Министерство образования Московской области

Государственное образовательное учреждение высшего образования Московской области

«Государственный гуманитарно-технологический университет»

**Ликино-Дулевский политехнический колледж – филиал ГГТУ**

**Отчёт по Учебной Практике № \_2\_**

по дисциплине

### «Разработка программных модулей в среде Visual Studio»

**Выполнил:**

Мелега Алексей,

студент(ка) группы

ИСП.20А 3 курса

09.02.07 Информационные системы и программирование

очной формы обучения

Проверил:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка

\_\_\_\_\_\_\_\_\_/\_\_ \_\_\_\_\_\_\_\_\_\_/

Ссылка github:

1)Постановка Задачи

Вариант №14

Необходимо написать программу, которая будет хранить информацию о абонентах сотового оператора. (например такую как: название тарифа, абонентская плата, сколько Гб интернета, фио клиента, паспорт).

Реализуйте следующий функционал:

* Добавление новой записи (телефон должно быть уникальным).
* Удаление записи по телефону (учитывать, что такой номер может отсутствовать в словаре).
* Вывести все записи по тарифам с объемом Гб < 10
* Вывести все записи о тарифах, где абонентская плата менее указанного значения (тарифы не должны повторяться).
* Вывести данные о клиенте: фио, паспорт, телефон по указанному тарифу
* Вывести количество абонентов на каждом тарифе.
* Вывести среднюю абонентскую по всем тарифам.
* Любая ваша функция.

**День 1.**

Разработать структуру базы данных в среде SQL SERVER по своей предметной области.

Сформировать данные для импорта в таблицах MS Excel и выполнить импорт.

Создать проект WPF в среде разработки MS Visual Studio, подключить базу данных к проекту через создание модели .edmx.

Спроектировать интерфейс приложения (XAML, Figma, draw.io и т.п.):

1. главная страница – название, логотип, переходы на остальные страницы;
2. страница для работы со справочниками;
3. страница для работы с основной информацией;
4. дополнительные страницы (при необходимости).



Рис.1 «Главное меню»

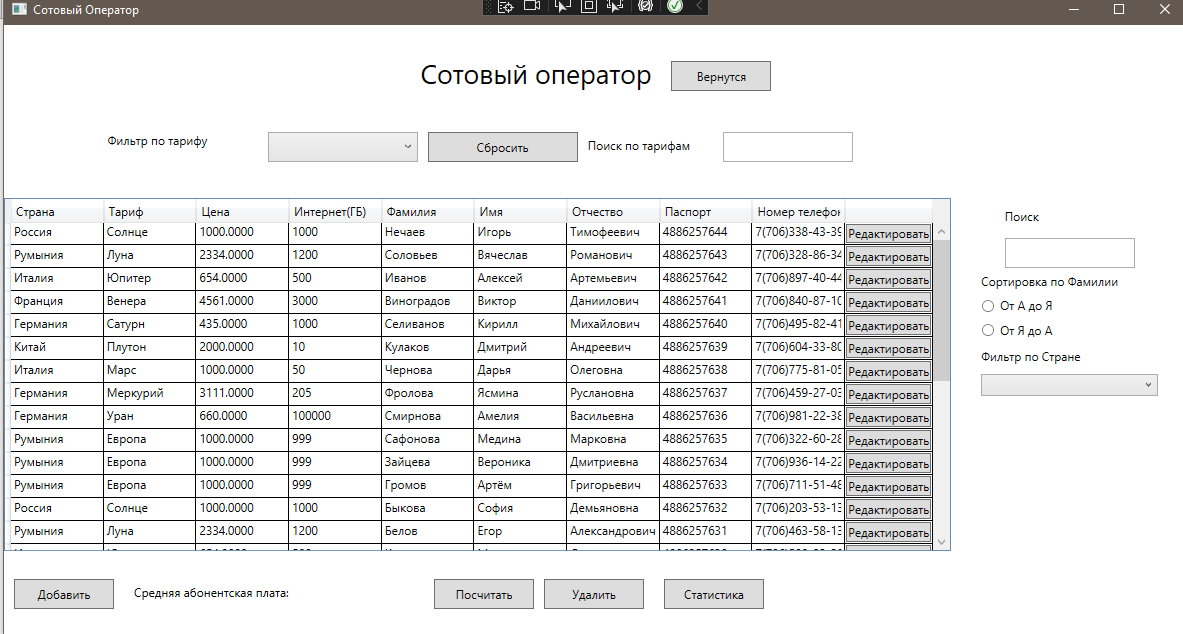


Рис.2 «Страница с основной таблицей»

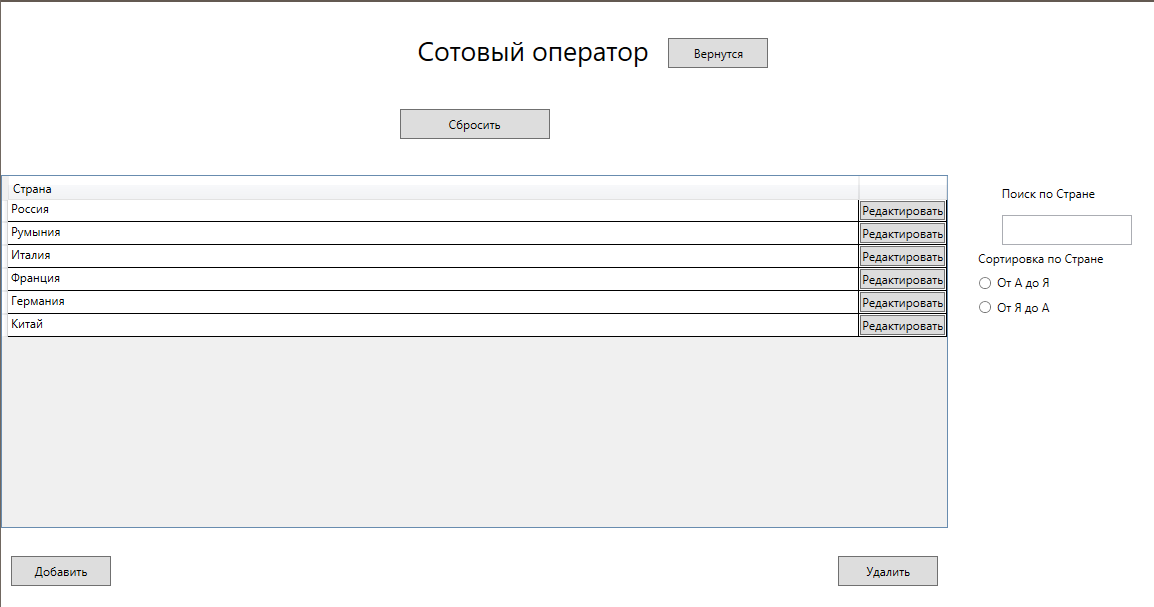


Рис.3 «Страница со справочной таблицей о странах»

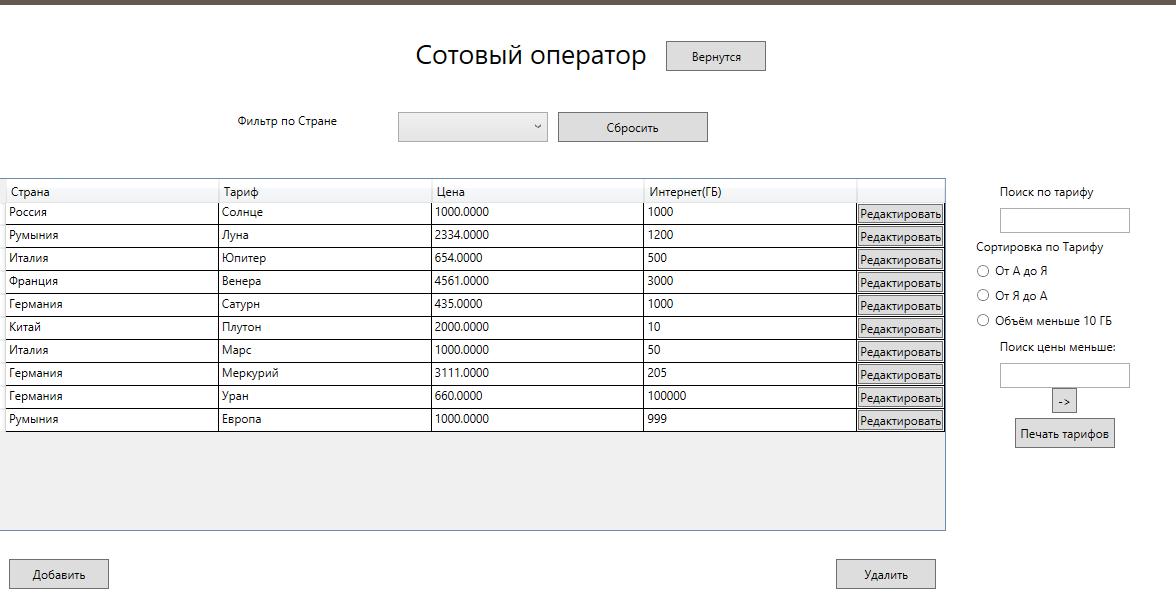


Рис.4 «Страница со справочной таблицей о тарифах»

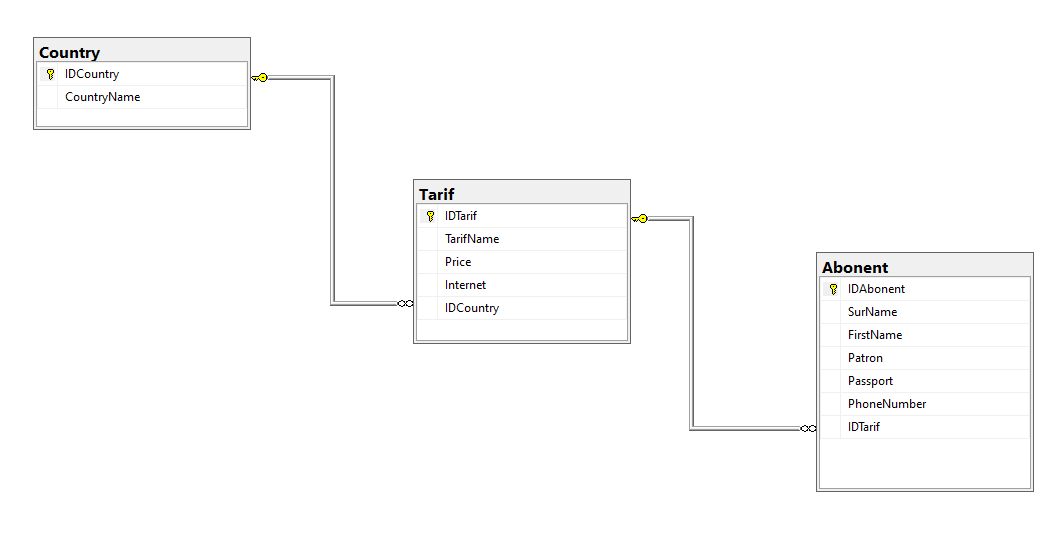


Рис.5 «Схема базы данных»

**День 2.**

Разработка главной страницы.

Реализация функций для работы со справочными таблицами: добавление, редактирование, просмотр, удаление данных.



Рис.6 «Главная страница»

Код C# главной страницы

using PhoneOperator.Classes;

using PhoneOperator.Pages;

namespace PhoneOperator.Pages

{

/// <summary>

/// Логика взаимодействия для MainMenu.xaml

/// </summary>

public partial class MainMenu : Page

{

public MainMenu()

{

InitializeComponent();

}

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

{

this.NavigationService.Navigate(new PageOperators());

}

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

{

this.NavigationService.Navigate(new TableCountry());

}

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

{

this.NavigationService.Navigate(new TableTarif());

}

}

}

Переходы на другие страницы по нажатию на кнопки

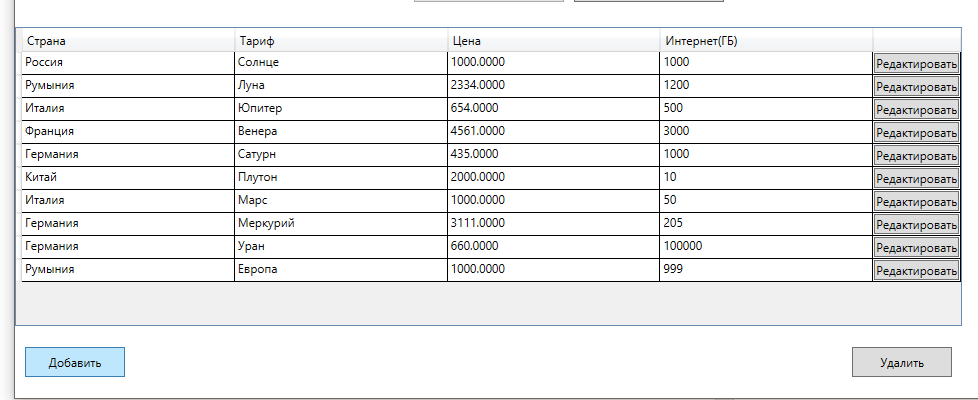


Рис.7 «Кнопка добавления на странице с тарифами»

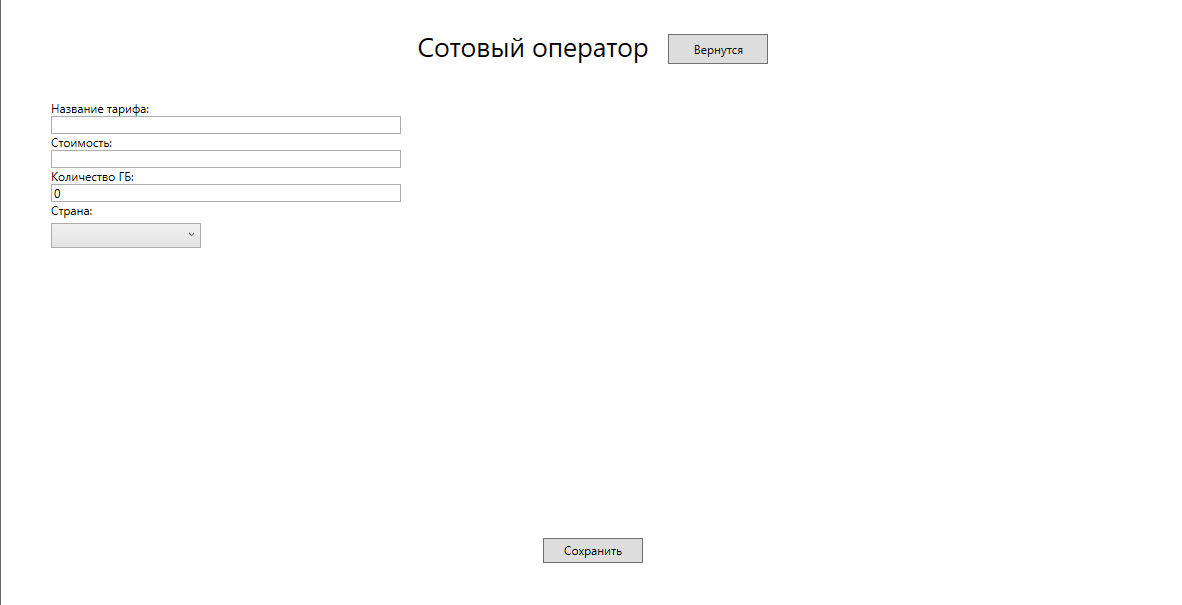


Рис.8 «Страница добавления нового тарифа»

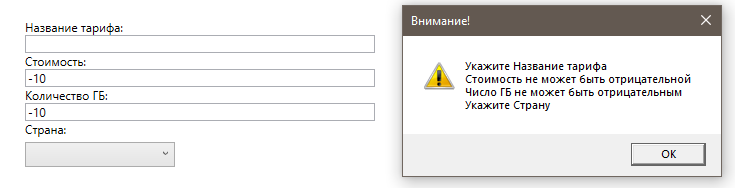


Рис.9 «Отлов ошибки в случае ввода не достоверных данных»

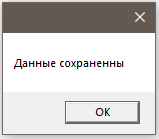


Рис.10 «Уведомление об успешном добавлении нового тарифа»

Код C# перехода со справочной страницы с тарифами на страницу добавления:

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

{

this.NavigationService.Navigate(new Pages.AddEditTarif(null));

}

Код C# Отлова ошибок, вывода предупреждения и добавления в базу данных:

public partial class AddEditTarif : Page

{

private Tarif \_currentperson = new Tarif(); //экземпляр добавляемого пользователя

public AddEditTarif(Tarif selectedUser) // в конструктор добавлен праметр типа User

{

InitializeComponent();

if (selectedUser != null)

\_currentperson = selectedUser;

DataContext = \_currentperson;

CmbFiltrTarifAD.ItemsSource = cellularproviderEntities.GetContext().Country.ToList();

CmbFiltrTarifAD.SelectedValuePath = "IDCountry";

CmbFiltrTarifAD.DisplayMemberPath = "CountryName";

}

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

{

StringBuilder error = new StringBuilder();// Объект для сообщение об ошибке

//проверка полей объекта

if (string.IsNullOrWhiteSpace(\_currentperson.TarifName))

error.AppendLine("Укажите Название тарифа");

if (string.IsNullOrWhiteSpace(price.Text))

error.AppendLine("Укажите стоимость");

if (string.IsNullOrWhiteSpace(internet.Text))

error.AppendLine("Укажите количество гб");

if (0 > Int32.Parse(price.Text))

error.AppendLine("Стоимость не может быть отрицательной");

if (0 > Int32.Parse(internet.Text))

error.AppendLine("Число ГБ не может быть отрицательным");

if (string.IsNullOrWhiteSpace(CmbFiltrTarifAD.Text))

error.AppendLine("Укажите Страну");

if (error.Length > 0)

{

MessageBox.Show(error.ToString(), "Внимание!", MessageBoxButton.OK, MessageBoxImage.Warning);

return;

}

if (\_currentperson.IDTarif == 0)

cellularproviderEntities.GetContext().Tarif.Add(\_currentperson);//Добавить в контекст

try

{

cellularproviderEntities.GetContext().SaveChanges();// Сохранить изменения

MessageBox.Show("Данные сохраненны");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

private void CmbFiltrTarifAD\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

} } }

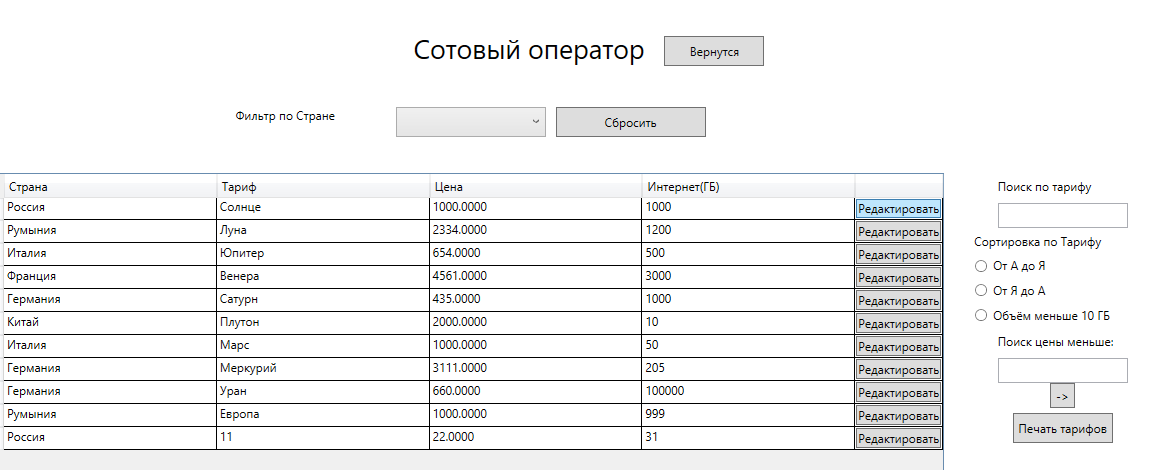


Рис.11 «Кнопка редактирования»

Код C# перехода со справочной страницы с тарифами на страницу редактирования:

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

{

this.NavigationService.Navigate(new Pages.AddEditTarif((sender as Button).DataContext as Tarif));

}

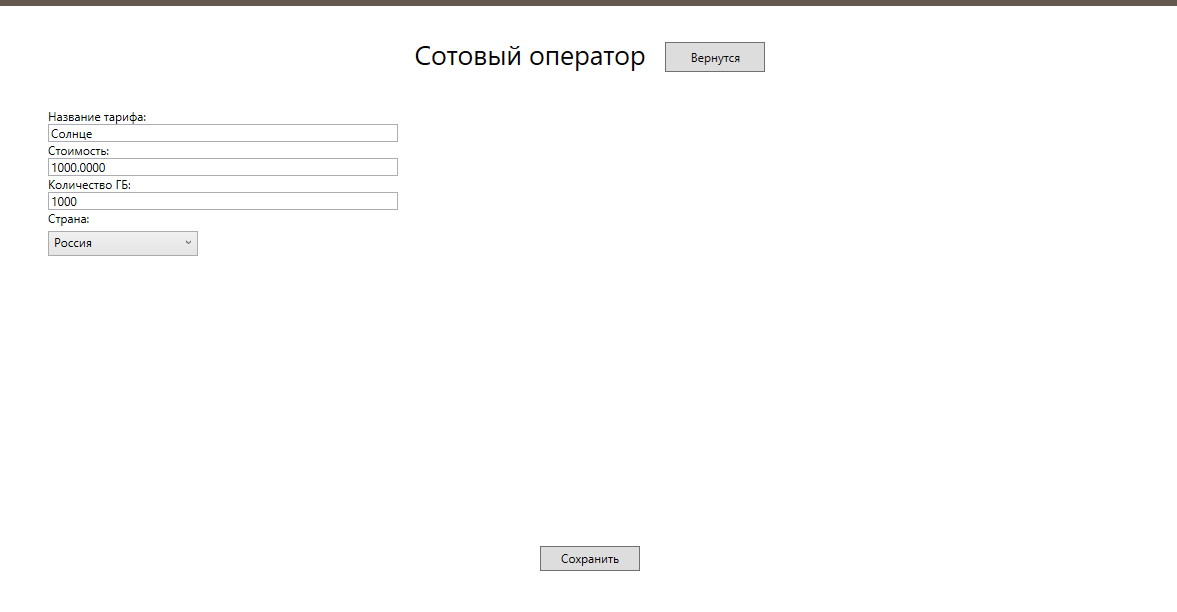


Рис.12 «Страница редактирования тарифа»

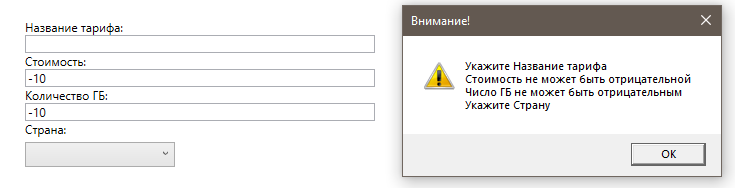


Рис.13 «Отлов ошибки в случае ввода не достоверных данных»

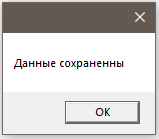


Рис.14 «Уведомление об успешном изменении тарифа»

Код C# Отлова ошибок, вывода предупреждения и изменения элемента в базе данных:

public partial class AddEditTarif : Page

{

private Tarif \_currentperson = new Tarif(); //экземпляр добавляемого пользователя

public AddEditTarif(Tarif selectedUser) // в конструктор добавлен праметр типа User

{

InitializeComponent();

if (selectedUser != null)

\_currentperson = selectedUser;

DataContext = \_currentperson;

CmbFiltrTarifAD.ItemsSource = cellularproviderEntities.GetContext().Country.ToList();

CmbFiltrTarifAD.SelectedValuePath = "IDCountry";

CmbFiltrTarifAD.DisplayMemberPath = "CountryName";

}

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

{

StringBuilder error = new StringBuilder();// Объект для сообщение об ошибке

//проверка полей объекта

if (string.IsNullOrWhiteSpace(\_currentperson.TarifName))

error.AppendLine("Укажите Название тарифа");

if (string.IsNullOrWhiteSpace(price.Text))

error.AppendLine("Укажите стоимость");

if (string.IsNullOrWhiteSpace(internet.Text))

error.AppendLine("Укажите количество гб");

if (0 > Int32.Parse(price.Text))

error.AppendLine("Стоимость не может быть отрицательной");

if (0 > Int32.Parse(internet.Text))

error.AppendLine("Число ГБ не может быть отрицательным");

if (string.IsNullOrWhiteSpace(CmbFiltrTarifAD.Text))

error.AppendLine("Укажите Страну");

if (error.Length > 0)

{

MessageBox.Show(error.ToString(), "Внимание!", MessageBoxButton.OK, MessageBoxImage.Warning);

return;

}

if (\_currentperson.IDTarif == 0)

cellularproviderEntities.GetContext().Tarif.Add(\_currentperson);//Добавить в контекст

try

{

cellularproviderEntities.GetContext().SaveChanges();// Сохранить изменения

MessageBox.Show("Данные сохраненны");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

private void CmbFiltrTarifAD\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

}

}

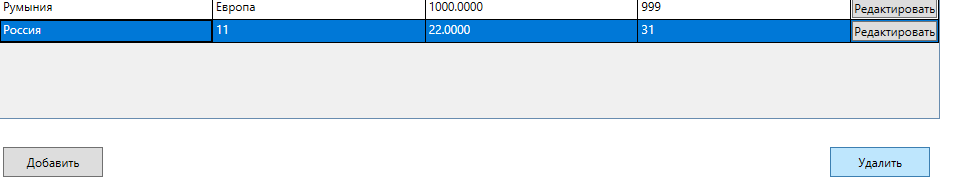
****

Рис.15 «Кнопка удаления тарифа»

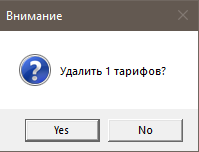


Рис.16 «Подтверждение удаления тарифа»

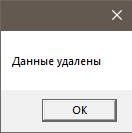


Рис.17 «Сообщение в случае удаления тарифа»

Код C# Удаления данных о тарифе из базы данных:

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

{ //Удаление нескольких пользователей

var personForRemoving = DGridTarif.SelectedItems.Cast<Tarif>().ToList();

//Подтверждение удаления

if (MessageBox.Show($"Удалить {personForRemoving.Count()} тарифов?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

try

{ //удаление

cellularproviderEntities.GetContext().Tarif.RemoveRange(personForRemoving);

cellularproviderEntities.GetContext().SaveChanges();

MessageBox.Show("Данные удалены");

DGridTarif.ItemsSource = cellularproviderEntities.GetContext().Tarif.ToList();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}}

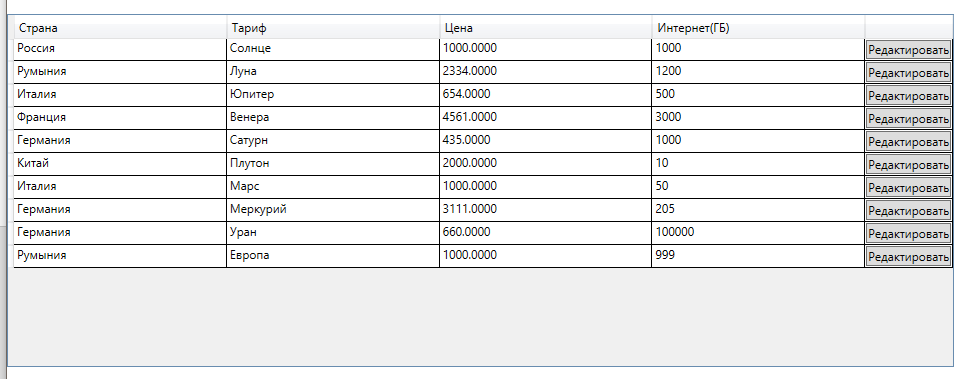


Рис.18 «Отображение данных о тарифах из базы данных»

Код C# вывода данных о тарифе из базы данных в таблицу:

public TableTarif()

{

InitializeComponent();

DGridTarif.ItemsSource = cellularproviderEntities.GetContext().Tarif.ToList();

CmbFiltrTarif.ItemsSource = cellularproviderEntities.GetContext().Country.ToList();

CmbFiltrTarif.SelectedValuePath = "IDCountry";

CmbFiltrTarif.DisplayMemberPath = "CountryName";

}

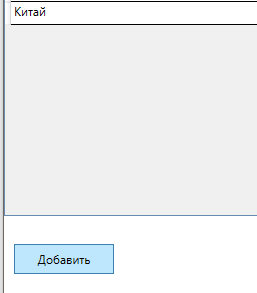


Рис.19 «Кнопка добавления на странице со странами»

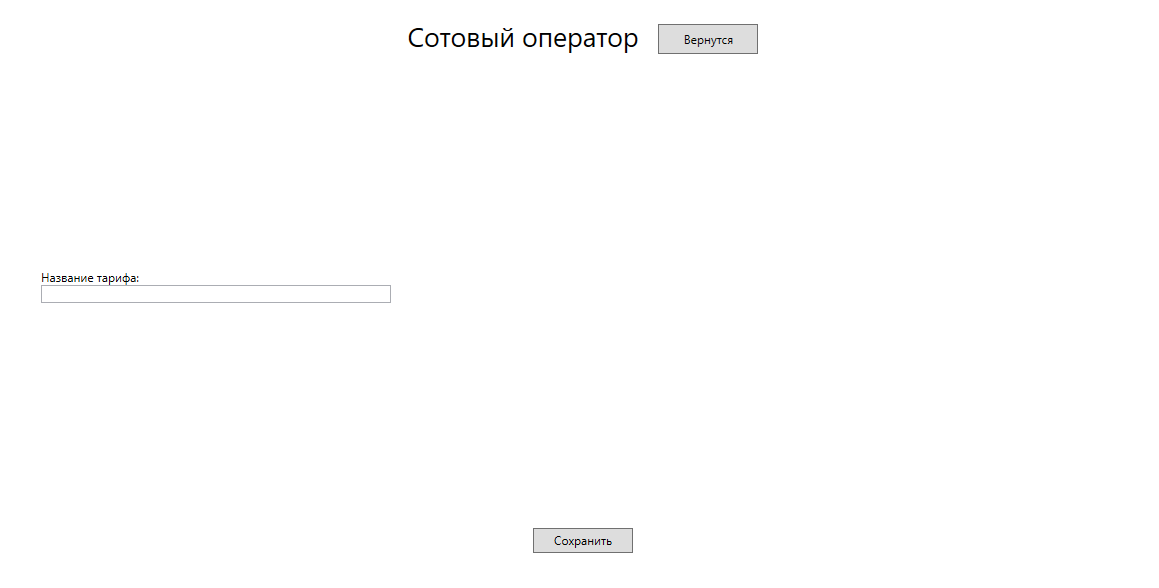


Рис.20 «Страница добавления новой страны»

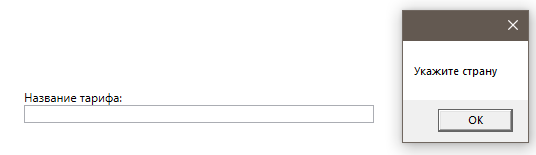


Рис.21 «Отлов ошибки в случае ввода не достоверных данных»

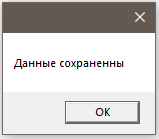


Рис.22 «Уведомление об успешном добавлении новой страны»

Код C# перехода с справочной страницы со странами на страницу добавления:

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

{

this.NavigationService.Navigate(new Pages.AddEditCountry(null));

}

Код C# Отлова ошибок, вывода предупреждения и добавления в базу данных:

public partial class AddEditCountry : Page

{

private Country \_currentperson = new Country(); //экземпляр добавляемого пользователя

public AddEditCountry(Country selectedUser) // в конструктор добавлен праметр типа User

{

InitializeComponent();

if (selectedUser != null)

\_currentperson = selectedUser;

DataContext = \_currentperson;

}

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

{

StringBuilder error = new StringBuilder();// Объект для сообщение об ошибке

//проверка полей объекта

if (string.IsNullOrWhiteSpace(\_currentperson.CountryName))

error.AppendLine("Укажите страну");

if (error.Length > 0)

{

MessageBox.Show(error.ToString());

return;

}

if (\_currentperson.IDCountry == 0)

cellularproviderEntities.GetContext().Country.Add(\_currentperson);//Добавить в контекст

try

{

cellularproviderEntities.GetContext().SaveChanges();// Сохранить изменения

MessageBox.Show("Данные сохраненны");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}}

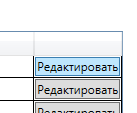


Рис.23 «Кнопка редактирования»

Код C# перехода со справочной страницы со странами на страницу редактирования:

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

{

this.NavigationService.Navigate(new Pages.AddEditCountry((sender as Button).DataContext as Country));

}

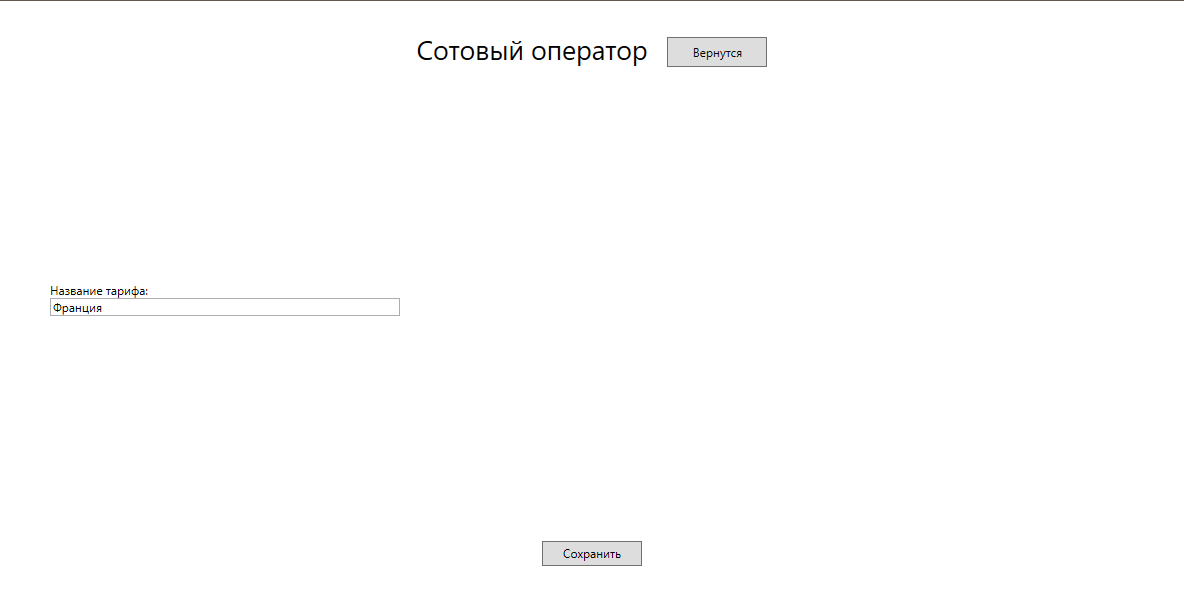


Рис.24 «Страница редактирования страны»

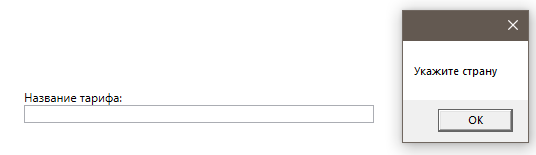


Рис.25 «Отлов ошибки в случае ввода не достоверных данных»

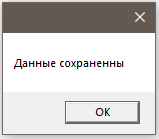


Рис.26 «Уведомление об успешном изменении страны»

Код C# Отлова ошибок, вывода предупреждения и изменения элемента в базе данных:

public partial class AddEditCountry : Page

{

private Country \_currentperson = new Country(); //экземпляр добавляемого пользователя

public AddEditCountry(Country selectedUser) // в конструктор добавлен праметр типа User

{

InitializeComponent();

if (selectedUser != null)

\_currentperson = selectedUser;

DataContext = \_currentperson;

}

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

{

StringBuilder error = new StringBuilder();// Объект для сообщение об ошибке

//проверка полей объекта

if (string.IsNullOrWhiteSpace(\_currentperson.CountryName))

error.AppendLine("Укажите страну");

if (error.Length > 0)

{

MessageBox.Show(error.ToString());

return;

}

if (\_currentperson.IDCountry == 0)

cellularproviderEntities.GetContext().Country.Add(\_currentperson);//Добавить в контекст

try

{

cellularproviderEntities.GetContext().SaveChanges();// Сохранить изменения

MessageBox.Show("Данные сохраненны");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}}

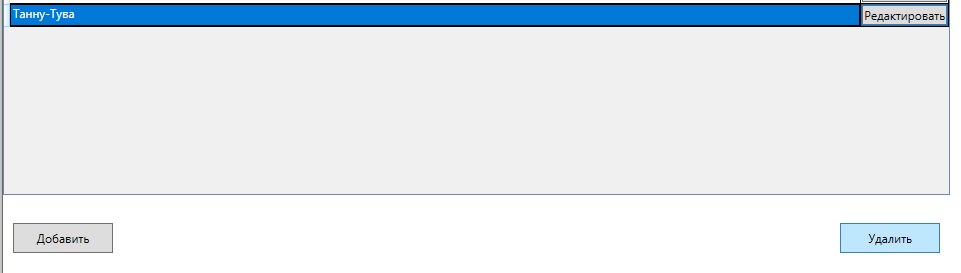
****

Рис.27 «Кнопка удаления страны»

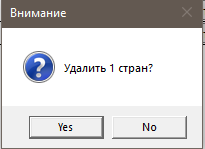


Рис.28 «Подтверждение удаления страны»

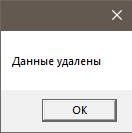


Рис.29 «Сообщение в случае удаления тарифа»

Код C# Удаления данных о тарифе из базы данных:

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

{

//Удаление нескольких

var personForRemoving = DGridCountry.SelectedItems.Cast<Country>().ToList();

if (MessageBox.Show($"Удалить {personForRemoving.Count()} стран?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

try

{

cellularproviderEntities.GetContext().Country.RemoveRange(personForRemoving);

cellularproviderEntities.GetContext().SaveChanges();

MessageBox.Show("Данные удалены");

DGridCountry.ItemsSource = cellularproviderEntities.GetContext().Country.ToList();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

}

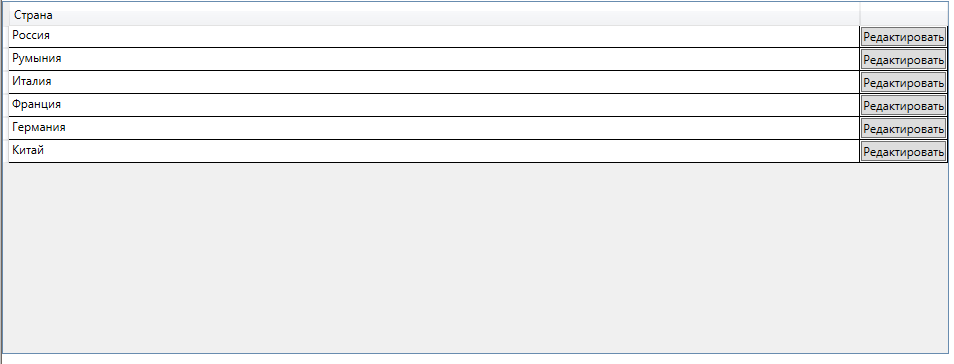


Рис.30 «Отображение данных о странах из базы данных»

Код C# вывода данных о стране из базы данных в таблицу:

public TableCountry()

{

InitializeComponent();

DGridCountry.ItemsSource = cellularproviderEntities.GetContext().Country.ToList();

}

**День 3.**

Реализация функций для работы с основной информацией: добавление/редактирование данных, поиск по заданному параметру, сортировка, фильтрация по критерию, подсчет итоговых значений.

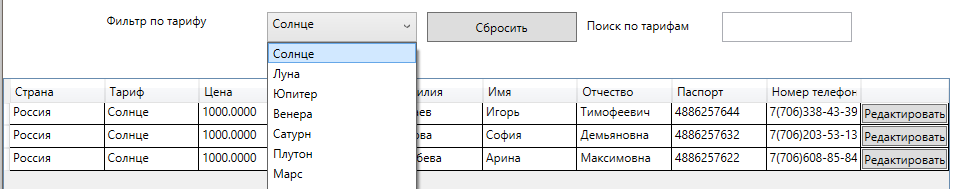


Рис.31 «Сортировка абонентов по тарифу»

Код C# сортировки по тарифу:

private void CmbFiltrGroup\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

int id = Convert.ToInt32(CmbFiltrGroup.SelectedValue);

DGridAbonents.ItemsSource = cellularproviderEntities.GetContext().Abonent.Where(x => x.IDTarif == id).ToList();

}



Рис.32 «Подсчёт средней абонентской платы»

Код C# подсчёта средней абонентской платы:

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

{

decimal? sum = cellularproviderEntities.GetContext().Abonent.Average(x => x.Tarif.Price);

countrez.Text = $"{sum.ToString()} руб.";

}

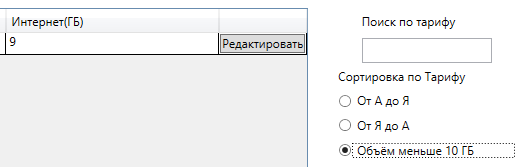


Рис.33 «Сортировка тарифов где объём меньше 10 ГБ»

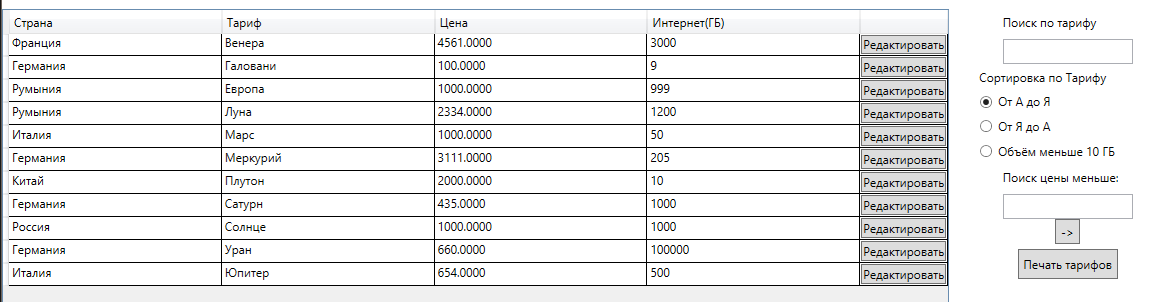


Рис.34 «Сортировка по алфавиту от А до Я»

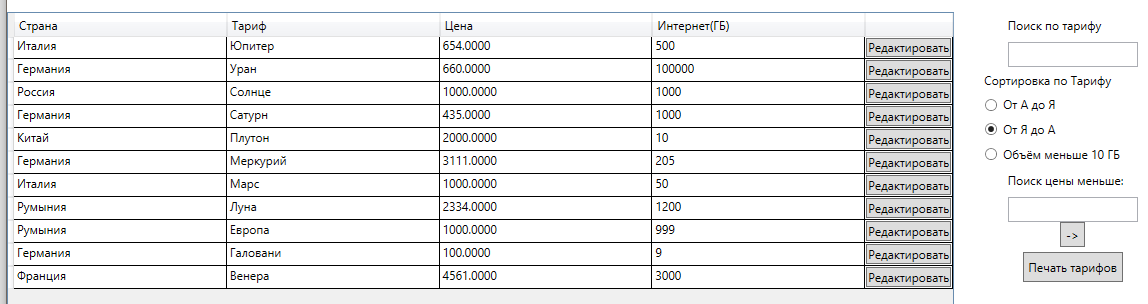


Рис.35 «Сортировка по алфавиту от Я до А»

Код C# сортировки тарифов по условиям:

private void RbUp\_Checked(object sender, RoutedEventArgs e)

{

DGridTarif.ItemsSource = cellularproviderEntities.GetContext().Tarif.OrderBy(x => x.TarifName).ToList();

}

private void RbDown\_Checked(object sender, RoutedEventArgs e)

{

DGridTarif.ItemsSource = cellularproviderEntities.GetContext().Tarif.OrderByDescending(x => x.TarifName).ToList();

}

private void RbChecked(object sender, RoutedEventArgs e)

{

DGridTarif.ItemsSource = cellularproviderEntities.GetContext().Tarif.Where(x => x.Internet < 10).ToList();

}

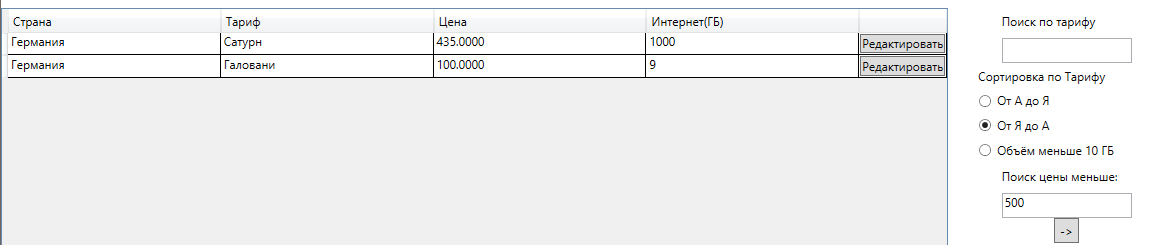


Рис.36 «Вывод тарифов с ценой ниже указанной»

Код C# вывода цены ниже указанной:

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

{

try

{

int pr1 = int.Parse(priceSearch.Text);

DGridTarif.ItemsSource = cellularproviderEntities.GetContext().Tarif.Where(x => x.Price < pr1).ToList();

}

catch (Exception ex)

{

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

}

}

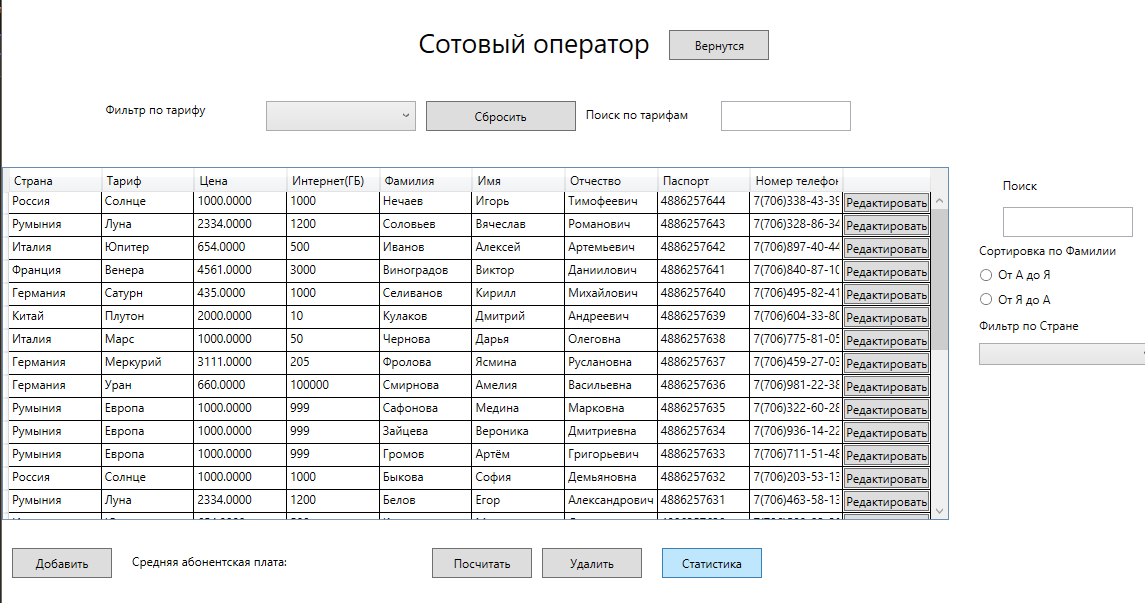


Рис.37 «Кнопка для вывода количества абонентов по тарифам»

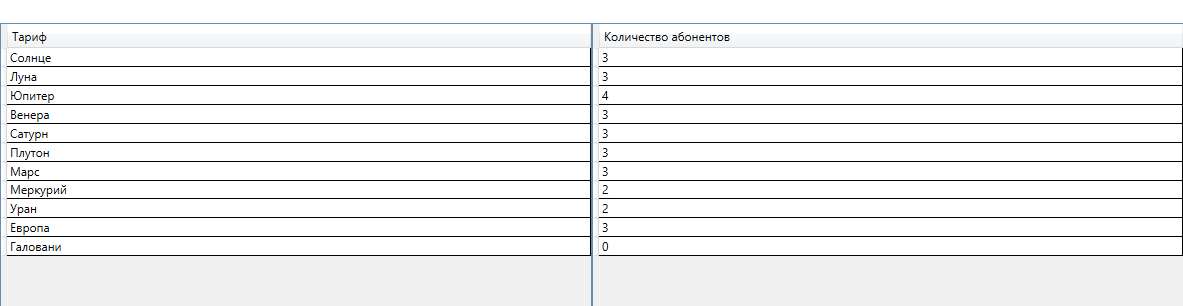


Рис.38 «Вывод количества абонентов по тарифам»

Код C# вывода количества абонентов по тарифам:

public partial class PageStatistics : Page

{

public PageStatistics()

{

InitializeComponent();

DGridTarifCount.ItemsSource = cellularproviderEntities.GetContext().Tarif.ToList();

List<TarifCountClass> listaboncnt = new List<TarifCountClass>();

foreach (int t in cellularproviderEntities.GetContext().Tarif.Select(x => x.IDTarif).ToList())

{

TarifCountClass abon = new TarifCountClass (

(

cellularproviderEntities.GetContext().Abonent.Where(x => x.IDTarif == t).Count()

));

listaboncnt.Add(abon);

}

DGridTarifCount2.ItemsSource = listaboncnt;

}

}

}

Код C# кнопки перехода на страницу со статистикой:

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

{

this.NavigationService.Navigate(new PageStatistics());

}

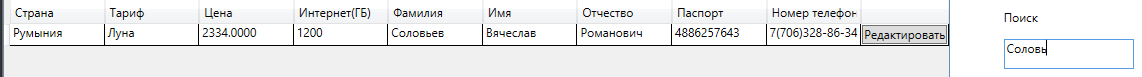


Рис.39 «Сортировка по фамилии»

Код C# поиска по фамилии абонента:

private void TxtSearch\_TextChanged(object sender, TextChangedEventArgs e)

{

DGridAbonents.ItemsSource = cellularproviderEntities.GetContext().Abonent.Where(x => x.SurName.ToLower().Contains(TxtSearch.Text.ToLower())).ToList();

}

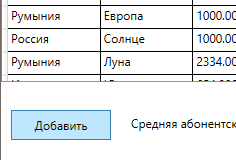


Рис.40 «Добавление нового абонента»

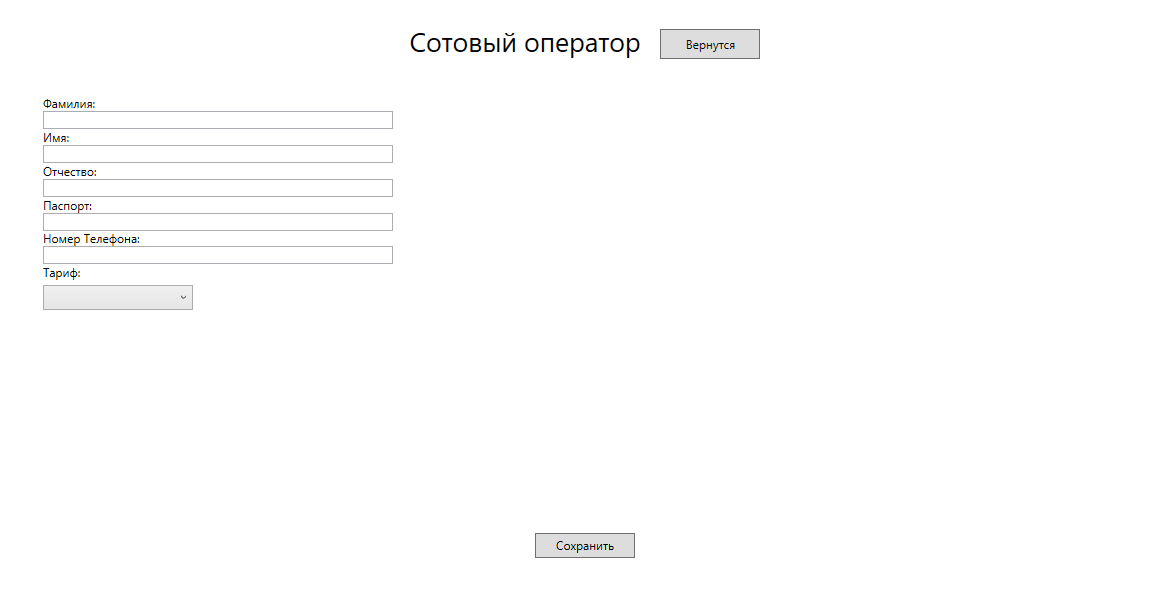


Рис.41 «Страница добавления абонента»



Рис.42 «Отлов ошибки в случае ввода не достоверных данных»



Рис.43 «Уведомление об успешном добавлении»

Код C# перехода на страницу добавления:

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

{

ClassFrame.frmObj.Navigate(new Pages.AddEditPage(null));

}

Код C# добавления элемента и проверки на ошибки ввода:

public partial class AddEditPage : Page

{

private Abonent \_currentperson = new Abonent();

public AddEditPage(Abonent selectedUser)

{

InitializeComponent();

if (selectedUser != null)

\_currentperson = selectedUser;

DataContext = \_currentperson;

CmbFiltrTarif.ItemsSource = cellularproviderEntities.GetContext().Tarif.ToList();

CmbFiltrTarif.SelectedValuePath = "IDTarif";

CmbFiltrTarif.DisplayMemberPath = "TarifName";

}

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

{

StringBuilder error = new StringBuilder();// Объект для сообщение об ошибке

if (string.IsNullOrWhiteSpace(\_currentperson.SurName))

error.AppendLine("Укажите Фамилию");

if (string.IsNullOrWhiteSpace(\_currentperson.FirstName))

error.AppendLine("Укажите Имя");

if (string.IsNullOrWhiteSpace(patron.Text))

error.AppendLine("Укажите Отчество");

if (string.IsNullOrWhiteSpace(passport.Text))

error.AppendLine("Укажите паспорт");

if (string.IsNullOrWhiteSpace(phonenumber.Text))

error.AppendLine("Укажите номер телефона");

if (string.IsNullOrWhiteSpace(CmbFiltrTarif.Text))

error.AppendLine("Укажите Тариф");

//if (cellularproviderEntities.GetContext().Abonent.Contains());

//error.AppendLine("Номер телефона уже занят");

if (error.Length > 0)

{

MessageBox.Show(error.ToString());

return;

}

if (\_currentperson.IDAbonent == 0)

cellularproviderEntities.GetContext().Abonent.Add(\_currentperson);

try

{

cellularproviderEntities.GetContext().SaveChanges();// Сохранить изменения

MessageBox.Show("Данные сохраненны");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

private void CmbFiltrTarif\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

}

}



Рис.44 «Кнопка редактирования»

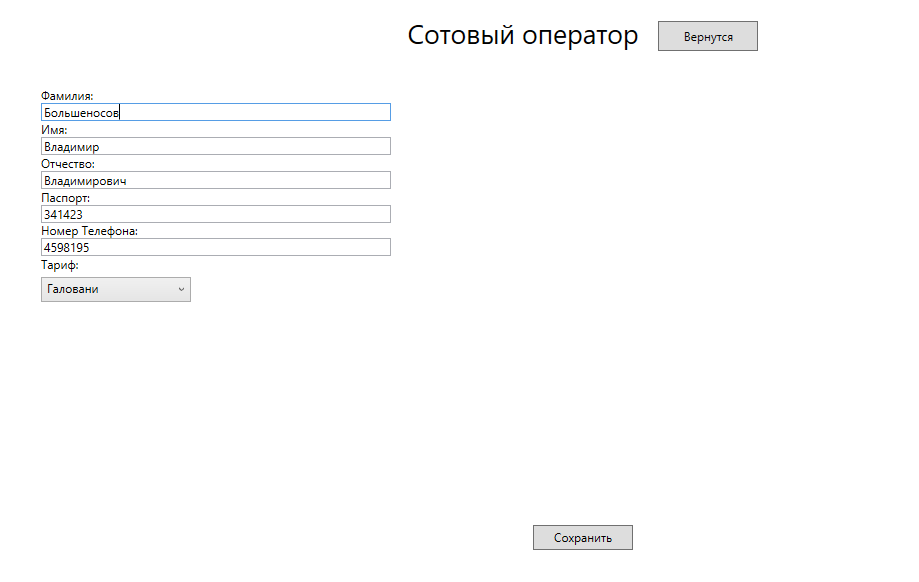


Рис.45 «Страница редактирования»



Рис.46 «Отлов ошибки в случае ввода не достоверных данных»

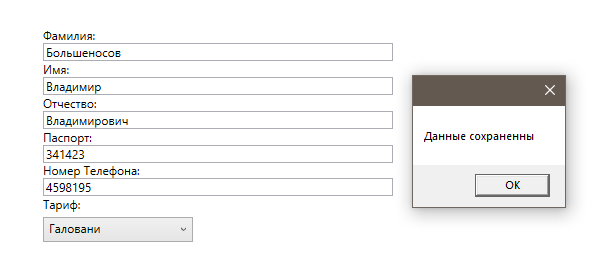


Рис.47 «Уведомление об успешном сохранении»

Код C# перехода на страницу редактирования:

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

{

ClassFrame.frmObj.Navigate(new Pages.AddEditPage((sender as Button).DataContext as Abonent));

}

Код C# изменения элемента и проверки на ошибки ввода:

public partial class AddEditPage : Page

{

private Abonent \_currentperson = new Abonent();

public AddEditPage(Abonent selectedUser)

{

InitializeComponent();

if (selectedUser != null)

\_currentperson = selectedUser;

DataContext = \_currentperson;

CmbFiltrTarif.ItemsSource = cellularproviderEntities.GetContext().Tarif.ToList();

CmbFiltrTarif.SelectedValuePath = "IDTarif";

CmbFiltrTarif.DisplayMemberPath = "TarifName";

}

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

{

StringBuilder error = new StringBuilder();// Объект для сообщение об ошибке

if (string.IsNullOrWhiteSpace(\_currentperson.SurName))

error.AppendLine("Укажите Фамилию");

if (string.IsNullOrWhiteSpace(\_currentperson.FirstName))

error.AppendLine("Укажите Имя");

if (string.IsNullOrWhiteSpace(patron.Text))

error.AppendLine("Укажите Отчество");

if (string.IsNullOrWhiteSpace(passport.Text))

error.AppendLine("Укажите паспорт");

if (string.IsNullOrWhiteSpace(phonenumber.Text))

error.AppendLine("Укажите номер телефона");

if (string.IsNullOrWhiteSpace(CmbFiltrTarif.Text))

error.AppendLine("Укажите Тариф");

//if (cellularproviderEntities.GetContext().Abonent.Contains());

//error.AppendLine("Номер телефона уже занят");

if (error.Length > 0)

{

MessageBox.Show(error.ToString());

return;

}

if (\_currentperson.IDAbonent == 0)

cellularproviderEntities.GetContext().Abonent.Add(\_currentperson);

try

{

cellularproviderEntities.GetContext().SaveChanges();// Сохранить изменения

MessageBox.Show("Данные сохраненны");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

private void CmbFiltrTarif\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

}

}

}

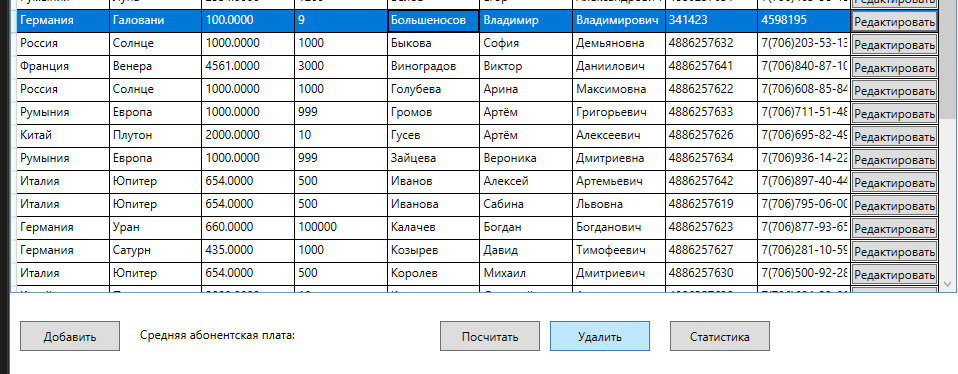


Рис.48 «Кнопка удаления абонента»

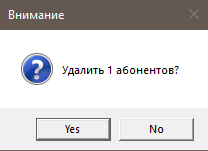


Рис.49 «Подтверждение удаления абонента»

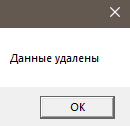


Рис.50 «Уведомление в случае успешного удаления»

Код C# удаления пользователей:

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

{

//Удаление нескольких

var personForRemoving = DGridAbonents.SelectedItems.Cast<Abonent>().ToList();

if (MessageBox.Show($"Удалить {personForRemoving.Count()} абонентов?", "Внимание", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

try

{

cellularproviderEntities.GetContext().Abonent.RemoveRange(personForRemoving);

cellularproviderEntities.GetContext().SaveChanges();

MessageBox.Show("Данные удалены");

DGridAbonents.ItemsSource = cellularproviderEntities.GetContext().Abonent.ToList();

}

catch (Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

}

}

**День 4.**

Подобрать документ по своей предметной области (чек, бланк, договор и т.п.).

Реализовать функцию вывода данных на печать (MS Excel, MS Word).

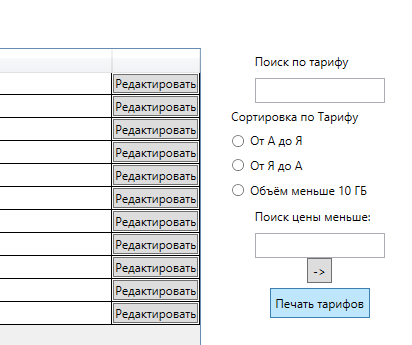


Рис.51 «Кнопка перехода на страницу печати»

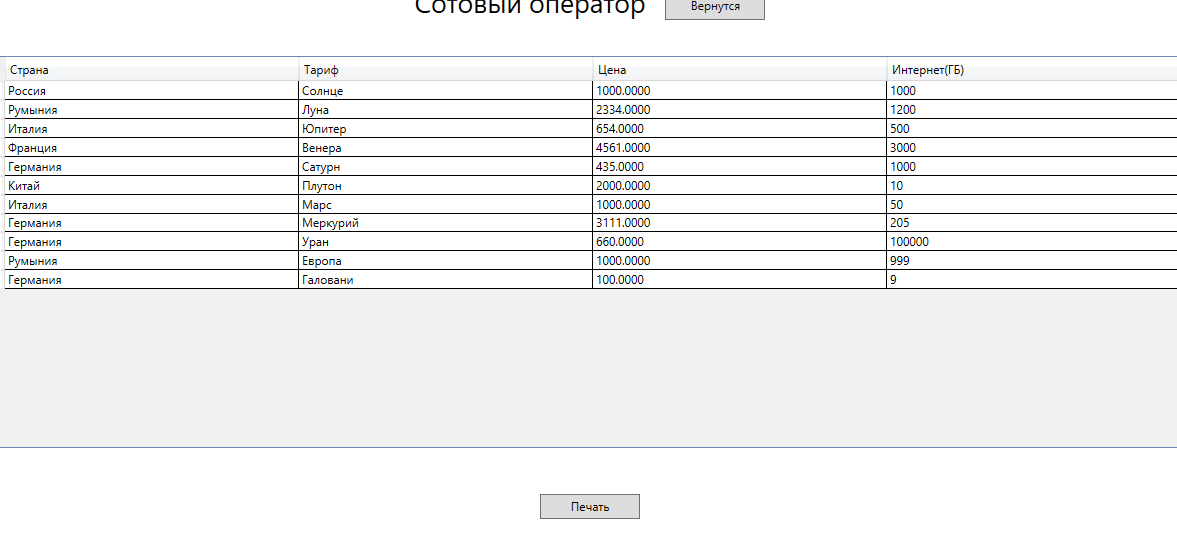


Рис.52 «Страница печати»

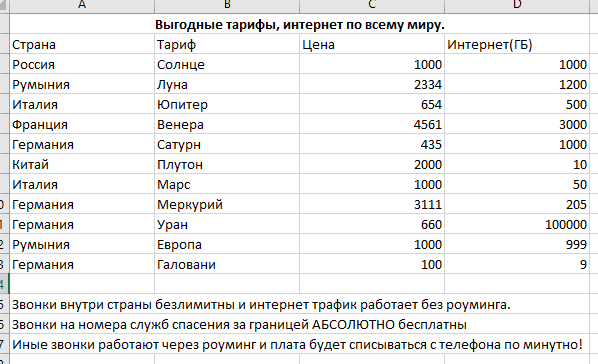


Рис.53 «Результат вывода в Excel»

Код C# вывода документа на печать:

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

{

//объект Excel

var app = new Excel.Application();

//книга

Excel.Workbook wb = app.Workbooks.Add();

//лист

Excel.Worksheet worksheet = app.Worksheets.Item[1];

int indexRows = 2;

//ячейка

Excel.Range \_excelCells1 = (Excel.Range)worksheet.get\_Range("A1", "D1").Cells;

// Производим объединение

\_excelCells1.Merge(Type.Missing);

worksheet.Cells[1, 1] = "Выгодные тарифы, интернет по всему миру.";

\_excelCells1.HorizontalAlignment = Excel.XlHAlign.xlHAlignCenter;

\_excelCells1.Font.Bold = true;

worksheet.Cells[1][indexRows] = "Страна";

worksheet.Cells[2][indexRows] = "Тариф";

worksheet.Cells[3][indexRows] = "Цена";

worksheet.Cells[4][indexRows] = "Интернет(ГБ)";

//список пользователей

var printItems = cellularproviderEntities.GetContext().Tarif.ToList();

//цикл по данным из таблицы

foreach (var item in printItems)

{

worksheet.Cells[1][indexRows + 1] = item.Country.CountryName;

worksheet.Cells[2][indexRows + 1] = item.TarifName;

worksheet.Cells[3][indexRows + 1] = item.Price;

worksheet.Cells[4][indexRows + 1] = item.Internet;

indexRows++;

}

Excel.Range range = worksheet.Range[worksheet.Cells[1][indexRows + 1],

worksheet.Cells[5][indexRows + 1]];

range.ColumnWidth = 20; //ширина столбцов

range.HorizontalAlignment = Excel.XlHAlign.xlHAlignLeft;//выравнивание по левому краю

Excel.Range \_excelCells2 = (Excel.Range) worksheet.Range[worksheet.Cells[1, indexRows + 2], worksheet.Cells[4, indexRows + 2]];

// Производим объединение

\_excelCells2.Merge(Type.Missing);

worksheet.Cells[1][indexRows + 2] = "Звонки внутри страны безлимитны и интернет трафик работает без роуминга.";

worksheet.Cells[1][indexRows + 3] = "Звонки на номера служб спасения за границей АБСОЛЮТНО бесплатны";

worksheet.Cells[1][indexRows + 4] = "Иные звонки работают через роуминг и плата будет списываться с телефона по минутно!";

\_excelCells2.HorizontalAlignment = Excel.XlHAlign.xlHAlignLeft;

// показать Excel

app.Visible = true;

}

}

}