**ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ И НАУКИ ГОРОДА МОСКВЫ**

**ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**«Колледж автоматизации и информационных технологий № 20»**

**ДОПУСКАЕТСЯ К ЗАЩИТЕ:**

Заместитель директора по

качеству образования

\_\_\_\_\_\_\_\_\_\_\_\_ / Глазков П.А.

(подпись) (Ф.И.О.)

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2025 г.

**Выпускная квалификационная работа (дипломный проект)**

**студента, завершающего освоение специальности**

09.02.07 Информационные системы и программирование

(код, наименование специальности)

Тема «Разработка информационной системы по автоматизации процесса бронирования и продаж для ООО «Глобал Текс»

Квалификация: программист

Группа ИСП321

Выполнил студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Андреева Софья Леонидовна

(подпись) (Ф.И.О. студента)

Руководитель выпускной квалификационной работы (дипломного проекта)

\_\_\_\_\_\_\_\_\_ /Лисавина Алёна Вадимовна

(подпись) (Ф.И.О. руководителя)

Москва, 2025 г.

**ДЕПАРТАМЕНТ ОБРАЗОВАНИЯ И НАУКИ ГОРОДА МОСКВЫ**

**ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**

**«Колледж автоматизации и информационных технологий № 20»**

|  |  |
| --- | --- |
| **РАССМОТРЕНО**:  на заседании ППО  «Информационные технологии»  от «04» апреля 2025 г.  Протокол № 5  Заведующий учеб. отделением  \_\_\_\_\_\_\_\_\_\_\_\_/ Первов Г.А.  (подпись) (Ф.И.О.)  Руководитель по ВКР  \_\_\_\_\_\_\_\_\_\_\_\_/Лисавина А.В.  (подпись) (Ф.И.О.) | **УТВЕРЖДАЮ**:  Заместитель директора по  качеству образования  \_\_\_\_\_\_\_\_\_\_\_\_/ Глазков П.А.  (подпись) (Ф.И.О.)  «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 202\_\_ г. |

**ЗАДАНИЕ   
на выпускную квалификационную работу (дипломный проект)**

Студенту Андреевой Софье Леонидовне

Тема выпускной квалификационной работы (дипломного проекта): «Разработка информационной системы по автоматизации процесса бронирования и продаж для ООО «Глобал Текс»

|  |  |  |
| --- | --- | --- |
| **Содержание выпускной квалификационной работы** | **Сроки исполнения задания** | **Примечание** |
| ВВЕДЕНИЕ (~ 5%) | 19.05.2025 |  |
| 1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И СУЩЕСТВУЮЩИХ РЕШЕНИЙ (~25-30%)  1.1 Описание предметной области | 20.05.2025-21.05.2025 |  |
| 1.2 Анализ бизнес-процессов художественной галереи  1.3 Целевые пользователи и их потребности  1.4 Анализ аналогов и существующих решений  1.5 Требования к функционалу приложения | 21.05.2025 |  |
| 1.6 Технологический стек  1.6.1 Выбор системы управления базой данных  1.6.2 Выбор языка программирования  1.6.3 Выбор среды разработки | 22.05.2025 |  |
| 1.7 Особенности конфиденциальности и безопасности данных  1.8 Проблемы текущих подходов к управлению коллекцией | 22.05.2025-23.05.2025 |  |
| 2 (~55-60%) | 23.05.2025-26.05.2025 |  |
|  | 26.05.2025 |  |
|  | 27.05.2025-30.05.2025 |  |
|  | 30.05.2025-04.06.2025 |  |
|  | 04.06.2025 |  |
| ЗАКЛЮЧЕНИЕ (~ 8 %) | 04.06.2025 |  |
| СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ | 05.06.2025 |  |

Дата выдачи задания «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_202\_\_ г.

Задание получил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_202\_\_ г.

(подпись студента)

СОДЕРЖАНИЕ

[СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ 6](#_Toc199806379)

[ВВЕДЕНИЕ 7](#_Toc199806380)

[1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И СУЩЕСТВУЮЩИХ РЕШЕНИЙ 9](#_Toc199806381)

[1.1 Описание и анализ предметной области 9](#_Toc199806382)

[1.2 Анализ взаимодействия музеев и галерей с поставщиками 12](#_Toc199806383)

[1.3 Целевые пользователи и их потребности 13](#_Toc199806384)

[1.4 Анализ аналогов и существующих решений 15](#_Toc199806385)

[1.5 Требования к функционалу приложения 22](#_Toc199806386)

[1.6 Технологический стек 23](#_Toc199806387)

[1.6.1 Выбор системы управления базой данных 23](#_Toc199806388)

[1.6.2 Выбор языка программирования 29](#_Toc199806389)

[1.6.3 Выбор среды разработки 32](#_Toc199806390)

[1.7 Особенности конфиденциальности и безопасности данных 34](#_Toc199806391)

[1.8 Недостатки существующих методов управления оборудованием 35](#_Toc199806392)

[2 ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ 37](#_Toc199806393)

[2.1 Архитектура приложения 37](#_Toc199806394)

[2.2 Технология ADO.NET 38](#_Toc199806395)

[2.3 Язык разметки XAML 41](#_Toc199806396)

[2.4 Библиотека Microsoft.Office.Interop.Word 44](#_Toc199806397)

[2.5 Язык LINQ 46](#_Toc199806398)

[2.6 Проектирование системы 48](#_Toc199806399)

[2.6.1 Диаграмма активностей 49](#_Toc199806400)

[2.6.2 Диаграмма прецедентов 51](#_Toc199806401)

[2.7 Проектирование базы данных 52](#_Toc199806402)

[2.8 Разработка приложения 55](#_Toc199806403)

[2.8.1 Авторизация 56](#_Toc199806404)

[2.8.3 Генерация накладных в формате Word-документа 59](#_Toc199806405)

[2.8.4 Структура хранения и визуализации изображений 61](#_Toc199806406)

[2.8.5 Личный кабинет пользователя 63](#_Toc199806407)

[2.8.5.1 Добавление товаров в корзину 64](#_Toc199806408)

[2.8.6 Личный кабинет сотрудника 66](#_Toc199806409)

[2.8.6.1 Управление товарами 68](#_Toc199806410)

[2.8.6.2 Формирование накладных 69](#_Toc199806411)

[2.8.6.3 Фильтрация и поиск товаров 70](#_Toc199806412)

[2.8.7 Личный кабинет администратора 71](#_Toc199806413)

[2.8.7.1 Управление пользователями 72](#_Toc199806414)

[2.8.7.1 Управление товарами 74](#_Toc199806415)

## **СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ**

ПО – программное обеспечение.

ИС – информационная система.

LINQ – синтаксис языка запросов, напоминающего SQL

WPF – Windows Presentation Foundation

## **ВВЕДЕНИЕ**

В настоящее время, когда развиваются технологии и повышается уровень осведомленности в области дизайна пространства, а также его влияния на чувства пользователя, увеличивается интерес у соответствующей аудитории к различным способам освещения и влиянию освещения на пространство. Освещение играет одну из ключевых ролей восприятие человеком окружающей его среды, особенно в пространствах, полагающихся на атмосферность, располагающую или оказывающих конкретное влияние на пользователей. К пространствам, которым требуется такое освещение можно отнести культурные институции, такие как музеи и галереи. Но востребованность освещения у таких культурных институций подразумевает требование к большому и разнообразному выбору, ведь освещение взаимодействует не только с человеком или предметами, которые легко заменить. Некоторые экспонаты, с которыми освещение должно постоянно находится в контакте, невероятно чувствительны к свету. Поэтому к каждому типу необходимо выбирать освещение, которое должно соответствовать замыслу архитекторов пространства, дизайнеров и кураторов.

Как и раньше, многие компании, как и раньше, стремятся к развитию и расширению своего влияния на рынке, всячески повышая востребованность своих продуктов и расширяя ассортимент. Особенно сейчас требуется это делать быстро, адаптируя свои продукты под нужды новых сегментов. К одному из таких направлений расширения компании можно отнести поставку и внедрение профессиональных систем освещения для музеев и галерей. Всё это требует учёта многих специфических требований именно для этой области, таких как: уровень освещенности, цветопередачи, отсутствия ультрафиолетового и инфракрасного излучения, а также соответствие нормам, установленном в области музейного дела.

Заказчик ранее был ориентирован на сферу коммерческой и бытовой подсветки. Однако, с недавнего времени планирует выйти на новый рынок и предложить новые решения для учреждений культуры. Разработанный проект ориентирован на изучение и адаптацию ассортимента компании под требования и нужды галерей, а также на подготовку рекомендаций по организации эффективной и качественной системы освещения в выставочных залах.

Объектом исследования является управление деятельностью компании, занимающейся поставкой светового оборудования.

Предметом исследования выступает разработка информационной системы для автоматизации бизнес-процессов предприятия.

Целью проекта является разработка информационной системы по автоматизации процесса бронирования и продаж.

Актуальность темы определяется стремлением компаний расширить сферу своей деятельности, внедряя профессиональные решения освещения для чувствительных экспозиций.

Сформулированы следующие задачи:

1. Проанализировать особенности освещения в музейных и галерейных пространствах;
2. Изучить существующие стандарты и нормы освещённости для чувствительных экспонатов;
3. Осуществить выбор технологий для разработки;
4. Спроектировать архитектуру системы;
5. Разработать базу данных и пользовательский интерфейс;
6. Реализовать функциональные модули системы;
7. Провести оценку экономической эффективности.

**1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И СУЩЕСТВУЮЩИХ РЕШЕНИЙ**

* 1. ****Описание и анализ предметной области****

**Влияние освещения на сохранность исторических артефактов всегда являлась важной темой и насущной проблемой для всех исторических музеях в чьих недрах хранится любая графика. Ведь второй важнейшей задачей музея, кроме сохранения исторических артефактов является знакомство и просвещение пользователей с ними. Поэтому музеи вынуждены выбирать между стремлением сохранить работу в её лучшем состоянии или все же представить её зрителям. Таким образом, главным фактором, который влияет на выбор экспонирования каки-либо объектов становится освещение.**

**Разные типы исторических артефактов изготовлены из разных материалов, и каждый из них обладает разной степенью устойчивости к воздействию света, особенно в его самых агрессивных состояниях, а это в ультрафиолетовые и инфракрасные спектры.[7] Предметы сделаны в основном из органических материалов, которые довольно чувствительны к фотохимическим реакциям. Эти реакции, которые могут начинаться, даже при быстром воздействии световых потоков достаточной интенсивности.[10]**

**После изучения информации по взаимодействию материалов со светом, стала понятна важность выбора системы освещения и строгие требования к оборудованию и его настройке. Значимость этой части в процессе формирования новой экспозиции нельзя преуменьшить, это невероятно важный аспект в музейном пространстве. Музеи и галереи чаще всего используют несколько типов источников света, у каждого из которых есть достоинства и ограничения, в которые также некоторые риски. Уже почти повсеместно применяются галогенные и металлогалогенные лампы, люминесцентные источники, а также светодиодные системы (LED**[[1]](#footnote-1)**).**

**Первые, галогенные лампы**[[2]](#footnote-2) **отличаются от других тем, что они передают насыщенные и теплые оттенки с хорошей цветопередачей. Но также они излучают интенсивный инфракрасный спектр, реакция с предметами с которым может привести к перегреву экспонатов. Для того, чтобы избежать риска повреждений, такие лампы требуют дополнительной защиты в виде специальных фильтрах и их размещении на безопасном расстоянии. Без этих предосторожностей использование галогенов может быть нежелательно рискованным.[9]**

**Металлогалогенные лампы – ещё один из распространённых вариантов. Эти лампы обеспечивают более яркое освещение, чем галогеновые, которое более стабильное и обычно с нейтральной цветовой температурой. Эти параметры выделяют их среди других ламп и делают удобными и востребованными для использования в просторных залах и, например, в павильонах с высокими потолками. Однако, как и в случае с гало генами, ультрафиолетовое излучение требует фильтрации. Без неё есть риск воздействия на чувствительные материалы.**

**Люминесцентные лампы обеспечивают широкую зону равномерного освещения, но обладают низкой стабильностью. Они мерцают, у них плохая цветопередача и ко всему прочему их сложно точно настраивать – всё это снижает их востребованность на музейном рынке. Поэтому такие лампы применяются всё реже, потому что сейчас уже существуют более надёжные и менее рискованные альтернативы.**

**Наиболее универсальным и способным к адаптации решением сейчас считаются LED-источники. Светодиоды почти не выделяют тепла, они обладают длительным сроком службы и могут настраиваться по параметрам цветовой температуры – от холодного спектра, что примерно 6500 К до тёплого, а это около 2700 К. Холодный свет часто используется архитекторами пространства для того, чтобы подчеркнуть форму и текстуру металлических объектов, а тёплый усиливает глубину цвета древесины, текстиля, живописи. Но все же даже с LED-технологиями важно избегать чрезмерной яркости и направленного света, особенно в отношении уязвимых материалов. Любое пренебрежение к деталям может повредить исторический артефакт.**

**Необходимость продумывать небольшие детали повышает вероятность ошибки из-за человеческого фактора, а это ведет к росту потребности в автоматизации и также для избавления служащих хотя бы частично от документирования. А автоматизация позволяет минимизировать влияние человеческого фактора, ускорить документооборот и точность ведения записей. Поэтому в музейной практике активно внедряются системы, позволяющие управлять светотехническим оборудованием централизованно, отслеживать историю изменений и обеспечивать безопасность экспонатов в режиме реального времени. Именно поэтому, было решено создать систему и для поставщика светового оборудования для упрощения взаимодействия с музеями и четко видимой истории покупок и продаж.**

**И конечно, важно принимать во внимание, что свет в музее – это не только средство освещения, но и инструмент создания атмосферы. Он влияет на восприятие зрителя, также он способен подчеркнуть уникальные черты объекта или же скрыть его уникальные характеристики, сделав трудным для восприятия. Поэтому проектирование систем освещения требует междисциплинарного подхода. Здесь важны и инженерные знания, и понимание художественного контекста, и юридическая подготовка.[6]**

**Грамотно выстроенное освещение в музее – это результат совместной работы многих специалистов. Только так можно снизить риск неизбежных повреждений исторических артефактов и обеспечить пользователю возможность взаимодействия с ними.**

* 1. ****Анализ взаимодействия музеев и галерей с поставщиками****

Нужно четкое и детальное понимания фотохимических реакций, особенностей спектрального излучения[[3]](#footnote-3) и допустимой длительности светового воздействия. Без этих базовых знаний для поставщика будет невозможно правильно экспонировать экспонаты в музее. Любые ошибки или, кажется, небольшие отклонения в настройке параметров освещения могут создать риски нанесения вреда экспонатам, которые иногда могут стать необратимыми. Степень повреждения будет зависеть от степени устойчивости экспоната к световому излучению и от степени силы самого света. Когда каждый предмет коллекции обладает уникальными физико-химическими свойствами, даже незначительное превышение интенсивности света, либо неправильная траектория его направления способна нанести ущерб.

Поэтому поставщик уже не может быть просто звеном из логистической цепочки, которая доставляет оборудование. Ему следует знать всю подноготную процессов и уметь наглядно показать, и рассказать про свойства и влияние своего светового оборудования, потому что эти знания ценятся в сотрудничестве с музеем намного больше, чем в коммерческих предприятиях других направленностей. Иногда такой поставщик может претендовать на звание партнера организации, если его вклад достаточно значителен, и он берет на себя большую ответственность за соответствие условий экспонирования принятым нормативам и требованиям музейной консервации. Так что роль поставщика в такой связи потребует от него инженерных знаний и понимания архитектурной среды, в которой осуществляется размещение оборудования.

К пониманию архитектурной среды следует отнести внедрение нового оборудования, которое поставщик должен адаптировать под пространственную организацию зала, архитектурные элементы и сценарии освещения конкретных объектов. Поставщик еще обязан помнить и учитывать возможности фильтрации ультрафиолетового и инфракрасного излучения, он должен знать правила для оптимального распределения светового потока, понимать индекс цветопередачи и остальные параметры, критичные для сохранения экспонатов.[8]

Структура музеев и галерей плотно связана с подробной системой внешнего взаимодействия. Такие учреждения зависят от партнёров и поставщиков, которые помогают обеспечивать не просто снабжение, а поддержание условий, определяющих качество экспонирования и, в конечном итоге, безопасность коллекции. Если взглянуть с такой стороны, то светотехнические компании в таких случаях действуют в статусе интеграторов, их решения напрямую влияют на атмосферу залов, восприятие объектов публикой и долговечность самих экспонатов.

В итоге получается комплексный подход, в котором важно учитывать не только параметры оборудования, но и последствия его влияния на среду. Успешное сотрудничество требует профессионализма, точности и глубокого погружения в специфику музейной деятельности. Но за этим следует большая ответственность, потому что ошибка здесь – это не просто материальный ущерб, а в худшем случае утрата культурной ценности, которую восстановить невозможно.

* 1. ****Целевые пользователи и их потребности****

**До начала установления полного партнёрства с поставщиком музейные учреждения сперва инициируют специальные встречи, где проводят консультации и технические сессии с потенциальными поставщиками. Это происходит потому, что большая часть музеев не располагает ни необходимыми ресурсами, ни специалистами требуемого профиля в своем штабе для самостоятельной разработки, в данном случае, эффективной схемы освещения. Так что желание с самого начала руководства музея получить наиболее полную, объективную и, главное, независимую информацию, зависит от отсутствие специального оборудования и компетентного инженерного персонала.**

**Так как у музея нет в штабе подходящих сотрудников, часто привлекаются независимые эксперты, специалисты в требуемой области, здесь это бы были в области реставрации и консервации, также потребовались бы знания профильных инженеров, которые обладают достаточными компетенциями в проведении светотехнического аудита, кроме того, поставщики обычно тоже участвуют, хотя обычно и не в тесном контакте с нанятыми специалистами. Такой подход к выбору партнеров позволяет сразу понизить риск связи с плохим поставщиком и также понизится вероятность использования неподходящего оборудования, которое может сказаться на сохранности коллекции.**

**В современных условиях поставщик, работающий с музеями, уже не воспринимается исключительно как продавец. Это – многопрофильный партнёр, сочетающий в себе роли консультанта, эксперта по музейному освещению и технического исполнителя. От такого поставщика ожидается высокий уровень квалификации, это включает требуемое понимание физических характеристик световых потоков, понимание, как работают фотохимических реакций, возникающих при воздействии света на разные виды материалов и не менее важные знания в области норм и рекомендаций таких организаций, как ICOM и CIE**[[4]](#footnote-4)**.**

**Музейные учреждения, стремящиеся к устойчивому развитию, нередко акцентируют внимание на энергоэффективности, снижении уровня вредных излучений, а также на степени автоматизации светотехнических решений. Поэтому всё чаще предъявляются требования к экологической и технологической составляющей оборудования, что сказывается на критериях выбора поставщиков.**

**Если структурировать рынок поставщиков, с которыми взаимодействуют музеи, можно выделить три основные группы. Первая – это производители профессионального оборудования, заточенного под нужды культурных учреждений. Их решения часто стоят дороже, но характеризуются высочайшим качеством, стабильной работой и поддержкой сложных систем управления светом. Вторая категория – дистрибьюторы и крупные мультибрендовые поставщики, предлагающие широкий ассортимент. Они предоставляют больше гибкости по цене, но музей в этом случае берёт на себя задачу фильтрации и подбора конкретных моделей под свои задачи. И, наконец, третья группа – локальные подрядчики и компании по монтажу оборудования. Их преимущество в быстрой реакции на технические инциденты, но ограниченный доступ к узкопрофильной продукции может быть препятствием при реализации сложных проектов.**

**Таким образом, результативное взаимодействие между музеем и поставщиком начинается задолго до физической поставки светильников. Это длительный процесс согласования, технической экспертизы и консультаций, цель которого не просто осветить пространство, а сохранить в нём культурное наследие.**

* 1. ****Анализ аналогов и существующих решений****

**Для создания приложения было решено сперва провести сравнение с аналогами или похожими, но созданные для более обширных целей программами. Было изучено три похожих аналога, это: 1С: Управление торговлей, SAP Business One,** Microsoft Dynamics 365 Business Central. Было изучено, как сильно они похожи друг на друга функционала, можно ли их адаптировать, какие из них подходят лучше всего, а какие не подходят вовсе.

**Надо принять во внимание то, что хоть компания лишь формирует свою модель развития и работы на рынке музейного освещения, особенно важно учесть потенциальное развитие. Поэтому проектируемая система создается максимально доступно гибко, ориентируясь на будущие масштабируемость, интеграцию с внешними платформами и возможную автоматизацию поставок.**

***1С: Управление торговлей***

**1С: Управление торговлей – «это современный инструмент для повышения эффективности бизнеса торгового предприятия. Программа позволяет в комплексе автоматизировать задачи оперативного и управленческого учета, анализа и планирования торговых операций, обеспечивая тем самым эффективное управление современным торговым предприятием». Система рассчитана на предприятия малого, среднего и крупного бизнеса, она служит для обеспечивания всего цикла продаж компании, то есть от формирования заказов до анализа эффективности торговой деятельности.**

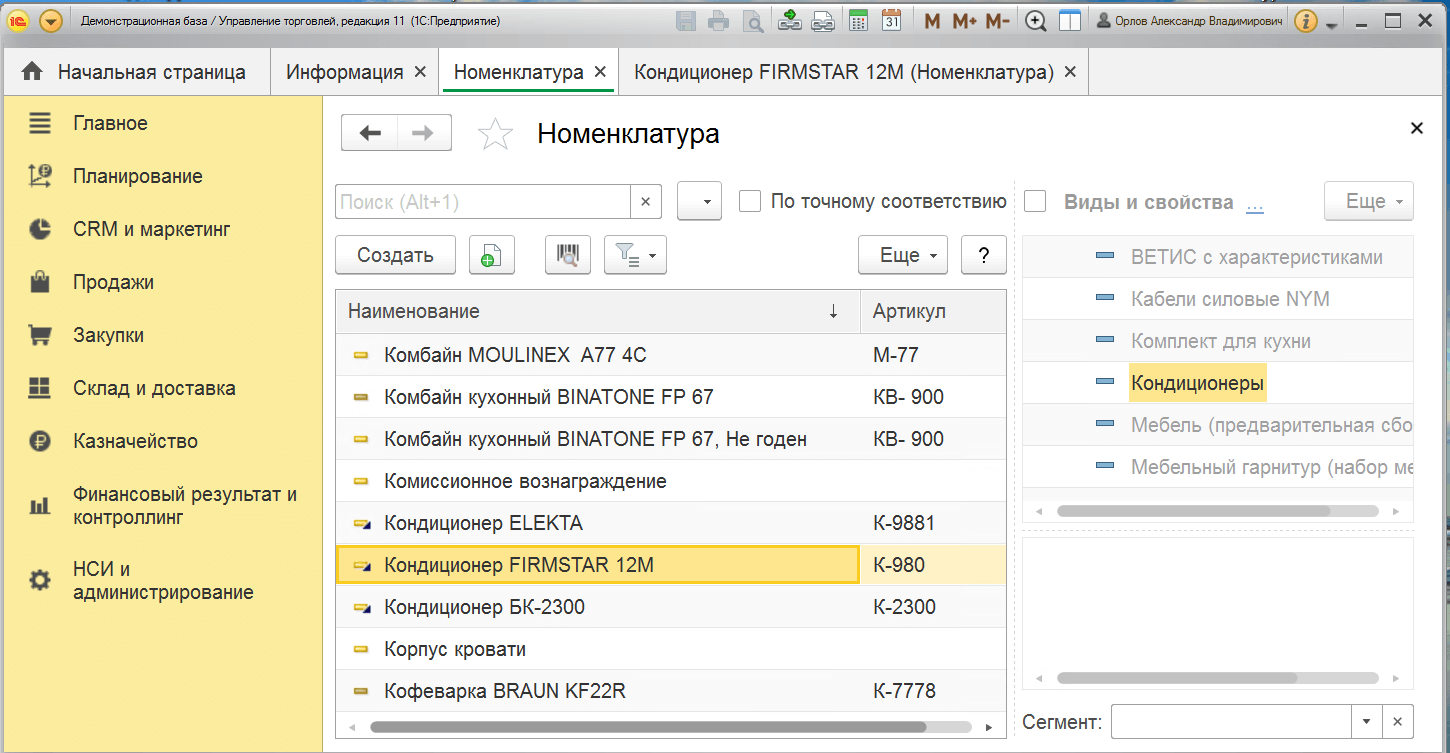
**Однако главной сложностью при использовании 1С в условиях специализированного бизнеса является необходимость привлечения программистов 1С для изменения структуры базы данных, создания специализированных форм документов, фильтрации по нестандартным техническим параметрам, например, углу рассеивания или уровню защиты IP, и это значительно увеличивает стоимость и сроки внедрения.**

**Преимущества:**

1. **Система обладает широким функционалом, позволяющим охватывать все этапы торговой деятельности, например, закупки, продажи, складской учёт, логистику, аналитику;**
2. **Пользователи могут адаптировать конфигурацию под специфику отрасли или внутренние бизнес-процессы;**
3. **Система позволяет интегрирование с другими программами;**
4. **Система поддерживает многоскладской учет и упрощает работу предприятий с подробной логистикой;**
5. **Система предоставляет множество инструментов для принятия управленческих решений на основе свода данных;**
6. **Система способствует снижению количества ошибок и ускорению обработки операций;**
7. **Система позволяет своевременно учитывать изменения в законодательстве и бизнес-среде.**

**Недостатки:**

1. **В начале требуется обучение персонала для эффективной работы с системой;**
2. **Изменения программы требуют привлечения квалифицированных специалистов;**
3. **Качество внедрения во многом зависит от исполнителя проекта;**
4. **При работе с большими объёмами данных могут потребоваться серверные мощности;**
5. **Системе требуется постоянное сопровождение в виде регулярной поддержки и обновлений;**
6. **Перенос данных и логики может быть трудоёмким;**
7. **Для пользователей без опыта работы с 1С интерфейс неудобен в изучении обилием информации.**

****

**Рисунок 1 – Интерфейс «1С: Управление торговлей»**

*SAP Business One[[5]](#footnote-5)*

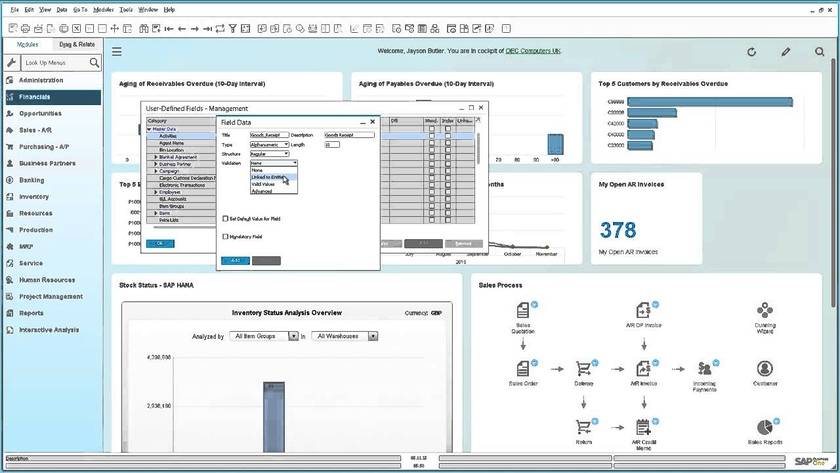
SAP Business One – «это доступное ERP-решение для управления малыми и средними предприятиями, охватывающее все: от бухгалтерского учета и финансов, закупок, инвентаризации, продаж и взаимоотношений с клиентами (CRM) до отчетности и аналитики». Разработанная компанией SAP, эта платформа объединяет модули для управления финансами, продажами, закупками, запасами, производством и взаимоотношениями с клиентами. Эта программа ориентирована на повышение прозрачности процессов, минимизацию ручного труда и обеспечение единой базы данных для принятия обоснованных управленческих решений.

Преимущества:

1. Эта система объединяет все ключевые модули, например, финансы, CRM, склад, закупки в единой базе данных;
2. Она позволяет адаптировать интерфейс и бизнес-логику под отраслевые и индивидуальные требования;
3. Она эффективна для компаний с распределённой сетью подразделений и представительств;
4. Широкий выбор языков и валют делает систему удобной для международного бизнеса;
5. Система включает встроенные средства для анализа данных, построения графиков и KPI-отчётов;
6. Возможна синхронизация с более масштабными решениями;
7. Система обеспечивает удалённую работу с данными в режиме реального времени.

**Недостатки:**

1. **Высокая стоимость лицензирования и поддержки может оказаться существенной для малого бизнеса;**
2. **Сложность внедрения требует привлечения специалистов и значительных временных ресурсов на настройку и обучение;**
3. **Для стабильной работы необходимы соответствующие серверные ресурсы и надёжные каналы связи;**
4. **Не все функции доступны в мобильной версии, некоторые процессы можно выполнять только через десктопный интерфейс;**
5. **Сопровождение и расширение функционала часто требует дополнительных затрат.**



**Рисунок 2 – Интерфейс «SAP Business One»**

*Microsoft Dynamics 365 Business Central [[6]](#footnote-6)*

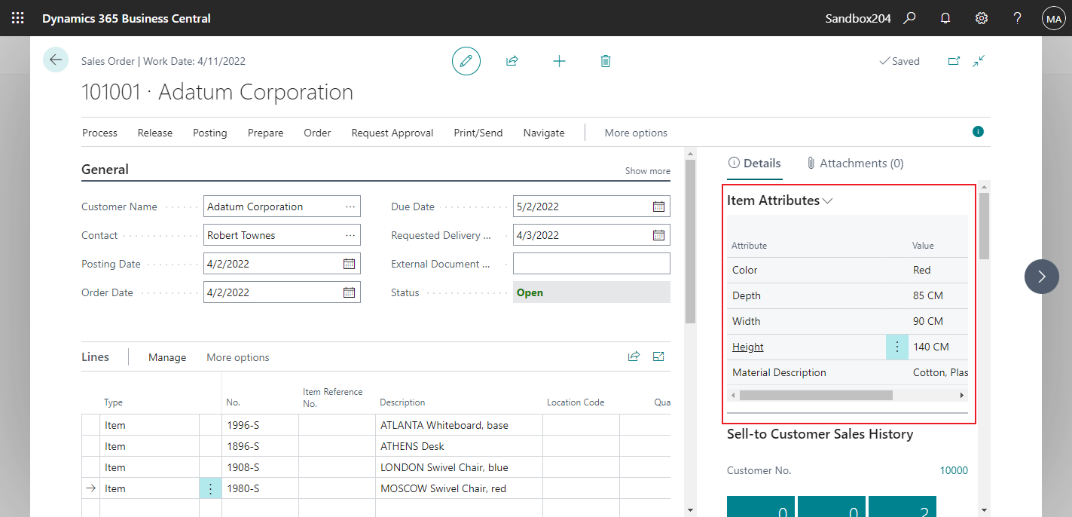
Microsoft Dynamics 365 Business Central – это «интегрированная система управления предприятием для среднего и малого бизнеса, поставляемая компанией Microsoft в линейке продуктов Microsoft Dynamics, объединяющих бизнес-решения ERP и CRM». Эта программа включает в себя модули управления финансами, поставками, проектами, продажами, обслуживанием клиентов и другими направлениями. Интеграция с экосистемой Microsoft, включая Office 365 и Power BI, делает Business Central эффективным инструментом для компаний, стремящихся к цифровизации и централизованному управлению.

Преимущества:

1. Интеграция с Microsoft 365 – бесшовное взаимодействие с Outlook, Excel, Teams и другими офисными сервисами;
2. Поддерживает удалённую работу и масштабирование без необходимости собственной серверной инфраструктуры;
3. Система адаптируется под особенности отрасли и конкретной компании;
4. Получение регулярных новых обновлений от Microsoft;
5. Соответствие международным стандартам безопасности и защита конфиденциальной информации;
6. Возможность подключения к внешним сервисам и платформам без серьёзных доработок.

Недостатки:

1. Стоимость лицензий и внедрения;
2. Требуется тщательное планирование перехода на другие платформы;
3. Большая часть функций реализуется через облако, что требует постоянного интернет-соединения;
4. Интерфейс и логика работы требуют предварительной подготовки пользователей;
5. Итоговая эффективность во многом зависит от квалификации интегратора;
6. Глубокая доработка и добавление вариативности требует разработки на AL-языке.

******

**Рисунок 3 – Интерфейс «Microsoft Dynamics 365 Business Central»**

**Для изучения существующих решений, более ориентированных на базы данных оборудования для освещения, далее будут сравниваться платформы-посредники между коллекционерами и художниками.**

**Таблица 1 – Сравнение аналогов**

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **1С: Управление торговлей** | **SAP Business One** | **Microsoft Dynamics 365 Business Central** |
| **Целевая аудитория** | Компании розничной и оптовой торговли. | **Компании малого и среднего бизнеса.** | Организации малого и среднего масштаба. |
| **Географический охват** | Продукт можно использовать по всей территории РФ и в странах СНГ | Используется в более чем 170 странах мира. | Это международное решение, адаптированное под более чем 40 стран, включая Россию. |
| **Технологическая база** | Поддерживает клиент-серверную архитектуру, может использовать СУБД MS SQL Server, PostgreSQL и др. | Основана на архитектуре клиента-сервера, может работать как в облаке, так и на локальных серверах; поддерживает SQL Server, SAP HANA. | Построен на облачной платформе Microsoft Azure, поддерживает работу как в облаке, так и в локальной версии (on-premises). |
| **Пользовательский интерфейс** | Интерфейс в стиле 1С:Предприятие с возможностью настройки. | Интуитивно понятный графический интерфейс с поддержкой ролей. | Современный веб-интерфейс, оптимизированный под ПК, планшеты и мобильные устройства |
| **Масштабируемость** | Система может быть адаптирована как для малого предприятия, так и для крупного. | Позволяет добавлять пользователей, модули и расширения по мере роста компании. | Позволяет увеличивать функциональность по мере роста бизнеса. |
| **Обновляемость** | Регулярные релизы и обновления конфигурации. | Регулярные обновления с новыми функциями и исправлениями. | Автоматические обновления в облачной версии 2 раза в год. |
| **Стоимость внедрения** | Стоимость может варьироваться от нескольких десятков до сотен тысяч рублей. | Внедрение варьируется от нескольких сотен тысяч до миллионов рублей. | Стоимость зависит от числа пользователей, модуляции, объёма доработок и способа развертывания. |
| **Поддержка и обучение** | Официальные партнёры «1С» предоставляют техническую поддержку, проводят обучение пользователей и администраторов, доступна обширная база документации. | Официальные партнёры SAP предлагают широкий спектр обучающих программ, онлайн-курсов, документации и технической поддержки с SLA. | Microsoft и партнёрская сеть предлагают онлайн-курсы, техническую документацию, обучающие видео и сопровождение внедрения с персональной поддержкой. |

* 1. ****Требования к функционалу приложения****

**Система должна содержать три личных кабинета: пользователь, сотрудник и администратор. Далее рассмотрим функциональные возможности каждого из них.**

**Для входа в систему необходимо пройти процедуру авторизации, используя логин и пароль.**

***Личный кабинет пользователя***

1. **Предполагает следующие функции:**
2. **Просмотр товаров с фильтрацией;**

**Формирование корзины.**

***Личный кабинет сотрудника***

1. **Предполагает следующие функции:**
2. **Просмотр товаров с фильтрацией;**
3. **Редактирование товаров;**
4. **Добавление товаров;**
5. **Формирование накладных с выгрузкой в текстовый документ.**

***Личный кабинет администратора***

1. **Предполагает следующие функции:**
2. **Просмотр товаров с фильтрацией;**
3. **Добавление товаров;**
4. **Формирование накладных с выгрузкой в текстовый документ;**
5. **Добавление пользователей в систему.**

**Интерфейс приложения должен быть ориентирован на оперативную работу. Поэтому основной акцент в визуальном оформлении сделан на упрощённую навигацию и фильтрацию – по техническим характеристикам, брендам, ценовому уровню и другим параметрам. Все это позволяет находить нужные модели оборудования за считанные секунды.**

* 1. ****Технологический стек****
     1. ****Выбор системы управления базой данных****

**Всего есть несколько систем управления базами данных: Реляционные СУБД,** Документные **СУБД, Ключ-значение СУБД, Графовые СУБД, Колонко-ориентированные СУБД, Объектно-ориентированные СУБД.**

**Реляционные СУБД – это СУБД, где данные хранятся, как таблицы и их структура определена пользователем вручную. К таким относятся: Microsoft SQL Server, MySQL, PostgreSQL, Oracle Database.**

Документные СУБД – это СУБД, основанные на хранении баз данных, как документов. Примером такой СУБД может быть MongoDB. Ключ-Значения СУБД – СУБД с простой структурой, где первичные ключи ассоциируются со значениями, например Amazon DynamoDB.

Графовые СУБД [[7]](#footnote-7)– это такие СУБД, которые обычно используются в социальных сетях для данных с множественными взаимосвязями, самая знаменитая Amazon Neptune. Объектно-ориентированные СУБД поддерживают объектные структуры данных. Колонко-ориентированные СУБД оптимизированы для аналитических запросов. Для создания приложения для галереи больше всего подходят, либо реляционные СУБД, либо Документные СУБД.

**Таблица 2** – **Сравнение реляционных СУБД и документных СУБД**

|  |  |  |
| --- | --- | --- |
| **Характеристика** | **Реляционные СУБД** | **Документные СУБД** |
| **Хранение данных** | Таблицы с заранее заданной схемой. | Документы (обычно JSON). |
| **Гибкость структуры** | Строгая, заранее заданная схема. | Высокая, структура документов может различаться |
| **Сложность запросов** | Используют SQL. | Менее сложный, поддерживает NoSQL |
| **Примеры** | Microsoft SQL Server, MySQL. | MongoDB, Couchbase |
| **Применение** | Финансы, ERP, CRM. | Big Data, приложения с быстрым изменением структуры данных |
| **Масштабируемость** | Вертикальная | Горизонтальная |
| **Производительность** | Высокая для транзакций | Высокая для больших объемов неструктурированных данных |

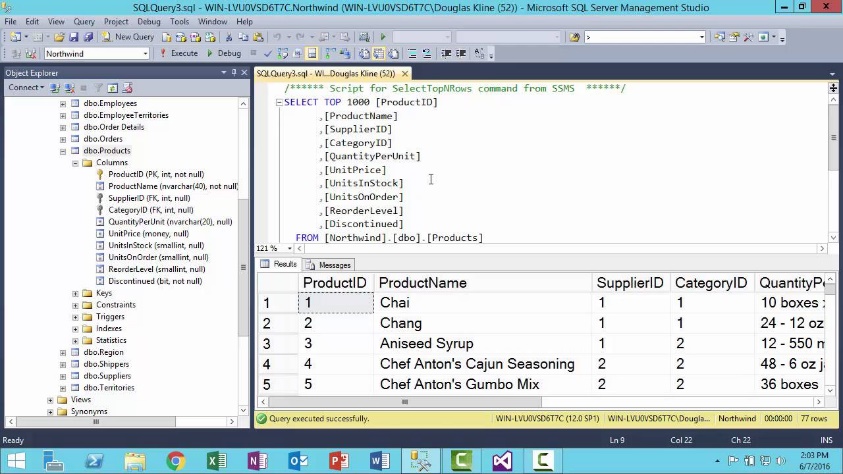
**Было отобрано несколько СУБД для выбора из них самой подходящей для этого проекта.**

**Таблица 3** – **Сравнение СУБД**

|  |  |  |  |
| --- | --- | --- | --- |
| **База данных** | **Тип** | **Преимущества** | **Недостатки** |
| **Microsoft SQL Server** | Реляционная | Высокая производительность, интеграция с продуктами Microsoft, продвинутые функции безопасности | Высокая стоимость лицензий, завязка на экосистему Microsoft |
| **MySQL** | Реляционная | Бесплатная (Community Edition), простота использования, широкая поддержка сообществом | Ограниченная функциональность в бесплатной версии, менее эффективна для масштабируемых систем |
| **PostgreSQL** | Реляционная | Открытый исходный код, высокая совместимость с другими системами, поддержка сложных запросов | Более сложное администрирование и настройка по сравнению с другими СУБД |
| **Oracle Database** | Реляционная | Высокая производительность, поддержка крупных корпоративных проектов | Высокая стоимость лицензий, сложность освоения и администрирования |
| **MongoDB** | Документная | Гибкость при работе с неструктурированными данными, масштабируемость | Ограничения в сложных запросах, необходимость адаптации для работы с реляционными системами |
| **SQLite** | Реляционная | Простота и портативность, встроенная в приложения | Ограничения в масштабируемости, не подходит для многопользовательских системы |

***Microsoft SQL Server***

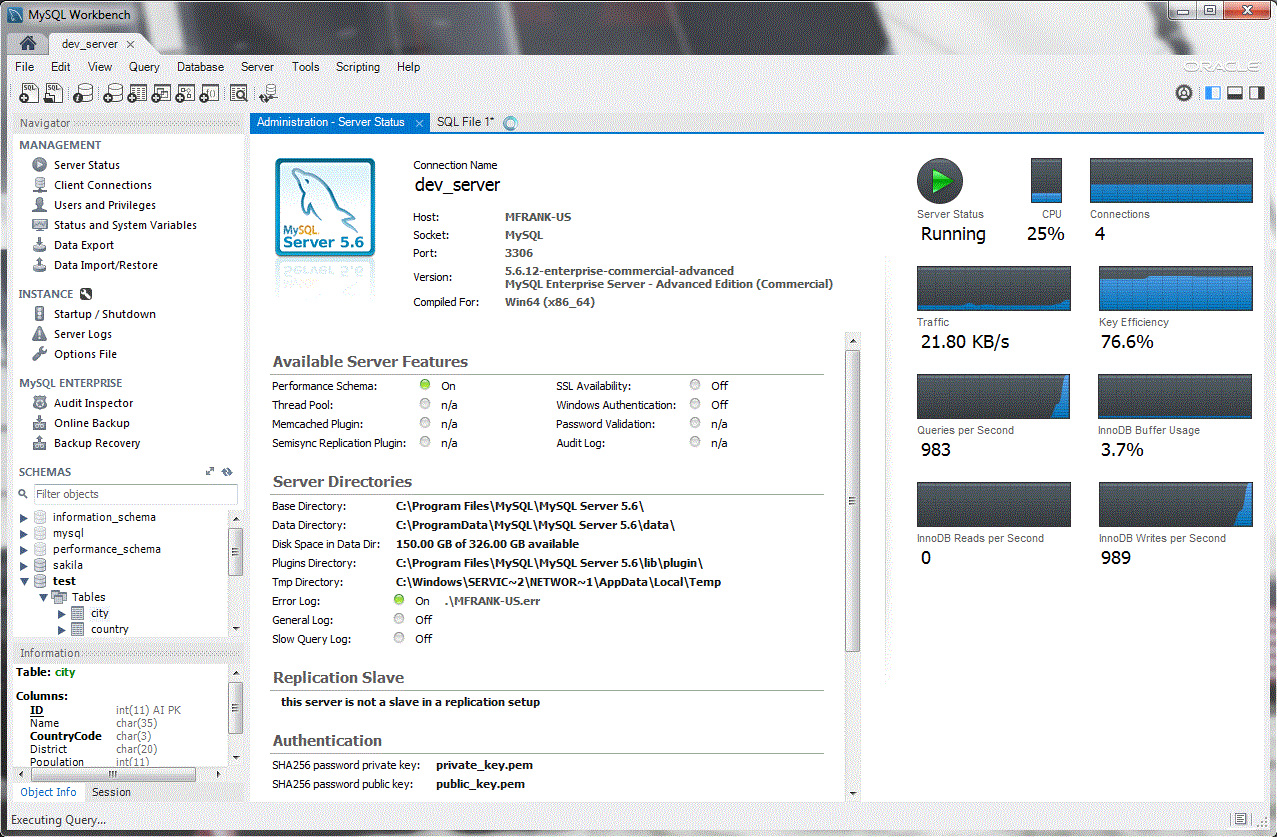
**Microsoft SQL Server – это реляционная система управления базами данных. Она довольно мощна, потому что обеспечивает высокую скорость работы и расширенные возможности защиты. И, как остальные продукты Microsoft, легко интегрируется с продуктами Microsoft, это делает её удобной для корпоративного использования. А бесплатная версия SQL Server Express позволяет разрабатывать приложения без финансовых затрат на лицензии.**



**Рисунок 4 – Интерфейс Microsoft SQL Server**

***MySQL***

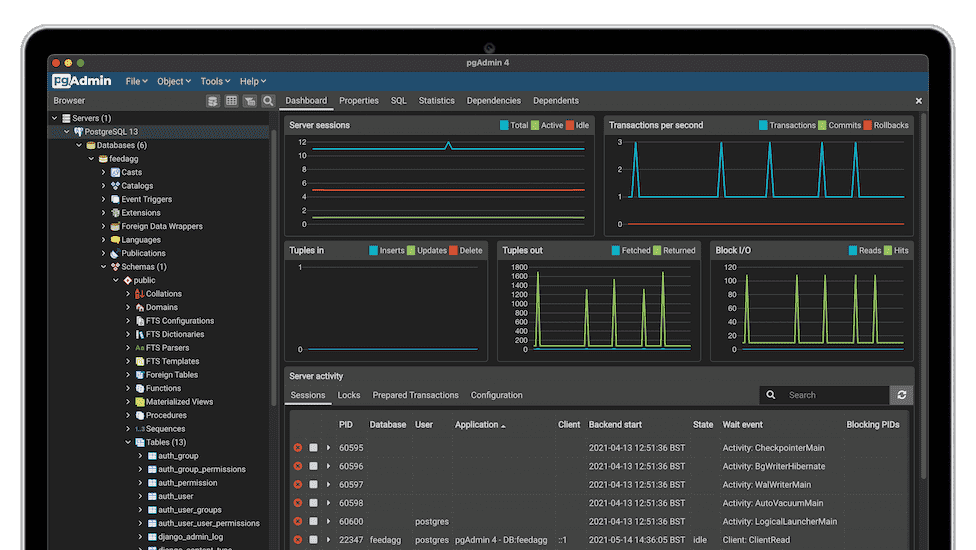
**MySQL – популярная реляционная СУБД с открытым кодом, её легко освоить и она поддерживает высокую скорость обработки данных. Её поддерживает большое сообщество разработчиков, и она широко применяется в веб-разработке, особенно совместно с PHP. Однако бесплатный вариант имеет определённые ограничения по сравнению с коммерческими версиями.**



**Рисунок 5 – Интерфейс MySQL**

***PostgreSQL***

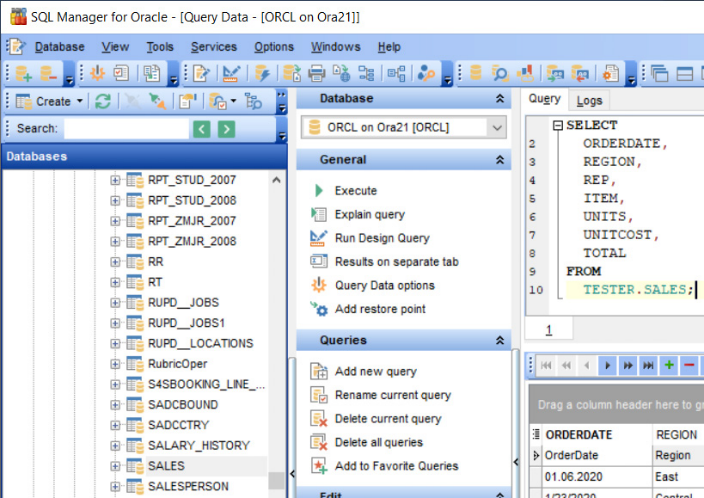
**PostgreSQL – это СУБД с открытым исходным кодом, поддерживающая сложные SQL-запросы и возможность расширения функционала с помощью дополнительных модулей. PostgreSQL может поддерживать высокую стабильность и надёжность, но требует более детальной настройки и управления. PostgreSQL часто применяется в аналитических системах и крупных корпоративных проектах.**



**Рисунок 6 – Интерфейс PostgreSQL**

***Oracle Database***

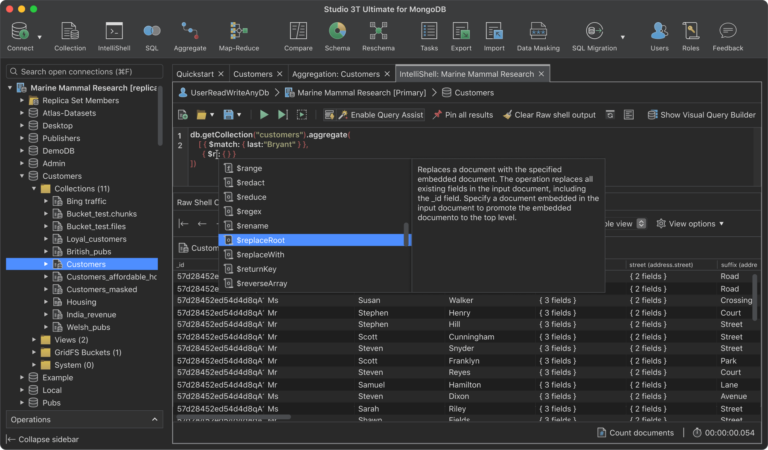
**Oracle Database – это реляционная база данных, предназначенная для работы с крупными объёмами информации в корпоративных средах. Она поддерживает масштабирование, сложные транзакционные операции и отличается высокой надёжностью. Однако её использование требует значительных затрат на лицензирование и хорошей подготовки специалистов.**



**Рисунок 7 – Интерфейс Oracle Database**

***MongoDB***

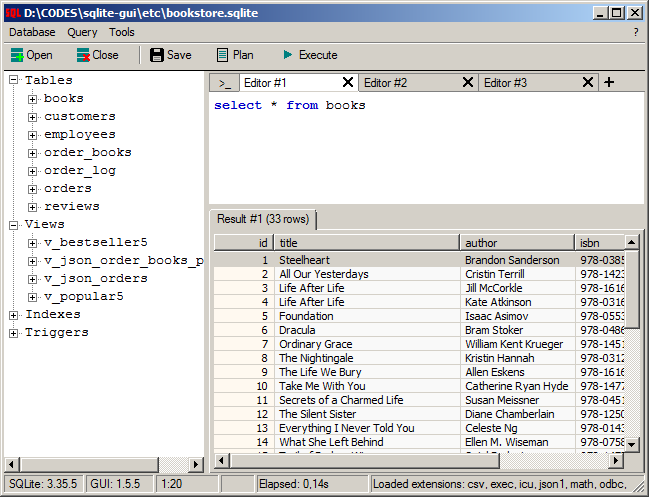
**MongoDB – это система управления базами данных, ориентированная на работу с документами, где информация хранится в формате JSON-подобных объектов. PostgreSQL хорошо подходит для веб-приложений, требующих гибкой структуры данных и лёгкого масштабирования. Однако обработка сложных запросов в этой системе менее эффективна, чем в классических реляционных системах.**



**Рисунок 8 – Интерфейс MongoDB**

***SQLite***

**SQLite – это тоже реляционная база данных, не требующая отдельного сервера и идеально подходящая для мобильных и настольных приложений. Её основными преимуществами являются простота в использовании и отсутствие необходимости в сложной конфигурации. Однако из-за ограниченной производительности она не подходит для многопользовательских проектов с высокой нагрузкой.**



**Рисунок 9 – Интерфейс SQLite**

**Была выбрана Microsoft SQL Server. В ней оптимально сочетание цены и функционала, версия Microsoft SQL Server бесплатна, что помогает избежать трат на лицензию, при этом она предоставляет все базовые необходимые функции. Проект создается на базе WPF и Microsoft SQL Server идеально с ней интегрируется, это упрощает разработку и поддержку приложения. Для базы данных внутреннего пользования не требуется сложной и дорогой СУБД. В Microsoft SQL Server отсутствует сложная система и она предоставляет удобный инструмент для управления базой данных без лишних затрат и долгого обучения. Также, когда потребуется расширить функционал или изменить масштаб проекта, можно просто перейти на более мощную версию Microsoft SQL Server, например SQL Server,** **сохранив при этом все данные и настройки.** **Microsoft SQL Server может обеспечить высокую производительность, особенно при работе с транзакциями и она подходит для надежного хранения данных.**

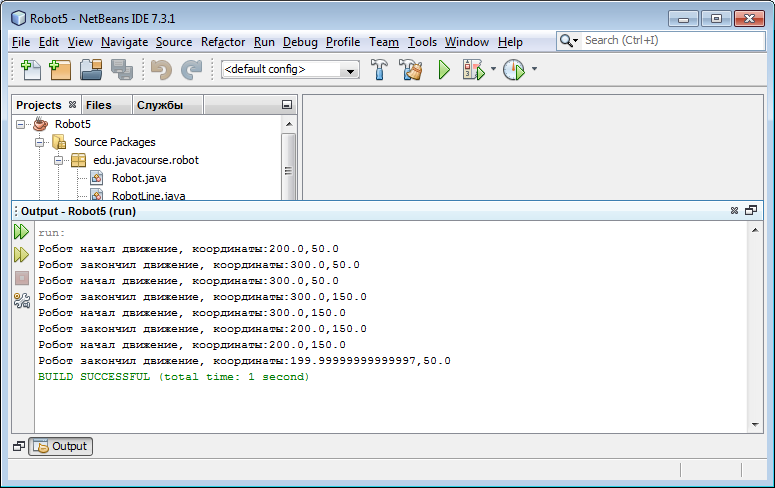
**Приложение для внутреннего, но более удобного мониторинга продуктов для заказчика** – **это небольшой проект с относительно простыми требованиями к базе данных. Использование Microsoft SQL Server позволит сосредоточиться на разработке приложения, не беспокоясь о сложностях настройки и администрирования базы данных.**

* + 1. ****Выбор языка программирования****

**При создании приложения для внутренней работы поставщика с ассортиментом товаров важно выбрать подходящий язык программирования с удобным доступом к базе данных.**

***Java***

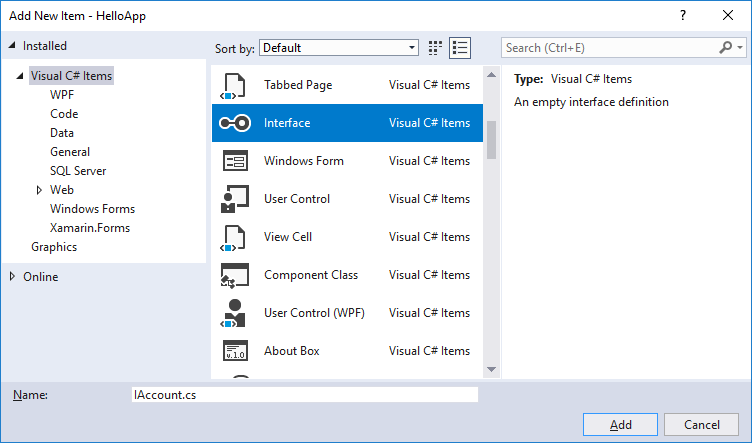
**Программы, написанные на языке Java работают в виртуальной машине (JVM**[[8]](#footnote-8)**) и это обеспечивает возможность запуска на различных ОС. У Java сформировано широкое сообщество разработчиков и большое количеством библиотек для облегчения создания корпоративных приложений. Но при этом, для создания нативных**[[9]](#footnote-9) **настольных приложений под Windows его возможности ограничены, а производительность может уступать компилируемым языкам.**



**Рисунок 10 – Синтаксис языка программирования Java**

***C#***

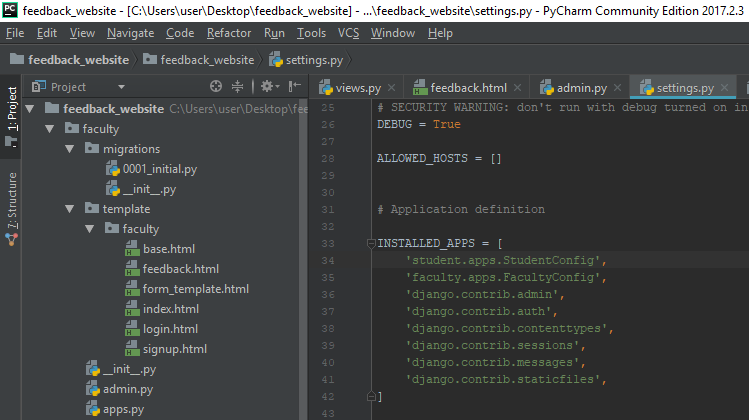
**C# – это язык высокого уровня, обладающий интеграцией с платформой .NET. Его главные преимущества в автоматическом управлении памятью, строгой типизации и возможности создания масштабируемых и надежных приложений. Интегрированная среда разработки Visual Studio содержит большой набор инструментов для отладки, рефакторинга**[[10]](#footnote-10) **и визуального проектирования, это сильно ускоряет процесс создания и тестирования ПО.**

****

**Рисунок** **11 – Синтаксис языка программирования C#**

***Python***

**Python прост и лаконичен, этим он больше всего и известен. Этот язык используется для недолгой в разработке реализации каких-либо прототипов и создания функционала, не требующего слишком высоких вычислительных ресурсов. Его огромная экосистема модулей и библиотек позволяет решать разнообразные задачи. Но несмотря на это, язык не всегда способен обеспечить необходимую для компаний скорость работы и качественную поддержку сложных пользовательских интерфейсов, что особенно видно в приложениях, ориентированных на обработку больших массивов данных и требующих взаимодействия с ОС на глубоком уровне Windows.**



**Рисунок 12 – Синтаксис языка программирования Python**

**Таблица 5 – Сравнение языков программирования**

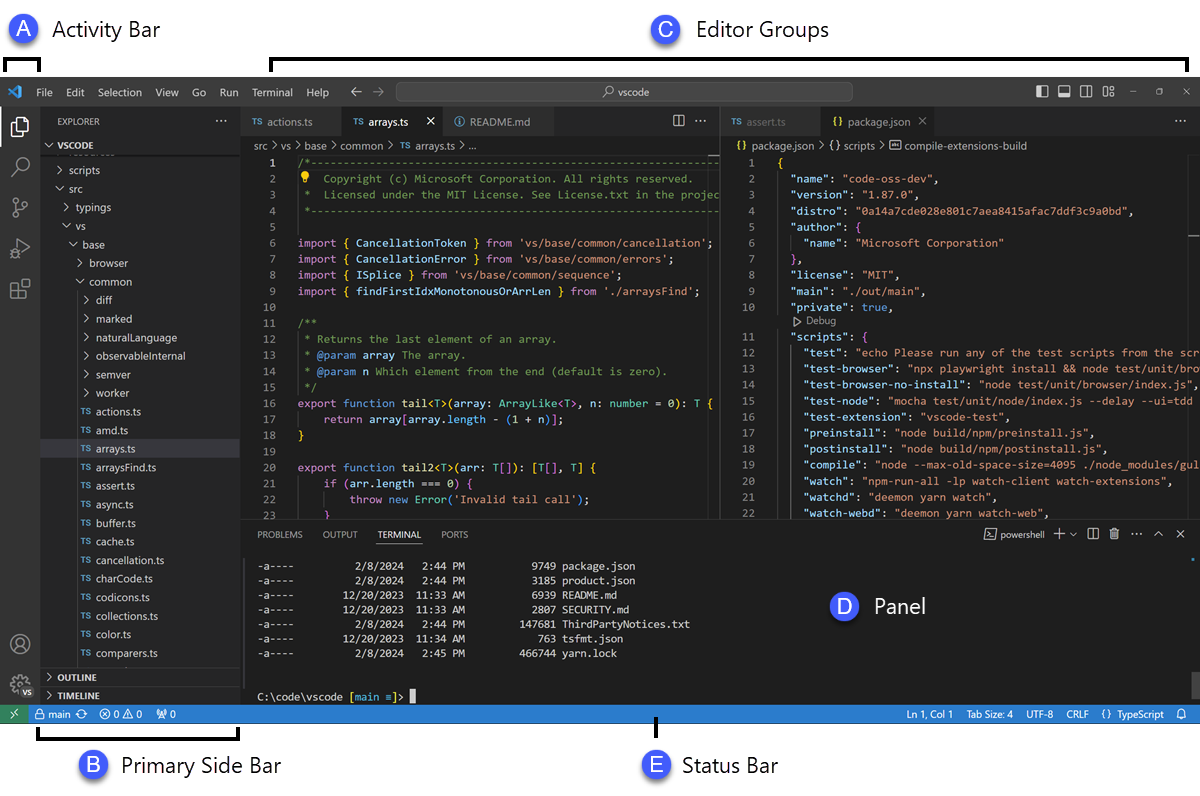
|  |  |  |  |
| --- | --- | --- | --- |
| **Показатель** | **Java** | **C#** | **Python** |
| **Производительность** | **Хорошая производительность, но уступает компилируемым языкам в ресурсоемких приложениях.** | **Отличная производительность для бизнес-приложений, оптимизированных под Windows.** | **Производительность ниже, особенно в вычислительно интенсивных сценариях.** |
| **Управление памятью** | **Автоматическое благодаря сборщику мусора, но может приводить к непредсказуемым задержкам.** | **Автоматическое управление памятью снижает вероятность ошибок, связанных с ресурсами.** | **Автоматизированное, однако накладные расходы могут сказываться на скорости исполнения.** |
| **Сложность синтаксиса** | **Объектно-ориентированный, синтаксис понятен, однако может быть громоздким в крупных проектах.** | **Современный и лаконичный синтаксис, упрощающий разработку, и снижает порог вхождения.** | **Очень простой и удобный для быстрого прототипирования, но недостаточно строгий для больших систем.** |
| **Поддержка UI** | **Не ориентирован непосредственно на нативный Windows UI, требует дополнительных библиотек.** | WPF предоставляет мощный и гибкий набор инструментов для создания современных UI под Windows. | **Возможна реализация через сторонние библиотеки, но функциональность ограничена.** |
| **Инструменты разработки** | Поддерживается различными средами (Eclipse, NetBeans), однако интеграция с Windows не на высоте. | Глубокая интеграция с Visual Studio, что обеспечивает богатый функционал для отладки и дизайна. | Широкий выбор редакторов и IDE, но отсутствует глубокая интеграция с платформой Windows. |
| **Кроссплатформенность** | **Благодаря JVM работает на разных платформах, но нативный вид на Windows может быть утрачено.** | Основная направленность – Windows, что является плюсом для приложений с WPF и узконаправленным функционалом. | Универсальный язык с кроссплатформенной поддержкой, но не всегда соответствует требованиям настольных приложений. |
| **Сообщество и экосистема** | **Огромное сообщество, множество готовых решений и библиотек, но сложность настройки может быть помехой.** | Активное сообщество .NET, отличная поддержка от Microsoft, широкая экосистема компонентов и интеграция с корпоративными решениями. | Большое сообщество, отличная документация, однако недостаточная строгость для крупных корпоративных решений. |

**Из-за оптимального сочетания производительности C#, её удобства разработки и поддержки современных технологий, этот язык с использованием WPF в Visual Studio подошел для текущих задач больше всего.**

* + 1. ****Выбор среды разработки****

***Visual Studio***

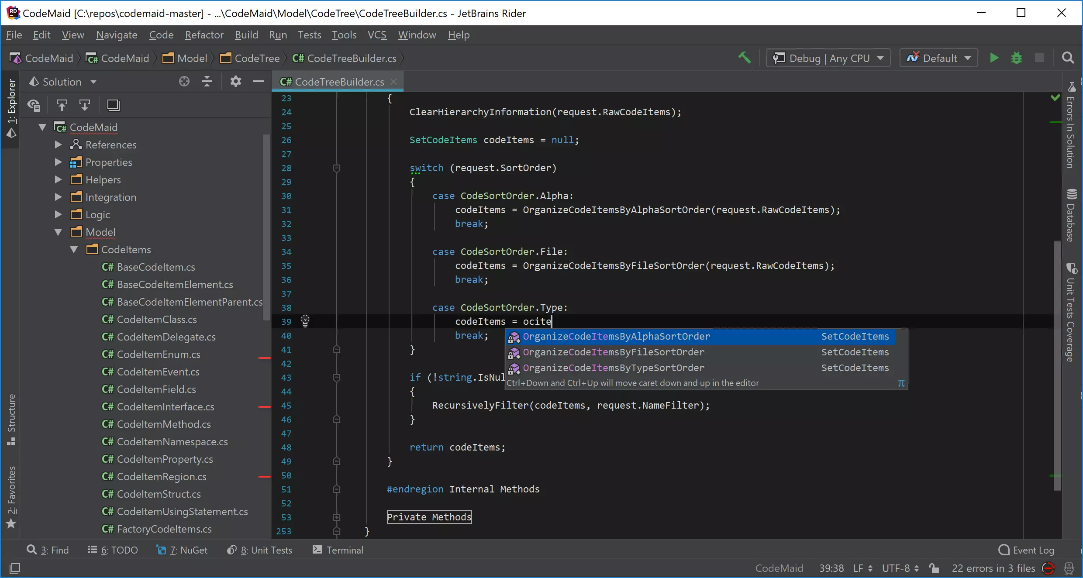
**Visual Studio – это мощная интегрированная среда разработки (IDE), разработанная Microsoft, поддерживающая множество языков программирования, включая C#, C++ и Python. Она располагает многими инструментами для отладки, рефакторинга и автоматизированного тестирования, что делает её удобной для профессиональных разработчиков. Благодаря интеграции с Azure и GitHub, она отлично подходит для создания корпоративных и облачных приложений.**



**Рисунок 13 – Интерфейс Visual Studio**

***JetBrains Rider***

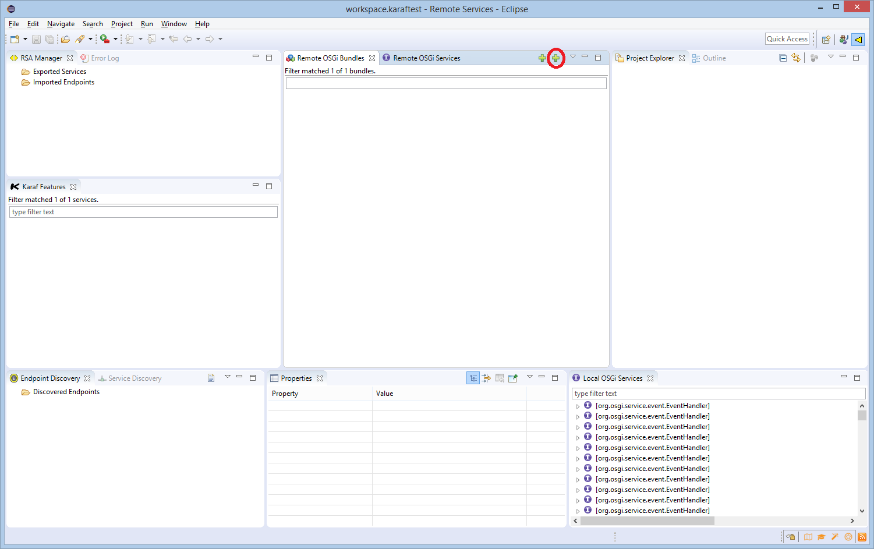
**JetBrains Rider – это современная среда разработки, ориентированная на работу с языками платформы .NET, включая C# и F#. Она отличается высокой скоростью работы и интеллектуальной системой подсказок, а также глубокой интеграцией с инструментами JetBrains, такими как ReSharper. Rider поддерживает кроссплатформенную разработку и является хорошим выбором для тех, кто использует альтернативные операционные системы, помимо Windows.**



**Рисунок 14 – Интерфейс JetBrains Rider**

***Eclipse***

**Eclipse – это бесплатная и кроссплатформенная среда разработки с открытым исходным кодом, поддерживающая Java, C++ и другие языки. Она обладает гибкостью за счёт системы плагинов, позволяя адаптировать функционал под конкретные задачи. Однако её производительность может зависеть от конфигурации системы, а интерфейс требует некоторого привыкания.**



**Рисунок 15 – Синтаксис языка программирования Eclipse**

**Была выбрана Visual Studio, так как она предоставляет наиболее удобные инструменты для работы с C# и интеграции с Microsoft SQL Server. Её мощные средства отладки, встроенная поддержка WPF и возможность расширения делают её оптимальным выбором для данного проекта.**

* 1. ****Особенности конфиденциальности и безопасности данных****

Решение создать роли пользователям влияет на защитные функции приложения, а не только на организационные. В приложении предусмотрено три роли: администратор, сотрудник и пользователь. Последней ролью пользуются те, кому удобнее онлайн посмотреть ассортимент поставщика. Это обычно представители музеев или других учреждений, заинтересованных в подборе оборудования. Администратор предоставлено большего всего контроля в приложении: он вносит и обновляет данные, создаёт учётные записи, а также модерирует изменения. Сотрудники же обладают правом редактирования товарных позиций и подачи информации на утверждение. Внешние пользователи, то есть пользователи, ограничены только просмотром каталога, что позволяет обеспечить безопасный и односторонний доступ к данным.

Наряду с защитой доступа, в приложении предусмотрен механизм шифрования всей критически важной информации. Он используется как для пользовательских данных, так и истории перемещений оборудования, договоров и любых связанных транзакций. Шифрование применяется на этапе хранения и в процессе передачи данных, что делает систему устойчивой к попыткам перехвата или несанкционированного доступа, также на это влияет и отсутствие возможности завести аккаунт, его может создать только администратор.

Особое внимание уделено гибкости поиска и фильтрации. Специалистам важно быстро находить нужные модели оборудования: типу освещения, бренду, техническим характеристикам. Благодаря подробной фильтрации удаётся сократить время на подбор и минимизировать необходимость лишнего взаимодействия между сотрудниками, это ценно особенно в условиях интенсивного процесса работы.

Автоматическое резервное копирование базы данных осуществляется на регулярной основе. Это важная мера предосторожности: в случае сбоев или внешних угроз вся информация будет восстановлена без потерь. Такой подход повышает устойчивость системы и гарантирует, что данные о поставках, заказах и клиентах сохранятся в актуальном виде.

* 1. ****Недостатки существующих методов управления оборудованием****

**В структуре работы организаций, плотно связанных с учетом каких-либо предметов всё ещё встречается практика ручного ведения учёта товаров, особенно в небольших компаниях. Такие решения повышают вероятность самых распространенных ошибок, которые случаются из-за человеческого фактора, к ним относятся ошибки в записях базы данных. Последствиями становятся искажения информации о характеристиках оборудования или потерянные сведения о его передвижении. Такие неточности затрудняют обслуживание клиентов, снижают скорость реагирования и могут нанести вред деловой репутации организации или даже хуже.**

**Ещё одной значительной проблемой является отсутствие централизованной базы данных. Это означает, что информация о товарах может храниться в разрозненных файлах электронных таблиц, отдельных папках или даже передаваться через мессенджеры или флеш-накопители. Это сильно тормозит и усложняет совместную работу, делает невозможным одновременный доступ нескольких сотрудников к актуальным данным, какую-то информацию сотрудники вынуждены искать у других сотрудников, а также увеличивает риск потери информации. Кроме того, формирование отчётов, анализ продаж и подготовка коммерческих предложений усложняются из-за увеличения времени коммуникации между сотрудниками.**

**Сотрудники вынуждены тратить значительное время на рутинные операции, особенно когда приходится просматривать обширный каталог продукции без подручных средств в виде фильтров или продвинутого поиска. При расширении ассортимента и работе с новыми клиентами, например, музеями или специализированными учреждениями, предъявляющими жёсткие требования к техническим характеристикам светового оборудования, такая задержка становится особенно ощутимой и серьёзно тормозит общий темп работы организации.**

**Чтобы избавиться от этих проблем, можно внедрить в процесс работы автоматизированную систему управления данными. Общая база данных оборудования поможет применить к предметам стандартизацию: добавить процедуру учёта, что должно снизить риск ошибок, ускорит сбор и анализ статистики, а также ускорит работу в коллективе.**

1. **ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ**

****2.1 Архитектура приложения****

Создаваемое программное решение – это настольное приложение на базе WPF, разработанное для упрощения и автоматизации ключевых процессов во внутренней работе в компании-поставщике осветительного оборудования. Оно обеспечивает учет осветительного оборудования, и оно особенно необходимо для большого ассортимента товаров, то есть светильников и других необходимых комплектующих. Также оно поможет фиксировать и обрабатывать заказы от музейных и выставочных учреждений, а также распределение ролей с различными уровнями доступа станет необходимым контролем для персонала с учётом их должностных полномочий. При проектировании архитектуре задумывалось создать приложение способное к расширению без особых сложностей и легкое в сопровождении, а при необходимости оно быстро адаптировалось под новые требования.

Основа пользовательской части создана на Windows Presentation Foundation. Использование WPF помогает создать гибкий, отзывчивый интерфейс, который также создавался на довольно удобном языке XAML-разметки. Все элементы, кнопки, тестовые поля, выпадающие списки, связаны с данными через механизм data binding. Благодаря этому изменение состояния внутренних объектов мгновенно отражается на экране, что особенно важно при одновременном отображении большого числа позиций различной мощности.

Логика приложения написана на языке C# и технологическом стеке ADO.NET в паре с Entity Framework. Логика обеспечивает прямо настроенный доступ к базе данных MS SQL Server, а технологический стек даёт возможность работать с таблицами как с коллекциями привычных C#-объектов, тем самым ускоряя разработку и снижая вероятность ошибок. При этом связь с внешними библиотеками не ограничивается только базой данных: для автоматического формирования отчётных документов в формате Word используется Microsoft.Office.Interop.Word, что освобождает сотрудников от ручного набора и гарантирует единообразие выходных файлов.

В этом приложении было реализовано разделение прав между пользователями. Всего создано три уровня доступа: администратор, сотрудник и пользователь. Первые два получают расширенные функции: от ведения учётных записей и каталога до создания сопроводительных документов и последняя роль пользователя может только просматривать каталог и формировать корзину. Такая модель не только повышает безопасность, но и упрощает работу разрозненным группам пользователей.

Также приложение постоянно соединяется с базой данных. Изменения, созданные через приложения: удаление или добавление товара, добавление пользователей, оформление заказов сразу записывается на сервер. Это уменьшает риски потери информации и упрощает последующий анализ данных.

Архитектура разработанного приложения довольно проста, но гибка. Это приложение является инструментом для учёта и управления складом светотехнического оборудования, а также обеспечивает удобное взаимодействие с музейными и выставочными партнёрами. При дальнейшем развитии системы вполне реальна её масштабируемость от добавления новых модулей до интеграции с внешними сервисами что делает её долгосрочным и эффективным решением.

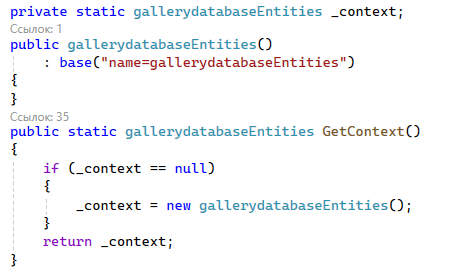
****2.2 Технология ADO.NET****

При разработке было решено сделать большой упор на использование технологии ADO.NET[[11]](#footnote-11). Она является неотъемлемой частью платформы .NET и предназначена для эффективного взаимодействия с реляционными источниками данных. В проекте, рассматриваемом здесь, в роли реляционного хранилища выступила система управления базами данных Microsoft SQL Server.

ADO.NET входит в состав библиотеки System.Data и предоставляет широкий набор средств для выполнения различных операций с данными: чтения, записи, редактирования и удаления. В рамках текущей разработки технология была использована преимущественно через объектный контекст, построенный на основе Entity Framework, что позволило сочетать удобство ORM[[12]](#footnote-12) и возможности классической архитектуры ADO.NET. В частности, при этом сохраняется традиционная работа с подключениями, командами и поставщиками данных, что является базовым принципом ADO.NET.

В качестве основного класса для взаимодействия с базой данных использовались Entities. Этот класс создается автоматически по EDMX-модели (Entity Data Model) и является наследником DbContext. Он предоставляет разработчикам доступ ко всем таблицам базы данных в форме коллекций объектов наследников DbSet<T>. Это позволяет работать с данными на уровне объектов, что упрощает написание и поддержку кода.

Для обеспечения централизованного доступа к объекту контекста и оптимизации работы с ресурсами был реализован метод GetContext(), который следует паттерну одиночки. Благодаря этому методу, во всех частях приложения удобно можно использовать один и тот же экземпляр контекста, это снижает количество избыточных подключений к базе и улучшает управление памятью и потоками.



**Рисунок 16 – Метод GetContext()**

Этот метод, например, применяется в процессе верификации учетных данных пользователя, когда через объект контекста осуществляется обращение к таблице Users. Запрос, созданный с использованием LINQ, автоматически конвертируется в SQL-код и исполняется через ADO.NET, который является провайдером для SQL Server. В этом случае формируется параметризованный объект SqlCommand, что обеспечивает безопасность и защищает от уязвимостей, связанных с SQL-инъекциями.

Для получения списка доступного светового оборудования информация точно таким же образом извлекается через коллекцию DbSet. Данный вызов запускает создание SQL-запроса SELECT и после этого результаты автоматически преобразуются в объекты модели данных. Механизм мониторинга изменений, осуществляется через ObjectStateManager, это даёт возможность редактировать поля объектов и легко сохранять их в базу. Когда сотрудник создает новый заказ, начинает формироваться экземпляр класса Order, который помещается в соответствующую коллекцию DbSet. Вызов метода SaveChanges() сохраняет новые изменения и потом конвертирует в SQL-команду INSERT INTO, которая передаётся ADO.NET и исполняется с помощью выбранного провайдера.

Entity Framework используется вместе с ADO.NET для обеспечения строгой типизации всего кода. Данная систематизация снижает вероятность появления ошибок в процессе компиляции. Параметризация всех SQL-запросов также повышает безопасность, поскольку она предотвращает использование SQL-инъекций.

ADO.NET повышает и сохраняет высокую производительность, обеспечивая выполнение выборочных запросов с учетом ленивой загрузки (lazy loading), что оптимизирует использование памяти и ресурсы сервера. Внутреннее управление состоянием объектов гарантирует правильное взаимодействие с данными без ненужных затрат.

Иногда, когда нужна более точная оптимизация или обход ограничений ORM, следует использовать ADO.NET, задействуя классы SqlConnection и SqlCommand. Это предоставляет разработчику полный контроль над выполнением запросов и способствует повышению производительности при необходимости. Применение ADO.NET в данном приложении для предоставления осветительного оборудования гарантирует эффективное, безопасное и стабильное взаимодействие с базой данных. Низкоуровневые возможности ADO.NET, применяемые вместе с высокоуровневой абстракцией Entity Framework, способствовали ускорению разработки, улучшению читаемости кода и упрощению дальнейшего сопровождения программного обеспечения.

****2.3 Язык разметки XAML****

Часть приложения, предназначенная для пользователей, разработана с применением WPF. Использование WPF даёт возможность разработать адаптируемый и отзывчивый интерфейс, который создан с помощью языка разметки XAML. Все элементы для управления приложением, включая кнопки, текстовые поля и выпадающие списки, соединены с данными из базы данных с помощью механизма привязки данных. Это обеспечивает постоянное соединение и отображение изменений состояния внутренних объектов на экране, что особенно актуально при одновременном отображении многочисленных позиций различной силы.

Благодаря этому языку разметки, в коде появляется типизация и четкое разделение визуального представления и логики приложения. Поэтому xaml лучше всего использовать для внешнего вида интерфейса, который отделяется от программного кода, написанного на C#. Такое разделение позволяет повысить структуру и читаемость кода, а также значительно облегчает поддержку и расширение проекта в дальнейшем.

XAML лучше всего подходит для декларативного описания визуальных компонентов приложения: элементов управления, их свойств, взаимного расположения и иерархии. Важной особенностью языка является возможность создания адаптивных, гибких и динамичных интерфейсов без необходимости вручную создавать объекты пользовательского интерфейса в коде C#. Это значительно ускоряет процесс разработки и облегчает внесение изменений в дизайн.

К примеру, окно авторизации (LoginPage) оформляется с помощью XAML-разметки, которая задаёт структуру окна и основные элементы взаимодействия: текстовые поля для ввода, кнопки, метки и прочие компоненты. Всё это реализуется без единой строки кода, что обеспечивает удобство и скорость прототипирования.

*Механизм привязки данных (Data Binding)*

Важной функциональной возможностью XAML является механизм привязки данных. Он широко применяется в рассматриваемом приложении для отображения и автоматического обновления информации о товарах, заказах, пользователях и т.д. Пример использования binding заключается в том, что список оборудования в пользовательском интерфейсе автоматически синхронизируется с коллекцией LampCollection в коде на C#. Это реализуется посредством паттерна MVVM (Model-View-ViewModel) либо с помощью интерфейса INotifyPropertyChanged, обеспечивающего оповещение интерфейса о произошедших изменениях.

*Архитектура пользовательского интерфейса*

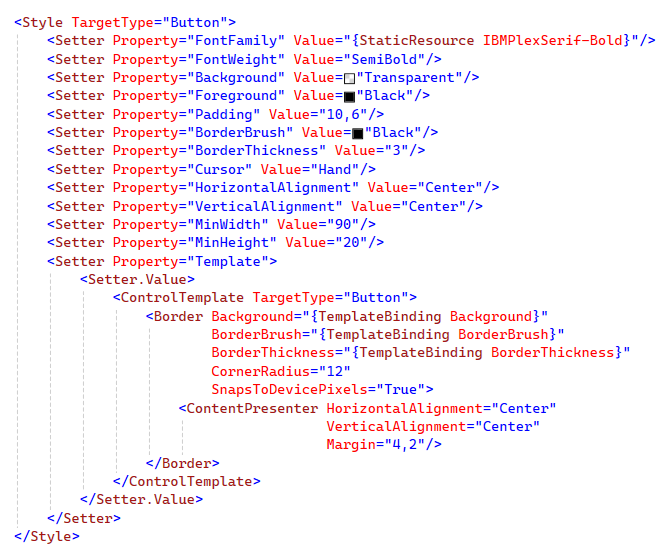
Проект включает в себя несколько ключевых страниц, которые оформлены в XAML и обеспечивают выполнение основных функций приложения/

XAML также позволяет легко привязывать обработчики событий к элементам управления. Сами обработчики реализуются в связанной с разметкой файле с расширением.xaml.cs, что позволяет держать логику отдельно от визуального представления. Такой подход способствует поддерживаемости и модульности кода.

Применение языка XAML в данном проекте обеспечило высокую скорость разработки клиентской части, улучшило удобство взаимодействия с визуальными элементами, позволило создать единый стиль оформления и повысило расширяемость интерфейса. Благодаря декларативному подходу и встроенной поддержке привязки данных интерфейс стал более адаптивным, понятным и простым в сопровождении на протяжении всего жизненного цикла приложения.

XAML предоставляет мощные инструменты для единообразного оформления элементов управления. В проекте используются ресурсы, определённые в <ResourceDictionary>, и стили, которые применяются к кнопкам, спискам, заголовкам и другим элементам интерфейса. Это помогает создавать визуально целостный и профессионально выглядящий дизайн без повторения кода.

Например, стиль кнопок можно определить следующим образом (см. рис. 17).



**Рисунок 17 – Стили кнопок**

Такой стиль применяется ко всем кнопкам в приложении, что избавляет от необходимости вручную задавать свойства каждому элементу, улучшая поддержку и консистентность[[13]](#footnote-13) оформления. И, если понадобиться изменить стиль какого-то конкретного одного объекта, можно воспользоваться иерархичностью стилей в xaml и написать стили прямо в теги объекта.

****2.4 Библиотека Microsoft.Office.Interop.Word****

Основное назначение использования библиотеки Microsoft.Office.Interop.Word в проекте заключается в автоматическом создании различных сопроводительных документов, таких как коммерческие предложения, накладные и другие формы документации, необходимых для взаимодействия с клиентами, в том числе музейными и галерейными учреждениями. Ручное оформление подобной документации связано с высокой трудоёмкостью и повышенным риском ошибок, поэтому автоматизация процесса позволяет существенно снизить нагрузку на сотрудников, повысить качество и обеспечить стандартизацию выходных файлов.

В рамках задачи автоматизации формирования документации и обеспечения экспорта данных из разрабатываемой информационной системы в формат Microsoft Word была интегрирована библиотека Microsoft.Office.Interop.Word. Эта библиотека представляет собой один из компонентов набора COM-интероп-средств от компании Microsoft и обеспечивает доступ к функционалу приложения Microsoft Word через .NET API[[14]](#footnote-14), что позволяет управлять документами Word программным способом.

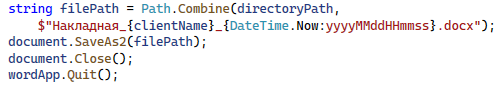
Выбор именно этой библиотеки был обусловлен её тесной интеграцией с Microsoft Office и возможностью создавать, редактировать и сохранять документы Word напрямую, без необходимости промежуточного экспорта данных в сторонние форматы, такие как RTF или HTML. Это упрощает архитектуру решения и обеспечивает более надёжную работу с файлами.

Библиотека предоставляет объектную модель, максимально приближенную к структуре пользовательского интерфейса Word. Ключевыми объектами для взаимодействия являются:

1. Application – объект, представляющий запущенное приложение Word;
2. Document – объект документа, который создаётся или открывается для редактирования;
3. Paragraph и Range – объекты, отвечающие за отдельные текстовые блоки и их форматирование;
4. Table, Row, Cell – объекты, используемые для построения и модификации таблиц внутри документа.

Для корректного использования Microsoft.Office.Interop.Word в проекте необходимо добавить в решение ссылку на соответствующий COM-компонент – Microsoft Word XX.0 Object Library, где XX – версия установленного Microsoft Word.

В приложении реализована функция автоматической генерации накладной на основании заказа, размещённого клиентом. Ниже приводится упрощённый фрагмент кода на языке C#, демонстрирующий создание нового документа с помощью Interop.Word.



**Рисунок 18 – Стили кнопок**

В процессе работы система извлекает необходимую информацию из базы данных с использованием Entity Framework (контекст).

Далее полученные данные динамически подставляются в текстовые блоки, таблицы и другие элементы документа.

Для обеспечения единообразия оформления накладных применяются готовые шаблоны документов Word. В ряде случаев создаётся базовый файл с уже подготовленными заголовками, подписями и общим стилем, в который программно добавляются таблицы с актуальными данными. Для этого осуществляется загрузка существующего .docx файла и изменение его содержимого с помощью средств Interop.

Несмотря на широкие возможности, Microsoft.Office.Interop.Word обладает некоторыми важными особенностями и ограничениями:

Для работы библиотеки на целевой машине должен быть установлен Microsoft Word соответствующей версии;

При использовании создаётся экземпляр приложения Word, который запускается в фоновом режиме, что может оказывать влияние на производительность системы;

Данная библиотека не предназначена для серверного использования и обработки документов в многопоточной среде, что накладывает ограничения на применение в веб-сервисах и высоконагруженных приложениях.

****2.5 Язык LINQ****

В процессе разработки информационной системы, предназначенной для внутреннего документооборота поставщика осветительного оборудования, важную роль сыграл язык интегрированных запросов LINQ (Language Integrated Query). Этот язык представляет собой мощный механизм, встроенный в платформу .NET, который позволяет формировать запросы к данным непосредственно на языке программирования C#. Использование LINQ обеспечивает высокую читаемость кода, строгую типизацию и тесную интеграцию запросов в объектно-ориентированную модель приложения.

Одним из ключевых преимуществ LINQ является его универсальность. Он предоставляет единый подход к работе с различными источниками данных: коллекциями в памяти, XML-документами, внешними базами данных и другими структурами. В контексте разрабатываемого программного обеспечения LINQ применялся в основном для взаимодействия с реляционной базой данных, доступ к которой осуществлялся через ORM-технологию Entity Framework

Интеграция помогла сократить объём программного кода за счёт лаконичного синтаксиса. Повысить читаемость и понятность запросов даже при наличии сложных фильтрационных и сортировочных условий. Снизить вероятность логических ошибок, благодаря типобезопасности[[15]](#footnote-15).

Упростить реализацию динамических выборок данных в зависимости от пользовательских параметров. Обеспечить удобную модульную структуру обработки данных в слоях бизнес-логики и интерфейса.

LINQ стал основным инструментом извлечения, фильтрации, группировки и подготовки данных для их последующего отображения в пользовательском интерфейсе. Это способствовало улучшению производительности и стабильности системы в целом, а также обеспечило высокий уровень согласованности между слоями данных и логики приложения.

Для реализации поиска осветительного оборудования по наименованию используется следующий LINQ-запрос:

Запрос обращается к таблице Lamp, отбирая записи, в названии которых содержится указанная пользователем подстрока. Метод Contains позволяет реализовать гибкий и интуитивно понятный поиск без необходимости вручную формировать SQL-запросы.

Функция фильтрации оборудования по типу основывается на связях между таблицами Lamp и TypeLamp. Запрос реализуется с использованием навигационного свойства и выглядит следующим образом.



**Рисунок 19 – Стили кнопок**

Таким образом осуществляется фильтрация по связанной сущности без необходимости явного объединения (join), что упрощает реализацию и поддерживает читаемость.

*Сортировка по стоимости*

Для сортировки списка оборудования по цене, в зависимости от предпочтений пользователя (по возрастанию или убыванию), используются стандартные методы OrderBy и OrderByDescending:



**Рисунок 20 – Стили кнопок**

Данный подход обеспечивает простую смену порядка сортировки без повторения избыточного кода.

Внедрение LINQ в архитектуру программного продукта позволило значительно упростить взаимодействие с базой данных, минимизировать дублирование логики выборок, повысить читаемость и сопровождаемость кода. Язык интегрированных запросов стал неотъемлемой частью логики формирования динамически изменяемых пользовательских представлений, обеспечив эффективную обработку и вывод информации. Благодаря LINQ удалось достигнуть высокого уровня согласованности между внутренними слоями системы и интерфейсной частью, что особенно важно для обеспечения надёжного документооборота и корректного отображения данных в контексте взаимодействия с клиентами и сотрудниками компании.

****2.6 Проектирование системы****

На этапе проектирования была определена общая структура системы, включающая пользовательский интерфейс, управляющие компоненты и модуль взаимодействия с базой данных. Было принято решение реализовать приложение в десктопной форме с использованием Windows Presentation Foundation (WPF), что помогло создать настраиваемый визуальный интерфейс и обеспечить высокий уровень интерактивности. Внутренняя логика системы была организована с опорой на чёткое разграничение ролей пользователей.

ПО, предназначенного для автоматизации внутренних операций компании, специализирующейся на поставках осветительного оборудования, начался сперва с тщательного анализа требований и формализации ключевых бизнес-процессов. Одной из главных целей было управлению товарным каталогом, обработке заказов и взаимодействию различных категорий пользователей. Цель заключалась в создании удобной и надёжной системы, способной учитывать особенности бизнес-логики предприятия и обеспечивать гибкость при возможных изменениях.

При проектировании рассматривалось корректное отражение предметной области в структуре базы данных. Таблицы и связи между ними разрабатывались таким образом, чтобы обеспечить непротиворечивость данных, ускорить выполнение операций выборки и оптимизировать дальнейшую обработку информации. Были задействованы механизмы первичных и внешних ключей, нормализация данных и логическое распределение сущностей по функциональным категориям.

Архитектура разработанного приложения получилась простой, но гибкой, обеспечивая учёт и управление складом светотехнического оборудования, а также удобное взаимодействие с музейными и выставочными партнёрами. При дальнейшем развитии системы предусмотрена её масштабируемость от добавления новых модулей до интеграции с внешними сервисами, что делает её долгосрочным и эффективным решением.

****2.6.1 Диаграмма активностей****

Диаграмма активностей оказалась одним из ключевых инструментов визуализации логики взаимодействия между пользователями и системой, разрабатываемой для внутреннего использования в компании, специализирующейся на поставках осветительного оборудования. Её применение позволило отразить последовательность операций, начиная от входа в систему и заканчивая формированием и подтверждением заказов, тем самым обеспечив наглядное представление архитектуры бизнес-процессов.

Одним из центральных кейсов для проектирования стала активность клиента при оформлении заказа. Диаграмма помогла структурировать процесс выбора оборудования, его добавления в корзину, ввода комментария, указания адреса и выбора варианта доставки. Каждое из этих действий сопровождалось встроенными проверками: корректность введённых данных, наличие товаров в системе, выбор доступных параметров. На схеме были чётко обозначены ветвления, условия переходов, а также возможные откаты к предыдущим этапам в случае ошибок или отсутствия нужных данных.

Проектирование активности сотрудника также было наглядно отражено при помощи этой диаграммы. Добавление и редактирование светильников, просмотр заявок, подтверждение или отклонение заказов – все этапы работы специалиста были декомпозированы на логически связанные действия. Это позволило грамотно встроить проверки прав доступа, реализовать логи[[16]](#footnote-16) и предусмотреть корректную последовательность обработки пользовательских запросов.

Особое внимание уделялось отображению альтернативных сценариев. Например, оформление заказа могло начинаться как с авторизованного, так и с нового пользователя, и в каждом случае последовательность действий отличалась. Использование условных операторов и параллельных потоков позволило избежать неоднозначности и заранее предусмотреть корректное поведение системы в различных ситуациях.

Важно, что диаграмма активностей служила не только техническим целям. Она стала мостом между разработчиками и представителями заказчика: графическое отображение процессов упростило коммуникацию и позволило обсуждать поведение приложения без необходимости вникать в программную реализацию. Это особенно ценно при работе с конечными пользователями, не имеющими технического образования.

Диаграмма активностей представлена в ПРИЛОЖЕНИИ А.

****2.6.2 Диаграмма прецедентов****

Диаграмма прецедентов сыграла важнейшую роль на начальной стадии проектирования системы, предназначенной для внутреннего использования поставщиком осветительного оборудования. Её основное назначение – отобразить основные сценарии взаимодействия между пользователями системы и её функциональными возможностями – было реализовано в полной мере. Благодаря данной диаграмме стало возможно не только структурировать поведение различных ролей, но и заранее определить границы доступа к ключевым функциям.

В диаграмме были отражены такие акторы, как администратор, сотрудник и клиент, каждый из которых взаимодействует с системой в рамках строго определённого набора сценариев. Прецеденты, соответствующие этим ролям, включали как базовые действия (просмотр каталога, регистрация, оформление заказа), так и административные функции (добавление и редактирование оборудования, формирование отчётной документации). Такое графическое разделение позволило избежать путаницы между ролями и ускорило согласование функциональных требований между участниками проекта.

На этапе обсуждения с представителями компании диаграмма оказалась особенно полезной. Она помогла визуализировать структуру обязанностей, что способствовало лучшему пониманию архитектуры системы со стороны заказчика. Благодаря этому удалось избежать формулировок, допускающих неоднозначную трактовку, и зафиксировать набор пользовательских функций с максимальной точностью. Кроме того, на основании диаграммы был составлен перечень ролей и доступных им операций, что стало основой для последующей реализации системы контроля доступа.

Диаграмма use case представлена в ПРИЛОЖЕНИИ Г.

Диаграмма также помогла выявить незадокументированные, но практически необходимые функции. В ходе анализа работы сотрудника возникла идея о добавлении функции просмотра истории изменений по заказам, которая не была предусмотрена в изначальном задании. Аналогичным образом была замечена необходимость в отображении текущего статуса заказа для клиента. Эти детали стали важным дополнением, обеспечившим полноту и завершённость пользовательского опыта.

****2.7 Проектирование базы данных****

Проектирование базы данных ИС, предназначенной для поставщика осветительного оборудования, являлось одним из ключевых этапов всего цикла разработки. От правильности этого этапа зависела не только стабильность работы приложения, но и его способность масштабироваться в будущем. На начальных стадиях особое внимание было сосредоточено на анализе предметной области, с целью выявления основных сущностей, отражающих ключевые бизнес-процессы предприятия. В результате были определены такие логические компоненты, как пользователи, товары, заказы, типы освещения, роли сотрудников, способы доставки и поставки. На базе полученного анализа сформирована концептуальная модель, ставшая основой всей структуры базы данных.

При переходе к логическому проектированию большое значение имело нормализованное представление данных. Это позволило устранить избыточность информации, исключить логические противоречия и обеспечить согласованность хранимых данных. Все связи между таблицами продумывались с учётом требований референциальной целостности[[17]](#footnote-17) и оптимизации частых операций выборки. Так, таблица пользователей связана с таблицами ролей и должностей через дополнительную таблицу рабочих данных, позволяя хранить информацию централизованно без дублирования.

Каждый этап проектирования базы данных – от построения ER-диаграммы до выбора типов данных и задания ограничений – проводился с учётом конкретных требований будущей системы. Были определены ключевые поля, предусмотрены первичные и внешние ключи, а также установлены уникальные и обязательные ограничения там, где это было необходимо. Все таблицы были сконструированы таким образом, чтобы соответствовать своей роли в архитектуре. Например, таблица заказов фиксирует не только информацию о клиенте, но и идентификатор товара, способ доставки, а также комментарии и адрес, что полностью отражает бизнес-логику обработки заказов.

ER-диаграмма представлена в ПРИЛОЖЕНИИ Б

Реализация базы данных осуществлялась с использованием технологии ADO.NET Entity Framework. В этом подходе каждая таблица базы представлена соответствующим классом, а связи между таблицами реализованы посредством навигационных свойств. Это позволило отказаться от прямой работы с SQL-запросами, сделав взаимодействие между логикой приложения и базой данных более прозрачным и удобным. Подобная реализация обеспечила надёжность, масштабируемость и высокую производительность, что особенно важно для корпоративных систем, ориентированных на внутреннее использование.

Одним из преимуществ проектной модели стало наличие элементов, закладывающих основу для дальнейшего расширения функционала. Например, поля для описания товаров и комментариев к заказам были предусмотрены заранее, что даёт возможность гибко адаптировать базу данных к меняющимся требованиям предприятия без необходимости её полной переработки. Также был реализован удобный механизм фильтрации, сортировки и поиска информации, что особенно актуально для пользователей с административным доступом. Благодаря этому удалось добиться высокой степени автоматизации рутинных операций и улучшить пользовательский опыт при работе с системой.

***Таблица Users***

Содержит сведения о зарегистрированных пользователях системы. В таблице фиксируются имя, отчество, фамилия, логин, дата рождения, адрес электронной почты, номер телефона, а также хэш[[18]](#footnote-18) пароля и соль. Поле UserType используется для связи с таблицами ролей и должностей, определяя статус и функциональные права каждого пользователя в системе. Хранение хэша и соли позволяет обеспечить безопасную авторизацию.

***Таблица WorkerInfo***

Служит связующим звеном между таблицей пользователей и справочниками ролей и должностей. Содержит поля Id, IdRole и IdPosition, реализуя связь многие-к-одному. Такая структура позволяет гибко распределять права доступа и настраивать иерархию обязанностей сотрудников.

***Таблица Role***

Представляет справочник ролей пользователей системы. Каждая запись содержит уникальный идентификатор и название роли (например, администратор, сотрудник, клиент). Используется для реализации ролевой модели безопасности и разграничения доступа к функциям приложения.

***Таблица Position***

Содержит список должностей, назначаемых пользователям. Таблица включает идентификатор и наименование позиции. Она используется совместно с таблицей ролей для более точного определения обязанностей сотрудников внутри системы.

***Таблица Order***

Фиксирует данные о заказах, оформляемых клиентами. Поля включают идентификаторы пользователя и товара, адрес доставки, тип доставки и поле для комментариев. Эта таблица отражает весь цикл работы с заказом – от его создания до передачи клиенту.

***Таблица ShippingType***

Является справочником способов доставки. Содержит наименование и при необходимости дополнительную информацию, такую как стоимость или ограничения. Используется в таблице заказов для указания способа получения товара.

***Таблица Lamp***

Содержит сведения о светильниках, доступных для заказа. Здесь хранятся данные о названии товара, его цене, мощности, кратком описании и ссылках на справочники: тип светильника, способ крепления и производитель. Таблица является центральной для товарного каталога.

***Таблица LampType***

Представляет классификатор типов светильников. Записи в таблице позволяют различать, например, потолочные, настенные или встраиваемые модели. Это необходимо для реализации фильтрации и сортировки в пользовательском интерфейсе.

***Таблица MountingType***

Содержит список возможных способов крепления светильников: подвесной, накладной, встраиваемый и т.д. Используется для точного описания характеристик товаров и подбора оборудования в зависимости от требований помещения.

***Таблица Manufacturer***

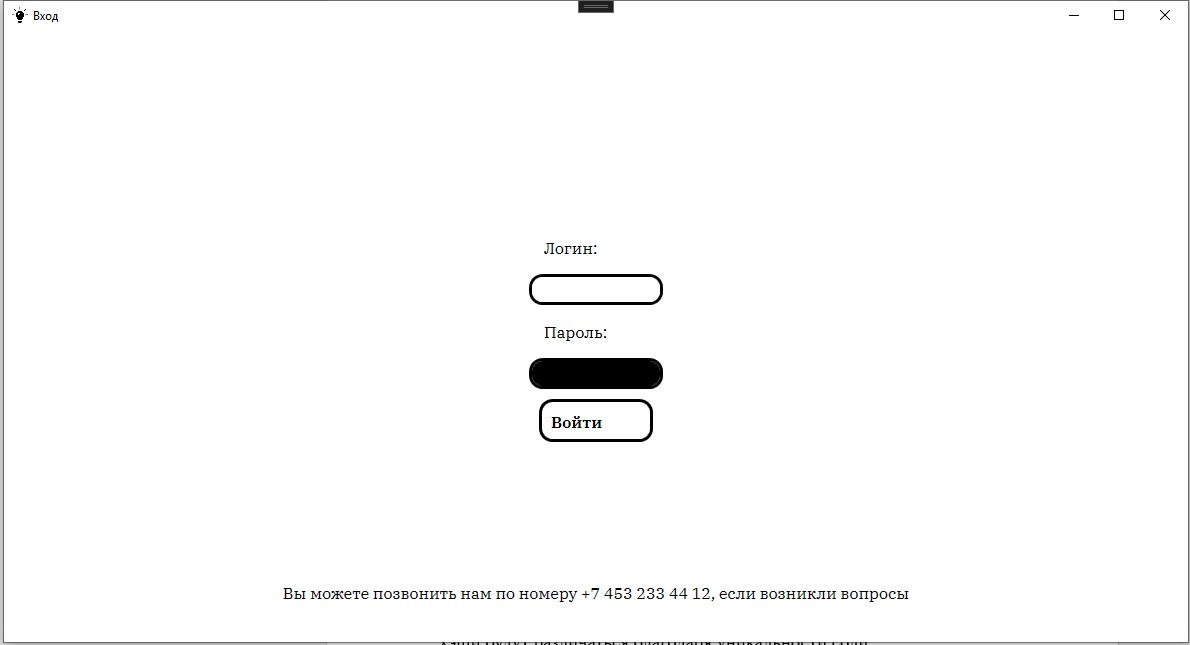
Содержит список производителей осветительного оборудования. И используется для точного более удобной фильтрации товаров.

****2.8 Разработка приложения****

Разработка приложения для поставщика осветительного оборудования представляла собой комплексный и поэтапный процесс, охватывающий создание интуитивного пользовательского интерфейса, построение устойчивого взаимодействия с базой данных и реализацию бизнес-логики, отвечающей требованиям всех категорий пользователей. Основное внимание уделялось максимальной эффективности управления заказами, ассортиментом товаров и учётными записями, что позволило добиться высокого уровня автоматизации внутренних процессов предприятия.

****2.8.1**** Авторизация

Авторизация в приложении играет ключевую роль в обеспечении безопасности и разграничении доступа пользователей в зависимости от их ролей – администратора, сотрудника или пользователя. Процесс начинается с ввода пользователем логина и пароля, которые проходят проверку на сервере. Для надежной защиты учетных данных применяется современный подход, основанный на хэшировании пароля с использованием уникальной соли для каждого пользователя. Такая реализация исключает хранение пароля в открытом виде и существенно снижает риск компрометации данных.

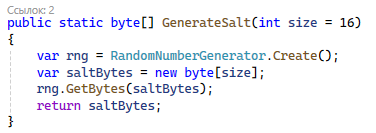


**Рисунок 21 – Авторизация**

Хранение паролей в открытом виде недопустимо – это не просто плохая практика, а реальная угроза безопасности. Поэтому в системе применяется метод добавления уникальной соли и последующего хэширования. Эта криптографически стойкая техника обеспечивает, что для каждого пользователя создаётся своя уникальная соль, генерируемая при регистрации.

При регистрации или создании пользователя вручную исходный пароль не сохраняется в базе данных напрямую. Сначала пароль преобразуется в байтовый массив, затем к нему добавляется соль – случайный набор байтов, созданный с помощью криптографически стойкого генератора из пространства System.Security.Cryptography. Важно, что для каждого пользователя создаётся новая соль. Даже если два пользователя выберут одинаковый пароль, итоговые хэши будут различаться благодаря уникальности соли.

Генерация соли представлена на рис. 22.

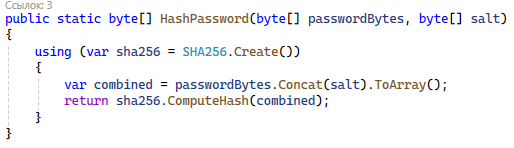


**Рисунок 22 – Генерация соли**

В этом коде создаётся массив байтов заданного размера, который заполняется случайными значениями, обеспечивая необходимый уровень энтропии. На этом этапе данные ещё не сохраняются в базе – формируется лишь материал для будущего хэша.

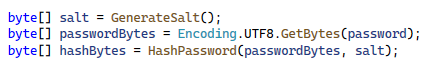
После этого к байтам пароля добавляется соль, и получившаяся последовательность подвергается хэшированию с помощью алгоритма SHA-256, который широко используется и признан надёжным. При этом даже при получении хэша злоумышленник не сможет восстановить исходный пароль.

Процесс хэширования выглядит так:



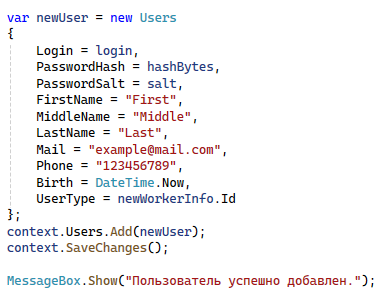
**Рисунок 23 – Хэширование**

Итоговый хэш сохраняется в базе вместе с солью. При регистрации пользователя сначала создаётся соль, затем вычисляется хэш, и только после этого данные сохраняются.



**Рисунок 24 – Итоговый хэш**

Создаётся объект пользователя с заполненными полями логина, хэша пароля, соли и дополнительной информации:



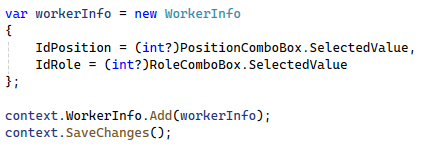
**Рисунок 25 – Пользователь**

Перед добавлением проверяется, что в системе отсутствует пользователь с таким же логином.



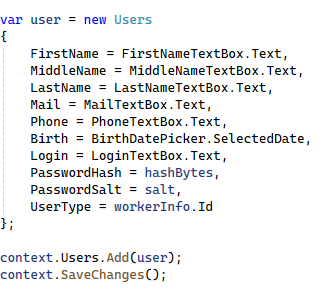
**Рисунок 25 – Проверка пользователя**

Также создаётся объект, хранящий информацию о должности и роли пользователя.



**Рисунок 26 – Создание объекта с хранимыми данными о пользователях**

После этого создаётся сам пользователь, которому присваивается соответствующая роль:



**Рисунок 27 – Создание пользователя**

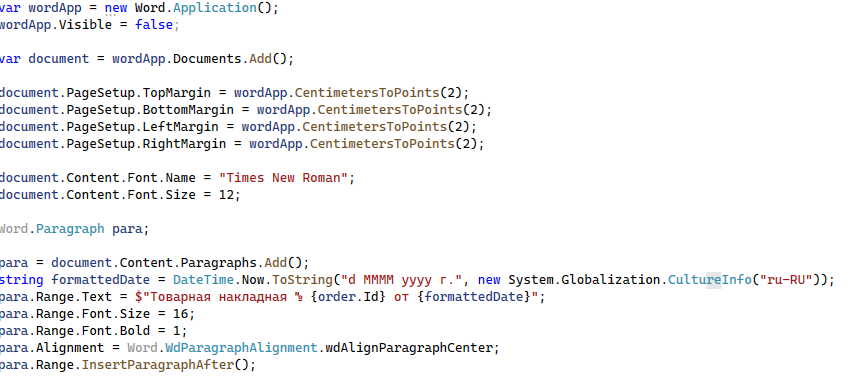
Таким образом, пользователь надёжно защищён: даже если база данных будет скомпрометирована, восстановить исходные пароли невозможно. Безопасность основана на принципе необратимости хэша и уникальности соли, что значительно усложняет подбор паролей злоумышленниками.

При последующих попытках входа в систему пароль, введённый пользователем, проходит аналогичное преобразование (хэшируется с сохранённой солью), после чего полученный результат сравнивается с сохранённым в базе хэшем. При совпадении пользователю предоставляется доступ с учетом его роли, что гарантирует правильное разграничение прав и защиту данных.

****2.8.3**** Генерация накладных в формате Word-документа

Генерация накладных в формате Word-документа является важным элементом функционала приложения для поставщика осветительного оборудования, так как автоматизирует процесс оформления документов, значительно сокращая время на подготовку отчетной и сопроводительной документации. Этот процесс предусматривает извлечение данных о заказах из базы, включая информацию о клиентах, перечне товаров, их количестве, ценах, а также условиях доставки и дополнительных комментариях. Для создания документов используется технология работы с шаблонами в формате .docx, что позволяет заранее подготовить стандартный макет накладной с заполнителями, которые при генерации заменяются актуальными данными из заказа. Такой подход обеспечивает единообразие оформления и соблюдение корпоративных стандартов.

В качестве инструментария для работы с документами применяется библиотека Microsoft.Office.Interop.Word, позволяющая программно создавать, изменять и сохранять Word-файлы без участия пользователя. В ходе работы приложение открывает шаблон, подставляет в нужные места сведения о заказе, форматирует текст, таблицы и другие элементы документа, после чего сохраняет готовый файл для последующей печати или отправки. Данный метод исключает ручной ввод данных и снижает вероятность ошибок, обеспечивая автоматизацию и надежность документооборота. Внедрение генерации накладных в формате Word-документа позволяет сотрудникам эффективно формировать сопроводительную документацию, что повышает качество обслуживания клиентов и ускоряет процесс выполнения заказов. Ниже приведён пример кода, демонстрирующий основные этапы работы с библиотекой Microsoft.Office.Interop.Word для создания и сохранения накладной.



**Рисунок 28 – Создание документа Word**

Полный программный код представлен в ПРИЛОЖЕНИИ Е

При необходимости функционал можно расширить, добавив поддержку более сложных структур документов, форматирование таблиц, вставку изображений и т.д. Таким образом, автоматизированная генерация накладных существенно упрощает документооборот, обеспечивает стандартизацию и точность данных, а также экономит время сотрудников, что положительно сказывается на общем уровне сервиса и эффективности бизнес-процессов.

****2.8.4**** Структура хранения и визуализации изображений

В программе цифровой галереи ключевым элементом является организация метода загрузки, хранения и отображения изображений, так как визуальные материалы представляют собой центральную составляющую системы и существенно влияют на восприятие интерфейса пользователем, а также на наполнение базы данных. Вместо хранения изображений в виде двоичных данных непосредственно в базе, была выбрана альтернативная стратегия, при которой сами изображения размещаются во внешней файловой системе – в пределах папки проекта – и связаны с записями базы данных через сохранение относительных путей к файлам. Такой подход позволяет значительно оптимизировать работу приложения: база данных не перегружается при большом объёме графического контента, что обеспечивает высокую скорость обработки запросов и повышает производительность.

Кроме того, он упрощает миграцию базы и резервное копирование, поскольку визуальные файлы могут обрабатываться отдельно от данных, а также обеспечивает гибкость при переносе системы на сервер или облачную платформу.

Отображение изображений реализуется уже на уровне XAML-разметки главного окна приложения, где элементы управления <Image> используют относительные пути к изображениям для задания оформления. Например, логотипы и элементы брендинга задаются с помощью подобного кода.



**Рисунок 29 – Заглушка изображения**

Хранение изображений в папке Resources централизует структуру проекта и облегчает его масштабирование, позволяя не разбрасывать файлы по различным каталогам.

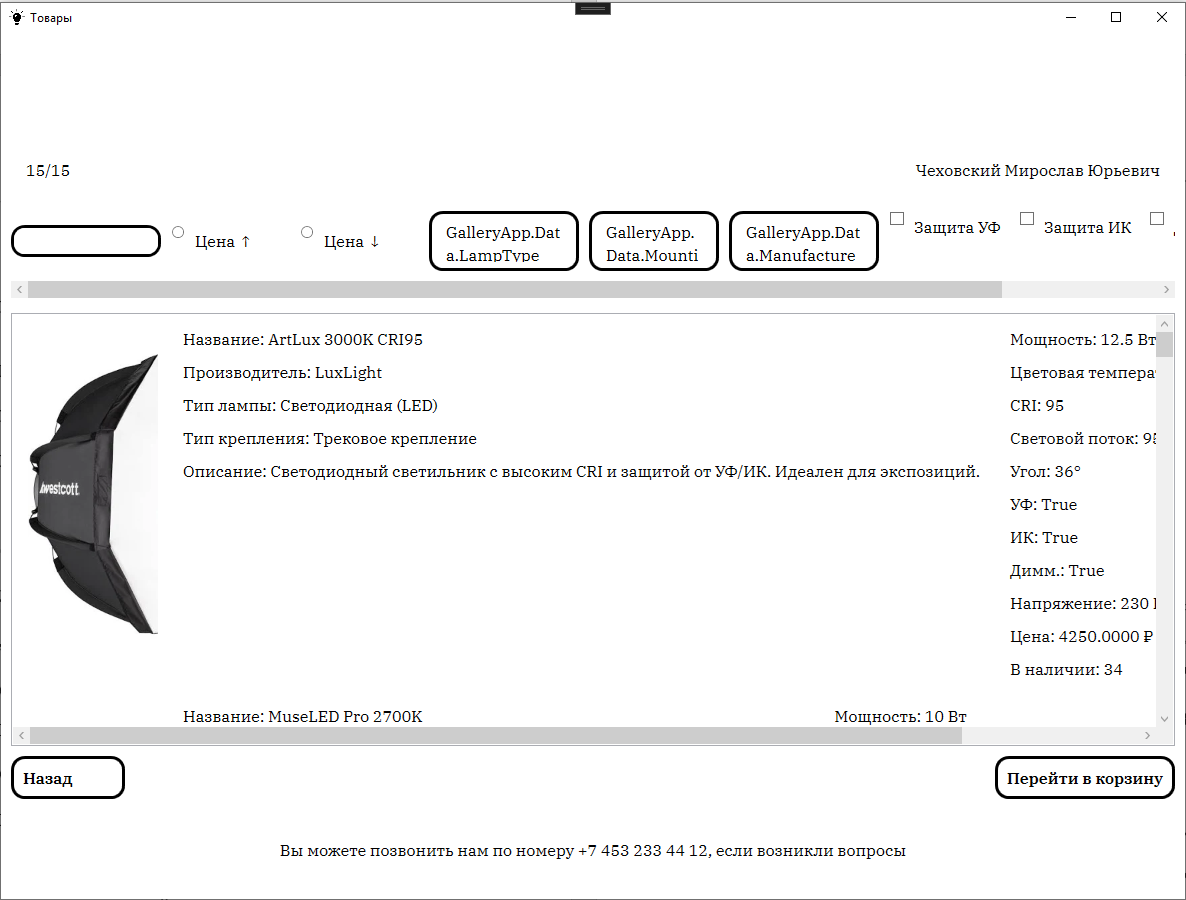
Особое значение имеет этап добавления изображений пользователями – администраторами и сотрудниками – при загрузке новых картин. Для выбора файла используется стандартный диалог OpenFileDialog из пространства имён Microsoft.Win32. После выбора происходит копирование изображения в специальную внутреннюю папку проекта (например, Images), откуда оно загружается при отображении. В коде это реализовано следующим образом: пользователь выбирает файл, затем из полного пути извлекается имя файла, формируется конечный путь в папке проекта, и файл копируется с перезаписью при необходимости. При этом в объекте Lamp сохраняется относительный путь к файлу, который затем записывается в базу данных в поле ImagePath. Таким образом, в таблице Lamp каждому изображению соответствует строка с относительным путём, а не бинарный массив, что значительно облегчает управление базой и повышает её производительность.

Все изображения загружаются через элемент интерфейса Image. Путь добавления изображения может быть, как относительным, так и абсолютным. В этом приложении все изображения выводятся по уже сохраненному пути, но вариативность выбора дает гибкость для адаптации под разные условия: локальные файлы, сетевые ресурсы или удалённые хранилища. Это также открывает перспективу для последующего перехода к хранению изображений в облаке с использованием URL-ссылок.

****2.8.5**** Личный кабинет пользователя

Личный кабинет пользователя является страницей просмотра с функциями, доступными только пользователю. Главная задача – обеспечить интуитивно понятный интерфейс для оформления заказов, просмотра ассортимента товаров и управления своими действиями без необходимости непосредственного обращения к сотрудникам компании. В личный кабинет пользователь попадает после успешной авторизации, там он может ознакомиться с ассортиментом: доступными светильниками и осветительным оборудованием, все товары структурированы по категориям и дополнены иллюстрациями, и подробными описаниями.

Пользователь может ознакомиться с характеристиками товаров и добавить выбранные позиции в корзину. Кроме того, через личный кабинет можно попасть на страницу корзины и там её оформит, добавив адрес доставки и комментарии к заказу, если потребуется, что даёт возможность индивидуализировать заявки и уточнить предпочтения клиента. После формирования заказа система автоматически сохраняет его в базе данных, делая доступным для дальнейшей обработки сотрудниками и администраторами.



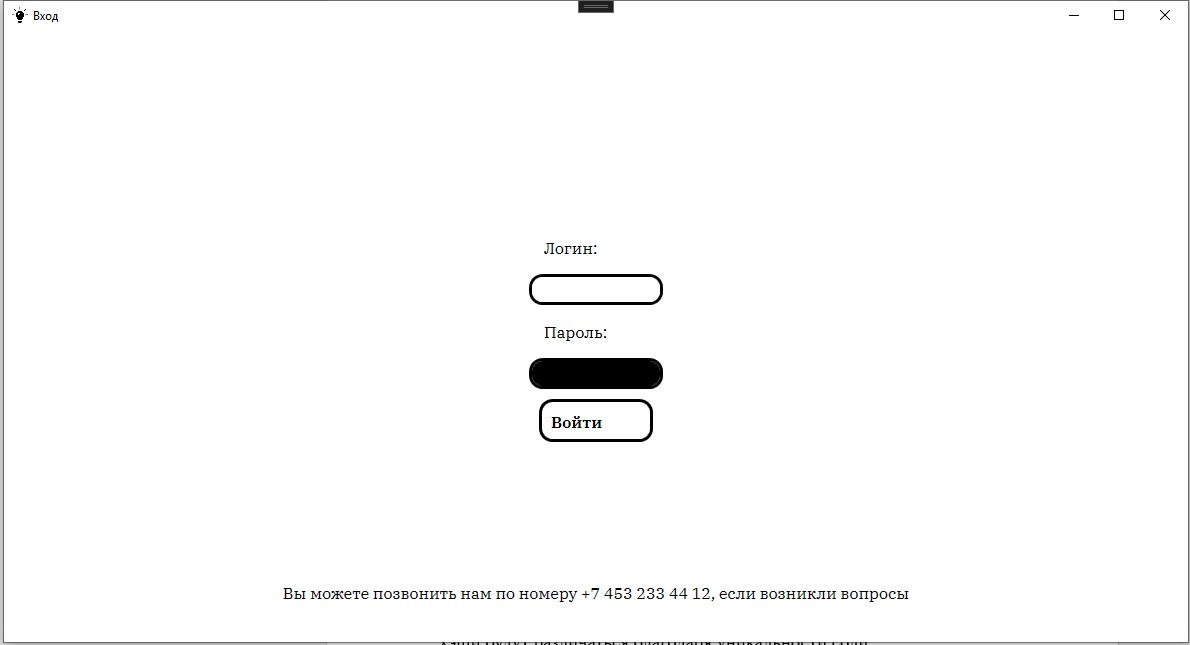
**Рисунок 30 – Личный кабинет пользователя**

****2.8.5.1**** Добавление товаров в корзину

Механизм добавления товаров в корзину разработан для обеспечения максимально простого и быстрого взаимодействия пользователя с системой. После входа в личный кабинет пользователь получает доступ к каталогу продукции, где каждый товар виден с изображением, наименованием, кратким описанием и актуальной ценой. Пользователь может просматривать доступные позиции, фильтровать их по различным параметрам, таким как тип, размер или цена, и выбирать интересующие модели.

Для добавления выбранного товара в корзину следует нажать кнопку рядом с карточкой изделия. После этого товар автоматически сохраняется в локальном представлении корзины, которое отображается в интерфейсе пользователя.

Перед оформлением заказа пользователь может проверить содержимое корзины, внести необходимые изменения и перейти к следующему шагу – вводу адреса доставки и добавлению комментариев к заказу. Интеграция корзины с базой данных позволяет автоматически привязывать заказ к конкретному пользователю, облегчая последующую обработку заявок и взаимодействие с сотрудниками. Этот механизм делает процесс заказа осветительного оборудования более структурированным, прозрачным и удобным, даже для новых клиентов.



**Рисунок 30 – Корзина пользователя**



**Рисунок 30 – Фрагмент кода создания таблицы**

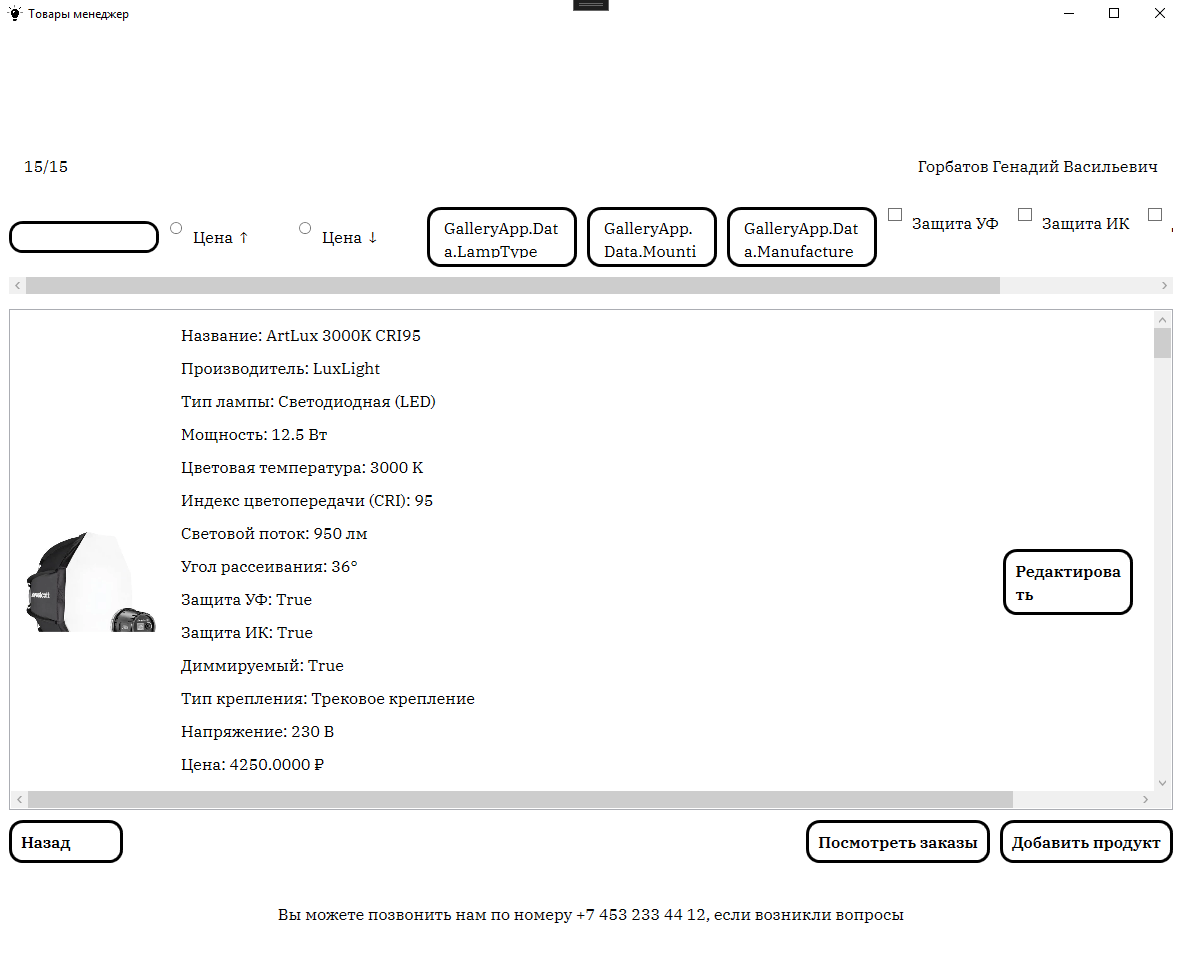
Полный программный код представлен в ПРИЛОЖЕНИИ Д.

****2.8.6**** Личный кабинет сотрудника

Личный кабинет сотрудника предоставляет расширенный набор возможностей по сравнению с кабинетом обычного пользователя, так как ориентирован на выполнение административно-организационных задач, связанных с поддержанием товарного каталога и обработкой поступающих заказов. Интерфейс разработан с учётом специфики обязанностей сотрудника и необходимости оперативного принятия решений.

Сразу после входа в систему сотрудник получает доступ к полному списку заказов, оформленных пользователями, с возможностью просмотра подробной информации о каждом заказе: составе корзины, адресе доставки, комментариях клиента и текущем статусе выполнения. Это обеспечивает оперативное реагирование на новые заявки, уточнение деталей и формирование накладных при необходимости.

Помимо работы с заказами, сотрудник обладает правом редактирования ассортимента продукции – через удобную форму он может добавлять новые товары, изменять описание существующих, корректировать цены и устанавливать их доступность. Все изменения автоматически синхронизируются с базой данных и мгновенно отображаются в клиентской части приложения. Кроме того, сотрудник управляет изображениями товаров, включая загрузку новых и удаление устаревших, что позволяет поддерживать визуальный контент каталога в актуальном состоянии и способствует более лёгкому выбору клиентом. Некоторые функции сотрудника дублируют возможности пользователя, например добавление товаров в корзину и оформление заказов, поэтому они в этом разделе не повторяются. В целом, личный кабинет сотрудника представляет собой комплексный инструмент для контроля ассортимента, управления заказами и визуального сопровождения продукции, что делает его важнейшим элементом цифровой торговой платформы компании.



**Рисунок 31 – Личный кабинет сотрудника**

****2.8.6.1**** Управление товарами

Работа роли сотрудника в этом приложении заключается в управление товарным каталогом, поэтому она доступна сотруднику в его личном кабинете. Она позволяет поддерживать актуальность ассортимента, быстро вносить изменения в карточки продукции и оперативно добавлять новые позиции. Эта подсистема обеспечивает прямое взаимодействие сотрудника с базой данных, где хранятся сведения обо всех светильниках и осветительном оборудовании, доступных для продажи.

В интерфейсе были созданы инструменты для создания новых записей о товарах. При добавлении новой позиции сотрудник указывает её наименование, категорию, подробное описание, цену, параметры размера и тип освещения. После сохранения введённые данные мгновенно отображаются в клиентском интерфейсе, что обеспечивает оперативную доступность товара для покупателей без задержек.

Кроме того, предусмотрена возможность редактирования существующих товаров. Это необходимо для обновления технических характеристик, корректировки цен, применения скидок и других изменений. В случае устаревания модели или её снятия с производства сотрудник может временно скрыть товар из общего каталога, не удаляя его из базы данных, что позволяет сохранить информацию для учёта и аналитики.

Функция удаления записи о товаре доступна только в исключительных случаях, когда изделие больше не продаётся и не имеет аналитической ценности. Для предотвращения случайного удаления процесс требует обязательного подтверждения действия.

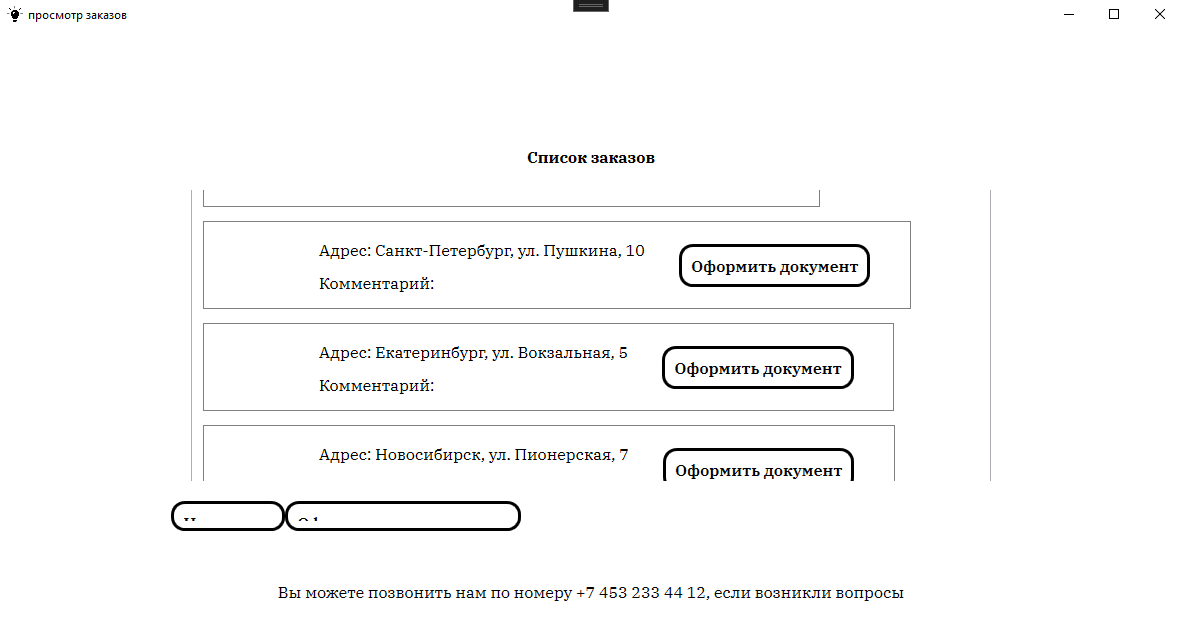
Все изменения синхронизируются с базой данных посредством Entity Framework, что гарантирует сохранность целостности данных и корректное взаимодействие с остальными компонентами приложения. Управление ассортиментом даёт сотруднику возможность гибко адаптировать витрину товаров под текущие потребности рынка и улучшать качество взаимодействия с клиентами.

****2.8.6.2**** Формирование накладных

Формирование накладных в личном кабинете сотрудника очень помогает сотрудникам компании справляться с нагрузкой, потому что любое снижение рутинных и нудных задач сильно влияет на производительность. Никакой человек не хочет долго работать в Word. Так что автоматизация отчетов – это важная функция приложения. Она работает после того, как пользователь оформляет заказ, а сотрудник получает доступ к его подробной информации. К этой информации относится перечни светильников, адрес доставки и дополнительные комментарии. На основе этих данных запускается процесс автоматического создания накладной.

Для генерации документа применяется интеграция с библиотекой Microsoft.Office.Interop.Word, которая позволяет автоматически заполнять заранее подготовленный шаблон Word-документа данными из базы. В сформированной накладной создаётся таблица, содержащая наименования товаров, их количество, цену за единицу и итоговую сумму заказа. Кроме того, в документ включаются дата оформления, сведения о сотруднике, а при необходимости – фирменный логотип компании и подписи ответственных лиц.

Пользовательский интерфейс генерации накладной реализован максимально просто и удобно: сотрудник выбирает нужный заказ и нажимает кнопку «Создать накладную». После этого появляется возможность сохранить готовый файл на компьютер или отправить на печать. Такой подход значительно упрощает документооборот, исключает ручной ввод данных и снижает вероятность ошибок.

Созданные накладные могут использоваться как для внутреннего учёта, так и для передачи в логистическую службу с целью организации отгрузки. Автоматизация этого процесса позволяет сократить время обработки заказов и повысить качество обслуживания клиентов, делая работу сотрудника более эффективной и структурированной.

**Рисунок 32 – Страница оформления заказов**

****2.8.7**** Личный кабинет администратора

Личный кабинет администратора представляет собой многофункциональный раздел приложения, предоставляющий наиболее полный и всесторонний доступ ко всем возможностям системы. Он предназначен для комплексного управления учётными записями сотрудников и управления каталога осветительного оборудования.

Самой важной обязанностью администратора должно быть управление пользователями. Через интерфейс он может заводить новые учётные записи для сотрудников или пользователей, там же он назначает им соответствующие роли и вводит их данные.

Кроме управления пользователями, администратор обладает правом и удалять сведения о товарах. Он может добавлять новые позиции в каталог осветительного оборудования, изменять характеристики уже существующих товаров, загружать и обновлять изображения, назначать категории и корректировать ценовые параметры. Эти возможности особенно важны для своевременного обновления ассортимента и быстрой реакции на изменения в поставках. Данная функциональность была подробно описана ранее в пункте 2.8.6.1 «Управление товарами».

Функция формирования накладных в формате Word-документов также доступна администратору, однако так как она была подробно рассмотрена в пункте 2.8.6.2, здесь её описание не дублируется.

Благодаря расширенному уровню доступа и высокой ответственности, личный кабинет администратора выступает центральным элементом управления системой, обеспечивая её стабильность, безопасность и актуальность данных.



**Рисунок 33 – Личный кабинет администратора**

Полный программный код представлен в ПРИЛОЖЕНИИ Ж

****2.8.7.1**** Управление пользователями

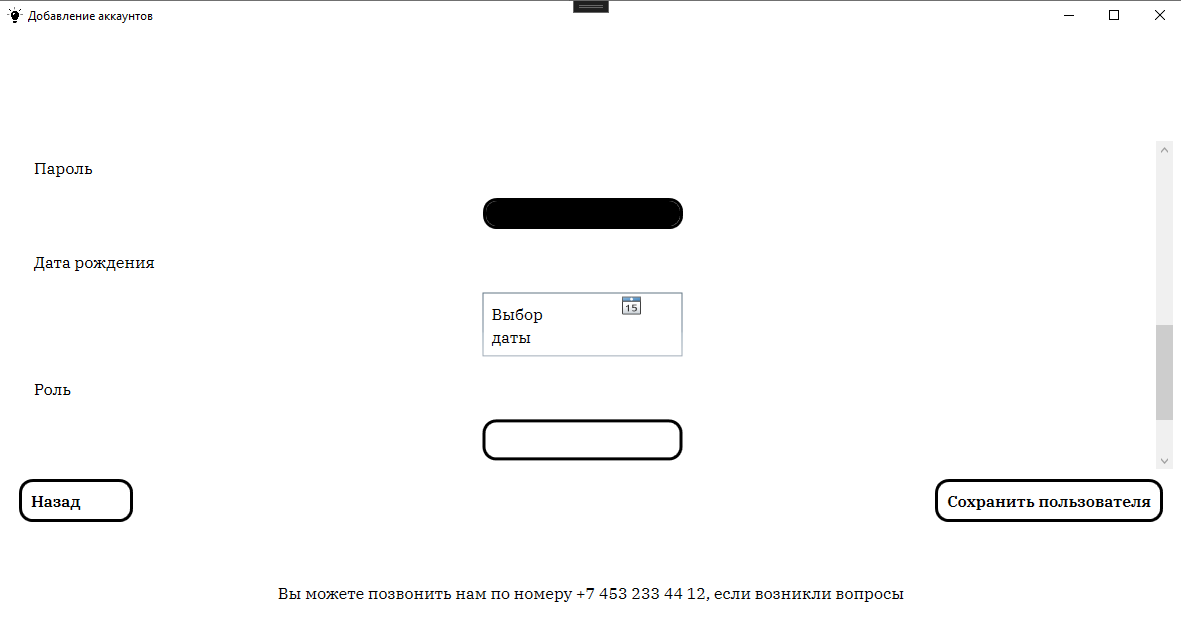
Функция управления пользователями является одной из ключевых возможностей личного кабинета администратора и предоставляет широкий набор инструментов для создания, редактирования и удаления учетных записей сотрудников. Главная задача данного функционала – обеспечить правильную организацию прав доступа в системе и поддерживать актуальность кадровых данных.

При добавлении нового пользователя администратор вводит основные сведения: имя, логин, пароль и выбирает соответствующую роль (например, сотрудник). В момент сохранения система автоматически генерирует соль и выполняет хеширование пароля, что гарантирует надежное и безопасное хранение учетной информации. Механизм работы с паролями подробно описан в пункте 2.4.1 «Авторизация».

Редактирование учетных записей позволяет обновлять данные пользователей, изменять их роли, а также блокировать или разблокировать доступ при необходимости, что особенно важно при изменениях в кадровом составе или в политике безопасности компании. Такая гибкость обеспечивает оперативное реагирование на любые внутренние изменения.

Удаление учетной записи применяется в случае увольнения сотрудника либо при утрате необходимости в доступе к системе. При этом сохраняется история действий пользователя для последующего аудита и анализа, что способствует повышению прозрачности и безопасности работы с системой.

Таким образом, управление пользователями является основным инструментом контроля за кадровым составом и разграничением прав доступа, играя важную роль в обеспечении информационной безопасности и порядке внутри приложения.



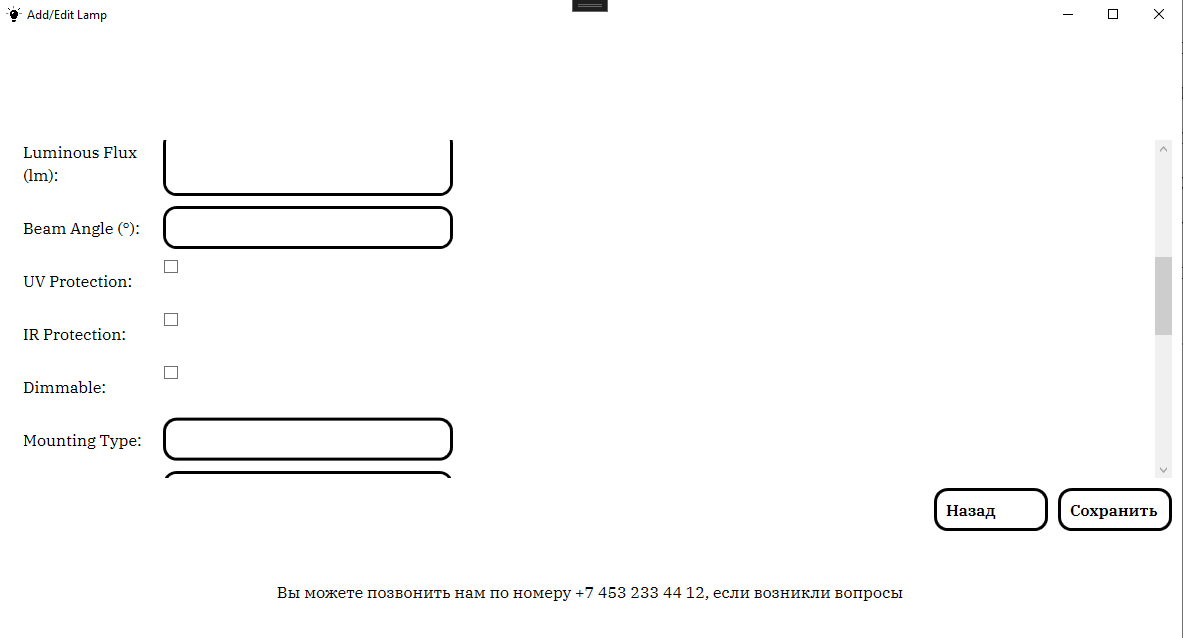
**Рисунок 34 – Добавление пользователей**

****2.8.7.1**** Управление товарами

Функция товарами доступна и администраторам, и сотрудникам, чтобы все сотрудники имели возможность заведения новых позиций в базу данных. Товары также можно редактировать и удалять и эти функции также доступны всем сотрудникам.

При добавлении нового товара администратор или сотрудник вводит все сведения о товаре.

Редактирование товаров позволяет менять их картинки и другую важную информацию, которая поменялась или при создании товара была допущена ошибка. Возможность изменений в любой момент увеличивает гибкость структуры приложения и удобство работы с ним. То, что все возможности для изменения товара доступны всем пользователем ускоряет работу компании, ведь не нужно ждать подтверждения администратора каждый раз, когда добавляется товар.



**Рисунок 35 – Добавление товаров**

****ЗАКЛЮЧЕНИЕ****

Процесс проектирования программного обеспечения, предназначенного для автоматизации внутренних операций компании, специализирующейся на поставках осветительного оборудования, начался с тщательного анализа требований и формализации ключевых бизнес-процессов. Особое внимание уделялось управлению товарным каталогом, обработке заказов и взаимодействию различных категорий пользователей. Цель заключалась в создании удобной и надёжной системы, способной учитывать особенности бизнес-логики предприятия и обеспечивать гибкость при возможных изменениях.

На этапе проектирования была определена общая структура системы, включающая пользовательский интерфейс, управляющие компоненты и модуль взаимодействия с базой данных. Было принято решение реализовать приложение в десктопной форме с использованием Windows Presentation Foundation (WPF), что позволило создать настраиваемый визуальный интерфейс и обеспечить высокий уровень интерактивности. Внутренняя логика системы была организована с опорой на чёткое разграничение ролей пользователей, включая административные функции и доступ к справочной информации.

Особое внимание при проектировании было уделено корректному отражению предметной области в структуре базы данных. Таблицы и связи между ними разрабатывались таким образом, чтобы обеспечить непротиворечивость данных, ускорить выполнение операций выборки и оптимизировать дальнейшую обработку информации. Были задействованы механизмы первичных и внешних ключей, нормализация данных и логическое распределение сущностей по функциональным категориям.

В результате, архитектура разработанного приложения получилась простой, но гибкой, обеспечивая учёт и управление складом светотехнического оборудования, а также удобное взаимодействие с музейными и выставочными партнёрами. Одна из главных целей была дальнейшая масштабируемость, гибкость и удобность расширяемости приложения. Это удалось достичь, предусмотрены возможности добавления новых модулей до интеграции с внешними сервисами и это делает программу долгосрочным и эффективным решением.

****СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ****

**Стандарты:**

1. **ГОСТ 19.101-77. Единая система программной документации. Виды программ и программных документов. [Электронный ресурс] – URL: https://docs.cntd.ru/document/1200000790 Режим доступа: свободный (дата обращения: 20.04.2025).**
2. **ГОСТ 19.201-78. Единая система программной документации. Требования к программам. [Электронный ресурс] – URL: https://docs.cntd.ru/document/1200000791 Режим доступа: свободный (дата обращения: 08.03.2025);**
3. **ГОСТ 19.201-78. Единая система программной документации. Требования к программам. [Электронный ресурс] – URL: https://docs.cntd.ru/document/1200000791 Режим доступа: свободный (дата обращения: 08.03.2025);**
4. **ГОСТ 34.602-89. Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы. [Электронный ресурс] – URL: https://docs.cntd.ru/document/1200000792 Режим доступа: свободный (дата обращения: 18.03.2025);**

Книги под фамилией автора**:**

1. **C# 9 и .NET 5. Современная кроссплатформенная разработка / Марк Дж. Прайс. – Питер, 2021;**
2. **Световая архитектура и светодизайн города и интерьера: Глоссарий / Н.И. Щепетков. – Перо, 2024;**
3. **Цвет и свет / Джеймс Гарни; пер. с англ. И.Ю. Наумовой. – Эксмо, 2022;**
4. **Свет и цвет в архитектуре и дизайне / под ред. А.В. Иванова. – Питер, 2021;**
5. **Архитектурное освещение: теория и практика / Е.С. Кузнецов. – Архитектура-С, 2020;**
6. **Музей как объект светового дизайна // Светотехника. – № 6, 2022.**

**Электронные ресурсы:**

1. **Нативное приложения – приложение, создаваемое под конкретную платформу. [Электронный ресурс] – URL:** [**https://appmaster.io/ru/blog/otlichie-nativnyh-mobilnyh-prilozhenij-ot-vseh-ostalnyh**](https://appmaster.io/ru/blog/otlichie-nativnyh-mobilnyh-prilozhenij-ot-vseh-ostalnyh) **Режим доступа: свободный (дата обращения: 06.02.2025).**
2. **Хеширование – это преобразование информации с помощью особых математических формул. [Электронный ресурс] – URL: https://blog.skillfactory.ru/glossary/heshirovanie/ Режим доступа: свободный (дата обращения: 02.05.2025).**
3. **C# и WPF | Привязка [Электронный ресурс] – URL: https://metanit.com/sharp/wpf/11.php Режим доступа: свободный (дата обращения: 22.04.2025);**
4. **Entity Framework | Microsoft Learn [Электронный ресурс] – URL: https://learn.microsoft.com/ru-ru/aspnet/entity-framework Режим доступа: свободный (дата обращения: 06.03.2025);**

**JVM (Java Virtual Machine) – это платформенно-независимый язык программирования. [Электронный ресурс] – URL:** [**https://www.ibm.com/docs/en/b2b-integrator/6.1.1?topic=management-java-virtual-machine**](https://www.ibm.com/docs/en/b2b-integrator/6.1.1?topic=management-java-virtual-machine) **Режим доступа: свободный (дата обращения: 06.02.2025).**

**Microsoft Dynamics 365 Business Central [Электронный ресурс] – URL: https://www.microsoft.com/en-us/dynamics-365/products/business-central Режим доступа: свободный (дата обращения: 06.02.2025). SAP Business One [Электронный ресурс] – URL: https://www.sap.com/products/erp/business-one.html Режим доступа: свободный (дата обращения: 06.02.2025).**

**NET API в веб-разработке: [Электронный ресурс] – URL: https://habr.com/ru/companies/jugru/articles/328090/ Режим доступа: свободный (дата обращения: 02.05.2025).**

1. **WPF и C# | Полное руководство [Электронный ресурс] – URL: https://metanit.com/sharp/wpf/ Режим доступа: свободный (дата обращения: 23.04.2025).**
2. **Безопасное хранение соли [Электронный ресурс] – URL: https://ru.stackoverflow.com/questions/1186646/ Режим доступа: свободный (дата обращения: 16.04.2025);**
3. **Введение в ADO.NET [Электронный ресурс] – URL: https://metanit.com/sharp/adonet/1.1.php Режим доступа: свободный (дата обращения: 08.04.2025);**

**Графовые базы данных на примере простых запросов [Электронный ресурс] – URL: https://habr.com/ru/companies/neoflex/articles/654267/ Режим доступа: свободный (дата обращения: 26.02.2025).**

1. **Интеллектуальные системы управления освещением в современных зданиях [Электронный ресурс] – URL: https://smartlighting.org/intelligent-lighting-systems/. Режим доступа: свободный (дата обращения: 01.06.2025);**
2. **Как правильно хранить пароли своих пользователей [Электронный ресурс] – URL: https://www.kaspersky.ru/blog/how-to-store-passwords/36161/ Режим доступа: свободный (дата обращения: 23.04.2025);**

**КОНСИСТЕНТНОСТЬ - взаимная согласованность между эмпирическими индикаторами [Электронный ресурс] – URL: https://ht-lab.ru/obuchenie/slovari/gipertekstovyy/konsistentnost/#:~:text=КОНСИСТЕНТНОСТЬ%20%20взаимная%20согласованность%20между%20эмпирическими,или%20к%20одному%20диагностическому%20конструкту). Режим доступа: свободный (дата обращения: 02.05.2025).**

**Логирование: понятие, механизмы и уровни: [Электронный ресурс] – URL: https://habr.com/ru/articles/795445/ Режим доступа: свободный (дата обращения: 02.05.2025).**

1. **Обзор - ADO.NET | Microsoft Learn [Электронный ресурс] – URL: https://learn.microsoft.com/ru-ru/dotnet/framework/data/adonet/ado-net-overview Режим доступа: свободный (дата обращения: 21.04.2025);**
2. **Основы светотехники: принципы и применение [Электронный ресурс] – URL: https://www.lightingacademy.com/basics-of-lighting/. Режим доступа: свободный (дата обращения: 01.06.2025);**
3. **Пароли, шифрование, хеширование и СОЛЬ [Электронный ресурс] – URL: https://www.youtube.com/watch?v=VbB99SOR8OE Режим доступа: свободный (дата обращения: 20.04.2025).**
4. **Работа с базами данных в C# и .NET [Электронный ресурс] – URL: https://metanit.com/sharp/ado.php Режим доступа: свободный (дата обращения: 20.04.2025);**
5. **Разница между реляционными и нереляционными базами данных. [Электронный ресурс] – URL: https://habr.com/ru/companies/sberbank/Lampicles/672022/ Режим доступа: свободный (дата обращения: 06.02.2025);**

**Референциальная целостность: [Электронный ресурс] – URL: https://appmaster.io/ru/glossary/referentsial-naia-tselostnost Режим доступа: свободный (дата обращения: 02.05.2025).**

1. **Руководство по C# – управляемый язык .NET [Электронный ресурс] – URL: https://learn.microsoft.com/ru-ru/dotnet// Режим доступа: свободный (дата обращения: 23.04.2025);**

**Светодиоды: что это, принцип действия, характеристики [Электронный ресурс] – URL: https://dip8.ru/articles/chto-takoe-svetodiody/ Режим доступа: свободный (дата обращения: 26.02.2025).**

1. **Сравнение светодиодных и люминесцентных ламп: эффективность и долговечность [Электронный ресурс] – URL: https://www.energy-efficient-lighting.com/led-vs-fluorescent/. Режим доступа: свободный (дата обращения: 01.06.2025);**

**Типобезопасность в JavaScript: Flow и TypeScript: [Электронный ресурс] – URL: https://habr.com/ru/articles/552760/ Режим доступа: свободный (дата обращения: 02.05.2025).**

1. **Учебник по языку C# 13 и платформе .NET 9 [Электронный ресурс] – URL: https://metanit.com/sharp/tutorial/ Режим доступа: свободный (дата обращения: 16.04.2025);**

**Цветовое пространство CIE 1931 [Электронный ресурс] – URL: Цветовое пространство CIE 1931Режим доступа: свободный (дата обращения: 26.02.2025). Спектры, спектральный анализ [Электронный ресурс] – URL: https://school97.ru/metod/index.php?SECTION\_ID=163&ELEMENT\_ID=4179 Режим доступа: свободный (дата обращения: 26.02.2025).**

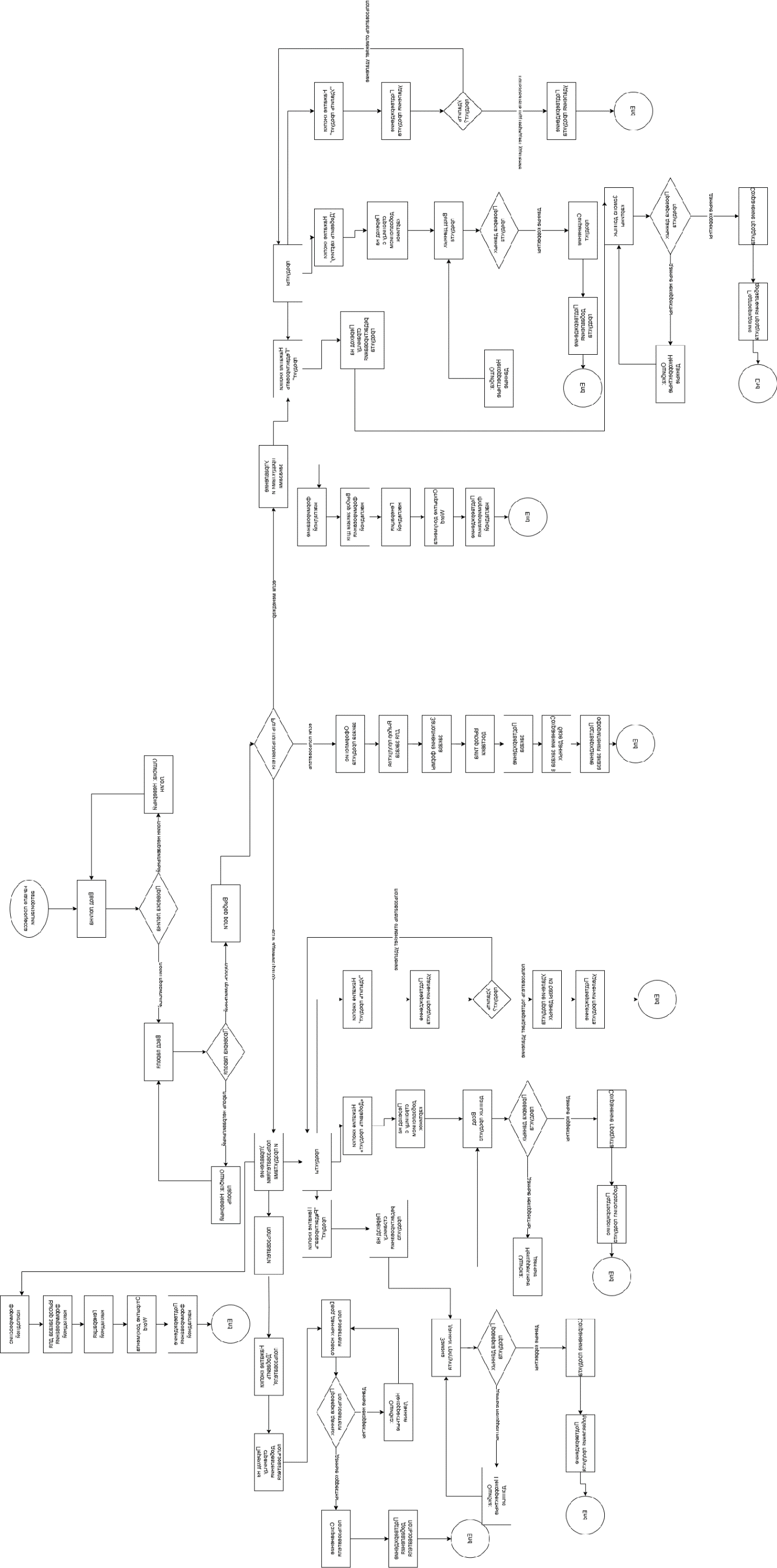
**Чем галогенные лампы отличаются от ламп накаливания? [Электронный ресурс] – URL: https://www.mdm-light.ru/publications/articles/chem-galogennye-lampy-otlichayutsya-ot-lamp-nakalivaniya/ Режим доступа: свободный (дата обращения: 26.02.2025).**

**Что такое ADO.NET [Электронный ресурс] – URL: https://skillbox.ru/media/gamedev/chto-takoe-adonet/ Режим доступа: свободный (дата обращения: 06.02.2025). Рефакторинг – это переработка исходного кода программы, чтобы он стал более простым и понятным. [Электронный ресурс] – URL:https://skillbox.ru/media/code/chto\_takoe\_refaktoring\_koda\_i\_zachem\_on\_nuzhen/ Режим доступа: свободный (дата обращения: 21.02.2025).**

1. **Что такое Entity Framework и как его использовать [Электронный ресурс] – URL: https://skillbox.ru/media/code/entity\_framework/ Режим доступа: свободный (дата обращения: 23.04.2025);**
2. **Что такое Windows Presentation Foundation - WPF .NET [Электронный ресурс] – URL: https://learn.microsoft.com/ru-ru/dotnet/desktop/wpf/overview/?view=netdesktop-9.0 Режим доступа: свободный (дата обращения: 16.04.2025);**

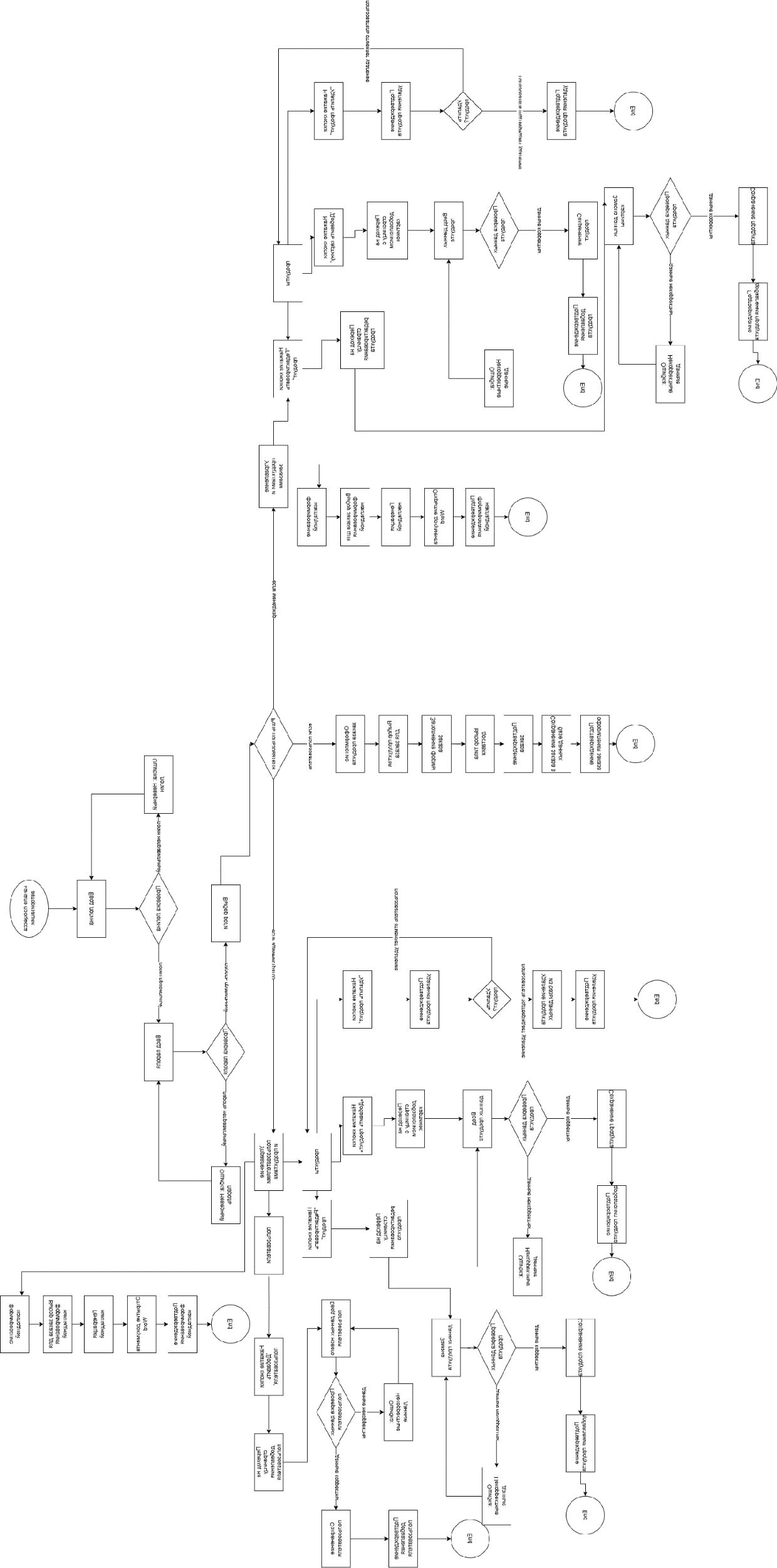
**ПРИЛОЖЕНИЯ**

**ПРИЛОЖЕНИЕ А**

**Диаграмма активностей**

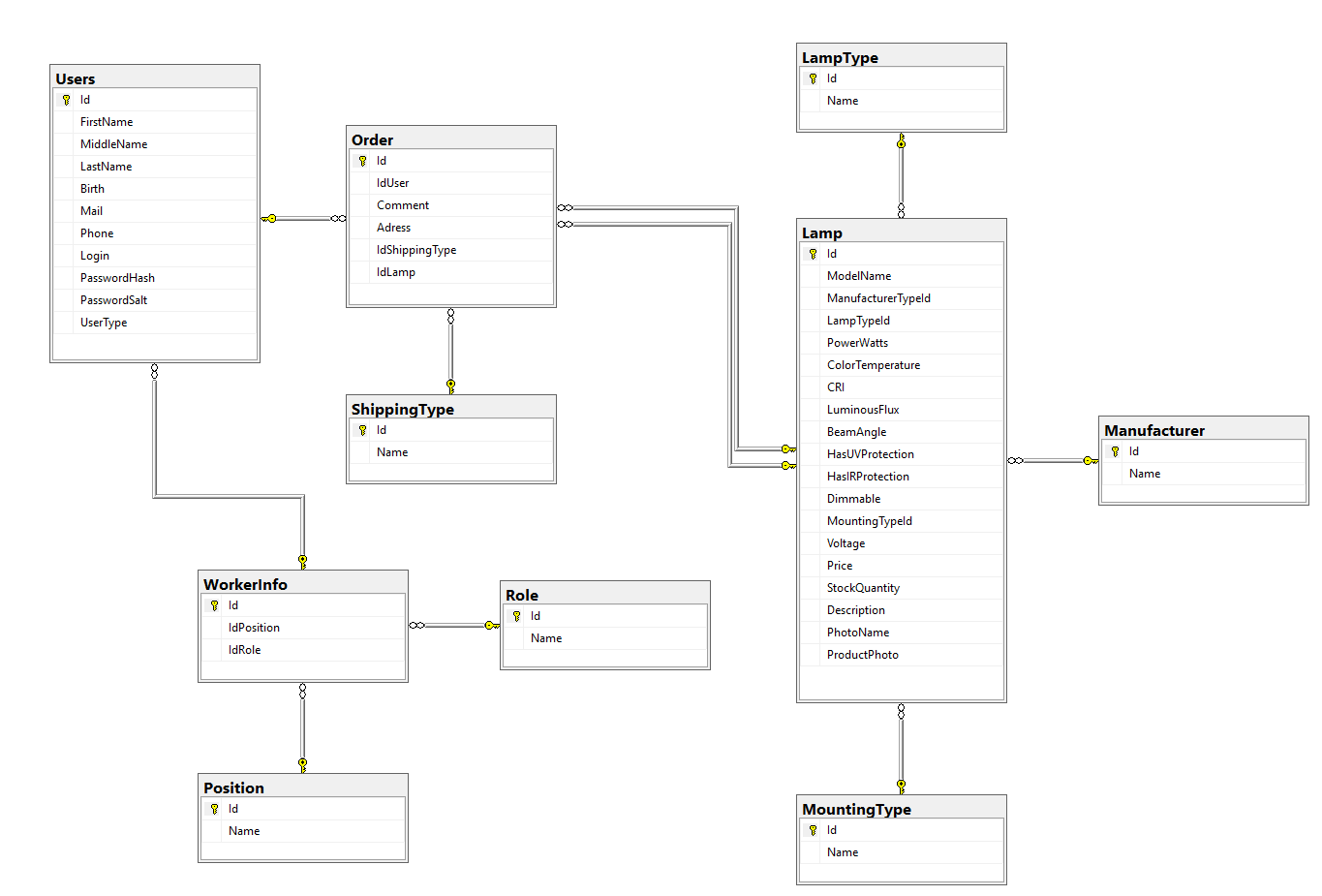
**ПРОДОЛЖЕНИЕ ПРИЛОЖЕНИЯ А**

**Диаграмма активностей**

****

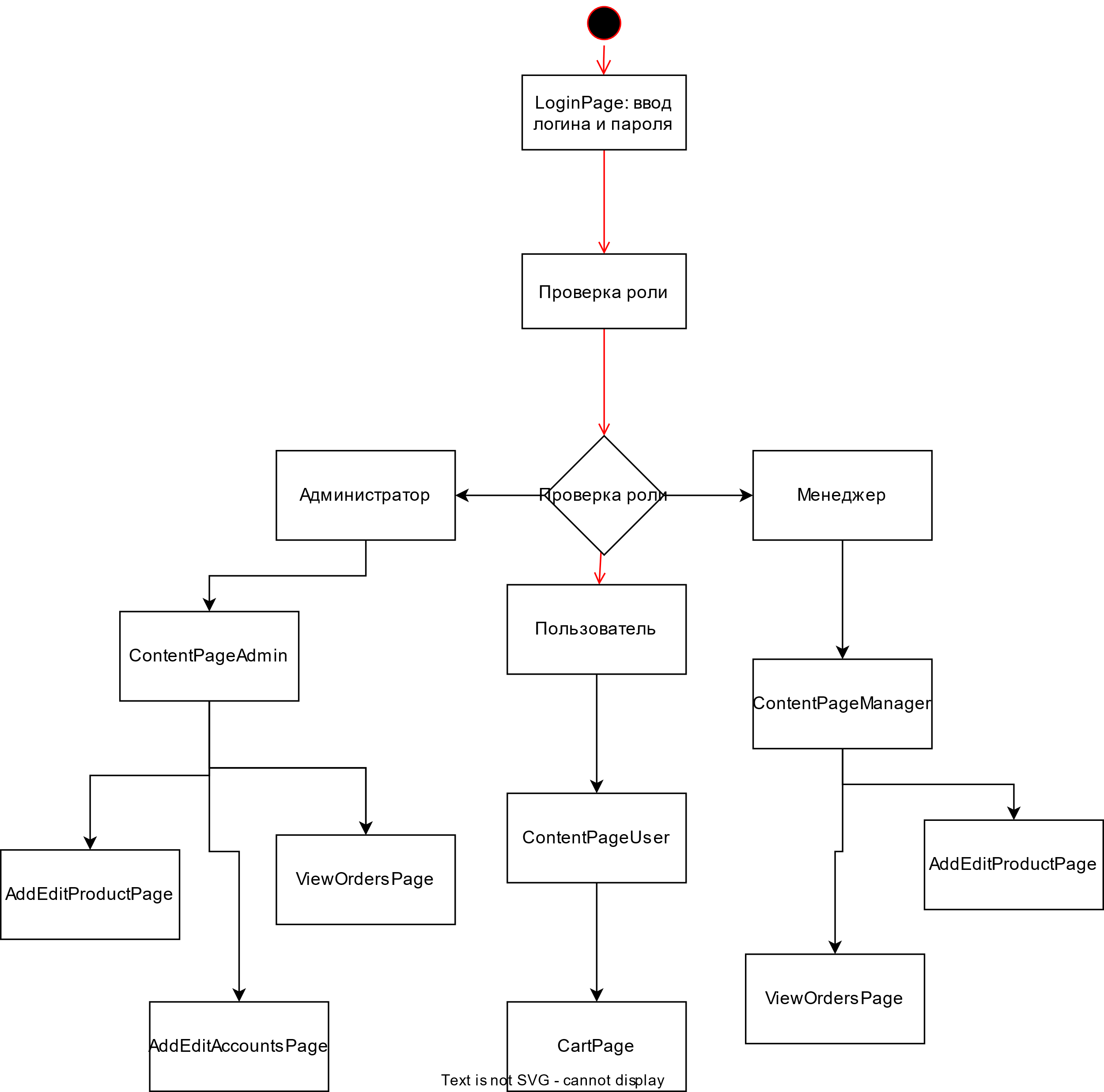
**Приложение Б**

**ER-диаграмма**



**Приложение В**

**Диаграмма прецедентов**



**Приложение Г**

**Диаграмма use-case**



**Приложение Д**

**Код внутренней логики страницы Cart**

public ObservableCollection<Lamp> CartItems { get; set; }

public CartPage()

{

InitializeComponent();

CartItems = new ObservableCollection<Lamp>(Manager.GetCartForCurrentUser());

this.DataContext = this;

LoadShippingTypes();

DisplayUserInfo();

}

private void DisplayUserInfo()

{

if (Manager.CurrentUser != null)

{

UserInfoLabel.Visibility = Visibility.Visible;

UserInfoLabel.Content = $"{Manager.CurrentUser.LastName} {Manager.CurrentUser.FirstName} {Manager.CurrentUser.MiddleName}";

}

else

{

UserInfoLabel.Visibility = Visibility.Hidden;

}

}

private void RemoveButton\_Click(object sender, RoutedEventArgs e)

{

Button button = sender as Button;

if (button != null)

{

var item = button.CommandParameter as Lamp;

if (item != null)

{

Manager.RemoveFromCart(item.Id);

CartItems.Remove(item);

}

}

}

private void SubmitOrderButton\_Click(object sender, RoutedEventArgs e)

{

if (CartItems == null || CartItems.Count == 0)

{

MessageBox.Show("Корзина пуста.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

return;

}

if (Manager.CurrentUser == null)

{

MessageBox.Show("Пользователь не авторизован.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

if (ShippingTypeComboBox.SelectedItem == null)

{

MessageBox.Show("Выберите тип доставки.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

return;

}

if (string.IsNullOrWhiteSpace(CommentTextBox.Text) || string.IsNullOrWhiteSpace(AddressTextBox.Text))

{

MessageBox.Show("Заполните все обязательные поля.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

return;

}

try

{

var context = gallerydatabaseEntities.GetContext();

foreach (var lamp in CartItems)

{

var selectedShippingType = ShippingTypeComboBox.SelectedItem as ShippingType;

var newOrder = new Order

{

IdUser = Manager.CurrentUser.Id,

Comment = CommentTextBox.Text,

Adress = AddressTextBox.Text,

IdShippingType = selectedShippingType.Id

};

context.Order.Add(newOrder);

}

context.SaveChanges();

MessageBox.Show("Заказ оформлен успешно!", "Успех", MessageBoxButton.OK, MessageBoxImage.Information);

Manager.CartItems.Clear();

CartItems.Clear();

CommentTextBox.Text = string.Empty;

AddressTextBox.Text = string.Empty;

ShippingTypeComboBox.SelectedIndex = -1;

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при оформлении заказа: {ex.Message}", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void LoadShippingTypes()

{

var context = gallerydatabaseEntities.GetContext();

ShippingTypeComboBox.ItemsSource = context.ShippingType.ToList();

ShippingTypeComboBox.SelectedIndex = 0;

}

**Приложение Д**

**Код класса для генерации Word**

public static class WordDocumentGenerator

{

public static void CreateInvoice(int orderId)

{

try

{

var context = Data.gallerydatabaseEntities.GetContext();

var order = context.Order.FirstOrDefault(o => o.Id == orderId);

if (order == null)

{

MessageBox.Show("Заказ не найден!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

var user = context.Users.FirstOrDefault(u => u.Id == order.IdUser);

var art = context.Art.FirstOrDefault(a => a.id == order.Id);

var shippingType = context.ShippingType.FirstOrDefault(s => s.Id == order.IdShippingType);

if (user == null || art == null || shippingType == null)

{

MessageBox.Show("Не удалось загрузить все данные для накладной!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

var wordApp = new Word.Application();

wordApp.Visible = false;

var document = wordApp.Documents.Add();

document.PageSetup.TopMargin = wordApp.CentimetersToPoints(2);

document.PageSetup.BottomMargin = wordApp.CentimetersToPoints(2);

document.PageSetup.LeftMargin = wordApp.CentimetersToPoints(2);

document.PageSetup.RightMargin = wordApp.CentimetersToPoints(2);

document.Content.Font.Name = "Times New Roman";

document.Content.Font.Size = 12;

Word.Paragraph para;

para = document.Content.Paragraphs.Add();

string formattedDate = DateTime.Now.ToString("d MMMM yyyy г.", new System.Globalization.CultureInfo("ru-RU"));

para.Range.Text = $"Товарная накладная № {order.Id} от {formattedDate}";

para.Range.Font.Size = 16;

para.Range.Font.Bold = 1;

para.Alignment = Word.WdParagraphAlignment.wdAlignParagraphCenter;

para.Range.InsertParagraphAfter();

para = document.Content.Paragraphs.Add();

para.Range.Text = "Поставщик:\nООО «Глобал Текс», ИНН 7701234567, г. Москва, ул. Арбат, д. 15, оф. 23\nТел.: +7 (495) 123-45-67, e-mail: info@galleryart.ru";

para.Range.Font.Size = 12;

para.Range.Font.Bold = 0;

para.Alignment = Word.WdParagraphAlignment.wdAlignParagraphLeft;

para.Format.LeftIndent = wordApp.CentimetersToPoints(1.25f);

para.Range.InsertParagraphAfter();

para = document.Content.Paragraphs.Add();

string clientName = $"{user.LastName} {user.FirstName} {user.MiddleName}";

para.Range.Text = $"Покупатель:\n{clientName}, Адрес: {order.Adress}\nТелефон: {user.Phone}";

para.Range.InsertParagraphAfter();

Word.Table table = document.Tables.Add(para.Range, 2, 5);

table.Range.ParagraphFormat.Alignment = Word.WdParagraphAlignment.wdAlignParagraphCenter;

table.Borders.Enable = 1;

table.Rows.Alignment = Word.WdRowAlignment.wdAlignRowCenter;

table.Cell(1, 1).Range.Text = "№";

table.Cell(1, 2).Range.Text = "Наименование произведения";

table.Cell(1, 3).Range.Text = "Количество";

table.Cell(1, 4).Range.Text = "Цена за ед., руб.";

table.Cell(1, 5).Range.Text = "Сумма, руб.";

table.Cell(2, 1).Range.Text = "1";

table.Cell(2, 2).Range.Text = art.title;

table.Cell(2, 3).Range.Text = "1";

table.Cell(2, 4).Range.Text = art.price.ToString("N2");

table.Cell(2, 5).Range.Text = art.price.ToString("N2");

para = document.Content.Paragraphs.Add();

para.Range.InsertParagraphAfter();

para = document.Content.Paragraphs.Add();

para.Range.Text = $"Комментарий к заказу:\n{order.Comment}";

para.Format.LeftIndent = wordApp.CentimetersToPoints(1.25f);

para.Range.InsertParagraphAfter();

para = document.Content.Paragraphs.Add();

para.Range.Text = $"Тип доставки:\n{shippingType.Name}";

para.Format.LeftIndent = wordApp.CentimetersToPoints(1.25f);

para.Range.InsertParagraphAfter();

para = document.Content.Paragraphs.Add();

para.Range.Text = $"Адрес доставки:\n{order.Adress}";

para.Format.LeftIndent = wordApp.CentimetersToPoints(1.25f);

para.Range.InsertParagraphAfter();

para = document.Content.Paragraphs.Add();

para.Range.Text = "\n\nПодписи сторон:\n\n" +

"От поставщика: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Иванов И.И./\n\n" +

$"От покупателя: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /{user.LastName} {user.FirstName}/";

para.Format.LeftIndent = wordApp.CentimetersToPoints(1.25f);

string directoryPath = Path.Combine(Environment.GetFolderPath(Environment.SpecialFolder.Desktop), "Накладные");

if (!Directory.Exists(directoryPath))

Directory.CreateDirectory(directoryPath);

string filePath = Path.Combine(directoryPath, $"Накладная\_{clientName}\_{DateTime.Now:yyyyMMddHHmmss}.docx");

document.SaveAs2(filePath);

document.Close();

wordApp.Quit();

MessageBox.Show($"Накладная успешно создана:\n{filePath}", "Успех", MessageBoxButton.OK, MessageBoxImage.Information);

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка создания документа:\n{ex.Message}", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

**Приложение Ж**

**Код личного кабинета администратора**

private void LoadDefaultImage()

{

try

{

var uri = new Uri("pack://application:,,,/Resources/default\_lamp.png", UriKind.Absolute);

var resourceStream = Application.GetResourceStream(uri);

if (resourceStream != null)

{

using (var memoryStream = new MemoryStream())

{

resourceStream.Stream.CopyTo(memoryStream);

\_defaultImage = memoryStream.ToArray();

}

}

}

catch

{

\_defaultImage = null;

}

}

private List<Lamp> SortLamps(List<Lamp> lamps)

{

if (SortUpRadioButton.IsChecked == true)

return lamps.OrderBy(l => l.Price).ToList();

if (SortDownRadioButton.IsChecked == true)

return lamps.OrderByDescending(l => l.Price).ToList();

return lamps;

}

private void InitializeComboBoxes()

{

var context = gallerydatabaseEntities.GetContext();

var lampTypes = context.LampType.OrderBy(x => x.Name).ToList();

var mountingTypes = context.MountingType.OrderBy(x => x.Name).ToList();

var manufacturers = context.Manufacturer.OrderBy(x => x.Name).ToList();

InitializeComboBox(LampTypeComboBox, lampTypes, "Все типы ламп");

InitializeComboBox(MountingTypeComboBox, mountingTypes, "Все типы креплений");

InitializeComboBox(ManufacturerComboBox, manufacturers, "Все производители");

}

private void InitializeComboBox<T>(ComboBox comboBox, List<T> items, string defaultName) where T : class, new()

{

var idProp = typeof(T).GetProperty("Id");

var nameProp = typeof(T).GetProperty("Name");

if (idProp == null || nameProp == null)

{

throw new InvalidOperationException($"Класс {typeof(T).Name} должен содержать свойства Id и Name.");

}

var defaultItem = new T();

idProp.SetValue(defaultItem, 0);

nameProp.SetValue(defaultItem, defaultName);

var itemList = new List<T> { defaultItem };

itemList.AddRange(items);

comboBox.ItemsSource = itemList;

comboBox.DisplayMemberPath = "Name";

comboBox.SelectedValuePath = "Id";

comboBox.SelectedValue = 0;

}

private void InitializePage()

{

try

{

var currentUser = Manager.CurrentUser;

if (currentUser != null)

{

FIOLabel.Visibility = Visibility.Visible;

FIOLabel.Content = $"{currentUser.LastName} {currentUser.FirstName} {currentUser.MiddleName}";

}

else

{

FIOLabel.Visibility = Visibility.Hidden;

FIOLabel.Content = "ФИО не найдено";

}

var context = gallerydatabaseEntities.GetContext();

var lamps = context.Lamp.ToList();

SetDefaultImages(lamps);

ProductListView.ItemsSource = lamps;

CountOfLabel.Content = $"{lamps.Count}/{context.Lamp.Count()}";

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

private void SetDefaultImages(List<Lamp> lamps)

{

foreach (var lamp in lamps)

{

if (lamp.ProductPhoto == null || lamp.ProductPhoto.Length == 0)

lamp.ProductPhoto = \_defaultImage;

}

}

1. Светодиоды: что это, принцип действия, характеристики [Электронный ресурс] – URL: https://dip8.ru/articles/chto-takoe-svetodiody/ Режим доступа: свободный (дата обращения: 26.02.2025). [↑](#footnote-ref-1)
2. Чем галогенные лампы отличаются от ламп накаливания? [Электронный ресурс] – URL: https://www.mdm-light.ru/publications/articles/chem-galogennye-lampy-otlichayutsya-ot-lamp-nakalivaniya/ Режим доступа: свободный (дата обращения: 26.02.2025). [↑](#footnote-ref-2)
3. Спектры, спектральный анализ [Электронный ресурс] – URL: https://school97.ru/metod/index.php?SECTION\_ID=163&ELEMENT\_ID=4179 Режим доступа: свободный (дата обращения: 26.02.2025). [↑](#footnote-ref-3)
4. Цветовое пространство CIE 1931 [Электронный ресурс] – URL: Цветовое пространство CIE 1931Режим доступа: свободный (дата обращения: 26.02.2025). [↑](#footnote-ref-4)
5. SAP Business One [Электронный ресурс] – URL: https://www.sap.com/products/erp/business-one.html Режим доступа: свободный (дата обращения: 06.02.2025). [↑](#footnote-ref-5)
6. Microsoft Dynamics 365 Business Central [Электронный ресурс] – URL: https://www.microsoft.com/en-us/dynamics-365/products/business-central Режим доступа: свободный (дата обращения: 06.02.2025). [↑](#footnote-ref-6)
7. Графовые базы данных на примере простых запросов [Электронный ресурс] – URL: https://habr.com/ru/companies/neoflex/articles/654267/ Режим доступа: свободный (дата обращения: 26.02.2025). [↑](#footnote-ref-7)
8. JVM (Java Virtual Machine) – это платформенно-независимый язык программирования. [Электронный ресурс] – URL: <https://www.ibm.com/docs/en/b2b-integrator/6.1.1?topic=management-java-virtual-machine> Режим доступа: свободный (дата обращения: 06.02.2025). [↑](#footnote-ref-8)
9. Нативное приложения – приложение, создаваемое под конкретную платформу. [Электронный ресурс] – URL: <https://appmaster.io/ru/blog/otlichie-nativnyh-mobilnyh-prilozhenij-ot-vseh-ostalnyh> Режим доступа: свободный (дата обращения: 06.02.2025). [↑](#footnote-ref-9)
10. Рефакторинг – это переработка исходного кода программы, чтобы он стал более простым и понятным. [Электронный ресурс] – URL: https://skillbox.ru/media/code/chto\_takoe\_refaktoring\_koda\_i\_zachem\_on\_nuzhen/ Режим доступа: свободный (дата обращения: 21.02.2025). [↑](#footnote-ref-10)
11. Что такое ADO.NET [Электронный ресурс] – URL: https://skillbox.ru/media/gamedev/chto-takoe-adonet/ Режим доступа: свободный (дата обращения: 06.02.2025). [↑](#footnote-ref-11)
12. ORM или как забыть о проектировании БД [Электронный ресурс] – URL: https://habr.com/ru/articles/237889/ Режим доступа: свободный (дата обращения: 02.05.2025). [↑](#footnote-ref-12)
13. КОНСИСТЕНТНОСТЬ - взаимная согласованность между эмпирическими индикаторами [Электронный ресурс] – URL: https://ht-lab.ru/obuchenie/slovari/gipertekstovyy/konsistentnost/#:~:text=КОНСИСТЕНТНОСТЬ%20-%20взаимная%20согласованность%20между%20эмпирическими,или%20к%20одному%20диагностическому%20конструкту). Режим доступа: свободный (дата обращения: 02.05.2025). [↑](#footnote-ref-13)
14. .NET API в веб-разработке: [Электронный ресурс] – URL: https://habr.com/ru/companies/jugru/articles/328090/ Режим доступа: свободный (дата обращения: 02.05.2025). [↑](#footnote-ref-14)
15. . Типобезопасность в JavaScript: Flow и TypeScript: [Электронный ресурс] – URL: https://habr.com/ru/articles/552760/ Режим доступа: свободный (дата обращения: 02.05.2025). [↑](#footnote-ref-15)
16. . Логирование: понятие, механизмы и уровни: [Электронный ресурс] – URL: https://habr.com/ru/articles/795445/ Режим доступа: свободный (дата обращения: 02.05.2025). [↑](#footnote-ref-16)
17. Референциальная целостность: [Электронный ресурс] – URL: https://appmaster.io/ru/glossary/referentsial-naia-tselostnost Режим доступа: свободный (дата обращения: 02.05.2025). [↑](#footnote-ref-17)
18. Хеширование — это преобразование информации с помощью особых математических формул. [Электронный ресурс] – URL: https://blog.skillfactory.ru/glossary/heshirovanie/ Режим доступа: свободный (дата обращения: 02.05.2025). [↑](#footnote-ref-18)