**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1

по дисциплине: «Объектно-ориентированное проектирование и программирование»

на тему: **«**Работа с файловой системой. Потоки ввода/вывода**»**

Выполнил: студент гр. ИТИ-21

Полторан М. Д.

Принял: преподаватель

Башаримов Ю. С.

Гомель 2024

**Цель работы**: изучить основы *XML*, получить навыки работы с *XML*

документами средствами .*NET* (с использованием *DOM* дерева или *SAX* парсера согласно варианту), освоить средства валидации *XML* документов с применением *XSD*.

**Задание:**

1.Разработать структуру *XML* документа и соответствующую ему *XSD* схему. В *XSD* обязательно задавать типы данных, которые принимают узлы, и указывать пределы допустимых значений. В случае если документ не соответствует схеме, выводить соответствующее сообщение. Обосновать выбранный вид схемы *XML*.

2. Выделенные курсивом данные оформить как перечисление.

3. Разработать библиотеку классов для работы с данными в формате *XML* согласно варианту. Для работы с *XML* использовать *DOM* или *SAX* parser (*XML*-сериализацию или *LINQ2XML* использовать нельзя!)

4. Разработать *WPF*-приложение, поддерживающее работу с данными в формате *XML* (чтение, отображение, редактирование и сохранение) согласно варианту.

5. Приложение должно быть простым в использовании и включать в себя полную обработку исключений.

6. При написании и оформлении кода обязательно руководствоваться *Code* *Convention*, принципами ООП, *SOLID* и использовать элементы авто документирования с генерацией соответствующих файлов.

7. Написать *unit*-тесты и исходные файлы (не менее 30 записей) для тестирования разработанных библиотечных классов, тестирование должно покрывать более 80*%* библиотечного кода.

**Индивидуальное задание**: Дневник таксиста: номер автомобиля, марка, координаты начального пункта и пункта назначения, расстояние, дата и время, расход бензина, тариф.

**Ход работы**

На рисунке 1 изображено пользовательское меню в *WPF* приложении.

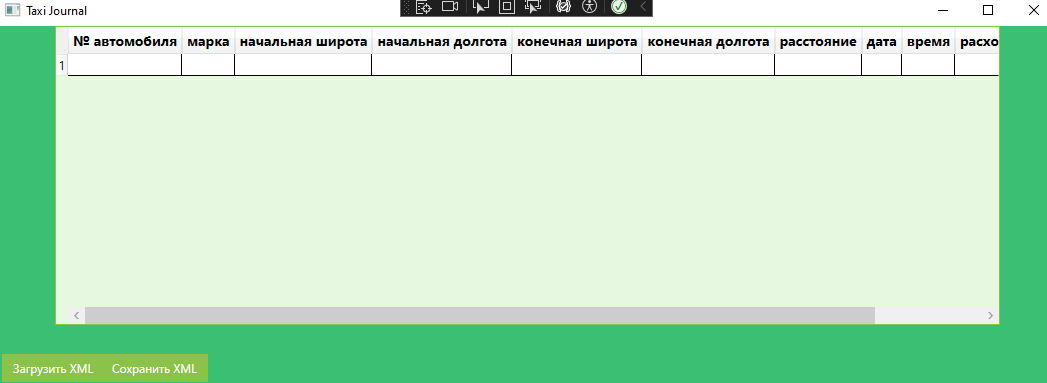


Рисунок 1 – Пользовательское меню

На рисунке 1 можно увидеть, что у нас есть пустая таблица. При нажатии на кнопку Загрузить *XML* пользователю откроется окно, где он может выбрать свой файл.

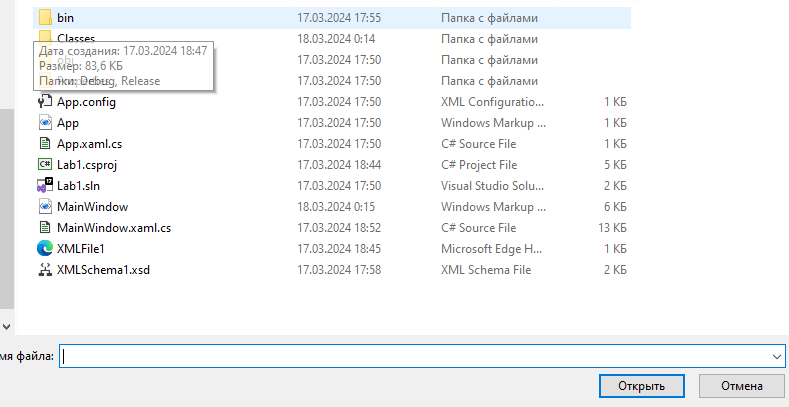


Рисунок 2 – выбор файла

На рисунке 2 мы можем увидеть, что пользователю нужно выбрать *XML*-файл для того, чтобы начать его редактирование.

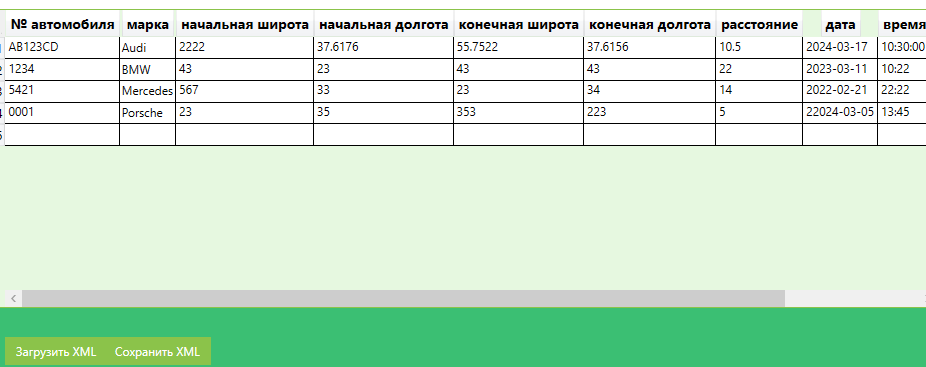


Рисунок 3 – Открытие файла

На рисунке 3 мы можем увидеть, что при открытии файла XMLFile1, в нашу таблицу заносятся данные, которые мы можем редактировать, удалять и добавлять новые данные.

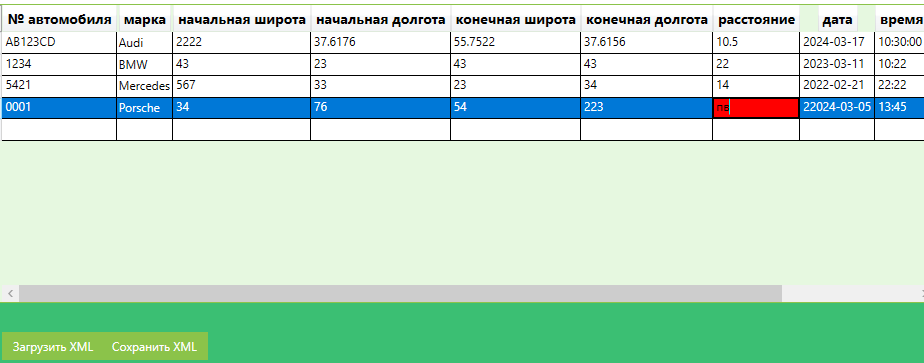


Рисунок 4 – Редактирование данных

На рисунке 4 можно увидеть, что при неправильно введенных данных, ячейка в таблице будет подсвечена красным.

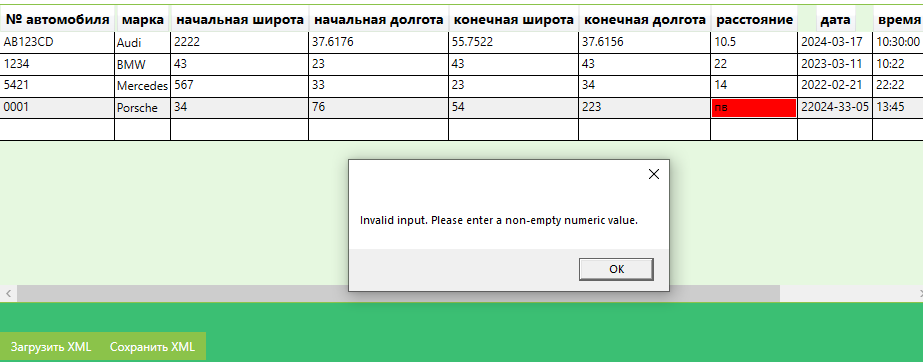


Рисунок 5 – Демонстрация ошибки

На рисунке 5 мы можем увидеть, что при неправильно введённых данных у нас выводится диалоговое окно с предупреждением об ошибке.

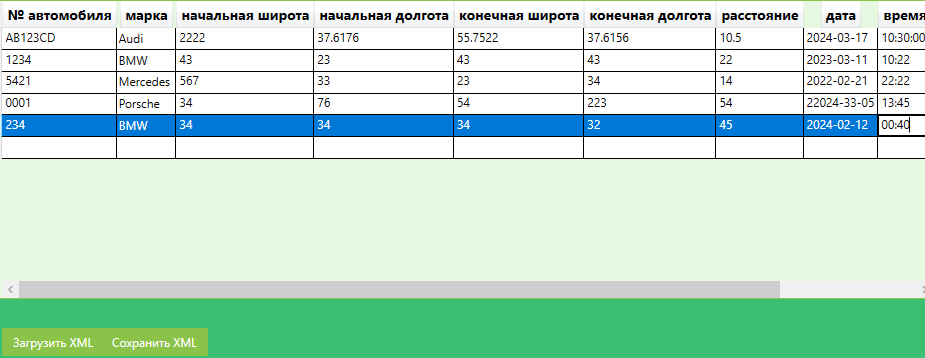


Рисунок 6 – Добавление «Записи»



Рисунок 7 – Успешное добавление записи

На рисунке 7 можно увидеть, что запись была добавлена успешно.

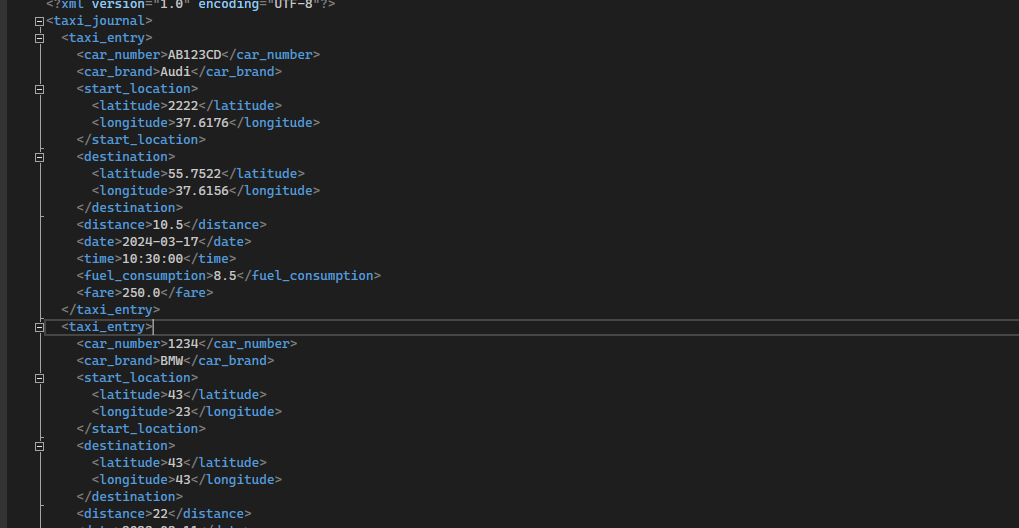


Рисунок 8 – Файл до добавления

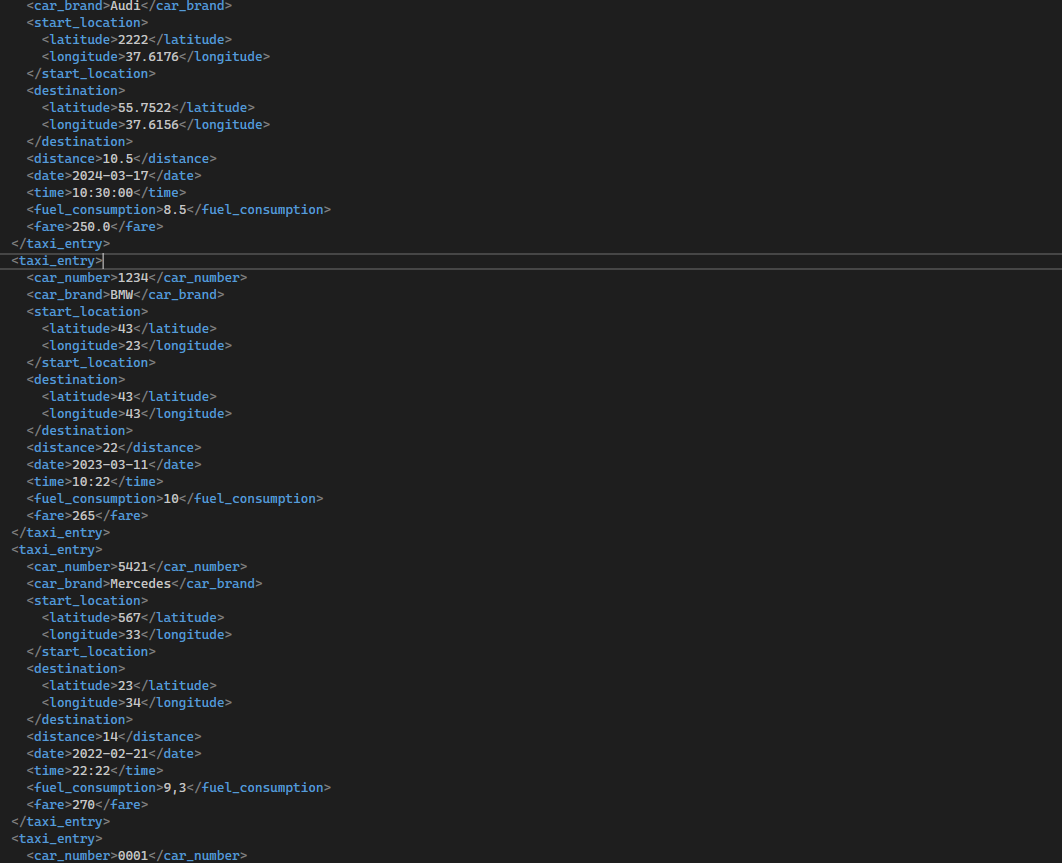


Рисунок 9 – Файл после добавления

На рисунке 9 мы можем увидеть, что при нажатии на кнопку Сохранить XML, наши данные в файле будут обновлены.

На рисунке 10 изображено возможный выбор нескольких ячеек сразу при помощи удерживания *ctrl* либо при удерживании лкм.



Рисунок 10 – выбор нескольких записей



Рисунок 11 – Удаление

При нажатии на кнопку *delete* на клавиатуре, записи удаляются.

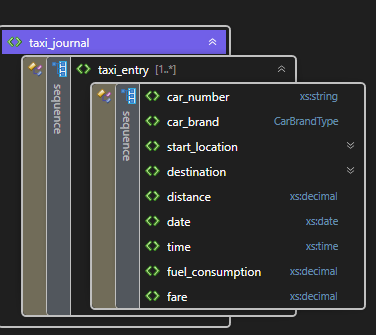
. 

Рисунок 14 – Создание *XSD*

На рисунке 14 можно увидеть созданную *XSD* схему.

В приложении А представлен код программы. В приложении Б представлен *XAML* код разметки.

**Вывод:** в результате выполнения лабораторной познакомились с методами и механизмами инкапсуляции и организации доступа к элементам объекта.

**ПРИЛОЖЕНИЕ А**

(обязательное)

**Текст программы**

***MainWindowXaml.cs***

**using Microsoft.Win32;**

**using System;**

**using System.Collections.Generic;**

**using System.IO;**

**using System.Windows;**

**using System.Windows.Controls;**

**using System.Windows.Media;**

**using System.Xml;**

**namespace Lab1**

**{**

**public partial class MainWindow : Window**

**{**

**private List<TaxiEntry> taxiEntries;**

**private string currentFilePath; // Track the current XML file path**

**public MainWindow()**

**{**

**InitializeComponent();**

**taxiEntries = new List<TaxiEntry>();**

**dataGrid.ItemsSource = taxiEntries;**

**// Subscribe to events**

**dataGrid.CellEditEnding += DataGrid\_CellEditEnding;**

**dataGrid.LoadingRow += DataGrid\_LoadingRow;**

**}**

**private void LoadXml\_Click(object sender, RoutedEventArgs e)**

**{**

**OpenFileDialog openFileDialog = new OpenFileDialog();**

**if (openFileDialog.ShowDialog() == true)**

**{**

**try**

**{**

**// Чтение данных из XML**

**XmlDocument xmlDoc = new XmlDocument();**

**xmlDoc.Load(openFileDialog.FileName);**

**XmlNodeList entryNodes = xmlDoc.GetElementsByTagName("taxi\_entry");**

**foreach (XmlNode entryNode in entryNodes)**

**{**

**TaxiEntry entry = new TaxiEntry();**

**entry.CarNumber = entryNode.SelectSingleNode("car\_number").InnerText;**

**entry.CarBrand = entryNode.SelectSingleNode("car\_brand").InnerText;**

**entry.StartLatitude = entryNode.SelectSingleNode("start\_location/latitude").InnerText;**

**entry.StartLongitude = entryNode.SelectSingleNode("start\_location/longitude").InnerText;**

**entry.DestLatitude = entryNode.SelectSingleNode("destination/latitude").InnerText;**

**entry.DestLongitude = entryNode.SelectSingleNode("destination/longitude").InnerText;**

**entry.Distance = entryNode.SelectSingleNode("distance").InnerText;**

**entry.Date = entryNode.SelectSingleNode("date").InnerText;**

**entry.Time = entryNode.SelectSingleNode("time").InnerText;**

**entry.FuelConsumption = entryNode.SelectSingleNode("fuel\_consumption").InnerText;**

**entry.Fare = entryNode.SelectSingleNode("fare").InnerText;**

**taxiEntries.Add(entry);**

**}**

**dataGrid.Items.Refresh();**

**currentFilePath = openFileDialog.FileName; // Update the current file path**

**}**

**catch (Exception ex)**

**{**

**MessageBox.Show("Error loading XML: " + ex.Message);**

**}**

**}**

**}**

**private void SaveXml\_Click(object sender, RoutedEventArgs e)**

**{**

**if (string.IsNullOrEmpty(currentFilePath))**

**{**

**// If the current file path is not set, open a save dialog**

**SaveFileDialog saveFileDialog = new SaveFileDialog();**

**saveFileDialog.Filter = "XML files (\*.xml)|\*.xml";**

**if (saveFileDialog.ShowDialog() == true)**

**{**

**currentFilePath = saveFileDialog.FileName;**

**}**

**else**

**{**

**return; // User canceled the save operation**

**}**

**}**

**try**

**{**

**// Создание XML документа**

**XmlDocument xmlDoc = new XmlDocument();**

**XmlDeclaration xmlDeclaration = xmlDoc.CreateXmlDeclaration("1.0", "UTF-8", null);**

**XmlElement rootElement = xmlDoc.CreateElement("taxi\_journal");**

**xmlDoc.InsertBefore(xmlDeclaration, xmlDoc.DocumentElement);**

**xmlDoc.AppendChild(rootElement);**

**foreach (TaxiEntry entry in taxiEntries)**

**{**

**XmlElement entryElement = xmlDoc.CreateElement("taxi\_entry");**

**XmlElement carNumberElement = xmlDoc.CreateElement("car\_number");**

**carNumberElement.InnerText = entry.CarNumber;**

**entryElement.AppendChild(carNumberElement);**

**XmlElement carBrandElement = xmlDoc.CreateElement("car\_brand");**

**carBrandElement.InnerText = entry.CarBrand;**

**entryElement.AppendChild(carBrandElement);**

**XmlElement startLocationElement = xmlDoc.CreateElement("start\_location");**

**XmlElement startLatitudeElement = xmlDoc.CreateElement("latitude");**

**startLatitudeElement.InnerText = entry.StartLatitude;**

**startLocationElement.AppendChild(startLatitudeElement);**

**XmlElement startLongitudeElement = xmlDoc.CreateElement("longitude");**

**startLongitudeElement.InnerText = entry.StartLongitude;**

**startLocationElement.AppendChild(startLongitudeElement);**

**entryElement.AppendChild(startLocationElement);**

**XmlElement destinationElement = xmlDoc.CreateElement("destination");**

**XmlElement destLatitudeElement = xmlDoc.CreateElement("latitude");**

**destLatitudeElement.InnerText = entry.DestLatitude;**

**destinationElement.AppendChild(destLatitudeElement);**

**XmlElement destLongitudeElement = xmlDoc.CreateElement("longitude");**

**destLongitudeElement.InnerText = entry.DestLongitude;**

**destinationElement.AppendChild(destLongitudeElement);**

**entryElement.AppendChild(destinationElement);**

**XmlElement distanceElement = xmlDoc.CreateElement("distance");**

**distanceElement.InnerText = entry.Distance;**

**entryElement.AppendChild(distanceElement);**

**XmlElement dateElement = xmlDoc.CreateElement("date");**

**dateElement.InnerText = entry.Date;**

**entryElement.AppendChild(dateElement);**

**XmlElement timeElement = xmlDoc.CreateElement("time");**

**timeElement.InnerText = entry.Time;**

**entryElement.AppendChild(timeElement);**

**XmlElement fuelConsumptionElement = xmlDoc.CreateElement("fuel\_consumption");**

**fuelConsumptionElement.InnerText = entry.FuelConsumption;**

**entryElement.AppendChild(fuelConsumptionElement);**

**XmlElement fareElement = xmlDoc.CreateElement("fare");**

**fareElement.InnerText = entry.Fare;**

**entryElement.AppendChild(fareElement);**

**rootElement.AppendChild(entryElement);**

**}**

**// Сохранение XML документа**

**xmlDoc.Save(currentFilePath);**

**MessageBox.Show("XML saved successfully.");**

**}**

**catch (Exception ex)**

**{**

**MessageBox.Show("Error saving XML: " + ex.Message);**

**}**

**}**

**private void AddEntry\_Click(object sender, RoutedEventArgs e)**

**{**

**taxiEntries.Add(new TaxiEntry());**

**dataGrid.Items.Refresh();**

**}**

**private void DeleteEntry\_Click(object sender, RoutedEventArgs e)**

**{**

**if (dataGrid.SelectedItem != null)**

**{**

**taxiEntries.Remove((TaxiEntry)dataGrid.SelectedItem);**

**dataGrid.Items.Refresh();**

**}**

**}**

**private void DataGrid\_CellEditEnding(object sender, DataGridCellEditEndingEventArgs e)**

**{**

**// Update the edited value in the underlying list**

**var editedEntry = (TaxiEntry)e.Row.Item;**

**var column = e.Column as DataGridBoundColumn;**

**if (column != null)**

**{**

**var bindingPath = (column.Binding as System.Windows.Data.Binding).Path.Path;**

**var textBox = e.EditingElement as TextBox;**

**if (textBox != null)**

**{**

**var newValue = textBox.Text.Trim(); // Trim leading and trailing spaces**

**bool isValid = true;**

**if (string.IsNullOrEmpty(newValue))**

**{**

**isValid = false;**

**}**

**else**

**{**

**switch (bindingPath)**

**{**

**case "CarNumber":**

**editedEntry.CarNumber = newValue;**

**break;**

**case "CarBrand":**

**editedEntry.CarBrand = newValue;**

**break;**

**case "StartLatitude":**

**if (!double.TryParse(newValue, out double startLatitude))**

**isValid = false;**

**else**

**editedEntry.StartLatitude = startLatitude.ToString();**

**break;**

**case "StartLongitude":**

**if (!double.TryParse(newValue, out double startLongitude))**

**isValid = false;**

**else**

**editedEntry.StartLongitude = startLongitude.ToString();**

**break;**

**case "DestLatitude":**

**if (!double.TryParse(newValue, out double destLatitude))**

**isValid = false;**

**else**

**editedEntry.DestLatitude = destLatitude.ToString();**

**break;**

**case "DestLongitude":**

**if (!double.TryParse(newValue, out double destLongitude))**

**isValid = false;**

**else**

**editedEntry.DestLongitude = destLongitude.ToString();**

**break;**

**case "Distance":**

**if (!double.TryParse(newValue, out double distance))**

**isValid = false;**

**else**

**editedEntry.Distance = distance.ToString();**

**break;**

**case "Date":**

**editedEntry.Date = newValue;**

**break;**

**case "Time":**

**editedEntry.Time = newValue;**

**break;**

**case "FuelConsumption":**

**if (!double.TryParse(newValue, out double fuelConsumption))**

**isValid = false;**

**else**

**editedEntry.FuelConsumption = fuelConsumption.ToString();**

**break;**

**case "Fare":**

**if (!double.TryParse(newValue, out double fare))**

**isValid = false;**

**else**

**editedEntry.Fare = fare.ToString();**

**break;**

**}**

**}**

**// Highlight the cell in red if the data is invalid**

**if (!isValid)**

**{**

**textBox.Background = new SolidColorBrush(Colors.Red);**

**e.Cancel = true; // Cancel the edit operation**

**MessageBox.Show("Invalid input. Please enter a non-empty numeric value.");**

**}**

**else**

**{**

**textBox.ClearValue(TextBox.BackgroundProperty);**

**}**

**}**

**}**

**}**

**private void DataGrid\_LoadingRow(object sender, DataGridRowEventArgs e)**

**{**

**// Set the data grid row header to the row index**

**e.Row.Header = (e.Row.GetIndex() + 1).ToString();**

**}**

**}**

**}**

**Car.cs**

using System;

using System.IO;

using System.Xml.Serialization;

namespace Lab1

{

public class Car

{

public string CarNumber { get; set; }

public CarBrand CarBrand { get; set; } // Modified property name to match XAML binding

public double StartLatitude { get; set; }

public double StartLongitude { get; set; }

public double DestinationLatitude { get; set; }

public double DestinationLongitude { get; set; }

public double Distance { get; set; }

public DateTime DateTime { get; set; }

public double FuelConsumption { get; set; }

public double Fare { get; set; }

}

}

**CarBrand.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab1

{

public enum CarBrand

{

Audi,

BMW,

Mercedes,

Porsche

}

}

**TaxiEntry.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab1

{

public class TaxiEntry

{

public string CarNumber { get; set; }

public string CarBrand { get; set; }

public string StartLatitude { get; set; }

public string StartLongitude { get; set; }

public string DestLatitude { get; set; }

public string DestLongitude { get; set; }

public string Distance { get; set; }

public string Date { get; set; }

public string Time { get; set; }

public string FuelConsumption { get; set; }

public string Fare { get; set; }

}

}

**ПРИЛОЖЕНИЕ Б**

(обязательное)

XMLSchema1.xsd:

<?xml version="1.0" encoding="UTF-8"?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="taxi\_journal">

<xs:complexType>

<xs:sequence>

<xs:element name="taxi\_entry" maxOccurs="unbounded">

<xs:complexType>

<xs:sequence>

<xs:element name="car\_number" type="xs:string"/>

<xs:element name="car\_brand" type="CarBrandType"/>

<xs:element name="start\_location">

<xs:complexType>

<xs:sequence>

<xs:element name="latitude" type="xs:decimal"/>

<xs:element name="longitude" type="xs:decimal"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="destination">

<xs:complexType>

<xs:sequence>

<xs:element name="latitude" type="xs:decimal"/>

<xs:element name="longitude" type="xs:decimal"/>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:element name="distance" type="xs:decimal"/>

<xs:element name="date" type="xs:date"/>

<xs:element name="time" type="xs:time"/>

<xs:element name="fuel\_consumption" type="xs:decimal"/>

<xs:element name="fare" type="xs:decimal"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:sequence>

</xs:complexType>

</xs:element>

<xs:simpleType name="CarBrandType">

<xs:restriction base="xs:string">

<xs:enumeration value="Toyota"/>

<xs:enumeration value="Nissan"/>

<xs:enumeration value="Honda"/>

<!-- Добавьте другие марки автомобилей по вашему выбору -->

</xs:restriction>

</xs:simpleType>

</xs:schema>

**ПРИЛОЖЕНИЕ В**

(обязательное)

XMLFile1.Xml:

<?xml version="1.0" encoding="UTF-8"?>

<taxi\_journal>

<taxi\_entry>

<car\_number>424</car\_number>

<car\_brand>BMW</car\_brand>

<start\_location>

<latitude>43</latitude>

<longitude>34</longitude>

</start\_location>

<destination>

<latitude>34</latitude>

<longitude>34</longitude>

</destination>

<distance>10.5</distance>

<date>2024-03-17</date>

<time>10:30:00</time>

<fuel\_consumption>8.5</fuel\_consumption>

<fare>250.0</fare>

</taxi\_entry>

<taxi\_entry>

<car\_number>4334</car\_number>

<car\_brand>BMW</car\_brand>

<start\_location>

<latitude>43</latitude>

<longitude>43</longitude>

</start\_location>

<destination>

<latitude>4</latitude>

<longitude>4</longitude>

</destination>

<distance>546</distance>

<date>65</date>

<time>56</time>

<fuel\_consumption>4</fuel\_consumption>

<fare>44</fare>

</taxi\_entry>

</taxi\_journal>