МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«Гомельский государственный технический университет имени П.О.Сухого»

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

Специальность 1-40 05 01-01 «Информационные системы и технологии

(в проектировании и производстве)»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

по дисциплине «Объектно-ориентированное программирование»

на тему «Информационная система распространения билетов в театре»

Составил: студент гр. ИТП-21 Пронуза М.Ю.

(подпись, дата)

Руководитель: Курочка К.С.

(должность)

(подпись, дата)

Дата защиты

Оценка

Подписи членов комиссии

Гомель 2024

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#__RefHeading___Toc4052_3340417902)

[1. ЛИТЕРАТУРНЫЙ ОБЗОР 4](#__RefHeading___Toc4054_3340417902)

[1.1 Обзор языков программирования 4](#__RefHeading___Toc4056_3340417902)

[1.2 Анализ технологий для разработки пользовательского интерфейса и системы управления базами данных 4](#__RefHeading___Toc1105_4269907934)

[1.3 Анализ средств разработки пользовательского интерфейса 5](#__RefHeading___Toc1441_3825721904)

[1.4 Сравнение систем управления базами данных (СУБД) 6](#__RefHeading___Toc1443_3825721904)

[1.5 Рассмотрение других технологических решений 7](#__RefHeading___Toc1445_3825721904)

# ВВЕДЕНИЕ

Разработка программного обеспечения для специфических предметных областей, таких как учет и реализация продаж билетов на спектакли, требует тщательного подхода к выбору технологических инструментов. Эффективное решение данной задачи не только облегчает процесс разработки и последующую поддержку программного продукта, но и определяет его функциональность, производительность, удобство использования и масштабируемость. В современном мире информационных технологий существует множество языков программирования, платформ разработки, систем управления базами данных (СУБД), и других инструментов, каждый из которых обладает уникальными особенностями, предназначен для определенных целей и задач.

В рамках данной работы, основное внимание уделяется выбору средств для реализации приложения, способного эффективно справиться с задачами учета и реализации продаж билетов на спектакли в театре. При этом, особое внимание уделяется следующим аспектам: выбору языка программирования, технологии разработки пользовательского интерфейса, а также системы управления базами данных. Учитывая специфику проекта, важно также рассмотреть возможности интеграции выбранных инструментов, их производительность, безопасность, и удобство использования как для разработчиков, так и для конечных пользователей.

Выбор языка программирования *C#*, технологии *Windows Presentation Foundation (WPF)* для разработки графического интерфейса пользователя, а также системы управления базами данных *Microsoft SQL Server* обусловлен рядом факторов, включая требования к функциональности приложения, его масштабируемости, безопасности, а также специфические требования к удобству работы с данными и визуализации информации. Проведено сравнение предложенных технологий с альтернативными решениями, а также детально рассмотрены преимущества выбранных инструментов с точки зрения реализации проекта.

Прежде чем перейти к анализу и сравнению технологических решений, необходимо четко определить требования к разрабатываемому приложению, его основные функции и ожидаемую нагрузку. Так, приложение должно предоставлять функциональность для учета и реализации продаж билетов, управления пользователями с различными уровнями доступа (администратор, гость, курьер, зарегистрированный пользователь), формирования статистических отчетов, а также поиска информации по различным критериям (дата представления, жанр спектакля, наличие свободных билетов). Эти и другие функциональные требования играют ключевую роль в процессе выбора технологий для реализации проекта.

# 1. **ЛИТЕРАТУРНЫЙ ОБЗОР**

## 1.1 Обзор языков программирования

Выбор языка программирования оказывает значительное влияние на весь жизненный цикл проекта, начиная от разработки и заканчивая поддержкой. В современной практике разработки ПО много языков с уникальными особенностями. Для данного проекта, предназначенного для учета и реализации продаж билетов на спектакли, рассматривались *C#*, *Java* и *Python*.

*C#* является объектно-ориентированным языком от *Microsoft*, встроенным в платформу .*NET*. Он идеально подходит для разработки настольных приложений для *Windows*, благодаря поддержке *WPF*, что позволяет создавать богатые пользовательские интерфейсы. Этот язык обладает строгой типизацией и обеспечивает высокую производительность и безопасность приложений.

*Java*, разработанный *Sun Microsystems*, широко используется для веб-приложений, мобильных приложений и корпоративных решений. Он обеспечивает высокую переносимость благодаря *JVM*, что позволяет *Java*-приложениям работать на различных платформах. Несмотря на это, *Java* может не быть оптимальным выбором для разработки настольных приложений, специализированных под *Windows*, в сравнении с возможностями, предлагаемыми *C#* и .*NET*.

*Python* известен своей универсальностью и широко используется в веб-разработке, научных исследованиях и анализе данных. Его синтаксис легко читаем, что делает язык доступным для начинающих. Однако, несмотря на удобство и гибкость, *Python* не предоставляет такого же уровня поддержки для разработки настольных приложений, как *C#*, особенно когда речь идет о создании сложных интерфейсов и требованиях к производительности.

При сравнении этих языков *C#* выделяется как наиболее подходящий для нашего проекта, особенно учитывая требования к функциональности, производительности и безопасности. Тесная интеграция с .*NET* и *Windows*, а также поддержка *WPF* делают *C#* оптимальным выбором для создания настольного приложения, направленного на учет и реализацию продаж билетов. Выбор в пользу *C#* подкрепляется его способностью к созданию богатых интерактивных пользовательских интерфейсов, важность которых нельзя недооценивать в приложениях, предназначенных для широкого круга пользователей. Таким образом, исходя из анализа доступных языков программирования и учитывая специфические требования проекта, *C#* является наилучшим выбором для разработки.

## 1.2 Анализ технологий для разработки пользовательского интерфейса и системы управления базами данных

После выбора *C#* в качестве основного языка программирования для разработки настольного приложения, следующим шагом является выбор подходящих технологий для разработки пользовательского интерфейса и системы управления базами данных. Эти компоненты являются критически важными для обеспечения функциональности, производительности и удобства использования приложения.

В сфере разработки пользовательских интерфейсов для настольных приложений на платформе *Windows*, *WPF*, или *Windows Presentation Foundation*, выделяется своими возможностями создания богатых и интерактивных интерфейсов. Использование *WPF* позволяет разработчикам и дизайнерам эффективно сотрудничать, разделяя логику приложения и его визуальное представление. Благодаря поддержке привязки данных, анимации и шаблонов проектирования, таких как *MVVM*, *WPF* делает возможным создание интуитивно понятных и визуально привлекательных приложений.

Когда дело доходит до управления данными, выбор подходящей системы управления базами данных (СУБД) критически важен. *Microsoft SQL Server*, благодаря своей масштабируемости, производительности и интеграции с экосистемой *Microsoft*, представляет собой оптимальный выбор для проектов, требующих надежного управления данными. *MS SQL Server* предлагает расширенные функции безопасности, поддержку транзакций и инструменты анализа данных, что делает его идеальным для приложений с большими объемами данных. Интеграция с *C#* и .*NET* упрощает разработку и поддержку, предоставляя мощные инструменты, такие как *LINQ*, для удобной и типобезопасной работы с данными.

Таким образом, комбинация *C#*, *WPF* и *Microsoft SQL Server* создает мощный инструментарий для разработки настольного приложения. Эти технологии не только удовлетворяют требованиям к функциональности и производительности, но и обеспечивают разработку сложного пользовательского интерфейса с превосходной поддержкой визуализации данных и взаимодействия с пользователем. Выбор этих инструментов лежит в основе успешной реализации проекта, направленного на учет и реализацию продаж билетов на спектакли, обеспечивая высокое качество и удобство использования разрабатываемого программного продукта.

## 1.3 Анализ средств разработки пользовательского интерфейса

Анализ средств разработки пользовательского интерфейса является ключевым аспектом при выборе технологий для разработки настольного приложения, особенно когда требования включают разработку высококачественного, интерактивного и визуально привлекательного интерфейса пользователя. В контексте создания приложений для Windows, основными кандидатами являются *Windows Presentation Foundation (WPF), Windows Forms (WinForms)* и *Universal Windows Platform (UWP)*.

*WPF*, разработанный *Microsoft*, представляет собой фреймворк для создания настольных приложений с использованием .*NET*. Он использует *XAML* для определения пользовательского интерфейса и предлагает мощные возможности для создания динамичных и настраиваемых интерфейсов, поддерживает разделение логики приложения и интерфейса с помощью модели *MVVM (Model-View-ViewModel)*, что упрощает тестирование и поддержку кода. *WPF* идеально подходит для разработки сложных пользовательских интерфейсов с расширенными графическими возможностями, такими как анимация, трехмерная графика и стилизация.

*Windows Forms*, будучи более старым фреймворком для создания пользовательского интерфейса в .*NET*, обеспечивает простоту и быстроту разработки приложений благодаря обширной библиотеке готовых к использованию элементов управления. Несмотря на свою доступность и простоту, *WinForms* ограничивается в плане возможностей кастомизации интерфейса и меньшей поддержкой сложных графических возможностей по сравнению с *WPF*.

*Universal Windows Platform (UWP)* – это платформа от *Microsoft* для создания приложений, работающих на всех устройствах под управлением *Windows 10*, включая настольные компьютеры, планшеты, телефоны и даже *Xbox*. *UWP* предлагает современный подход к разработке пользовательского интерфейса, обеспечивая адаптивность и масштабируемость. Однако, несмотря на его гибкость и универсальность, *UWP* может оказаться избыточным для проектов, ориентированных исключительно на настольные приложения для *Windows*.

В контексте разработки приложения для реализации и учета продаж билетов на спектакли, выбор *WPF* как средства разработки пользовательского интерфейса обусловлен несколькими факторами. Во-первых, богатая поддержка графических возможностей и анимации позволяет создавать привлекательный и интуитивно понятный интерфейс, который может повысить удовлетворенность пользователя и облегчить взаимодействие с функционалом приложения. Во-вторых, гибкая архитектура и поддержка модели *MVVM* облегчают масштабирование приложения и внесение изменений в дизайн без необходимости переписывания бизнес-логики. Наконец, тесная интеграция с .*NET* *Framework* и *C#* предоставляет разработчикам широкий спектр инструментов и библиотек для эффективной работы над проектом, включая доступ к данным, сериализацию и асинхронное программирование, что делает *WPF* оптимальным выбором для разработки комплексных настольных приложений.

## 1.4 Сравнение систем управления базами данных (СУБД)

В современном программировании выбор системы управления базами данных (СУБД) играет ключевую роль в успешной реализации и поддержке проектов, требующих надежного, эффективного и безопасного хранения данных. СУБД, такие как *MS SQL*, *MySQL*, и *PostgreSQL*, предлагают различные функции, производительность, модели лицензирования и поддержку, что делает их более или менее подходящими для определенных типов приложений.

*MS SQL Server* от *Microsoft* часто выбирают за его надежность, масштабируемость и глубокую интеграцию с другими продуктами *Microsoft*, что делает его идеальным выбором для предприятий, уже использующих экосистему Microsoft. Он предлагает мощные инструменты для анализа данных, комплексную поддержку транзакций, высокую производительность, сложные механизмы безопасности и обширные возможности для управления данными и разработки приложений. Это особенно ценно для приложений, где требуется высокая степень согласованности данных, поддержка сложных транзакций и строгие требования к безопасности.

*MySQL*, с другой стороны, является открытым исходным кодом и широко используется для веб-приложений, благодаря своей простоте, надежности и легкости в использовании. Она хорошо подходит для проектов с ограниченным бюджетом или там, где необходима высокая степень переносимости между различными операционными системами. *MySQL* оптимизирована для операций чтения, что делает ее подходящим выбором для веб-сайтов и приложений, где чтение данных происходит гораздо чаще, чем их запись.

*PostgreSQL*, еще одна популярная открытая СУБД, выделяется своей расширяемостью и стандартами соответствия *SQL*. Она предлагает богатую функциональность, включая сложные запросы, внешние ключи, процедурные языки и многие типы индексов. *PostgreSQL* часто выбирают для сложных приложений, требующих расширенной обработки транзакций и аналитики в реальном времени. Эта СУБД предлагает высокую производительность для комплексных запросов и больших объемов данных, что делает ее предпочтительным выбором для предприятий, которым требуются сложные аналитические возможности и поддержка больших данных.

Выбор между этими СУБД зависит от множества факторов, включая требования проекта, существующую инфраструктуру, бюджет, требования к производительности и предпочтения разработчиков. *MS SQL* часто предпочитают за его производительность, безопасность и масштабируемость в корпоративной среде, особенно когда уже присутствует интеграция с другими продуктами *Microsoft*. *MySQL* привлекает своей простотой и надежностью для малых и средних веб-проектов. *PostgreSQL* же выбирают за его мощные возможности и гибкость в более сложных и данных-интенсивных приложениях.

В контексте разработки приложения для учета продаж билетов на спектакли в театре, важно учитывать специфику данных, операций и требований к безопасности. *MS SQL* может предложить преимущества в управлении данными, интеграции и безопасности, которые будут важны для обработки финансовых транзакций, хранения чувствительной информации о пользователях и обеспечения надежной работы приложения в корпоративной среде.

## 1.5 Рассмотрение других технологических решений

В разработке современных программных решений, помимо выбора языка программирования и системы управления базами данных, имеет значение и применение дополнительных технологических инструментов и подходов, которые могут значительно повысить эффективность разработки, облегчить поддержку и расширение проекта, а также улучшить взаимодействие с пользователем. В контексте разработки приложения для учета и продажи билетов на спектакли, важными технологиями являются *LINQ* для обработки данных, паттерны проектирования для организации кода, а также подходы к тестированию для обеспечения надежности и безопасности приложения.

*LINQ (Language Integrated Query)* представляет собой мощный инструмент для запросов к данным непосредственно из *C#*. Его использование позволяет упростить и унифицировать работу с данными, делая код более читаемым и легким для понимания. *LINQ* поддерживает множество источников данных, включая коллекции в памяти, базы данных и *XML*-файлы, что делает его универсальным решением для многих задач по обработке данных. Применение *LINQ* в проекте позволяет разработчикам выполнять сложные запросы к данным без необходимости писать объемный и сложный *SQL*-код, что сокращает время разработки и упрощает поддержку приложения.

Паттерны проектирования играют ключевую роль в создании структурированного, масштабируемого и легко поддерживаемого кода. Они предоставляют проверенные решения для часто встречающихся проблем в программировании. В контексте разработки приложения для театра могут быть особенно полезны такие паттерны, как *MVC (Model-View-Controller)* для разделения логики приложения, данных и пользовательского интерфейса, *Singleton* для обеспечения единственного экземпляра объекта (например, подключения к базе данных) и *Observer* для реализации подписки на события (например, обновления данных о билетах). Использование этих и других паттернов помогает организовать код таким образом, чтобы его было легче модифицировать и тестировать, что является важным аспектом при разработке крупных и сложных систем.

Тестирование является неотъемлемой частью процесса разработки программного обеспечения, обеспечивающей высокий уровень качества и надежности приложения. Модульные тесты позволяют проверить отдельные компоненты программы в изоляции от остальной системы, что помогает выявлять и устранять ошибки на ранних этапах разработки. Использование фреймворков для модульного тестирования, таких как *NUnit* или *xUnit* для *C#*, упрощает создание и выполнение тестов, а также поддержку тестового кода в актуальном состоянии. Применение практик тестирования повышает уверенность в том, что приложение будет функционировать как ожидается, и помогает предотвратить потенциальные проблемы при его эксплуатации.

В целом, использование дополнительных технологических решений, таких как *LINQ*, паттерны проектирования и подходы к тестированию, существенно улучшает процесс разработки и качество конечного продукта. Эти инструменты и методики помогают создать надежное, легко масштабируемое и поддерживаемое приложение, что особенно важно для систем, требующих высокой стабильности и производительности, таких как система учета и продажи билетов на спектакли.

**1.6 Выводы**

В первой главе курсовой работы был проведен тщательный обзор и анализ доступных технологий для разработки приложения по учету и продаже билетов на спектакли в театре. Основное внимание уделялось выбору подходящего языка программирования, фреймворка для разработки пользовательского интерфейса, системы управления базами данных, а также других технологических решений, способных повысить эффективность и надежность разработки.

**Основные выводы:**

1. **Выбор языка программирования и фреймворка:** *C#* в сочетании с *WPF* были выбраны как основные технологии для разработки настольного приложения. Этот выбор обосновывается мощной поддержкой .*NET* *Framework*, возможностью создания богатых и интерактивных пользовательских интерфейсов с помощью *WPF*, а также хорошей интеграцией с операционной системой *Windows*.
2. **Система управления базами данных:** *MS SQL Server* была выбрана в качестве СУБД для данного проекта из-за ее надежности, производительности, масштабируемости и глубокой интеграции с экосистемой Microsoft. Сравнение с другими СУБД, такими как *MySQL* и *PostgreSQL*, показало, что *MS SQL* лучше соответствует требованиям проекта, особенно в плане обработки транзакций и безопасности данных.
3. **Дополнительные технологические решения:** Использование *LINQ* значительно упрощает работу с данными, позволяя писать запросы непосредственно на *C#*, что обеспечивает высокую читаемость и удобство поддержки кода. Паттерны проектирования и модульные тесты вносят весомый вклад в качество и надежность приложения, облегчая его разработку, тестирование и последующее сопровождение.

Выбор *C#*, *WPF* и *MS SQL Server* в качестве основных технологий для разработки приложения по продаже билетов в театре является обоснованным решением, отвечающим всем требованиям проекта. Эти технологии обеспечивают необходимую производительность, безопасность и масштабируемость приложения, а также предлагают разработчикам мощные инструменты для создания качественного и удобного в использовании программного продукта. Дополнительное применение *LINQ*, паттернов проектирования и модульных тестов повышает эффективность разработки и гарантирует высокое качество конечного продукта, что делает выбранное сочетание технологий наиболее предпочтительным для реализации поставленных задач.