УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«МОГИЛЕВСКИЙ ГОСУДАРСТВЕННЫЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

Специальность 2-40 01 01

Учебная группа ПО-455

Учебная предмет Базы данных и системы управления базами данных

**КУРСОВОЙ ПРОЕКТ**

**РАЗРАБОТКА** **ИНФОРМАЦИОННОЙ МОДЕЛИ**

**«УЧЕТ РЕЙСОВ АВТОПАРКА»**

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КП ПО.00.00.000 ПЗ**

Разработал A.О.Савич

Руководитель проекта А.Д. Базанова

2024

**Содержание**

Введение 4

1 Техническое задание 5

1.1 Анализ предметной области 6

1.2 Постановка задачи 6

2. Проектирование программного модуля 7

2.1 Функциональная модель 7

2.2 Информационная модель 16

2.3 Верификация спроектированной логической модели 17

3. Реализация системы 18

3.1 SQL-определения регламентированных запросов 18

3.2 Описание клиентских приложений 24

4 Функциональное и общее тестирование приложения 26

5 Энерго- и ресурсосбережение 27

Заключение 28

Список используемых источников 29

Приложение А Текст программы 30

**Введение**

**За последние десятилетия, программное обеспечение претерпело огромные изменения, пройдя путь от программ, которые способны выполнять лишь простейшие логические и арифметические операции, до сложных систем управления организациями. На сегодняшний день управление организацией без компьютера просто немыслимо. Для принятия какого-либо решения в условиях неопределенности и риска нужно регулярно держать под контролем разные аспекты финансово-хозяйственной деятельности. На современном этапе развития информатизации и автоматизации характерно применение распределенной обработки информации. В качестве более эффективной и перспективной сферы применения концепции распределенной обработки информации выступает автоматизация планово-управленческих функций на основе персональных компьютеров, которые установлены непосредственно на рабочем месте того или иного специалиста. Системы, которые выполняют данные функции, наделены широким распространением, которые называются автоматизированные рабочие места (АРМ).**

Программа автоматизации работы автопарка автобусов представляет собой программу управления базой данных. Реализация базы данных возможна несколькими способами:

– использование существующих систем управления базами данных (СУБД);

– создание собственной структурированной базы данных.

К преимуществу СУБД можно отнести централизованное хранение всей информации о базе данных в самой базе данных (БД). В системе изначально созданы и реализованы алгоритмы обработки массивов данных без внешнего вмешательства. Все операции по изменению, сохранению информации выполняются самой СУБД, следовательно, нет необходимости программисту самому реализовывать методы сохранения и открытия файла данных. К недостаткам СУБД можно отнести избыточность хранимой информации в файле БД, что существенно увеличивает размер файла БД. А также необходимость наличия СУБД на компьютере пользователя, что существенно усложняет поставку программ конечному пользователю.

**Цель данного курсового проекта - создание программы для автоматизации работы автопарка автобусов, удовлетворяющий перечисленным ранее требованиям, а также реализованный с использованием соответствующих СУБД и программного обеспечения.**

**1. Постановка задачи.**

**1.1 Анализ предметной области.**

**Услуги по перевозкам пассажиров в пределах какого-то региона должен обеспечивать городской автовокзал. Для этого организация должна иметь у себя в штате водителей, кассиров, системных администраторов (если необходимо), бухгалтеров, юристов, механиков, кондукторов, офис менеджера и т.д. Также имеется парк автобусов. Основные клиенты автовокзала - лица, проживающие в данном регионе.**

**Автопарк - Территория с комплексом зданий, сооружений и устройств для стоянки, технического обслуживания и текущего ремонта подвижного состава грузового или пассажирского автомобильного транспорта**

**Любая транспортная компания, которая дорожит своей репутацией на рынке автоперевозок, содержит большой автопарк, и старается регулярно обновлять модельный ряд автомобилей, которые в нем представлены. От качества автомобилей напрямую зависит качество предоставляемых клиенту услуг. Автомобиль, который имеет технические проблемы, может отказать в самый неожиданный момент, что неизбежно приведет к финансовым убыткам. Автопарк любой компании обычно состоит из той техники, которая больше всего востребована на рынке. Можно сказать, что есть несколько значений управления автопарком для бизнеса. Остановимся на них подробнее:**

**– отслеживание местоположения транспортных средств компании;**

**– управление маршрутами;**

– управление топливом;

– управление водителем.

Учитывая все вышесказанное можно заключить, что компания, инвестирующая в систему управления транспортом, может обладать следующими преимуществами:

– снижение затрат. Благодаря многим функциям, упомянутым выше, таким как планирование, выполнение перевозок, логистика, отслеживание перевозок, отчеты и анализ — все это может привести к сокращению затрат;

Улучшенное обслуживание клиентов и подотчетность. Выбор подходящего перевозчика, а также улучшенное управление логистикой и отчетами могут повысить качество обслуживания клиентов и помочь компании быть более ответственной.

Улучшенный финансовый менеджмент. Улучшенное планирование маршрутов, оптимизация загрузки, лучшие выбранные перевозчики могут обеспечить лучшее управление финансами, включая сокращение расходов, упомянутых выше.

Гибкость. Системы управления транспортом обеспечивают гибкость при изменении планов доставки.

Простая установка и использование веб-системы управления транспортировкой. Большинство транспортных систем имеют простую установку и должны быть удобными для пользователя.

В результате использования системы управления автопарком компания получает обработанную информацию, которая может использоваться для организационных решений по управлению автопарком и управлению ресурсами компании.

Эффективная работа каждого предприятия зависит от оперативности и рациональности выполняемых действий. В настоящее время для хозяйственного субъекта очень важно быть конкурентно способным, особенно в сфере предоставления услуг. От качества предоставляемых услуг напрямую зависит имидж предприятия среди клиентов, а также авторитет среди конкурирующих организаций на рынке. Одной из самых значимых сфер оказания услуг является сфера предоставления транспортных услуг. Ежедневно в мире перевозятся миллионы тонн грузов и сотни тысяч пассажиров.

Разрабатываемая программа позволит упростить работу сотрудников автопарка автобусов а также учет информации о продажах билетов, техническом обслуживании автобусов, сотрудниках автопарка автобусов, маршрутах автобусов.

**1.2 Постановка задачи.**

В результате выполнения курсового проекта требуется разработать программу, позволяющую оперативно учитывать следующую информацию: маршруты автобусов; обеспечение учета информации о продажах билетов; состояние технического обслуживания автобусов; список сотрудников автопарка автобусов.

В программе также присутствует функционал создания и экспорта отчетов. В своей основе программа имеет базу данных Microsoft Sql Server.

Список функций:

* просмотр отчетов;
* экспорт простых и сложных отчетов;
* просмотр и занесение информации о продажах билетов;
* просмотр и занесение информации о техническом обслуживании автобусов;
* просмотр и занесение информации о сотрудниках автопарка автобусов;
* просмотр и занесение информации о маршрутах автобусов.

Значительным преимуществом автоматизации данных операций является скорость их выполнения, возможность в любой момент осуществить запрос к данным, находящимся в базе и получить по этому запросу исчерпывающую информацию. Программа позволяет отказаться от необходимости ведения бумажных архивов.

**2. Проектирование программного модуля.**

**2.1 Функциональная модель**

Контекстная диаграмма представляет собой схему общей работы программного продукта.

Организация работы учета товаров магазина техники состоит из входных и выходных данных, под правилами приёма покупателей и прайс-листом. Механизмом исполнения является менеджер и продавец-консультант.

Входные данные включают в себя: Стоимость билета, Дата следующего обслуживания автобуса, Модель автобуса, Ключ модели автобуса, Номер продажи, Номер маршрута, Код сотрудника, Фамилия сотрудника, Имя сотрудника, Отчество сотрудника, Годен ли к дороге автобус, Инженер обслуживания, Сотрудник Работает с (дата), Номер обслуживания, Стажировка, Код должности, Наименование Должности, Номер билета, Время проезда, Начальный/Конечный пункт маршрута, Дата последнего обслуживания

Выходные данные включают в себя: информацию о маршрутах, информацию о сотрудниках, информацию о автобусах,информацию о техническом обслуживании, информацию о билетах и продажах.

Диаграммы декомпозиции предназначены для детализации функций и получаются при разбиении контекстной диаграммы на крупные подсистемы (функциональная декомпозиция) и описывающие каждый подсистему и их взаимодействие.

Контекстная диаграмма IDEF0 «Информационная система «Учет Рейсов Автопарка» детализируется на 6 процессов:

– учет сотрудников;

– учет состояния обслуживанияя автобусов;

– учет маршрутов;

– учет продаж ;

– учет билетов;

– учет автобусов в автопарке.

Диаграммы Bpwin приведены ниже на рисунке 1 – 4.

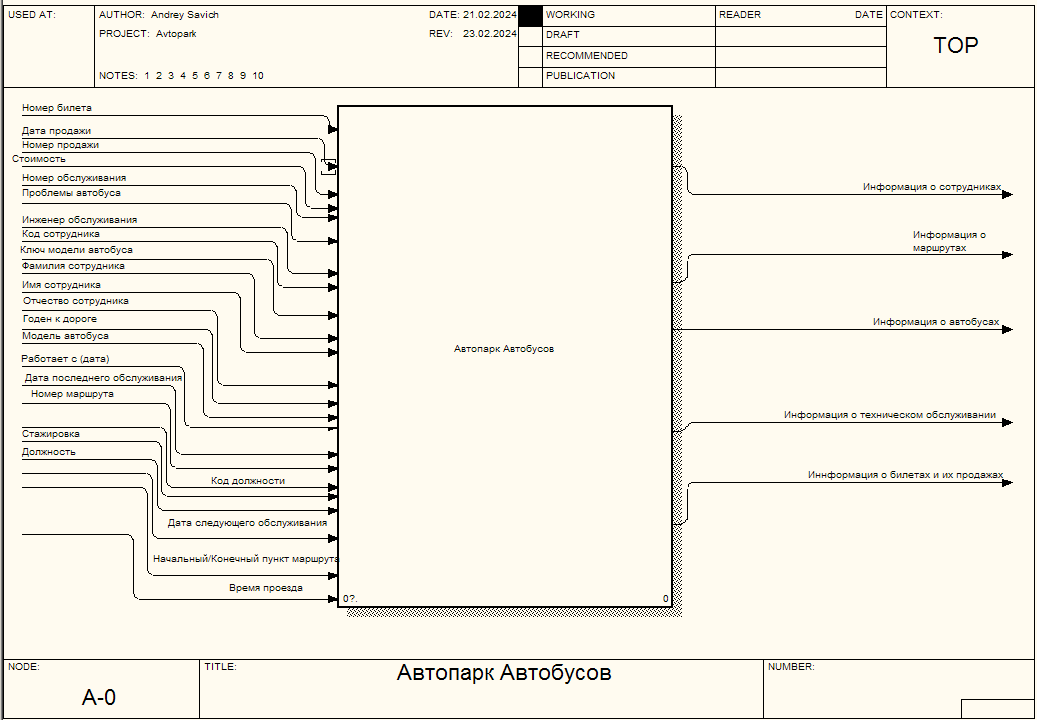


Рисунок 1 – Диаграмма IDEF0

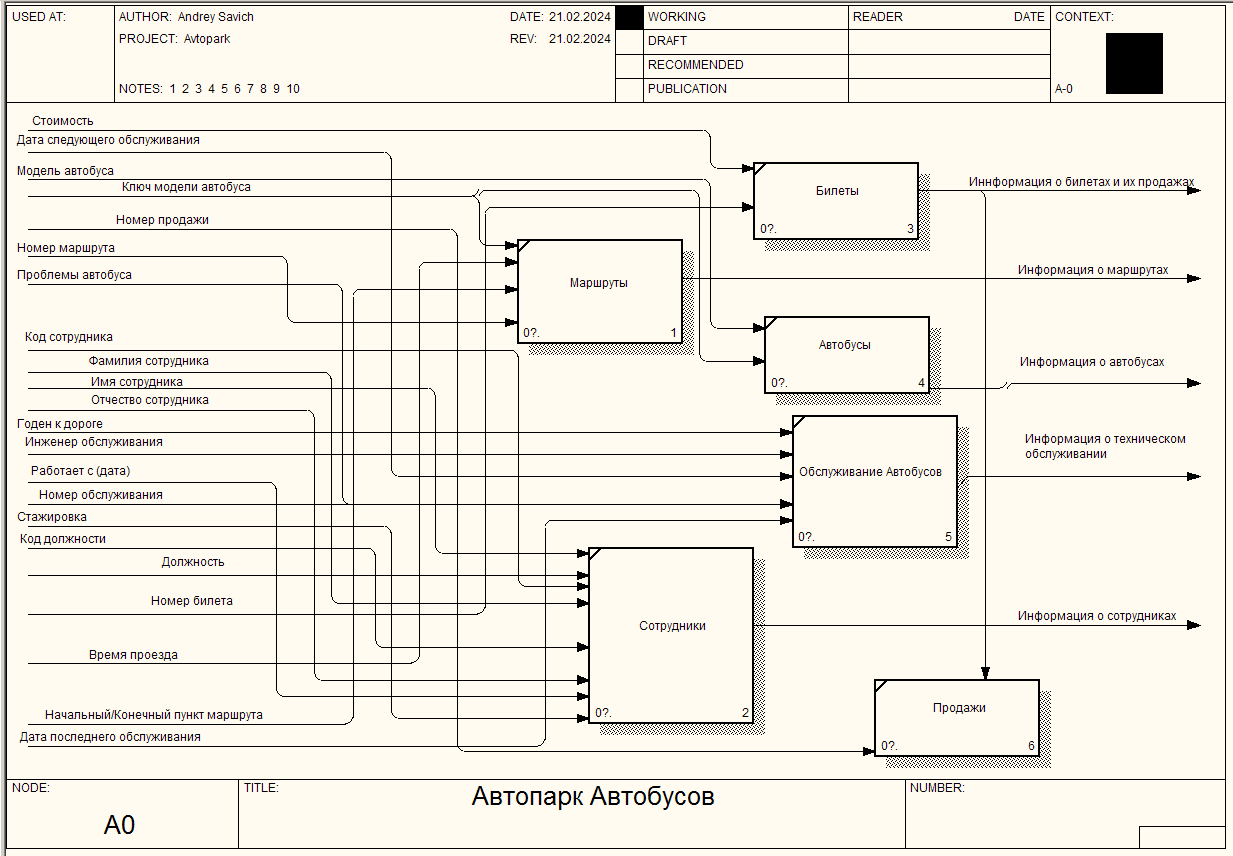
****

Рисунок 2 – Первый уровень диаграммы IDEF0

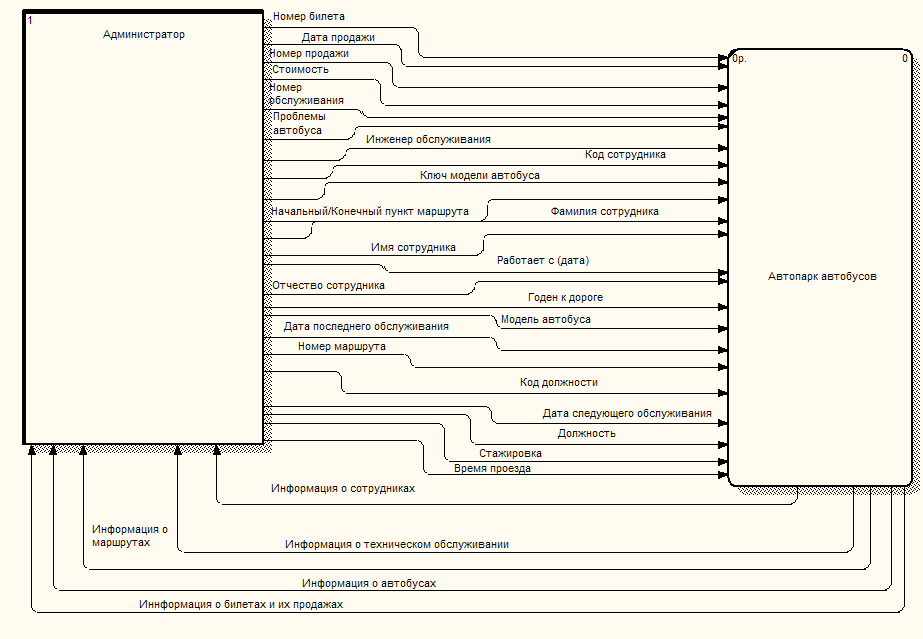
****

Рисунок 3 – Диаграмма DFD

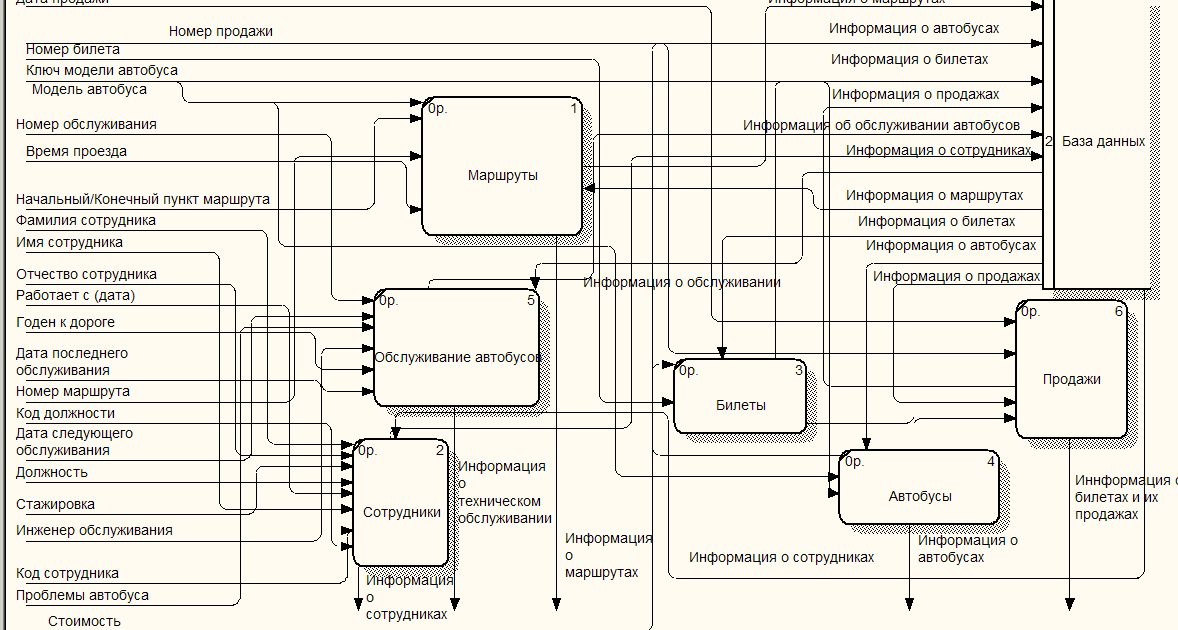
****

Рисунок 4 – Диаграмма DFD первого уровня

**2.2 Информационная модель**

Интегрированная среда разработки (IDE) - это многофункциональная программа, которая поддерживает многие аспекты разработки программного обеспечения. Интегрированная среда разработки Visual Studio - это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Помимо стандартного редактора и отладчика, которые есть в большинстве сред IDE, Visual Studio включает в себя компиляторы, средства автозавершения кода, графические конструкторы и многие другие функции для улучшения процесса **разработки. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight.Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).**

**Система Microsoft Visual Studio является самой сложной интегрированной средой разработки (integrated development enviroment - IDE), доступной для программистов в настоящее время. Она является результатом долгой истории развития языков программирования и интерфейсов и вобрала в себя достижения многих сред разработки программного обеспечения.**

**Microsoft SQL Server - система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов - Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.**

**Программа для автоматизации работы автопарка автобусов функционирует на компьютерах с определенным аппаратным и программным** обеспечением.

Минимальные требования для корректной работы системы:

* центральный процессор - 2 ГГц ;
* оперативная память - 2 ГБ;
* свободное место на HDD - не менее 5 ГБ;
* мышь;
* клавиатура;
* монитор 14 и выше дюймов;
* операционная система Windows.

Диаграмма Erwin приведена ниже на рисунке 5.

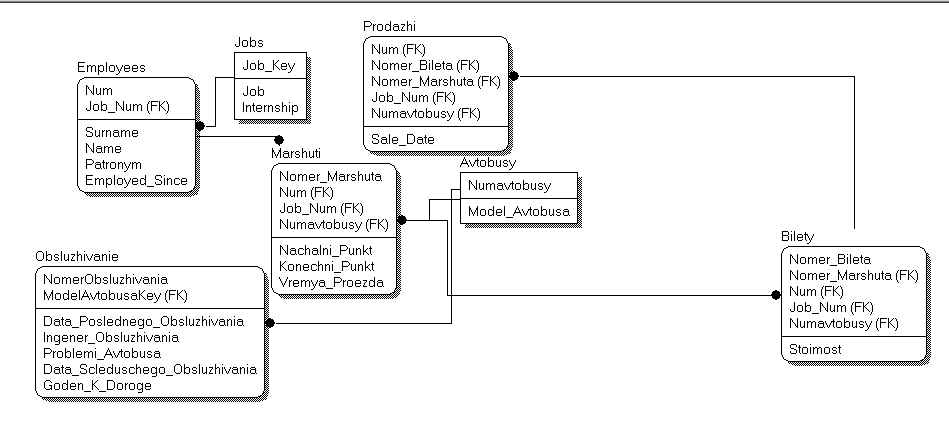


Рисунок 5 – Диаграмма ERWIN

Таблица 1 – Описание процесса выполнения варианта использования.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Пользователь | Система | Экранная форма | Условие:  последующий шаг |
| 1 | 2 | 3 | 4 | 5 |
| 1. | Открывает программу | Программа загружает главное окно и диалог настройки соединения с базой данных | Главное окно | Выбор одного из пунктов меню |
| 2. | Выбирает страницу соответствующую тем данным которые пользователь желает отредактировать | Программа загружает страницу, например, страницу с информацией о сотрудниках | Главное Окно | Выбор одного из пунктов меню |
| 3. | Выбор пункта меню подключится к базе данных | Обновление DataGrid | Информация о маршрутах. | Выбор одного из пунктов меню или ввод информации в Textbox для добавления или обновления данных |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 4. | Ввод информации и нажатие кнопки отправить данные в БД | Добавление введенной информации в таблицу Marshuti в базе данных после проверки на ошибки | Информация о маршрутах. | Выбор одного из пунктов меню |
| 5. | Ввод информации и нажатие кнопки обновить данные о маршрутах | Обновление данных в столбце с введенным ключом | Информация о маршрутах. | Выбор одного из пунктов меню |
| 6. | Выбор пункта меню продажа билетов | Открытие формы продажи билетов | Информация о маршрутах. | Выполнение ввода информации в форме продаже билетов и выбор одного из действий в форме или выбор одного из пунктов меню |
| 7. | Нажатие кнопки обновить DataGrid | Обновление DataGrid | Продажа билетов | Ввод информации в Textbox для добавления или обновления данных или выбор одного из пунктов меню в форме информации о маршрутах |
| 8. | Ввод информации и нажатие кнопки отправить данные в БД | Добавление введенной информации в таблицу Prodazhi в базе данных после проверки на ошибки | Продажа билетов | Выбор одного из пунктов меню в форме информации о маршрутах |
| 9. | Ввод информации и нажатие кнопки обновить данные о продажах | Обновление данных в столбце с введенным ключом в таблице Prodazhi в базе данных | Продажа билетов | Выбор одного из пунктов меню в форме информации о маршрутах |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 10. | Выбор пункта меню удаление данных | Открытие формы удаления данных | Информация о маршрутах. | Выполнение ввода информации в форме удаления данных или выбор одного из пунктов меню |
| 11. | Ввод наименования таблицы и ключа столбца и нажатие кнопки удалить из таблицы | Удаление данных из введенной таблицы | Удаление данных | Выбор одного из пунктов меню в форме информации о маршрутах |
| 12. | Выбор пункта меню техническое обслуживание автобусов | Открытие формы технического обслуживания автобусов | Информация о маршрутах. | Выполнение ввода информации в форме технического обслуживания автобусов и выбор одного из действий в форме или выбор одного из пунктов меню |
| 13. | Нажатие кнопки обновить DataGrid | Обновление DataGrid | Техническое обслуживание автобусов | Ввод информации в Textbox для добавления или обновления данных или выбор одного из пунктов меню в форме информации о маршрутах |
| 14. | Ввод информации и нажатие кнопки отправить данные в БД | Добавление введенной информации в таблицу Maintenance в базе данных после проверки на ошибки | Техническое обслуживание автобусов | Выбор одного из пунктов меню в форме информации о маршрутах |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 15. | Ввод информации и нажатие кнопки обновить данные о обслуживании автобусов | Обновление данных в столбце с введенным ключом в таблице Maintenance в базе данных | Техническое обслуживание автобусов | Выбор одного из пунктов меню в форме информации о маршрутах |
| 16. | Выбор пункта меню информация о сотрудниках | Открытие формы информации о сотрудниках | Информация о маршрутах. | Выполнение ввода информации в форме информации о сотрудниках и выбор одного из действий в форме или выбор одного из пунктов меню |
| 17. | Нажатие кнопки обновить DataGrid | Обновление DataGrid | Информация о сотрудниках | Ввод информации в Textbox для добавления или обновления данных или выбор одного из пунктов меню в форме информации о маршрутах |
| 18. | Ввод информации и нажатие кнопки отправить данные в БД | Добавление введенной информации в таблицу Employees в базе данных после проверки на ошибки | Информация о сотрудниках | Выбор одного из пунктов меню в форме информации о маршрутах |
| 19. | Ввод информации и нажатие кнопки обновить данные о обслуживании автобусов | Обновление данных в столбце с введенным ключом в таблице Employees в базе данных | Информация о сотрудниках | Выбор одного из пунктов меню в форме информации о маршрутах |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 20. | Выбор пункта меню просмотр отчетов | Запрос пути к отдельно поставляемой программе просмотра отчетов и ее открытие | Информация о маршрутах. | Открытие и просмотр файла отчетов в программе просмотра отчетов либо выбор одного из пунктов меню в форме информации о маршрутах |
| 21. | Нажатие кнопки открытия файла | Открытие диалога выбора файла и загрузка файла отчета | Просмотр отчетов | Выбор одного из пунктов меню в форме информации о маршрутах |
| 22. | Нажатие кнопки First | Переход на первую страницу отчета | Просмотр отчетов | Выбор одного из пунктов меню в форме информации о маршрутах |
| 23. | Нажатие кнопки назад | Переход на предыдущую страницу отчета | Просмотр отчетов | Выбор одного из пунктов меню в форме информации о маршрутах |
| 24. | Нажатие кнопки вперед | Переход на следующую страницу отчета | Просмотр отчетов | Выбор одного из пунктов меню в форме информации о маршрутах |
| 25. | Нажатие кнопки Last | Переход на последнюю страницу отчета | Просмотр отчетов | Выбор одного из пунктов меню в форме информации о маршрутах |
| 26. | Нажатие кнопки + | Увеличение масштаба | Просмотр отчетов | Выбор одного из пунктов меню в форме информации о маршрутах |

Продолжение таблицы 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 |
| 7. | Нажатие кнопки - | Уменьшение масштаба | Просмотр отчетов | Выбор одного из пунктов меню в форме информации о маршрутах |
| 28. | Выбор пункта меню экспорт отчетов | Открытие формы экспорта отчетов | Информация о маршрутах. | Выбор одного из действий в форме экспорта отчетов или выбор одного из пунктов меню в форме информации о маршрутах |
| 29. | Нажатие кнопки экспортировать простой отчет | Подготовка и экспорт простого отчета в форматах .pdf и .frx | Экспорт отчетов | Выбор одного из действий в форме экспорта отчетов или выбор одного из пунктов меню в форме информации о маршрутах |
| 30. | Нажатие кнопки экспортировать сложный отчет | Подготовка и экспорт сложного отчета в форматах .pdf и .frx | Экспорт отчетов | Выбор одного из пунктов меню в форме информации о маршрутах |
| 31. | Выбор пункта меню о программе | Открытие формы о программе | Информация о маршрутах. | Закрытие программы |

**2.3 Верификация спроектированной логической модели.**

После разработки информационной модели ее следует связать с функциональной моделью. Такая связь гарантирует завершенность анализа, гарантирует, что есть источники данных (сущности) для всех работ. Связывание моделей способствует согласованности, корректности и завершенности анализа. Стрелки в функциональной модели обозначают некоторую информацию, использующуюся в моделируемой системе. В информационной модели на логическом уровне информация изображается в виде сущностей. Сущности состоят из совокупностей экземпляров сущностей (кортежи отношений). К информационной модели предъявляется требование нормализации, что должно обеспечить компактность и непротиворечивость хранения данных. Информация, которая моделируется одной стрелкой в функциональной модели, может содержаться в нескольких сущностях и атрибутах информационной модели. На функциональной модели могут присутствовать различные стрелки, изображающие одни и те же данные. Информация о таких стрелках находится в одних и тех же сущностях. Следовательно, одной и той же стрелке в функциональной модели могут соответствовать несколько сущностей в информационной модели и, наоборот, одной сущности может соответствовать несколько стрелок.

Работы в функциональной модели могут создавать или изменять данные, которые соответствуют входящим и выходящим стрелкам. Они могут воздействовать как целиком на сущности (создавая и модифицируя экземпляры сущности), так и на отдельные атрибуты сущности.

Первым шагом связывания модели данных и модели процессов является экспорт данных из Erwin в Bpwin. Для успешного связывания моделей необходимо, чтобы версии Erwin в Bpwin соответствовали друг другу.

BPwin позволяет связывать элементы информационной модели, созданной с помощью Erwin, документировать влияние работ на данные и, тем самым позволяет создать спецификации на права доступа к данным для каждого процесса.

Результаты верификации спроектированной логической модели представлены на рисунке 6.

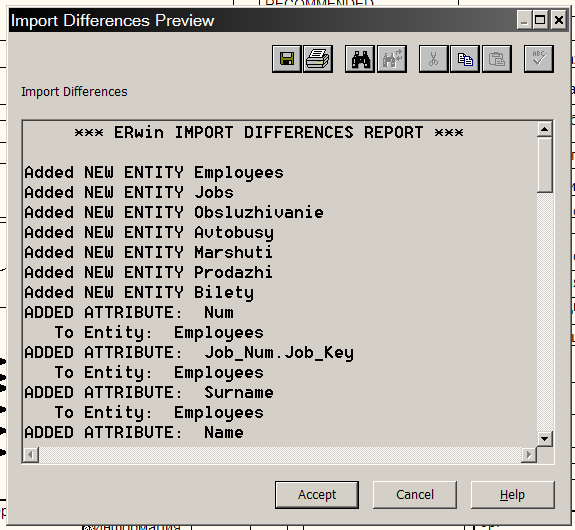
****

Рисунок 6– Протокол импорта

Для связывания атрибута со стрелкой достаточно щелкнуть по иконке выбора в иерархическом списке атрибутов. При этом сущность автоматически связывается со стрелкой. Каждая стрелка в модели процессов может быть связана с несколькими атрибутами различных сущностей.

На рисунке 7 показано диалоговое окно BPwin Data Usage Editor.

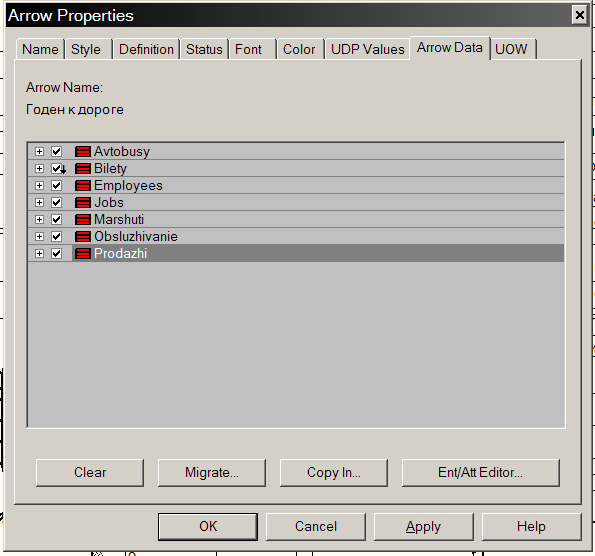


Рисунок 7 – Вкладка «Arrow Data» диалога Arrow Property

Кнопка CopyIn позволяет копировать связанные данные из другой стрелки.

Кнопка Clear - все связи стрелки с данными.

Кнопка Migrate вызывает диалог ChangestoArrowDataAssociations, в котором отображаются данные, мигрирующие от дочерних к родительским стрелкам (для разветвляющихся и сливающихся стрелок). При миграции возможны изменения связывания данных:

– Deletions - если данные связаны с родительской стрелкой, но не связаны с дочерней, связи с родительской стрелкой удаляются;

– Additions - если данные связаны с дочерней стрелкой и не связаны с родительской, добавляется связь с родительской стрелкой.

Для подтверждения изменений в диалоге ChangestoArrowDataAssociations следует щелкнуть по кнопке Commit. Миграция возможна только в моделях IDEFO и DFD. Как было указано выше, работы могут воздействовать на данные. Для документирования такого воздействия необходимо щелкнуть правой кнопкой мыши по работе и выбрать пункт меню Data UsageEditor.

В появившемся диалоге DataUsageEditor в виде иерархического списка показываются все работы модели, стрелки, которые касаются работ, сущности и атрибуты, которые были связаны со стрелками. В верхнем списке нужно щелкнуть по имени стрелки, с которой были связаны сущности и атрибуты. Для задания ассоциации достаточно щелкнуть по окошечку в иерархическом списке.

Для сущностей задается ассоциация CRUD (Create, Read, Update, Delete), для атрибутов - IRUN (Insert, Read, Update, Nullify), Ассоциации CRUD и IRUN — это правила использования сущностей и атрибутов работами, т. е. то, что могут делать работы с входящими или исходящими данными. Данные не могут использоваться работами произвольно. Стрелки входа представляют данные, которые работа преобразует в выход или потребляет. Такие данные могут быть обновлены (Update) или удалены (Delete), но не могут быть созданы (Create). Данные, связанные со стрелками управления, могут быть только прочитаны (Read), но не могут быть изменены - процедуры и стратегии не могут изменяться в работе. Данные, связанные со стрелками выхода, могут быть обновлены (если им соответствуют данные стрелок входа), удалены (Delete) или созданы (Create). Для стрелок механизма ассоциации не устанавливаются.

DataUsageReport. Результат связывания объектов модели процессов можно отобразить в отчете DataUsageReport (меню Report/Data Usage Report).

Для его создания выбираем пункт менюTools\Reports\Data Usage Report.

Появится окно DataUsageReport, в котором указываем, какие поля будут в отчете (рисунок 8).

Нажимаем Report.

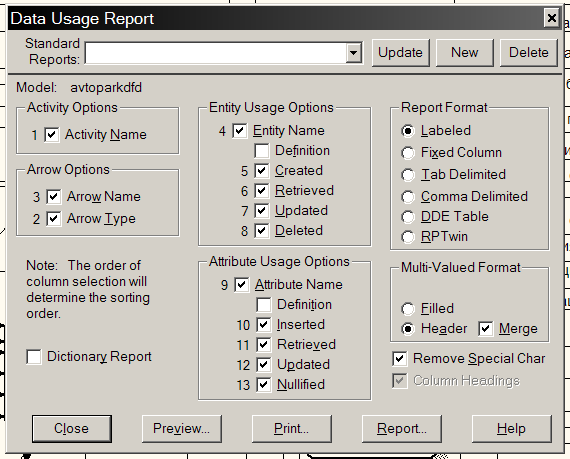


Рисунок 8 – Окно «Data Usage Repor»

На рисунке 9 изображено сохранение данных «Data Usage Repor» на основании верификации спроектированной логической модели.

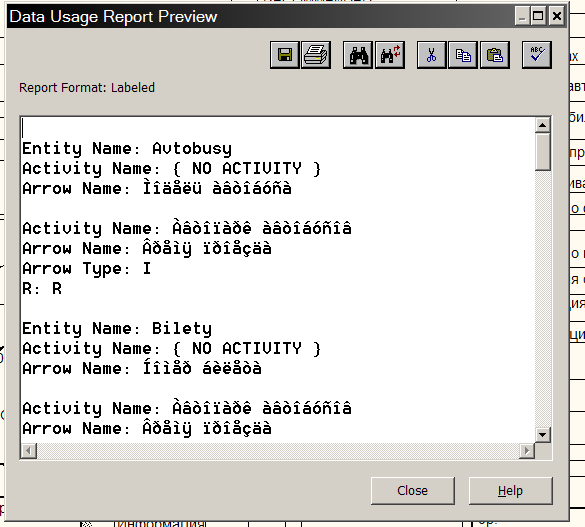
****

Рисунок 9 – Отчет верификации спроектированной логической модели

**3. Реализация программного модуля.**

**3.1 SQL-определения регламентированных запросов**

SQL (сокращение от англ. Structured Query Language) - это язык запросов, который применяют, чтобы [работать с базами данных](https://practicum.yandex.ru/blog/chto-takoe-bazy-dannyh/), структурированных особым образом. Главные задачи SQL - составлять запросы так, чтобы находить среди большого объёма информации ту, что нужна для конкретных целей, сортировать её, структурировать и представлять в наиболее простом и понятном виде.

Любая команда должна начинаться с ключевого слова — или действия, которое должно произойти. Например, выбрать строку, вставить новую, изменить старую или удалить таблицу целиком.

Одно из таких ключевых слов - SELECT. Оно выбирает отдельные столбцы или таблицу целиком, чтобы потом передать данные другим запросам на обработку.

FROM ставится после SELECT и нужен затем, чтобы указать, из какой таблицы или источника данных приходит информация. Здесь прописывается имя таблицы, с которой мы хотим работать.

WHERE используется для фильтрации данных. После него указывается условие, которому должны удовлетворять строки, чтобы они попали в результат выполнения запроса.

**INSERT**

Позволяет добавить новую строку в таблицу. Для этого нужно указать, какие столбцы мы хотим заполнить и передать значения для них с помощью команды VALUES.

UPDATE

Изменяет существующие записи. Чтобы использовать эту команду, укажите уникальный идентификатор изменяемой записи. Либо характеристику, по которой можно получить одну запись или группу из нескольких записей.

DELETE

Удаляет записи из таблицы по поисковому запросу

Код используемых запросов приведен ниже

UPDATE Employees SET Surname=@a1,Name=@a2, Patronym=@a3, Employed\_Since=@a4, Job\_Num=@a5 WHERE Num=@a7"

INSERT INTO Employees(Num, Surname, Name, Patronym,Employed\_Since,Job\_Num) "Values('"Num"', '"EmpName"', '"EmpsurName"', '"Emppat"', '"EmpES"', '" Empnum"')

SELECT \* FROM Employees

**3.2 Описание клиентских приложений**

При открытии программы вам предстает главное окно программы и диалог о соединении с базой данных и настройке данного соединения.

По потверждению соединения вам предстает форма с выбором страниц первой пунктом в которой является страница с информацией о маршрутах.

Форма «Информация о маршрутах» cлужит для просмотра информации о существующих маршрутах, а также для вызова остального функционала программы такого как: о программе, продажа билетов, подключение к базе данных, удаление данных, техническое обслуживание автобусов, информация о сотрудниках, просмотр отчетов, экспорт отчетов.

Внешний вид формы приведен на рисунке 10.

Форма «Информация о маршрутах» содержит следующие компоненты:

– Label -служит для показа названия полей базы данных;

– Textbox – служит для ввода данных добавляемых в базу данных или обновления существующих данных;

– Button – отправка новых данных в базу данных или обновление существующих;

– DataGrid – служит для показа информации находящийся в базе данных;

– TitleBarButton – служит для открытия меню по левому щелчку мыши.

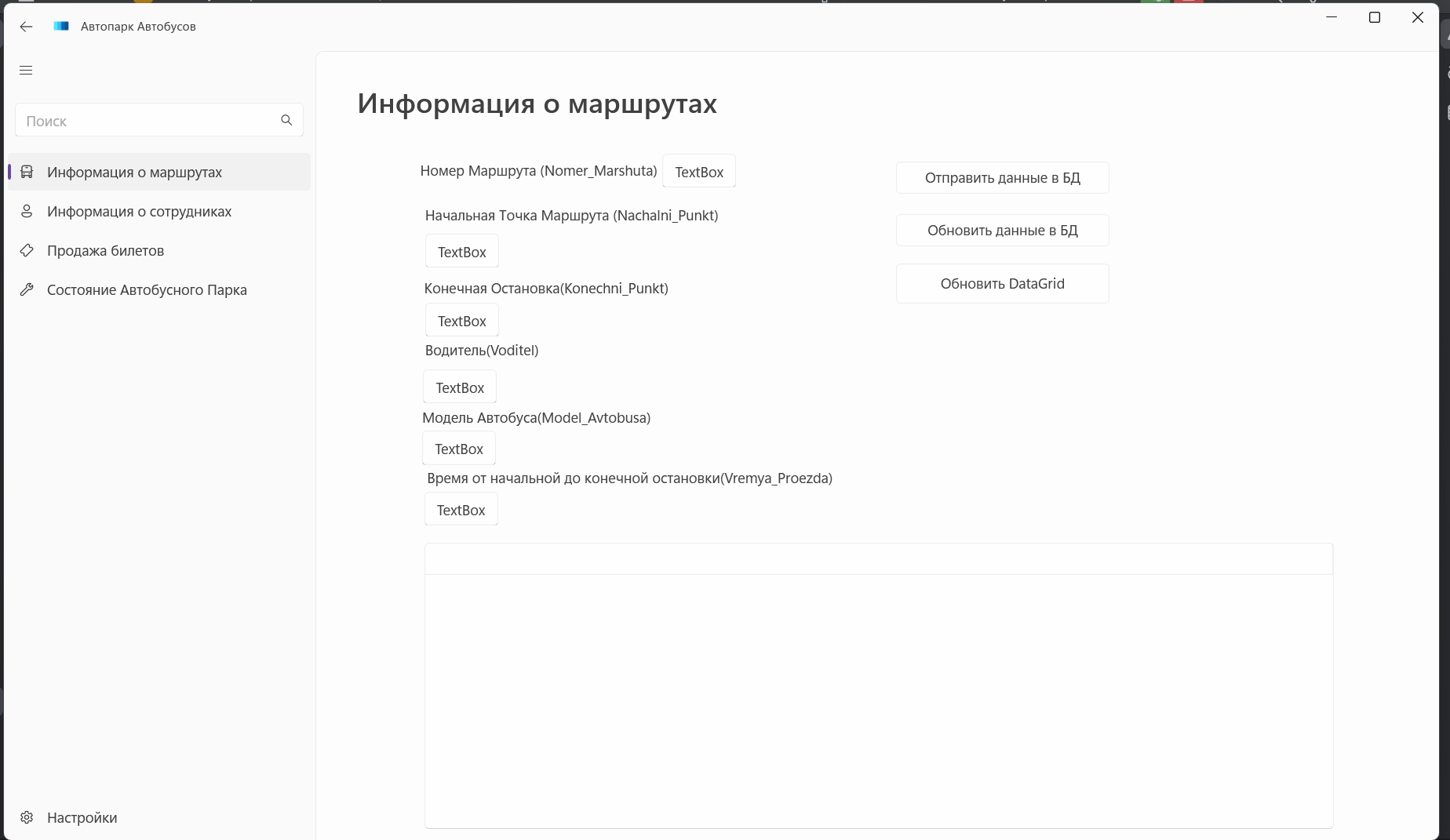


Рисунок 10 – Форма «Информация о маршрутах»

Основное назначение формы «Продажа билетов» – просмотр данных о уже проданных билетах и занесение информации о новых проданных билетах

Внешний вид формы приведен на рисунке 11

Форма «Продажа билетов» содержит следующие компоненты:

– Label -служит для показа названия полей базы данных;

– Textbox – служит для ввода данных добавляемых в базу данных или обновления существующих данных;

– Button – Обновление DataGrid или отправка новых данных в базу данных или обновление существующих;

– DataGrid – служит для показа информации находящийся в базе данных.

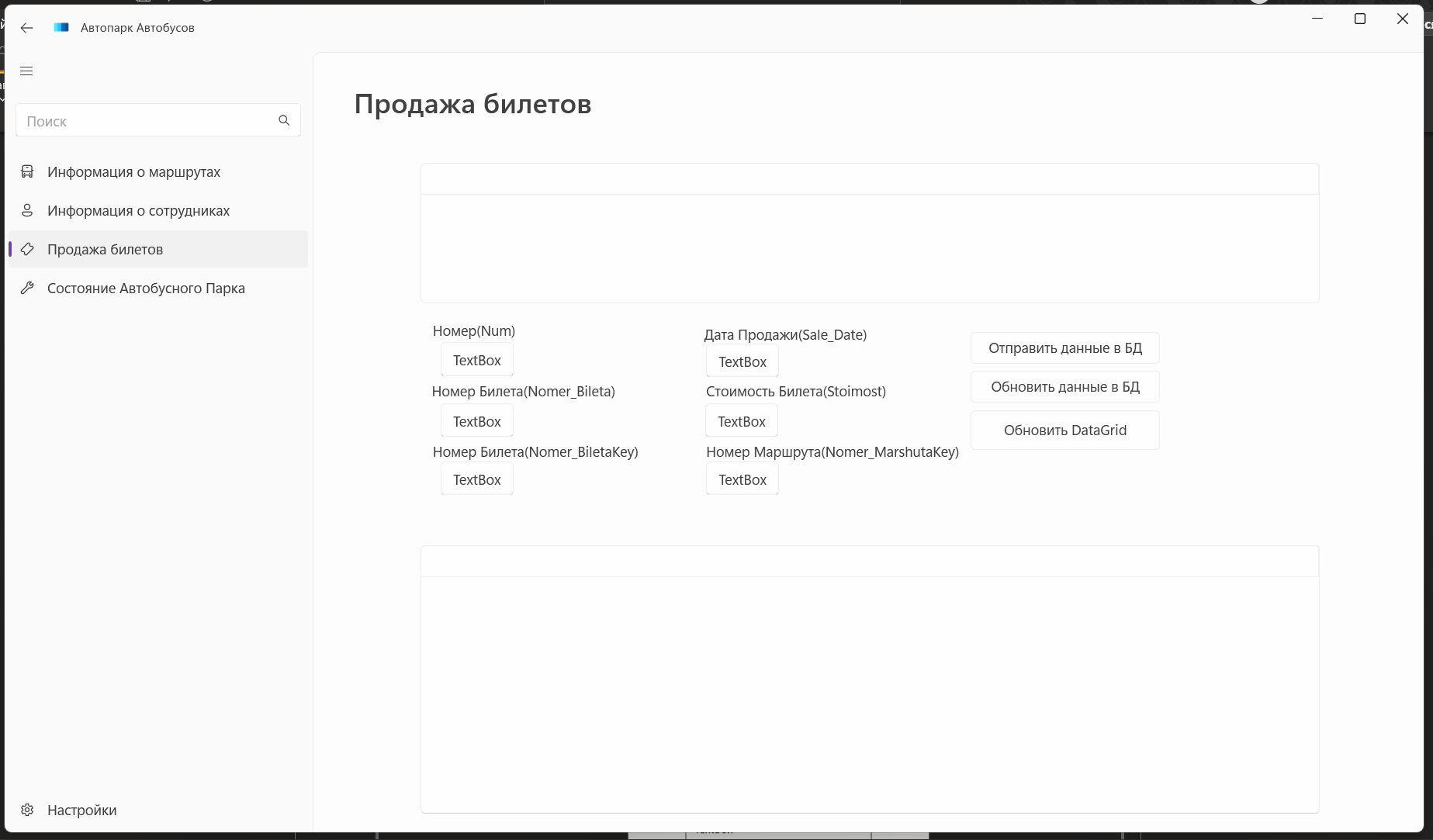


Рисунок 11 – Форма «Продажа билетов».

Основное назначение компонета «Удаление данных» - запрос на удаление определенной строки с введенным ключом из введенной таблицы

Управление компонентом «Удаление данных» находится в форме настроек вид которой приведен на рисунке 15 и содержит следующие компоненты:

– Label -служит для показа назначения TextBox;

– Textbox – служит для ввода таблицы и ключа для удаляемых данных;

– Button – Служит для отправки запроса на удаление данных.

Основное назначение формы «Техническое обслуживание автобусов» - просмотр данных о состоянии технического обслуживания автобусов и занесение или обновление данных о техническом обслуживании автобусов

Вид формы Техническое обслуживание автобусов приведен на рисунке 12.

Форма «Техническое обслуживание автобусов» содержит следующие компоненты:

– Label -служит для показа названия полей базы данных;

– Textbox – служит для ввода данных добавляемых в базу данных или обновления существующих данных;

– Button – Обновление DataGrid или отправка новых данных в базу данных или обновление существующих;

– DataGrid – служит для показа информации находящийся в базе данных.

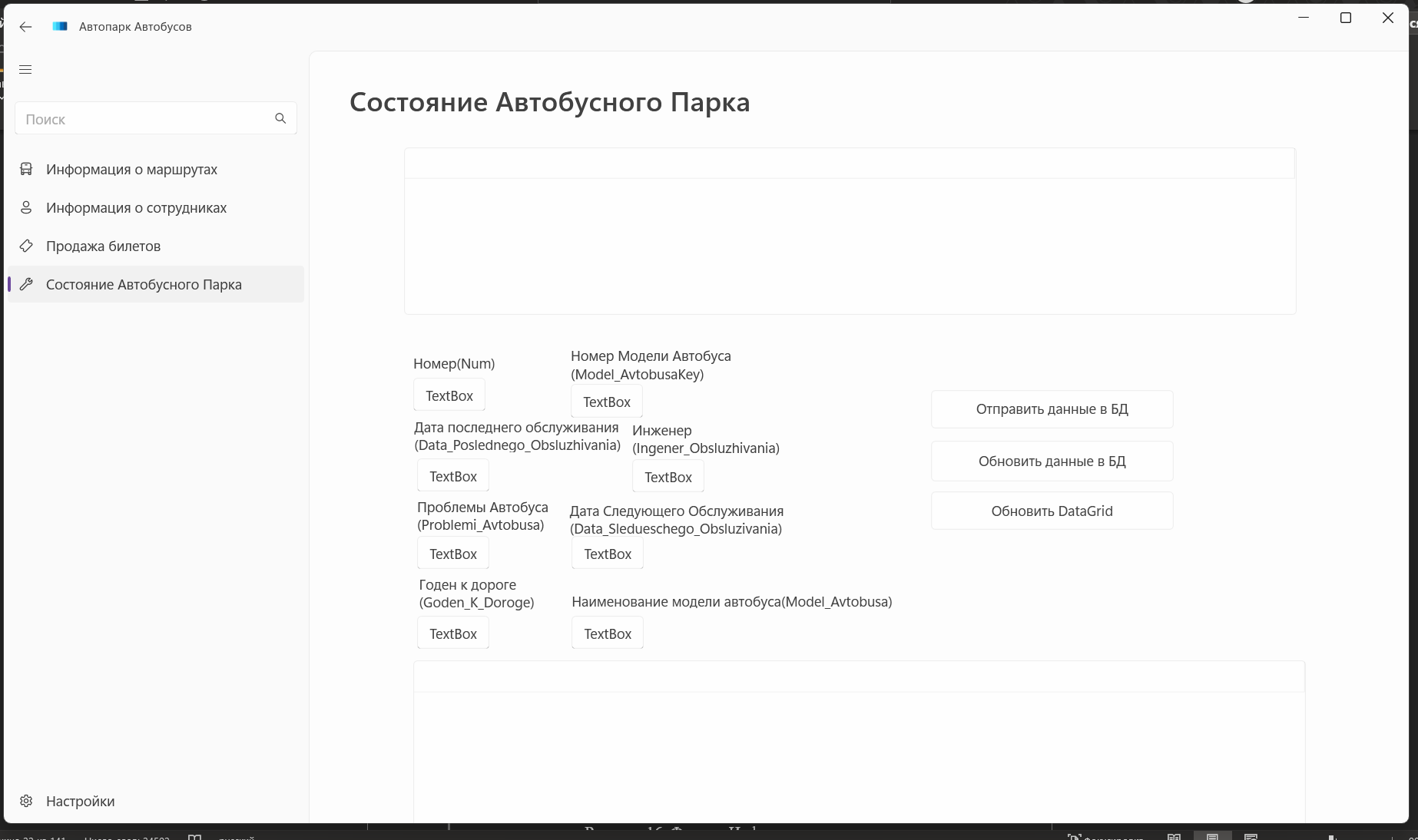


Рисунок 12 – Форма «Техническое обслуживание автобусов».

Основное назначение формы «Информация о сотрудниках» - просмотр информации о сотрудниках автопарка автобусов и добавление или редактирование данных о сотрудниках.Внешний вид формы приведен на рисунке 13

Форма «Информация о сотрудниках» содержит следующие компоненты:

– Label -служит для показа названия полей базы данных;

– Textbox – служит для ввода данных добавляемых в базу данных или обновления существующих данных;

– Button – Обновление DataGrid или отправка новых данных в базу данных или обновление существующих;

– DataGrid – служит для показа информации находящийся в базе данных.

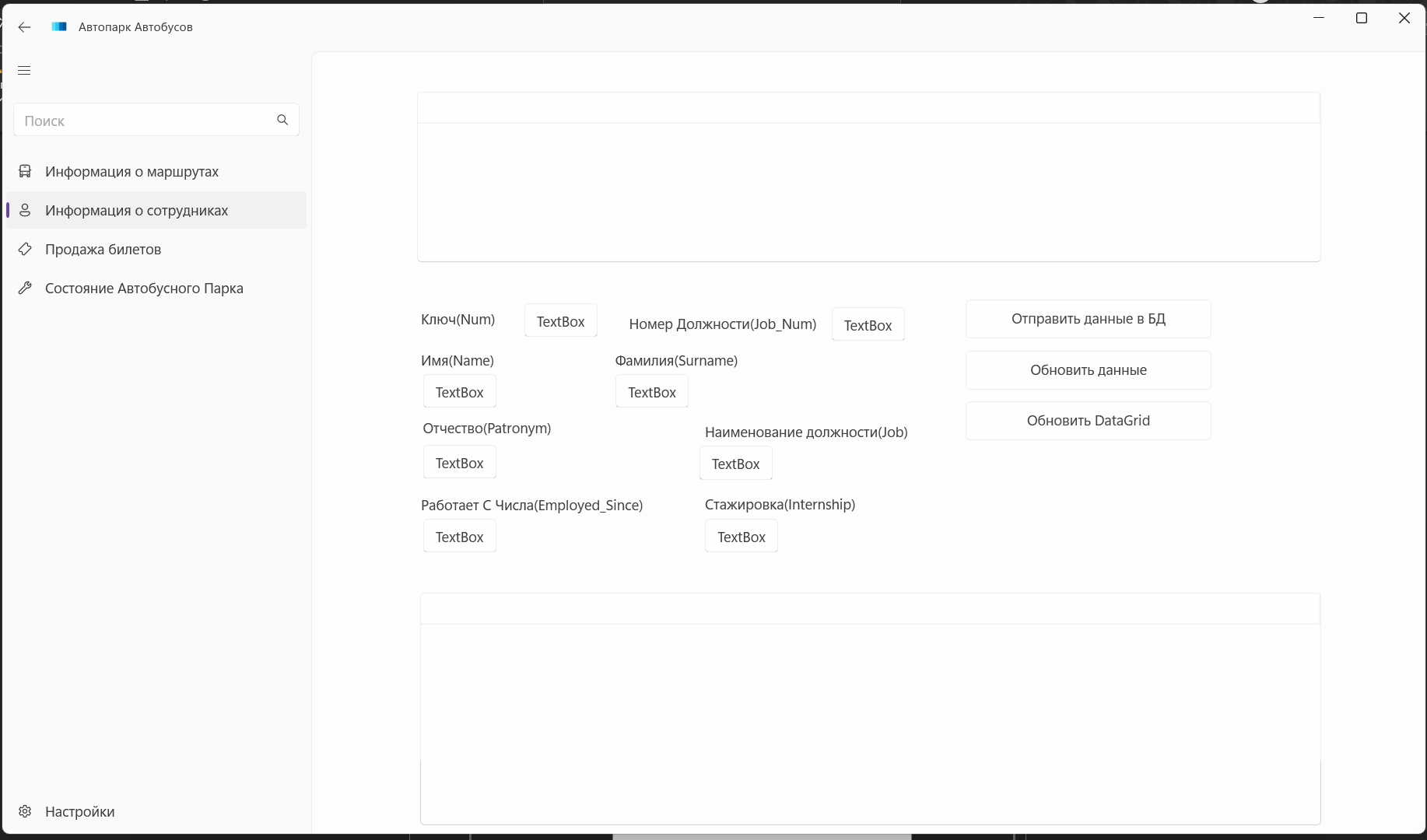


Рисунок 13 – Форма «Информация о сотрудниках».

Основное назначение формы «Просмотр отчетов» - просмотр файлов отчетов в формате .frx. Внешний вид формы приведен на рисунке 14.

Форма «Просмотр отчетов» содержит следующие компоненты:

– Button – служит для функционала открытия файлов, перехода на первую, следующую, предыдущую, последнюю страницы, а также для уменьшения и увеличения масштаба;

– ScrollViewer – служит для показа отчета.



Рисунок 14 – Форма «Просмотр отчетов»

Назначение компонета «Экспорт отчетов» - экспорта файлов сложного и простого отчета в формате .frx и формате .pdf.

Управление компонентом «Экспорт отчетов» находится в форме настроек внешний вид которой приведен на рисунке 15 и содержит следующие компоненты:

Button – служит для запуска получения данных из базы данных и формирования файлов отчета (сложного или простого).

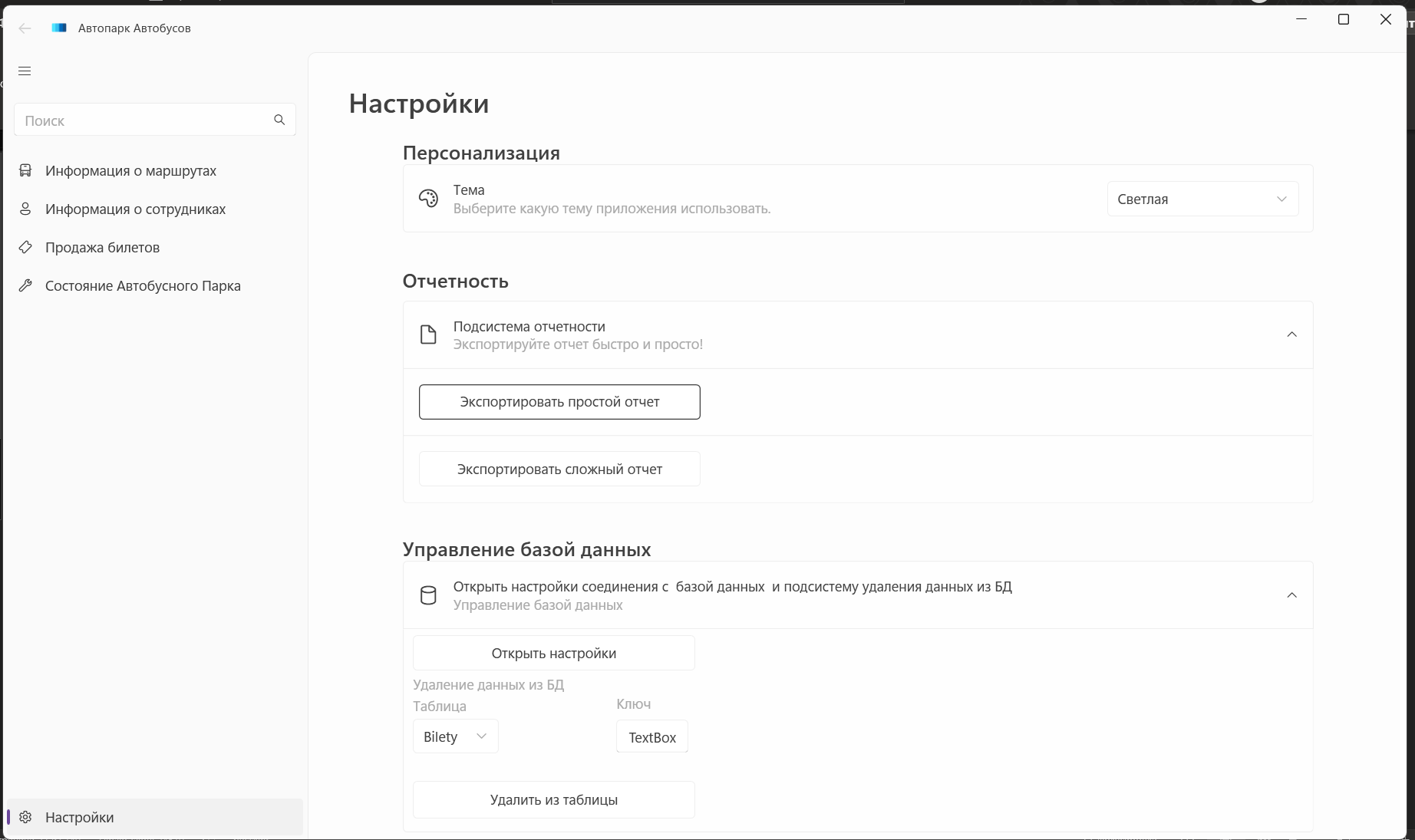


Рисунок 15 – Форма «Настройки».

**4 Функциональное и общее тестирование приложения**

**В программу из таблицы** ««**Employees**» базы данных Microsoft SQL server входят данные представленные в таблице 2.

Таблица 2 – Данные таблицы **Employees.**

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Пояснение | Тип данных | Ключ поля |
| **Num** | Код сотрудника | **bigint** | Первичный |
| **Surname** | **Фамилия сотрудника** | **ntext** |  |
| **Name** | **Имя сотрудника** | **ntext** |  |
| **Patronym** | **Отчество сотрудника** | **ntext** |  |
| **Employed\_Since** | **Работает с** | **ntext** |  |
| **Job\_Num** | **Внешний ключ с информацией о должности** | **bigint** | **Внешний** |

Таблица 3 – Данные таблицы Jobs**.**

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Пояснение | Тип данных | Ключ поля |
| **Job\_Key** | **Код должности** | **bigint** | Первичный |
| **Job** | **Должность** | **ntext** |  |
| **Internship** | **Стажировка** | **ntext** |  |

Из таблицы «**Avtobusy**» в программу входят данные представленные в таблице 5.

Из таблицы «Obsluzhivanie» в программу входят данные представленные в таблице 4.

Таблица 4 – Данные таблицы Obsluzhivanie**.**

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Пояснение | Тип данных | Ключ поля |
| 1 | 2 | 3 | 4 |
| NomerObsluzhivania | **Код обслуживания** | **bigint** | **Первичный** |
| ModelAvtobusaKey | **Внешний ключ с информацией о модели автобуса** | **bigint** | **Внешний** |
| **Data\_Poslednego\_Obsluzhivania** | **Дата последнего обслуживания** | **ntext** |  |

Продолжение таблицы 4

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| **Ingener\_Obsluzhivania** | **Инженер, проводивший обслуживание** | **ntext** |  |
| **Problemi\_Avtobusa** | **Найденные проблемы автобуса** | **ntext** |  |
| **Data\_Sledueschego\_Obsluzivania** | **Дата следующего обслуживания** | **ntext** |  |
| **Goden\_K\_Doroge** | **Годность автобуса к дороге** | **ntext** |  |

Таблица 5 – Данные таблицы Avtobusy.

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Пояснение | Тип данных | Ключ поля |
| Numavtobusy | Код | **bigint** | Первичный |
| **Model\_Avtobusa** | **Модель автобуса** | **ntext** |  |

Из таблицы «**Marshuti**» в программу входят данные представленные в таблице 6.

Таблица 6 – Данные таблицы **Marshuti.**

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Пояснение | Тип данных | Ключ поля |
| **Nomer\_Marshuta** | Код маршрута | **bigint** | Первичный |
| **Nachalni\_Punkt** | **Начальный пункт** | **ntext** |  |
| **Konechni\_Punkt** | **Конечный пункт** | **ntext** |  |
| **Emp\_Num** | **Код сотрудника автопарка, назначенного водителем на маршрут** | **bigint** |  |
| ModelAvtobusaKey | **Внешний ключ с информацией о модели автобуса** | **bigint** | **Внешний** |
| **Vremya\_Proezda** | **Время проезда маршрута** | **ntext** |  |

Из таблицы «Prodazhi» в программу входят данные представленные в таблице 7.

Таблица 7 – Данные таблицы Prodazhi

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Пояснение | Тип данных | Ключ поля |
| **Num** | Код | **bigint** | Первичный |
| **Sale\_Date** | **Дата продажи** | **ntext** |  |
| **Nomer\_BiletaKey** | **Внешний ключ с информацией о номере билета** |  |  |

Таблица 8 – Данные таблицы Bilety.

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Пояснение | Тип данных | Ключ поля |
| **Nomer\_Bileta** | Код | **bigint** | Первичный |
| **Nomer\_MarshutaKey** | **Внешний ключ с информацией о маршуте** | **bigint** | **Внешний** |
| **Stoimost** | **Стоимость билета** | **ntext** |  |

Данные, которые отправляются в базу данных из программы представлены в таблице 9.

Таблица 9 – Данные, которые отправляются в базу данных из программы.

|  |  |  |  |
| --- | --- | --- | --- |
| Выход | Имя Таблицы | Имя Поля Таблицы | Тип Поля |
| 1 | 2 | 3 | 4 |
| Выход в | **Employees** | **Num** | **bigint** |
| Выход в | **Employees** | **Surname** | **ntext** |
| Выход в | **Employees** | **Name** | **ntext** |
| Выход в | **Employees** | **Patronym** | **ntext** |
| Выход в | **Employees** | **Employed\_Since** | **ntext** |
| Выход в | **Employees** | **Job\_Num** | **bigint** |
| Выход в | **Jobs** | **Job\_Key** | **bigint** |
| Выход в | Jobs | **Job** | **ntext** |
| Выход в | **Jobs** | **Internship** | **ntext** |
| Выход в | Obsluzhivanie | **Num** | **bigint** |
| Выход в | Obsluzhivanie | **Model\_AvtobusaKey** | **bigint** |
| Выход в | Obsluzhivanie | **Data\_Poslednego\_Obsluzhivania** | **ntext** |
| Выход в | Obsluzhivanie | **Ingener\_Obsluzhivania** | **ntext** |
| Выход в | Obsluzhivanie | **Problemi\_Avtobusa** | **ntext** |
| Выход в | Obsluzhivanie | **Data\_Sledueschego\_Obsluzivania** | **ntext** |
| Выход в | Obsluzhivanie | **Goden\_K\_Doroge** | **ntext** |
| Выход в | Avtobusy | Numavtobusy | **bigint** |
| Выход в | Avtobusy | **Model\_Avtobusa** | **ntext** |
| Выход в | **Marshuti** | **Nomer\_Marshuta** | **bigint** |

Продолжение таблицы 9

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Выход в | **Marshuti** | **Nachalni\_Punkt** | **ntext** |
| Выход в | **Marshuti** | **Konechni\_Punkt** | **ntext** |
| Выход в | **Marshuti** | **Emp\_Num** | **bigint** |
| Выход в | **Marshuti** | ModelAvtobusaKey | **bigint** |
| Выход в | **Marshuti** | **Vremya\_Proezda** | **ntext** |
| Выход в | Prodazhi | **Num** | **bigint** |
| Выход в | Prodazhi | **Sale\_Date** | **ntext** |
| Выход в | Prodazhi | **Nomer\_BiletaKey** | **bigint** |
| Выход в | **Bilety** | **Nomer\_Bileta** | **bigint** |
| Выход в | **Bilety** | **Nomer\_MarshutaKey** | **bigint** |
| Выход в | **Bilety** | **Stoimost** | **ntext** |

Тестирование проводилось с целью проверки корректности работы программы и её отладки.

Тестирование программы осуществлялось на персональном компьютере со следующими техническими характеристиками:

* процессор – AMD Ryzen 5 7530U with Radeon Graphics 2.00 GHz
* видеокарта – AMD Radeon Vega 7 Graphics
* оперативная память - 16,00 ГБ LPDDR4
* операционная система – Windows 11

Для обеспечения устойчивости программы и базы данных в каждой форме присутствует проверка на ошибки. Сообщение об ошибке имеет вид приведенный на рисунке 13.

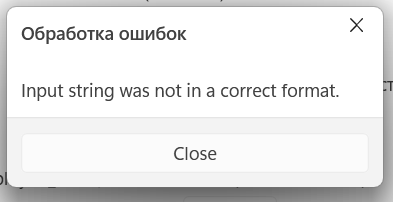


Рисунок 16 – Вид сообщения об ошибке

**В результате тестирования установлено, что программа работает корректно, выполняет все заложенные в нее функции.**

**Защита информации – комплекс действий, направленных на обеспечение информационной безопасности (целостности, доступности и конфиденциальности информации и ресурсов, используемых для обработки и передачи данных).**

**Система считается безопасной, если используя соответствующие средства, доступ к информации имеют только авторизованн**ые лица или же действующие от их имени процессы.

Очевидно, что абсолютно безопасных систем нет, и здесь речь идет о надежной системе в смысле «система, которой можно доверять». Система считается надежной, если она с использованием достаточных средств обеспечивает одновременную обработку информации разной степени секретности группой пользователей без нарушения прав доступа.

В программном модуле для автоматизации работы автопарка автобусов присутствует система входа для обеспечения различных прав доступа к базе данных. Вид формы приведен ниже на рисунке 17.

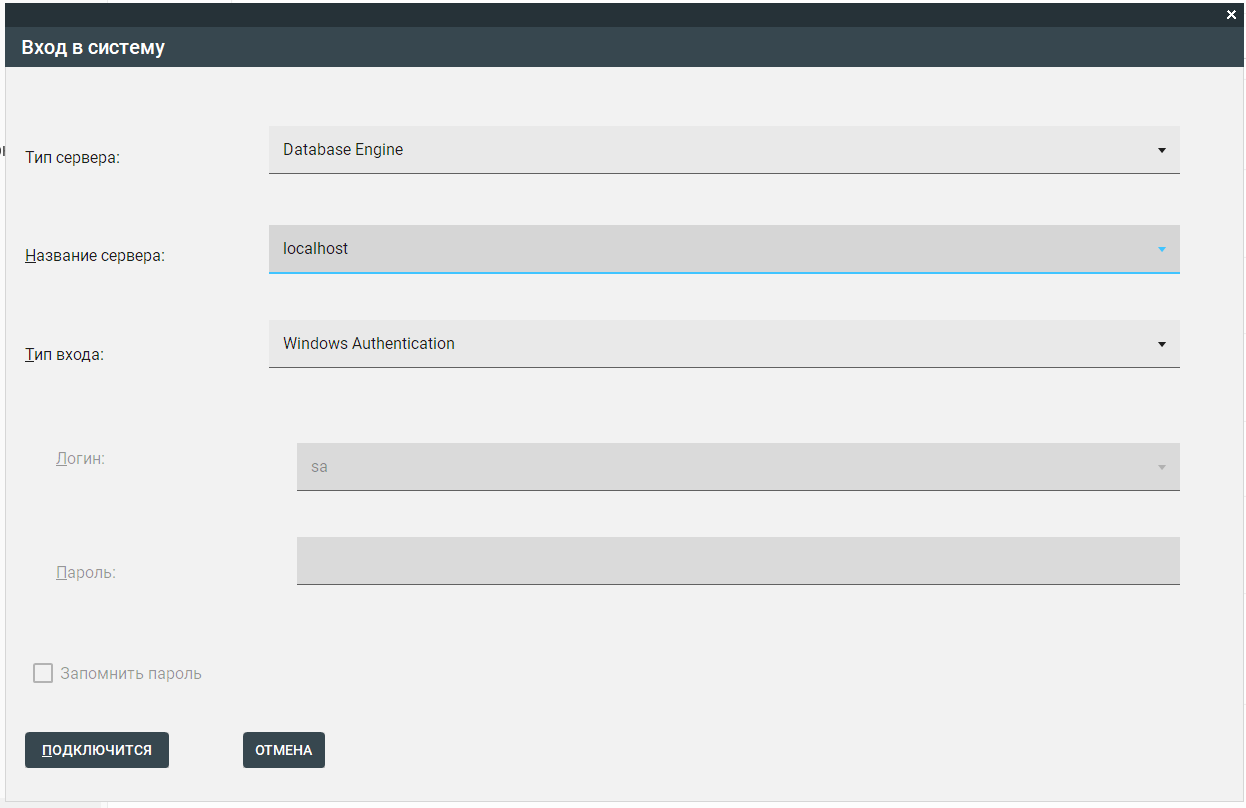


Рисунок 17 – Форма входа в систему

Вход в систему работает через логин и пароль от базы данных и позволяет при отсутствии у пользователя прав доступа к определенным таблицам выдать окно ошибки показанное на рисунке 16.

Проверка целостности осуществляется с помощью встроенных методов базы данных.

Разработанная справочная система для автоматизированной информационной системы «Автобусный автопарк» позволяет пользователю ознакомиться с краткой информацией об авторе программы и для чего она предназначена.

Вид справки «О программе» представлен на рисунке 18.

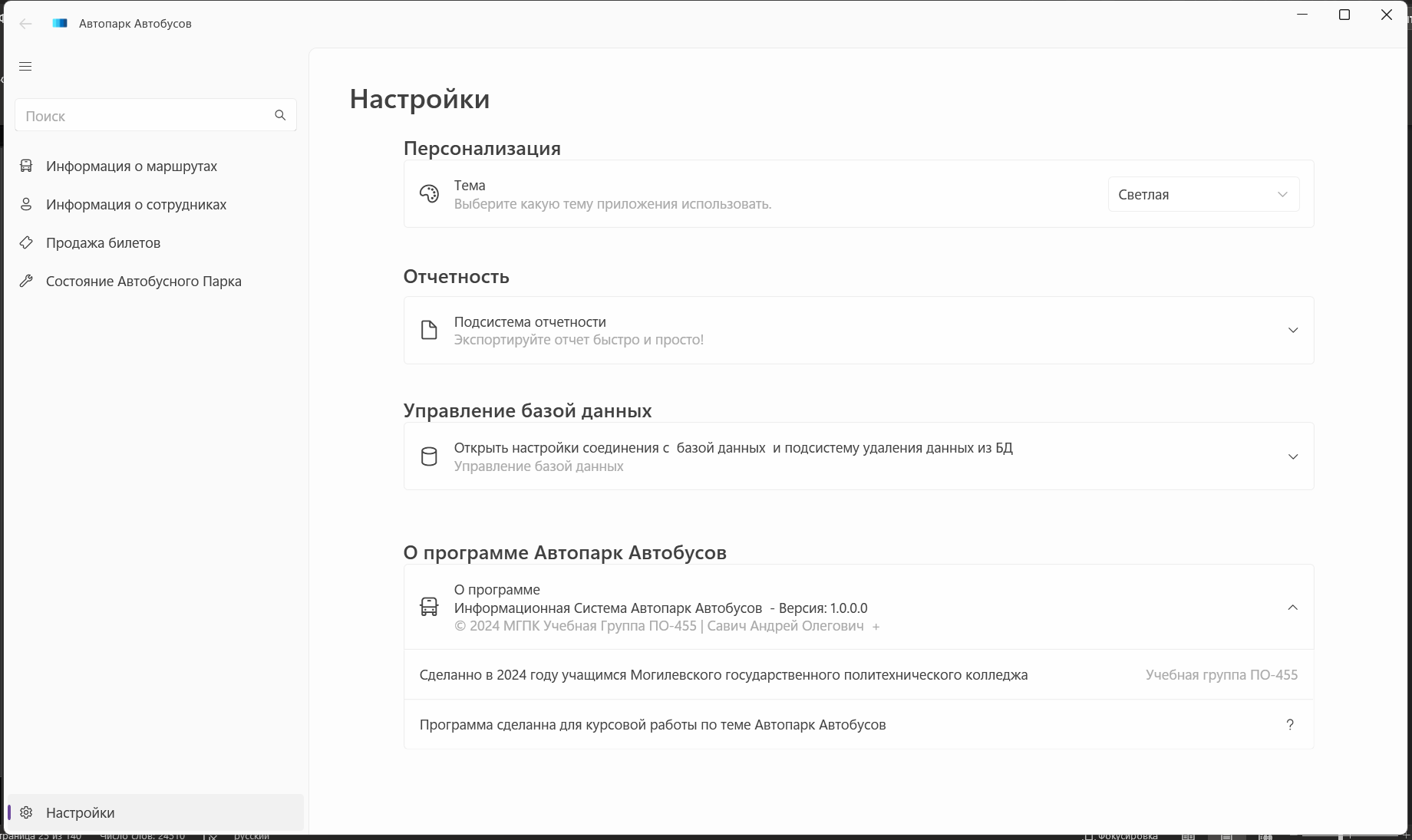


Рисунок 18 – Вид справки «О программе»

**5 Энерго- и ресурсосбережение**

В современном мире, в эпоху развивающейся промышленности необходимо принимать меры для сохранения окружающей нас среды. Поэтому в Республике Беларусь, как и в любой другой развитой стране, принята политика по ресурсо- и энергосбережению. Данная политика затрагивает все сферы общественной жизни, и конечно, не обошла и сферу образования. Стандарт управления энергопотреблением компьютеров, описывает три различных режима работы компьютера. Эти режимы отличаются потребляемой мощностью электроэнергии. Ждущий режим - позволяет экономить энергию, за счет отключения всех периферийных устройств, кроме ЦПУ, ОЗУ и материнской платы. Спящий режим - позволяет экономить электроэнергию на 100%, так как отключаются все периферийные устройства. Основной режим - при этом режиме экономии нет. Потребляемая мощность около 400Ватт. Для нахождения количества рабочих дней, в течение которых разрабатывался программный продукт, используем формулу:

n = Тпк /(8-tрп), (1)

где Тпк – время работы компьютера, ч;

Тпк =91,05 ч.

tрп – суммарное время регламентированных перерывов, в течение

рабочего дня, ч;

tрп =50 мин=5/6 часа.

дн



Для нахождения суммарной продолжительности регламентированных перерывов в течение всего времени разработки программного модуля:

Трп = n × tрп,(2)

где Трп – суммарная продолжительность регламентированных перерывов в течение всего времени разработки программного модуля;



Стоимость сэкономленной электроэнергии рассчитывается по формуле:

Сэн=Трп×(Wпк-Wсп)× Сэ, (3)

где Wпк – потребляемая мощность ПК, кВт;

Wсп – потребляемая мощность компьютера в ждущем режиме;

Сэ – стоимость 1 кВт электроэнергии, руб.

Wпк = 0,55 кВт;

Wсп = 0,35 кВт;

Сэ=0,31 руб/Кв;

Мероприятия по энерго и ресурсосбережению позволили сэкономить денежные средства в размере 62 копеек

### **Заключение**

В результате выполнения курсового проекта была разработана программа для автоматизации работы автопарка автобусов. Кроме этого были расширены знания из данной предметной области и укреплены знания языка программирования C#. При реализации проекта были пройдены этапы описания и постановки задачи автоматизации работы автопарка автобусов, кодирования программы на алгоритмический язык и тестирования полученного приложения.

Программа позволяет:

– просмотр отчетов;

– экспорт простых и сложных отчетов;

– просмотр и занесение информации о продажах билетов;

– просмотр и занесение информации о техническом обслуживании автобусов;

– просмотр и занесение информации о сотрудниках автопарка автобусов;

– просмотр и занесение информации о маршрутах автобусов;

– программа обладает интуитивно-понятным интерфейсом, проста и удобна в использовании.

### **Список используемых источников**

1. ГОСТ ИСО/ МЭК 2382-99. Информационные технологии. Словарь. Ч. 1. Основные термины.
2. ГОСТ19.701-90 (ИСО 5807-85). Описание символов. Правила применения символов и выполнения схем.
3. СТУ 01-32-2019. Стандарт учреждения. Общие требования к оформлению текстовых документов».
4. Абрамян, М.А. Visual C# на примерах / М.А. Абрамян. - Москва: БХВ-Петербург, 2016.
5. Вагнер, Б. С# Эффективное программирование / Б. Вагнер. - Москва: ЛОРИ, 2017.
6. Дейтел, П. Как программировать на Visual C# 2012 / П. Дейтел. - Москва: Питер, 2016.
7. Зиборов, В.В. Visual C# 2012 на примерах / В.В. Зиборов. - Москва: БХВ-Петербург, 2015.
8. Ишкова, Э. А. Самоучитель С#. Начала программирования / Э.А. Ишкова.- Москва: Наука и техника, 2017.
9. Магда Ю. С. NI Measurement Studio. Практика разработки систем измерения и управления на C# / Ю.С. Магда. - Москва: ДМК Пресс, 2017.
10. Прайс, Д.Visual C# 2.0. Полное руководство / Д.Прайс. - Москва: Век +, Корона-Век, Энтроп, 2015.
11. Рихтер,Д. CLR via C#. Программирование на платформе Microsoft.NET Framework 4.5 на языке C# / Д. Рихтер. - Москва: Питер, 2016.
12. Троелсен,Э.Язык программирования C#7 и платформы .NET и .NET Core/ Эндрю Троелсен,Филипп Джепикс – СПб.: ООО «Диалектика»,2018
13. Фленов, М.Е. Библия C# (+ CD-ROM) / М.Е. Фленов. - Москва: БХВ-Петербург, 2015.

# **Приложение А**

(обязательное)

**Текст программы**

using System;  
using System.Windows;  
using System.Windows.Controls;  
using Wpf.Ui.Controls;  
using FluentKursovayaAvtoparkA.Services.Contracts;  
using FluentKursovayaAvtoparkA.ViewModels.Windows;  
using FluentKursovayaAvtoparkA.Views.Pages;  
using Wpf.Ui;  
using Wpf.Ui.Appearance;  
using FluentKursovayaAvtoparkA.Controls;  
  
namespace FluentKursovayaAvtoparkA.Views.Windows;  
  
public partial class MainWindow : INavigationWindow  
{  
 public FluentKursovayaAvtoparkA.ViewModels.Windows.MainWindowViewModel ViewModel { get; }  
 private bool \_isUserClosedPane;  
  
 private bool \_isPaneOpenedOrClosedFromCode;  
 public static FluentKursovayaAvtoparkA.ViewModels.Windows.MainWindowViewModel ViewModelexport { get; set; }  
 //public static ContentPresenter ContentPresenterexport { get; set; }  
  
  
 public MainWindow(  
 FluentKursovayaAvtoparkA.ViewModels.Windows.MainWindowViewModel viewModel,  
 IPageService pageService,  
 INavigationService navigationService,  
 IServiceProvider serviceProvider,   
 //ISnackbarService snackbarService,  
 IContentDialogService contentDialogService  
 )  
 {  
 ViewModel = viewModel;  
 ViewModelexport = viewModel;  
 DataContext = this;  
   
  
  
  
 SystemThemeWatcher.*Watch*(this);  
  
 InitializeComponent();  
 //navigationService.SetNavigationControl(NavigationView);  
 ViewModel.\_dialogService.SetContentPresenter(RootContentDialog);  
  
 SetPageService(pageService);  
 //NavigationView.SetServiceProvider(serviceProvider);  
  
   
 //\_snackbarService = snackbarService;  
 // \_serviceProvider = serviceProvider;  
 }  
   
   
 #region INavigationWindow methods  
  
 public INavigationView GetNavigation() => NavigationView;  
  
 public bool Navigate(Type pageType) => NavigationView.Navigate(pageType);  
 public void SetServiceProvider(IServiceProvider serviceProvider)  
 {  
 throw new NotImplementedException();  
 }  
  
 public void SetPageService(IPageService pageService) => NavigationView.SetPageService(pageService);  
  
 public void ShowWindow() => Show();  
  
 public void CloseWindow() => Close();  
  
 #endregion INavigationWindow methods  
  
 */// <summary>  
 /// Raises the closed event.  
 /// </summary>* private void OnNavigationSelectionChanged(object sender, RoutedEventArgs e)  
 {  
 if (sender is not Wpf.Ui.Controls.NavigationView navigationView)  
 {  
 return;  
 }  
  
 NavigationView.HeaderVisibility =  
 navigationView.SelectedItem?.TargetPageType != typeof(DashboardPage)  
 ? Visibility.*Visible* : Visibility.*Collapsed*;  
 }  
  
 private void MainWindow\_OnSizeChanged(object sender, SizeChangedEventArgs e)  
 {  
 if (\_isUserClosedPane)  
 {  
 return;  
 }  
  
 \_isPaneOpenedOrClosedFromCode = true;  
 NavigationView.IsPaneOpen = !(e.NewSize.Width <= 1200);  
 \_isPaneOpenedOrClosedFromCode = false;  
 }  
  
 private void NavigationView\_OnPaneOpened(NavigationView sender, RoutedEventArgs args)  
 {  
 if (\_isPaneOpenedOrClosedFromCode)  
 {  
 return;  
 }  
  
 \_isUserClosedPane = false;  
 }  
  
 private void NavigationView\_OnPaneClosed(NavigationView sender, RoutedEventArgs args)  
 {  
 if (\_isPaneOpenedOrClosedFromCode)  
 {  
 return;  
 }  
  
 \_isUserClosedPane = true;  
 }  
  
 public async void FluentWindow\_Loaded(object sender, RoutedEventArgs e)  
 {  
 var termsOfUseContentDialog = new CallSQLConnectionSettingsContentDialog(ViewModel.\_dialogService.GetContentPresenter(), ViewModel);  
  
 \_ = await termsOfUseContentDialog.ShowAsync();  
 }  
}

using FastReport;

using FastReport.Data;

using FastReport.Export.Image;

using FastReport.Export.PdfSimple;

using FastReport.Format;

using FastReport.Table;

using FastReport.Utils;

using MsgBoxEx;

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.IO;

using System.Web.UI.WebControls;

using System.Web.UI.WebControls.WebParts;

using System.Windows;

namespace KursovayaAvtoparkAvtobusov

{

/// <summary>

/// Interaction logic for ReportExport.xaml

/// </summary>

public partial class ReportExport : Window

{

private static string outFolder = @"..\..\..\out\";

private static string inFolder = @"..\..\..\in\";

public ReportExport()

{

InitializeComponent();

}

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

{

RegisteredObjects.AddConnection(typeof(MsSqlDataConnection));

Report report = new Report();

MsSqlDataConnection connection = new MsSqlDataConnection();

connection.ConnectionString = ConfigurationManager.ConnectionStrings["KursovayaAvtoparkAvtobusov"].ToString();

connection.CreateAllTables();

report.Dictionary.Connections.Add(connection);

report.RegisterData(connection.DataSet, "mssql");

report.GetDataSource("Employees").Enabled = true;

report.GetDataSource("Maintenance").Enabled = true;

report.GetDataSource("Marshuti").Enabled = true;

report.GetDataSource("Prodazhi").Enabled = true;

ReportPage page = new ReportPage();

report.Pages.Add(page);

page.CreateUniqueName();

page.TopMargin = 10.0f;

page.LeftMargin = 10.0f;

page.RightMargin = 10.0f;

page.BottomMargin = 10.0f;

DataBand data = new DataBand();

data.CreateUniqueName();

page.AddChild(data);

data.DataSource = report.GetDataSource("Employees"); //adding a table to a bend

data.Height = Units.Centimeters \* 1.5f;

// Creating report title

page.ReportTitle = new ReportTitleBand();

page.ReportTitle.CreateUniqueName();

page.ReportTitle.Height = 4.0f \* Units.Centimeters;

TextObject titleText = new TextObject();

titleText.CreateUniqueName();

titleText.Left = 1.0f \* Units.Centimeters;

titleText.Top = 1.0f \* Units.Centimeters;

titleText.Width = 17.0f \* Units.Centimeters;

titleText.Height = 2.0f \* Units.Centimeters;

titleText.HorzAlign = HorzAlign.Center;

titleText.VertAlign = VertAlign.Center;

titleText.Font = new Font("Times New Roman", 32.0f);

titleText.TextColor = Color.Black;

titleText.FillColor = Color.Wheat;

titleText.Border.Color = Color.Black;

titleText.Border.Lines = BorderLines.All;

titleText.Border.Width = 4.0f;

titleText.Text = "Avtopark Report";

page.ReportTitle.Objects.Add(titleText);

// Header object

TextObject headerText = new TextObject();

headerText.CreateUniqueName();

headerText.Bounds = new RectangleF(0.0f, 2.7f \* Units.Centimeters,

19.0f \* Units.Centimeters, 0.8f \* Units.Centimeters);

headerText.HorzAlign = HorzAlign.Center;

headerText.VertAlign = VertAlign.Center;

headerText.Font = new Font("Times New Roman", 16.0f);

headerText.TextColor = Color.Black;

headerText.FillColor = Color.Wheat;

headerText.Border.Color = Color.Black;

headerText.Border.Lines = BorderLines.All;

headerText.Border.Width = 4.0f;

headerText.Text = "Employees" + " Table";

page.ReportTitle.Objects.Add(headerText);

// Employee Table Text bands

TextObject textx = new TextObject();

textx.Parent = data;

textx.CreateUniqueName();

textx.Bounds = new RectangleF(0 \* Units.Centimeters, 0, 1.0f \* Units.Centimeters, Units.Centimeters \* 0.5f);

textx.Font = new Font("Times New Roman", 5.0f);

textx.Text = "Num"; //table and its field

textx.Border.Lines = BorderLines.All;

TextObject textxx = new TextObject();

textxx.Parent = data;

textxx.CreateUniqueName();

textxx.Bounds = new RectangleF(1 \* Units.Centimeters, 0, Units.Centimeters \* 3, Units.Centimeters \* 0.5f);

textxx.Font = new Font("Times New Roman", 5.0f);

textxx.Text = "Surname"; //table and its field

textxx.Border.Lines = BorderLines.All;

TextObject textxxx = new TextObject();

textxxx.Parent = data;

textxxx.CreateUniqueName();

textxxx.Bounds = new RectangleF(4.0f \* Units.Centimeters, 0, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

textxxx.Font = new Font("Times New Roman", 5.0f);

textxxx.Text = "Name"; //table and its field

textxxx.Border.Lines = BorderLines.All;

TextObject textxxxx = new TextObject();

textxxxx.Parent = data;

textxxxx.CreateUniqueName();

textxxxx.Bounds = new RectangleF(7.4f \* Units.Centimeters, 0, Units.Centimeters \* 25, Units.Centimeters \* 0.5f);

textxxxx.Font = new Font("Times New Roman", 5.0f);

textxxxx.Text = "Patronym"; //table and its field

textxxxx.Border.Lines = BorderLines.All;

TextObject te = new TextObject();

te.Parent = data;

te.CreateUniqueName();

te.Bounds = new RectangleF(9.9f \* Units.Centimeters, 0, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

te.Font = new Font("Times New Roman", 5.0f);

te.Text = "Employed\_Since"; //table and its field

te.Border.Lines = BorderLines.All;

TextObject ter = new TextObject();

ter.Parent = data;

ter.CreateUniqueName();

ter.Bounds = new RectangleF(14 \* Units.Centimeters, 0, Units.Centimeters \* 30, Units.Centimeters \* 0.5f);

ter.Font = new Font("Times New Roman", 5.0f);

ter.Text = "Job"; //table and its field

ter.Border.Lines = BorderLines.All;

TextObject teXR = new TextObject();

teXR.Parent = data;

teXR.CreateUniqueName();

teXR.Bounds = new RectangleF(17.5f \* Units.Centimeters, 0, Units.Centimeters \* 25, Units.Centimeters \* 0.5f);

teXR.Font = new Font("Times New Roman", 5.0f);

teXR.Text = "Internship"; //table and its field

teXR.Border.Lines = BorderLines.All;

// Employee Table Fields

TextObject bandText = new TextObject();

bandText.CreateUniqueName();

bandText.HorzAlign = HorzAlign.Center;

bandText.Bounds = new RectangleF(0 \* Units.Centimeters, 25,

1.0f \* Units.Centimeters, 0.5f \* Units.Centimeters);

bandText.Font = new Font("Times New Roman", 5.0f);

bandText.Border.Lines = BorderLines.All;

bandText.Text = "[Employees.Num]";

data.AddChild(bandText);

TextObject text1 = new TextObject();

text1.Parent = data;

text1.CreateUniqueName();

text1.Bounds = new RectangleF(1 \* Units.Centimeters, 25, Units.Centimeters \* 3, Units.Centimeters \* 0.5f);

text1.Font = new Font("Times New Roman", 5.0f);

text1.Text = "[Employees.Surname]"; //table and its field

text1.Border.Lines = BorderLines.All;

TextObject text2 = new TextObject();

text2.Parent = data;

text2.CreateUniqueName();

text2.Bounds = new RectangleF(4.0f \* Units.Centimeters, 25, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

text2.Font = new Font("Times New Roman", 5.0f);

text2.Text = "[Employees.Name]"; //table and its field

text2.Border.Lines = BorderLines.All;

TextObject text3 = new TextObject();

text3.Parent = data;

text3.CreateUniqueName();

text3.Bounds = new RectangleF(7.4f \* Units.Centimeters, 25, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

text3.Font = new Font("Times New Roman", 5.0f);

text3.Text = "[Employees.Patronym]"; //table and its field

text3.Border.Lines = BorderLines.All;

TextObject text4 = new TextObject();

text4.Parent = data;

text4.CreateUniqueName();

text4.Bounds = new RectangleF(9.9f \* Units.Centimeters, 25, Units.Centimeters \* 30, Units.Centimeters \* 0.5f);

text4.Font = new Font("Times New Roman", 5.0f);

text4.Text = "[Employees.Employed\_Since]"; //table and its field

text4.Border.Lines = BorderLines.All;

TextObject text5 = new TextObject();

text5.Parent = data;

text5.CreateUniqueName();

text5.Bounds = new RectangleF(14 \* Units.Centimeters, 25, Units.Centimeters \* 5, Units.Centimeters \* 0.5f);

text5.Font = new Font("Times New Roman", 5.0f);

text5.Text = "[Employees.Job]"; //table and its field

text5.Border.Lines = BorderLines.All;

TextObject text6 = new TextObject();

text6.Parent = data;

text6.CreateUniqueName();

text6.Bounds = new RectangleF(17.5f \* Units.Centimeters, 25, Units.Centimeters \* 5, Units.Centimeters \* 0.5f);

text6.Font = new Font("Times New Roman", 5.0f);

text6.Text = "[Employees.Internship]"; //table and its field

text6.Border.Lines = BorderLines.All;

// Maintenance table things\

DataBand datax = new DataBand();

datax.CreateUniqueName();

page.AddChild(datax);

datax.DataSource = report.GetDataSource("Maintenance"); //adding a table to a bend

datax.Height = Units.Centimeters \* 5f;

//Maintenance Header object

TextObject headerSText = new TextObject();

headerSText.CreateUniqueName();

headerSText.Bounds = new RectangleF(0.0f, 7.1f \* Units.Centimeters,

19.0f \* Units.Centimeters, 0.8f \* Units.Centimeters);

headerSText.HorzAlign = HorzAlign.Center;

headerSText.VertAlign = VertAlign.Center;

headerSText.Font = new Font("Times New Roman", 16.0f);

headerSText.TextColor = Color.Black;

headerSText.FillColor = Color.Wheat;

headerSText.Border.Color = Color.Black;

headerSText.Border.Lines = BorderLines.All;

headerSText.Border.Width = 4.0f;

headerSText.Text = "Maintenance" + " Table";

page.ReportTitle.Objects.Add(headerSText);

//Maintenance Table Text bands

TextObject textxxrx = new TextObject();

textxxrx.Parent = datax;

textxxrx.CreateUniqueName();

textxxrx.Bounds = new RectangleF(0 \* Units.Centimeters, 39.9f, 0.7f \* Units.Centimeters, Units.Centimeters \* 0.5f);

textxxrx.Font = new Font("Times New Roman", 6.0f);

textxxrx.Text = "Num"; //table and its field

textxxrx.Border.Lines = BorderLines.All;

TextObject textxxxr = new TextObject();

textxxxr.Parent = datax;

textxxxr.CreateUniqueName();

textxxxr.Font = new Font("Times New Roman", 6.0f);

textxxxr.Bounds = new RectangleF(0.7f \* Units.Centimeters, 39.9f, Units.Centimeters \* 7, Units.Centimeters \* 0.5f);

textxxxr.Text = "Model\_Avtobusa"; //table and its field

textxxxr.Border.Lines = BorderLines.All;

TextObject textxxxxr = new TextObject();

textxxxxr.Parent = datax;

textxxxxr.CreateUniqueName();

textxxxxr.Bounds = new RectangleF(4.5f \* Units.Centimeters, 39.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

textxxxxr.Font = new Font("Times New Roman", 5.0f);

textxxxxr.Text = "Data\_Poslednego\_Obsluzhivania"; //table and its field

textxxxxr.Border.Lines = BorderLines.All;

TextObject textxxxxx = new TextObject();

textxxxxx.Parent = datax;

textxxxxx.Font = new Font("Times New Roman", 5.0f);

textxxxxx.CreateUniqueName();

textxxxxx.Bounds = new RectangleF(7.7f \* Units.Centimeters, 39.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

textxxxxx.Text = "Ingener\_Obsluzhivania"; //table and its field

textxxxxx.Border.Lines = BorderLines.All;

TextObject terzx = new TextObject();

terzx.Parent = datax;

terzx.CreateUniqueName();

terzx.Bounds = new RectangleF(10.4f \* Units.Centimeters, 39.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terzx.Font = new Font("Times New Roman", 5.0f);

terzx.Text = "Problemi\_Avtobusa"; //table and its field

terzx.Border.Lines = BorderLines.All;

TextObject terxr = new TextObject();

terxr.Parent = datax;

terxr.CreateUniqueName();

terxr.Bounds = new RectangleF(15 \* Units.Centimeters, 39.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

terxr.Font = new Font("Times New Roman", 4.0f);

terxr.Text = "Data\_Sledueschego\_Obsluzivania"; //table and its field

terxr.Border.Lines = BorderLines.All;

TextObject teXRr = new TextObject();

teXRr.Parent = datax;

teXRr.CreateUniqueName();

teXRr.Bounds = new RectangleF(17.5f \* Units.Centimeters, 39.9f, Units.Centimeters \* 26, Units.Centimeters \* 0.5f);

teXRr.Font = new Font("Times New Roman", 6.0f);

teXRr.Text = "Goden\_K\_Doroge"; //table and its field

teXRr.Border.Lines = BorderLines.All;

//Maintenance table fields

TextObject textxxrxx = new TextObject();

textxxrxx.Parent = datax;

textxxrxx.CreateUniqueName();

textxxrxx.Bounds = new RectangleF(0 \* Units.Centimeters, 60.9f, 0.7f \* Units.Centimeters, Units.Centimeters \* 0.5f);

textxxrxx.Font = new Font("Times New Roman", 6.0f);

textxxrxx.Text = "[Maintenance.Num]"; //table and its field

textxxrxx.Border.Lines = BorderLines.All;

TextObject textxxxrx = new TextObject();

textxxxrx.Parent = datax;

textxxxrx.CreateUniqueName();

textxxxrx.Font = new Font("Times New Roman", 6.0f);

textxxxrx.Bounds = new RectangleF(0.7f \* Units.Centimeters, 60.9f, Units.Centimeters \* 7, Units.Centimeters \* 0.5f);

textxxxrx.Text = "[Maintenance.Model\_Avtobusa]"; //table and its field

textxxxrx.Border.Lines = BorderLines.All;

TextObject textxxxxrx = new TextObject();

textxxxxrx.Parent = datax;

textxxxxrx.CreateUniqueName();

textxxxxrx.Bounds = new RectangleF(4.5f \* Units.Centimeters, 60.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

textxxxxrx.Font = new Font("Times New Roman", 5.0f);

textxxxxrx.Text = "[Maintenance.Data\_Poslednego\_Obsluzhivania]"; //table and its field

textxxxxrx.Border.Lines = BorderLines.All;

TextObject textxxxxxx = new TextObject();

textxxxxxx.Parent = datax;

textxxxxxx.Font = new Font("Times New Roman", 5.0f);

textxxxxxx.CreateUniqueName();

textxxxxxx.Bounds = new RectangleF(7.7f \* Units.Centimeters, 60.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

textxxxxxx.Text = "[Maintenance.Ingener\_Obsluzhivania]"; //table and its field

textxxxxxx.Border.Lines = BorderLines.All;

TextObject terzxx = new TextObject();

terzxx.Parent = datax;

terzxx.CreateUniqueName();

terzxx.Bounds = new RectangleF(10.4f \* Units.Centimeters, 60.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terzxx.Font = new Font("Times New Roman", 5.0f);

terzxx.Text = "[Maintenance.Problemi\_Avtobusa]"; //table and its field

terzxx.Border.Lines = BorderLines.All;

TextObject terxrx = new TextObject();

terxrx.Parent = datax;

terxrx.CreateUniqueName();

terxrx.Bounds = new RectangleF(15 \* Units.Centimeters, 60.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

terxrx.Font = new Font("Times New Roman", 6.0f);

terxrx.Text = "[Maintenance.Data\_Sledueschego\_Obsluzivania]"; //table and its field

terxrx.Border.Lines = BorderLines.All;

TextObject teXRrr = new TextObject();

teXRrr.Parent = datax;

teXRrr.CreateUniqueName();

teXRrr.Bounds = new RectangleF(17.5f \* Units.Centimeters, 60.9f, Units.Centimeters \* 26, Units.Centimeters \* 0.5f);

teXRrr.Font = new Font("Times New Roman", 6.0f);

teXRrr.Text = "[Maintenance.Goden\_K\_Doroge]"; //table and its field

teXRrr.Border.Lines = BorderLines.All;

// Prodazhi Header

TextObject headersSText = new TextObject();

headersSText.CreateUniqueName();

headersSText.Bounds = new RectangleF(0.0f, 11.1f \* Units.Centimeters,

19.0f \* Units.Centimeters, 0.8f \* Units.Centimeters);

headersSText.HorzAlign = HorzAlign.Center;

headersSText.VertAlign = VertAlign.Center;

headersSText.Font = new Font("Times New Roman", 16.0f);

headersSText.TextColor = Color.Black;

headersSText.FillColor = Color.Wheat;

headersSText.Border.Color = Color.Black;

headersSText.Border.Lines = BorderLines.All;

headersSText.Border.Width = 4.0f;

headersSText.Text = "Prodazhi" + " Table";

page.ReportTitle.Objects.Add(headersSText);

// Prodazhi Table Info

DataBand dataxr = new DataBand();

dataxr.CreateUniqueName();

page.AddChild(dataxr);

dataxr.DataSource = report.GetDataSource("Prodazhi"); //adding a table to a bend

dataxr.Height = Units.Centimeters \* 5f;

TextObject texrxx = new TextObject();

texrxx.Parent = dataxr;

texrxx.CreateUniqueName();

texrxx.Bounds = new RectangleF(0 \* Units.Centimeters, 43.99f, 0.7f \* Units.Centimeters, Units.Centimeters \* 0.5f);

texrxx.Font = new Font("Times New Roman", 6.0f);

texrxx.Text = "Num"; //table and its field

texrxx.Border.Lines = BorderLines.All;

TextObject texrxs = new TextObject();

texrxs.Parent = dataxr;

texrxs.CreateUniqueName();

texrxs.Font = new Font("Times New Roman", 6.0f);

texrxs.Bounds = new RectangleF(0.7f \* Units.Centimeters, 43.99f, Units.Centimeters \* 7, Units.Centimeters \* 0.5f);

texrxs.Text = "Sale\_Date"; //table and its field

texrxs.Border.Lines = BorderLines.All;

TextObject terscw = new TextObject();

terscw.Parent = dataxr;

terscw.CreateUniqueName();

terscw.Bounds = new RectangleF(4.5f \* Units.Centimeters, 43.99f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

terscw.Font = new Font("Times New Roman", 5.0f);

terscw.Text = "Nomer\_Avtobusa"; //table and its field

terscw.Border.Lines = BorderLines.All;

TextObject terscwx = new TextObject();

terscwx.Parent = dataxr;

terscwx.Font = new Font("Times New Roman", 5.0f);

terscwx.CreateUniqueName();

terscwx.Bounds = new RectangleF(7.7f \* Units.Centimeters, 43.99f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terscwx.Text = "Punkt\_Posadki"; //table and its field

terscwx.Border.Lines = BorderLines.All;

TextObject terzrx = new TextObject();

terzrx.Parent = dataxr;

terzrx.CreateUniqueName();

terzrx.Bounds = new RectangleF(10.4f \* Units.Centimeters, 43.99f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terzrx.Font = new Font("Times New Roman", 5.0f);

terzrx.Text = "Stoimost"; //table and its field

terzrx.Border.Lines = BorderLines.All;

// Table Fields For Prodazhi

TextObject texrxxx = new TextObject();

texrxxx.Parent = dataxr;

texrxxx.CreateUniqueName();

texrxxx.Bounds = new RectangleF(0 \* Units.Centimeters, 53.99f, 0.7f \* Units.Centimeters, Units.Centimeters \* 0.5f);

texrxxx.Font = new Font("Times New Roman", 6.0f);

texrxxx.Text = "[Prodazhi.Num]"; //table and its field

texrxxx.Border.Lines = BorderLines.All;

TextObject texrxss = new TextObject();

texrxss.Parent = dataxr;

texrxss.CreateUniqueName();

texrxss.Font = new Font("Times New Roman", 6.0f);

texrxss.Bounds = new RectangleF(0.7f \* Units.Centimeters, 53.99f, Units.Centimeters \* 7, Units.Centimeters \* 0.5f);

texrxss.Text = "[Prodazhi.Sale\_Date]"; //table and its field

texrxss.Border.Lines = BorderLines.All;

TextObject terscwxxxr = new TextObject();

terscwxxxr.Parent = dataxr;

terscwxxxr.CreateUniqueName();

terscwxxxr.Bounds = new RectangleF(4.5f \* Units.Centimeters, 53.99f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

terscwxxxr.Font = new Font("Times New Roman", 5.0f);

terscwxxxr.Text = "[Prodazhi.Nomer\_Avtobusa]"; //table and its field

terscwxxxr.Border.Lines = BorderLines.All;

TextObject terscwxxr = new TextObject();

terscwxxr.Parent = dataxr;

terscwxxr.Font = new Font("Times New Roman", 5.0f);

terscwxxr.CreateUniqueName();

terscwxxr.Bounds = new RectangleF(7.7f \* Units.Centimeters, 53.99f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terscwxxr.Text = "[Prodazhi.Punkt\_Posadki]"; //table and its field

terscwxxr.Border.Lines = BorderLines.All;

TextObject terzrxr = new TextObject();

terzrxr.Parent = dataxr;

terzrxr.CreateUniqueName();

terzrxr.Bounds = new RectangleF(10.4f \* Units.Centimeters, 53.99f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terzrxr.Font = new Font("Times New Roman", 5.0f);

terzrxr.Text = "[Prodazhi.Stoimost]"; //table and its field

terzrxr.Border.Lines = BorderLines.All;

// Marshuti table header

TextObject headersrSText = new TextObject();

headersrSText.CreateUniqueName();

headersrSText.Bounds = new RectangleF(0.0f, 15.1f \* Units.Centimeters,

19.0f \* Units.Centimeters, 0.8f \* Units.Centimeters);

headersrSText.HorzAlign = HorzAlign.Center;

headersrSText.VertAlign = VertAlign.Center;

headersrSText.Font = new Font("Times New Roman", 16.0f);

headersrSText.TextColor = Color.Black;

headersrSText.FillColor = Color.Wheat;

headersrSText.Border.Color = Color.Black;

headersrSText.Border.Lines = BorderLines.All;

headersrSText.Border.Width = 4.0f;

headersrSText.Text = "Marshuti" + " Table";

page.ReportTitle.Objects.Add(headersrSText);

// Marshuti fields

DataBand datarxr = new DataBand();

datarxr.CreateUniqueName();

page.AddChild(datarxr);

datarxr.DataSource = report.GetDataSource("Marshuti"); //adding a table to a bend

datarxr.Height = Units.Centimeters \* 2f;

TextObject txtef = new TextObject();

txtef.Parent = datarxr;

txtef.CreateUniqueName();

txtef.Bounds = new RectangleF(0 \* Units.Centimeters, 58.9f, 0.7f \* Units.Centimeters, Units.Centimeters \* 0.5f);

txtef.Font = new Font("Times New Roman", 6.0f);

txtef.Text = "Nomer\_Marshuta"; //table and its field

txtef.Border.Lines = BorderLines.All;

TextObject tfeqa = new TextObject();

tfeqa.Parent = datarxr;

tfeqa.CreateUniqueName();

tfeqa.Font = new Font("Times New Roman", 6.0f);

tfeqa.Bounds = new RectangleF(0.7f \* Units.Centimeters, 58.9f, Units.Centimeters \* 7, Units.Centimeters \* 0.5f);

tfeqa.Text = "Nachalni\_Punkt"; //table and its field

tfeqa.Border.Lines = BorderLines.All;

TextObject tcqqcq = new TextObject();

tcqqcq.Parent = datarxr;

tcqqcq.CreateUniqueName();

tcqqcq.Bounds = new RectangleF(4.5f \* Units.Centimeters, 58.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

tcqqcq.Font = new Font("Times New Roman", 5.0f);

tcqqcq.Text = "Konechni\_Punkt"; //table and its field

tcqqcq.Border.Lines = BorderLines.All;

TextObject txtweq = new TextObject();

txtweq.Parent = datarxr;

txtweq.Font = new Font("Times New Roman", 5.0f);

txtweq.CreateUniqueName();

txtweq.Bounds = new RectangleF(7.7f \* Units.Centimeters, 58.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

txtweq.Text = "Voditel"; //table and its field

txtweq.Border.Lines = BorderLines.All;

TextObject terrzx = new TextObject();

terrzx.Parent = datarxr;

terrzx.CreateUniqueName();

terrzx.Bounds = new RectangleF(10.4f \* Units.Centimeters, 58.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terrzx.Font = new Font("Times New Roman", 5.0f);

terrzx.Text = "Model\_Avtobusa"; //table and its field

terrzx.Border.Lines = BorderLines.All;

TextObject txerxr = new TextObject();

txerxr.Parent = datarxr;

txerxr.CreateUniqueName();

txerxr.Bounds = new RectangleF(15 \* Units.Centimeters, 58.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

txerxr.Font = new Font("Times New Roman", 4.0f);

txerxr.Text = "Vremya\_Proezda"; //table and its field

txerxr.Border.Lines = BorderLines.All;

TextObject txteff = new TextObject();

txteff.Parent = datarxr;

txteff.CreateUniqueName();

txteff.Bounds = new RectangleF(0 \* Units.Centimeters, 82.9f, 0.7f \* Units.Centimeters, Units.Centimeters \* 0.5f);

txteff.Font = new Font("Times New Roman", 6.0f);

txteff.Text = "[Marshuti.Nomer\_Marshuta]"; //table and its field

txteff.Border.Lines = BorderLines.All;

TextObject tfeqafd = new TextObject();

tfeqafd.Parent = datarxr;

tfeqafd.CreateUniqueName();

tfeqafd.Font = new Font("Times New Roman", 6.0f);

tfeqafd.Bounds = new RectangleF(0.7f \* Units.Centimeters, 82.9f, Units.Centimeters \* 7, Units.Centimeters \* 0.5f);

tfeqafd.Text = "[Marshuti.Nachalni\_Punkt]"; //table and its field

tfeqafd.Border.Lines = BorderLines.All;

TextObject tcqqcqd = new TextObject();

tcqqcqd.Parent = datarxr;

tcqqcqd.CreateUniqueName();

tcqqcqd.Bounds = new RectangleF(4.5f \* Units.Centimeters, 82.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

tcqqcqd.Font = new Font("Times New Roman", 5.0f);

tcqqcqd.Text = "[Marshuti.Konechni\_Punkt]"; //table and its field

tcqqcqd.Border.Lines = BorderLines.All;

TextObject txtweqq = new TextObject();

txtweqq.Parent = datarxr;

txtweqq.Font = new Font("Times New Roman", 5.0f);

txtweqq.CreateUniqueName();

txtweqq.Bounds = new RectangleF(7.7f \* Units.Centimeters, 82.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

txtweqq.Text = "[Marshuti.Voditel]"; //table and its field

txtweqq.Border.Lines = BorderLines.All;

TextObject terrzxr = new TextObject();

terrzxr.Parent = datarxr;

terrzxr.CreateUniqueName();

terrzxr.Bounds = new RectangleF(10.4f \* Units.Centimeters, 82.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terrzxr.Font = new Font("Times New Roman", 5.0f);

terrzxr.Text = "[Marshuti.Model\_Avtobusa]"; //table and its field

terrzxr.Border.Lines = BorderLines.All;

TextObject txerxxr = new TextObject();

txerxxr.Parent = datarxr;

txerxxr.CreateUniqueName();

txerxxr.Bounds = new RectangleF(15 \* Units.Centimeters, 82.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

txerxxr.Font = new Font("Times New Roman", 4.0f);

txerxxr.Text = "[Marshuti.Vremya\_Proezda]"; //table and its field

txerxxr.Border.Lines = BorderLines.All;

// Footer

page.PageFooter = new PageFooterBand();

page.PageFooter.CreateUniqueName();

page.PageFooter.Height = 0.5f \* Units.Centimeters;

TextObject footerText = new TextObject();

footerText.CreateUniqueName();

footerText.HorzAlign = HorzAlign.Right;

footerText.VertAlign = VertAlign.Center;

footerText.Bounds = new RectangleF(0.0f, 0.0f,

19.0f \* Units.Centimeters, 0.5f \* Units.Centimeters);

footerText.TextColor = Color.Black;

footerText.FillColor = Color.Wheat;

titleText.Border.Color = Color.Black;

titleText.Border.Lines = BorderLines.All;

titleText.Border.Width = 4.0f;

footerText.Text = "Page [Page]";

page.PageFooter.Objects.Add(footerText);

try

{

// Exporting the report

report.Prepare();

report.SavePrepared("Prepared Report.fpx");

PDFSimpleExport pdf = new PDFSimpleExport();

// Save the report

report.Export(pdf, "ExportedPDF.pdf");

}

catch (Exception ex)

{

MsgBoxExtendedFunctionality ext = new MsgBoxExtendedFunctionality()

{

DetailsText = ex.StackTrace

};

MessageBoxEx.ShowEx("An error has occured! Traceback: " + ex.Message, "Unexpected situation handling", MessageBoxButtonEx.OK, MessageBoxImage.Error, ext);

}

}

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

{

RegisteredObjects.AddConnection(typeof(MsSqlDataConnection));

Report report = new Report();

MsSqlDataConnection connection = new MsSqlDataConnection();

connection.ConnectionString = ConfigurationManager.ConnectionStrings["KursovayaAvtoparkAvtobusov"].ToString();

connection.CreateAllTables();

report.Dictionary.Connections.Add(connection);

report.RegisterData(connection.DataSet, "mssql");

report.GetDataSource("Employees").Enabled = true;

report.GetDataSource("Maintenance").Enabled = true;

report.GetDataSource("Marshuti").Enabled = true;

report.GetDataSource("Prodazhi").Enabled = true;

ReportPage page = new ReportPage();

report.Pages.Add(page);

page.CreateUniqueName();

page.TopMargin = 10.0f;

page.LeftMargin = 10.0f;

page.RightMargin = 10.0f;

page.BottomMargin = 10.0f;

DataBand data = new DataBand();

data.CreateUniqueName();

page.AddChild(data);

data.DataSource = report.GetDataSource("Employees"); //adding a table to a bend

data.Height = Units.Centimeters \* 1.5f;

// Creating report title

page.ReportTitle = new ReportTitleBand();

page.ReportTitle.CreateUniqueName();

page.ReportTitle.Height = 4.0f \* Units.Centimeters;

TextObject title2Text = new TextObject();

title2Text.CreateUniqueName();

title2Text.Left = 1.0f \* Units.Centimeters;

title2Text.Top = 1.0f \* Units.Centimeters;

title2Text.Width = 17.0f \* Units.Centimeters;

title2Text.Height = 2.0f \* Units.Centimeters;

title2Text.HorzAlign = HorzAlign.Center;

title2Text.VertAlign = VertAlign.Center;

title2Text.Font = new Font("Times New Roman", 32.0f);

title2Text.TextColor = Color.Black;

title2Text.FillColor = Color.Wheat;

title2Text.Border.Color = Color.Black;

title2Text.Border.Lines = BorderLines.All;

title2Text.Border.Width = 4.0f;

title2Text.Text = "Avtopark Report";

page.ReportTitle.Objects.Add(title2Text);

// Header object

TextObject headerText = new TextObject();

headerText.CreateUniqueName();

headerText.Bounds = new RectangleF(0.0f, 3.2f \* Units.Centimeters,

19.0f \* Units.Centimeters, 0.8f \* Units.Centimeters);

headerText.HorzAlign = HorzAlign.Left;

headerText.VertAlign = VertAlign.Center;

headerText.Font = new Font("Times New Roman", 10.0f);

headerText.Text = "Employees";

page.ReportTitle.Objects.Add(headerText);

// Employee Table Text bands

TextObject textx = new TextObject();

textx.Parent = data;

textx.CreateUniqueName();

textx.Bounds = new RectangleF(0 \* Units.Centimeters, 0, 1.0f \* Units.Centimeters, Units.Centimeters \* 0.5f);

textx.Font = new Font("Times New Roman", 5.0f);

textx.Text = "Num"; //table and its field

textx.Border.Lines = BorderLines.All;

TextObject textxx = new TextObject();

textxx.Parent = data;

textxx.CreateUniqueName();

textxx.Bounds = new RectangleF(1 \* Units.Centimeters, 0, Units.Centimeters \* 3, Units.Centimeters \* 0.5f);

textxx.Font = new Font("Times New Roman", 5.0f);

textxx.Text = "Surname"; //table and its field

textxx.Border.Lines = BorderLines.All;

TextObject textxxx = new TextObject();

textxxx.Parent = data;

textxxx.CreateUniqueName();

textxxx.Bounds = new RectangleF(4.0f \* Units.Centimeters, 0, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

textxxx.Font = new Font("Times New Roman", 5.0f);

textxxx.Text = "Name"; //table and its field

textxxx.Border.Lines = BorderLines.All;

TextObject textxxxx = new TextObject();

textxxxx.Parent = data;

textxxxx.CreateUniqueName();

textxxxx.Bounds = new RectangleF(7.4f \* Units.Centimeters, 0, Units.Centimeters \* 25, Units.Centimeters \* 0.5f);

textxxxx.Font = new Font("Times New Roman", 5.0f);

textxxxx.Text = "Patronym"; //table and its field

textxxxx.Border.Lines = BorderLines.All;

TextObject te = new TextObject();

te.Parent = data;

te.CreateUniqueName();

te.Bounds = new RectangleF(9.9f \* Units.Centimeters, 0, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

te.Font = new Font("Times New Roman", 5.0f);

te.Text = "Employed\_Since"; //table and its field

te.Border.Lines = BorderLines.All;

TextObject ter = new TextObject();

ter.Parent = data;

ter.CreateUniqueName();

ter.Bounds = new RectangleF(14 \* Units.Centimeters, 0, Units.Centimeters \* 30, Units.Centimeters \* 0.5f);

ter.Font = new Font("Times New Roman", 5.0f);

ter.Text = "Job"; //table and its field

ter.Border.Lines = BorderLines.All;

TextObject teXR = new TextObject();

teXR.Parent = data;

teXR.CreateUniqueName();

teXR.Bounds = new RectangleF(17.5f \* Units.Centimeters, 0, Units.Centimeters \* 25, Units.Centimeters \* 0.5f);

teXR.Font = new Font("Times New Roman", 5.0f);

teXR.Text = "Internship"; //table and its field

teXR.Border.Lines = BorderLines.All;

// Employee Table Fields

TextObject bandText = new TextObject();

bandText.CreateUniqueName();

bandText.HorzAlign = HorzAlign.Center;

bandText.Bounds = new RectangleF(0 \* Units.Centimeters, 25,

1.0f \* Units.Centimeters, 0.5f \* Units.Centimeters);

bandText.Font = new Font("Times New Roman", 5.0f);

bandText.Border.Lines = BorderLines.All;

bandText.Text = "[Employees.Num]";

data.AddChild(bandText);

TextObject text1 = new TextObject();

text1.Parent = data;

text1.CreateUniqueName();

text1.Bounds = new RectangleF(1 \* Units.Centimeters, 25, Units.Centimeters \* 3, Units.Centimeters \* 0.5f);

text1.Font = new Font("Times New Roman", 5.0f);

text1.Text = "[Employees.Surname]"; //table and its field

text1.Border.Lines = BorderLines.All;

TextObject text2 = new TextObject();

text2.Parent = data;

text2.CreateUniqueName();

text2.Bounds = new RectangleF(4.0f \* Units.Centimeters, 25, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

text2.Font = new Font("Times New Roman", 5.0f);

text2.Text = "[Employees.Name]"; //table and its field

text2.Border.Lines = BorderLines.All;

TextObject text3 = new TextObject();

text3.Parent = data;

text3.CreateUniqueName();

text3.Bounds = new RectangleF(7.4f \* Units.Centimeters, 25, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

text3.Font = new Font("Times New Roman", 5.0f);

text3.Text = "[Employees.Patronym]"; //table and its field

text3.Border.Lines = BorderLines.All;

TextObject text4 = new TextObject();

text4.Parent = data;

text4.CreateUniqueName();

text4.Bounds = new RectangleF(9.9f \* Units.Centimeters, 25, Units.Centimeters \* 30, Units.Centimeters \* 0.5f);

text4.Font = new Font("Times New Roman", 5.0f);

text4.Text = "[Employees.Employed\_Since]"; //table and its field

text4.Border.Lines = BorderLines.All;

TextObject text5 = new TextObject();

text5.Parent = data;

text5.CreateUniqueName();

text5.Bounds = new RectangleF(14 \* Units.Centimeters, 25, Units.Centimeters \* 5, Units.Centimeters \* 0.5f);

text5.Font = new Font("Times New Roman", 5.0f);

text5.Text = "[Employees.Job]"; //table and its field

text5.Border.Lines = BorderLines.All;

TextObject text6 = new TextObject();

text6.Parent = data;

text6.CreateUniqueName();

text6.Bounds = new RectangleF(17.5f \* Units.Centimeters, 25, Units.Centimeters \* 5, Units.Centimeters \* 0.5f);

text6.Font = new Font("Times New Roman", 5.0f);

text6.Text = "[Employees.Internship]"; //table and its field

text6.Border.Lines = BorderLines.All;

// page2

ReportPage page2 = new ReportPage();

report.Pages.Add(page2);

page2.CreateUniqueName();

page2.TopMargin = 10.0f;

page2.LeftMargin = 10.0f;

page2.RightMargin = 10.0f;

page2.BottomMargin = 10.0f;

page2.ReportTitle = new ReportTitleBand();

page2.ReportTitle.CreateUniqueName();

page2.ReportTitle.Height = 4.0f \* Units.Centimeters;

TextObject titleText = new TextObject();

titleText.CreateUniqueName();

titleText.Left = 1.0f \* Units.Centimeters;

titleText.Top = 1.0f \* Units.Centimeters;

titleText.Width = 17.0f \* Units.Centimeters;

titleText.Height = 2.0f \* Units.Centimeters;

titleText.HorzAlign = HorzAlign.Center;

titleText.VertAlign = VertAlign.Center;

titleText.Font = new Font("Times New Roman", 32.0f);

titleText.TextColor = Color.Black;

titleText.FillColor = Color.Wheat;

titleText.Border.Color = Color.Black;

titleText.Border.Lines = BorderLines.All;

titleText.Border.Width = 4.0f;

titleText.Text = "Avtopark Report";

page2.ReportTitle.Objects.Add(titleText);

// Maintenance table things\

DataBand datax = new DataBand();

datax.CreateUniqueName();

page2.AddChild(datax);

datax.DataSource = report.GetDataSource("Maintenance"); //adding a table to a bend

datax.Height = Units.Centimeters \* 5f;

//Maintenance Header object

TextObject headerSText = new TextObject();

headerSText.CreateUniqueName();

headerSText.Bounds = new RectangleF(0.0f, 3.2f \* Units.Centimeters,

19.0f \* Units.Centimeters, 0.8f \* Units.Centimeters);

headerSText.HorzAlign = HorzAlign.Left;

headerSText.VertAlign = VertAlign.Center;

headerSText.Font = new Font("Times New Roman", 10.0f);

headerSText.Border.Width = 4.0f;

headerSText.Text = "Maintenance";

page2.ReportTitle.Objects.Add(headerSText);

//Maintenance Table Text bands

TextObject textxxrx = new TextObject();

textxxrx.Parent = datax;

textxxrx.CreateUniqueName();

textxxrx.Bounds = new RectangleF(0 \* Units.Centimeters, 39.9f, 0.7f \* Units.Centimeters, Units.Centimeters \* 0.5f);

textxxrx.Font = new Font("Times New Roman", 6.0f);

textxxrx.Text = "Num"; //table and its field

textxxrx.Border.Lines = BorderLines.All;

TextObject textxxxr = new TextObject();

textxxxr.Parent = datax;

textxxxr.CreateUniqueName();

textxxxr.Font = new Font("Times New Roman", 6.0f);

textxxxr.Bounds = new RectangleF(0.7f \* Units.Centimeters, 39.9f, Units.Centimeters \* 7, Units.Centimeters \* 0.5f);

textxxxr.Text = "Model\_Avtobusa"; //table and its field

textxxxr.Border.Lines = BorderLines.All;

TextObject textxxxxr = new TextObject();

textxxxxr.Parent = datax;

textxxxxr.CreateUniqueName();

textxxxxr.Bounds = new RectangleF(4.5f \* Units.Centimeters, 39.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

textxxxxr.Font = new Font("Times New Roman", 5.0f);

textxxxxr.Text = "Data\_Poslednego\_Obsluzhivania"; //table and its field

textxxxxr.Border.Lines = BorderLines.All;

TextObject textxxxxx = new TextObject();

textxxxxx.Parent = datax;

textxxxxx.Font = new Font("Times New Roman", 5.0f);

textxxxxx.CreateUniqueName();

textxxxxx.Bounds = new RectangleF(7.7f \* Units.Centimeters, 39.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

textxxxxx.Text = "Ingener\_Obsluzhivania"; //table and its field

textxxxxx.Border.Lines = BorderLines.All;

TextObject terzx = new TextObject();

terzx.Parent = datax;

terzx.CreateUniqueName();

terzx.Bounds = new RectangleF(10.4f \* Units.Centimeters, 39.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terzx.Font = new Font("Times New Roman", 5.0f);

terzx.Text = "Problemi\_Avtobusa"; //table and its field

terzx.Border.Lines = BorderLines.All;

TextObject terxr = new TextObject();

terxr.Parent = datax;

terxr.CreateUniqueName();

terxr.Bounds = new RectangleF(15 \* Units.Centimeters, 39.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

terxr.Font = new Font("Times New Roman", 4.0f);

terxr.Text = "Data\_Sledueschego\_Obsluzivania"; //table and its field

terxr.Border.Lines = BorderLines.All;

TextObject teXRr = new TextObject();

teXRr.Parent = datax;

teXRr.CreateUniqueName();

teXRr.Bounds = new RectangleF(17.5f \* Units.Centimeters, 39.9f, Units.Centimeters \* 26, Units.Centimeters \* 0.5f);

teXRr.Font = new Font("Times New Roman", 6.0f);

teXRr.Text = "Goden\_K\_Doroge"; //table and its field

teXRr.Border.Lines = BorderLines.All;

//Maintenance table fields

TextObject textxxrxx = new TextObject();

textxxrxx.Parent = datax;

textxxrxx.CreateUniqueName();

textxxrxx.Bounds = new RectangleF(0 \* Units.Centimeters, 60.9f, 0.7f \* Units.Centimeters, Units.Centimeters \* 0.5f);

textxxrxx.Font = new Font("Times New Roman", 6.0f);

textxxrxx.Text = "[Maintenance.Num]"; //table and its field

textxxrxx.Border.Lines = BorderLines.All;

TextObject textxxxrx = new TextObject();

textxxxrx.Parent = datax;

textxxxrx.CreateUniqueName();

textxxxrx.Font = new Font("Times New Roman", 6.0f);

textxxxrx.Bounds = new RectangleF(0.7f \* Units.Centimeters, 60.9f, Units.Centimeters \* 7, Units.Centimeters \* 0.5f);

textxxxrx.Text = "[Maintenance.Model\_Avtobusa]"; //table and its field

textxxxrx.Border.Lines = BorderLines.All;

TextObject textxxxxrx = new TextObject();

textxxxxrx.Parent = datax;

textxxxxrx.CreateUniqueName();

textxxxxrx.Bounds = new RectangleF(4.5f \* Units.Centimeters, 60.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

textxxxxrx.Font = new Font("Times New Roman", 5.0f);

textxxxxrx.Text = "[Maintenance.Data\_Poslednego\_Obsluzhivania]"; //table and its field

textxxxxrx.Border.Lines = BorderLines.All;

TextObject textxxxxxx = new TextObject();

textxxxxxx.Parent = datax;

textxxxxxx.Font = new Font("Times New Roman", 5.0f);

textxxxxxx.CreateUniqueName();

textxxxxxx.Bounds = new RectangleF(7.7f \* Units.Centimeters, 60.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

textxxxxxx.Text = "[Maintenance.Ingener\_Obsluzhivania]"; //table and its field

textxxxxxx.Border.Lines = BorderLines.All;

TextObject terzxx = new TextObject();

terzxx.Parent = datax;

terzxx.CreateUniqueName();

terzxx.Bounds = new RectangleF(10.4f \* Units.Centimeters, 60.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terzxx.Font = new Font("Times New Roman", 5.0f);

terzxx.Text = "[Maintenance.Problemi\_Avtobusa]"; //table and its field

terzxx.Border.Lines = BorderLines.All;

TextObject terxrx = new TextObject();

terxrx.Parent = datax;

terxrx.CreateUniqueName();

terxrx.Bounds = new RectangleF(15 \* Units.Centimeters, 60.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

terxrx.Font = new Font("Times New Roman", 6.0f);

terxrx.Text = "[Maintenance.Data\_Sledueschego\_Obsluzivania]"; //table and its field

terxrx.Border.Lines = BorderLines.All;

TextObject teXRrr = new TextObject();

teXRrr.Parent = datax;

teXRrr.CreateUniqueName();

teXRrr.Bounds = new RectangleF(17.5f \* Units.Centimeters, 60.9f, Units.Centimeters \* 26, Units.Centimeters \* 0.5f);

teXRrr.Font = new Font("Times New Roman", 6.0f);

teXRrr.Text = "[Maintenance.Goden\_K\_Doroge]"; //table and its field

teXRrr.Border.Lines = BorderLines.All;

//page3

ReportPage page3 = new ReportPage();

report.Pages.Add(page3);

page3.CreateUniqueName();

page3.TopMargin = 10.0f;

page3.LeftMargin = 10.0f;

page3.RightMargin = 10.0f;

page3.BottomMargin = 10.0f;

page3.ReportTitle = new ReportTitleBand();

page3.ReportTitle.CreateUniqueName();

page3.ReportTitle.Height = 4.0f \* Units.Centimeters;

TextObject Title3 = new TextObject();

Title3.CreateUniqueName();

Title3.Left = 1.0f \* Units.Centimeters;

Title3.Top = 1.0f \* Units.Centimeters;

Title3.Width = 17.0f \* Units.Centimeters;

Title3.Height = 2.0f \* Units.Centimeters;

Title3.HorzAlign = HorzAlign.Center;

Title3.VertAlign = VertAlign.Center;

Title3.Font = new Font("Times New Roman", 32.0f);

Title3.TextColor = Color.Black;

Title3.FillColor = Color.Wheat;

Title3.Border.Color = Color.Black;

Title3.Border.Lines = BorderLines.All;

Title3.Border.Width = 4.0f;

Title3.Text = "Avtopark Report";

page3.ReportTitle.Objects.Add(Title3);

TextObject headersSText = new TextObject();

headersSText.CreateUniqueName();

headersSText.Bounds = new RectangleF(0.0f, 3.2f \* Units.Centimeters,

19.0f \* Units.Centimeters, 0.8f \* Units.Centimeters);

headersSText.HorzAlign = HorzAlign.Left;

headersSText.VertAlign = VertAlign.Center;

headersSText.Font = new Font("Times New Roman", 10.0f);

headersSText.Text = "Prodazhi";

page3.ReportTitle.Objects.Add(headersSText);

// Prodazhi Table Info

DataBand dataxr = new DataBand();

dataxr.CreateUniqueName();

page3.AddChild(dataxr);

dataxr.DataSource = report.GetDataSource("Prodazhi"); //adding a table to a bend

dataxr.Height = Units.Centimeters \* 5f;

TextObject texrxx = new TextObject();

texrxx.Parent = dataxr;

texrxx.CreateUniqueName();

texrxx.Bounds = new RectangleF(0 \* Units.Centimeters, 43.99f, 0.7f \* Units.Centimeters, Units.Centimeters \* 0.5f);

texrxx.Font = new Font("Times New Roman", 6.0f);

texrxx.Text = "Num"; //table and its field

texrxx.Border.Lines = BorderLines.All;

TextObject texrxs = new TextObject();

texrxs.Parent = dataxr;

texrxs.CreateUniqueName();

texrxs.Font = new Font("Times New Roman", 6.0f);

texrxs.Bounds = new RectangleF(0.7f \* Units.Centimeters, 43.99f, Units.Centimeters \* 7, Units.Centimeters \* 0.5f);

texrxs.Text = "Sale\_Date"; //table and its field

texrxs.Border.Lines = BorderLines.All;

TextObject terscw = new TextObject();

terscw.Parent = dataxr;

terscw.CreateUniqueName();

terscw.Bounds = new RectangleF(4.5f \* Units.Centimeters, 43.99f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

terscw.Font = new Font("Times New Roman", 5.0f);

terscw.Text = "Nomer\_Avtobusa"; //table and its field

terscw.Border.Lines = BorderLines.All;

TextObject terscwx = new TextObject();

terscwx.Parent = dataxr;

terscwx.Font = new Font("Times New Roman", 5.0f);

terscwx.CreateUniqueName();

terscwx.Bounds = new RectangleF(7.7f \* Units.Centimeters, 43.99f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terscwx.Text = "Punkt\_Posadki"; //table and its field

terscwx.Border.Lines = BorderLines.All;

TextObject terzrx = new TextObject();

terzrx.Parent = dataxr;

terzrx.CreateUniqueName();

terzrx.Bounds = new RectangleF(10.4f \* Units.Centimeters, 43.99f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terzrx.Font = new Font("Times New Roman", 5.0f);

terzrx.Text = "Stoimost"; //table and its field

terzrx.Border.Lines = BorderLines.All;

// Table Fields For Prodazhi

TextObject texrxxx = new TextObject();

texrxxx.Parent = dataxr;

texrxxx.CreateUniqueName();

texrxxx.Bounds = new RectangleF(0 \* Units.Centimeters, 53.99f, 0.7f \* Units.Centimeters, Units.Centimeters \* 0.5f);

texrxxx.Font = new Font("Times New Roman", 6.0f);

texrxxx.Text = "[Prodazhi.Num]"; //table and its field

texrxxx.Border.Lines = BorderLines.All;

TextObject texrxss = new TextObject();

texrxss.Parent = dataxr;

texrxss.CreateUniqueName();

texrxss.Font = new Font("Times New Roman", 6.0f);

texrxss.Bounds = new RectangleF(0.7f \* Units.Centimeters, 53.99f, Units.Centimeters \* 7, Units.Centimeters \* 0.5f);

texrxss.Text = "[Prodazhi.Sale\_Date]"; //table and its field

texrxss.Border.Lines = BorderLines.All;

TextObject terscwxxxr = new TextObject();

terscwxxxr.Parent = dataxr;

terscwxxxr.CreateUniqueName();

terscwxxxr.Bounds = new RectangleF(4.5f \* Units.Centimeters, 53.99f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

terscwxxxr.Font = new Font("Times New Roman", 5.0f);

terscwxxxr.Text = "[Prodazhi.Nomer\_Avtobusa]"; //table and its field

terscwxxxr.Border.Lines = BorderLines.All;

TextObject terscwxxr = new TextObject();

terscwxxr.Parent = dataxr;

terscwxxr.Font = new Font("Times New Roman", 5.0f);

terscwxxr.CreateUniqueName();

terscwxxr.Bounds = new RectangleF(7.7f \* Units.Centimeters, 53.99f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terscwxxr.Text = "[Prodazhi.Punkt\_Posadki]"; //table and its field

terscwxxr.Border.Lines = BorderLines.All;

TextObject terzrxr = new TextObject();

terzrxr.Parent = dataxr;

terzrxr.CreateUniqueName();

terzrxr.Bounds = new RectangleF(10.4f \* Units.Centimeters, 53.99f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terzrxr.Font = new Font("Times New Roman", 5.0f);

terzrxr.Text = "[Prodazhi.Stoimost]"; //table and its field

terzrxr.Border.Lines = BorderLines.All;

//page4

ReportPage page4 = new ReportPage();

report.Pages.Add(page4);

page4.CreateUniqueName();

page4.TopMargin = 10.0f;

page4.LeftMargin = 10.0f;

page4.RightMargin = 10.0f;

page4.BottomMargin = 10.0f;

page4.ReportTitle = new ReportTitleBand();

page4.ReportTitle.CreateUniqueName();

page4.ReportTitle.Height = 4.0f \* Units.Centimeters;

TextObject title4 = new TextObject();

title4.CreateUniqueName();

title4.Left = 1.0f \* Units.Centimeters;

title4.Top = 1.0f \* Units.Centimeters;

title4.Width = 17.0f \* Units.Centimeters;

title4.Height = 2.0f \* Units.Centimeters;

title4.HorzAlign = HorzAlign.Center;

title4.VertAlign = VertAlign.Center;

title4.Font = new Font("Times New Roman", 32.0f);

title4.TextColor = Color.Black;

title4.FillColor = Color.Wheat;

title4.Border.Color = Color.Black;

title4.Border.Lines = BorderLines.All;

title4.Border.Width = 4.0f;

title4.Text = "Avtopark Report";

page4.ReportTitle.Objects.Add(title4);

TextObject headtextx = new TextObject();

headtextx.CreateUniqueName();

headtextx.Bounds = new RectangleF(0.0f, 3.2f \* Units.Centimeters,

19.0f \* Units.Centimeters, 0.8f \* Units.Centimeters);

headtextx.HorzAlign = HorzAlign.Left;

headtextx.VertAlign = VertAlign.Center;

headtextx.Font = new Font("Times New Roman", 10.0f);

headtextx.Text = "Marshuti";

page4.ReportTitle.Objects.Add(headtextx);

// Marshuti fields

DataBand datarxr = new DataBand();

datarxr.CreateUniqueName();

page4.AddChild(datarxr);

datarxr.DataSource = report.GetDataSource("Marshuti"); //adding a table to a bend

datarxr.Height = Units.Centimeters \* 2f;

TextObject txtef = new TextObject();

txtef.Parent = datarxr;

txtef.CreateUniqueName();

txtef.Bounds = new RectangleF(0 \* Units.Centimeters, 58.9f, 0.7f \* Units.Centimeters, Units.Centimeters \* 0.5f);

txtef.Font = new Font("Times New Roman", 6.0f);

txtef.Text = "Nomer\_Marshuta"; //table and its field

txtef.Border.Lines = BorderLines.All;

TextObject tfeqa = new TextObject();

tfeqa.Parent = datarxr;

tfeqa.CreateUniqueName();

tfeqa.Font = new Font("Times New Roman", 6.0f);

tfeqa.Bounds = new RectangleF(0.7f \* Units.Centimeters, 58.9f, Units.Centimeters \* 7, Units.Centimeters \* 0.5f);

tfeqa.Text = "Nachalni\_Punkt"; //table and its field

tfeqa.Border.Lines = BorderLines.All;

TextObject tcqqcq = new TextObject();

tcqqcq.Parent = datarxr;

tcqqcq.CreateUniqueName();

tcqqcq.Bounds = new RectangleF(4.5f \* Units.Centimeters, 58.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

tcqqcq.Font = new Font("Times New Roman", 5.0f);

tcqqcq.Text = "Konechni\_Punkt"; //table and its field

tcqqcq.Border.Lines = BorderLines.All;

TextObject txtweq = new TextObject();

txtweq.Parent = datarxr;

txtweq.Font = new Font("Times New Roman", 5.0f);

txtweq.CreateUniqueName();

txtweq.Bounds = new RectangleF(7.7f \* Units.Centimeters, 58.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

txtweq.Text = "Voditel"; //table and its field

txtweq.Border.Lines = BorderLines.All;

TextObject terrzx = new TextObject();

terrzx.Parent = datarxr;

terrzx.CreateUniqueName();

terrzx.Bounds = new RectangleF(10.4f \* Units.Centimeters, 58.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terrzx.Font = new Font("Times New Roman", 5.0f);

terrzx.Text = "Model\_Avtobusa"; //table and its field

terrzx.Border.Lines = BorderLines.All;

TextObject txerxr = new TextObject();

txerxr.Parent = datarxr;

txerxr.CreateUniqueName();

txerxr.Bounds = new RectangleF(15 \* Units.Centimeters, 58.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

txerxr.Font = new Font("Times New Roman", 4.0f);

txerxr.Text = "Vremya\_Proezda"; //table and its field

txerxr.Border.Lines = BorderLines.All;

TextObject txteff = new TextObject();

txteff.Parent = datarxr;

txteff.CreateUniqueName();

txteff.Bounds = new RectangleF(0 \* Units.Centimeters, 82.9f, 0.7f \* Units.Centimeters, Units.Centimeters \* 0.5f);

txteff.Font = new Font("Times New Roman", 6.0f);

txteff.Text = "[Marshuti.Nomer\_Marshuta]"; //table and its field

txteff.Border.Lines = BorderLines.All;

TextObject tfeqafd = new TextObject();

tfeqafd.Parent = datarxr;

tfeqafd.CreateUniqueName();

tfeqafd.Font = new Font("Times New Roman", 6.0f);

tfeqafd.Bounds = new RectangleF(0.7f \* Units.Centimeters, 82.9f, Units.Centimeters \* 7, Units.Centimeters \* 0.5f);

tfeqafd.Text = "[Marshuti.Nachalni\_Punkt]"; //table and its field

tfeqafd.Border.Lines = BorderLines.All;

TextObject tcqqcqd = new TextObject();

tcqqcqd.Parent = datarxr;

tcqqcqd.CreateUniqueName();

tcqqcqd.Bounds = new RectangleF(4.5f \* Units.Centimeters, 82.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

tcqqcqd.Font = new Font("Times New Roman", 5.0f);

tcqqcqd.Text = "[Marshuti.Konechni\_Punkt]"; //table and its field

tcqqcqd.Border.Lines = BorderLines.All;

TextObject txtweqq = new TextObject();

txtweqq.Parent = datarxr;

txtweqq.Font = new Font("Times New Roman", 5.0f);

txtweqq.CreateUniqueName();

txtweqq.Bounds = new RectangleF(7.7f \* Units.Centimeters, 82.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

txtweqq.Text = "[Marshuti.Voditel]"; //table and its field

txtweqq.Border.Lines = BorderLines.All;

TextObject terrzxr = new TextObject();

terrzxr.Parent = datarxr;

terrzxr.CreateUniqueName();

terrzxr.Bounds = new RectangleF(10.4f \* Units.Centimeters, 82.9f, Units.Centimeters \* 20, Units.Centimeters \* 0.5f);

terrzxr.Font = new Font("Times New Roman", 5.0f);

terrzxr.Text = "[Marshuti.Model\_Avtobusa]"; //table and its field

terrzxr.Border.Lines = BorderLines.All;

TextObject txerxxr = new TextObject();

txerxxr.Parent = datarxr;

txerxxr.CreateUniqueName();

txerxxr.Bounds = new RectangleF(15 \* Units.Centimeters, 82.9f, Units.Centimeters \* 15, Units.Centimeters \* 0.5f);

txerxxr.Font = new Font("Times New Roman", 4.0f);

txerxxr.Text = "[Marshuti.Vremya\_Proezda]"; //table and its field

txerxxr.Border.Lines = BorderLines.All;

// Footer

page.PageFooter = new PageFooterBand();

page.PageFooter.CreateUniqueName();

page.PageFooter.Height = 0.5f \* Units.Centimeters;

page2.PageFooter = new PageFooterBand();

page2.PageFooter.CreateUniqueName();

page2.PageFooter.Height = 0.5f \* Units.Centimeters;

page3.PageFooter = new PageFooterBand();

page3.PageFooter.CreateUniqueName();

page3.PageFooter.Height = 0.5f \* Units.Centimeters;

page4.PageFooter = new PageFooterBand();

page4.PageFooter.CreateUniqueName();

page4.PageFooter.Height = 0.5f \* Units.Centimeters;

TextObject footerText = new TextObject();

footerText.CreateUniqueName();

footerText.HorzAlign = HorzAlign.Right;

footerText.VertAlign = VertAlign.Center;

footerText.Bounds = new RectangleF(0.0f, 0.0f,

19.0f \* Units.Centimeters, 0.5f \* Units.Centimeters);

footerText.TextColor = Color.Black;

footerText.FillColor = Color.Wheat;

titleText.Border.Color = Color.Black;

titleText.Border.Lines = BorderLines.All;

titleText.Border.Width = 4.0f;

footerText.Text = "Page [Page]";

TextObject f2 = new TextObject();

f2.CreateUniqueName();

f2.HorzAlign = HorzAlign.Right;

f2.VertAlign = VertAlign.Center;

f2.Bounds = new RectangleF(0.0f, 0.0f,

19.0f \* Units.Centimeters, 0.5f \* Units.Centimeters);

f2.TextColor = Color.Black;

f2.FillColor = Color.Wheat;

titleText.Border.Color = Color.Black;

titleText.Border.Lines = BorderLines.All;

titleText.Border.Width = 4.0f;

f2.Text = "Page [Page]";

TextObject f3x = new TextObject();

f3x.CreateUniqueName();

f3x.HorzAlign = HorzAlign.Right;

f3x.VertAlign = VertAlign.Center;

f3x.Bounds = new RectangleF(0.0f, 0.0f,

19.0f \* Units.Centimeters, 0.5f \* Units.Centimeters);

f3x.TextColor = Color.Black;

f3x.FillColor = Color.Wheat;

titleText.Border.Color = Color.Black;

titleText.Border.Lines = BorderLines.All;

titleText.Border.Width = 4.0f;

f3x.Text = "Page [Page]";

TextObject f4x = new TextObject();

f4x.CreateUniqueName();

f4x.HorzAlign = HorzAlign.Right;

f4x.VertAlign = VertAlign.Center;

f4x.Bounds = new RectangleF(0.0f, 0.0f,

19.0f \* Units.Centimeters, 0.5f \* Units.Centimeters);

f4x.TextColor = Color.Black;

f4x.FillColor = Color.Wheat;

titleText.Border.Color = Color.Black;

titleText.Border.Lines = BorderLines.All;

titleText.Border.Width = 4.0f;

f4x.Text = "Page [Page]";

page.PageFooter.Objects.Add(footerText);

page2.PageFooter.Objects.Add(f2);

page3.PageFooter.Objects.Add(f3x);

page4.PageFooter.Objects.Add(f4x);

try

{

// Exporting the report

report.Prepare();

report.SavePrepared("PreparedComplexReport.fpx");

PDFSimpleExport pdf = new PDFSimpleExport();

// Save the report

report.Export(pdf, "ExportedComplexPDF.pdf");

}

catch (Exception ex)

{

MsgBoxExtendedFunctionality ext = new MsgBoxExtendedFunctionality()

{

DetailsText = ex.StackTrace

};

MessageBoxEx.ShowEx("An error has occured! Traceback: " + ex.Message, "Unexpected situation handling", MessageBoxButtonEx.OK, MessageBoxImage.Error, ext);

}

}

}

}

using System.Data;  
using System.Windows;  
using System;  
using System.Data.SqlClient;  
using System.Windows.Input;  
using Wpf.Ui.Controls;  
using MessageBox = Wpf.Ui.Controls.MessageBox;  
using TextBlock = System.Windows.Controls.TextBlock;  
  
namespace FluentKursovayaAvtoparkA.Views.Pages  
{  
 /// <summary>  
 /// Interaction logic for DashboardPage.xaml  
 /// </summary>  
 public partial class DashboardPage : INavigableView<ViewModels.DashboardViewModel>  
 {  
 public ViewModels.DashboardViewModel ViewModel  
 {  
 get;  
 }  
  
 public DashboardPage(ViewModels.DashboardViewModel viewModel)  
 {  
 ViewModel = viewModel;  
  
 InitializeComponent();  
 }  
 private void UpdateGrid\_Click(object sender, RoutedEventArgs e)  
 {  
  
 try  
 {  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 Console.Write("OPENING DB CONNECTION!!! \n");  
 Console.Write("[MENU] Connect db clicked - test \n");  
 var select = "SELECT \* FROM Employees";  
   
 var select2 = "SELECT \* FROM Jobs";  
 var commandBuilder = new SqlCommand(select, cnn);  
 commandBuilder.ExecuteNonQuery();  
 var commandBuilder2 = new SqlCommand(select2, cnn);  
 commandBuilder2.ExecuteNonQuery();  
 var dataAdapter = new SqlDataAdapter(commandBuilder);  
 var dataAdapter2 = new SqlDataAdapter(commandBuilder2);  
 var ds = new DataTable("Avtobusx");  
 dataAdapter.Fill(ds);  
 var ds2 = new DataTable("Avtobusxi");  
 dataAdapter2.Fill(ds2);  
 dataGrid1.ItemsSource = ds.DefaultView;  
 dataGrid2.ItemsSource = ds2.DefaultView;  
 Console.WriteLine("Connection established and the datagrid filled!");  
 cnn.Close();  
 }  
 catch (Exception ex)  
 {  
 var uiMessageBox = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
   
 Width = 800,  
 Height = 300,  
  
 };  
   
  
  
 uiMessageBox.ShowDialogAsync();  
 }  
 }  
  
 private void Button\_Click(object sender, RoutedEventArgs e)  
 {  
 try  
 {  
 var Num = Exed1.Text;  
 var EmpName = Exed2.Text;  
 var EmpsurName = Exed3.Text;  
 var Emppat = Exed4.Text;  
 var EmpES = Exed5.Text;  
 var Empnum = int.Parse(Exed6.Text);  
 var Empinternyesorno = Exed7.Text;  
 var Empjobtitle = Exed8.Text;  
 var query = "INSERT INTO Employees(Num, Surname, Name, Patronym,Employed\_Since,Job\_Num) " +  
 "Values('" + Num + "', '" + EmpName + "', '" + EmpsurName + "', '" + Emppat + "','" + EmpES +  
 "','" + Empnum + "')";  
 var query2 = "INSERT INTO Jobs(Job\_Num, Job, Internship) " +  
 "Values('" + Empnum + "', '" + Empjobtitle + "', '" + Empinternyesorno + "')";  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 var command2 = new SqlCommand(query2, cnn);  
 command2.ExecuteNonQuery();  
 var command = new SqlCommand(query, cnn);  
 command.ExecuteNonQuery();  
   
 Console.WriteLine("Data inserted!\n Closing connection ");  
 cnn.Close();  
 Console.WriteLine("Connection has been closed , database ready for next operation!");  
 UpdateGrid\_Click(sender, e);  
 }  
 catch (Exception ex)  
 {  
 var uiMessageBox = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
   
 Width = 800,  
 Height = 300,  
  
 };  
   
  
  
 uiMessageBox.ShowDialogAsync();  
 }  
 }  
  
 private void EmpUpdat\_Click(object sender, RoutedEventArgs e)  
 {  
  
   
   
   
 try  
 {  
 var Num = Exed1.Text;  
 var EmpName = Exed2.Text;  
 var EmpsurName = Exed3.Text;  
 var Emppat = Exed4.Text;  
 var EmpES = Exed5.Text;  
 var Empnum = int.Parse(Exed6.Text);  
 var Empinternyesorno = Exed7.Text;  
 var Empjobtitle = Exed8.Text;  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 Console.Write("OPENING DB CONNECTION!!! \n");  
 var select =  
 "UPDATE Employees SET Surname=@a1,Name=@a2," +  
 "Patronym=@a3,Employed\_Since=@a4," +  
 "Job\_Num=@a5 WHERE Num=@a7";  
 var querier2 =  
 "UPDATE Jobs SET Job=@a4,Internship=@a3 WHERE Job\_Num=@a6";  
 var commandBuilder = new SqlCommand(select, cnn);  
 var commandBuilder2 = new SqlCommand(querier2, cnn);  
 commandBuilder.Parameters.AddWithValue("a1",EmpsurName);  
 commandBuilder.Parameters.AddWithValue("a2", EmpName);  
 commandBuilder.Parameters.AddWithValue("a3", Emppat);  
 commandBuilder.Parameters.AddWithValue("a4", EmpES);  
 commandBuilder.Parameters.AddWithValue("a5", Empnum);  
 commandBuilder2.Parameters.AddWithValue("a6", Empnum);  
 commandBuilder2.Parameters.AddWithValue("a4", Empjobtitle);  
 commandBuilder2.Parameters.AddWithValue("a3", Empinternyesorno);  
 commandBuilder.Parameters.AddWithValue("a7", int.Parse(Num));  
 commandBuilder.ExecuteNonQuery();  
 commandBuilder2.ExecuteNonQuery();  
 Console.WriteLine("Connection established and the table updated!");  
 cnn.Close();  
 UpdateGrid\_Click(sender, e);  
 }  
  
 catch (Exception ex)  
 {  
 var uiMessageBox = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
   
 Width = 800,  
 Height = 300,  
  
 };  
   
  
  
 uiMessageBox.ShowDialogAsync();  
 }  
 }  
  
 private void Exed1\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Exed1.Clear();  
 }  
  
 private void Exed4\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Exed4.Clear();  
 }  
  
 private void Exed7\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Exed7.Clear();  
 }  
  
 private void Exed2\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Exed2.Clear();  
 }  
  
 private void Exed5\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Exed5.Clear();  
 }  
  
 private void Exed3\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Exed3.Clear();  
 }  
  
 private void Exed6\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Exed6.Clear();  
 }  
  
 private void Exed8\_MouseDown(object sender, MouseButtonEventArgs e)  
 {  
 Exed8.Clear();  
 }  
 }  
}

using MsgBoxEx;  
using System.Data.SqlClient;  
using System.Windows;  
using System;  
using System.Data;  
using Wpf.Ui.Controls;  
using MessageBox = Wpf.Ui.Controls.MessageBox;  
using System.Windows.Controls;  
  
namespace FluentKursovayaAvtoparkA.Views.Pages  
{  
 /// <summary>  
 /// Interaction logic for DataView.xaml  
 /// </summary>  
 public partial class DataPage : INavigableView<ViewModels.DataViewModel>  
 {  
 public ViewModels.DataViewModel ViewModel  
 {  
 get;  
 }  
  
 public DataPage(ViewModels.DataViewModel viewModel)  
 {  
 ViewModel = viewModel;  
  
 InitializeComponent();  
 }  
 private void UpdateTransportTable\_Click(object sender, RoutedEventArgs e)  
 {  
 try  
 {  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 Console.Write("OPENING DB CONNECTION!!! \n");  
 var select =  
 "UPDATE Marshuti SET Nachalni\_Punkt=@a1,Konechni\_Punkt=@a2,Emp\_Num=@a3,ModelAvtobusaKey=@a4,Vremya\_Proezda=@a5 WHERE Nomer\_Marshuta=@a6";  
 var commandBuilder = new SqlCommand(select, cnn);  
 commandBuilder.Parameters.AddWithValue("a1", Edit2.Text);  
 commandBuilder.Parameters.AddWithValue("a2", Edit3.Text);  
 commandBuilder.Parameters.AddWithValue("a3", int.Parse(Edit4.Text));  
 commandBuilder.Parameters.AddWithValue("a4", int.Parse(Edit5.Text));  
 commandBuilder.Parameters.AddWithValue("a5", Edit6.Text);  
 commandBuilder.Parameters.AddWithValue("a6", int.Parse(Edit1.Text));  
 commandBuilder.ExecuteNonQuery();  
 Console.WriteLine("Connection established and the table updated!");  
 cnn.Close();  
   
 }  
 catch (Exception ex)  
 {  
 MsgBoxExtendedFunctionality ext = new MsgBoxExtendedFunctionality()  
 {  
 DetailsText = ex.StackTrace  
 };  
 MessageBoxEx.ShowEx("An error has occured! Traceback: " + ex.Message, "Unexpected situation handling", MessageBoxButtonEx.OK, MessageBoxImage.Error, ext);  
 }  
 }  
 private void Button\_Click(object sender, RoutedEventArgs e)  
 {  
 try  
 {  
 var NomeMarshuta = Edit1.Text;  
 var NachalnPunkt = Edit2.Text;  
 var KonechnPunkt = Edit3.Text;  
 var Vodit = int.Parse(Edit4.Text);  
 var Modelvtobusa = int.Parse(Edit5.Text);  
 var VremProezda = Edit6.Text;  
 var query =  
 "INSERT INTO Marshuti(Nomer\_Marshuta, Nachalni\_Punkt, Konechni\_Punkt, Emp\_Num,ModelAvtobusaKey,Vremya\_Proezda) " +  
 "Values('" + NomeMarshuta + "', '" + NachalnPunkt + "', '" + KonechnPunkt + "', '" + Vodit + "','" +  
 Modelvtobusa + "', '" + VremProezda + "')";  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 var command = new SqlCommand(query, cnn);  
 command.ExecuteNonQuery();  
 Console.WriteLine("Data inserted!\n Closing connection ");  
 cnn.Close();  
 Console.WriteLine("Connection has been closed , database ready for next operation!");  
 }  
 catch (Exception ex)  
 {  
 MsgBoxExtendedFunctionality ext = new MsgBoxExtendedFunctionality()  
 {  
 DetailsText = ex.StackTrace  
 };  
 MessageBoxEx.ShowEx("An error has occured! Traceback: " + ex.Message, "Unexpected situation handling", MessageBoxButtonEx.OK, MessageBoxImage.Error, ext);  
 }  
 }  
 private void MessageBox\_LeftButtonClick() {   
   
 }  
 private void UpdateGrid\_Click(object sender, RoutedEventArgs e)  
 {  
  
 try  
 {  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 Console.Write("OPENING DB CONNECTION!!! \n");  
 Console.Write("[MENU] Connect db clicked - test \n");  
 var select = "SELECT \* FROM Marshuti";  
  
 var commandBuilder = new SqlCommand(select, cnn);  
 commandBuilder.ExecuteNonQuery();  
  
 var dataAdapter = new SqlDataAdapter(commandBuilder);  
 var ds = new DataTable("Avtobusx");  
 dataAdapter.Fill(ds);  
 dataGrid1.ItemsSource = ds.DefaultView;  
 Console.WriteLine("Connection established and the datagrid filled!");  
 cnn.Close();  
 }  
 catch (Exception ex)  
 {  
 var uiMessageBox = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new Wpf.Ui.Controls.TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
   
 Width = 800,  
 Height= 500,  
   
 };  
   
   
  
 uiMessageBox.ShowDialogAsync();  
 }  
 }  
 }  
}

using System;  
using System.Data.SqlClient;  
using System.Data;  
using System.Windows;  
using Wpf.Ui.Controls;  
using MessageBox = Wpf.Ui.Controls.MessageBox;  
using TextBlock = System.Windows.Controls.TextBlock;  
  
namespace FluentKursovayaAvtoparkA.Views.Pages  
{  
 /// <summary>  
 /// Логика взаимодействия для MaintenancePage.xaml  
 /// </summary>  
 public partial class MaintenancePage : INavigableView<ViewModels.MaintenancePageViewModel>  
 {  
   
 public ViewModels.MaintenancePageViewModel ViewModel  
 {  
 get;  
 }  
  
 public MaintenancePage(ViewModels.MaintenancePageViewModel viewModel)  
 {  
 ViewModel = viewModel;  
  
 InitializeComponent();  
 }  
 private void Button\_Click(object sender, RoutedEventArgs e)  
 {  
 try  
 {  
 var Num = int.Parse(AvEditor1.Text);  
 var ModAvtobus = int.Parse(AvEditor2.Text);  
 var DatObslug = AvEditor3.Text;  
 var Engineer = AvEditor4.Text;  
 var ErAvto = AvEditor5.Text;  
 var DatSlegObslug = AvEditor6.Text;  
 var GodDorog = AvEditor7.Text;  
 var ModelAvtobusaText = AvEditor8.Text;  
 var query =  
 "INSERT INTO Obsluzhivanie(NomerObsluzhivania, Model\_AvtobusaKey, Data\_Poslednego\_Obsluzhivania, Ingener\_Obsluzhivania,Problemi\_Avtobusa,Data\_Sledueschego\_Obsluzivania,Goden\_K\_Doroge) " +  
 "Values('" + Num + "', '" + ModAvtobus + "', '" + DatObslug + "', '" + Engineer + "','" + ErAvto +  
 "','" + DatSlegObslug + "','" + GodDorog + "')";  
 var query2 = "INSERT INTO Avtobusy(Numavtobusy,Model\_Avtobusa) " +  
 "Values('" + ModAvtobus + "', '" + ModelAvtobusaText + "')";  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 var command2 = new SqlCommand(query2, cnn);  
 command2.ExecuteNonQuery();  
 var command = new SqlCommand(query, cnn);  
 command.ExecuteNonQuery();  
 Console.WriteLine("Data inserted!\n Closing connection ");  
 cnn.Close();  
 Console.WriteLine("Connection has been closed , database ready for next operation!");  
 UpdateGrid\_Click(sender, e);  
 }  
 catch (Exception ex)  
 {  
 var uiMessageBox = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
   
 Width = 800,  
 Height = 300,  
  
 };  
   
  
  
 uiMessageBox.ShowDialogAsync();  
 }  
 }  
  
 private void UpdateGrid\_Click(object sender, RoutedEventArgs e)  
 {  
 try  
 {  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 Console.Write("OPENING DB CONNECTION!!! \n");  
 Console.Write("[MENU] Connect db clicked - test \n");  
 var select = "SELECT \* FROM Obsluzhivanie";  
  
 var select2 = "SELECT \* FROM Avtobusy";  
 var commandBuilder = new SqlCommand(select, cnn);  
 commandBuilder.ExecuteNonQuery();  
 var commandBuilder2 = new SqlCommand(select2, cnn);  
 commandBuilder2.ExecuteNonQuery();  
 var dataAdapter = new SqlDataAdapter(commandBuilder);  
 var dataAdapter2 = new SqlDataAdapter(commandBuilder2);  
 var ds = new DataTable("Avtobusx");  
 dataAdapter.Fill(ds);  
 var ds2 = new DataTable("Avtobusxi");  
 dataAdapter2.Fill(ds2);  
 dataGrid1.ItemsSource = ds.DefaultView;  
 dataGrid2.ItemsSource = ds2.DefaultView;  
 Console.WriteLine("Connection established and the datagrid filled!");  
 cnn.Close();  
 }  
 catch (Exception ex)  
 {  
 var uiMessageBox = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
   
 Width = 800,  
 Height = 300,  
  
 };  
   
  
 uiMessageBox.ShowDialogAsync();  
 }  
 }  
  
 private void UpdateMaintenaceTable\_Click(object sender, RoutedEventArgs e)  
 {  
   
 try  
 {  
 var Num = int.Parse(AvEditor1.Text);  
 var ModAvtobus = int.Parse(AvEditor2.Text);  
 var DatObslug = AvEditor3.Text;  
 var Engineer = AvEditor4.Text;  
 var ErAvto = AvEditor5.Text;  
 var DatSlegObslug = AvEditor6.Text;  
 var GodDorog = AvEditor7.Text;  
 var ModelAvtobusaText = AvEditor8.Text;  
 SqlConnection cnn;  
 var conStrs = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStrs);  
 cnn.Open();  
 Console.Write("OPENING DB CONNECTION!!! \n");  
 var querier =  
 "UPDATE Obsluzhivanie SET Model\_AvtobusaKey=@a1,Data\_Poslednego\_Obsluzhivania=@a2," +  
 "Ingener\_Obsluzhivania=@a3,Problemi\_Avtobusa=@a4," +  
 "Data\_Sledueschego\_Obsluzivania=@a5,Goden\_K\_Doroge=@a6" +  
 " WHERE NomerObsluzhivania=@a7";  
 var querier2 =  
 "UPDATE Avtobusy SET Model\_Avtobusa=@a4 WHERE Numavtobusy=@a6";  
 var commandBuilder = new SqlCommand(querier, cnn);  
 var commandBuilder2 = new SqlCommand(querier2, cnn);  
 commandBuilder.Parameters.AddWithValue("a1", ModAvtobus);  
 commandBuilder.Parameters.AddWithValue("a2", DatObslug);  
 commandBuilder.Parameters.AddWithValue("a3", Engineer);  
 commandBuilder.Parameters.AddWithValue("a4", ErAvto);  
 commandBuilder2.Parameters.AddWithValue("a4", ModelAvtobusaText);  
 commandBuilder.Parameters.AddWithValue("a5", DatSlegObslug);  
 commandBuilder.Parameters.AddWithValue("a6", GodDorog);  
 commandBuilder.Parameters.AddWithValue("a7", Num);  
 commandBuilder2.Parameters.AddWithValue("a6", ModAvtobus);  
 commandBuilder.ExecuteNonQuery();  
 commandBuilder2.ExecuteNonQuery();  
 Console.WriteLine("Connection established and the table updated!");  
 cnn.Close();  
 UpdateGrid\_Click(sender, e);  
 }  
  
 catch (Exception ex)  
 {  
 var uiMessageBox = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
   
 Width = 800,  
 Height = 300,  
  
 };  
   
  
  
 uiMessageBox.ShowDialogAsync();  
 }  
 }  
  
 private void AvEditor1\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 AvEditor1.Clear();  
 }  
  
 private void AvEditor2\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 AvEditor2.Clear();  
 }  
  
 private void AvEditor3\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 AvEditor3.Clear();  
 }  
  
 private void AvEditor4\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 AvEditor4.Clear();  
 }  
  
 private void AvEditor5\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 AvEditor5.Clear();  
 }  
  
 private void AvEditor6\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 AvEditor6.Clear();  
 }  
  
 private void AvEditor7\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 AvEditor7.Clear();  
 }  
 }  
}

using System;  
using System.Data.SqlClient;  
using System.Linq;  
using System.Threading.Tasks;  
using SQLServerLoginTemplate;  
using System.Windows;  
using System.Windows.Controls;  
using CommunityToolkit.Mvvm.ComponentModel;  
using CommunityToolkit.Mvvm.Input;  
using Wpf.Ui.Controls;  
using MessageBox = Wpf.Ui.Controls.MessageBox;  
using TextBlock = System.Windows.Controls.TextBlock;  
using Wpf.Ui.Controls;  
using Wpf.Ui;  
using Wpf.Ui.Extensions;  
using FluentKursovayaAvtoparkA.Controls;  
using FluentKursovayaAvtoparkA.ViewModels.Windows;  
using FluentKursovayaAvtoparkA.Views.Windows;  
using Wpf.Ui.Appearance;  
  
namespace FluentKursovayaAvtoparkA.Views.Pages  
{  
 /// <summary>  
 /// Interaction logic for SettingsPage.xaml  
 /// </summary>  
 ///  
 public partial class ContentDialogViewModel(IContentDialogService contentDialogService) : ObservableObject  
 {  
 [ObservableProperty] private string \_dialogResultText = String.Empty;  
  
 [RelayCommand]  
 private async Task OnShowDialog(object content)  
 {  
 ContentDialogResult result = await contentDialogService.ShowSimpleDialogAsync(  
 new SimpleContentDialogCreateOptions()  
 {  
 Title = "Save your work?",  
 Content = content,  
 PrimaryButtonText = "Save",  
 SecondaryButtonText = "Don't Save",  
 CloseButtonText = "Cancel",  
 }  
 );  
  
 DialogResultText = result switch  
 {  
 ContentDialogResult.Primary => "User saved their work",  
 ContentDialogResult.Secondary => "User did not save their work",  
 \_ => "User cancelled the dialog"  
 };  
 }  
  
 [RelayCommand]  
 public async Task OnShowSignInContentDialog(MainWindowViewModel ViewModel)  
 {  
  
 }  
 }  
  
 public partial class SettingsPage : INavigableView<ViewModels.SettingsViewModel>  
 {  
 public static string a;  
  
 public static string formations =  
 "server=localhost;Initial Catalog=KursovayaAvtoparkAvtobusov;User ID=sa;Password=ctrt55xx;";  
  
 public static string exporttheme = "theme\_dark";  
 public ViewModels.SettingsViewModel ViewModel { get; }  
  
 public SettingsPage(ViewModels.SettingsViewModel viewModel)  
 {  
 ViewModel = viewModel;  
 DataContext = this;  
  
 InitializeComponent();  
 }  
  
 private void SimpleRepExport\_Click(object sender, RoutedEventArgs e)  
 {  
 Services.ReportSystem.SimpleRepExport\_Handle();  
 }  
  
 private void ComplexRepExport\_Click(object sender, RoutedEventArgs e)  
 {  
 Services.ReportSystem.ComplexRepExport\_Handle();  
 }  
  
  
 private async void DatabaseOptionbutton\_OnClick(object sender, RoutedEventArgs e)  
 {  
 try  
 {  
 System.Windows.Forms.Application.EnableVisualStyles();  
  
  
 var form1 = new FormConnectToSQLServer();  
 form1.ShowDialog();  
 var a = form1.ConnectionString;  
 SettingsPage.formations = a + ";Initial Catalog=KursovayaAvtoparkAvtobusov";  
 form1.Close();  
  
 }  
 catch (Exception exception)  
 {  
 var uiMessageBox = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = exception.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
  
 Width = 800,  
 Height = 300,  
  
 };  
  
  
 uiMessageBox.ShowDialogAsync();  
 }  
  
 }  
  
 private void ComboBox\_SelectionChanged(object sender, SelectionChangedEventArgs e)  
 {  
 ComboBoxItem typeItem = (ComboBoxItem)cmbSelect.SelectedItem;  
 string value = typeItem.Content.ToString();  
 var CurrentApplicationTheme = ApplicationThemeManager.GetAppTheme();  
 var selectedthemedef = CurrentApplicationTheme.ToString();  
 Console.WriteLine(selectedthemedef);  
  
  
  
 switch (value)  
 {  
 case "Темная":  
 if (CurrentApplicationTheme != ApplicationTheme.Dark)  
 {  
 ApplicationThemeManager.Apply(ApplicationTheme.Dark);  
 CurrentApplicationTheme = ApplicationTheme.Dark;  
 }  
 else  
 break;  
  
 break;  
  
 case "Светлая":  
 if (CurrentApplicationTheme != ApplicationTheme.Light)  
 {  
 ApplicationThemeManager.Apply(ApplicationTheme.Light);  
 CurrentApplicationTheme = ApplicationTheme.Light;  
 }  
 else  
 break;  
  
  
 break;  
  
 case "Высокий контраст":  
 if (CurrentApplicationTheme != ApplicationTheme.HighContrast)  
 {  
 ApplicationThemeManager.Apply(ApplicationTheme  
 .HighContrast);  
 CurrentApplicationTheme = ApplicationTheme.HighContrast;  
 }  
 else  
 break;  
  
  
 break;  
  
  
  
  
 break;  
 default:  
 ApplicationThemeManager.Apply(ApplicationTheme.Dark);  
 CurrentApplicationTheme = ApplicationTheme.Dark;  
  
  
 break;  
 }  
 }  
  
 private void deletedbdata\_Click(object sender, RoutedEventArgs e)  
 {  
 Console.WriteLine("в процессе");  
 string value = SlctTable.SelectionBoxItem.ToString();  
 switch (value)  
 {  
 case "Prodazhi":  
 {  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 Console.Write("OPENING DB CONNECTION!!! \n");  
 var select = "DELETE FROM Prodazhi WHERE Num=@a2";  
 var commandBuilder = new SqlCommand(select, cnn);  
 commandBuilder.Parameters.AddWithValue("a2", keybox.Text);  
 try  
 {  
 commandBuilder.ExecuteNonQuery();  
 Console.WriteLine("Delete operation successful!");  
 }  
 catch (SqlException ex)  
 {  
 var uiMessageBox1 = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
  
 Width = 800,  
 Height = 300,  
  
 };  
  
  
 uiMessageBox1.ShowDialogAsync();  
  
 }  
  
 break;  
 }  
  
 case "Marshuti":  
 {  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 Console.Write("OPENING DB CONNECTION!!! \n");  
 var select = "DELETE FROM Marshuti WHERE Nomer\_Marshuta=@a2";  
 var commandBuilder = new SqlCommand(select, cnn);  
 commandBuilder.Parameters.AddWithValue("a2", keybox.Text);  
 try  
 {  
 commandBuilder.ExecuteNonQuery();  
 Console.WriteLine("Delete operation successful!");  
 }  
 catch (SqlException ex)  
 {  
 var uiMessageBox2 = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
  
 Width = 800,  
 Height = 300,  
  
 };  
 uiMessageBox2.ShowDialogAsync();  
  
 }  
  
 break;  
 }  
  
 case "Maintenance":  
 {  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 Console.Write("OPENING DB CONNECTION!!! \n");  
 var select = "DELETE FROM Obsluzhivanie WHERE NomerObsluzhivania=@a2";  
 var commandBuilder = new SqlCommand(select, cnn);  
 commandBuilder.Parameters.AddWithValue("a2", keybox.Text);  
 try  
 {  
 commandBuilder.ExecuteNonQuery();  
 Console.WriteLine("Delete operation successful!");  
 }  
 catch (SqlException ex)  
 {  
 var uiMessageBox4 = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
  
 Width = 800,  
 Height = 300,  
  
 };  
 uiMessageBox4.ShowDialogAsync();  
  
 }  
  
 break;  
 }  
  
 case "Employees":  
 {  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 Console.Write("OPENING DB CONNECTION!!! \n");  
 var select = "DELETE FROM Employees WHERE Num=@a2";  
 var commandBuilder = new SqlCommand(select, cnn);  
 commandBuilder.Parameters.AddWithValue("a2", keybox.Text);  
 try  
 {  
 commandBuilder.ExecuteNonQuery();  
 Console.WriteLine("Delete operation successful!");  
 }  
 catch (SqlException ex)  
 {  
 var uiMessageBox3 = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
  
 Width = 800,  
 Height = 300,  
  
 };  
 uiMessageBox3.ShowDialogAsync();  
  
 }  
  
 break;  
 }  
  
 default:  
   
 var uiMessageBox = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = "Введено неправильное имя таблицы! Пожалуйста введите одно из следующих" +  
 " - Prodazhi или Marshuti или Maintenance или Employees!",  
 TextWrapping = TextWrapping.Wrap,  
 },  
  
 Width = 800,  
 Height = 300,  
  
 };  
 uiMessageBox.ShowDialogAsync();  
   
 //("Введено неправильное имя таблицы!  
 //Пожалуйста введите одно из следующих - Prodazhi или Marshuti или  
 //Maintenance или Employees!");  
  
 break;  
 }  
 }  
 }  
}

using System;  
using System.Data.SqlClient;  
using System.Data;  
using System.Windows;  
using Wpf.Ui.Controls;  
using MessageBox = Wpf.Ui.Controls.MessageBox;  
using TextBlock = System.Windows.Controls.TextBlock;  
  
namespace FluentKursovayaAvtoparkA.Views.Pages  
{  
 /// <summary>  
 /// Логика взаимодействия для TicketPage.xaml  
 /// </summary>  
 public partial class TicketPage : INavigableView<ViewModels.TicketPageViewModel>  
 {  
   
 public ViewModels.TicketPageViewModel ViewModel  
 {  
 get;  
 }  
  
 public TicketPage(ViewModels.TicketPageViewModel viewModel)  
 {  
 ViewModel = viewModel;  
  
 InitializeComponent();  
 }  
 private void UpdateGrid\_Click(object sender, RoutedEventArgs e)  
 {  
 try  
 {  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 Console.Write("OPENING DB CONNECTION!!! \n");  
 Console.Write("[MENU] Connect db clicked - test \n");  
 var select = "SELECT \* FROM Prodazhi";  
 var select2 = "SELECT \* FROM Bilety";  
 var commandBuilder = new SqlCommand(select, cnn);  
 commandBuilder.ExecuteNonQuery();  
 var commandBuilder2 = new SqlCommand(select2, cnn);  
 commandBuilder2.ExecuteNonQuery();  
 var dataAdapter = new SqlDataAdapter(commandBuilder);  
 var dataAdapter2 = new SqlDataAdapter(commandBuilder2);  
 var ds = new DataTable("Avtobusx");  
 dataAdapter.Fill(ds);  
 var ds2 = new DataTable("Avtobusxi");  
 dataAdapter2.Fill(ds2);  
 dataGrid1.ItemsSource = ds.DefaultView;  
 dataGrid2.ItemsSource = ds2.DefaultView;  
 Console.WriteLine("Connection established and the datagrid filled!");  
 cnn.Close();  
 }  
 catch (Exception ex)  
 {  
 var uiMessageBox = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
   
 Width = 800,  
 Height = 300,  
  
 };  
   
  
  
 uiMessageBox.ShowDialogAsync();  
 }  
 }  
  
 private void Button\_Click(object sender, RoutedEventArgs e)  
 {  
 try  
 {  
 var Num = Edit1.Text;  
 var SaleDat = Edit2.Text;  
 var NomBiltetaKey = int.Parse(Edit3.Text);  
 var Nomer\_Biletaex = Edit4.Text;  
 var Nomer\_MarshutaKey = Edit5.Text;  
 var Stoimost = Edit6.Text;  
 var query = "INSERT INTO Prodazhi(Num, Sale\_Date, Nomer\_BiletaKey) " +  
 "Values('" + Num + "', '" + SaleDat + "', '" + NomBiltetaKey + "')";  
 var query2 = "INSERT INTO Bilety(Nomer\_Bileta, Nomer\_MarshutaKey, Stoimost) " +  
 "Values('" + Nomer\_Biletaex + "', '" + Nomer\_MarshutaKey + "', '" + Stoimost + "')";  
 SqlConnection cnn;  
 var conStr = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStr);  
 cnn.Open();  
 var command2 = new SqlCommand(query2, cnn);  
 command2.ExecuteNonQuery();  
 var command = new SqlCommand(query, cnn);  
 command.ExecuteNonQuery();  
 Console.WriteLine("Data inserted!\n Closing connection ");  
 cnn.Close();  
 Console.WriteLine("Connection has been closed , database ready for next operation!");  
 UpdateGrid\_Click(sender, e);  
 }  
  
 catch (Exception ex)  
 {  
 var uiMessageBox = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
   
 Width = 800,  
 Height = 300,  
  
 };  
   
  
  
 uiMessageBox.ShowDialogAsync();  
 }  
 }  
  
 private void SaleUpdateDB\_Click(object sender, RoutedEventArgs e)  
 {  
 var Num = int.Parse(Edit1.Text);  
 var SaleDat = Edit2.Text;  
 var NomBiltetaKey = int.Parse(Edit3.Text);  
 var Nomer\_Biletaex = Edit4.Text;  
 var Nomer\_MarshutaKey = Edit5.Text;  
 var Stoimost = Edit6.Text;  
 try  
 {  
 SqlConnection cnn;  
 var conStrs = SettingsPage.formations;  
  
 cnn = new SqlConnection(conStrs);  
 cnn.Open();  
 Console.Write("OPENING DB CONNECTION!!! \n");  
 var querier =  
 "UPDATE Prodazhi SET Sale\_Date=@a1,Nomer\_BiletaKey=@a2 WHERE Num=@a5";  
 var querier2 =  
 "UPDATE Bilety SET Stoimost=@a4,Nomer\_MarshutaKey=@a3 WHERE Nomer\_Bileta=@a6";  
 var commandBuilder = new SqlCommand(querier, cnn);  
 var commandBuilder2 = new SqlCommand(querier2, cnn);  
 commandBuilder.Parameters.AddWithValue("a1", SaleDat);  
 commandBuilder.Parameters.AddWithValue("a2", NomBiltetaKey);  
 commandBuilder2.Parameters.AddWithValue("a3", Nomer\_MarshutaKey);  
 commandBuilder2.Parameters.AddWithValue("a4", Stoimost);  
 commandBuilder.Parameters.AddWithValue("a5", Num);  
 commandBuilder2.Parameters.AddWithValue("a6", Nomer\_Biletaex);  
 commandBuilder.ExecuteNonQuery();  
 commandBuilder2.ExecuteNonQuery();  
 Console.WriteLine("Connection established and the table updated!");  
 cnn.Close();  
 UpdateGrid\_Click(sender, e);  
 }  
  
 catch (Exception ex)  
 {  
 var uiMessageBox = new MessageBox  
 {  
 Title = "Обработка ошибок",  
 Content = new TextBlock  
 {  
 Text = ex.Message,  
 TextWrapping = TextWrapping.Wrap,  
 },  
   
 Width = 800,  
 Height = 300,  
  
 };  
   
  
 uiMessageBox.ShowDialogAsync();  
 }  
 }  
  
 private void Edit1\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Edit1.Clear();  
 }  
  
 private void Edit2\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Edit2.Clear();  
 }  
  
 private void Edit3\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Edit3.Clear();  
 }  
  
 private void Edit4\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Edit4.Clear();  
 }  
  
 private void Edit5\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Edit5.Clear();  
 }  
 private void Edit6\_MouseDown(object sender, System.Windows.Input.MouseButtonEventArgs e)  
 {  
 Edit6.Clear();  
 }  
 }  
}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Security.Cryptography;

using System.Text;

using System.Threading.Tasks;

namespace SQLServerLoginTemplate

{

// https://thomaslevesque.com/2013/05/21/an-easy-and-secure-way-to-store-a-password-using-data-protection-api/

public static class DataProtectionExtensions

{

public static string Protect(

this string clearText,

string optionalEntropy = null,

DataProtectionScope scope = DataProtectionScope.CurrentUser)

{

if (clearText == null)

throw new ArgumentNullException("clearText");

byte[] clearBytes = Encoding.UTF8.GetBytes(clearText);

byte[] entropyBytes = string.IsNullOrEmpty(optionalEntropy)

? null

: Encoding.UTF8.GetBytes(optionalEntropy);

byte[] encryptedBytes = ProtectedData.Protect(clearBytes, entropyBytes, scope);

return Convert.ToBase64String(encryptedBytes);

}

public static string Unprotect(

this string encryptedText,

string optionalEntropy = null,

DataProtectionScope scope = DataProtectionScope.CurrentUser)

{

if (encryptedText == null)

throw new ArgumentNullException("encryptedText");

byte[] encryptedBytes = Convert.FromBase64String(encryptedText);

byte[] entropyBytes = string.IsNullOrEmpty(optionalEntropy)

? null

: Encoding.UTF8.GetBytes(optionalEntropy);

byte[] clearBytes = new byte[0];

try

{

clearBytes = ProtectedData.Unprotect(encryptedBytes, entropyBytes, scope);

}

catch

{

//Do nothing - just return empty if unable to decrypt the original password

}

return Encoding.UTF8.GetString(clearBytes);

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace SQLServerLoginTemplate

{

public partial class FormConnectToSQLServer : Form

{

private readonly string \_WINDOWSAUTH = "Windows Authentication";

private readonly string \_SQLAUTH = "SQL Server Authentication";

private readonly string \_LOGINTYPEWINDOWS = "0";

private readonly string \_LOGINTYPESQL = "1";

private readonly string \_SERVERANDUSERSEPARATOR = "@&";

private readonly string \_USERANDPASSWORDSEPARATOR = "~~~";

public string ConnectionString = ""; //The code that calls this form should retrieve the connection string from this public property

public FormConnectToSQLServer()

{

InitializeComponent();

}

private void buttonCancel\_Click(object sender, EventArgs e)

{

this.DialogResult = DialogResult.Cancel;

this.Close();

}

private void FormConnectToSQLServer\_Load(object sender, EventArgs e)

{

GetDPI();

LoadSharedDefaultValues();

InitializePropertiesForFirstTimeUsers();

LoadLastServerSelectedByThisUser();

FormConnectToSQLServer\_Resize(sender, e);

}

bool \_highDPI = false;

private void GetDPI()

{

float dx, dy;

Graphics g = this.CreateGraphics();

try

{

dx = g.DpiX;

dy = g.DpiY;

if (dx > 96)

\_highDPI = true;

}

finally

{

g.Dispose();

}

}

public void LoadLastServerSelectedByThisUser()

{

if (Properties.Settings.Default.ServerNames != null)

{

var savedNames = Properties.Settings.Default.ServerNames;

foreach (var item in savedNames)

{

this.comboBoxServerName.Items.Add(item);

}

}

if (Properties.Settings.Default.LoginNames != null)

{

var savedNames = Properties.Settings.Default.LoginNames;

foreach (var item in savedNames)

{

this.comboBoxLogin.Items.Add(item);

}

}

//The changed event on this combobox will trigger other values to load

comboBoxServerName.Text = Properties.Settings.Default.LastServerName;

}

private void buttonConnect\_Click(object sender, EventArgs e)

{

EnableOrDisableAllFields(false);

if (TestUserCredentials())

{

Properties.Settings.Default.Save();

this.Close();

}

else

{

EnableOrDisableAllFields(true);

EnableOrDisableFields(comboBoxAuthentication.Text.Equals(\_SQLAUTH));

this.DialogResult = DialogResult.None;

}

}

private bool TestUserCredentials()

{

string selectedServer = comboBoxServerName.Text.Trim();

string authType = comboBoxAuthentication.Text.Trim();

string authNum = \_LOGINTYPEWINDOWS;

if (authType.Equals(\_SQLAUTH))

authNum = \_LOGINTYPESQL;

string selectedLogin = comboBoxLogin.Text.Trim();

string concat = selectedServer + \_SERVERANDUSERSEPARATOR + authNum + selectedLogin;

if (!Properties.Settings.Default.ServerNames.Contains(selectedServer))

{

Properties.Settings.Default.ServerNames.Add(selectedServer);

}

if (!string.IsNullOrWhiteSpace(selectedLogin))

{

if (!Properties.Settings.Default.LoginNames.Contains(selectedLogin))

{

Properties.Settings.Default.LoginNames.Add(selectedLogin);

}

}

Properties.Settings.Default.LastServerName = selectedServer;

Properties.Settings.Default.LastLogin = selectedLogin;

Properties.Settings.Default.RememberChecked = checkBoxRememberPassword.Checked;

if (checkBoxRememberPassword.Checked)

{

string encryptedPassword = textBoxPassword.Text.Trim().Protect(selectedServer);

concat += \_USERANDPASSWORDSEPARATOR + encryptedPassword;

}

bool found = false;

foreach (var item in Properties.Settings.Default.ConcatStuff)

{

if (item.StartsWith(selectedServer + \_SERVERANDUSERSEPARATOR))

{

Properties.Settings.Default.ConcatStuff.Remove(item);

Properties.Settings.Default.ConcatStuff.Add(concat);

found = true;

break;

}

}

if (found == false)

{

Properties.Settings.Default.ConcatStuff.Add(concat);

}

string login = "Integrated Security=true";

if (authType.Equals(\_SQLAUTH))

{

login = $"user id={selectedLogin};password={textBoxPassword.Text.Trim()}";

}

return AttemptDatabaseConnection("Data Source=" + selectedServer + ";" + login, selectedServer);

}

private bool AttemptDatabaseConnection(string connString, string selectedServer)

{

try

{

using (System.Data.SqlClient.SqlConnection conn = new System.Data.SqlClient.SqlConnection(connString))

{

conn.Open();

ConnectionString = connString; //The code that calls this form should retrieve the connection string from this public property

}

#if DEBUG

MessageBox.Show("Connection string " + connString + " result: true");

#endif

}

catch (Exception ex)

{

MessageBox.Show("Cannot connect to " + selectedServer + "." + Environment.NewLine + Environment.NewLine + "Additional information: " + Environment.NewLine + ex.Message, "Connect to Server", MessageBoxButtons.OK);

return false;

}

return true;

}

private void comboBoxServerName\_SelectedIndexChanged(object sender, EventArgs e)

{

string selectedServer = comboBoxServerName.Text.Trim();

foreach (var item in Properties.Settings.Default.ConcatStuff)

{

if (item.StartsWith(selectedServer + \_SERVERANDUSERSEPARATOR))

PopulateDefaultsForSelectedServer(item.Replace(selectedServer + \_SERVERANDUSERSEPARATOR, ""));

}

if (string.IsNullOrWhiteSpace(comboBoxAuthentication.Text))

{

comboBoxAuthentication.Text = comboBoxAuthentication.Items[0].ToString();

}

}

private void PopulateDefaultsForSelectedServer(string concat)

{

string lastAuth = concat.Substring(0, 1);

if (lastAuth == \_LOGINTYPEWINDOWS)

{

comboBoxAuthentication.Text = \_WINDOWSAUTH;

}

else

{

comboBoxAuthentication.Text = \_SQLAUTH;

concat = concat.Substring(1);

int indexOfPassword = concat.IndexOf(\_USERANDPASSWORDSEPARATOR);

if (indexOfPassword >= 0)

{

comboBoxLogin.Text = concat.Substring(0, indexOfPassword);

string lastPwd = concat.Substring(indexOfPassword + \_USERANDPASSWORDSEPARATOR.Length);

textBoxPassword.Text = lastPwd.Unprotect(comboBoxServerName.Text.Trim());

checkBoxRememberPassword.Checked = true;

}

else

{

comboBoxLogin.Text = concat;

textBoxPassword.Text = "";

checkBoxRememberPassword.Checked = false;

}

}

}

private void comboBoxAuthentication\_SelectedIndexChanged(object sender, EventArgs e)

{

EnableDisableBasedOnAuthMethod(comboBoxAuthentication.Text.Equals(\_SQLAUTH));

}

#region Helper Methods

private static void InitializePropertiesForFirstTimeUsers()

{

if (Properties.Settings.Default.ServerNames == null)

{

Properties.Settings.Default.ServerNames = new System.Collections.Specialized.StringCollection();

}

if (Properties.Settings.Default.LoginNames == null)

{

Properties.Settings.Default.LoginNames = new System.Collections.Specialized.StringCollection();

}

if (Properties.Settings.Default.ConcatStuff == null)

{

Properties.Settings.Default.ConcatStuff = new System.Collections.Specialized.StringCollection();

}

}

private void EnableOrDisableAllFields(bool enable)

{

EnableOrDisableFields(enable);

comboBoxAuthentication.Enabled = enable;

comboBoxServerName.Enabled = enable;

labelAuthentication.Enabled = enable;

labelServerName.Enabled = enable;

comboBoxServerType.Enabled = enable;

labelServerType.Enabled = enable;

buttonConnect.Enabled = enable;

buttonCancel.Enabled = enable; //The real SQL Server UI allows you to cancel

}

private void EnableDisableBasedOnAuthMethod(bool sqlAuth)

{

EnableOrDisableFields(sqlAuth);

if (sqlAuth == false)

{

comboBoxLogin.Text = "";

textBoxPassword.Text = "";

}

}

private void EnableOrDisableFields(bool enable)

{

comboBoxLogin.Enabled = enable;

textBoxPassword.Enabled = enable;

checkBoxRememberPassword.Enabled = enable;

labelLogin.Enabled = enable;

labelPassword.Enabled = enable;

}

private void LoadSharedDefaultValues()

{

buttonHelp.Visible = false;

buttonOptions.Visible = false;

comboBoxAuthentication.Items.Add(\_WINDOWSAUTH);

comboBoxAuthentication.Items.Add(\_SQLAUTH);

comboBoxAuthentication.Text = \_WINDOWSAUTH;

comboBoxAuthentication.DropDownStyle = ComboBoxStyle.DropDownList;

comboBoxServerType.Items.Add("Database Engine");

comboBoxServerType.Text = "Database Engine";

comboBoxServerType.DropDownStyle = ComboBoxStyle.DropDownList;

}

#endregion

private void FormConnectToSQLServer\_Resize(object sender, EventArgs e)

{

int rightoffset = 64;

if (!\_highDPI)

{

rightoffset = 28;

this.Height = 320;

}

comboBoxServerType.Width = this.Width - comboBoxServerType.Left - rightoffset;

comboBoxServerName.Width = this.Width - comboBoxServerName.Left - rightoffset;

comboBoxLogin.Width = this.Width - comboBoxLogin.Left - rightoffset;

comboBoxAuthentication.Width = this.Width - comboBoxAuthentication.Left - rightoffset;

textBoxPassword.Width = this.Width - textBoxPassword.Left - rightoffset;

}

}

}

namespace SQLServerLoginTemplate

{

partial class FormConnectToSQLServer

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

this.labelServerName = new System.Windows.Forms.Label();

this.labelAuthentication = new System.Windows.Forms.Label();

this.labelLogin = new System.Windows.Forms.Label();

this.labelPassword = new System.Windows.Forms.Label();

this.comboBoxServerName = new System.Windows.Forms.ComboBox();

this.comboBoxLogin = new System.Windows.Forms.ComboBox();

this.textBoxPassword = new System.Windows.Forms.TextBox();

this.buttonConnect = new System.Windows.Forms.Button();

this.buttonCancel = new System.Windows.Forms.Button();

this.buttonHelp = new System.Windows.Forms.Button();

this.buttonOptions = new System.Windows.Forms.Button();

this.checkBoxRememberPassword = new System.Windows.Forms.CheckBox();

this.label1 = new System.Windows.Forms.Label();

this.labelSQLServer = new System.Windows.Forms.Label();

this.comboBoxAuthentication = new System.Windows.Forms.ComboBox();

this.labelServerType = new System.Windows.Forms.Label();

this.comboBoxServerType = new System.Windows.Forms.ComboBox();

this.panel1 = new System.Windows.Forms.Panel();

this.panel2 = new System.Windows.Forms.Panel();

this.panel1.SuspendLayout();

this.SuspendLayout();

//

// labelServerName

//

this.labelServerName.AutoSize = true;

this.labelServerName.Location = new System.Drawing.Point(13, 108);

this.labelServerName.Margin = new System.Windows.Forms.Padding(1, 0, 1, 0);

this.labelServerName.Name = "labelServerName";

this.labelServerName.Size = new System.Drawing.Size(105, 13);

this.labelServerName.TabIndex = 0;

this.labelServerName.Text = "&Название сервера:";

//

// labelAuthentication

//

this.labelAuthentication.AutoSize = true;

this.labelAuthentication.Location = new System.Drawing.Point(13, 136);

this.labelAuthentication.Margin = new System.Windows.Forms.Padding(1, 0, 1, 0);

this.labelAuthentication.Name = "labelAuthentication";

this.labelAuthentication.Size = new System.Drawing.Size(61, 13);

this.labelAuthentication.TabIndex = 1;

this.labelAuthentication.Text = "&Тип входа:";

//

// labelLogin

//

this.labelLogin.AutoSize = true;

this.labelLogin.Location = new System.Drawing.Point(33, 162);

this.labelLogin.Margin = new System.Windows.Forms.Padding(1, 0, 1, 0);

this.labelLogin.Name = "labelLogin";

this.labelLogin.Size = new System.Drawing.Size(41, 13);

this.labelLogin.TabIndex = 2;

this.labelLogin.Text = "&Логин:";

//

// labelPassword

//

this.labelPassword.AutoSize = true;

this.labelPassword.Location = new System.Drawing.Point(33, 187);

this.labelPassword.Margin = new System.Windows.Forms.Padding(1, 0, 1, 0);

this.labelPassword.Name = "labelPassword";

this.labelPassword.Size = new System.Drawing.Size(48, 13);

this.labelPassword.TabIndex = 3;

this.labelPassword.Text = "&Пароль:";

//

// comboBoxServerName

//

this.comboBoxServerName.FormattingEnabled = true;

this.comboBoxServerName.Location = new System.Drawing.Point(171, 105);

this.comboBoxServerName.Margin = new System.Windows.Forms.Padding(1);

this.comboBoxServerName.Name = "comboBoxServerName";

this.comboBoxServerName.Size = new System.Drawing.Size(306, 21);

this.comboBoxServerName.TabIndex = 6;

this.comboBoxServerName.SelectedIndexChanged += new System.EventHandler(this.comboBoxServerName\_SelectedIndexChanged);

//

// comboBoxLogin

//

this.comboBoxLogin.FormattingEnabled = true;

this.comboBoxLogin.Location = new System.Drawing.Point(191, 159);

this.comboBoxLogin.Margin = new System.Windows.Forms.Padding(1);

this.comboBoxLogin.Name = "comboBoxLogin";

this.comboBoxLogin.Size = new System.Drawing.Size(287, 21);

this.comboBoxLogin.TabIndex = 8;

//

// textBoxPassword

//

this.textBoxPassword.Location = new System.Drawing.Point(191, 184);

this.textBoxPassword.Margin = new System.Windows.Forms.Padding(1);

this.textBoxPassword.Name = "textBoxPassword";

this.textBoxPassword.PasswordChar = '\*';

this.textBoxPassword.Size = new System.Drawing.Size(287, 20);

this.textBoxPassword.TabIndex = 9;

//

// buttonConnect

//

this.buttonConnect.DialogResult = System.Windows.Forms.DialogResult.OK;

this.buttonConnect.Location = new System.Drawing.Point(112, 242);

this.buttonConnect.Margin = new System.Windows.Forms.Padding(1);

this.buttonConnect.Name = "buttonConnect";

this.buttonConnect.Size = new System.Drawing.Size(110, 21);

this.buttonConnect.TabIndex = 10;

this.buttonConnect.Text = "&Подключится";

this.buttonConnect.UseVisualStyleBackColor = true;

this.buttonConnect.Click += new System.EventHandler(this.buttonConnect\_Click);

//

// buttonCancel

//

this.buttonCancel.DialogResult = System.Windows.Forms.DialogResult.Cancel;

this.buttonCancel.Location = new System.Drawing.Point(228, 242);

this.buttonCancel.Margin = new System.Windows.Forms.Padding(1);

this.buttonCancel.Name = "buttonCancel";

this.buttonCancel.Size = new System.Drawing.Size(69, 21);

this.buttonCancel.TabIndex = 11;

this.buttonCancel.Text = "Отмена";

this.buttonCancel.UseVisualStyleBackColor = true;

this.buttonCancel.Click += new System.EventHandler(this.buttonCancel\_Click);

//

// buttonHelp

//

this.buttonHelp.Location = new System.Drawing.Point(302, 241);

this.buttonHelp.Margin = new System.Windows.Forms.Padding(1);

this.buttonHelp.Name = "buttonHelp";

this.buttonHelp.Size = new System.Drawing.Size(81, 22);

this.buttonHelp.TabIndex = 12;

this.buttonHelp.Text = "&Помощь";

this.buttonHelp.UseVisualStyleBackColor = true;

//

// buttonOptions

//

this.buttonOptions.Location = new System.Drawing.Point(385, 242);

this.buttonOptions.Margin = new System.Windows.Forms.Padding(1);

this.buttonOptions.Name = "buttonOptions";

this.buttonOptions.Size = new System.Drawing.Size(91, 21);

this.buttonOptions.TabIndex = 13;

this.buttonOptions.Text = "&Настройки >>";

this.buttonOptions.UseVisualStyleBackColor = true;

//

// checkBoxRememberPassword

//

this.checkBoxRememberPassword.AutoSize = true;

this.checkBoxRememberPassword.Location = new System.Drawing.Point(191, 212);

this.checkBoxRememberPassword.Margin = new System.Windows.Forms.Padding(1);

this.checkBoxRememberPassword.Name = "checkBoxRememberPassword";

this.checkBoxRememberPassword.Size = new System.Drawing.Size(121, 17);

this.checkBoxRememberPassword.TabIndex = 14;

this.checkBoxRememberPassword.Text = "Запомнить пароль";

this.checkBoxRememberPassword.UseVisualStyleBackColor = true;

//

// label1

//

this.label1.BorderStyle = System.Windows.Forms.BorderStyle.Fixed3D;

this.label1.Location = new System.Drawing.Point(11, 230);

this.label1.Margin = new System.Windows.Forms.Padding(1, 0, 1, 0);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(463, 4);

this.label1.TabIndex = 15;

//

// labelSQLServer

//

this.labelSQLServer.AutoSize = true;

this.labelSQLServer.Font = new System.Drawing.Font("Microsoft Sans Serif", 22F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

this.labelSQLServer.Location = new System.Drawing.Point(126, 9);

this.labelSQLServer.Margin = new System.Windows.Forms.Padding(1, 0, 1, 0);

this.labelSQLServer.Name = "labelSQLServer";

this.labelSQLServer.Size = new System.Drawing.Size(204, 36);

this.labelSQLServer.TabIndex = 16;

this.labelSQLServer.Text = "Авторизация";

//

// comboBoxAuthentication

//

this.comboBoxAuthentication.FormattingEnabled = true;

this.comboBoxAuthentication.Location = new System.Drawing.Point(171, 133);

this.comboBoxAuthentication.Margin = new System.Windows.Forms.Padding(1);

this.comboBoxAuthentication.Name = "comboBoxAuthentication";

this.comboBoxAuthentication.Size = new System.Drawing.Size(306, 21);

this.comboBoxAuthentication.TabIndex = 7;

this.comboBoxAuthentication.SelectedIndexChanged += new System.EventHandler(this.comboBoxAuthentication\_SelectedIndexChanged);

//

// labelServerType

//

this.labelServerType.AutoSize = true;

this.labelServerType.Location = new System.Drawing.Point(13, 80);

this.labelServerType.Margin = new System.Windows.Forms.Padding(1, 0, 1, 0);

this.labelServerType.Name = "labelServerType";

this.labelServerType.Size = new System.Drawing.Size(74, 13);

this.labelServerType.TabIndex = 17;

this.labelServerType.Text = "Тип сервера:";

//

// comboBoxServerType

//

this.comboBoxServerType.FormattingEnabled = true;

this.comboBoxServerType.Location = new System.Drawing.Point(171, 80);

this.comboBoxServerType.Margin = new System.Windows.Forms.Padding(1);

this.comboBoxServerType.Name = "comboBoxServerType";

this.comboBoxServerType.Size = new System.Drawing.Size(306, 21);

this.comboBoxServerType.TabIndex = 18;

//

// panel1

//

this.panel1.BackColor = System.Drawing.Color.White;

this.panel1.Controls.Add(this.panel2);

this.panel1.Controls.Add(this.labelSQLServer);

this.panel1.Location = new System.Drawing.Point(-2, 0);

this.panel1.Name = "panel1";

this.panel1.Size = new System.Drawing.Size(503, 63);

this.panel1.TabIndex = 19;

//

// panel2

//

this.panel2.BackColor = System.Drawing.Color.DarkOrange;

this.panel2.Location = new System.Drawing.Point(0, 54);

this.panel2.Name = "panel2";

this.panel2.Size = new System.Drawing.Size(500, 10);

this.panel2.TabIndex = 17;

//

// FormConnectToSQLServer

//

this.AcceptButton = this.buttonConnect;

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.CancelButton = this.buttonCancel;

this.ClientSize = new System.Drawing.Size(484, 282);

this.Controls.Add(this.panel1);

this.Controls.Add(this.comboBoxServerType);

this.Controls.Add(this.labelServerType);

this.Controls.Add(this.label1);

this.Controls.Add(this.checkBoxRememberPassword);

this.Controls.Add(this.buttonOptions);

this.Controls.Add(this.buttonHelp);

this.Controls.Add(this.buttonCancel);

this.Controls.Add(this.buttonConnect);

this.Controls.Add(this.textBoxPassword);

this.Controls.Add(this.comboBoxLogin);

this.Controls.Add(this.comboBoxAuthentication);

this.Controls.Add(this.comboBoxServerName);

this.Controls.Add(this.labelPassword);

this.Controls.Add(this.labelLogin);

this.Controls.Add(this.labelAuthentication);

this.Controls.Add(this.labelServerName);

this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedDialog;

this.Margin = new System.Windows.Forms.Padding(1);

this.MaximizeBox = false;

this.MinimizeBox = false;

this.Name = "FormConnectToSQLServer";

this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;

this.Text = "Вход в систему";

this.Load += new System.EventHandler(this.FormConnectToSQLServer\_Load);

this.Resize += new System.EventHandler(this.FormConnectToSQLServer\_Resize);

this.panel1.ResumeLayout(false);

this.panel1.PerformLayout();

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Label labelServerName;

private System.Windows.Forms.Label labelAuthentication;

private System.Windows.Forms.Label labelLogin;

private System.Windows.Forms.Label labelPassword;

private System.Windows.Forms.ComboBox comboBoxServerName;

private System.Windows.Forms.ComboBox comboBoxLogin;

private System.Windows.Forms.TextBox textBoxPassword;

private System.Windows.Forms.Button buttonConnect;

private System.Windows.Forms.Button buttonCancel;

private System.Windows.Forms.Button buttonHelp;

private System.Windows.Forms.Button buttonOptions;

private System.Windows.Forms.CheckBox checkBoxRememberPassword;

private System.Windows.Forms.Label label1;

private System.Windows.Forms.Label labelSQLServer;

private System.Windows.Forms.ComboBox comboBoxAuthentication;

private System.Windows.Forms.Label labelServerType;

private System.Windows.Forms.ComboBox comboBoxServerType;

private System.Windows.Forms.Panel panel1;

private System.Windows.Forms.Panel panel2;

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace SQLServerLoginTemplate

{

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new FormConnectToSQLServer());

}

}

}

namespace SQLServerLoginTemplate.Properties {

// This class allows you to handle specific events on the settings class:

// The SettingChanging event is raised before a setting's value is changed.

// The PropertyChanged event is raised after a setting's value is changed.

// The SettingsLoaded event is raised after the setting values are loaded.

// The SettingsSaving event is raised before the setting values are saved.

internal sealed partial class Settings {

public Settings() {

// // To add event handlers for saving and changing settings, uncomment the lines below:

//

// this.SettingChanging += this.SettingChangingEventHandler;

//

// this.SettingsSaving += this.SettingsSavingEventHandler;

//

}

private void SettingChangingEventHandler(object sender, System.Configuration.SettingChangingEventArgs e) {

// Add code to handle the SettingChangingEvent event here.

}

private void SettingsSavingEventHandler(object sender, System.ComponentModel.CancelEventArgs e) {

// Add code to handle the SettingsSaving event here.

}

}

}