Государственное бюджетное профессиональное образовательное учреждение «Южно-Уральский Государственный колледж»

ПЦК «Информационных технологий»

**КУРСОВАЯ РАБОТА**

по междисциплинарному курсу  
 01.01 Разработка программных модулей

Тема: Разработка АИС «ТехноГид»

Выполнил:

студент группы ИСп320Д

Гирфанов А.Р./\_\_\_\_\_\_\_\_\_\_\_

«12» февраля 2024 г.

Проверил:

преподаватель ПЦК ИТ

Исакова В.В. /\_\_\_\_\_\_\_\_\_\_\_\_

«12» февраля 2024 г.

Работа защищена

«12» февраля 2024 г.

с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Челябинск 2024 г.

Содержание

[Введение 3](#_Toc158058360)

[1.Теоретическая часть 4](#_Toc158058361)

[2.Практическая часть 8](#_Toc158058362)

[2.1. Разработка приложения 8](#_Toc158058363)

[2.2. Руководство программиста 24](#_Toc158058364)

[2.3. Руководство пользователя 25](#_Toc158058365)

[Заключение 27](#_Toc158058366)

[Список литературы 28](#_Toc158058367)

[Приложение 29](#_Toc158058368)

[Код страницы *MainWindow* 29](#_Toc158058369)

[Код страницы *WorkingWindow* 30](#_Toc158058370)

[Код страницы *NotesPage* 35](#_Toc158058371)

[Код страницы *ClientDataPage* 40](#_Toc158058372)

Введение

Благодаря активному развитию технологий в современном мире вырос спрос на средства, предназначенные упростить решение повседневных задач и сэкономить время. Данный проект поможет достичь этого путем сокращения в несколько раз траты временного ресурса старших и опытных сотрудников для обучения начинающих. С помощью взаимодействия с базой данных будут решаться типовые проблемы, возникающие у клиентов.

Целью курсовой работы является разработка АИС «Техно Гид» для технической поддержки.

Задачи курсовой работы состоят из следующих пунктов:

* Разработать алгоритмы функционала приложения.
* Разработать дизайн форм для АИС.
* Интегрировать функционал в формы приложения.

1. Теоретическая часть

Преимущества языка программирования C# и среды разработки Visual Studio заключаются в следующих пунктах:

* Кроссплатформенность.

С# может быть использован при разработке приложений для различных операционных систем, таких как Linux, macOS или Windows. Эта возможно благодаря .NET Core – платформа разработки ПО. Она представляет собой среду выполнения CLR (Common Language Runtime), которая обеспечивает управление памятью, безопасность типов, сборку мусора и другие функции, необходимые для выполнения приложений, включает в себя различные готовые классы и инструменты, созданные для решения различных задач при разработке ПО

* Широкий инструментарий.

C# имеет богатую стандартную библиотеку классов, которая предоставляет широкий набор функциональности для разработки приложений.

* Простота использования:

C# имеет простой и понятный синтаксис, что делает его легким для изучения и использования даже начинающим разработчикам.

* Поддержка объектно-ориентированного программирования:

C# полностью поддерживает принципы ООП, такие как наследование, полиморфизм и инкапсуляцию, что делает код более организованным и легко поддерживаемым.

* Безопасность:

C# обеспечивает высокий уровень безопасности благодаря своей типизации и проверке границ массивов во время выполнения.

* Интеграция с платформой .NET:

C# является основным языком разработки для платформы .NET, что обеспечивает доступ к множеству инструментов и библиотек для разработки приложений.

* Поддержка асинхронного и многопоточного программирования:

C# предоставляет инструменты для удобного написания параллельно выполняющихся блоков кода, что позволяет использовать все преимущества многоядерных процессоров.

* Большое сообщество разработчиков:

C# имеет большое и активное сообщество разработчиков, которые готовы помочь и поделиться опытом.

* Интеграция с другими языками программирования:

C# может быть использован вместе с другими языками программирования, такими как C++ и F#, что позволяет создавать более сложные и мощные приложения.

* Поддержка разработки игр:

C# является одним из основных языков программирования для разработки игр на платформе Unity, что делает его привлекательным выбором для разработчиков игр.

*Преимущества языка программирования JAVA:*

1. Платформонезависимость: программа, написанная на языке Java, может выполняться на любой операционной системе без изменений.

2. Безопасность: Java имеет встроенную систему безопасности, которая предотвращает некорректное использование памяти и обеспечивает защиту от вирусов и других угроз.

3. Многопоточность: Java поддерживает многопоточность, что позволяет создавать многопоточные приложения, работающие более эффективно.

4. Объектно-ориентированный подход: Java построен на принципах объектно-ориентированного программирования, что облегчает разработку сложных приложений.

5. Богатая стандартная библиотека: Java имеет обширную стандартную библиотеку, которая содержит множество классов и методов для решения различных задач.

*Недостатки языка программирования JAVA:*

1. Низкая производительность: Java не является самым быстрым языком программирования, что может быть проблемой для некоторых приложений.

2. Высокое потребление памяти: Java требует большого объема памяти для выполнения программ, что может быть проблемой для устройств с ограниченными ресурсами.

3. Сложность: Java может быть сложным для начинающих программистов из-за его многопоточности и объектно-ориентированного подхода.

4. Неудобочитаемый код: Иногда код на Java может быть более сложным для чтения и понимания, чем код на других языках программирования.

При разработке АИС «ТехноГид» была выбрана Visual Studio 2022 и высокоуровневый язык программирования C#, входящий в стандартный набор языков программирования IDE Visual Studio по ряду причин:

1. Многофункциональная среда разработки.
2. C# имеет в себе множество инструментов, позволяющих писать код, более компактно и быстро.
3. Имеет «Сборщик мусора», работающий в автоматическом режиме, позволяя избегать возможных шибак при взаимодействии с памятью.
4. Практическая часть
   1. Разработка приложения

При запуске приложения пользователь видит окно авторизации пользователя.

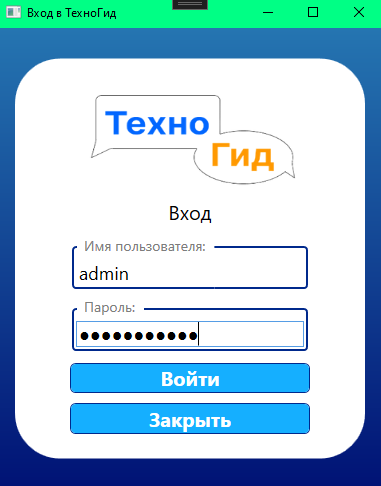


Рисунок 1. Окно авторизации

После нажатия на кнопку «Войти» с помощью «Entity Framework» происходит получение экземпляра базы данных, после чего выполняется циклическое получение экземпляра таблицы User из базы данных и, с помощью логических операций, сравнивается с введенным логином пользователя, в случае совпадения выполняется такая же проверка, но уже для введенного пользователем пароля.



Рисунок 2. Код MainWindow

При успешной попытке авторизации, создается экземпляр класса User и передается в качестве аргумента в конструктор WorkingWindow (рабочее окно), в котором получает доступ к главной странице АИС «ТехноГид» (рисунок 3).

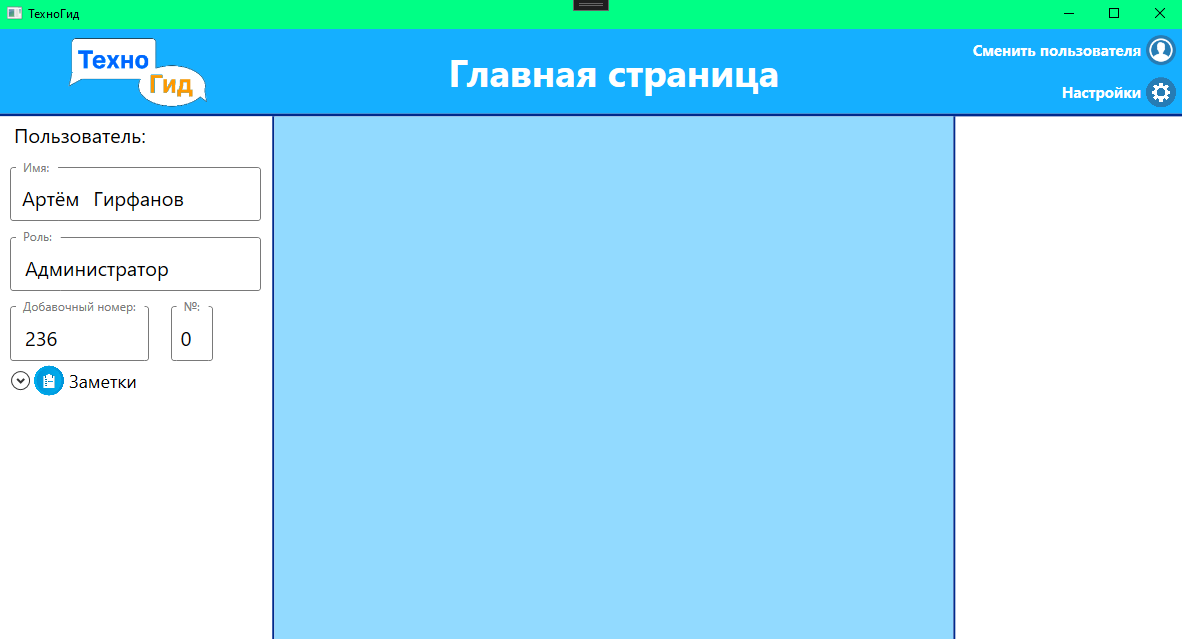


Рисунок 3. Главная страница администратора

Для вывода информации о пользователе используется Binding (привязка данных) на объектах формы. Она позволяет с помощью «контекста» передавать ссылки на данные в свойства объектов формы.

При смене пользователя срабатывает событие «Click» которое выполняет метод «ChangeUserClick» (рисунок 4). Его суть заключается в вызове конструктора для MainWindow (окна авторизации) и передаче логина текущего пользователя в качестве аргумента и закрывает текущее окно, благодаря этому, при открытии окна авторизации с помощью данного метода, в поле «Логин» будет записан логин пользователя, нажавшего на кнопку.

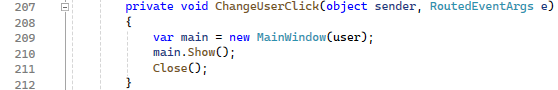
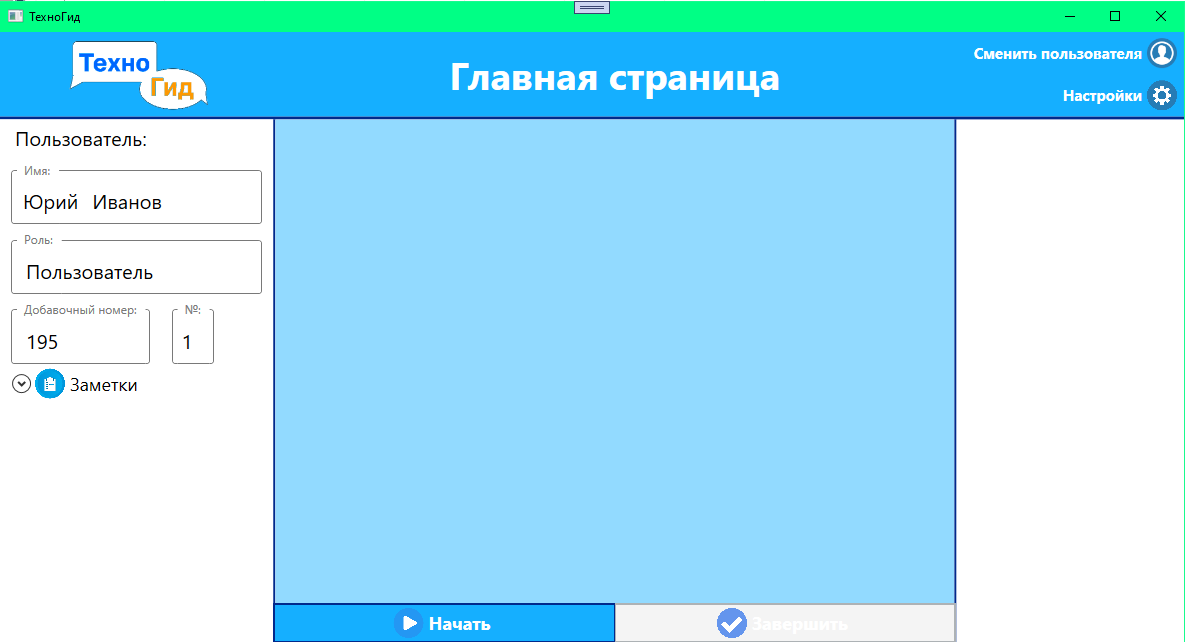


Рисунок 4. Метод ChangeUserClick

Каждый пользователь имеет свою роль в данной АИС, у каждого из них есть уникальные возможности и уровень доступа к возможностям других пользователей.

Например, пользователь с правами администратора не обрабатывает обращения клиентов, поэтому ему недоступно начало рабочей сессии, как для «Пользователя» или «Редактора» (рисунок 5).



Информация о клиенте

Завершить сессию

Начать сессию

Рисунок 5. Главная страница пользователя

«Заметки» (рисунок 6) доступны всем пользователям. При взаимодействии с данным объектом срабатывает событие «Expanded», выполняющее метод «NotesExpanded» (рисунок 7). Данный метод передает ссылку на экземпляр класса «NotesPage» в метод «Navigate» объекта класса «Frame» и выполняет метод «WriteToNotes» (рисунок 8), который записывает в текстовое поле «заметок» данные из текстового файла, путь к которому указывается в настройках.

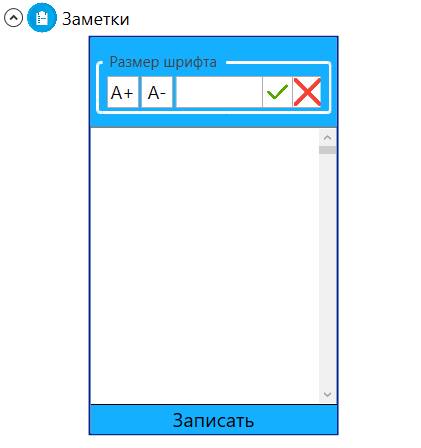


Рисунок 6. Заметки

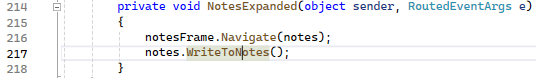


Рисунок 7. Метод NotesExpanded



Рисунок 8. Метод WriteToNotes

Страница заметок (рисунок 6) - объект класса «NotesPage», который реализует интерфейс «IWorkWithFileInterface» (рисунок 9). Этот интерфейс обязывает в структуре класса «NotesPage» использовать методы для работы с файловой системой –«WriteToFile» - для записи данных в файл (рисунок 10) и «ReadToFile» - для считывания данных из файла (рисунок 11).

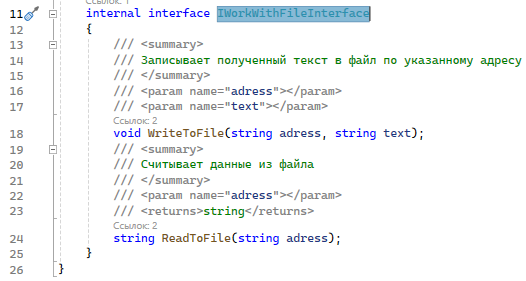


Рисунок 9. Интерфейс IWorkWithFileInterface

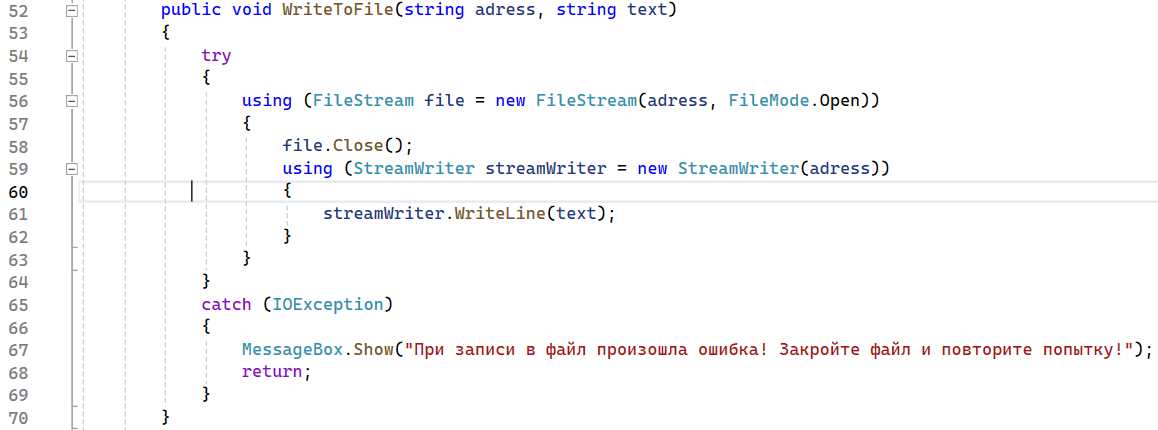


Рисунок 10. Код метода WriteToFile

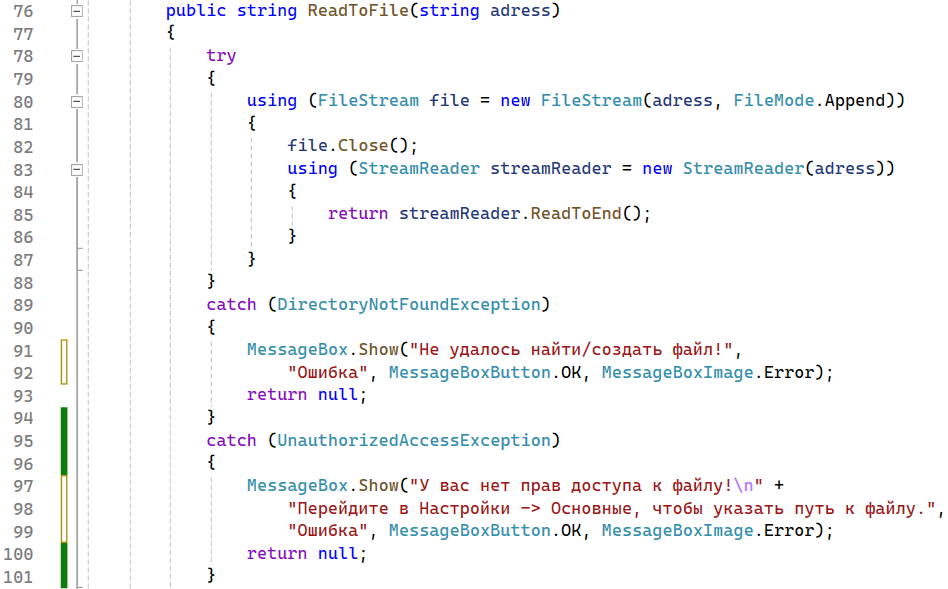


Рисунок 11. Код метода ReadToFile

Данная страница является текстовым редактором, имеющим минимально необходимый инструментарий для заметок:

1. Возможность изменять размер шрифта;
2. Записывание данных в текстовый файл;
3. Считывание сохраненных данных из текстового файла.

Для изменения размера шрифта используются 2 кнопки:

* Повышение размера шрифта на 1 позицию (A+).
* Понижение размера шрифта на 1 позицию (A-).

А также текстовое поле для ввода необходимого значения размера шрифта.

При повышении/понижении размера шрифта на 1 позицию происходит логическое сравнение со значениями первого и последнего элементов массива часто используемых размеров шрифта, после этого изменяется значение свойства «FontSize» для текстового поля заметок (рисунок 12).



Рисунок 12. Код повышения шрифта на 1 позицию.

Начало рабочей сессии запускает событие «Click», выполняющее метод «StartButtonClick» (рисунок 17). Данный метод запускает конструктор для окна «ClientChoise» - выбор клиента (рисунок 13). После выбора клиента по одному из выбранных свойств происходит поиск в базе данных (рисунок 14), если поиск удачный – ссылка на объект класса «Client» передаётся в «WorkingWindow» и с помощью привязки данных выводится информация о клиенте в отведенное для этого поле (рисунок 4). Так же каждый пользователь может добавить клиента в базу данных, нажав на кнопку «Добавить» (рисунок 13) в окне выбора клиента. В таком случае запускается конструктор для «AddEditWindow» - окна добавления/изменения записей базы данных (рисунок 15), и передается новый пользователь в качестве аргумента для конструктора.

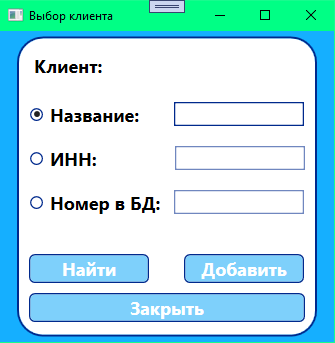


Рисунок 13. Окно выбора клиента



Рисунок 14. Код метода «SearchClientClick»

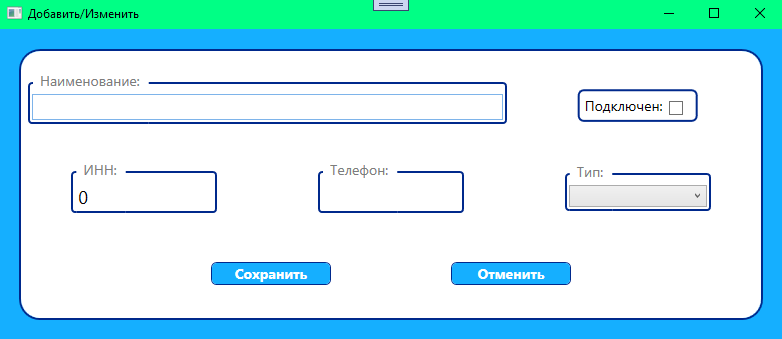


Рисунок 15. Окно добавления/изменения записей базы данных

«AddEditWindow» (рисунок 15) имеет множество перегрузок конструктора, в зависимости от передаваемого класса меняется структура страницы, которая вызывается в теле конструктора.



Рисунок 16. Код окна AddEditWindow

Каждая из страниц, присутствующих в коде данного окна, состоит из полей для ввода/выбора данных о тех или иных свойствах объектах классов.

После начала рабочей сессии и удачном «поиске клиента» происходит обращение к таблице «QuestionAfterAnswer», затем выполняется поиск объекта, свойство «AnswerID» которого равно «null». При удачном поиске вызывается конструктор класса «ButtonQuestion». Свойствам данного объекта присваиваются данные, найденные по «QuestionID» в таблице «Question». Затем созданный объект помещается в предназначенное для него поле «WorkingFrame» с помощью метода «Children».

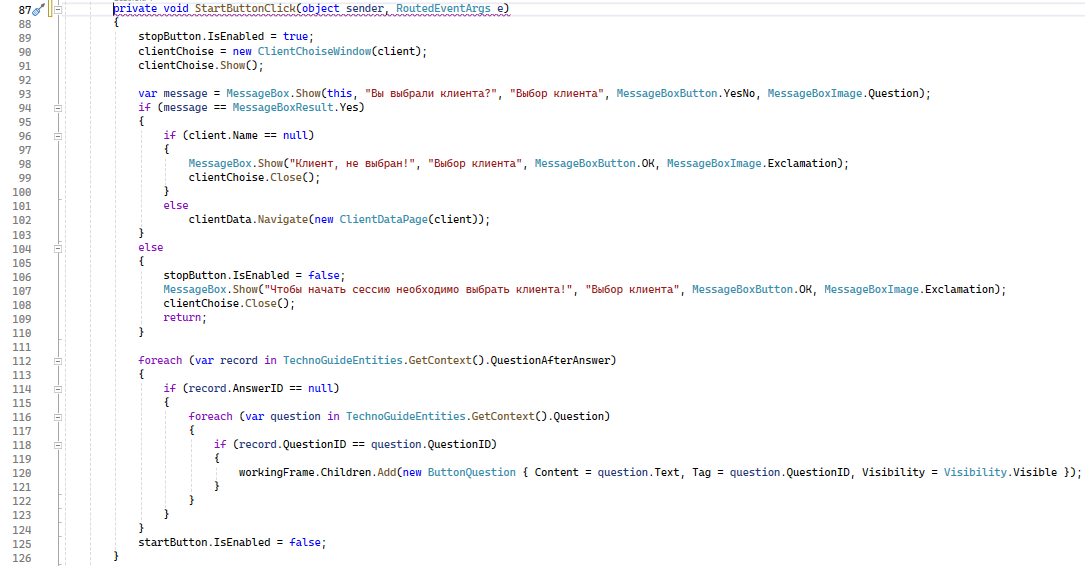


Рисунок 17. Код метода StartButtonClick

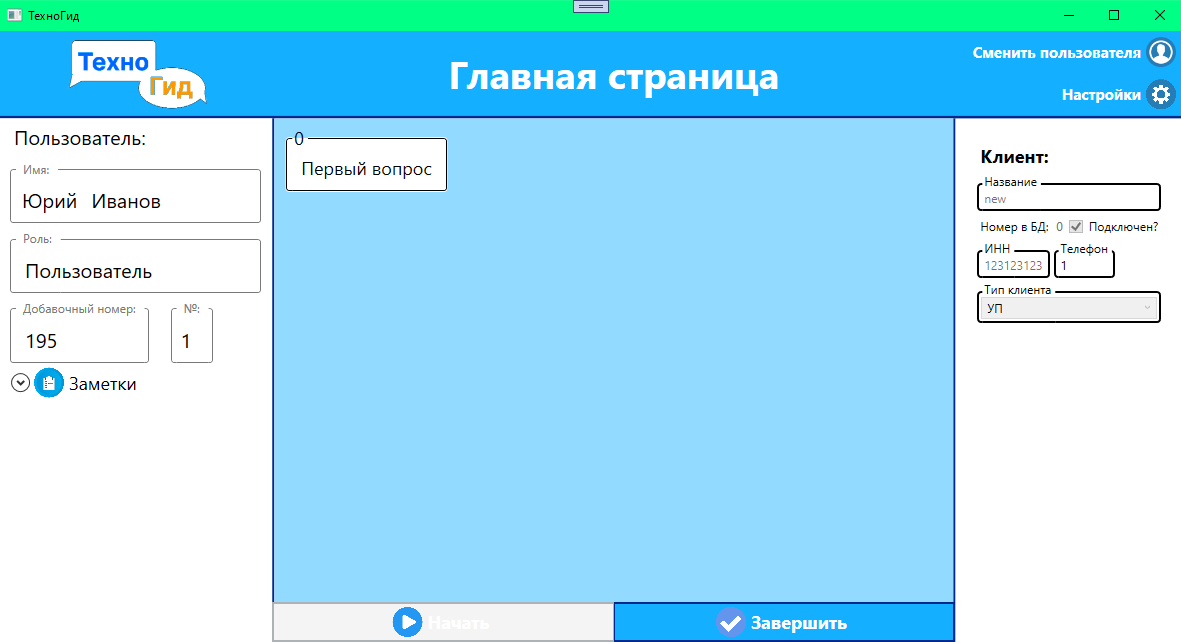


Рисунок 18. Главная страница пользователя после начала рабочей сессии

При нажатии на созданную после начала рабочей сессии кнопку запускается событие «Click» выполняющее метод «ButtonQuestionClick» (рисунок 19). Данный метод выполняет поиск объектов в таблице «AnswerAfterQuestion» по свойству «QuestionID» и при совпадении данного свойства у записи таблицы и кнопки создаётся объект класса «ButtonAnswer» аналогично «ButtonQuestion». Эти объекты представляют из себя варианты ответа на выбранный вопрос.

При выборе варианта срабатывает событие «Click» выполняющее метод «ButtonAnswerClick» (рисунок 18), кнопка данного ответа добавляется в последовательность вопросов и ответов аналогично вопросам и запускает поиск вопроса (аналогично «ButtonQuestion»).



Рисунок 19. Код методов ButtonQuestionClick и ButtonAnswerClick

После завершения сессии запускается событие «Click», выполняющее метод «StopButtonClick». Данный метод выводит окно с предложением сохранить данное обращение клиента в базу данных (рисунок 20). Текст обращения формируется из последовательности вопросов и ответов.

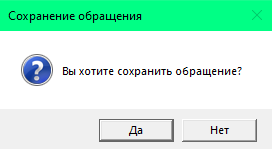


Рисунок 20. Сохранение обращения

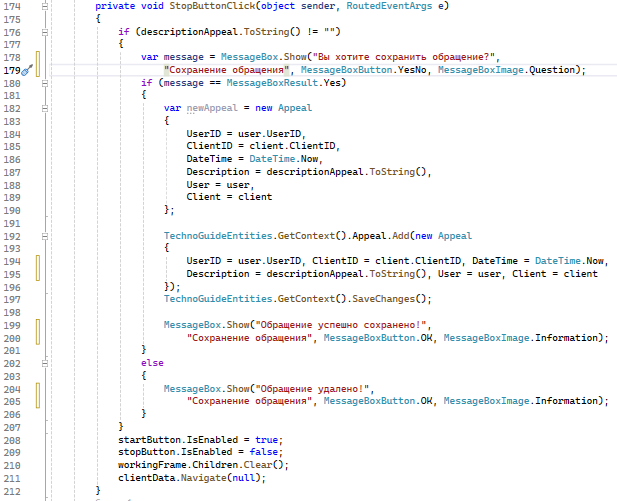


Рисунок 21. Код StopButtonClick

Кнопка «Настройки» (рисунок 3) так же доступна всем пользователям. При нажатии срабатывает событие «Click», выполняющее метод «SettingsButtonClick» (рисунок 22). Данный метод запускает конструктор для объекта класса «SettingsWindow» - окна настроек (рисунок 23, 24, 25), передавая пользователя в качестве аргумента.

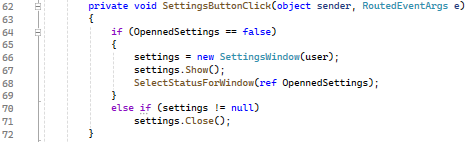


Рисунок 22. Код метода SettingsButtonClick

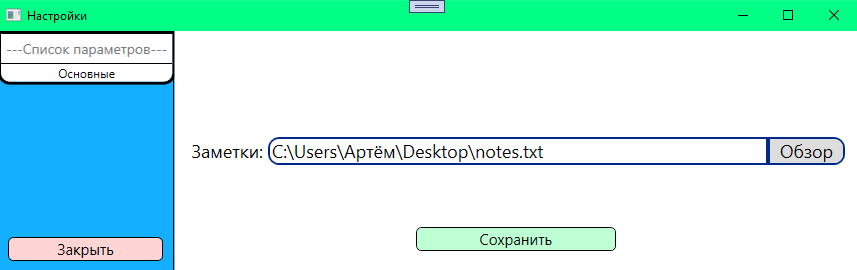


Рисунок 23. Окно настроек пользователя

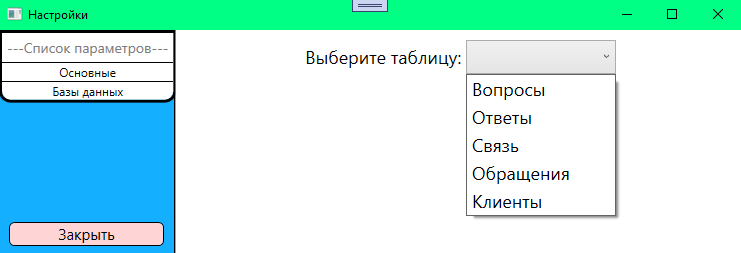


Рисунок 24. Окно настроек редактора

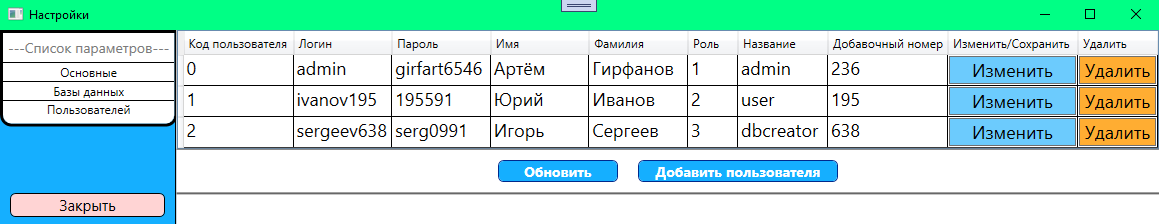


Рисунок 25. Окно настроек администратора

Доступ к функционалу окна настроек разделен на 3 уровня:

1. Уровень пользователя – возможность изменять только путь к файлу с заметками.
2. Уровень редактора – возможность редактировать базу данных, добавлять, изменять и удалять записи таблиц, представленных на рисунке 24.
3. Уровень администратора – возможность добавлять, изменять и удалять учетные записи пользователей.
   1. Руководство программиста

* OS Windows 8 - 11.
* Процессор: Intel Pentium Dual Core E6500K 2.93Ghz / AMD Athlon 64 X2 Dual Core 6400+
* Оперативная память: 2 - 6 GB (зависит от объема базы данных).
* Место на диске: 128 МB.
* Видеокарта: достаточно встроенной видеопамяти процессора.
  1. Руководство пользователя
* Главная страница (рисунок 5):

Главная страница имеет 4 поля:

1. Данные пользователя – включает в себя информацию о текущем пользователе приложения и «Заметки» – текстовый редактор.
2. Поле обработки обращений – содержит последовательность вопросов и ответов, выбранных пользователем, а также кнопки начала и завершения рабочей сессии
3. Данные клиента – содержит данные о выбранном при начале рабочей сессии клиенте.
4. Главная панель приложения – включает в себя 2 кнопки, «Настройки» и «Смена пользователя»

* Заметки (рисунок 6):

Включают в себя панель редактирования шрифта, поле для записи и кнопку «Записать» сохраняющую записи в файл, путь которого выбран в настройках (Настройки с=> Основные).

* Настройки (рисунок 23-25):

Содержимое окна настроек зависит от роли пользователя.

*Окно настроек пользователя* (рисунок 23)*:*

* + - Содержит путь к файлу заметок.

*Окно настроек редактора* (рисунок 24)*:*

* + - Содержит путь к файлу заметок;
    - Содержит страницу редактирования таблиц базы данных.

*Окно настроек администратора (*рисунок 25)*:*

* + - Содержит путь к файлу заметок;
    - Содержит страницу редактирования таблиц базы данных.
    - Содержит страницу редактирования учетных записей пользователей.
* Страница редактирования таблиц базы данных:

Имеет поле выбора таблицы для редактирования:

* + - Вопросы – список вопросов кнопки для обновления списка и добавления/изменения записей;
    - Ответы – список ответов, кнопки для обновления списка и добавления/изменения записей;
    - Связь – список связей между вопросами и ответами, таблицы вопросов и ответов, поле для создания связей (1 к 1);
    - Обращения – список обращений, кнопки для обновления списка и добавления/изменения записей;
    - Клиенты – список клиентов, кнопки для обновления списка и добавления/изменения записей.

*Примечание:*

Для создания записей в таблицу связей для вызова вопросов необходимо выбрать 1 вопрос и ответы для его вызова при выборе варианта. Ответ, выбранный для создания связи не должен вызывать другой вопрос, иначе это вызовет ошибку.

Для создания записей в таблицу связей для вызова ответов необходимо выбрать вопросы и ответы (количество не ограничено). При создании связи будет учитываться номер в последовательности выбранных для создания связи вопросов и ответов (первый вопрос свяжется с первым ответом, второй со вторым и тд.).

Для того, чтобы удалить выбранный вопрос из поля создания связи необходимо выбрать его в поле, нажав на него, и нажать на кнопку «**<<**».

Для того, чтобы удалить выбранный ответ из поля создания связи необходимо выбрать его в поле, нажав на него, и нажать на кнопку «**>>**».

Для того чтобы очистить список выбранных вопросов и ответов для создания связи нажмите на кнопку «**Х**» под списком.

Заключение

Благодаря активному развитию технологий в современном мире вырос спрос на средства, предназначенные упростить решение повседневных задач и сэкономить время. «Техно Гид» позволяет достичь этого путем сокращения в несколько раз траты временного ресурса старших и опытных сотрудников для обучения начинающих.

Цель курсовой работы, разработка АИС «Техно Гид» для технической поддержки, была достигнута.

Разработаны алгоритмы функционала приложения, получение / добавление / изменение информации из базы данных, взаимодействие ролей пользователей и их ограничений с АИС, создан дизайн форм окон и страниц АИС, успешно интегрированы функции данного приложения и настроена связь между формами и алгоритмами АИС.

Список литературы

1. ГОСТ 19.504-79 ЕСПД. Руководство программиста.

2. ГОСТ 19.505-79 ЕСПД. Руководство оператора.

3. Джереми Гибсона Бонда «‎Unity и С#. Геймдев от идеи до реализации» ‎. - 2-е изд. - СПб: Питер, 2022. - 928 с.

4. Майк Гейг «‎Разработка игр на Unity 2018 за 24 часа». - 1-е изд. - М: БОМБОРА, 2022. - 464 с.

5. Денисов Д.В. «‎Разработка игры в Unity. С нуля до реализации» ‎. - 1-е изд. - М: ЛитРес: Самиздат, 2021. - 227 с.

6. Харрисон, Ферроне Изучаем C# через разработку игр на Unity. / Ферроне Харрисон. – Спб.: ООО Издательство «Питер», 2022. – 401 с.

7. Ларкович, С.Н. Справочник Unity / С. Н. Ларкович — 1 изд. — Спб.: Наукаитехника, 2020—288c.

8. Албахари Д., Албахари Б. C# 8.0. Карманный справочник/ Албахари Д., Албахари Б. — 2 изд. — Спб.: Диалектика, 2020 — 240 c.

9. Корнилов А. Unity полное руководство / Корнилов А. — 2 изд. — Спб.: Наука и техника, 2021 — 496 c.

10. Ларкович С. Н., Евдокимов П. В. C# для UNITY-разработчиков. Практическое руководство по созданию игр / Ларкович С. Н., Евдокимов П. В. — 1 изд. — Спб.: Наука и техника, 2023 — 368 c.

11. Ферроне Харрисон Изучаем C# через разработку игр на Unity. / Ферроне Харрисон — 5-изд. — Спб.: Питер, 2022 — 400 c.

12. Торн Алан Искусство создания сценариев в Unity/ Торн Алан — 1-изд. — М.: ДМК Пресс, 2019 — 360 c.

Приложение

Код страницы *MainWindow*

using System.Windows;

namespace SpeechHelper

{

public partial class MainWindow : Window

{

private static User user;

public MainWindow()

{

InitializeComponent();

}

public MainWindow(User user)

{

InitializeComponent();

inputLogin.Text = user.Login;

}

private void LoginBtnClick(object sender, RoutedEventArgs e)

{

foreach (var user in TechnoGuideEntities.GetContext().User)

{

if (user.Login == inputLogin.Text)

{

if (user.Password == inputPassword.Password)

{

MainWindow.user = user;

var workingWindow = new WorkingWindow(MainWindow.user);

workingWindow.Show();

Close();

return;

}

else

{

MessageBox.Show("Неверный пароль!",

"Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

}

}

if (user == null)

MessageBox.Show("Пользователя с таким логином не существует!",

"Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

private void ExitBtnClick(object sender, RoutedEventArgs e)

{

Close();

}

}

}

Код страницы *WorkingWindow*

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Text;

using System.Windows;

using System.Windows.Controls;

namespace SpeechHelper

{

public partial class WorkingWindow : Window

{

public static bool OpennedSettings = false;

private static User user;

public static Client client = new Client();

private static NotesPage notes = new NotesPage();

private static SettingsWindow settings;

private static ClientChoiseWindow clientChoise;

private StringBuilder descriptionAppeal = new StringBuilder();

List<ButtonAnswer> answers = new List<ButtonAnswer>();

public WorkingWindow(User user)

{

InitializeComponent();

WorkingWindow.user = user;

if (user.RoleID == 1)

{

startButton.Visibility = Visibility.Collapsed;

stopButton.Visibility = Visibility.Collapsed;

userRole.Content = "Администратор";

}

else if (user.RoleID == 2)

userRole.Content = "Пользователь";

else

userRole.Content = "Редактор";

userData.DataContext = user;

}

public static void SelectStatusForWindow(ref bool status)

{

if (status == true)

status = false;

else

status = true;

}

public void SetClient(Client client)

{

WorkingWindow.client = client;

}

private void SettingsButtonClick(object sender, RoutedEventArgs e)

{

if (OpennedSettings == false)

{

settings = new SettingsWindow(user);

settings.Show();

SelectStatusForWindow(ref OpennedSettings);

}

else if (settings != null)

settings.Close();

}

private void WorkingWindowClosing(object sender, CancelEventArgs e)

{

try

{

if (OpennedSettings != false)

settings.Close();

}

catch (NullReferenceException)

{

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

}

}

private void StartButtonClick(object sender, RoutedEventArgs e)

{

stopButton.IsEnabled = true;

clientChoise = new ClientChoiseWindow(client);

clientChoise.Show();

var message = MessageBox.Show(this, "Вы выбрали клиента?", "Выбор клиента", MessageBoxButton.YesNo, MessageBoxImage.Question);

if (message == MessageBoxResult.Yes)

{

if (client.Name == null)

{

MessageBox.Show("Клиент, не выбран!", "Выбор клиента", MessageBoxButton.OK, MessageBoxImage.Exclamation);

clientChoise.Close();

}

else

clientData.Navigate(new ClientDataPage(client));

}

else

{

stopButton.IsEnabled = false;

MessageBox.Show("Чтобы начать сессию необходимо выбрать клиента!", "Выбор клиента", MessageBoxButton.OK, MessageBoxImage.Exclamation);

clientChoise.Close();

return;

}

foreach (var record in TechnoGuideEntities.GetContext().QuestionAfterAnswer)

{

if (record.AnswerID == null)

{

foreach (var question in TechnoGuideEntities.GetContext().Question)

{

if (record.QuestionID == question.QuestionID)

{

workingFrame.Children.Add(new ButtonQuestion { Content = question.Text, Tag = question.QuestionID, Visibility = Visibility.Visible });

}

}

}

}

startButton.IsEnabled = false;

}

private void ButtonQuestionClick(object sender, RoutedEventArgs e)

{

(sender as Button).IsEnabled = false;

descriptionAppeal.Append((sender as ButtonQuestion).Content + "\n");

foreach (var record in TechnoGuideEntities.GetContext().AnswerAfterQuestion)

{

if (record.QuestionID == Convert.ToInt32((sender as Button).Tag))

{

foreach (var answer in TechnoGuideEntities.GetContext().Answer)

{

if (record.AnswerID == answer.AnswerID)

answers.Add(new ButtonAnswer

{ Tag = answer.AnswerID, Content = answer.Text, Visibility = Visibility.Visible });

}

}

}

for (int i = 0; i < answers.Count; i++)

answerChoise.Children.Add(answers[i]);

}

private void ButtonAnswerClick(object sender, RoutedEventArgs e)

{

answerChoise.Children.Clear();

workingFrame.Children.Add(sender as ButtonAnswer);

(sender as Button).IsEnabled = false;

descriptionAppeal.Append((sender as ButtonAnswer).Content + "\n");

foreach (var record in TechnoGuideEntities.GetContext().QuestionAfterAnswer)

{

if (record.AnswerID == Convert.ToInt32((sender as ButtonAnswer).Tag))

{

foreach (var question in TechnoGuideEntities.GetContext().Question)

{

if (record.QuestionID == question.QuestionID)

workingFrame.Children.Add(new ButtonQuestion

{ Tag = question.QuestionID, Content = question.Text, Visibility = Visibility.Visible });

}

}

}

answers.Clear();

}

private void StopButtonClick(object sender, RoutedEventArgs e)

{

if (descriptionAppeal.ToString() != "")

{

var message = MessageBox.Show("Вы хотите сохранить обращение?",

"Сохранение обращения", MessageBoxButton.YesNo, MessageBoxImage.Question);

if (message == MessageBoxResult.Yes)

{

var newAppeal = new Appeal

{

UserID = user.UserID,

ClientID = client.ClientID,

DateTime = DateTime.Now,

Description = descriptionAppeal.ToString(),

User = user,

Client = client

};

TechnoGuideEntities.GetContext().Appeal.Add(new Appeal

{

UserID = user.UserID, ClientID = client.ClientID, DateTime = DateTime.Now,

Description = descriptionAppeal.ToString(), User = user, Client = client

});

TechnoGuideEntities.GetContext().SaveChanges();

MessageBox.Show("Обращение успешно сохранено!",

"Сохранение обращения", MessageBoxButton.OK, MessageBoxImage.Information);

}

else

{

MessageBox.Show("Обращение удалено!",

"Сохранение обращения", MessageBoxButton.OK, MessageBoxImage.Information);

}

}

startButton.IsEnabled = true;

stopButton.IsEnabled = false;

workingFrame.Children.Clear();

clientData.Navigate(null);

}

private void ChangeUserClick(object sender, RoutedEventArgs e)

{

var main = new MainWindow(user);

main.Show();

Close();

}

private void NotesExpanded(object sender, RoutedEventArgs e)

{

notesFrame.Navigate(notes);

notes.WriteToNotes();

}

}

}

Код страницы *NotesPage*

using SpeechHelper.Properties;

using System;

using System.IO;

using System.Windows;

using System.Windows.Controls;

namespace SpeechHelper

{

public partial class NotesPage : Page, IWorkWithFileInterface

{

private readonly double[] FontSizeCollection = { 8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 28, 36, 48, 72 }; // Массив часто используемых размеров шрифта.

public NotesPage()

{

InitializeComponent();

notesText.FontSize = Settings.Default.FontSizeNotes; // Задает последний установленный размер шрифта.

}

private void ShowFontSize()

{

try

{

fontSizeTB.Text = Convert.ToString(notesText.FontSize);

Settings.Default.FontSizeNotes = notesText.FontSize;

Settings.Default.Save();

}

catch (FormatException)

{

Settings.Default.FontSizeNotes = 14;

Settings.Default.Save();

fontSizeTB.Text = Convert.ToString(Settings.Default.FontSizeNotes);

}

}

public void WriteToFile(string adress, string text)

{

try

{

using (FileStream file = new FileStream(adress, FileMode.Open))

{

file.Close();

using (StreamWriter streamWriter = new StreamWriter(adress))

{

streamWriter.WriteLine(text);

}

}

}

catch (IOException)

{

MessageBox.Show("При записи в файл произошла ошибка! Закройте файл и повторите попытку!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return;

}

}

public string ReadToFile(string adress)

{

try

{

using (FileStream file = new FileStream(adress, FileMode.Append))

{

file.Close();

using (StreamReader streamReader = new StreamReader(adress))

{

return streamReader.ReadToEnd();

}

}

}

catch (DirectoryNotFoundException)

{

MessageBox.Show("Не удалось найти/создать файл!",

"Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return null;

}

catch (UnauthorizedAccessException)

{

MessageBox.Show("У вас нет прав доступа к файлу!\n" +

"Перейдите в Настройки -> Основные, чтобы указать путь к файлу.",

"Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

return null;

}

}

public void WriteToNotes()

{

notesText.Text = ReadToFile(Settings.Default.PathToFileWithNotes);

}

private void WriteToFileBtnClick(object sender, RoutedEventArgs e)

{

WriteToFile((Settings.Default.PathToFileWithNotes), notesText.Text);

}

private void FontUpBtnClick(object sender, RoutedEventArgs e)

{

try

{

if (notesText.FontSize < 8)

{

notesText.FontSize++;

ShowFontSize();

}

else if (notesText.FontSize >= 72)

{

notesText.FontSize += 2;

ShowFontSize();

}

else

{

for (int i = 0; i < FontSizeCollection.Length; i++)

{

if (notesText.FontSize >= FontSizeCollection[i])

continue;

else

{

notesText.FontSize = FontSizeCollection[i];

ShowFontSize();

break;

}

}

}

}

catch (ArgumentException)

{

MessageBox.Show("Превышено МИНИМАЛЬНОЕ/МАКСИМАЛЬНОЕ допустимое значение размера шрифта!",

"Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

ShowFontSize();

}

catch (Exception ex)

{

MessageBox.Show("Критическая ошибка!\n" + ex.ToString(),

"Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

private void FontDownBtnClick(object sender, RoutedEventArgs e)

{

try

{

if ((notesText.FontSize <= 9) && (notesText.FontSize > 0))

{

notesText.FontSize--;

ShowFontSize();

}

else

{

for (int i = FontSizeCollection.Length - 1; i > 0; i--)

{

if (notesText.FontSize <= FontSizeCollection[i])

continue;

else

{

notesText.FontSize = FontSizeCollection[i];

ShowFontSize();

break;

}

}

}

}

catch (ArgumentException)

{

MessageBox.Show("Превышено МИНИМАЛЬНОЕ/МАКСИМАЛЬНОЕ допустимое значение размера шрифта!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

ShowFontSize();

}

}

private void FontSizeDeleteBtnClick(object sender, RoutedEventArgs e)

{

fontSizeTB.Text = "";

}

private void FontSizeApplyBtnClick(object sender, RoutedEventArgs e)

{

try

{

if (fontSizeTB.Text == "")

fontSizeTB.Text = Convert.ToString(notesText.FontSize);

else if ((Convert.ToDouble(fontSizeTB.Text) >= 1) && (Convert.ToDouble(fontSizeTB.Text) <= 256))

{

notesText.FontSize = Convert.ToDouble(fontSizeTB.Text);

ShowFontSize();

}

else

{

MessageBox.Show("Ошибка ввода! Укажите размер шрифта в диапазоне от 1 до 256.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

return;

}

}

catch (FormatException)

{

MessageBox.Show("Ошибка ввода! Используйте цифры для указания размера шрифта.", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

ShowFontSize();

return;

}

catch (ArgumentException)

{

MessageBox.Show("Превышено МИНИМАЛЬНОЕ/МАКСИМАЛЬНОЕ допустимое значение размера шрифта!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Warning);

ShowFontSize();

}

catch (Exception)

{

MessageBox.Show("Критическая ошибка!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

}

}

}

Код страницы *ClientDataPage*

using System.Linq;

using System.Windows.Controls;

namespace SpeechHelper

{

public partial class ClientDataPage : Page

{

public ClientDataPage(Client client)

{

InitializeComponent();

if (client == null)

{

client = new Client();

}

else

{

clientId.IsEnabled = false;

clientInn.IsEnabled = false;

clientLinked.IsEnabled = false;

clientName.IsEnabled = false;

clientType.IsEnabled = false;

}

clientType.ItemsSource = TechnoGuideEntities.GetContext().TypeOfClient.ToList();

DataContext = client;

if (client.Linked == true)

clientLinked.IsChecked = true;

else

clientLinked.IsChecked = false;

}

}

}