управляемая на основе общих принципов построения систем.
2. Информационные системы являются динамичными и развивающимися, так как их структура и функционал могут адаптироваться к изменениям в предметной области.
3. При проектировании информационных систем необходимо использовать системный подход, включающий анализ всех элементов и их взаимодействия.
4. Основной продукцией информационной системы является информация, на основе которой принимается информация и операционные решения.

Процессы, обеспечивающие работу информационной системы любого назначения, условно можно представить в виде схемы , состоящей из блоков:

1. Ввод информации из внешних или внутренних источников (например, данные о бронировании билетов, информация о пользователях).
2. Обработка данных с преобразованием входной информации в удобный формат (учет уже забронированных билетов, анализ загруженности кинотеатра).
3. Вывод информации для представления пользователям или передачи в другие системы (отображение доступных сеансов).
4. Обратная связь, позволяющая корректировать и уточнять вводимые данные (например, изменение брони, возврат билетов).

Работа информационных систем заключается в обслуживании двух встречных потоков информации: ввода новой информации и выдачи актуальных данных по запросам пользователей. Так как основная цель информационной системы — эффективное обслуживание клиентов, она должна обеспечивать быстрый и точный отклик на любые запросы. Это достигается за счет применения алгоритмов поиска информации и организации данных.

# 1.3 Выбор СУБД

При разработке программного обеспечения важным этапом является выбор системы управления базами данных (СУБД), так как от него зависят производительность, масштабируемость и надежность будущей системы. Различные СУБД обладают своими преимуществами и особенностями, что делает процесс выбора критически важным для обеспечения оптимальной работы приложения [4].

В данном разделе рассматриваются ключевые характеристики и возможности популярных реляционных СУБД, таких как MySQL, Oracle Database и SQL Server, с целью определения наиболее подходящего решения для реализации проекта. К ним относятся:

1. **SQL Server Management Studio (SSMS)** — интегрированная среда для управления инфраструктурой SQL Server.

Данная среда предоставляет широкий набор инструментов для администрирования, разработки и мониторинга баз данных, а также поддерживает интеграцию с облачными сервисами Microsoft Azure.

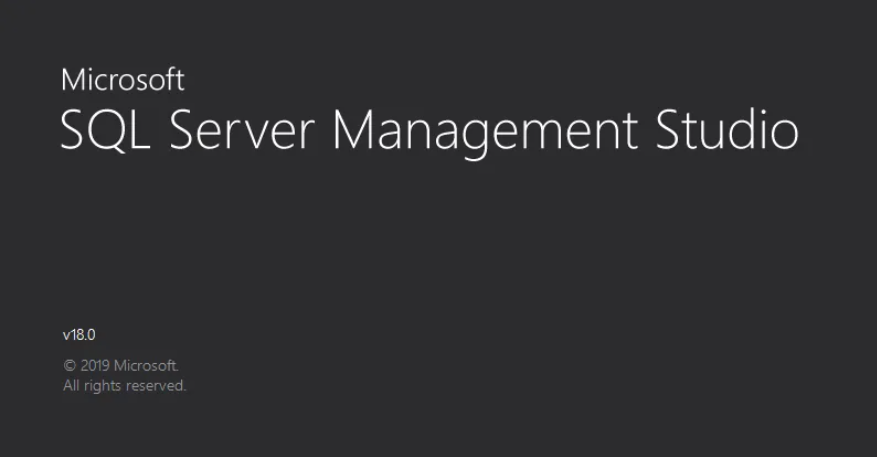


Рисунок 1 - Окно входа вSSMS

SSMS используется для доступа, настройки и управления следующими компонентами:

1. SQL Server,
2. Azure SQL Database,
3. SQL Server на виртуальных машинах Azure,
4. Azure Synapse Analytics.

Кроме того, SSMS предлагает развитые возможности для работы с Analysis Services, включая резервное копирование, обработку данных и разработку многомерных выражений (MDX), интеллектуального анализа данных (DMX) и XMLA. В рамках интеграции с Integration Services пользователи могут организовывать пакеты, выполнять их импорт и экспорт, а также контролировать выполнение заданий.

Основные нововведения SQL Server Management Studio 2022

Последняя версия SQL Server 2022 ориентирована на взаимодействие с облачными технологиями Microsoft, что значительно расширяет возможности гибридного хранения и управления данными. В числе ключевых обновлений:

* 1. Интеграция с облачными сервисами Microsoft Azure, что позволяет обеспечивать высокую доступность баз данных, их аварийное восстановление и бесшовную миграцию в облако.
  2. Реализация реестра базы данных SQL Azure, обеспечивающего контроль целостности данных с использованием технологии блокчейн.
  3. Использование Azure Synapse Link для SQL Server, что позволяет автоматизировать передачу изменений в базе данных в аналитические сервисы Azure Synapse Analytics.

Кроме того, интеграция с Azure Preview расширяет возможности администрирования и безопасности, включая:

1. автоматическое сканирование локальных серверов для сбора метаданных;
2. классификацию данных с применением инструментов Microsoft Information Protection;
3. гибкую настройку и контроль прав доступа к SQL Server.
4. **Oracle Database**

Oracle Database – это объектно-реляционная система управления базами данных (СУБД), поддерживающая ряд технологий, реализующих объектно-ориентированный подход. Данная СУБД обеспечивает эффективное управление созданием, хранением и использованием данных, что делает ее одним из ведущих решений в сфере корпоративных баз данных.



Рисунок 2 - Логотип Oracle Database

Oracle Database обладает широким функционалом, направленным на повышение производительности, отказоустойчивости и удобства администрирования:

1. Real Application Cluster (RAC) – технология, обеспечивающая работу одного экземпляра базы данных на нескольких узлах в составе Grid-инфраструктуры. Это позволяет динамически управлять нагрузкой и масштабировать систему при необходимости.
2. Automatic Storage Management (ASM) – механизм автоматического распределения данных между ресурсами хранения, что повышает отказоустойчивость системы и снижает совокупную стоимость владения (TCO).
3. Высокая производительность – Oracle Database позволяет управлять уровнями сервиса и тиражировать эталонные конфигурации в рамках сети, обеспечивая стабильную и быструю работу.
4. Инструменты разработки – встроенный инструмент HTML DB упрощает процесс создания приложений для работы с базами данных, позволяя пользователям без глубоких технических знаний разрабатывать эффективные решения.
5. Самоуправляемость – автоматические механизмы оптимизации позволяют перераспределять нагрузку, корректировать SQL-запросы, а также выявлять и прогнозировать возможные ошибки.
6. Поддержка больших баз данных – максимальный размер экземпляра базы данных в Oracle может достигать 8 экзабайт, что делает платформу оптимальным выбором для масштабных корпоративных решений.
7. Поддержка распределенных баз данных – реализована возможность использования переносимых табличных пространств, системы Oracle Streams для управления потоками данных, а также модели распределенных SQL-запросов.
8. **MySQL**

MySQL — это система управления базами данных (СУБД) с открытым исходным кодом, разработанная шведской компанией MySQL AB. Она распространяется по лицензии GNU General Public License (GPL), а также по коммерческим лицензиям для использования в проектах, не подпадающих под действие GPL.

MySQL является реляционной СУБД и полностью поддерживает структурированный язык запросов SQL, начиная с версии 5.0. Благодаря этому MySQL совместима с другими реляционными базами данных и широко используется в различных программных решениях.



Рисунок 3 - Логотип MySQL

Преимущества MySQL

MySQL обладает рядом особенностей, которые делают ее одной из самых популярных СУБД:

1. Высокая производительность и стабильность — MySQL считается одной из самых быстрых реляционных СУБД, наряду с Oracle.
2. Открытый исходный код — пользователи могут изучать, модифицировать и улучшать систему, что способствует ее постоянному развитию [2].
3. Кроссплатформенность — MySQL протестирована и поддерживается на множестве операционных систем, включая Windows, Linux, FreeBSD, Mac OS X, OS/2, Solaris и другие.
4. Гибкость в интеграции — поддержка API для C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, Tcl позволяет интегрировать MySQL в широкий спектр программных решений.
5. Широкий выбор типов таблиц — MySQL поддерживает как встроенные механизмы хранения данных, так и решения сторонних разработчиков, что позволяет адаптировать систему под конкретные задачи.
6. Система обеспечивает надежную обработку данных на различных языках, включая русский.

MySQL является одной из наиболее популярных и универсальных СУБД, широко используемых в веб-разработке, корпоративных системах и облачных платформах. Ее высокая производительность, гибкость и поддержка современных технологий делают MySQL оптимальным выбором для множества проектов, требующих надежного и эффективного управления данными.

В качестве системы управления базами данных для моего курсового проекта была выбрана SQL Server Management Studio (SSMS). Данный выбор обусловлен тем, что я изучал эту СУБД на втором курсе, и за время обучения приобрел необходимые навыки работы с ней.

# 1.4 Выбор языка программирования

При разработке программного обеспечения выбор языка программирования играет ключевую роль, так как он влияет на скорость разработки, производительность системы, удобство работы и интеграцию с другими технологиями.

Современные языки программирования обладают разными возможностями и предназначены для решения различных задач. В данном разделе рассматриваются четыре популярных языка: Python, C++, C# и Java, их особенности, преимущества и области применения, чтобы определить наиболее подходящий вариант для реализации проекта.

1. **Python**

Python — это высокоуровневый язык программирования с открытым исходным кодом, который отличается простой и читаемой синтаксической структурой. Благодаря своей гибкости и большому количеству библиотек Python стал популярным выбором для разработки веб-приложений, научных вычислений, автоматизации задач и работы с большими данными. Язык активно используется в области анализа данных, машинного обучения и разработки различных программных решений [3].



Рисунок 4 - Логотип Python

Одним из значительных преимуществ Python является поддержка различных систем управления базами данных (СУБД), таких как MySQL, PostgreSQL, SQLite и SSMS. Он предоставляет многочисленные библиотеки и фреймворки, такие как SQLAlchemy, Django ORM, и Pandas, которые позволяют легко взаимодействовать с базами данных, выполнять запросы и обрабатывать данные.

Кроме того, Python предлагает поддержку для создания серверных приложений, включая интеграцию с веб-серверами, что делает его удобным для разработки как на стороне клиента, так и на сервере. Это делает Python отличным выбором для реализации программных решений, требующих взаимодействия с базами данных в реальном времени, таких как системы для автоматизации обработки билетов онлайн-кинотеатра.

Для разработки системы автоматизированной обработки билетов онлайн-кинотеатра был выбран язык программирования Python, так как он удобен, прост в освоении и уже изучался мной ранее. Кроме того, Python поддерживает автоматизацию процессов, что делает его оптимальным выбором для данного проекта.

* 1. **C++**

C++ — это компилируемый, статически типизированный язык программирования общего назначения, который предоставляет разработчику огромные возможности для создания программ различной сложности. C++ поддерживает несколько стилей программирования, включая объектно-ориентированное, процедурное и общее программирование, что дает гибкость в решении различных задач. Хотя часто его ассоциируют с объектно-ориентированным программированием, C++ позволяет применять и другие парадигмы, не ограничивая разработчиков только одной концепцией.



Рисунок 5 - Логотип C++

C++ унаследовал синтаксис от языка C, сохраняя совместимость с ним. Однако со временем различия между этими языками становятся все более заметными, так как каждый из них развивается независимо, и стандарты для них разрабатываются разными комитетами. Это влечет за собой изменения в структуре и возможностях каждого языка.

Одним из основных достоинств C++ является его производительность. Благодаря низкоуровневому доступу к памяти, C++ позволяет разработчикам создавать программы с высокой скоростью исполнения, что делает его идеальным для разработки системных программ, игр и приложений с высокими требованиями к вычислительным ресурсам.

Стандартная библиотека C++ представляет собой мощный набор инструментов, включающий такие компоненты, как контейнеры (например, вектор, список, карта), алгоритмы сортировки и поиска, работу с регулярными выражениями и многое другое. Включение в стандартную библиотеку поддержки многопоточности позволяет разрабатывать высокоэффективные многозадачные приложения. Это особенно полезно в приложениях, требующих параллельных вычислений или работы с большими объемами данных.

* 1. **Java**

Java — это язык программирования общего назначения, который относится к объектно-ориентированным языкам с сильной типизацией. Он был разработан с акцентом на создание универсальных приложений, которые могут работать на разных платформах. Один из принципов, реализованных в Java, называется WORA (Write Once, Run Anywhere) — «пиши один раз, запускай везде». Это означает, что код, написанный на Java, можно выполнить на любой платформе, на которой установлена среда исполнения Java (JRE, Java Runtime Environment), что делает Java идеальным выбором для разработки кроссплатформенных приложений.



Рисунок 6 - Логотип Java

Этот принцип достигается через компиляцию исходного кода в байт-код промежуточный код, который затем исполняется с помощью виртуальной машины Java (JVM). Виртуальная машина не зависит от операционной системы или аппаратной платформы, что позволяет запускать одно и то же приложение на Windows, Linux, macOS и других системах без изменения исходного кода.

Одним из важнейших преимуществ Java является её встроенная система управления памятью, основанная на механизме сборщика мусора (garbage collector). В Java разработчик создает объекты, и когда они больше не используются, сборщик мусора автоматически очищает память, освобождая ресурсы.

Это снижает риск утечек памяти, поскольку разработчик не обязан вручную управлять удалением объектов, как это происходит в некоторых других языках. Такой подход значительно упрощает разработку, делая систему более стабильной и уменьшает вероятность ошибок, связанных с управлением памятью.

Java также известна своей безопасностью и масштабируемостью, что делает её популярным выбором для создания крупных распределённых систем, финансовых приложений и облачных сервисов.

* 1. **C#**

C# на сегодняшний день является одним из самых мощных, быстро развивающихся и востребованных языков программирования в IT-отрасли. Этот язык используется для создания широкого спектра приложений от небольших десктопных программ до крупных веб-порталов и сервисов, которые ежедневно обслуживают миллионы пользователей.



Рисунок 7 - Логотип C#

Язык C# обладает многолетней историей развития и совершенствования. Первая версия языка была выпущена в феврале 2002 года вместе с релизом Microsoft Visual Studio .NET, а последняя версия — C# 10.0 — появилась 8 ноября 2021 года вместе с релизом .NET 6.

C# является языком, заимствовавшим основные принципы и механизмы от таких языков, как Java и C++.

Одним из ключевых преимуществ C# является его интеграция с платформой .NET, которая представляет собой мощную экосистему для разработки приложений. Среди главных особенностей C# и платформы .NET можно выделить:

Кроссплатформенность. .NET является переносимой платформой, поддерживающей большинство современных операционных систем, включая Windows, macOS, Linux, а также мобильные платформы Android, iOS, Tizen. Это дает возможность разрабатывать приложения, которые могут работать на различных устройствах и операционных системах с минимальными усилиями.

Большая библиотека классов. .NET предоставляет единую для всех поддерживаемых языков библиотеку классов, которая значительно ускоряет разработку приложений. Независимо от того, разрабатывается ли текстовый редактор, чат или сложный веб-сайт, разработчик всегда может использовать богатый функционал библиотеки классов .NET.

Разнообразие технологий. Базовая библиотека классов и CLR служат основой для целого стека технологий, включая ADO.NET и Entity Framework Core для работы с базами данных, WPF и WinUI для создания графических приложений с насыщенным интерфейсом, Windows Forms для более простых графических приложений, а также Blazor, который позволяет создавать современные веб-приложения, работающие как на серверной, так и на клиентской стороне. В будущем Blazor будет поддерживать создание мобильных и десктопных приложений.

Высокая производительность. Согласно множеству тестов, веб-приложения на .NET 6 значительно опережают аналогичные решения, разработанные с использованием других технологий. Приложения на платформе .NET известны своей высокой производительностью и эффективностью.

# Среда разработки

Среда разработки — играет ключевую роль в процессе создания программного обеспечения, предоставляя разработчикам удобные инструменты для написания, отладки и тестирования кода.

Современные программы предлагают широкий спектр функционала, который может значительно ускорить процесс разработки и повысить его эффективность. Рассмотрим три популярных и мощных инструмента для разработки приложений:

1. **Visual Studio Code**

Visual Studio Code (VS Code) — это редактор, разработанный Microsoft для Windows, Linux и macOS. Он ориентирован на кроссплатформенную разработку веб- и облачных приложений, предоставляя пользователям удобные инструменты для работы с кодом. VS Code включает в себя встроенный отладчик, средства работы с Git, подсветку синтаксиса, IntelliSense (автодополнение кода) и функции рефакторинга. В редакторе предусмотрены возможности для кастомизации, такие как настройка тем, сочетаний клавиш и конфигурационных файлов.

Программа распространяется бесплатно и разрабатывается как ПО с открытым исходным кодом, но собранные версии распространяются c лицензией. VS Code основан на Electron, и его функционал реализуется через веб-редактор Monaco, который был изначально разработан для Visual Studio Online.

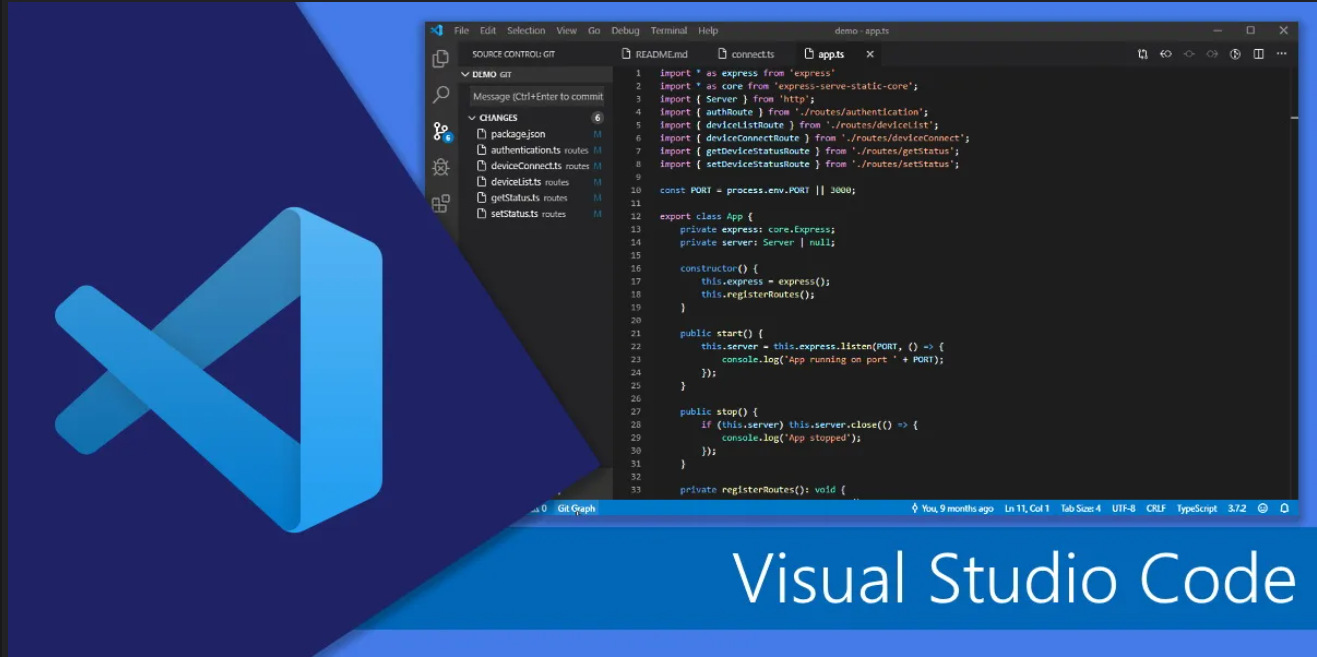


Рисунок 8 - Пример окна Visual Studio Code

Особенности Visual Studio Code:

1. Поддержка множества языков программирования (C++, C#, Python, JavaScript и другие).
2. Наличие встроенного отладчика, инструментов для работы с Git, а также средств для рефакторинга и навигации по коду.
3. Поддержка разработки как консольных приложений, так и приложений с графическим интерфейсом, в том числе для платформ ASP.NET и Node.js.
4. Легкость и гибкость в работе: благодаря возможности использования палитры команд и JSON-файлов для настройки, разработчик может кастомизировать среду под свои потребности.
5. Расширения для работы с Docker, Git, линтерами и темами, что значительно расширяет функциональность редактора.
6. **JetBrains Rider**

JetBrains Rider — это платная кроссплатформенная среда разработки, предназначенная для работы с приложениями на платформе .NET. Разрабатываемая компанией JetBrains, Rider поддерживает языки C#, VB.NET и F#, а также работает с такими платформами, как .NET Framework, .NET Core и Mono. Среда поддерживает работу на операционных системах Windows, macOS и Linux. Rider построен на базе популярного инструмента ReSharper, предоставляя разработчикам удобные и эффективные средства для создания высококачественного кода.

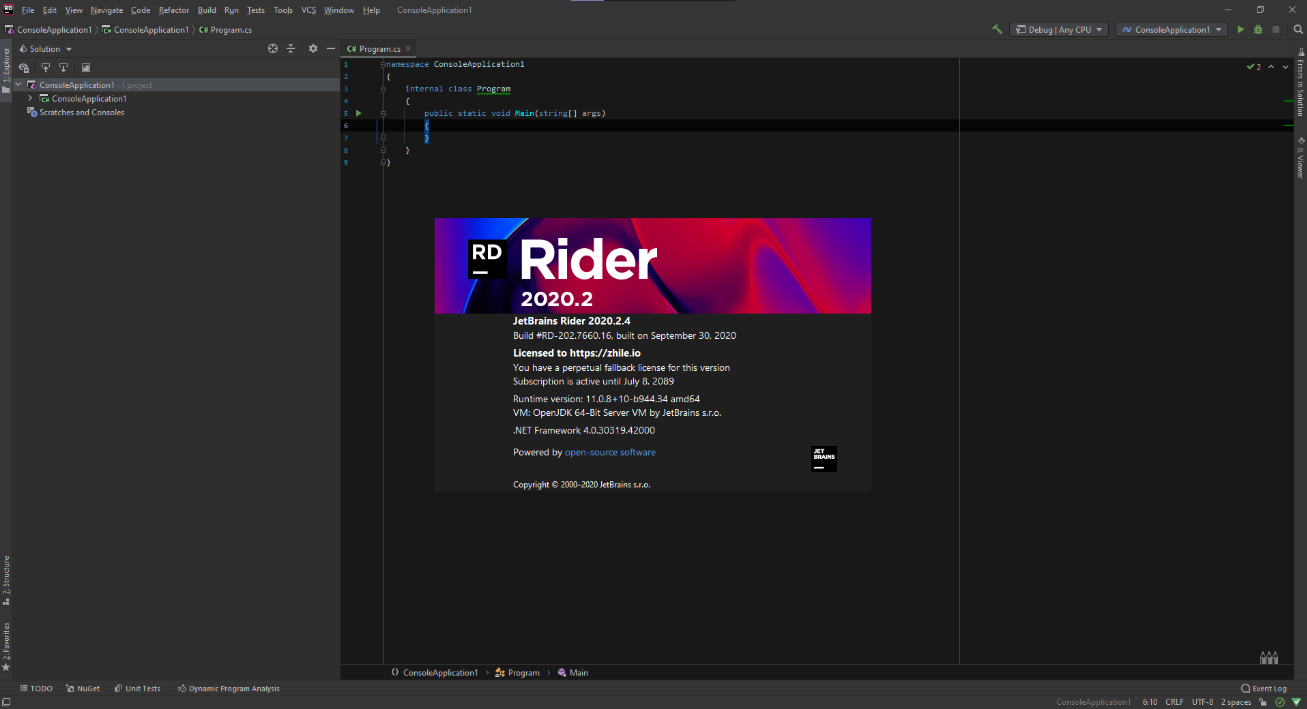


Рисунок 9 - Пример окна JetBrains Rider

Преимущества JetBrains Rider:

1. Интеллектуальный редактор кода: Rider предлагает множество вариантов автодополнения, шаблонов для типовых конструкций, постфиксные шаблоны, ссылки на контроллеры и действия в ASP.NET MVC, а также возможность редактировать код в нескольких местах одновременно. Разработчики могут быстро перемещаться по иерархии наследования и эффективно работать с кодом с помощью интуитивно понятных иконок.

2. Глубокий анализ кода: Rider проводит более 2200 инспекций кода для выявления ошибок и проблем в его структуре, а также предоставляет более 1000 автоматических исправлений, которые можно применять на лету. Встроенный инструмент анализа ошибок позволяет отслеживать проблемы по всему решению, повышая качество и стабильность кода.

3. Рефакторинг: Rider поддерживает более 60 рефакторингов, включая переименование, извлечение методов и классов, а также перемещение и копирование типов. Программный продукт предлагает более 450 контекстных действий для быстрого изменения и оптимизации кода.

4. Юнит-тестирование: Среда поддерживает модульные тесты для различных платформ, включая NUnit, xUnit.net и MSTest. Все тесты отображаются прямо в редакторе с соответствующими иконками, а управление тестами, запуск и отладка становятся простыми и интуитивно понятными.

5. Отладка: Rider включает в себя мощный отладчик для приложений на .NET Framework, Mono и .NET Core. Он поддерживает установку точек останова, пошаговое выполнение кода, а также позволяет детально контролировать выполнение программы с возможностью выхода и обхода кода.

6. Работа с базами данных и SQL: Rider позволяет разработчикам подключаться к базам данных, редактировать схемы и таблицы, выполнять SQL-запросы и анализировать базы данных с помощью UML-диаграмм. Встроенный редактор SQL предлагает подсветку синтаксиса, автодополнение и инструменты для удобной навигации по кодовой базе.

1. **Microsoft Visual Studio 2022**

Microsoft Visual Studio 2022 — это линейка продуктов компании Майкрософт, которая предоставляет все необходимые инструменты для создания современных приложений для различных платформ.

Включает как консольные и графические приложения, так и веб-сайты, веб-приложения и веб-службы, поддерживая широкий спектр технологий, таких как Windows Forms, .NET Framework и Microsoft Silverlight. Visual Studio идеально подходит для разработки приложений для экосистемы Microsoft и поддерживает работу на платформах Windows, Windows Mobile, Windows CE и других.

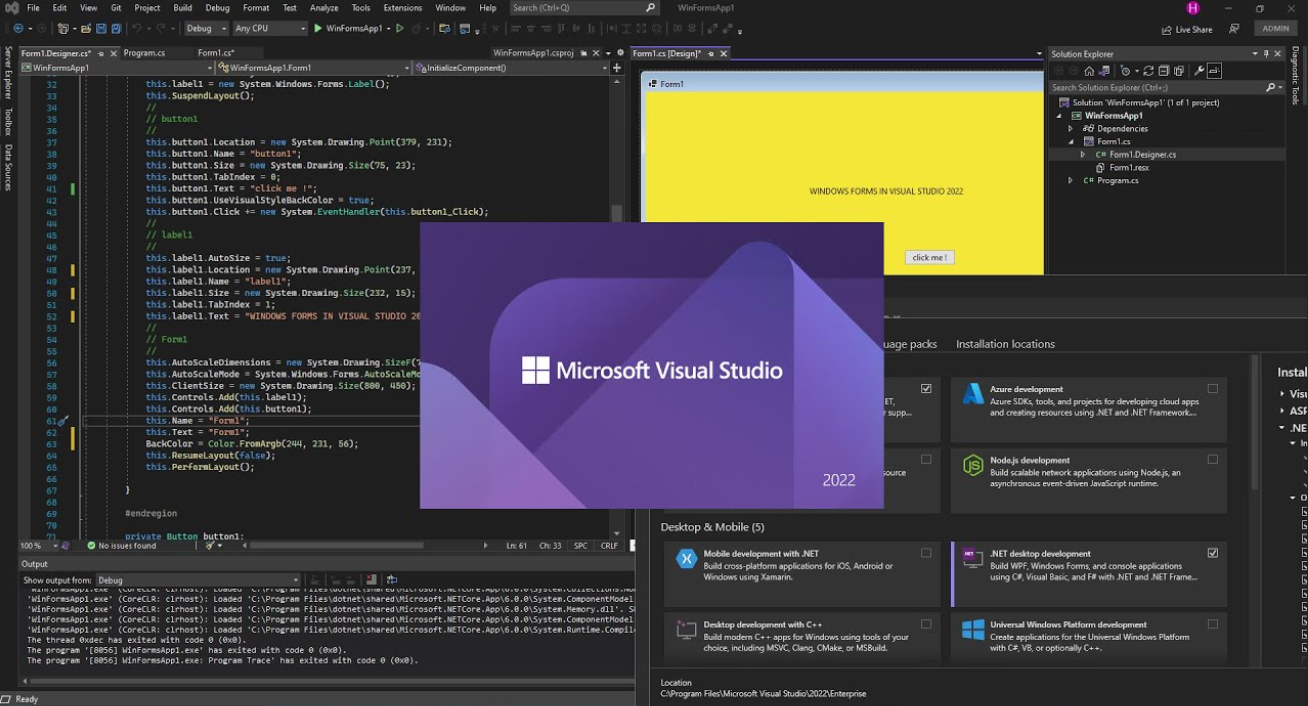


Рисунок 10 - Пример окна Microsoft Visual Studio 2022

Основные преимущества Microsoft Visual Studio:

1. Интеллектуальная поддержка кода: Visual Studio обладает встроенной поддержкой синтаксиса для различных языков программирования, что помогает избежать ошибок, таких как забытые точки с запятой или скобки. Функция IntelliSense автоматически подставляет часто используемые фрагменты кода, ускоряя процесс разработки и снижая вероятность ошибок.
2. Поддержка рефакторинга и отладки: Встроенный отладчик позволяет работать как на уровне исходного кода, так и на машинном уровне. Он помогает разработчикам анализировать и отлаживать код, обеспечивая высокую точность и эффективность при поиске ошибок. Возможность простого рефакторинга кода упрощает процесс его улучшения и поддержания.
3. Поддержка сторонних плагинов: Visual Studio позволяет интегрировать сторонние дополнения и плагины, что значительно расширяет функциональность IDE. Это дает разработчикам возможность добавить поддержку систем контроля версий, таких как Subversion и Visual SourceSafe, а также использовать дополнительные наборы инструментов, например, для визуального проектирования и редактирования кода.
4. Полная совместимость с SQL Server: Visual Studio идеально интегрируется с SQL Server, обеспечивая поддержку для работы с базами данных и их схемами. Это позволяет создавать приложения, которые эффективно взаимодействуют с базами данных, и делать разработку более удобной и быстрой.
5. Мощная поддержка Git: Visual Studio имеет встроенную поддержку Git, что позволяет разработчикам легко загружать и управлять исходным кодом, работать с версиями и отслеживать изменения прямо в процессе разработки.
6. **PyCharm**

PyCharm — это бесплатная программа для разработки, на языке программирования Python, разработанная компанией JetBrains. Она предоставляет удобные инструменты для написания, отладки и тестирования кода, что делает процесс разработки эффективным и комфортным. PyCharm поддерживает различные платформы, включая Windows, macOS и Linux, что позволяет разработчикам работать в любой удобной для них среде.



Рисунок 11 – PyCharm

Одной из ключевых особенностей PyCharm является интеллектуальный редактор кода, который предлагает автодополнение, анализ кода в реальном времени и удобную навигацию.

Среда также включает встроенный отладчик, поддержку тестирования, интеграцию с системами контроля версий (например, Git) и мощные инструменты для работы с базами данных. Кроме того, PyCharm поддерживает технологии веб-разработки, включая Django и Flask, что делает его отличным выбором для создания сложных проектов [5].

В рамках проекта выбран PyCharm в качестве среды разработки, потому что он предоставляет широкий набор инструментов, необходимых для удобной и продуктивной работы с Python. Благодаря интеллектуальному авто дополнению, встроенному отладчику и гибким настройкам, разработка становится быстрее и удобнее.

PyCharm также поддерживает работу с виртуальными окружениями, что важно при разработке сложных программных решений. Все эти преимущества делают его идеальным выбором для реализации моего проекта.

# Стадии и этапы разработки

Разработка системы для автоматизации обработки билетов онлайн-кинотеатра начинается с детального анализа требований к проекту. Основная цель – создание удобного, надежного и функционального программного продукта, который позволит пользователям управлять и бронировать билеты.

В качестве СУБД выбрана SQL Server Management Studio (SSMS), так как она удобна в использовании, была изучена на втором курсе и соответствует поставленным задачам.

Разработка ведется на языке Python. В качестве среды разработки используется PyCharm, так как она предоставляет лучшие инструменты для работы с кодом и тестирования.

Этапы разработки

1. Проектирование базы данных

На данном этапе определяется структура базы данных, включая ключевые сущности, такие как фильмы, сеансы, билеты и пользователи. Особое внимание уделяется нормализации структуры для устранения избыточности и предотвращения логических ошибок при выполнении запросов.

1. Разработка функционала информационной системы

Реализация ключевых функций, таких как бронирование билетов и управление расписанием сеансов. Для обеспечения взаимодействия приложения с базой данных создается серверная часть и реализуется API, позволяющее эффективно обрабатывать запросы и обмениваться данными.

1. Разработка пользовательского интерфейса

На этапе разработки пользовательского интерфейса разрабатывается удобное и интуитивно понятное оформление приложения. Интерфейс создается с учетом потребностей пользователей и предполагает совместимость с различными устройствами. Особое внимание уделяется удобству навигации и визуальной привлекательности приложения.

После завершения разработки проводится комплексное тестирование системы.

Тестирование системы включает:

1. Функциональное тестирование – проверка работы всех основных функций (бронирование, список фильмов, управление расписанием).
2. Тестирование безопасности – защита пользовательских данных.
3. Тестирование производительности – проверка скорости работы при большом количестве пользователей.

# Глава 2. Практическая часть

# 2.1 Проектирование и нормализация базы данных

В процессе разработки системы для автоматизации обработки билетов онлайн-кинотеатра одним из ключевых этапов является создание базы данных, отвечающей требованиям нормализации. Нормализация — это процесс приведения базы данных к такому виду, который позволяет устранить избыточность данных и избежать логических ошибок при выполнении запросов на выборку или изменение данных.

Применение нормализации позволяет обезопасить базу данных от возможных сбоев, связанных с дублированием информации и нарушением целостности данных. Например, если в таблице содержатся повторяющиеся записи, это может привести к непоследовательности данных при их обновлении. Нормализованная база данных минимизирует такие риски за счёт структурирования информации и установления чётких связей между сущностями.

В рамках данной курсовой работы была разработана структура базы данных онлайн-кинотеатра, которая будет использоваться для управления бронированием билетов, хранением информации о сеансах, фильмах, пользователей, а также для ведения учёта клиентов. Благодаря нормализации обеспечивается целостность и непротиворечивость данных, что позволяет эффективно выполнять автоматизированную обработку информации через разработанное приложение.

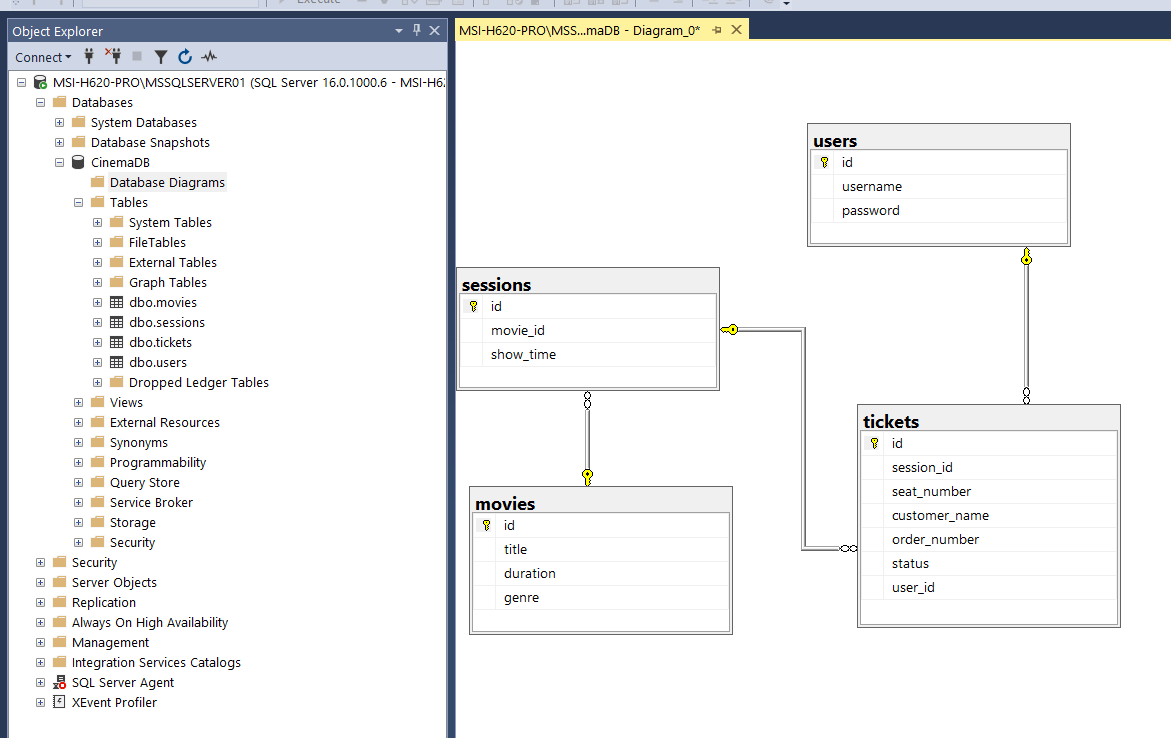


Рисунок 12 - Диаграмма базы данных

Основу данной базы данных составляют несколько ключевых таблиц, которые обеспечивают хранение и обработку данных в приложении. Основная работа с базой данных в процессе подключения к приложению сосредоточена на следующих четырёх таблицах:

1. Movies.dbo (Фильмы)

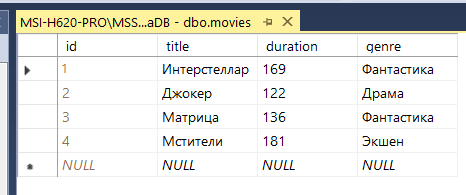


Рисунок 13 - Информация в Movies.dbo

Xранит информацию о фильмах, включая название, жанр и продолжительность. Данная таблица используется при выводе списка доступных фильмов для просмотра и бронирования.

1. Sessions.dbo (Сеансы)

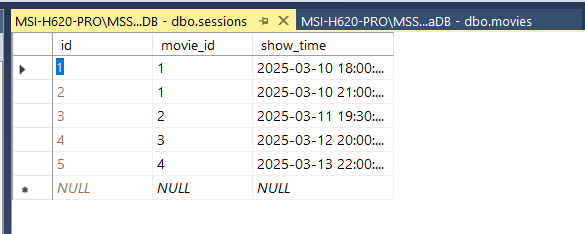


Рисунок 14 - Информация в Sessions.dbo

Cодержит данные о сеансах, включая время показа и привязку к фильму. Эта таблица позволяет отображать актуальное расписание и выбирать нужное время сеанса.

1. Tickets.dbo (Билеты)

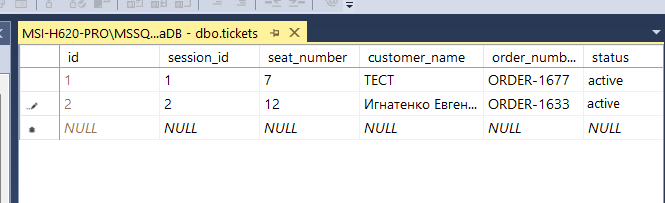


Рисунок 14 - Информация в Tickets.dbo

Xранит данные о забронированных билетах, включая номер заказа, имя клиента, номер места и статус бронирования (активный или отмененный). Эта таблица является главной при обработке и управлении бронированиями.

1. Users.dbo (Пользователи)

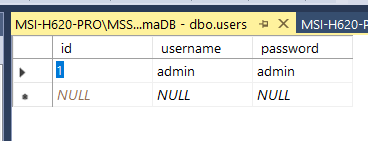


Рисунок 15 - Информация в Users.dbo

Cодержит учетные записи пользователей системы, что позволяет осуществлять вход и управление доступом к функционалу приложения.

Все таблицы связаны между собой посредством ключевых полей, что позволяет корректно формировать запросы и поддерживать целостность данных при выполнении операций с билетами, расписанием и фильмами.

База данных разрабатывалась с учётом нормализации, чтобы избежать дублирования и гарантировать целостность данных. Использование нормальных форм позволило минимизировать избыточность и упростить процесс управления данными при работе с приложением.

**2.2 Разработка информационной системы онлайн-кинотеатра и описание пользовательского интерфейса**

На данном этапе осуществляется настройка логики приложения и его графического интерфейса. Разработка дизайна предполагает создание всех графических элементов приложения — экранов, вкладок, главного экрана, заголовков для отдельных страниц, а также кнопок с индивидуальным дизайном при необходимости.

Перед тем как приступить к разработке приложения, необходимо подключить базу данных к проекту. В среде разработки PyCharm подключение базы данных SQL осуществляется с использованием встроенных инструментов и библиотек, что позволяет эффективно интегрировать базу данных, созданную на предыдущем этапе, с приложением.

После успешного подключения базы данных можно переходить к разработке приложения. Прежде чем приступить к созданию функциональных элементов и логики, необходимо определить графическую основу интерфейса. Выбор цветовой схемы играет важную роль в создании визуально привлекательного и удобного приложения.

Основные цвета приложения подобраны с учетом визуального восприятия и удобства использования. Они обеспечивают гармоничное сочетание элементов интерфейса и создают приятное впечатление для пользователя. В таблице ниже представлены основные цвета, использованные в дизайне приложения:

**Таблица 1**

**Таблица цветов используемых в приложении.**

|  |  |  |
| --- | --- | --- |
| **Назначение** | **Код цвета** | **Цвет** |
| Основной цвет | #E1F5FE | Светло-голубой |
| Цвет кнопок переключения страниц | #005BB5 | Синий |
| Цвет текста | #0078D7 | Голубой |

Графическая основа приложения продумана таким образом, чтобы элементы интерфейса были интуитивно понятными и удобными для пользователя. Используемые цвета создают комфортное восприятие и помогают легко ориентироваться в функционале системы.

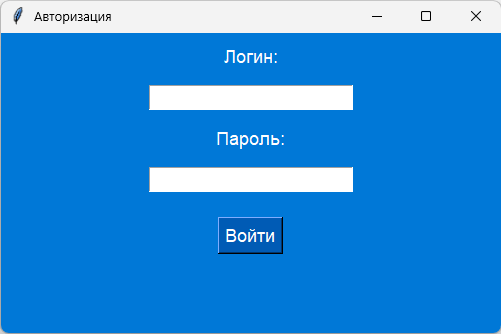


Рисунок 16 – Главная страница приложения.

При запуске приложения пользователю отображается окно авторизации, которое взаимодействует с базой данных пользователей и обрабатывает введенные данные. В зависимости от результата проверки могут возникать следующие ситуации:

1. Если пользователь пытается войти, не заполнив поля логина и пароля, система выдаст сообщение об ошибке с указанием на необходимость ввода данных.
2. Если введенный логин не найден в базе данных, появится сообщение об отсутствии такого пользователя в системе.
3. Если введенный логин существует, но пароль не совпадает с сохраненным в базе, система уведомит об ошибке и предложит повторить ввод.
4. Если пользователь ввел корректные данные, появится окно приложения с соответствующим функционалом.

Таким образом, процесс авторизации обеспечивает безопасный доступ к функционалу приложения и разделяет роли пользователей в системе.

1. **Вкладка «Фильмы».**

Вкладка «Фильмы» является одним из разделов приложения, предоставляющим пользователю удобный и наглядный доступ к актуальной информации о киносеансах. Основное предназначение данной вкладки — отображение перечня фильмов, доступных для просмотра в кинотеатре.

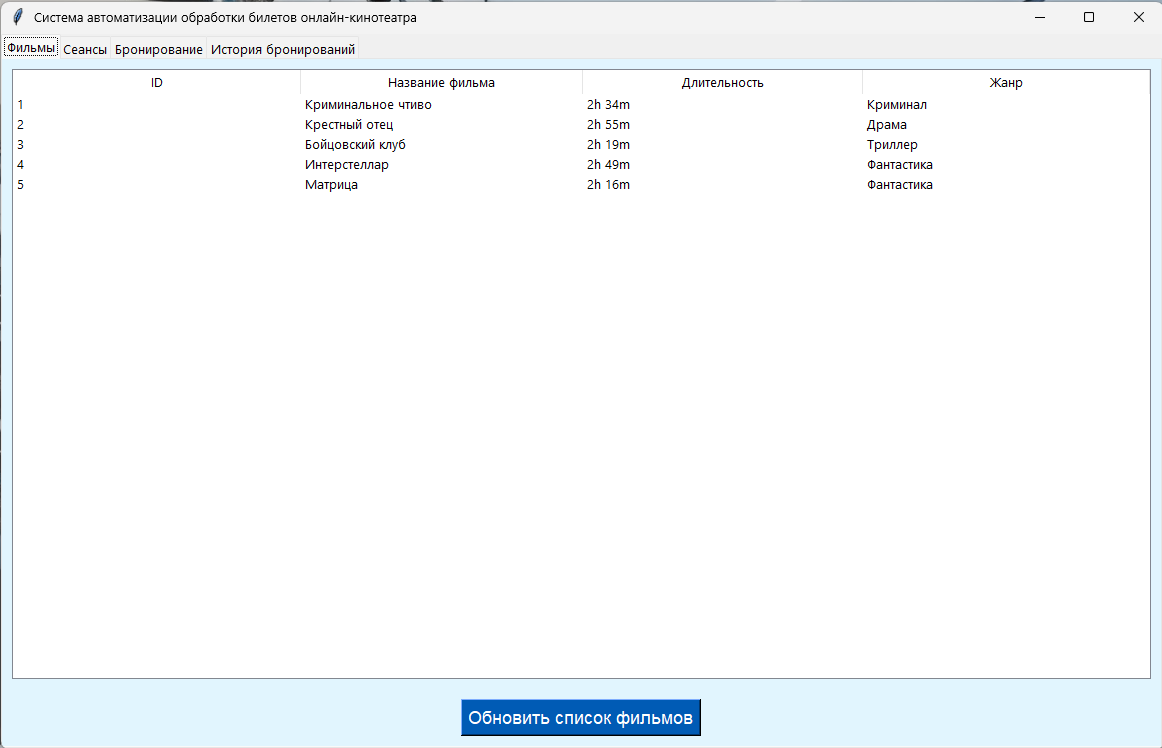


Рисунок 17 – Интерфейс вкладки Фильмы.

Интерфейс вкладки выполнен в простом и лаконичном стиле, что позволяет пользователю с легкостью ориентироваться в представленном списке. Основная часть вкладки занята таблицей, содержащей следующие столбцы:

1. ID — уникальный идентификатор фильма в базе данных, позволяющий однозначно различать все записи.
2. Название фильма — отображает наименование кинокартины, что позволяет быстро идентифицировать фильм по его названию.
3. Длительность — указывает продолжительность фильма в минутах, чтобы пользователь мог оценить временные затраты на просмотр.
4. Жанр — отображает жанровую принадлежность фильма (например, драма, комедия, боевик), что позволяет пользователю выбрать фильм в соответствии со своими предпочтениями.
5. **Вкладка «Сеансы».**

Предназначена для отображения расписания кинопоказов. Она содержит таблицу с перечнем всех сеансов, доступных в базе данных.

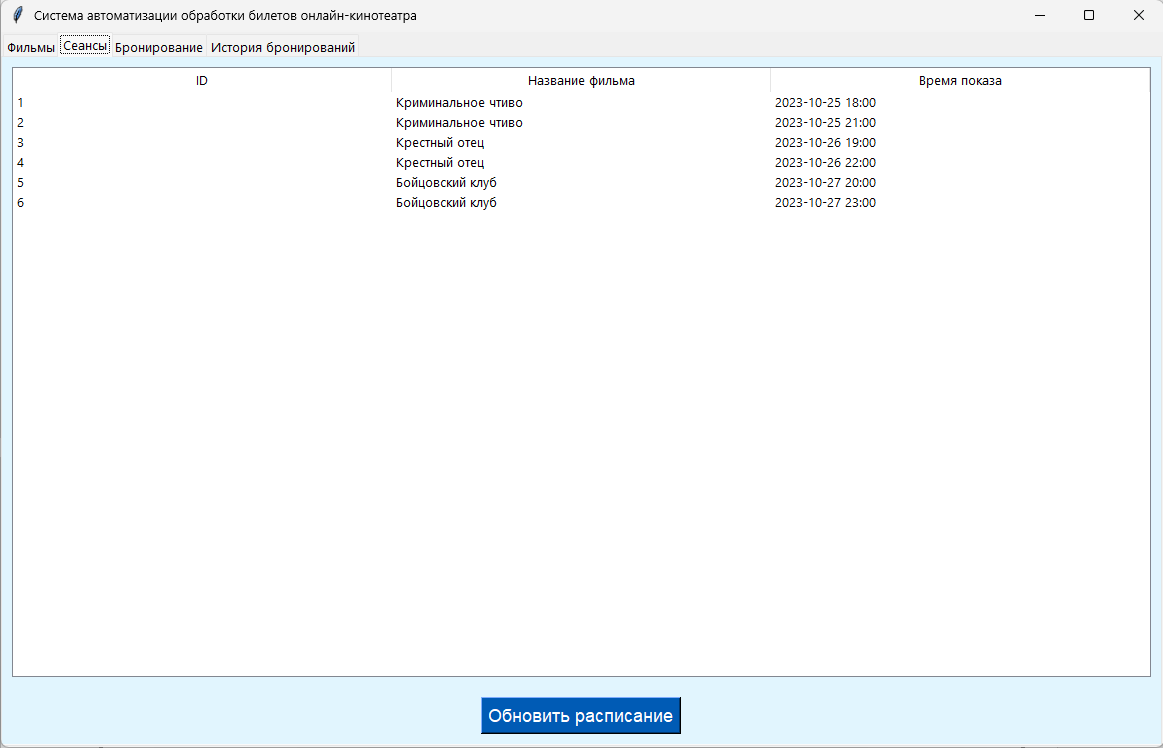


Рисунок 18 – Интерфейс вкладки Сеансы.

Интерфейс вкладки обеспечивает удобное визуальное восприятие информации. Пользователь имеет возможность просматривать актуальные сеансы, уточняя время начала показа и соответствующий фильм.

В нижней части вкладки расположена кнопка «Обновить расписание», которая позволяет пользователю обновить отображаемый список сеансов для получения актуальной информации из базы данных.

1. **Вкладка «Бронирование».**

Содержит управления бронированием билетов на выбранный сеанс. Она предоставляет пользователю возможность внести необходимые данные для создания брони и позволяет удобно управлять процессом бронирования.

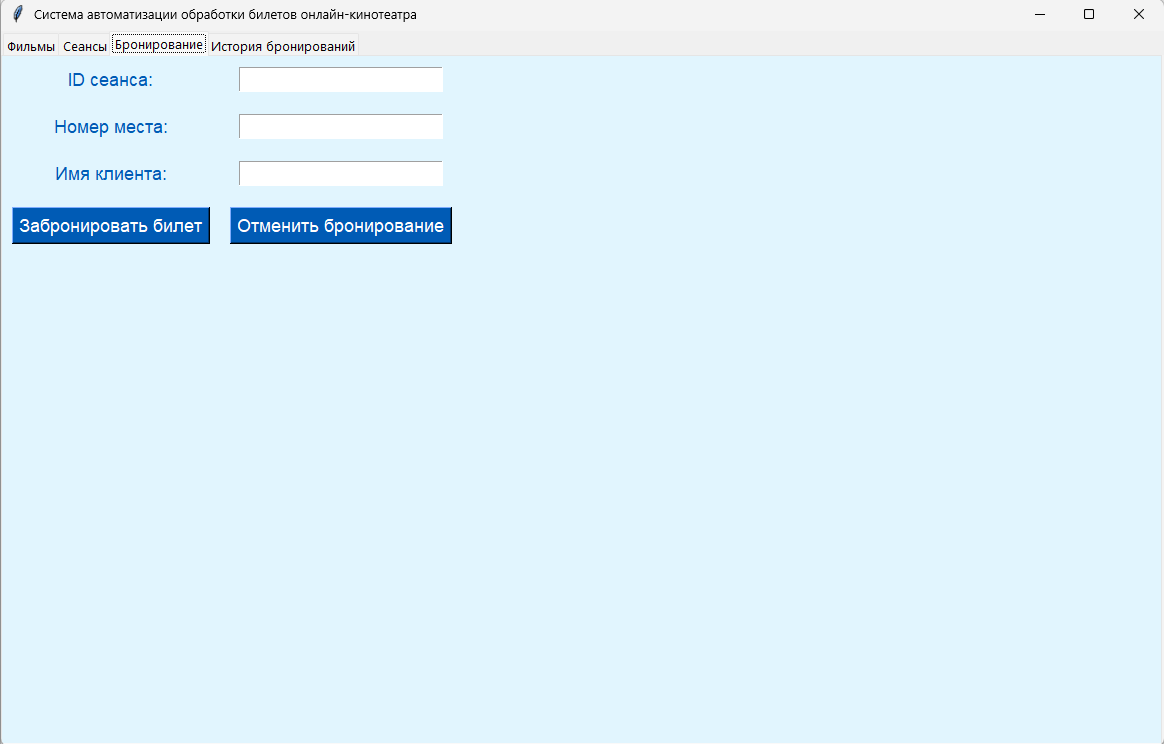


Рисунок 19 – Интерфейс вкладки Бронирование.

Интерфейс вкладки содержит следующие элементы:

1. Поле «ID сеанса» — позволяет ввести уникальный идентификатор сеанса, на который осуществляется бронирование.
2. Поле «Номер места» — используется для указания желаемого места в зале.
3. Поле «Имя клиента» — предназначено для ввода имени посетителя, на которого оформляется бронь.

После заполнения всех полей пользователь может нажать кнопку «Забронировать билет», которая сохраняется в базе данных. В случае успешного бронирования отображается уведомление с номером заказа, подтверждающим оформление билета.

В случае необходимости отмены ранее созданного бронирования на вкладке предусмотрена кнопка «Отменить бронирование». При ее нажатии пользователь может выбрать бронь из списка и снять ее, после чего данные обновляются в системе, и билет становится вновь доступным для заказа.

1. **Вкладка «История бронирования».**

Предназначена для просмотра и управления всеми бронированиями, осуществленными через систему.

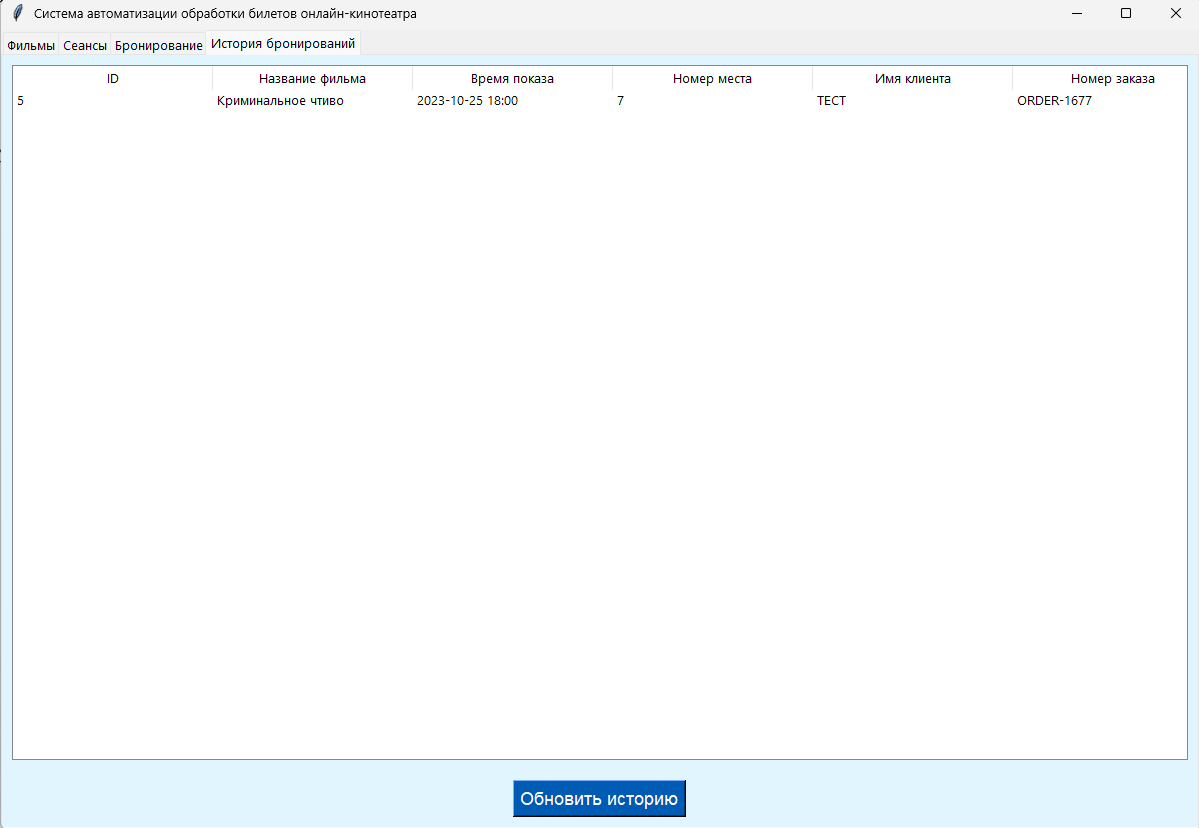


Рисунок 20 – Интерфейс вкладки История бронирования.

После успешного бронирования данные автоматически добавляются в таблицу истории, позволяя сотрудникам быстро получать информацию по каждому заказу.

Кроме того, на вкладке расположена кнопка «Обновить историю», которая позволяет оперативно получить актуальные данные о всех существующих бронированиях после внесенных изменений или новых бронирований.

**2.3 Тестирование созданного приложения**

Тестирование созданного приложения является важным этапом разработки, так как позволяет выявить возможные ошибки и убедиться в корректности работы всех функций. Основная цель тестирования — проверить, чтобы система работала стабильно и без сбоев при различных сценариях использования.

Функциональное тестирование

При запуске приложения пользователю необходимо пройти процесс авторизации, введя логин и пароль

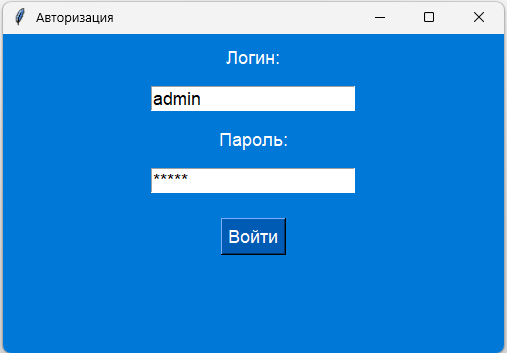


Рисунок 21 – Окно входа в приложение.

При вводе данных авторизации пароль скрыт в целях безопасного ввода конфиденциальной информации

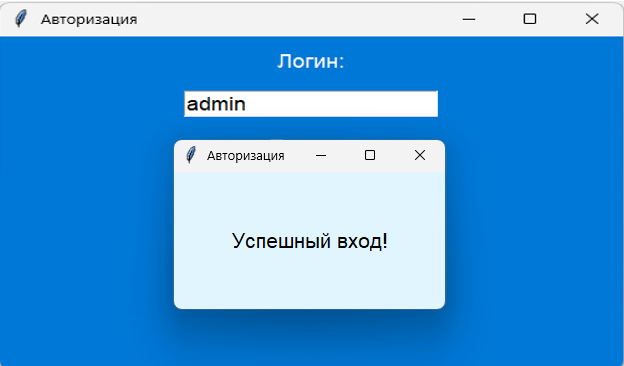


Рисунок 22 – Подтверждение системы о входе.

В процессе тестирования системы особое внимание уделялось проверке корректности работы функционала авторизации. Тестирование проводилось с использованием тест-кейсов, представляющих собой детально описанные алгоритмы проверки функциональности, включающие последовательность шагов и входные данные, приводящие к ожидаемому результату.

**Описание тест-кейса №1**

**Таблица 2**

|  |  |
| --- | --- |
| Название | Проверка авторизации |
| ID-теста | RC001 |
| Описание | Проверка функционала авторизации на подтверждение корректности обработки введённых данных (логин и пароль) и отображения соответствующего уведомления при успешной или неуспешной авторизации. |
| Ожидаемый результат | 1. При вводе корректных данных (логин: «admin», пароль: «admin») осуществляется успешная авторизация и открывается главное окно системы.  2. При вводе некорректных данных отображается сообщение об ошибке: “Неверный логин или пароль!”. |
| Выполняемые действия | 1. Открыть окно авторизации системы.  2. Ввести корректные данные (логин: “admin”, пароль: “admin”).  3. Нажать кнопку “Войти”.  4. Закрыть главное окно.  5. Повторно открыть окно авторизации.  6. Ввести некорректные данные (например, логин: “user”, пароль: “1234”).  7. Нажать кнопку “Войти”. |
| Фактический результат | Авторизация с корректными данными прошла успешно, окно открылось.  При неверных данных и пустых полях появилось сообщение «Неверный логин или пароль» |

В данном тестировании проводится проверка функционала бронирования и отмены билета в системе онлайн-кинотеатра. Целью данного тест-кейса является убедиться в корректной работе основных операций с билетами: создание брони и её последующая отмена.

**Описание тест-кейса №2**

**Таблица 3**

|  |  |
| --- | --- |
| Название | Проверка бронирования и отмены билета |
| ID-теста | RC002 |
| Описание | Проверка корректности работы функционала бронирования и отмены билета в системе онлайн-кинотеатра. |
| Ожидаемый результат | 1. После ввода данных бронирования в истории отображается запись с уникальным номером заказа (например, ORDER-1633).  2. При отмене бронирования запись удаляется. |
| Выполняемые действия | 1. Открыть вкладку «Бронирование». 2. Ввести данные  ID сеанса: 4  Номер места: 12  Имя клиента: Игнатенко Евгений  3. Нажать кнопку «Забронировать».  4. Проверить наличие записи в истории бронирования.  5. Выбрать созданный билет  6. В вкладе бронирование нажать отмена |
| Фактический результат | Запись успешно добавлена в историю бронирования с уникальным номером заказа «ORDER-1633»    После выбора билета и отмене бронирования запись удалилась. |

## ЗАКЛЮЧЕНИЕ

В процессе работы над данным курсовым проектом по теме «Разработка системы для автоматизации обработки билетов онлайн-кинотеатра» были выполнены следующие задачи:

1. Проведён анализ требований к системе автоматизации обработки билетов и определены ключевые функции приложения.

2. Разработана структура базы данных с учётом нормализации и оптимизации данных. Созданы основные таблицы: пользователи, фильмы, сеансы и бронирования.

3. Подобран стек технологий и выбрана среда разработки PyCharm с использованием языка программирования Python и базы данных SQL.

4. Разработано приложение с графическим пользовательским интерфейсом, включающее основные вкладки: авторизация, фильмы, сеансы, бронирование и история бронирований.

5. Проведено тестирование системы с целью проверки функционала, надёжности и корректности выполнения основных операций. Выполнено тестирование авторизации, бронирования билетов, отображения данных и их отмены.

В результате выполнения проекта была достигнута поставленная цель разработки системы автоматизации обработки билетов для онлайн-кинотеатра. Созданная система позволяет эффективно управлять процессом бронирования, обеспечивает удобство использования и корректное хранение данных.

В дальнейшем планируется развитие функционала системы, включая интеграцию с онлайн-платежами для покупки билетов через приложение, добавление уведомлений о предстоящих сеансах и реализации инструментов для анализа посещаемости и популярности фильмов.

## Список литературы

1. Александр, А. Н., Денис, К. И. Информационные системы: теория и практика. [Текст] / А. Н. Александр, К. И. Денис — 2-е изд.. — Москва: Академия, 2023 — 312 c.

2. Михеев, А. С., Громов, Е. В. SQL для начинающих. [Текст] / А. С. Михеев, Е. В. Громов — 1-е изд.. — Москва: Диалектика, 2022 — 412 c.

3. Смирнов, А. В. Программирование на Python: основы и приложения. [Текст] / А. В. Смирнов — 3-е изд.. — Москва: Диалектика, 2024 — 371 c.

4. Федоров, П. А., Климов, С. В. Технологии разработки пользовательских интерфейсов на Python. [Текст] / П. А. Федоров, С. В. Климов — 1-е изд.. — Казань: Инфра-М: Диалектика, 2024 — 340 c.

5. Иван, А. В., Вадим, Н. В. Тестирование программного обеспечения: методология и практика. [Текст] / А. В. Иван, Н. В. Вадим — 4-е изд.. — Новосибирск: Наука, 2022 — 275 c.

6. Кузнецов, Д. Н., Орлов, В. П. Разработка баз данных: проектирование и оптимизация. [Текст] / Д. Н. Кузнецов, В. П. Орлов — 1-е изд.. — Екатеринбург: Уральский университет, 2022 — 346 c.

7. Николаев, Ю. П., Сидоров, В. А. Архитектура информационных систем: теория и практика. [Текст] / Ю. П. Николаев, В. А. Сидоров — 2-е изд.. — Ростов-на-Дону: Феникс, 2023 — 384 c.

8. Григорьев, М. И., Платонов, Д. В. Разработка графических интерфейсов на Python с использованием Tkinter. [Текст] / М. И. Григорьев, Д. В. Платонов — 1-е изд.. — Самара: Университетское издательство, 2024 — 298 c.

9. Соколов, А. Н., Кузьмин, П. Т. Управление базами данных на платформе SQL Server. [Текст] / А. Н. Соколов, П. Т. Кузьмин — 3-е изд.. — Владивосток: Дальневосточный университет, 2023 — 430 c.

10. Петров А.В., Смирнов К.И. Документация по языку программирования Python. Python Documentation. / Петров А.В., Смирнов К.И. [Электронный ресурс] // Python Software Foundation. : [сайт]. — URL: https://docs.python.org/ (дата обращения: 22.03.2025).

11. Иванов Д.С., Козлов М.Н. Руководство по созданию графических интерфейсов с использованием Tkinter. / Иванов Д.С., Козлов М.Н. [Электронный ресурс] // TkDocs. TkDocs Community. : [сайт]. — URL: https://tkdocs.com/ (дата обращения: 22.03.2025).

12. Андреев В.К., Михайлов Ю.Н. Использование PyODBC для подключения к SQL Server. / Андреев В.К., Михайлов Ю.Н. [Электронный ресурс] // GitHub. GitHub Community. : [сайт]. — URL: https://github.com/mkleehammer/pyodbc (дата обращения: 22.03.2025).

13. Фёдоров А.И., Беляев П.С. Ответы на вопросы по разработке на Python и работе с базами данных. / Фёдоров А.И., Беляев П.С. [Электронный ресурс] // Stack Overflow. Stack Overflow Community. : [сайт]. — URL: https://stackoverflow.com/ (дата обращения: 22.03.2025).

14. Морозов Е.В., Гусев Л.Н. Статьи по созданию информационных систем и баз данных. / Морозов Е.В., Гусев Л.Н. [Электронный ресурс] // Хабр. Хабр Сообщество. : [сайт]. — URL: https://habr.com/ (дата обращения: 22.03.2025).

15. Сидоров А.В., Крылов П.М. Информационные системы: основы и принципы. / Сидоров А.В., Крылов П.М. [Электронный ресурс] // Википедия. : [сайт]. — URL: https://ru.wikipedia.org/wiki/Информационная\_система (дата обращения: 18.03.2025).