**Содержание**

[Введение 4](#_Toc160249999)

[1. Предварительное планирование и проверка осуществимости 5](#_Toc160250000)

[1.1. Требования к системе в целом 5](#_Toc160250001)

[1.2. Требования к функциям, выполняемым системой 7](#_Toc160250002)

[1.3. Требования к видам обеспечения 8](#_Toc160250003)

[1.4. Сравнение и анализ аналогов системы 9](#_Toc160250004)

[1.5. Инфологическая модель предметной области 11](#_Toc160250005)

[1.6. Прототипирование интерфейса 12](#_Toc160250006)

[2. Практическая часть 15](#_Toc160250007)

[2.1. Назначение и цели создания системы 15](#_Toc160250008)

[2.2. Характеристики объекта автоматизации 15](#_Toc160250009)

[2.3. Обзор инструментальных средств создания продукта 17](#_Toc160250010)

[2.4. Обоснование выбора средств реализации ИС 18](#_Toc160250011)

[2.5. Описание логики функционирования ПО 19](#_Toc160250012)

[2.6. Реализация интерфейса ПО 24](#_Toc160250013)

[2.7. Написание программного кода 27](#_Toc160250014)

[2.8. Защита информации 28](#_Toc160250015)

[Заключение 30](#_Toc160250016)

[Список использованных источников 31](#_Toc160250017)

[Приложение 32](#_Toc160250018)

[Приложение А 32](#_Toc160250019)

# Введение

Темой данного курсового проекта является «Разработка модуля информационной системы для учёта компьютерной техники и оборудования».

В любых организациях люди часто сталкиваются с наличием огромного количества компьютерных машин разного класса и уровня, выполняемых ими возможностей. Точный учет помогает своевременно продлять лицензии на используемое программное обеспечение; облегчает подбор необходимых комплектующих для ремонта или модернизации компьютерной техники.

Понимание того, как правильно вести учет компьютеров, серверов, периферийного оборудования и программного обеспечения — это основа для эффективного управления ИТ-инфраструктурой. В данном контексте учет компьютерной техники охватывает широкий спектр задач, начиная от регистрации новых устройств и их конфигурации, до контроля за обновлениями, техническим обслуживанием и утилизацией.

**Актуальность темы** – потребность автоматизации процесса учета оборудования в стенах своего технического отдела, начиная персональными компьютерами и заканчивая периферийным оборудованием

**Цель** – разработка модуля информационной системы для упрощения учета компьютерной техники и оборудования

**Объект** – деятельность технического отдела, занимающегося учетом компьютерной техники и оборудования

**Предмет** – модуль информационной системы для технического отдела

**Для достижения цели, требуется выполнить следующие задачи:**

1. Провести предварительное планирование и проверку осуществимости.
2. Определить требования к программному продукту.
3. Разработать прототип приложения.
4. Реализовать приложение в выбранной среде IDE.
5. Разместить проект на веб-сервисе GitHub или аналогичном.

# Предварительное планирование и проверка осуществимости

# 1.1. Требования к системе в целом

Требования к структуре и функционированию системе

Система должна состоять из сервера баз данных, приложения.

Сервер БД отвечает за хранение базы данных. Приложение организует обмен данными с сервером баз данных. Приложение – основная часть системы – обрабатывает данные полученные от сервера БД и выводит их в понятном для конечного пользователя виде.

Требования к численности и квалификации персонала

Пользователем приложения является администратор. Сотрудники используют систему локально, поэтому количество людей, желающих просмотреть информацию в приложении, ограничено.

Обслуживающий персонал:

• администратор системы и сервера БД – 1 человек;

• персонал технического отдела – 1 человек.

Пользователи должны иметь минимальные навыки работы на компьютере.

Обслуживающий персонал должен иметь навыки работы на компьютере в среде Microsoft Windows (или другой операционной системе) и в глобальной сети Internet. Персонал технического отдела должен иметь опыт работы с MS SQL, Visual Studio 19 и выше, а также знать язык программирования C#.

Требования к надежности

Программное обеспечение системы должно функционировать и не приводить к зависанию системы при любых ошибочных действиях операторов в среде данного программного обеспечения.

Требования к эргономике и технической эстетике

Программное обеспечение и технические средства должны иметь эстетически привлекательный вид и быть удобными в использовании.

Требования к эксплуатации и техническому обслуживанию

Для нормальной работы системы необходимо произвести ряд действий

На компьютере:

• установить лицензионную копию ОС;

• установить копию приложения;

• запустить программу.

На сервере базы данных:

• установить лицензионную копию ОС;

• установить копию SQL Server;

• импортировать базу данных;

• запустить программу установки сервера БД.

Требования к допустимым площадям размещения ИС, к параметрам сети энергоснабжения и т.д. определяются требованиями использующегося аппаратного обеспечения.

Требования к защите информации от несанкционированного доступа

Защита ИС и информации, с которой она работает, от несанкционированного доступа возлагается на операционную систему и специализированные программные и программно–аппаратные продукты.

Требования по сохранности информации при авариях

Сохранность данных должна быть обеспеченна:

• при сбоях в работе оборудования сети в любой момент времени;

• при сбоях электропитания или программного обеспечения

Требования к защите от влияния внешних воздействий

Защита ИС и информации, с которой она работает, от несанкционированного доступа возлагается частично на операционную систему и специализированные программные и программно–аппаратные продукты, частично на систему аутентификации, реализованную в компонентах ИС.

Требования к стандартизации и унификации

Автоматизированная система должна предоставлять пользователю привычный, общепринятый в среде Microsoft Windows интерфейс. Программная документация, поставляемая с программой, должна быть оформлена в соответствии со стандартом ЕСПД.

# 1.2. Требования к функциям, выполняемым системой

Функции подсистемы «Сервер базы данных»:

• получение запросов от «клиента БД», расположенного на SQL Server;

• проверка легальности выполнения запроса на основе идентификационной информации, вложенной в запрос;

• выполнение SQL – запросов к БД;

• передача результата SQL – запроса «клиенту БД».

Функции подсистемы «Приложение»:

1. Передача Администратору следующей информации:

• информация об оборудовании (тип, модель, серийный номер);

• информация о списании и перемещении оборудования

2. Изменение информации:

• изменение информации, хранящейся в разделах;

• формирование и передача запросов серверу БД;

• получение от сервера БД результатов запросов.

# 1.3. Требования к видам обеспечения

Требования к математическому обеспечению

В основе разрабатываемого ПО должна лежать реляционная модель базы данных. Все таблицы должны находиться в третьей или четвертой нормальной форме. Связи между таблицами должны быть вида: «один к одному» или «один ко многим». При проектировании используется метод «сущность – связь».

Требования к информационному обеспечению

Данные в модуле информационной системы должны быть организованны в виде связанных таблиц. Хранение и доступ к данным осуществляется с применением реляционные систем управления базами данных. Информационный обмен между компонентами системы осуществляется по технологии клиент-сервер. Желательно обеспечить возможность резервного копирования и восстановления данных.

Требования к лингвистическому обеспечению

В качестве языка программирования используются – C#, SQL Server, взаимодействие пользователя и программного обеспечения осуществляется через оконный интерфейс модуля информационной системы.

Требования к программному обеспечению

Система должна иметь следующее программное обеспечение: ОС Windows 10 или выше, SQL Server, библиотека NET Framework 4.8.

Требования к техническому обеспечению

Для работоспособности приложения необходимы следующие программные и технические средства:

• клавиатура;

• манипулятор мышь;

• процессор Intel Core i3 2200 и выше;

• оперативная память 2 ГБ и более;

• видеокарта 2 ГБ и более,

• монитор, поддерживающего разрешение 1200х700 точек и больше при глубине цвета 16 бит и более;

• дисковое пространство 5 ГБ и более;

# 1.4. Сравнение и анализ аналогов системы

В представленной таблице 1.1 проведен анализ и сравнение аналогов системы по ряду преимуществ и недостатков.

Таблица 1.1 – Сравнение и анализ аналогов системы

|  |  |
| --- | --- |
| Преимущества | Недостатки |
| Network Inventory Advisor | |
| Network Inventory Advisor Price, Reviews & Ratings - Capterra United Arab  Emirates 2024  Рисунок 1.1 – Интерфейс системы «Network Inventory Advisor» | |
| 1. Автоматическое обнаружение оборудования  2. Гибкость отчетности  3. Легкость использования | 1. Сложность настройки  2. Является платным ПО  3. Определенные требования к оборудованию |

Продолжение таблицы 1.1

|  |  |
| --- | --- |
| Spiceworks | |
| 1 IT Network Inventory - Track your network. 100% free  Рисунок 1.2 – Интерфейс системы «Spiceworks» | |
| 1. Предоставляет базовый инструментарий для управления информационной инфраструктурой бесплатно  2. Имеет возможность интеграции с другими сервисами, такими как HelpDesk  3. Программа предлагает возможность общения и получения помощи от других специалистов IT | 1. Реклама и спонсорский контент  2. Ограниченный функционал  3. Возможные проблемы с конфиденциальностью данных  4. Поддерживаемый язык только Английский |
| Мое приложение | |
| 1. Отсутствует реклама  2. Полный базовый функционал для инвентаризации  3. Легко настроить  4. Является бесплатным ПО | 1. Может быть установлена только на Windows-системы  2. Ограниченная персонализация |

В ходе анализа и сравнения схожих информационных систем мною был сделан вывод, что для разработки модуля информационной системы учета компьютерной техники и оборудования необходимо учесть следующие факторы.

Во-первых, из приведенных выше преимуществ программ следует, что идеальная система управления информационной инфраструктурой должна обеспечивать бесплатный базовый функционал, а также легко настраиваться

Во-вторых, интерфейс играет важную роль в удобстве использования. «Spiceworks» и «Network Inventory Advisory» предлагают сравнительно схожие интерфейсы, с уклоном в стандартный для учета оборудования дизайн. Мое приложение может выделиться простым и понятным интерфейсом, что делает его хорошим выбором для пользователей, предпочитающих легкость в использовании.

Исходя из сравнение представленных информационных систем, я сделал вывод, что для реализации модуля учета компьютерной техники и оборудования стоит уделить внимание на следующие детали:

• адаптивность и мобильность интерфейса;

• безопасность данных и защита конфиденциальной информации;

• удобство использования, автоматизация и интеграция с другими сервисами;

# 1.5. Инфологическая модель предметной области

Информационно-логическая модель (ИЛМ) отображает данные предметной области в виде совокупности информационных объектов и связей между ними. Для построения инфологической модели (концептуальной схемы) используют традиционные методы агрегации и обобщения. При агрегации объединяются информационные объекты (элементы данных) в один в соответствии с семантическими связями между объектами.

Администратор заполняет список перемещений оборудования. Список включает в себя оборудование, каждое из которых указывается какой тип оборудования, также пишется модель, дата, цена, а также присваивается инвентарный и серийные номер устройства.

Администратор может существовать без списка перемещений. Однако его невозможно заполнить без указания сотрудника и рабочего места. Администратор может заполнить неограниченное количество списков.

Инфологическая модель изображена на рисунке 1.3.

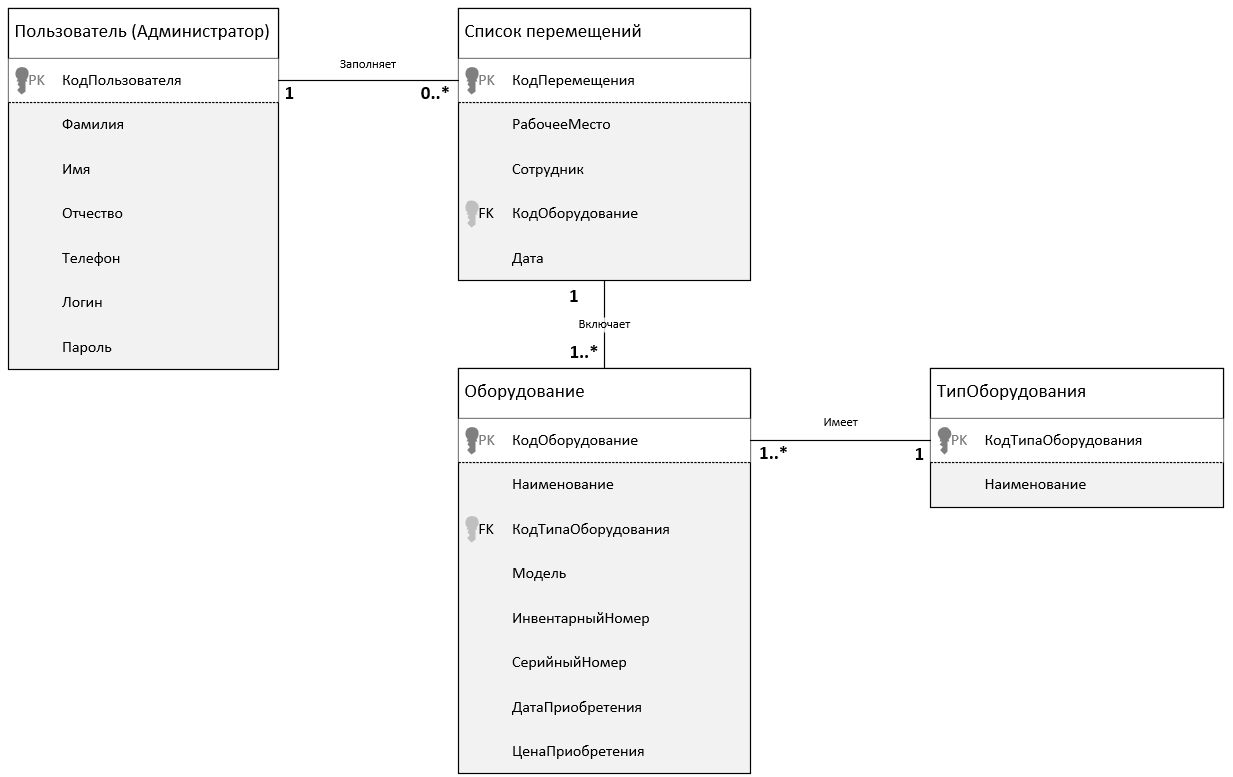


Рисунок 1.3 – Инфологическая модель

# 1.6. Прототипирование интерфейса

Модуль информационной системы будет состоять из следующих компонентов и страниц:

• окно авторизации;

• страница главного меню

• страница оборудование;

• страница тип оборудования;

• страница место установки;

• страница список сотрудников;

• страница рабочие места;

• страница о перемещении оборудования;

• страница о списании оборудования.

Страница авторизации (рисунок 1.4) является заготовкой для будущего разрабатываемого функционала, улучшения приложения. На ней содержатся поля для ввода логина и пароля, фон, логотип, а также кнопка входа.

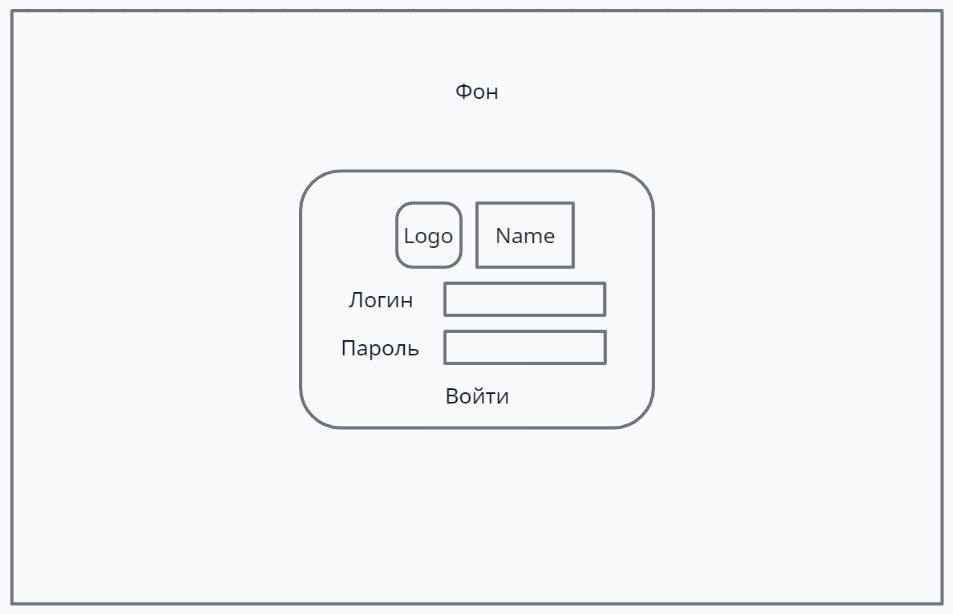


Рисунок 1.4 – Прототип страницы «Авторизация»

Главное окно на рисунке 1.5 содержит логотип, название, Frame для смены страниц. На вертикальном меню находятся кнопки для перехода к спискам.

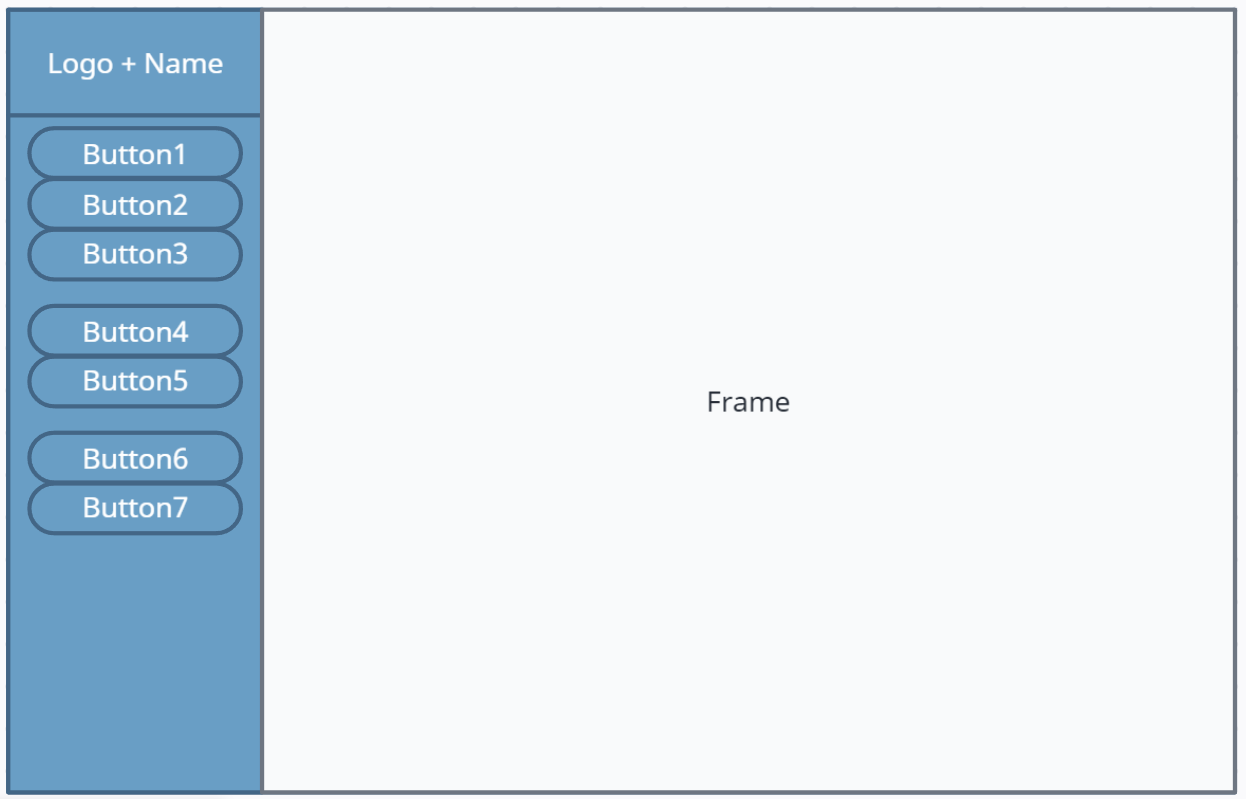


Рисунок 1.5 – Прототип страницы «Главное меню»

Прототип, который будет использоваться на всех страницах изображен на рисунке 1.6. На них содержатся вывод информации в таблицы и кнопки редактирования (Р), добавления (Д) и удаления данных (У), а также их поиск.

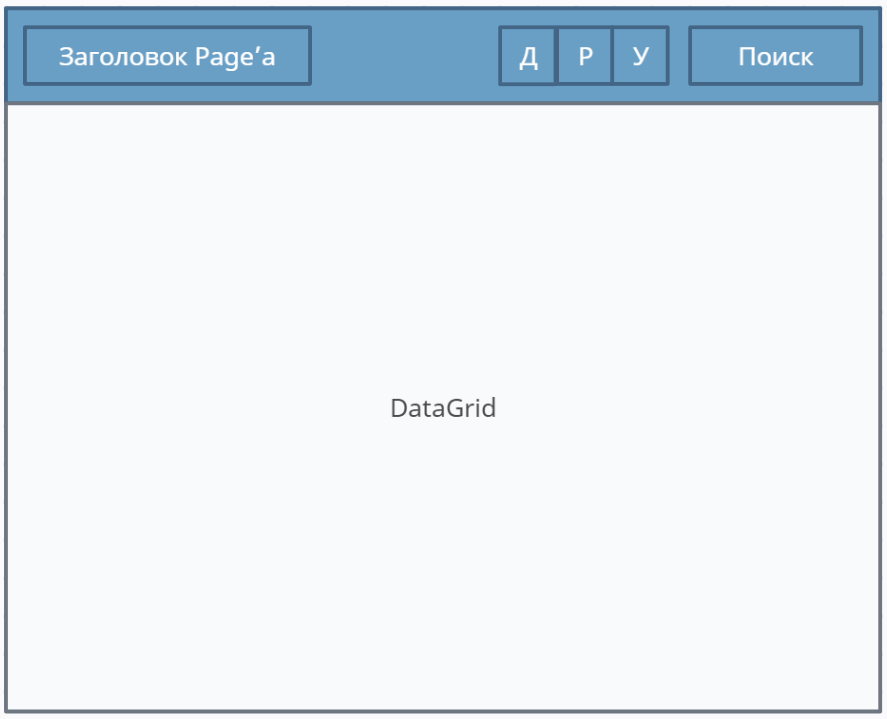


Рисунок 1.6 – Прототип для всех страниц в системе

Прототип страницы добавления и редактирования оборудования, показанный на рисунке 1.7. имеет поля с выпадающим списком для выбора статуса и типа. Ниже находятся кнопки сохранения и отмены добавления или редактирования единицы оборудования.

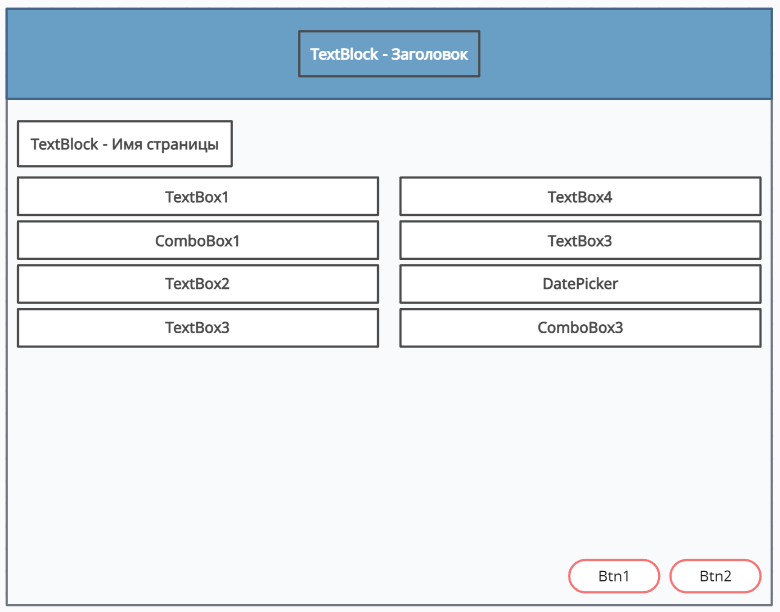


Рисунок 1.7 – Прототип страницы добавления и редактирования оборудования

# 2. Практическая часть

# 2.1. Назначение и цели создания системы

Полное наименование программного обеспечения и его условное обозначение

Полное наименование – Разработка модуля информационной системы для учета компьютерной техники и оборудования – «PC Inventory».

Назначение системы

Назначения создания модуля информационной системы для технического отдела учета компьютерной техники и оборудования заключается в упрощении и автоматизация процесса получения информации о состоянии компьютерного оборудования, его распределении, использовании и технической поддержке

Цели создания системы

• сокращение времени и затрат на учет и административные процессы;

• повышение оптимизация обслуживания и ремонта оборудования;

• повышение уровня доступности к предоставленной информации относительно установленного оборудования.

# 2.2. Характеристики объекта автоматизации

Краткие сведения об объекте автоматизации

Объекты автоматизации:

• процесс получения информации об оборудовании;

• процесс получения информации о сотрудниках

• процесс получения информации о перемещении оборудования;

• процесс изменения информации о перемещении оборудования;

В процессе выдачи информации о оборудовании должны отображаться следующие данные:

• наименование оборудования;

• полное описание оборудования.

• характеристики оборудования.

В процессе выдачи информации о сотрудниках, имеющих отношение к техническому отделу, отображаются следующие данные:

• название или Ф.И.О;

• роль;

• телефон,

• e-mail;

• логин пользователя.

Сведения об условиях эксплуатации объекта автоматизации и характеристик окружающей среды

Условия эксплуатации объекта автоматизации (на стороне сервера БД):

• операционная система MS Windows 10 и выше;

• SQL Server;

Условия эксплуатации объекта автоматизации (на стороне пользователя):

• операционная система MS Windows 10 и выше;

• SQL Server.

Физические условия эксплуатации и характеристики окружающей среды определяются соответствующими характеристиками окружающей среды используемого аппаратного обеспечения, указанными в документации на него. Дополнительно, важно учитывать и адаптировать физические условия эксплуатации в соответствии с требованиями производителя данного оборудования для обеспечения его оптимальной производительности и долговечности.

# 2.3. Обзор инструментальных средств создания продукта

Для реализации проекта можно воспользоваться следующими средствами:

Visual Studio – представляет собой полностью интегрированную среду разработки. Она спроектирована таким образом, чтобы делать процесс написания кода, его отладки и компиляции в сборку для поставки конечным потребителям как можно более простым. Является наиболее функционально насыщенной IDE-средой, в которой можно разрабатывать приложения на C#. Данный продукт поставляется с конструкторами графических пользовательских интерфейсов, с возможностями работы с базами данных, с поддержкой фрагментов кода, с возможностями для просмотра всего проекта в целом, с просмотром свойств объектов.

Standard Widget Toolkit – библиотека графических пользовательских элементов с открытым исходным кодом для разработки интерфейсов пользователя на языке Java. Предоставляет из себя переносимый API для разработчиков Java. Использует собственные библиотеки операционной системы через собственный интерфейс Java (Java Native Interface) для визуализации элементов графического интерфейса, аналогично программам, написанным с использованием собственного API операционной системы.

Qt Creator – кроссплатформенная свободная IDE для разработки на С, С++ и QML с помощью фреймворка Qt. Разработана Trolltech для работы с фреймворком Qt. Включает в себя графический интерфейс отладчика и визуальные средства разработки интерфейса как с использованием QtWidgets, так и QML.

CLion – продукт JetBrains. Многофункциональная IDE. В которой поддерживается разработка кода на C и C++, CSS, JavaScript, XML.. Основным преимуществом является статический и динамический анализ с безопасный рефакторингом кода. Позволяет кастомизировать редактор так, как удобно пользователю.

# 2.4. Обоснование выбора средств реализации ИС

Приложения можно реализовать, используя средства 1С: Предприятие и Visual Studio.

1С: Предприятие представляет собой систему прикладных решений, созданных на единой технологической платформе с общими принципами.

Visual Studio 2022 – это комплексный набор инструментов для разработки программного обеспечения, включая планирование, создание пользовательского интерфейса, разработку клиентских и серверных приложений, написание кода, тестирование, отладку и другие функции.

Выбор языка программирования C# для реализации информационной системы (ИС) может быть обоснован следующими причинами:

Поддержка платформы .NET: C# является основным языком программирования для разработки приложений на платформе .NET. Платформа .NET обеспечивает мощные инструменты и библиотеки для разработки ИС, включая возможности для работы с базами данных, сетевыми протоколами и другими важными компонентами.

Я решил использовать Visual Studio для разработки, потому что она обеспечивает высокое качество кода на всех этапах жизненного цикла программного обеспечения, от проектирования до внедрения. После изучения Windows Forms и WPF я выбрал WPF, поскольку он предоставляет мощный API для создания настольных графических приложений с богатым дизайном и интерактивностью.

Для разработки на WPF можно использовать любой язык, совместимый с .NET: C#, F#, VB.NET, C++, Ruby, Python, Delphi (Prism), Lua и другие. Мой выбор пал на C#, так как он прост в освоении, современен, многопоточен и безопасен, что делает его языком нового поколения в программировании. Кроме того, C# обладает богатой экосистемой инструментов и библиотек, что облегчает разработку и поддержку приложений на платформе WPF.

# 2.5. Описание логики функционирования ПО

В ходе разработки курсового проекта для приложения была разработана база данных, ЕR-диаграмма которой представлена на рисунке 2.1.

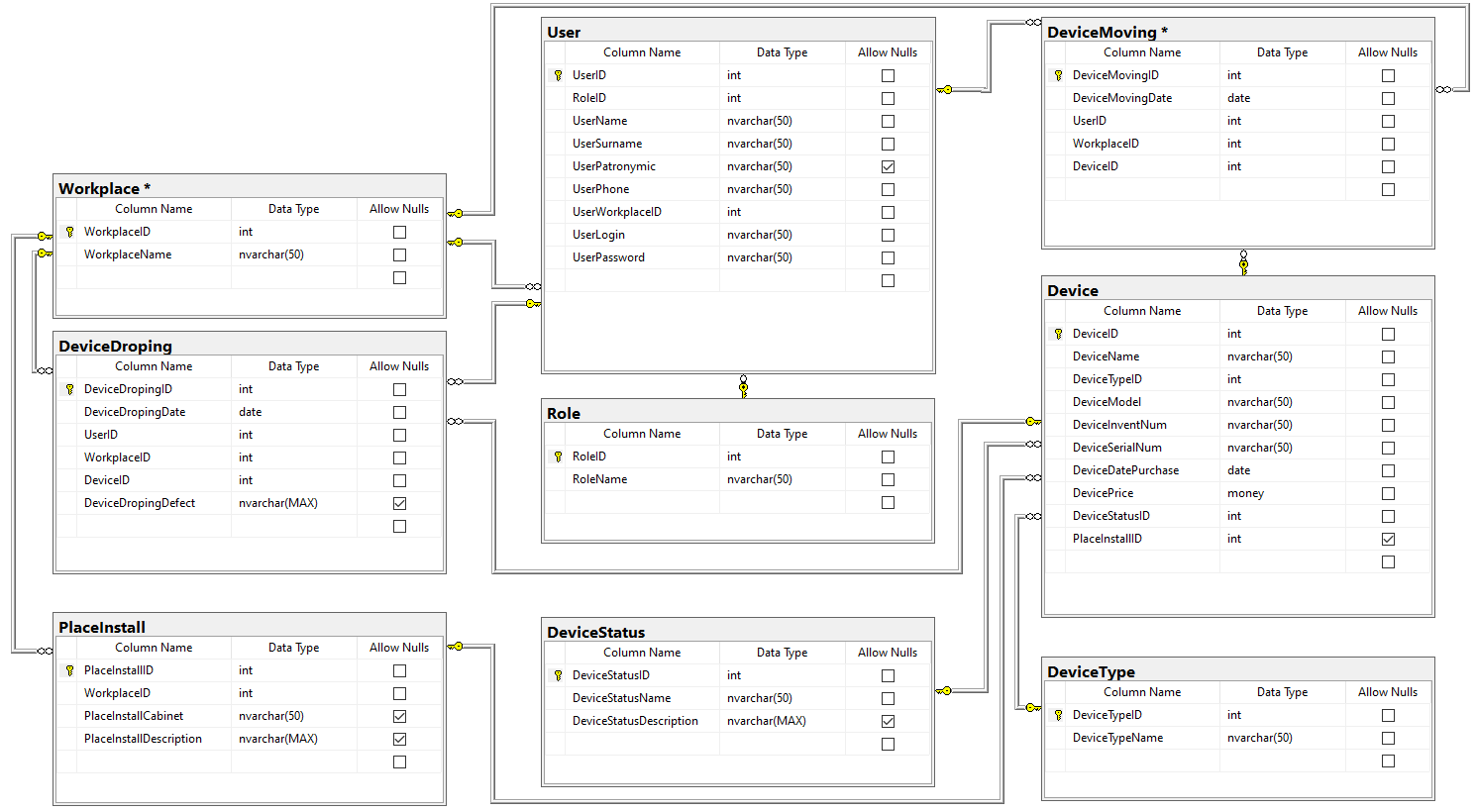


Рисунок 2.1 – ER-диаграмма

Таблица «Device» хранит информацию об оборудовании. Данные таблицы «Device» изображены на рисунке 2.2.

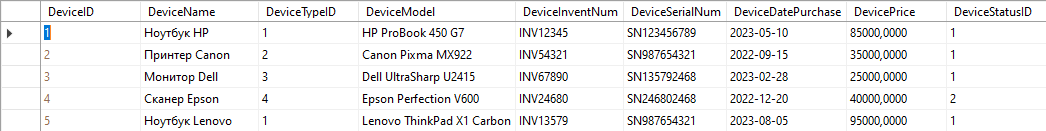


Рисунок 2.2 – Данные таблицы «Device»

Таблица 2.1 – Описание структуры «Device»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип данных** | **Описание** |
| DeviceID | int | Идентификатор оборудования |
| DeviceName | nvarchar(50) | Наименование оборудования |
| DeviceTypeID | int | Идентификатор типа оборудования |
| DeviceModel | nvarchar(50) | Модель оборудования |
| DeviceInventNum | nvarchar(50) | Инвентарный номер оборудования |
| DeviceSerialNum | nvarchar(50) | Серийный номер оборудования |
| DeviceDatePurchase | nvarchar(50) | Дата приобретения оборудования |
| DevicePrice | nvarchar(50) | Цена приобретения оборудования |
| DeviceStatusID | int | Идентификатор статуса оборудования |

Таблица «DeviceType» хранит информацию о типе оборудования. Данные таблицы «DeviceType» изображены на рисунке 2.3.

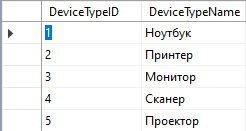


Рисунок 2.3 – Данные таблицы «DeviceType»

Таблица 2.2 – Описание структуры «DeviceType»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип данных** | **Описание** |
| DeviceTypeID | int | Идентификатор типа оборудования |
| DeviceTypeName | nvarchar(50) | Наименование типа оборудования |

Таблица «DeviceStatus» хранит информацию о статусе оборудования и его значения. Данные таблицы «DeviceStatus» изображены на рисунке 2.4.

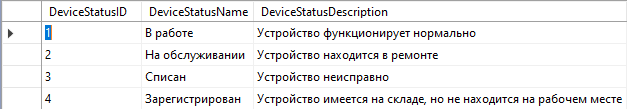


Рисунок 2.4 – Данные таблицы «DeviceStatus»

Таблица 2.3 – Описание структуры «DeviceStatus»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип данных** | **Описание** |
| DeviceStatusID | int | Идентификатор статуса оборудования |
| DeviceStatusName | nvarchar(50) | Наименование статуса оборудования |
| DeviceStatusDescription | nvarchar(MAX) | Описание статуса оборудования |

Таблица «DeviceMoving» хранит информацию о перемещении оборудования, например, дата перемещения, исполнитель и имя оборудования. Данные таблицы «DeviceMoving» изображены на рисунке 2.4.

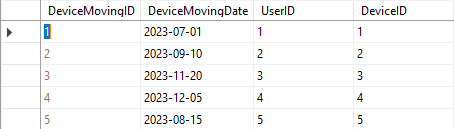


Рисунок 2.5 – Данные таблицы «DeviceMoving»

Таблица 2.4 – Описание структуры «DeviceMoving»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип данных** | **Описание** |
| DeviceMovingID | int | Идентификатор перемещения обор-ия |
| DeviceMovingDate | date | Дата перемещения оборудования |
| UserID | int | Идентификатор пользователя |
| DeviceID | int | Идентификатор оборудования |

Таблица «DeviceDroping» хранит информацию о списании оборудования, например, дата списания, ответственный за рабочее место, причина и т. д. Данные таблицы «DeviceDroping» изображены на рисунке 2.6.

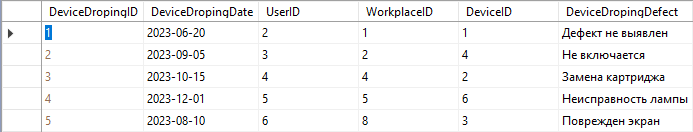


Рисунок 2.6 – Данные таблицы «DeviceDroping»

Таблица 2.5 – Описание структуры «DeviceDroping»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип данных** | **Описание** |
| DeviceDropingID | int | Идентификатор списания обор-ия |
| DeviceDropingDate | date | Дата списания оборудования |
| UserID | int | Идентификатор пользователя |
| WorkplaceID | int | Идентификатор рабочего места (отдела) |
| DeviceID | int | Идентификатор оборудования |
| DeviceDropingDefect | nvarchar(MAX) | Причина списания оборудования |

Таблица «Workplace» хранит список рабочих мест (отделов, кабинетов) в техническом отделе. Данные таблицы «Workplace» изображены на рисунке 2.7.

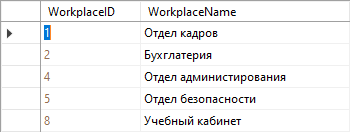


Рисунок 2.7 – Данные таблицы «Workplace»

Таблица 2.6 – Описание структуры «Workplace»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип данных** | **Описание** |
| WorkplaceID | int | Идентификатор рабочего места (отдела) |
| WorplaceName | nvarchar(50) | Наименование рабочего места (отдела) |

Таблица «PlaceInstall» хранит информацию о месте установки оборудования. Данные таблицы «PlaceInstall» изображены на рисунке 2.8.

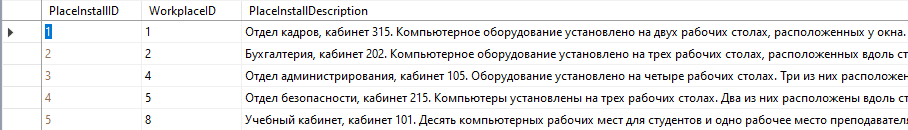


Рисунок 2.8 – Данные таблицы «PlaceInstall»

Таблица 2.7 – Описание структуры «PlaceInstall»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип данных** | **Описание** |
| PlaceInstallID | int | Идентификатор места установки |
| WorkplaceID | int | Идентификатор рабочего места (отдела) |
| PlaceInstallDescription | nvarchar(MAX) | Описание места установки |

Таблица «User» хранит информацию о пользователях, например, должность, ФИО, логин, пароль и т.д. Данные таблицы «User» изображены на рисунке 2.8.



Рисунок 2.9 – Данные таблицы «User»

Таблица 2.8 – Описание структуры «User»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип данных** | **Описание** |
| UserID | int | Идентификатор пользователя |
| RoleID | int | Идентификатор роли (должности) |
| UserName | nvarchar(50) | Имя пользователя |
| UserSurname | nvarchar(50) | Фамилия пользователя |
| UserPatronymic | nvarchar(50) | Отчество пользователя |
| UserPhone | nvarchar(50) | Телефон пользователя |
| UserWorkplaceID | int | Рабочее место пользователя |
| UserLogin | nvarchar(50) | Логин пользователя |
| UserPassword | nvarchar(50) | Пароль пользователя |

Таблица «Role» хранит информацию о роли (должности) пользователя в системе. Данные таблицы «Role» изображены на рисунке 2.10.

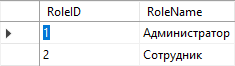


Рисунок 2.10 – Данные таблицы «Role»

Таблица 2.9 – Описание структуры «Role»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип данных** | **Описание** |
| RoleID | int | Идентификатор роли (должности) |
| RoleName | nvarchar(50) | Наименование роли (должности) |

# 2.6. Реализация интерфейса ПО

Входе разработки курсового проекта, был разработан логотип для моей информационной модели. Итоговый логотип, использованный в системе показан на рисунке 2.11.



Рисунок 2.11 – Логотип программы «PC Inventory»

Также для приложения была выбрана цветовая модель, показанная на таблице 2.10. В качестве основного фона используется белый цвет, а в качестве дополнительного: RGB (83, 105, 158).

Для акцентирования внимания пользователя на целевое действие интерфейса использован цвет RGB (49, 44, 77)

Таблица 2.10 – Цветовая модель «PC Inventory»

|  |  |  |
| --- | --- | --- |
| Основной фон | Дополнительный цвет | Акцентирование внимания |
| RGB  #FFFFFFFF  (255, 255, 255) | RGB  #FF53699E  (83, 105, 158) | RGB  #FF312C4D  (49, 44, 77) |
|  |  |  |

При запуске приложения открывается главное окно со страницей авторизации показанная на рисунке 2.12. В верхней части расположены логотип и название, ниже имеются поля ввода логина и пароля, а также кнопка для входа в систему

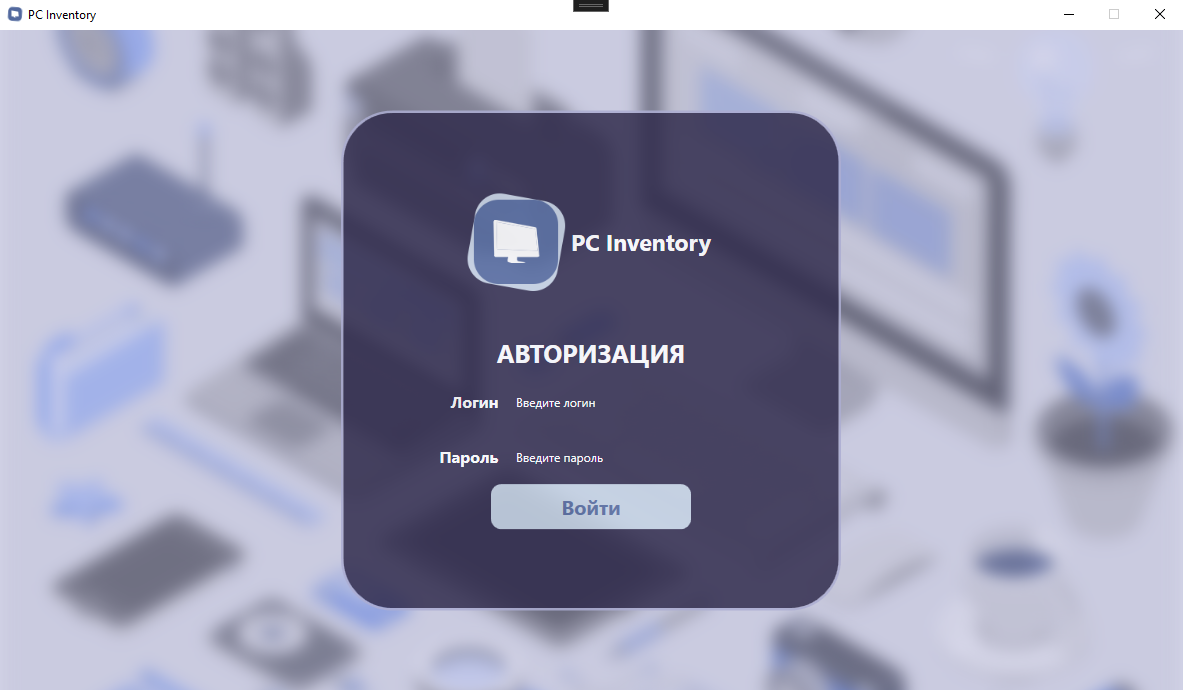


Рисунок 2.12 – Главное окно и страница «Авторизации»

При входе в систему, пользователя встречает главное меню приложения с предпоказанным списком оборудования, изображенное на рисунке 2.13, на этой странице реализованы кнопки для перехода по спискам программы



Рисунок 2.13 – Страница «Главное меню»

После нажатия на кнопку «Оборудование» открывается список оборудования, изображенный на рисунке 2.14, с нее можно перейти к редактированию или добавлению оборудования, а также удалить существующий из списка.



Рисунок 2.14 – Страница списка «Оборудование»

На странице «Оборудование» при нажатии на кнопку редактирование открывается страница для изменения единицы оборудования изображенная на рисунке 2.15

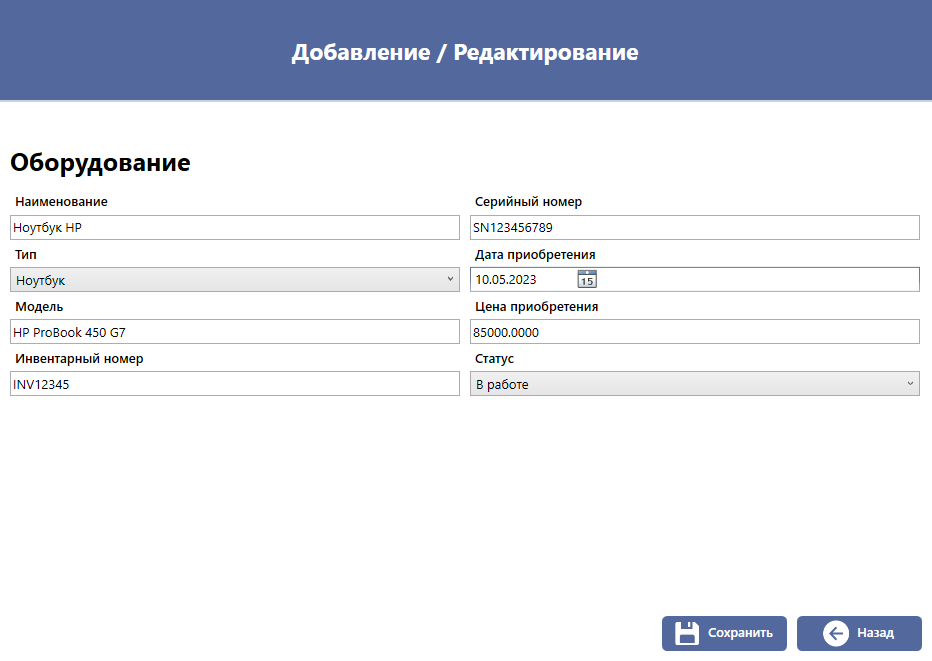


Рисунок 2.15 – Страница редактирования «Оборудование»

# 2.7. Написание программного кода

Соединение с базой данных происходит с помощью метода GetContext(), представленного в листинге 2.1.

Листинг 2.1 “DataBase.Context.cs”

private static DataBaseEntities \_context; public static DataBaseEntities GetContext() { if (\_context == null) \_context = new DataBaseEntities(); return \_context; }

Данный метод, представленный в листинге 2.2 обеспечивает базовую аутентификацию пользователя и обработку возможных ошибок ввода данных

Листинг 2.2 PageAuth.xaml.cs

StringBuilder errors = new StringBuilder(); if (string.IsNullOrEmpty(TxBoxLogin.Text)) errors.AppendLine("Поле \"Логин\" должно быть заполнено!"); if (string.IsNullOrEmpty(PsBoxPassword.Password)) errors.AppendLine("Поле \"Пароль\" должно быть заполнено!"); if (errors.Length > 0) { MessageBox.Show(errors.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error); PsBoxPassword.Password = string.Empty; } else { User \_currentUser = DatabaseEntities.GetContext().User.FirstOrDefault(x => x.UserLogin == TxBoxLogin.Text || x.UserPassword == PsBoxPassword.Password); if (\_currentUser != null) { Manager.AuthUser = \_currentUser; MessageBox.Show($"Здраствуйте, {Manager.AuthUser.FullName}!", "Информация", MessageBoxButton.OK, MessageBoxImage.Information); Manager.FrameMain.Navigate(new MainPage()); } else { MessageBox.Show("Пользователь с такими данными не найден!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error); } }

Перед занесением в базу данных данные проверяются на корректность, представленного в листинге 2.3.

Листинг 2.3 PageEditAddDevice.xaml.cs

StringBuilder errors = new StringBuilder(); if (string.IsNullOrEmpty(TxbDeviceName.Text)) errors.AppendLine("Укажите наименование оборудования!"); if (CmbDeviceType.SelectedItem == null) errors.AppendLine("Выберите тип оборудования!"); if (string.IsNullOrEmpty(TxbDeviceModel.Text)) errors.AppendLine("Укажите модель оборудования!"); if (string.IsNullOrEmpty(TxbDeviceInventNum.Text)) errors.AppendLine("Укажите инвентарный номер оборудования!"); if (string.IsNullOrEmpty(TxbDeviceSerialNum.Text)) errors.AppendLine("Укажите серийный номер оборудования!"); if (CmbDeviceStatus.SelectedItem == null) errors.AppendLine("Укажите статус оборудования!"); if (DtpDeviceDatePurchase.SelectedDate == null) errors.AppendLine("Укажите дату приобретения оборудования!"); if (string.IsNullOrEmpty(TxbDevicePrice.Text)) { errors.AppendLine("Укажите цену приобретения!"); } else { // Пытаемся преобразовать введенный текст в число if (!decimal.TryParse(TxbDevicePrice.Text, out decimal totalPrice)) { errors.AppendLine("Укажите корректную цену!"); } } if (errors.Length > 0) { MessageBox.Show(errors.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error); return; }

Данный метод, представленный на листинге 2.4 обеспечивает возможность удаления выбранных элементов из «DataGridDevice», с запросом подтверждения и обработкой возможных ошибок в процессе.

Листинг 2.4 PageDevice.xaml.cs

var ObjectForDelete = DataGridDevice.SelectedItems.Cast<Device>().ToList(); if (MessageBox.Show($"Вы действительно хотите удалить {ObjectForDelete.Count()} элемент", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes){ try { DatabaseEntities.GetContext().Device.RemoveRange(ObjectForDelete); DatabaseEntities.GetContext().SaveChanges(); MessageBox.Show("Данные удалены", "Информация", MessageBoxButton.OK, MessageBoxImage.Information); DataGridDevice.ItemsSource = DatabaseEntities.GetContext().Device.ToList(); } catch (Exception ex) { MessageBox.Show(ex.Message.ToString()); } }

# 2.8. Защита информации

Защита информации – важный и обязательный пункт при создании программного обеспечения. С этой задачей рано или поздно придется столкнуться каждому разработчику приложений. Каждая составляющая должна быть надежно защищена.

**Проверка целостности данных**

Все запросы взаимодействия приложения с базой данных проходят через обработчиков, которые осуществляют проверку целостности и корректности данных во избежания каких-либо ошибок.

**Безопасность Windows**

Ядро Windows предоставляет разные функции безопасности, которые формируют основу безопасности для всех приложений Windows, включая приложения, созданные с помощью WPF. Windows обеспечивает защиту путем перекомпиляции многих системных библиотек ядра, включая все зависимости WPF, такие как CLR, чтобы снизить вероятность переполнения буфера. В общих чертах, флаг компилятора -GS защищает от некоторых потенциальных переполнений буфера путем добавления специального файла cookie безопасности для защиты адреса функции, которая содержит буферы локальных строк. После возврата функции cookie безопасности сравнивается со своим предыдущим значением. Если значение изменилось, то возможно, произошло переполнение буфера, и процесс будет остановлен с состоянием ошибки. Остановка процесса предотвращает выполнение потенциально вредоносного кода.

**Развертывание ClickOnce**

ClickOnce — это комплексная технология развертывания, которая входит в состав .NET Framework и интегрируется с Visual Studio.

Приложения, развернутые с помощью ClickOnce, приобретают дополнительный уровень безопасности посредством управления доступом для кода. Приложения, развернутые с помощью ClickOnce, запрашивают разрешения, которые им необходимы. Им предоставляются только разрешения, не расширяющие набор разрешений для зоны, в которой развертывается приложения. Таким образом, уменьшается вероятность повреждений на клиентском компьютере в случае перехвата приложения.

**Резервное копирование базы данных (Backup)**

Регулярно создаются резервные копии всех данных модуля информационной системы, включая базы данных, файлы конфигурации и контент. Это обеспечивает возможность восстановления информации в случае ее случайного удаления, атаки злоумышленников или других инцидентов, которые могут привести к потере данных. Резервные копии хранятся на отдельных защищенных серверах или в облачном хранилище с ограниченным доступом только для авторизованных сотрудников. Проводится регулярная проверка работоспособности процесса создания резервных копий и их целостности, чтобы гарантировать их доступность в случае необходимости восстановления данных.

# Заключение

В ходе выполнения курсового проекта был разработан модуль информационной системы «PC Inventory» для технического отдела занимающиеся учетом компьютерной техники и оборудования, также была спроектирована база данных и ее схема.

Разработанный модуль информационной системы позволит упростить и автоматизировать процесс получения информации о состоянии компьютерного оборудования, его распределении, использовании и технической поддержке

Также закреплены теоретические знания и практические навыки работы с SQL Server, C# и XAML, умение работать с литературой, анализировать источники, делать обоснованные выводы.

Полученные теоретические и практические навыки будут использованы в дальнейшем обучении и также в написании дипломной работы в будущем. Провел предварительное планирование и проверку осуществимости.

В рамках курсового проекта были выполнены следующие цели

• сокращение времени и затрат на учет и административные процессы;

• повышение оптимизация обслуживания и ремонта оборудования;

• повышение уровня доступности к предоставленной информации относительно установленного оборудования.

А также поставленные задачи:

1. Определил требования к программному продукту.
2. Разработал прототип приложения.
3. Реализовал приложение в выбранной среде IDE.
4. Создал сборку для установки.
5. Разместил проект на веб-сервисе GitHub или аналогичном –https://github.com/YoulasXXX/PCInventory.git

# Список использованных источников

1. Казанский, А. А. Программирование на Visual C# : учебное пособие для среднего профессионального образования / А. А. Казанский. — 2-е изд., переработан и дополнен — Москва: Издательство Юрайт, 2024. — 192 с. — (Профессиональное образование). — ISBN 978-5-534-14130-6. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/538155

2. Тузовский, А. Ф. Объектно-ориентированное программирование: учебное пособие для вузов / А. Ф. Тузовский. — Москва: Издательство Юрайт, 2024. — 213 с. — (Высшее образование). — ISBN 978-5-534-16316-2. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/537332

3. Казанский, А. А. Программирование на Visual C#: учебное пособие для вузов / А. А. Казанский. — 2-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2024. — 192 с. — (Высшее образование). — ISBN 978-5-534-12338-8. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/537364

4. Демин, А. Ю. Информатика. Лабораторный практикум: учебное пособие для вузов / А. Ю. Демин, В. А. Дорофеев. — Москва: Издательство Юрайт, 2024. — 147 с. — (Высшее образование). — ISBN 978-5-534-18259-0. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/534628

5. Сысолетин, Е. Г. Разработка интернет-приложений: учебное пособие для вузов / Е. Г. Сысолетин, С. Д. Ростунцев; под научной редакцией Л. Г. Доронинского. — Москва: Издательство Юрайт, 2022. — 90 с. — (Высшее образование). — ISBN 978-5-9916-9975-4. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/492224 (дата обращения: 05.02.2024).

# Приложение

# Приложение А

**Листинги программного кода**

**App.xaml**

<Application.Resources> <BooleanToVisibilityConverter x:Key="BoolToVis"/> <Style TargetType="Separator"> <Setter Property="Height" Value="35"/> <Setter Property="Background" Value="Transparent"/> </Style> <!--Стиль для страницы "Авторизация"--> <Style TargetType="Border" x:Key="BrdAuth"> <Setter Property="Width" Value="500"/> <Setter Property="Height" Value="500"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="HorizontalAlignment" Value="Center"/> <Setter Property="Background" Value="#312c4d"/> <Setter Property="Opacity" Value="0.85"/> <Setter Property="BorderBrush" Value="#acadd2"/> <Setter Property="BorderThickness" Value="2.5"/> <Setter Property="CornerRadius" Value="50"/> <Setter Property="Margin" Value="0"/> </Style> <Style TargetType="TextBlock" x:Key="TxblTitle"> <Setter Property="HorizontalAlignment" Value="Center"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="Foreground" Value="White"/> <Setter Property="FontFamily" Value="Segoe UI"/> <Setter Property="FontWeight" Value="Bold"/> <Setter Property="FontSize" Value="25"/> <Setter Property="TextAlignment" Value="Center"/> <Setter Property="Margin" Value="5"/> </Style> <Style TargetType="TextBlock" x:Key="TxblLogPass"> <Setter Property="Width" Value="75"/> <Setter Property="TextAlignment" Value="Right"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="FontFamily" Value="Segoe UI"/> <Setter Property="Foreground" Value="White"/> <Setter Property="FontWeight" Value="Bold"/> <Setter Property="FontSize" Value="16"/> <Setter Property="Margin" Value="5"/> </Style> <Style TargetType="Button" x:Key="BtnSignIn"> <Setter Property="Width" Value="200"/> <Setter Property="Height" Value="45"/> <Setter Property="HorizontalAlignment" Value="Center"/> <Setter Property="Background" Value="Transparent"/> <Setter Property="FontSize" Value="20"/> <Setter Property="Foreground" Value="White"/> <Setter Property="FontWeight" Value="Bold"/> <Setter Property="FocusVisualStyle" Value="{x:Null}"/> <Setter Property="Cursor" Value="Hand"/> <Setter Property="Template"> <Setter.Value> <ControlTemplate TargetType="Button"> <Border Background="{TemplateBinding Background}" CornerRadius="10"> <ContentPresenter HorizontalAlignment="Center" VerticalAlignment="Center" Margin="{TemplateBinding Padding}"/> </Border> </ControlTemplate> </Setter.Value> </Setter> <Style.Triggers> <Trigger Property="IsMouseOver" Value="True"> <Setter Property="Background" Value="#FFC5D3E4"/> <Setter Property="Foreground" Value="#53699E"/> </Trigger> <Trigger Property="IsPressed" Value="True"> <Setter Property="Background" Value="#312c4d"/> <Setter Property="Foreground" Value="White"/> </Trigger> </Style.Triggers> </Style> <!--Placeholder для логина и пароля--> <Style TargetType="TextBlock" x:Key="TxBlLogin"> <Setter Property="Margin" Value="5"/> <Setter Property="Width" Value="235"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="Foreground" Value="#FFFFFF"/> <Setter Property="IsHitTestVisible" Value="False"/> <Setter Property="Panel.ZIndex" Value="1"/> <Setter Property="Visibility" Value="{Binding ElementName=TxBoxLogin, Path=Text.IsEmpty, Converter={StaticResource BoolToVis}}"/> </Style> <Style TargetType="TextBox" x:Key="TxBoxLogin"> <Setter Property="Margin" Value="5"/> <Setter Property="Width" Value="250"/> <Setter Property="Height" Value="35"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="VerticalContentAlignment" Value="Center"/> <Setter Property="HorizontalContentAlignment" Value="Left"/> <Setter Property="Background" Value="Transparent"/> <Setter Property="BorderBrush" Value="Transparent"/> <Setter Property="Foreground" Value="#FFFFFF"/> <Setter Property="FontSize" Value="16"/> </Style> <Style TargetType="TextBlock" x:Key="TxBlPassword"> <Setter Property="Margin" Value="5"/> <Setter Property="Width" Value="235"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="Foreground" Value="#FFFFFF"/> <Setter Property="IsHitTestVisible" Value="False"/> <Setter Property="Panel.ZIndex" Value="1"/> <Setter Property="Visibility" Value="Visible"/> </Style> <Style TargetType="PasswordBox" x:Key="PsBoxPassword"> <Setter Property="Margin" Value="5"/> <Setter Property="Width" Value="250"/> <Setter Property="Height" Value="35"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="VerticalContentAlignment" Value="Center"/> <Setter Property="HorizontalContentAlignment" Value="Left"/> <Setter Property="Background" Value="Transparent"/> <Setter Property="BorderBrush" Value="Transparent"/> <Setter Property="Foreground" Value="#FFFFFF"/> <Setter Property="FontSize" Value="16"/> </Style> <!--Стиль главного меню--> <Style TargetType="Border" x:Key="BrdMenu"> <Setter Property="Width" Value="240"/> <Setter Property="BorderBrush" Value="#FFC5D3E4"/> <Setter Property="BorderThickness" Value="2"/> <Setter Property="CornerRadius" Value="10"/> <Setter Property="VerticalAlignment" Value="Top"/> <Setter Property="HorizontalAlignment" Value="Center"/> <Setter Property="Margin" Value="0"/> </Style> <Style TargetType="Label" x:Key="LblMenuTitle"> <Setter Property="Width" Value="200"/> <Setter Property="Height" Value="30"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="HorizontalContentAlignment" Value="Left"/> <Setter Property="VerticalContentAlignment" Value="Center"/> <Setter Property="FontSize" Value="13"/> <Setter Property="Foreground" Value="White"/> <Setter Property="FontWeight" Value="Bold"/> </Style> <Style TargetType="Label" x:Key="LblLogo"> <Setter Property="HorizontalContentAlignment" Value="Left"/> <Setter Property="VerticalContentAlignment" Value="Center"/> <Setter Property="FontSize" Value="23"/> <Setter Property="Foreground" Value="White"/> <Setter Property="FontWeight" Value="Bold"/> </Style> <Style TargetType="TextBlock" x:Key="TxtBlMenu"> <Setter Property="FontSize" Value="13"/> <Setter Property="FontWeight" Value="Medium"/> <Setter Property="TextAlignment" Value="Left"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="Margin" Value="5"/> </Style> <Style TargetType="Image" x:Key="ImgMenu"> <Setter Property="Height" Value="32"/> <Setter Property="Width" Value="32"/> <Setter Property="VerticalAlignment" Value="Center"/> </Style> <Style TargetType="Button" x:Key="BtnMenu"> <Setter Property="Width" Value="240"/> <Setter Property="Height" Value="45"/> <Setter Property="HorizontalAlignment" Value="Center"/> <Setter Property="Background" Value="Transparent"/> <Setter Property="Foreground" Value="White"/> <Setter Property="FocusVisualStyle" Value="{x:Null}"/> <Setter Property="Cursor" Value="Hand"/> <Setter Property="Template"> <Setter.Value> <ControlTemplate TargetType="Button"> <Border Background="{TemplateBinding Background}" CornerRadius="10"> <ContentPresenter HorizontalAlignment="Left" VerticalAlignment="Center" Margin="{TemplateBinding Padding}"/> </Border> </ControlTemplate> </Setter.Value> </Setter> <Style.Triggers> <Trigger Property="IsMouseOver" Value="True"> <Setter Property="Background" Value="#FFC5D3E4"/> <Setter Property="Foreground" Value="#53699E"/> </Trigger> <Trigger Property="IsPressed" Value="True"> <Setter Property="Background" Value="#312c4d"/> <Setter Property="Foreground" Value="White"/> </Trigger> </Style.Triggers> </Style> <!--Стиль для страницы "Оборудование"--> <Style TargetType="Label" x:Key="LblDevices"> <Setter Property="Margin" Value="5"/> <Setter Property="FontSize" Value="23"/> <Setter Property="Foreground" Value="#FFFFFF"/> <Setter Property="FontWeight" Value="Bold"/> <Setter Property="HorizontalContentAlignment" Value="Left"/> <Setter Property="VerticalContentAlignment" Value="Center"/> </Style> <Style TargetType="TextBlock" x:Key="TxBlSearch"> <Setter Property="Margin" Value="5"/> <Setter Property="Width" Value="200"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="Foreground" Value="#FFFFFF"/> <Setter Property="IsHitTestVisible" Value="False"/> <Setter Property="Panel.ZIndex" Value="1"/> <Setter Property="Visibility" Value="{Binding ElementName=TxbSearch, Path=Text.IsEmpty, Converter={StaticResource BoolToVis}}"/> </Style> <Style TargetType="Image" x:Key="ImgSearch"> <Setter Property="Height" Value="20"/> <Setter Property="Width" Value="20"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="HorizontalAlignment" Value="Left"/> <Setter Property="Visibility" Value="{Binding ElementName=TxbSearch, Path=Text.IsEmpty, Converter={StaticResource BoolToVis}}"/> <Setter Property="Margin" Value="7"/> </Style> <Style TargetType="TextBox" x:Key="TxBoxSearch"> <Setter Property="Margin" Value="5"/> <Setter Property="Width" Value="250"/> <Setter Property="Height" Value="35"/> <Setter Property="VerticalAlignment" Value="Center"/> <Setter Property="VerticalContentAlignment" Value="Center"/> <Setter Property="HorizontalContentAlignment" Value="Left"/> <Setter Property="Background" Value="Transparent"/> <Setter Property="BorderBrush" Value="White"/> <Setter Property="Foreground" Value="#FFFFFF"/> <Setter Property="FontSize" Value="13"/> </Style> </Application.Resources>

**PageAuth.xaml**

<Grid> <Image Source="/Resources/ImgBackgroundAuth.png" HorizontalAlignment="Center" VerticalAlignment="Center" Stretch="UniformToFill"> <Image.Effect> <BlurEffect Radius="25"/> </Image.Effect> </Image> <Grid Background="#acadd2" Opacity="0.5"/> <Border Style="{StaticResource BrdAuth}"> <StackPanel HorizontalAlignment="Center" VerticalAlignment="Center"> <WrapPanel VerticalAlignment="Center" HorizontalAlignment="Center"> <Image Source="/Resources/LogoPCI.png" Width="125"/> <Label Content="PC Inventory" Style="{StaticResource LblLogo}"/> </WrapPanel> <Separator/> <TextBlock Text="АВТОРИЗАЦИЯ" Style="{StaticResource TxblTitle}"/> <WrapPanel Margin="5"> <TextBlock Text="Логин" Style="{StaticResource TxblLogPass}"/> <Grid> <TextBlock Text="Введите логин" Style="{StaticResource TxBlLogin}"/> <TextBox x:Name="TxBoxLogin" Style="{StaticResource TxBoxLogin}"/> </Grid> </WrapPanel> <WrapPanel Margin="5"> <TextBlock Text="Пароль" Style="{StaticResource TxblLogPass}"/> <Grid> <TextBlock x:Name="TxBlPassword" Text="Введите пароль" Style="{StaticResource TxBlPassword}"/> <PasswordBox x:Name="PsBoxPassword" PasswordChanged="PsBoxPassword\_PasswordChanged" Style="{StaticResource PsBoxPassword}"/> </Grid> </WrapPanel> <Button x:Name="BtnSingIn" Click="BtnSingIn\_Click" Content="Войти" Style="{StaticResource BtnSignIn}"/> </StackPanel> </Border> </Grid>

**PageAuth.xaml.cs**

public partial class PageAuth : Page { public PageAuth() { InitializeComponent(); } private void PsBoxPassword\_PasswordChanged(object sender, RoutedEventArgs e) { if (PsBoxPassword.Password.Length == 0) { TxBlPassword.Visibility = Visibility.Visible; } else { TxBlPassword.Visibility = Visibility.Collapsed; } } private void BtnSingIn\_Click(object sender, RoutedEventArgs e) { StringBuilder errors = new StringBuilder(); if (string.IsNullOrEmpty(TxBoxLogin.Text)) errors.AppendLine("Поле \"Логин\" должно быть заполнено!"); if (string.IsNullOrEmpty(PsBoxPassword.Password)) errors.AppendLine("Поле \"Пароль\" должно быть заполнено!"); if (errors.Length > 0) { MessageBox.Show(errors.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error); PsBoxPassword.Password = string.Empty; } else { User \_currentUser = DatabaseEntities.GetContext().User.FirstOrDefault(x => x.UserLogin == TxBoxLogin.Text || x.UserPassword == PsBoxPassword.Password); if (\_currentUser != null) { Manager.AuthUser = \_currentUser; MessageBox.Show($"Здраствуйте, {Manager.AuthUser.FullName}!", "Информация", MessageBoxButton.OK, MessageBoxImage.Information); Manager.FrameMain.Navigate(new MainPage()); } else { MessageBox.Show("Пользователь с такими данными не найден!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error); } } } }

**MainPage.xaml**

<Grid> <Grid.ColumnDefinitions> <ColumnDefinition Width="250"/> <ColumnDefinition Width="\*"/> </Grid.ColumnDefinitions> <Grid.RowDefinitions> <RowDefinition Height="100"/> <RowDefinition Height="\*"/> </Grid.RowDefinitions> <Grid Background="#53699E"> <WrapPanel HorizontalAlignment="Center" VerticalAlignment="Center"> <Image Source="/Resources/LogoPCI.png" Width="75"/> <Label Content="PC Inventory" Style="{StaticResource LblLogo}"/> </WrapPanel> </Grid> <Grid Grid.Row="1" Background="#53699E"> <StackPanel Grid.Row="1"> <Border Style="{StaticResource BrdMenu}"> <StackPanel> <Label Content="Техника" Style="{StaticResource LblMenuTitle}"/> <Button x:Name="BtnDevice" Click="BtnDevice\_Click" Style="{StaticResource BtnMenu}"> <StackPanel Orientation="Horizontal" Margin="5"> <Image Source="/Resources/ImgDevice.png" Style="{StaticResource ImgMenu}"/> <TextBlock Text="Оборудование" Style="{StaticResource TxtBlMenu}"/> </StackPanel> </Button> <Button x:Name="BtnDeviceType" Click="BtnDeviceType\_Click" Style="{StaticResource BtnMenu}"> <StackPanel Orientation="Horizontal" Margin="5"> <Image Source="/Resources/ImgDeviceType.png" Style="{StaticResource ImgMenu}"/> <TextBlock Text="Тип оборудования" Style="{StaticResource TxtBlMenu}"/> </StackPanel> </Button> <Button x:Name="BtnPlaceInstall" Click="BtnPlaceInstall\_Click" Style="{StaticResource BtnMenu}"> <StackPanel Orientation="Horizontal" Margin="5"> <Image Source="/Resources/ImgPlaceInstall.png" Style="{StaticResource ImgMenu}"/> <TextBlock Text="Место установки" Style="{StaticResource TxtBlMenu}"/> </StackPanel> </Button> </StackPanel> </Border> <Border Style="{StaticResource BrdMenu}" Margin="5"> <StackPanel> <Label Content="Отделы" Style="{StaticResource LblMenuTitle}"/> <Button x:Name="BtnUser" Click="BtnUser\_Click" Style="{StaticResource BtnMenu}"> <StackPanel Orientation="Horizontal" Margin="5"> <Image Source="/Resources/ImgUser.png" Style="{StaticResource ImgMenu}"/> <TextBlock Text="Сотрудники" Style="{StaticResource TxtBlMenu}"/> </StackPanel> </Button> <Button x:Name="BtnWorkplace" Click="BtnWorkplace\_Click" Style="{StaticResource BtnMenu}"> <StackPanel Orientation="Horizontal" Margin="5"> <Image Source="/Resources/ImgWorkplace.png" Style="{StaticResource ImgMenu}"/> <TextBlock Text="Рабочие места" Style="{StaticResource TxtBlMenu}"/> </StackPanel> </Button> </StackPanel> </Border> <Border Style="{StaticResource BrdMenu}"> <StackPanel> <Label Content="Учет техники" Style="{StaticResource LblMenuTitle}"/> <Button x:Name="BtnDevMoved" Click="BtnDevMoved\_Click" Style="{StaticResource BtnMenu}"> <StackPanel Orientation="Horizontal" Margin="5"> <Image Source="/Resources/ImgDevMoved.png" Style="{StaticResource ImgMenu}"/> <TextBlock Text="Перемещение оборудования" Style="{StaticResource TxtBlMenu}"/> </StackPanel> </Button> <Button x:Name="BtnDevDrop" Click="BtnDevDrop\_Click" Style="{StaticResource BtnMenu}"> <StackPanel Orientation="Horizontal" Margin="5"> <Image Source="/Resources/ImgDevDroped.png" Style="{StaticResource ImgMenu}"/> <TextBlock Text="Списание оборудования" Style="{StaticResource TxtBlMenu}"/> </StackPanel> </Button> </StackPanel> </Border> <Button x:Name="BtnExit" Click="BtnExit\_Click" Style="{StaticResource BtnMenu}"> <StackPanel Orientation="Horizontal" Margin="5"> <Image Source="/Resources/ImgExit.png" Style="{StaticResource ImgMenu}"/> <TextBlock Text="Выход" Style="{StaticResource TxtBlMenu}"/> </StackPanel> </Button> </StackPanel> </Grid> <Frame x:Name="SecondFrame" NavigationUIVisibility="Hidden" Grid.Column="1" Grid.Row="0" Grid.RowSpan="2"> </Frame> <Border Height="700" Width="2" Grid.RowSpan="2" Grid.Column="1" HorizontalAlignment="Left" VerticalAlignment="Top" Background="#FFC5D3E4"/> <Border Height="2" Width="1200" Grid.Row="1" Grid.Column="1" HorizontalAlignment="Left" VerticalAlignment="Top" Background="#FFC5D3E4"/> </Grid>

**MainPage.xaml.cs**

public partial class MainPage : Page { public MainPage() { InitializeComponent(); Manager.SecondFrame = SecondFrame; Manager.SecondFrame.Navigate(new PageDevice()); } private void BtnDevice\_Click(object sender, RoutedEventArgs e) { Manager.SecondFrame.Navigate(new PageDevice()); } private void BtnDeviceType\_Click(object sender, RoutedEventArgs e) { Manager.SecondFrame.Navigate(new PageTypeDevices()); } private void BtnPlaceInstall\_Click(object sender, RoutedEventArgs e) { Manager.SecondFrame.Navigate(new PagePlaceInstall()); } private void BtnUser\_Click(object sender, RoutedEventArgs e) { Manager.SecondFrame.Navigate(new PageUser()); } private void BtnWorkplace\_Click(object sender, RoutedEventArgs e) { Manager.SecondFrame.Navigate(new PageWorkplace()); } private void BtnDevMoved\_Click(object sender, RoutedEventArgs e) { Manager.SecondFrame.Navigate(new PageDevMoving()); } private void BtnDevDrop\_Click(object sender, RoutedEventArgs e) { Manager.SecondFrame.Navigate(new PageDevDroping()); } private void BtnExit\_Click(object sender, RoutedEventArgs e) { MessageBoxResult result = MessageBox.Show("Вы действительно хотите выйти?", "Подтверждение выхода", MessageBoxButton.YesNo, MessageBoxImage.Question); if (result == MessageBoxResult.Yes) { Manager.FrameMain.Navigate(new PageAuth()); } } }

**PageDevice.xaml**

<Grid> <Grid.RowDefinitions> <RowDefinition Height="100"/> <RowDefinition Height="\*"/> </Grid.RowDefinitions> <Grid Background="#53699e" x:Name="GrdHeader"> <WrapPanel HorizontalAlignment="Left" VerticalAlignment="Center"> <Label Content="Оборудование" Style="{StaticResource LblDevices}"/> </WrapPanel> <WrapPanel HorizontalAlignment="Right" VerticalAlignment="Center"> <Button x:Name="BtnAdd" Click="BtnAdd\_Click" Style="{StaticResource BtnPages}"> <Image Source="/Resources/ImgAdd.png" Style="{StaticResource ImgMenu}"/> </Button> <Button x:Name="BtnEdit" Click="BtnEdit\_Click" Style="{StaticResource BtnPages}"> <Image Source="/Resources/ImgEdit.png" Style="{StaticResource ImgMenu}"/> </Button> <Button x:Name="BtnDelete" Click="BtnDelete\_Click" Style="{StaticResource BtnPages}"> <Image Source="/Resources/ImgDelete.png" Style="{StaticResource ImgMenu}"/> </Button> <Grid> <TextBlock Text="Поиск..." Style="{StaticResource TxBlSearch}"/> <Image Source="/Resources/ImgSearch.png" Style="{StaticResource ImgSearch}"/> <TextBox x:Name="TxbSearch" Style="{StaticResource TxBoxSearch}"/> </Grid> </WrapPanel> </Grid> <DataGrid x:Name="DataGridDevice" BorderBrush="Transparent" AutoGenerateColumns="False" Height="600" Grid.Row="1"> <DataGrid.Columns> <DataGridTextColumn Header="Наименование" HeaderStyle="{StaticResource AutoSizeHeaderStyle}" Binding="{Binding DeviceName}" Width="\*" /> <DataGridTextColumn Header="Тип оборудования" HeaderStyle="{StaticResource AutoSizeHeaderStyle}" Binding="{Binding DeviceType.DeviceTypeName}" Width="\*" /> <DataGridTextColumn Header="Инвентарный номер" HeaderStyle="{StaticResource AutoSizeHeaderStyle}" Binding="{Binding DeviceInventNum}" Width="\*" /> <DataGridTextColumn Header="Модель" HeaderStyle="{StaticResource AutoSizeHeaderStyle}" Binding="{Binding DeviceModel}" Width="\*" /> <DataGridTextColumn Header="Дата приобретения" HeaderStyle="{StaticResource AutoSizeHeaderStyle}" Binding="{Binding DeviceDatePurchase, StringFormat=d}" Width="\*" /> <DataGridTextColumn Header="Цена приобретения" HeaderStyle="{StaticResource AutoSizeHeaderStyle}" Binding="{Binding DevicePrice, StringFormat={}{0} руб.}" Width="\*" /> <DataGridTextColumn Header="Статус" HeaderStyle="{StaticResource AutoSizeHeaderStyle}" Binding="{Binding DeviceStatus.DeviceStatusName}" Width="\*" /> </DataGrid.Columns> </DataGrid> </Grid>

**PageDevice.xaml.cs**

public partial class PageDevice : Page { public PageDevice() { InitializeComponent(); DataGridDevice.ItemsSource = DatabaseEntities.GetContext().Device.ToList(); } private void BtnAdd\_Click(object sender, RoutedEventArgs e) { Manager.SecondFrame.Navigate(new Pages.PagesEditAdd.PageEditAddDevice(null)); } private void BtnEdit\_Click(object sender, RoutedEventArgs e) { var ObjectForEdit = DataGridDevice.SelectedItem; Manager.SecondFrame.Navigate(new Pages.PagesEditAdd.PageEditAddDevice((Device)ObjectForEdit)); } private void BtnDelete\_Click(object sender, RoutedEventArgs e) { var ObjectForDelete = DataGridDevice.SelectedItems.Cast<Device>().ToList(); if (MessageBox.Show($"Вы действительно хотите удалить {ObjectForDelete.Count()} элемент", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes) { try { DatabaseEntities.GetContext().Device.RemoveRange(ObjectForDelete); DatabaseEntities.GetContext().SaveChanges(); MessageBox.Show("Данные удалены", "Информация", MessageBoxButton.OK, MessageBoxImage.Information); DataGridDevice.ItemsSource = DatabaseEntities.GetContext().Device.ToList(); } catch (Exception ex) { MessageBox.Show(ex.Message.ToString()); } } } }

**PageEditAddDevice.xaml**

<Grid> <Grid.RowDefinitions> <RowDefinition Height="100"/> <RowDefinition Height="\*"/> </Grid.RowDefinitions> <Grid Background="#53699e" Grid.Row="0"> <WrapPanel HorizontalAlignment="Center" VerticalAlignment="Center" Margin="5"> <Label Content="Добавление / Редактирование" x:Name="LblAdd" Style="{StaticResource LblDevices}"/> </WrapPanel> </Grid> <StackPanel Grid.Row="1" HorizontalAlignment="Center" Width="Auto"> <Separator Grid.Row="1" VerticalAlignment="Top"/> <TextBlock Text="Оборудование" Style="{StaticResource TxblTitle}" Foreground="Black" HorizontalAlignment="Left"/> <WrapPanel> <StackPanel Margin="5"> <TextBlock Text="Наименование" Style="{StaticResource TxtBlEditAdd}"/> <TextBox x:Name="TxbDeviceName" Text="{Binding DeviceName}" Style="{StaticResource TxtBoxEditAdd}"/> <TextBlock Text="Тип" Style="{StaticResource TxtBlEditAdd}"/> <ComboBox x:Name="CmbDeviceType" DisplayMemberPath="DeviceTypeName" SelectedItem="{Binding DeviceType}" Style="{StaticResource ComBoxEditAdd}"/> <TextBlock Text="Модель " Style="{StaticResource TxtBlEditAdd}"/> <TextBox x:Name="TxbDeviceModel" Text="{Binding DeviceModel}" Style="{StaticResource TxtBoxEditAdd}"/> <TextBlock Text="Инвентарный номер" Style="{StaticResource TxtBlEditAdd}"/> <TextBox x:Name="TxbDeviceInventNum" Text="{Binding DeviceInventNum}" Style="{StaticResource TxtBoxEditAdd}"/> </StackPanel> <StackPanel Margin="5"> <TextBlock Text="Серийный номер" Style="{StaticResource TxtBlEditAdd}"/> <TextBox x:Name="TxbDeviceSerialNum" Text="{Binding DeviceSerialNum}" Style="{StaticResource TxtBoxEditAdd}"/> <TextBlock Text="Дата приобретения" Style="{StaticResource TxtBlEditAdd}"/> <DatePicker x:Name="DtpDeviceDatePurchase" Loaded="DtpDeviceDatePurchase\_Loaded" SelectedDate="{Binding DeviceDatePurchase}" Style="{StaticResource DtPickEditAdd}"/> <TextBlock Text="Цена приобретения" Style="{StaticResource TxtBlEditAdd}"/> <TextBox x:Name="TxbDevicePrice" Text="{Binding DevicePrice}" Style="{StaticResource TxtBoxEditAdd}"/> <TextBlock Text="Статус" Style="{StaticResource TxtBlEditAdd}"/> <ComboBox x:Name="CmbDeviceStatus" DisplayMemberPath="DeviceStatusName" SelectedItem="{Binding DeviceStatus}" Style="{StaticResource ComBoxEditAdd}"/> </StackPanel> </WrapPanel> </StackPanel> <WrapPanel HorizontalAlignment="Right" VerticalAlignment="Bottom" Margin="5" Grid.Row="1"> <Button x:Name="BtnSave" Click="BtnSave\_Click" Style="{StaticResource BtnSaveBack}" Margin="5"> <WrapPanel> <Image Source="/Resources/ImgSave.png" Style="{StaticResource ImgMenu}"/> <TextBlock Text="Сохранить" Style="{StaticResource TxtBlEditAdd}"/> </WrapPanel> </Button> <Button x:Name="BtnBack" Click="BtnBack\_Click" Style="{StaticResource BtnSaveBack}" Margin="5"> <WrapPanel> <Image Source="/Resources/ImgBack.png" Style="{StaticResource ImgMenu}"/> <TextBlock Text="Назад" Style="{StaticResource TxtBlEditAdd}"/> </WrapPanel> </Button> </WrapPanel> </Grid>

**PageEditAddDevice.xaml.cs**

public partial class PageEditAddDevice : Page { public Device currentDevice = new Device(); public PageEditAddDevice(Device \_selectedDevice) { InitializeComponent(); CmbDeviceStatus.ItemsSource = DatabaseEntities.GetContext().DeviceStatus.ToList(); CmbDeviceType.ItemsSource = DatabaseEntities.GetContext().DeviceType.ToList(); if (\_selectedDevice != null) currentDevice = \_selectedDevice; DataContext = currentDevice; } private void DtpDeviceDatePurchase\_Loaded(object sender, RoutedEventArgs e) { DatePicker datePicker = sender as DatePicker; if (datePicker != null && (datePicker.SelectedDate == null || datePicker.SelectedDate == DateTime.MinValue)) { datePicker.SelectedDate = DateTime.Today; } } private void BtnSave\_Click(object sender, RoutedEventArgs e) { StringBuilder errors = new StringBuilder(); if (string.IsNullOrEmpty(TxbDeviceName.Text)) errors.AppendLine("Укажите наименование оборудования!"); if (CmbDeviceType.SelectedItem == null) errors.AppendLine("Выберите тип оборудования!"); if (string.IsNullOrEmpty(TxbDeviceModel.Text)) errors.AppendLine("Укажите модель оборудования!"); if (string.IsNullOrEmpty(TxbDeviceInventNum.Text)) errors.AppendLine("Укажите инвентарный номер оборудования!"); if (string.IsNullOrEmpty(TxbDeviceSerialNum.Text)) errors.AppendLine("Укажите серийный номер оборудования!"); if (CmbDeviceStatus.SelectedItem == null) errors.AppendLine("Укажите статус оборудования!"); if (DtpDeviceDatePurchase.SelectedDate == null) errors.AppendLine("Укажите дату приобретения оборудования!"); if (string.IsNullOrEmpty(TxbDevicePrice.Text)) { errors.AppendLine("Укажите цену приобретения!"); } else { // Пытаемся преобразовать введенный текст в число if (!decimal.TryParse(TxbDevicePrice.Text, out decimal totalPrice)) { errors.AppendLine("Укажите корректную цену!"); } } if (errors.Length > 0) { MessageBox.Show(errors.ToString(), "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error); return; } if (currentDevice.DeviceID == 0) { try { DatabaseEntities.GetContext().Device.Add(currentDevice); } catch (Exception ex) { MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning); } } try { DatabaseEntities.GetContext().SaveChanges(); MessageBox.Show("Информация сохранена!", "Информация", MessageBoxButton.OK, MessageBoxImage.Information); Manager.SecondFrame.Navigate(new PageDevice()); } catch (Exception ex) { MessageBox.Show(ex.Message, "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning); } } private void BtnBack\_Click(object sender, RoutedEventArgs e) { NavigationService.Navigate(new PageDevice()); } }

**PageDevMoving.xaml**

<Grid> <Grid.RowDefinitions> <RowDefinition Height="100"/> <RowDefinition Height="\*"/> </Grid.RowDefinitions> <Grid Background="#53699e" x:Name="GrdHeader"> <WrapPanel HorizontalAlignment="Left" VerticalAlignment="Center"> <Label Content="Перемещение оборудования" Style="{StaticResource LblDevices}"/> </WrapPanel> <WrapPanel HorizontalAlignment="Right" VerticalAlignment="Center"> <Button x:Name="BtnAdd" Click="BtnAdd\_Click" Style="{StaticResource BtnPages}"> <Image Source="/Resources/ImgAdd.png" Style="{StaticResource ImgMenu}"/> </Button> <Button x:Name="BtnEdit" Click="BtnEdit\_Click" Style="{StaticResource BtnPages}"> <Image Source="/Resources/ImgEdit.png" Style="{StaticResource ImgMenu}"/> </Button> <Button x:Name="BtnDelete" Click="BtnDelete\_Click" Style="{StaticResource BtnPages}"> <Image Source="/Resources/ImgDelete.png" Style="{StaticResource ImgMenu}"/> </Button> <Grid> <TextBlock Text="Поиск..." Style="{StaticResource TxBlSearch}"/> <Image Source="/Resources/ImgSearch.png" Style="{StaticResource ImgSearch}"/> <TextBox x:Name="TxbSearch" Style="{StaticResource TxBoxSearch}"/> </Grid> </WrapPanel> </Grid> <DataGrid x:Name="DataGridDevMoving" BorderBrush="Transparent" AutoGenerateColumns="False" Grid.Row="1"> <DataGrid.Columns> <DataGridTextColumn Header="Исполнитель" HeaderStyle="{StaticResource AutoSizeHeaderStyle}" Binding="{Binding User.UserSurname}" Width="\*" /> <DataGridTextColumn Header="Рабочее место" HeaderStyle="{StaticResource AutoSizeHeaderStyle}" Binding="{Binding Workplace.WorkplaceName}" Width="\*" /> <DataGridTextColumn Header="Оборудование" HeaderStyle="{StaticResource AutoSizeHeaderStyle}" Binding="{Binding Device.DeviceName}" Width="\*" /> <DataGridTextColumn Header="Дата перемещения" HeaderStyle="{StaticResource AutoSizeHeaderStyle}" Binding="{Binding DeviceMovingDate, StringFormat=d}" Width="\*" /> </DataGrid.Columns> </DataGrid> </Grid>

**PageDevMoving.xaml.cs**

public partial class PageDevMoving : Page { public PageDevMoving() { InitializeComponent(); DataGridDevMoving.ItemsSource = DatabaseEntities.GetContext().DeviceMoving.ToList(); } private void BtnAdd\_Click(object sender, RoutedEventArgs e) { Manager.SecondFrame.Navigate(new Pages.PagesEditAdd.PageEditAddDevMoving(null)); } private void BtnEdit\_Click(object sender, RoutedEventArgs e) { var ObjectForEdit = DataGridDevMoving.SelectedItem; Manager.SecondFrame.Navigate(new Pages.PagesEditAdd.PageEditAddDevMoving((DeviceMoving)ObjectForEdit)); } private void BtnDelete\_Click(object sender, RoutedEventArgs e) { var ObjectForDelete = DataGridDevMoving.SelectedItems.Cast<DeviceMoving>().ToList(); if (MessageBox.Show($"Вы действительно хотите удалить {ObjectForDelete.Count()} элемент", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes) { try { DatabaseEntities.GetContext().DeviceMoving.RemoveRange(ObjectForDelete); DatabaseEntities.GetContext().SaveChanges(); MessageBox.Show("Данные удалены", "Информация", MessageBoxButton.OK, MessageBoxImage.Information); DataGridDevMoving.ItemsSource = DatabaseEntities.GetContext().DeviceMoving.ToList(); } catch (Exception ex) { MessageBox.Show(ex.Message.ToString()); } } } }

**PageDevMoving.xaml.cs**

<Grid> <Grid.RowDefinitions> <RowDefinition Height="100"/> <RowDefinition Height="\*"/> </Grid.RowDefinitions> <Grid Background="#53699e" Grid.Row="0"> <WrapPanel HorizontalAlignment="Center" VerticalAlignment="Center" Margin="5"> <Label Content="Добавление / Редактирование" x:Name="LblAdd" Style="{StaticResource LblDevices}"/> </WrapPanel> </Grid> <StackPanel Grid.Row="1" HorizontalAlignment="Center" Width="Auto"> <Separator Grid.Row="1" VerticalAlignment="Top"/> <TextBlock Text="Перемещение оборудования" Style="{StaticResource TxblTitle}" Foreground="Black" HorizontalAlignment="Left"/> <WrapPanel> <StackPanel Margin="5"> <TextBlock Text="Исполнитель" Style="{StaticResource TxtBlEditAdd}"/> <ComboBox x:Name="CmbUser" SelectedItem="{Binding User}" Style="{StaticResource ComBoxEditAdd}" Width="915"> <ComboBox.ItemTemplate> <DataTemplate> <TextBlock> <Run Text="{Binding UserSurname}"/> <Run Text="{Binding UserName}"/> </TextBlock> </DataTemplate> </ComboBox.ItemTemplate> </ComboBox> <TextBlock Text="Рабочее место" Style="{StaticResource TxtBlEditAdd}"/> <ComboBox x:Name="CmbWorkplaceName" DisplayMemberPath="WorkplaceName" SelectedItem="{Binding Workplace}" Style="{StaticResource ComBoxEditAdd}" Width="915"/> <TextBlock Text="Оборудование" Style="{StaticResource TxtBlEditAdd}"/> <ComboBox x:Name="CmbDeviceName" DisplayMemberPath="DeviceName" SelectedItem="{Binding Device}" Style="{StaticResource ComBoxEditAdd}" Width="915"/> <TextBlock Text="Дата перемещения" Style="{StaticResource TxtBlEditAdd}"/> <DatePicker x:Name="DtpDevMovingDate" Loaded="DtpDevDropingDate\_Loaded" SelectedDate="{Binding DeviceMovingDate}" Style="{StaticResource DtPickEditAdd}" Width="915"/> </StackPanel> </WrapPanel> </StackPanel> <WrapPanel HorizontalAlignment="Right" VerticalAlignment="Bottom" Margin="5" Grid.Row="1"> <Button x:Name="BtnSave" Click="BtnSave\_Click" Style="{StaticResource BtnSaveBack}" Margin="5"> <WrapPanel> <Image Source="/Resources/ImgSave.png" Style="{StaticResource ImgMenu}"/> <TextBlock Text="Сохранить" Style="{StaticResource TxtBlEditAdd}"/> </WrapPanel> </Button> <Button x:Name="BtnBack" Click="BtnBack\_Click" Style="{StaticResource BtnSaveBack}" Margin="5"> <WrapPanel> <Image Source="/Resources/ImgBack.png" Style="{StaticResource ImgMenu}"/> <TextBlock Text="Назад" Style="{StaticResource TxtBlEditAdd}"/> </WrapPanel> </Button> </WrapPanel> </Grid>