**МИНИСТЕРСТВО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ**

**Государственное бюджетное профессиональное образовательное учреждение Московской области «Люберецкий техникум имени Героя Советского Союза, летчика-космонавта Ю.А.Гагарина»**

**ОТЧЕТ**

**ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИЧЕСКОЙ ПОДГОТОВКЕ**

Громов Никита Сергеевич

по профессиональному модулю

**ПМ. 01 Разработка модулей программного обеспечения для компьютерных систем**

**ПМ. 02 Осуществление интеграции программных модулей**

Специальность

09.02.07 "Информационные системы и программирование"

Код, название

Курс 4 Группа № 195ИС

Период практической подготовки

с «16» февраля 2023 г. по «12» апреля 2023г.

Руководитель практической подготовки

от техникума преподаватель Жирнова Ю. В. \_\_\_\_\_\_\_\_\_\_

*должность, ФИО*

Руководитель практической подготовки

от организации Рудаков М.С. \_\_\_\_\_\_\_\_\_\_

*должность, ФИО*

Люберцы 2023

**ХАРАКТЕРИСТИКА ПРАКТИКАНТА**

Громов Никита Сергеевич

(Ф.И.О. практиканта)

Работал в информационном отделе, разработчик WPF приложений

(подразделение , должность , сроки работы)

с «16» февраля 2023 г. по «12» апреля 2023 г.

Количество выходов на работу 39 дней.

Пропущено дней 0, из них по неуважительной причине 0

Прошел производственную (преддипломную) практическую подготовку по специальности

09.02.07 "Информационные системы и программирование"

Качество выполнения работы

За время прохождения производственной практической подготовки показал себя с положительной стороны. В течение всего периода Савицкий Даниил Валентинович внимательно и ответственно относился к выполняемой работе.

В отношениях с коллегами Савицкий Даниил Валентинович проявил себя с лучшей стороны: внимательность, умение выслушать и понять, стремление избежать конфликта, устойчивость к стрессам.

Особенно хочется отметить умение грамотно планировать свою деятельность в соответствии со стратегией развития коллектива и выполнение работы с максимальной эффективностью. Руководство организации оценивает работу Громова Никиты Сергеевича положительно.

Руководитель практической подготовки от организации

Руководитель информационного отдела Рудаков М.С. *(должность, ФИО)*

М.П.

**Описание организации**

1. **Название:** KiberOne
2. **Адрес:** ул. Камова д.6 корп. 2
3. **Телефон:** +7 (926) 036-37-67
4. **Почта:** нет
5. **Историческая справка:** Учреждение создано в 2015 году
6. **Форма отрасли:** Обучение детей.
7. **Пред история:** Первая в мире кибер школа будущего для детей KiberOne,основатели которой находятся в кремневой долине. Огромная сеть школ по всему миру помогающая детям стать программистами.
8. **Дочерние предприятия и/или филиалы:** да
9. **Предоставляемые услуги:** Обучения детей программированию.

**Введение**

[**Описание подразделения**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919559)

[**Описание предметной области**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919560)

[**Постановка задачи**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919561)

[**Установка и настройка инструментального ПО**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919562)

[**Microsoft SQL Server Management Studio**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919563)

[**SQL EXPRESS**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919564)

[**Visual Studio 2022**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919565)

[**Создание ERD-диаграммы в Visio**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919566)

[**User Story**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919567)

[**Use case**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919568)

[**Разработка базы данных автосалона**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919569)

[**Разработка WPF приложения**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919570)

[**Код приложения**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919571)

[**Страница авторизации**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919572)

[**Вывод данных в DataGrid**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919573)

[**Корзина**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919575)

[**Заключение**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919576)

[**Список литературы**](file:///C:\Users\Мустафа\Downloads\195IS_OtchetSaviPP2023%20(1).docx#_Toc130919577)

**Цель**- разработать Windows-приложение для учёта сотрудников в кибер школе программирования KiberOne, основанный на технологии WPF для.

# **Описание предметной области**

**Програ́ммное обеспе́чение** — программа или множество программ, используемых для управления компьютером.

Microsoft разработала Visual Studio 2022, интегрированную среду разработки (IDE), которая предназначена для создания приложений для операционных систем Windows, мобильных приложений и веб-разработки. В IDE поддерживаются множество языков программирования, включая C#, C++, Visual Basic, Python, JavaScript и многие другие.

С помощью Visual Studio разработчики получают доступ к широкому набору инструментов для создания, тестирования, отладки и развертывания приложений, что позволяет ускорить процесс разработки и облегчить взаимодействие между членами команды. В IDE также доступны множество плагинов и расширений, которые облегчают задачи разработчиков и расширяют функциональность IDE.

**Visual Studio** **2022** - это интегрированная среда разработки (IDE), которую разработала компания Microsoft для создания приложений для операционных систем Windows, а также для веб-разработки и разработки мобильных приложений. Среда поддерживает множество языков программирования, включая C#, C++, Visual Basic, Python, JavaScript и многие другие.

**Visual Studio** обеспечивает разработчикам широкий набор инструментов для создания, отладки, тестирования и развертывания приложений. Это позволяет ускорить процесс разработки и упростить взаимодействие между разработчиками в команде. В Visual Studio также есть множество плагинов и расширений, которые позволяют дополнительно расширить функциональность IDE для удобства разработки и облегчения задач.

1. **Выбор программного обеспечения для заданной системы**

При анализе заданных требований к данной компьютерной системе было принято использовать операционную систему Windows 10, так как она обладает рядом преимуществ, таких как:

1. Обеспечение совместимости с предыдущими версиями системы Windows.
2. Возможность настройки автоматического создания резервной копии системы.
3. Большим количеством доступным к установке ПО.
4. Хорошая встроенная системы защиты.
5. Удобная возможность поиска настроек и программ непосредственно в меню “Пуск”
6. Отличная русификация
7. Возобновление передачи файлов, а также поддержка больших файлов.
8. Кроссплатформенность. Работает на Windows, Linux, \*BSD, Mac OS X и других ОС
9. Поддержка drag & drop
10. Удаленная правка файлов и поиск файлов
11. Поддержка очереди скачивания и загрузки файлов
12. Синхронный просмотр директорий

Так же для работы был выбран браузер Opera. Основными преимуществами браузера являются:

* Высокая производительность
* Автоматическое сохранение прогресса, при выходе из браузера, что позволяет просматривать сайты и работать с интернет приложениями не разворачивая их с самого начала.

Для сохранения нейросети будет использоваться сайт Github.com и приложение GitHub Desktop.

**Что такое Git и зачем он нужен?**

Git - это консольная утилита, для отслеживания и ведения истории изменения файлов, в вашем проекте. Чаще всего его используют для кода, но можно и для других файлов. Например, для картинок - полезно для дизайнеров.

С помощью Git-a вы можете откатить свой проект до более старой версии, сравнивать, анализировать или сливать свои изменения в репозиторий.

Репозиторием называют хранилище вашего кода и историю его изменений. Git работает локально и все ваши репозитории хранятся в определенных папках на жестком диске.

Так же ваши репозитории можно хранить и в интернете. Обычно для этого используют три сервиса:

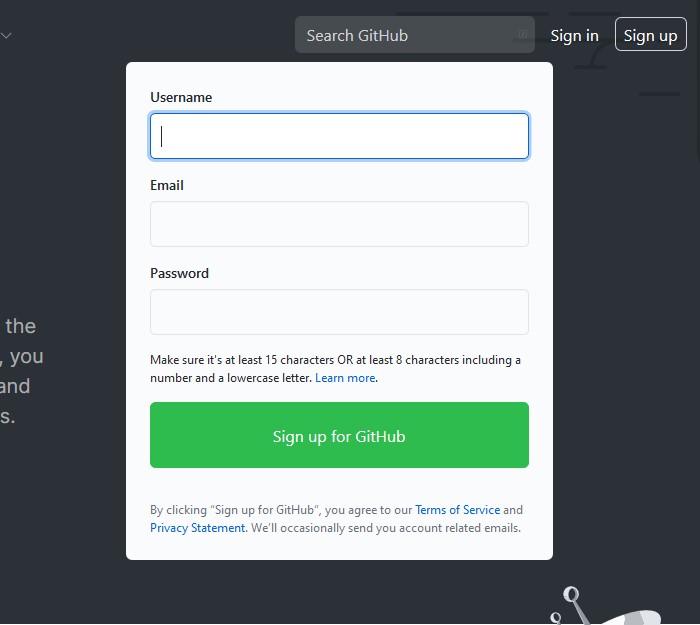
* GitHub
* Bitbucket
* GitLab

Каждая точка сохранения вашего проекта носит название коммит (commit). У каждого commit-a есть hash (уникальный id) и комментарий. Из таких commit-ов собирается ветка. Ветка - это история изменений. У каждой ветки есть свое название. Репозиторий может содержать в себе несколько веток, которые создаются из других веток или вливаются в них.

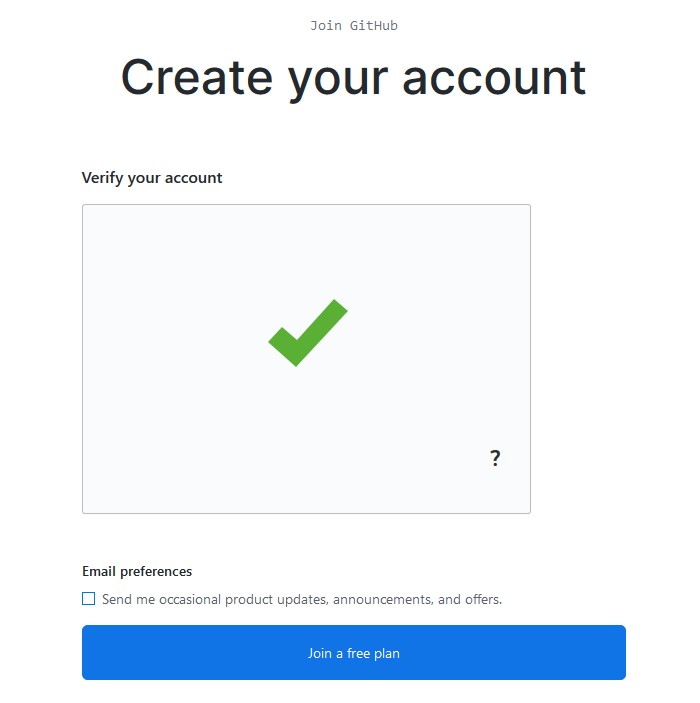
**Регистрация на GitHub**

Переходим на официальный сайт [GitHub](https://github.com/)

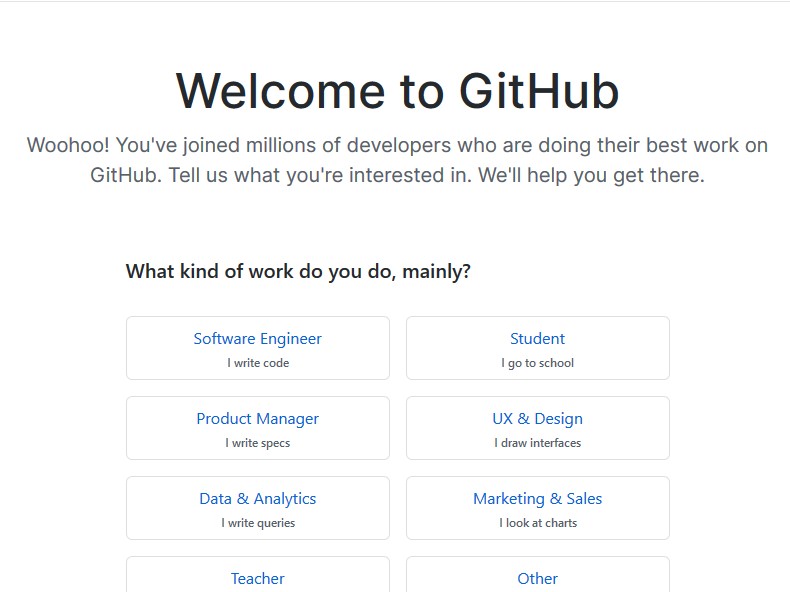
На главной странице заполняем форму справа и нажимаем **“Sign up for GitHub”**



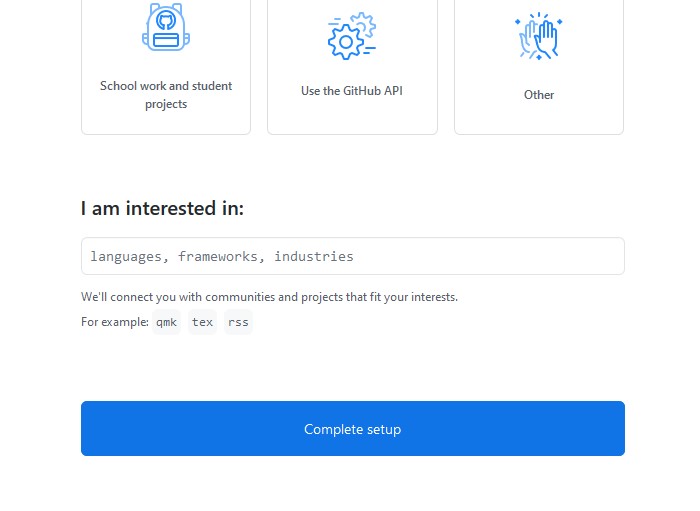
Проходим проверку и нажимаем **“Join a free plan”**



На следующей странице можно заполнить небольшую анкету (можно не заполнять)



На этой же странице спускаемся в самый низ и нажимаем **“Complete setup”**

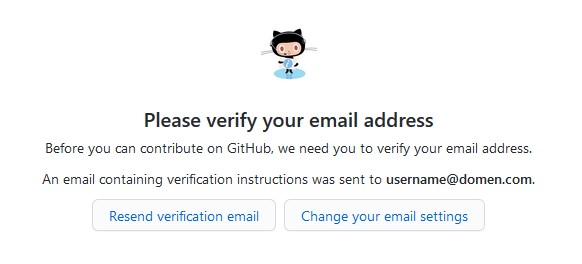


Далее для завершения регистрации просят подтвердить свой email адрес.

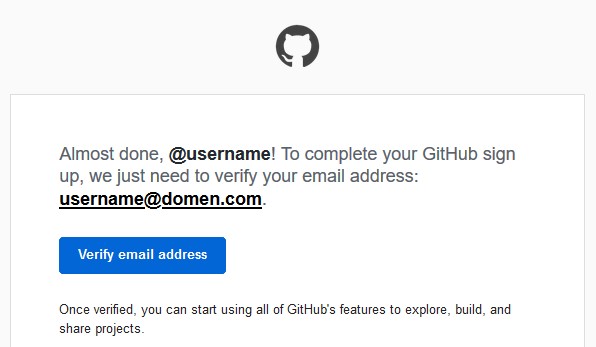
Проверяем свою почту. Если письмо пришло, переходим к следующему пункту.

Если письмо не пришло, нажимаем **“Resend verification email”**.

Если по-прежнему письмо не приходит, можно проверить и изменить настройки - нажимаем **“Change your email settings”**



В письме от **Github** нажимаем **“Verify email address”**



Аккаунт **GitHub** успешно создан

**Установка GitHub Desktop**

Переходим на официальный сайт [GitHub Desktop](https://desktop.github.com/)

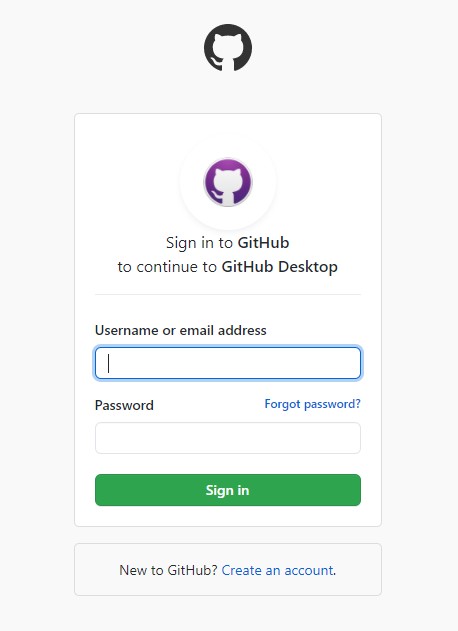
Нажимаем “Download for Windows (64bit)” (операционная система может отличаться)



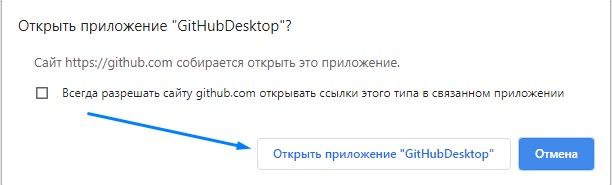
Запускаем скачанный файл. После установки в появившемся окне нажимаем **“Sign in to GitHub.com”**



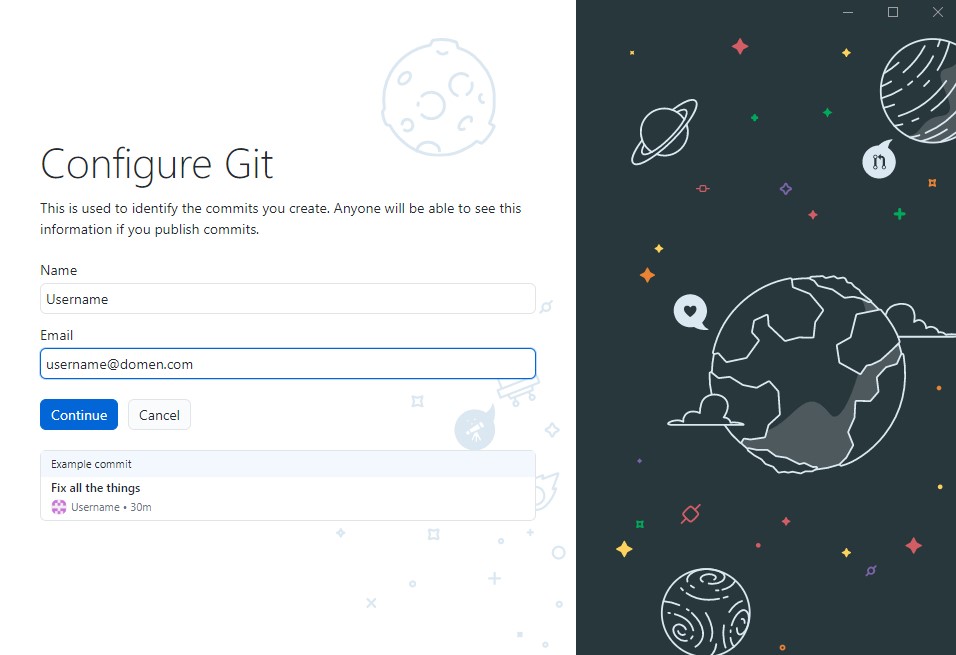
В открывшемся окне браузера вводим в форму свои данные, как при регистрации, и нажимаем **“Sign in”**



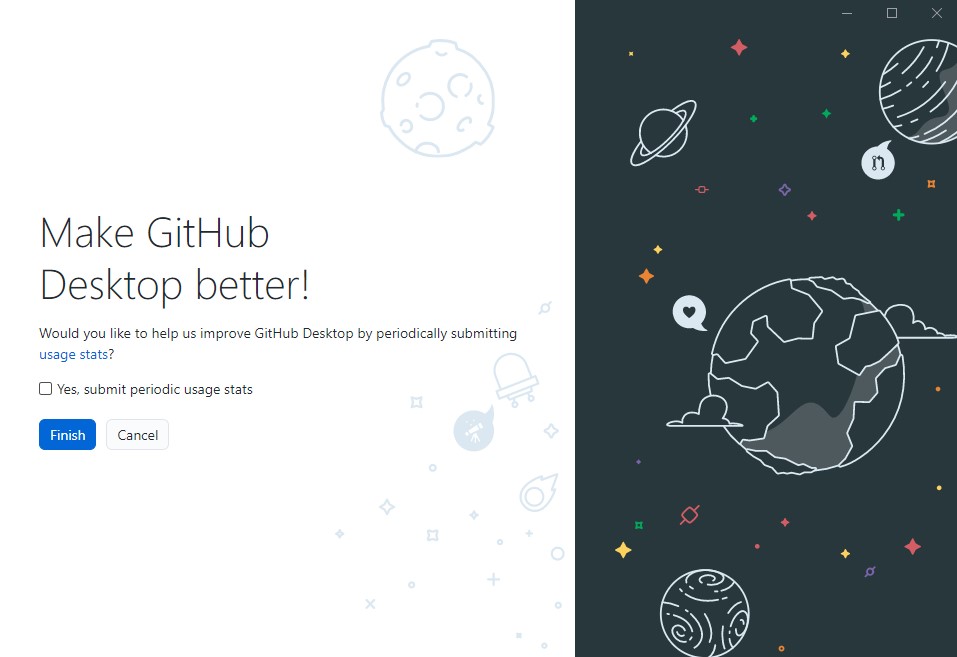
Если браузер запросит, то подтвердить, что нужно “Открыть приложение GitHub Desktop”



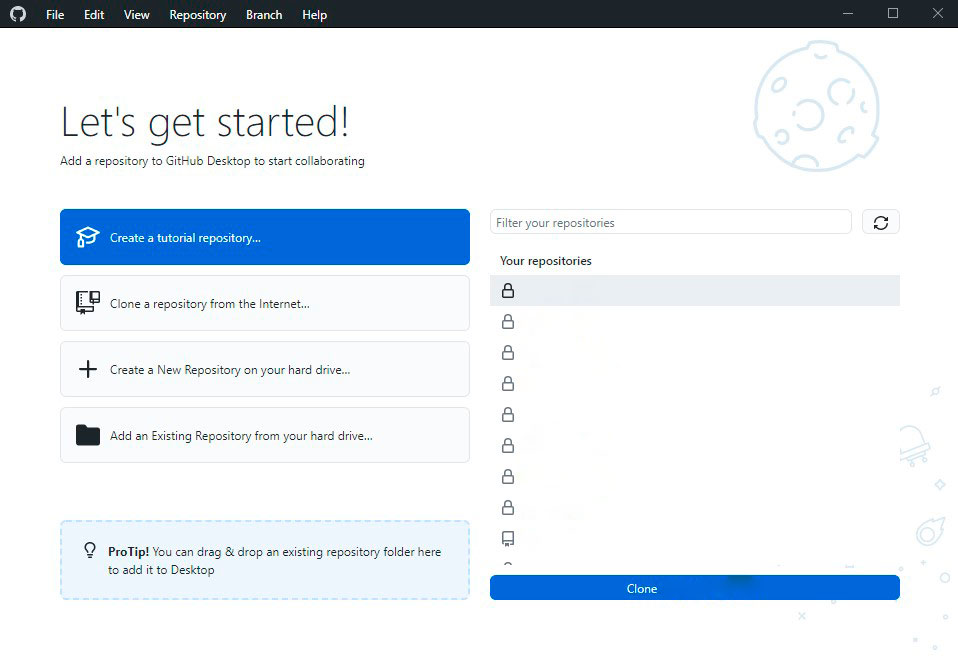
Далее регистрационные данные перенесутся в форму конфигурации (настроек) **Git** - нажимаем **“Continue”**



Отключаем пункт “Yes, submit periodic usage stats”, если не хотите периодически передавать статистику работы **GitHub Desktop** и нажимаем **“Finish”**



Далее видим начальное окно **GitHub Desktop**



**“Create a tutorial repository…“** - создать обучающий репозиторий

**“Clone repository from the Internet…“** - клонировать (скопировать/скачать) репозиторий из **GitHub** к себе на компьютер

**“Create a New Repository on your hard drive…“** - создать новый репозиторий на вашем жестком диске (на вашем компьютере) и добавить систему **Git** в проект

**“Add an Existing Repository from your hard drive…“** - добавить на **GitHub** репозиторий, который уже есть на вашем компьютере и использует **Git**

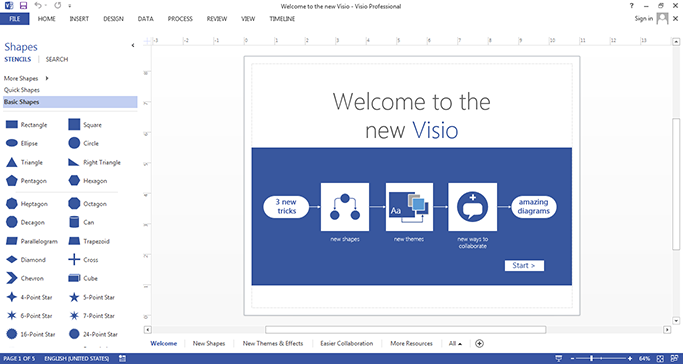
Справа будут отображаться ваши репозитории, которые уже загружены на **GitHub**, но если только что зарегистрировались, то список будет пуст.

**Microsoft Visio** — векторный графический редактор, редактор диаграмм и блок-схем для Windows.

Выпускается в трёх редакциях: Standard, Professional и Pro for Office 365.

Аналогично с Adobe Reader, в стандартный набор программ MC Office входит только средство для просмотра и печати диаграмм Microsoft Visio Viewer. Полнофункциональная версия Microsoft Visio Professional для создания и редактирования монограмм и диаграмм.

Первоначально Visio разрабатывался и выпускался компанией Shapeware, затем переименованной в Visio Corporation. Microsoft приобрела компанию в 2000 году, тогда продукт назывался Visio 2000. После этого к названию продукта был добавлен префикс Microsoft Office (так продолжалось до версии Visio 2007 (12.0)), несмотря на это продукт никогда в пакет Microsoft Office не входил и всегда распространяется отдельно.(Рисунок 2)



# Установка SQL Express и SQL Server Management Studio

Для установки SQL Server Management Studio нужно перейти на официальный сайт Microsoft и нажать на кнопку “Скачать SQL Server Management Studio”



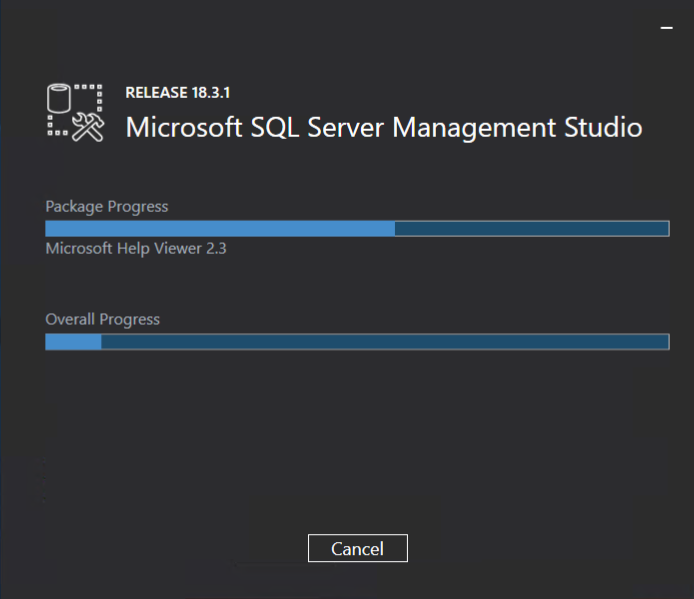
Скачивание SQL Server Management Studio

Следующим шагом выбираем место установки студии управления и нажимаем **Install:**



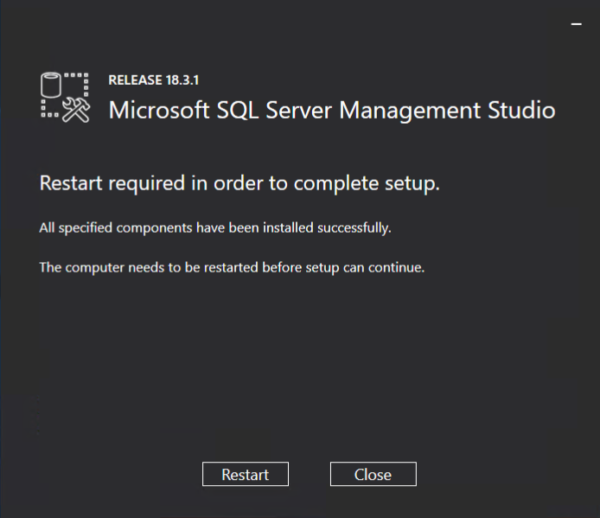
Место установки студии

После нажатия на кнопку Install процедура установки начнется автоматически:



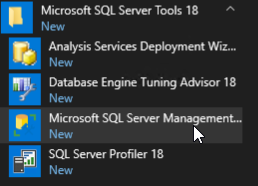
Установка

По завершении установки появится экран, предлагающий выполнить перезагрузку. Перезагружаемся, нажав на **Restart**:



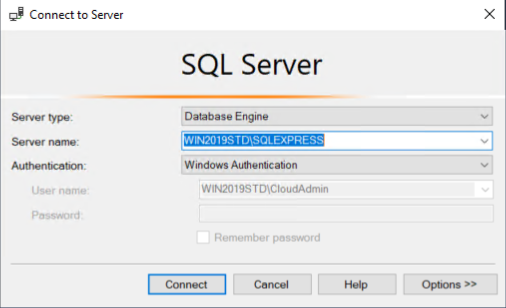
Экран перезагрузки

Программа SQL Server Management Studio готова к запуску. Чтобы подключиться к SQL Server с помощью Microsoft SQL Server Management Studio, выполните следующие действия:



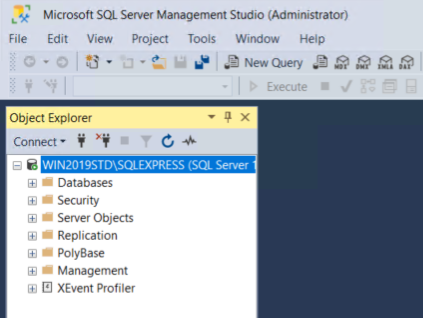
Запуск Microsoft SQL Server Management Studio из меню «Пуск»

Откроется окно аутентификации. Вы можете выбрать опцию, на основе которой настроили аутентификацию при установке MS SQL сервера. Далее нажмите **Connect**:



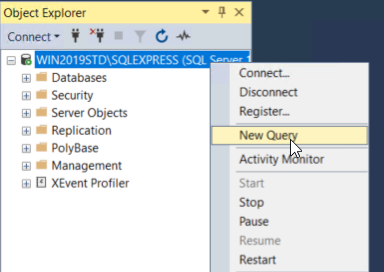
Окно аутентификации

Если соединение установлено успешно, вы увидите панель Object Explorer — обозревателя объектов:



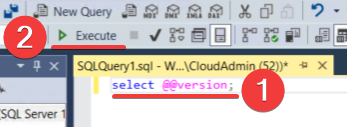
Обозреватель объектов

Чтобы выполнить запрос в базу данных, кликните правой кнопкой мыши по узлу сервера, и выберите пункт меню **New Query**:



Новый запрос

Введите в редакторе следующий запрос: select @@version. После чего нажмите **Execute** (Выполнить):



запрос возвращает версию SQL Server.

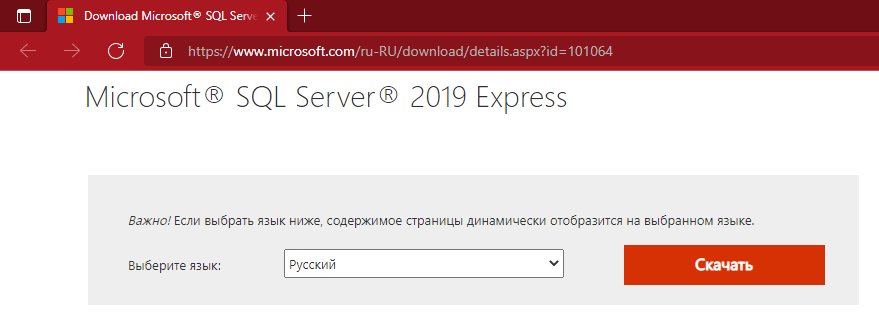
В окне «**Results**» отобразится версия SQL Server, как показано на снимке экрана ниже. Быстрый способ выполнить запрос — нажать клавишу **F5**:



Отоброжение версии SQL Server

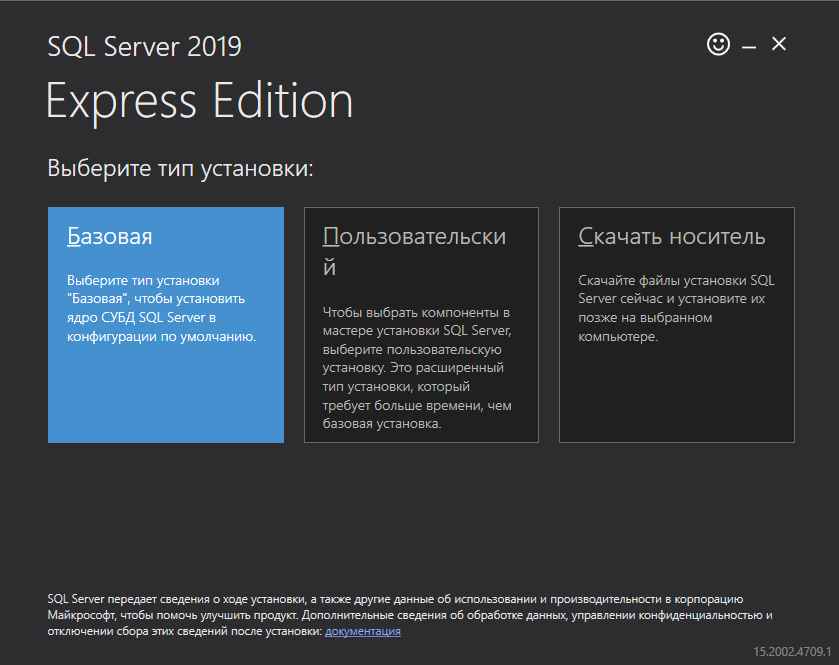
**Установка SQL Express**

Чтобы установить SQL Express, сначала необходимо его скачать. Для этого нужно перейти на официальный сайт Microsoft и нажать кнопку “Скачать” .



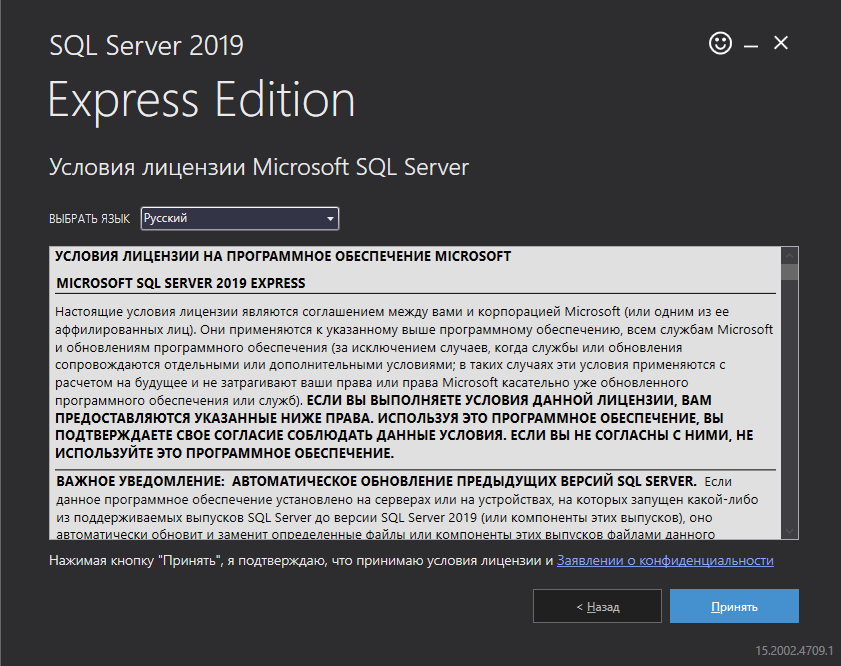
Официальный сайт Microsoft

После этого необходимо открыть установщик и выбрать тип установки, в данном случае будем устанавливать базовую версию

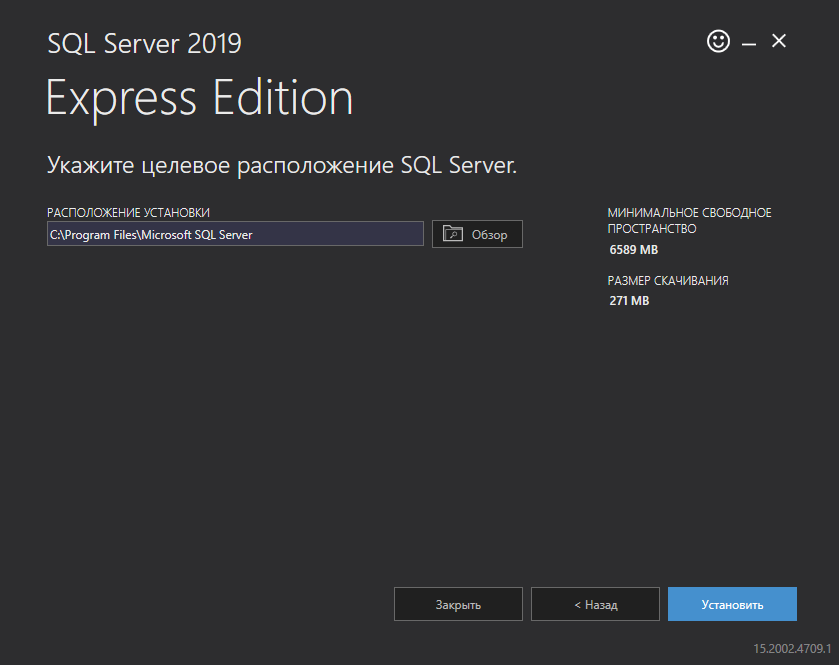


Выбор типа установки

Далее нужно принять условия лицензии Microsoft SQL Server, выбрать путь установки программы и нажать “Установить”

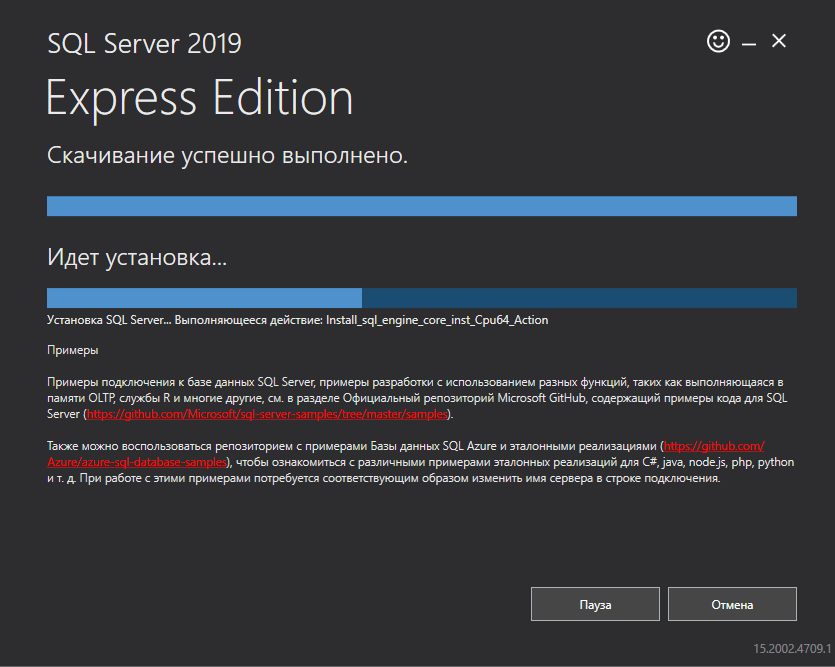


Принятие условий лицензии Microsoft SQL Server



Выбор пути установки программы

Осталось только подождать окончания установки, после чего программа будет готова к запуску



Установка SQL Server

## **Visual Studio 2022**

**Visual Studio** - это интегрированная среда разработки (IDE) от компании Microsoft, которая используется для создания программного обеспечения для различных платформ, включая Windows, macOS, Linux, Android и iOS. Visual Studio предоставляет разработчикам множество инструментов и функций, которые позволяют быстро создавать и отлаживать приложения, включая возможности для написания кода, визуального проектирования пользовательских интерфейсов, автоматической проверки кода, отладки приложений и тестирования.

Среди языков программирования, поддерживаемых Visual Studio, наиболее популярными являются C++, C#, Visual Basic, Python, JavaScript и TypeScript. Visual Studio также включает в себя инструменты для создания веб-приложений, включая ASP.NET и Razor, а также возможность создания приложений для платформы Windows Universal, мобильных устройств и игр.

Visual Studio имеет много полезных функций для коллективной работы, включая системы контроля версий и средства совместной разработки, что позволяет разработчикам работать в команде и синхронизировать свои работы.

Для загрузки Visual Studio перейдём на официальный сайт Microsoft, на страницу установщика Visual Studio 2022 Community Edition (Рисунок 12).

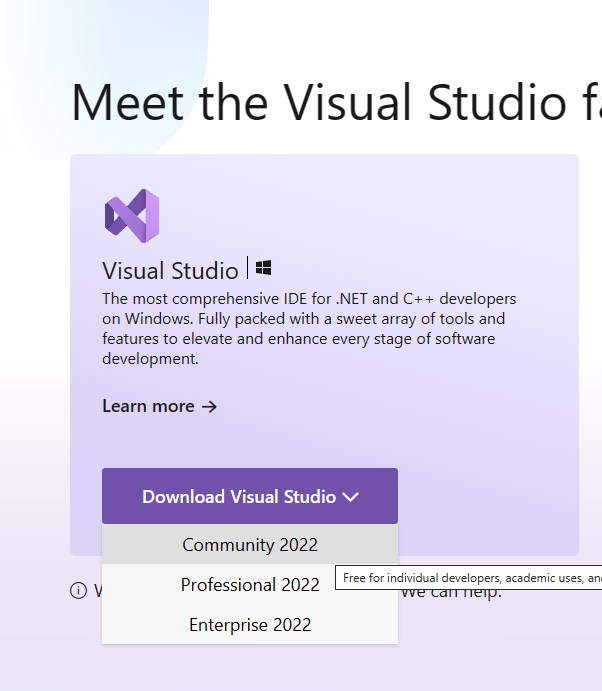


Рисунок 12. Страница загрузки инсталлятора Visual Studio 2022 Community Edition.

Дважды щёлкаем по иконке установщика, принимаем условия лицензионного соглашения по нажатию кнопки «Continue» (Рисунок 13).



Рисунок 13. Принятие лицензионного соглашения.

После установки Visual Studio Installer его можно использовать для настройки установки, выбрав нужные наборы компонентов или рабочие нагрузки. Вот как это сделать (Рисунок 14).

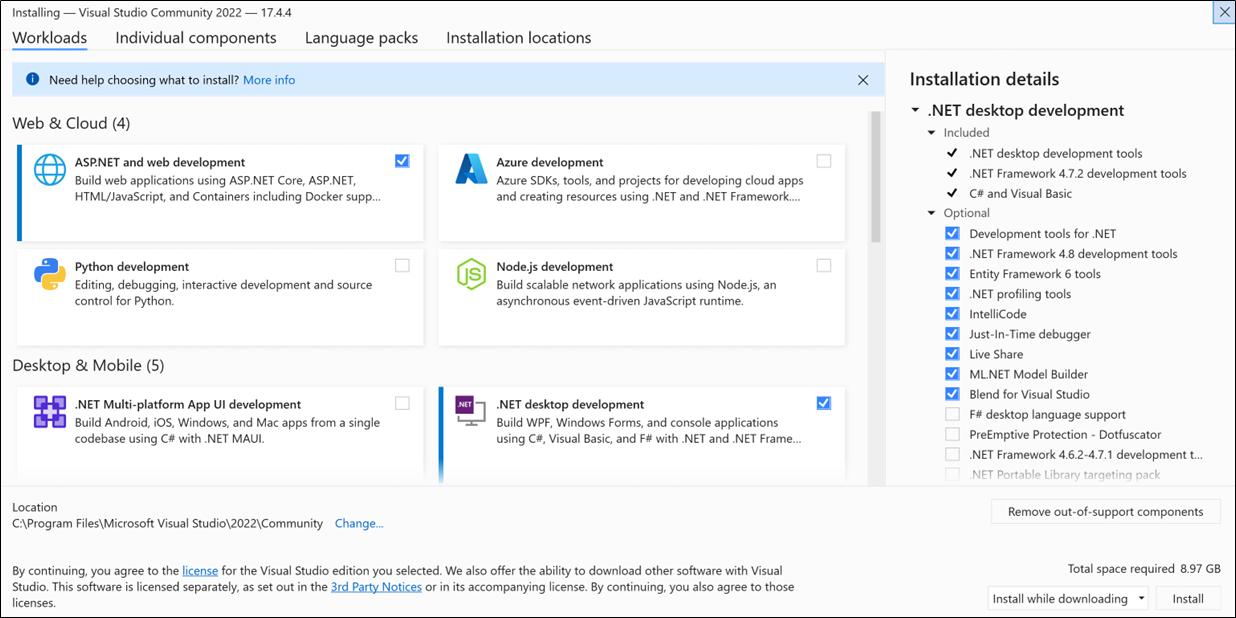


Рисунок 14. Выбор компонентов для установки.

Выбираем необходимые нам для работы компоненты и нажимаем «Install». Далее будут отображаться экраны состояния, на которых демонстрируется ход установки Visual Studio (Рисунок 15).

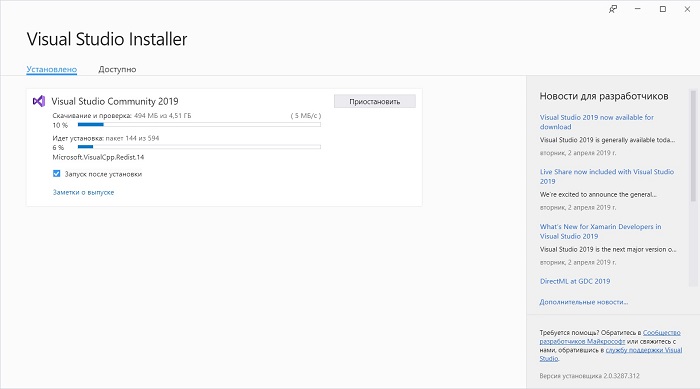


Рисунок 15. Процесс установки Visual Studio 2022.

Как только Visual Studio Installer закончит загрузку, необходимо будет перезагрузить компьютер, перезагружаемся (Рисунок 16).

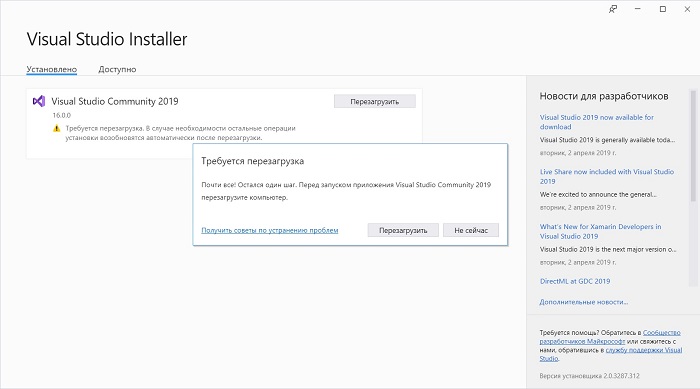
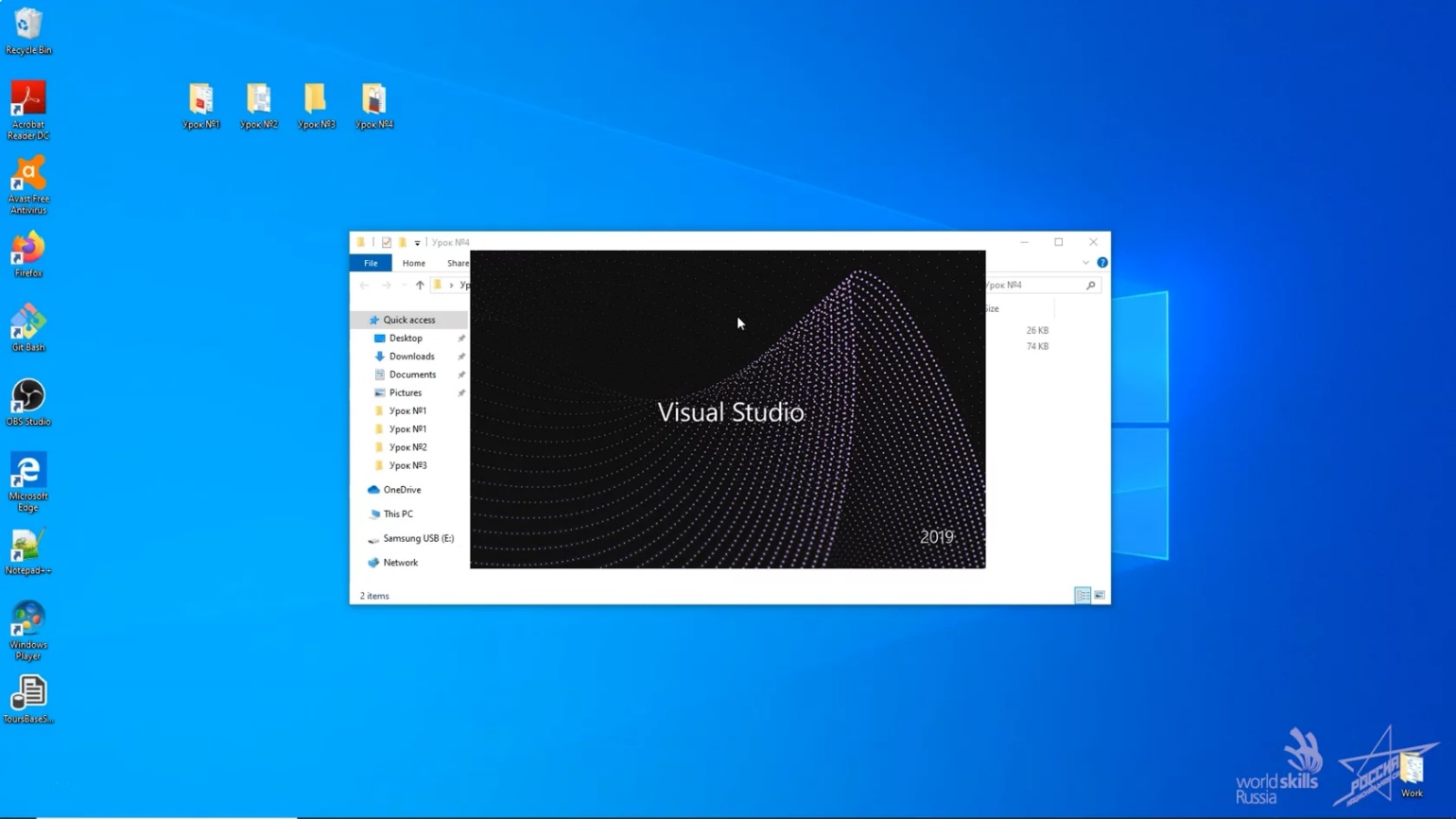


Рисунок 16. Завершение установки.

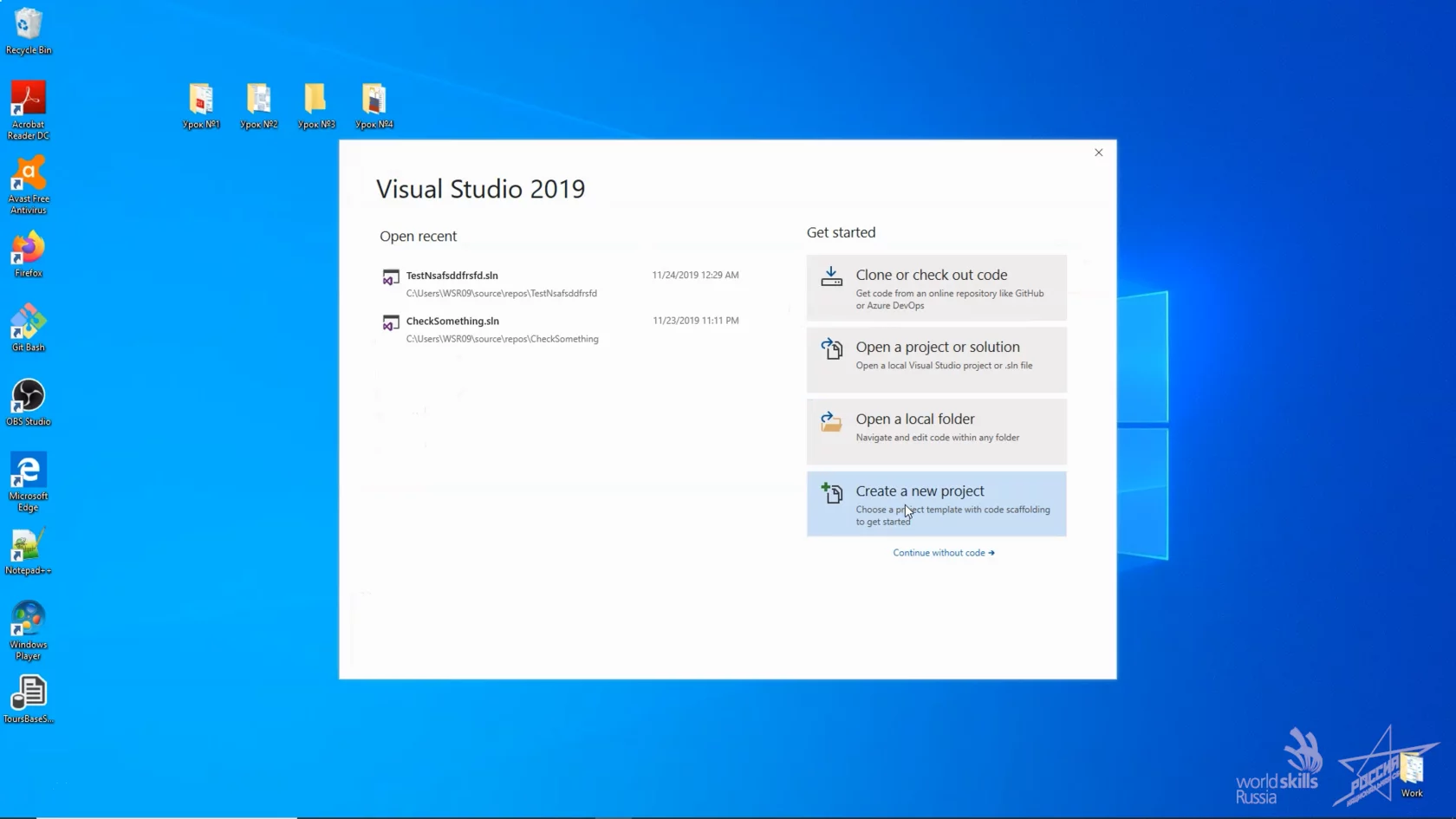
Visual Studio 2022 Успешно установлен.

**Разработка WPF приложения**

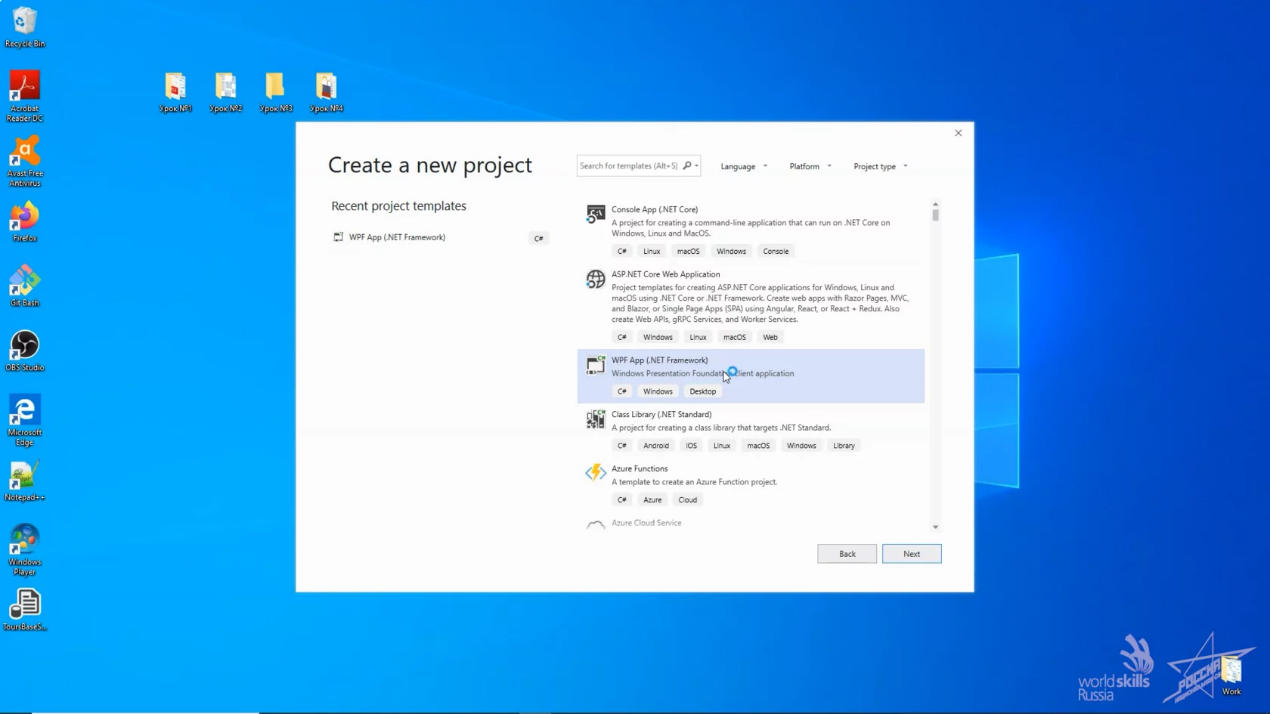
**Cоздание нового проекта**  
**1.** Запустить среду разработки Visual Studio



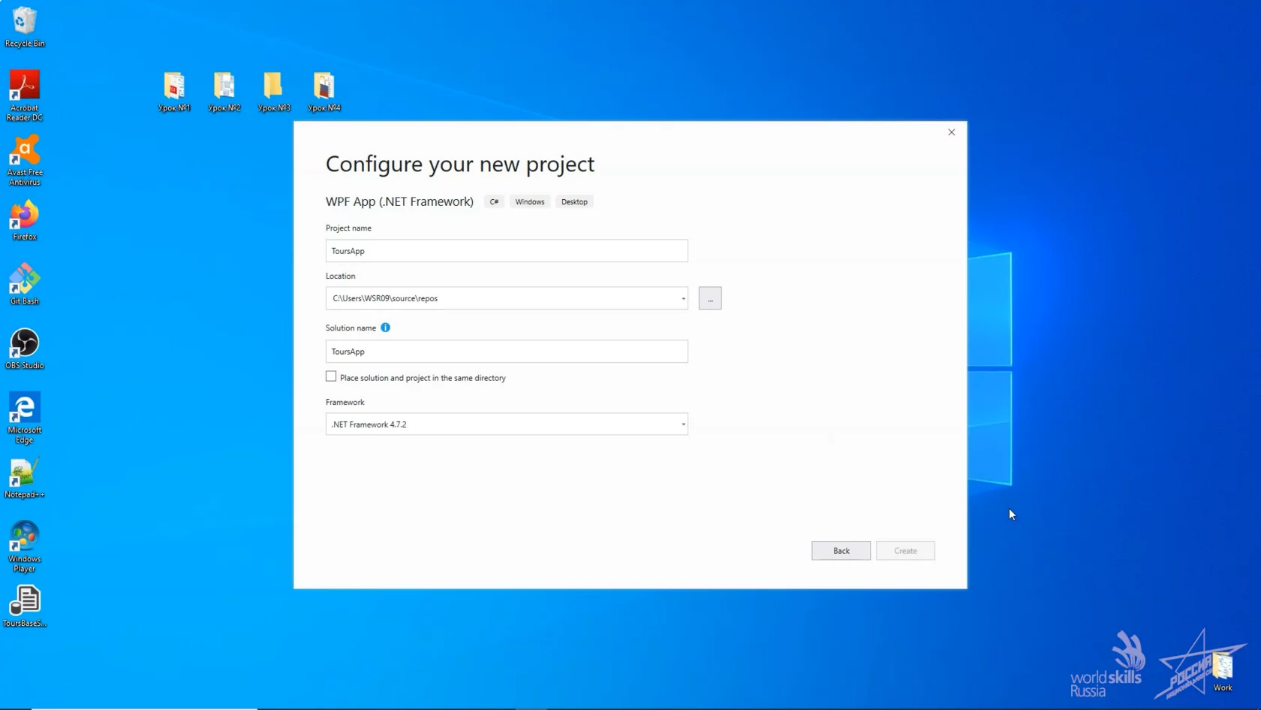
**2.** Выбрать «Create new project»



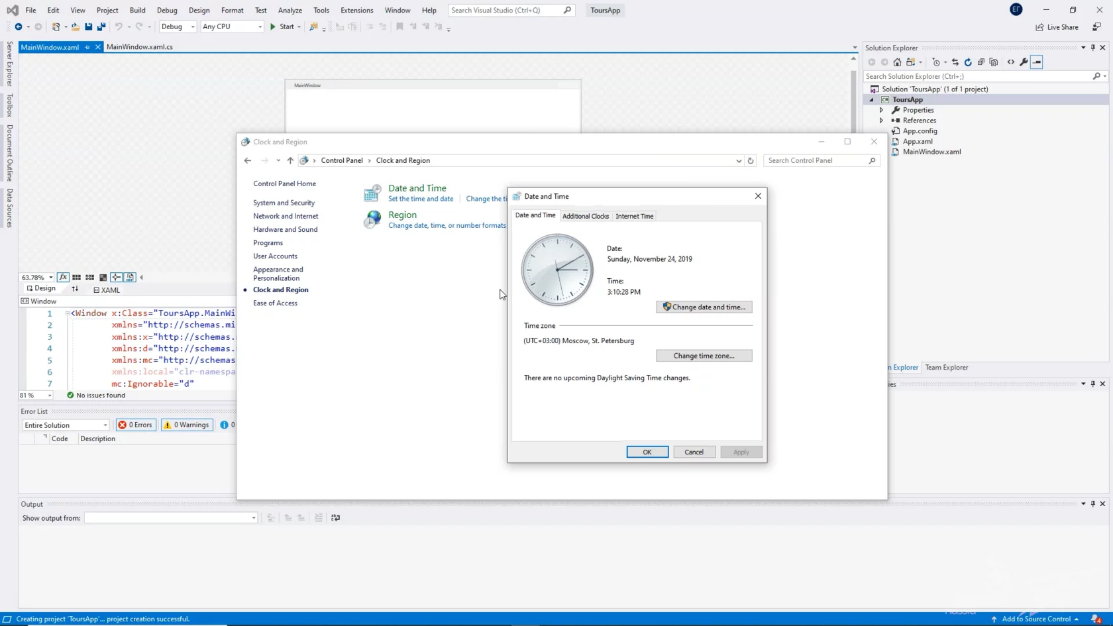
**3.** Выбрать «WPF App»



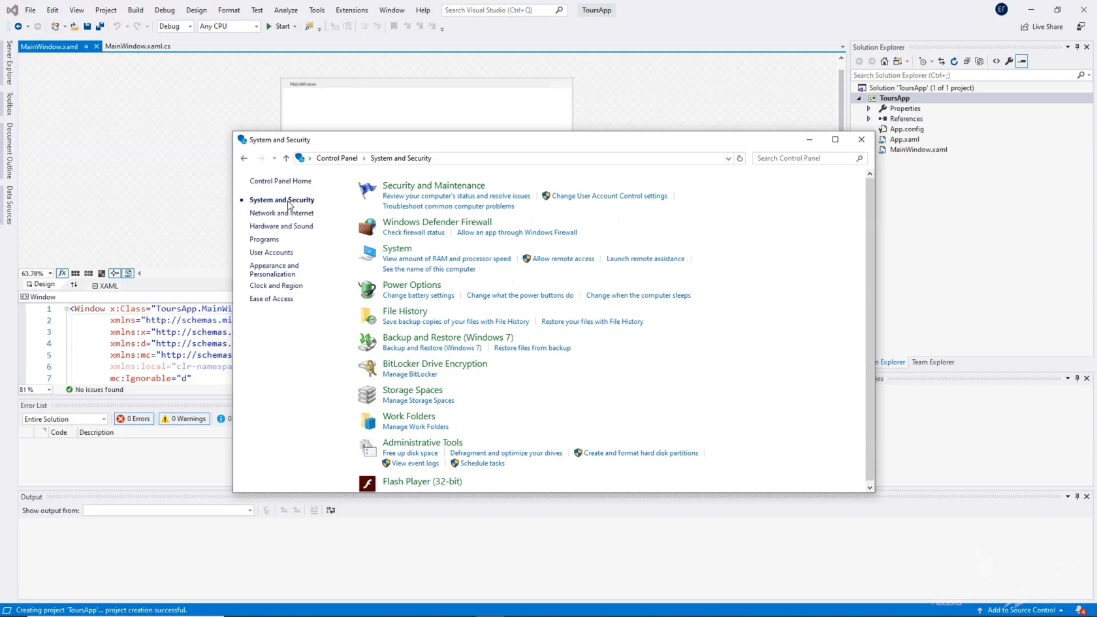
**4.** Заполнить графу «Project name»



При разработке интерфейсов разработчик может использовать две модели: **оконную**или **страничную**(в настоящее время используют чаще)  
 **Оконная модель:**



**Страничная модель**



После того, как проект будет открыт, вам откроется страница Main Window, на которой не будет ничего, кроме чистого листа.

Для того, чтобы начать делать каркас приложения нужно написать следующий код в xaml разметке.

<Window x:Class="Threehours.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:Threehours"

mc:Ignorable="d"

Title="MainWindow" Height="450" Width="800"

ResizeMode="NoResize">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="67\*"/>

<RowDefinition Height="317\*"/>

<RowDefinition Height="50\*"/>

</Grid.RowDefinitions>

<Grid Grid.Row="0" Background="Gray"></Grid>

<Grid Grid.Row="1" >

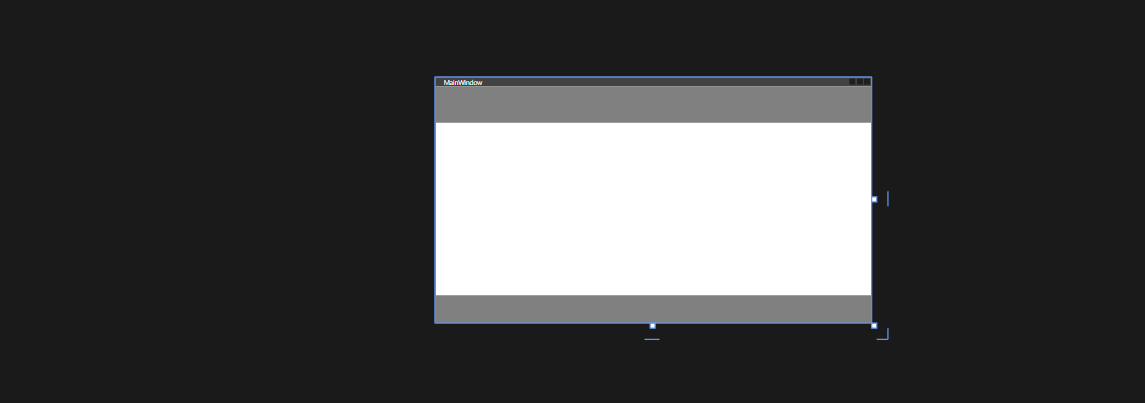
<Frame Name="MainFrame"></Frame>

</Grid>

<Grid Grid.Row="2" Background="Gray"></Grid>

</Grid>

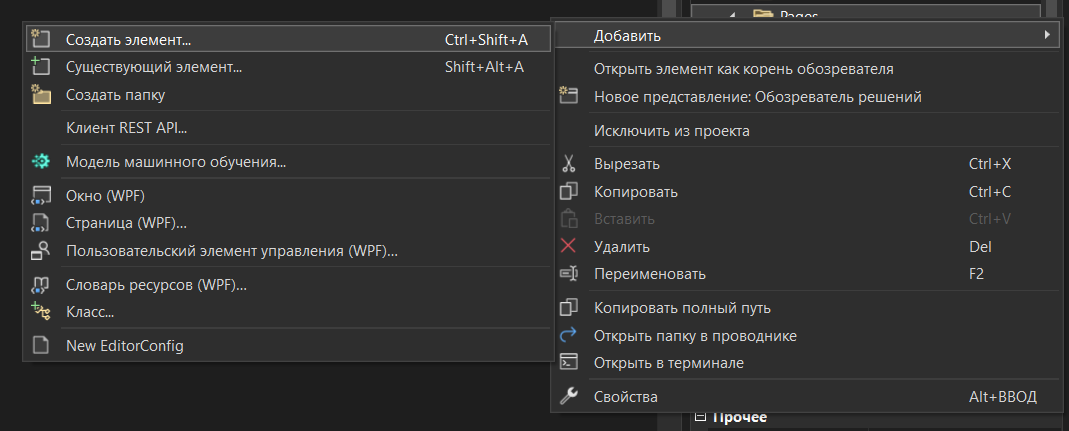
</Window>

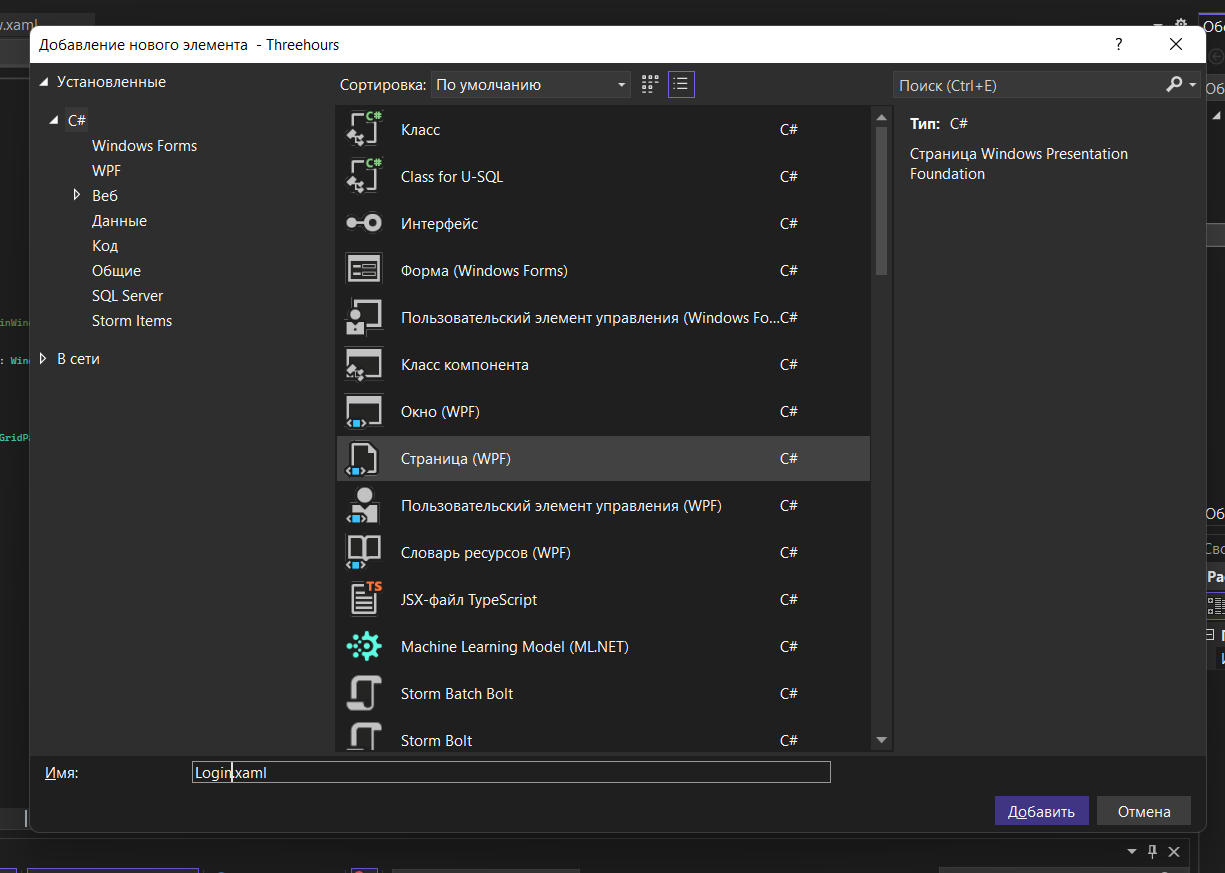
В результате у вас будет отрисовано вот такое окно: 

**Каркас MainWindow**

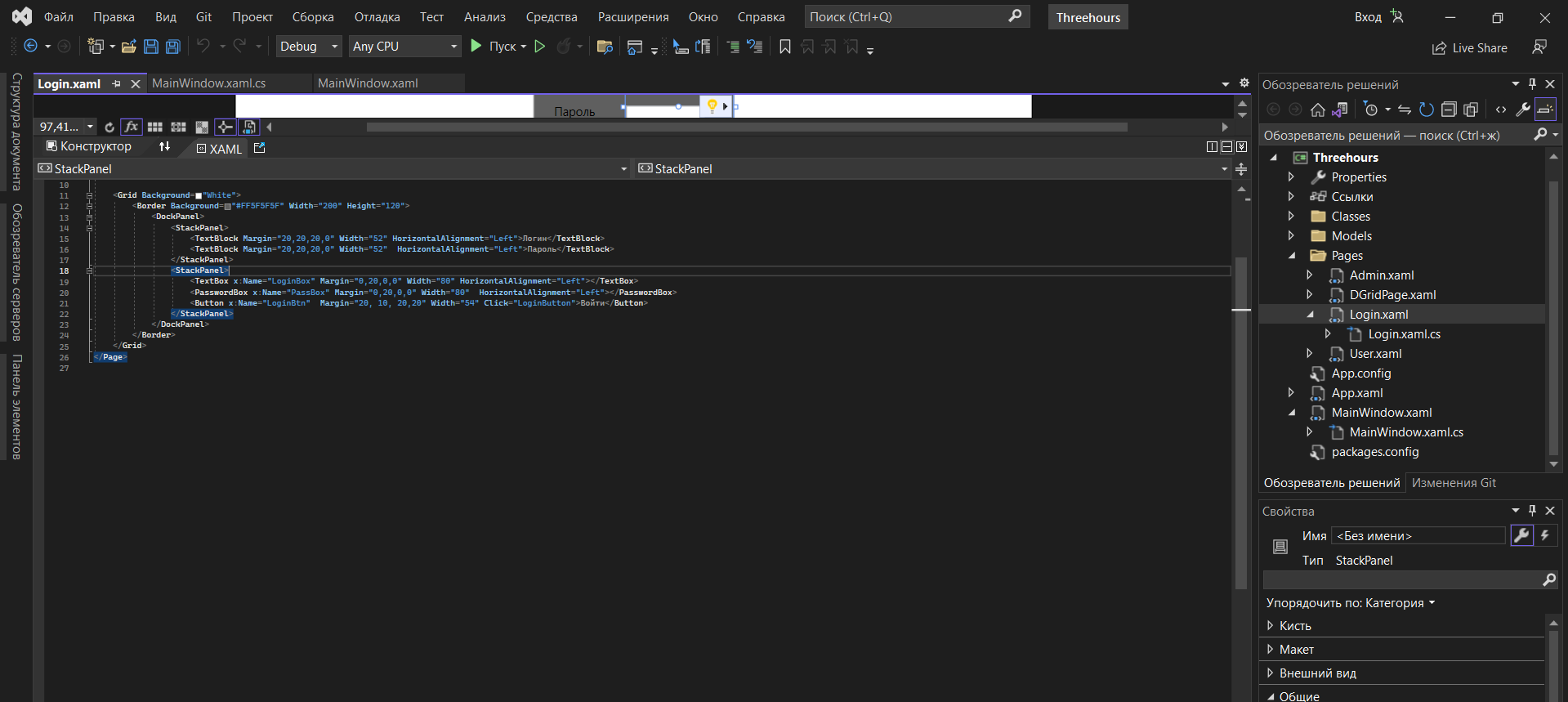
Каркас главной страницы разделен тремя Grid.Row. Нулевой и второй Grid.Row будет раскрашены цветом темы. В среднем же блоке будет размещен

MainFrame. Далее создаем страницу wpf. Для этого нужно нажать правой кнопкой мыши по проекту и выбрать «Добавить элемент» -» «Существующий элемент». В открывшимся окне нужно выбрать элемент «Окно WPF» и назвать его Login.xaml.



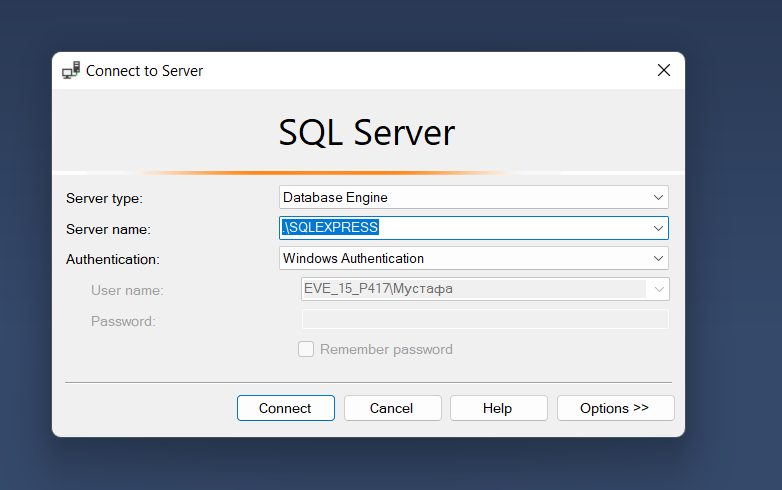


В xaml разметке пишем код для добавления полей ввода и кнопки входа в профиль

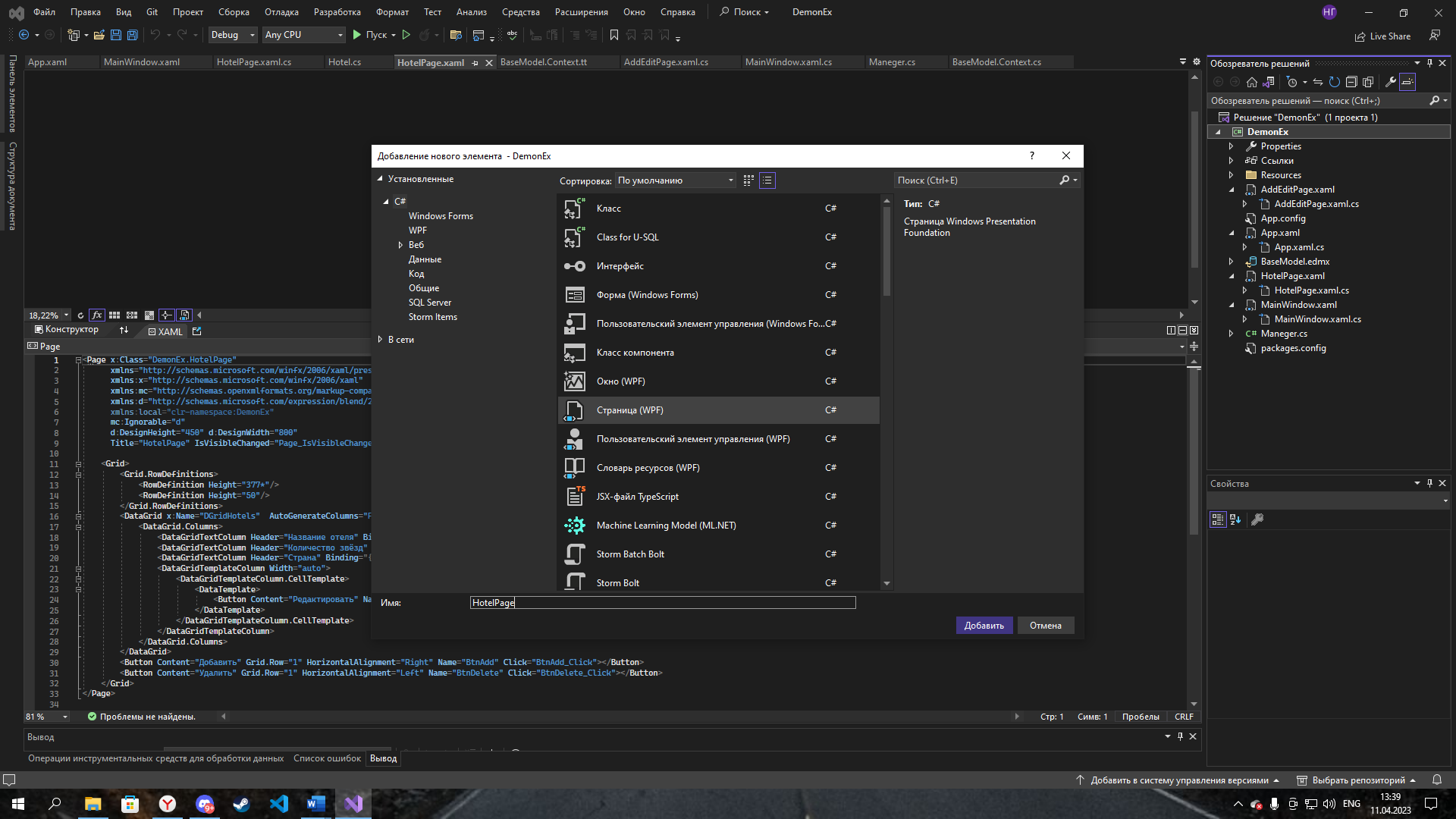


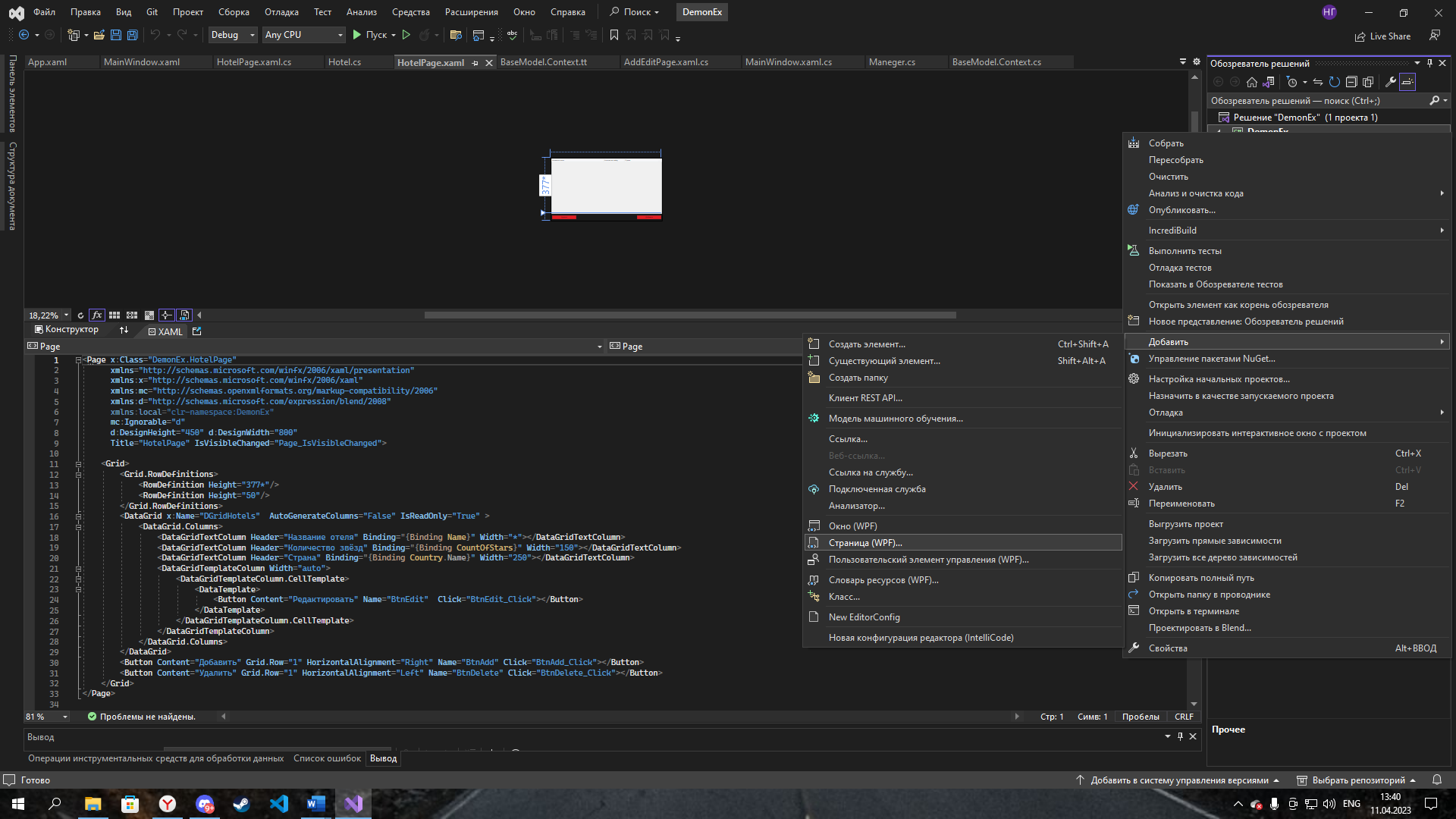
**Разработка БД**

Открываем SQL Server и вводим данные учетной записи.



Далее создаем страницу wpf. Для этого нужно нажать правой кнопкой мыши по проекту и выбрать «Добавить элемент» -» Страница WPF». В открывшимся окне нужно выбрать элемент «Страница WPF» и назвать его По имени вашего проекта. Разделяем страницу на 2 Grid.Row. Нулевой Grid.Row служит для отображения данных из базы данных с помощью DataGrid. А Grid.Row 1 служит для отображения кнопок «Добавления» и «Удаления»





<Page x:Class="DemonEx.HotelPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:DemonEx"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="HotelPage" IsVisibleChanged="Page\_IsVisibleChanged">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="377\*"/>

<RowDefinition Height="50"/>

</Grid.RowDefinitions>

<DataGrid x:Name="DGridHotels" AutoGenerateColumns="False" IsReadOnly="True" >

<DataGrid.Columns>

<DataGridTextColumn Header="Название отеля" Binding="{Binding Name}" Width="\*"></DataGridTextColumn>

<DataGridTextColumn Header="Количество звёзд" Binding="{Binding CountOfStars}" Width="150"></DataGridTextColumn>

<DataGridTextColumn Header="Страна" Binding="{Binding Country.Name}" Width="250"></DataGridTextColumn>

<DataGridTemplateColumn Width="auto">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Button Content="Редактировать" Name="BtnEdit" Click="BtnEdit\_Click"></Button>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<Button Content="Добавить" Grid.Row="1" HorizontalAlignment="Right" Name="BtnAdd" Click="BtnAdd\_Click"></Button>

<Button Content="Удалить" Grid.Row="1" HorizontalAlignment="Left" Name="BtnDelete" Click="BtnDelete\_Click"></Button>

</Grid>

</Page>

В .cs мы добавляет код описанный ниже:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace DemonEx

{

/// <summary>

/// Логика взаимодействия для HotelPage.xaml

/// </summary>

public partial class HotelPage : Page

{

public HotelPage()

{

InitializeComponent();

//DGridHotels.ItemsSource = TourAgenstvoEntities.GetContext().Hotel.ToList();

}

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

{

}

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

{

Maneger.MainFrame.Navigate(new AddEditPage((sender as Button).DataContext as Hotel));

}

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

{

Maneger.MainFrame.Navigate(new AddEditPage(null));

}

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

{

var hotelsForRemoving = DGridHotels.SelectedItems.Cast<Hotel>().ToList();

if (MessageBox.Show($"Вы точно хотите удалить слудующие {hotelsForRemoving.Count()} элементов?","Внимание",

MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

try

{

TourAgenstvoEntities.GetContext().Hotel.RemoveRange(hotelsForRemoving);

TourAgenstvoEntities.GetContext().SaveChanges();

MessageBox.Show("Данные удалены");

DGridHotels.ItemsSource = TourAgenstvoEntities.GetContext().Hotel.ToList();

}

catch (Exception ex)

{

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

}

}

}

private void Page\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e)

{

if (Visibility == Visibility.Visible)

{

TourAgenstvoEntities.GetContext().ChangeTracker.Entries().ToList().ForEach(p => p.Reload());

DGridHotels.ItemsSource = TourAgenstvoEntities.GetContext().Hotel.ToList();

}

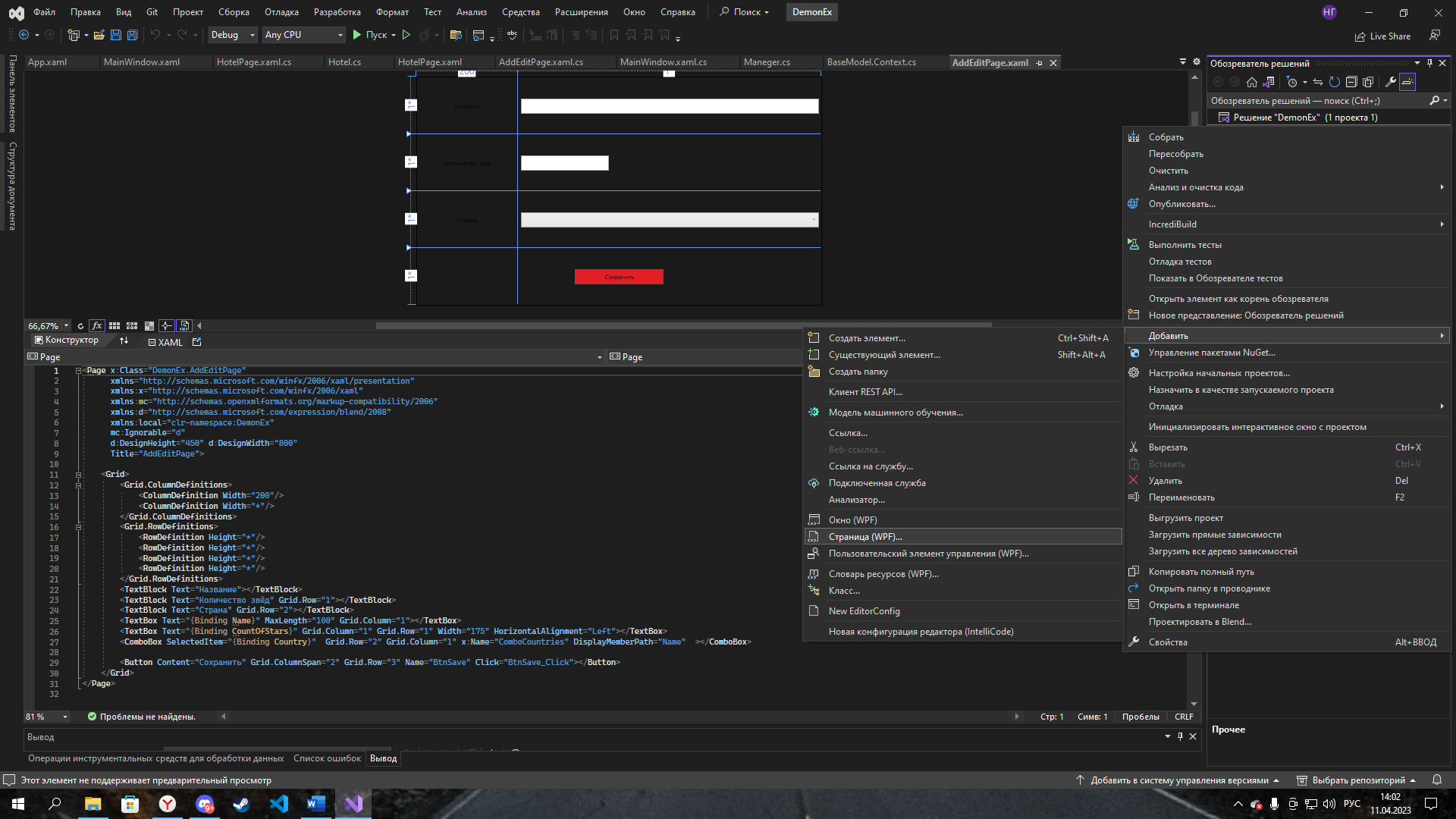
}

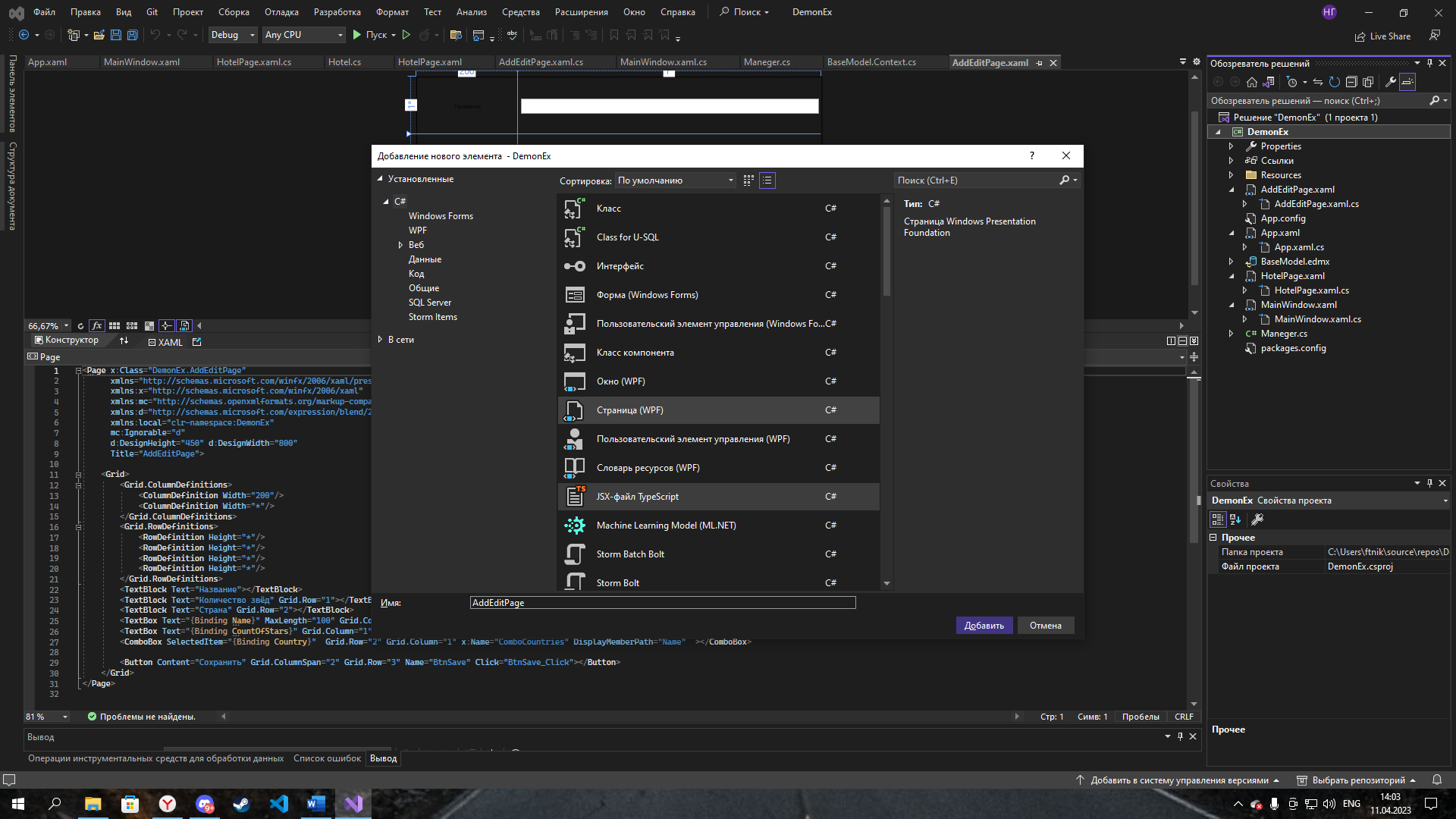
}

}

После того как мы вывели данные в Data.Grid.Мы создаем новую страницу для этого нам нужно: нажать правой кнопкой мыши по проекту и выбрать «Добавить элемент» -» Страница WPF». В открывшимся окне нужно выбрать элемент «Страница WPF» и назвать его По имени вашего проекта.

Делим нашу страницу на 2 столбца и 4 строчки. В некоторые столбцы добавляем TextBox для редактирования данных. В Grid.Row 1 добавляем TextBlock и ComboBox для добавления и редактирования данных.





<Page x:Class="DemonEx.AddEditPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:DemonEx"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="AddEditPage">

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="200"/>

<ColumnDefinition Width="\*"/>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<TextBlock Text="Название"></TextBlock>

<TextBlock Text="Количество звёд" Grid.Row="1"></TextBlock>

<TextBlock Text="Страна" Grid.Row="2"></TextBlock>

<TextBox Text="{Binding Name}" MaxLength="100" Grid.Column="1"></TextBox>

<TextBox Text="{Binding CountOfStars}" Grid.Column="1" Grid.Row="1" Width="175" HorizontalAlignment="Left"></TextBox>

<ComboBox SelectedItem="{Binding Country}" Grid.Row="2" Grid.Column="1" x:Name="ComboCountries" DisplayMemberPath="Name" ></ComboBox>

<Button Content="Сохранить" Grid.ColumnSpan="2" Grid.Row="3" Name="BtnSave" Click="BtnSave\_Click"></Button>

</Grid>

</Page>

В .cs мы добавляет код описанный ниже:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace DemonEx

{

/// <summary>

/// Логика взаимодействия для AddEditPage.xaml

/// </summary>

public partial class AddEditPage : Page

{

private Hotel \_currentHotel = new Hotel();

public AddEditPage(Hotel selectedHotel)

{

InitializeComponent();

if(selectedHotel != null)

\_currentHotel = selectedHotel;

DataContext = \_currentHotel;

ComboCountries.ItemsSource = TourAgenstvoEntities.GetContext().Country.ToList();

}

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

{

StringBuilder errors = new StringBuilder();

if (string.IsNullOrWhiteSpace(\_currentHotel.Name))

errors.AppendLine("Укажите название отеля");

if (\_currentHotel.CountOfStars < 1 || \_currentHotel.CountOfStars > 5)

errors.AppendLine("Количество звёзд - число от 1 до 5");

if (\_currentHotel.Country == null)

errors.AppendLine("Выберите страну");

if (errors.Length > 0)

{

MessageBox.Show(errors.ToString());

return;

}

if (\_currentHotel.Id == 0)

TourAgenstvoEntities.GetContext().Hotel.Add(\_currentHotel);

try

{

TourAgenstvoEntities.GetContext().SaveChanges();

MessageBox.Show("Информация сохранена!");

Maneger.MainFrame.GoBack();

}

catch (Exception ex)

{

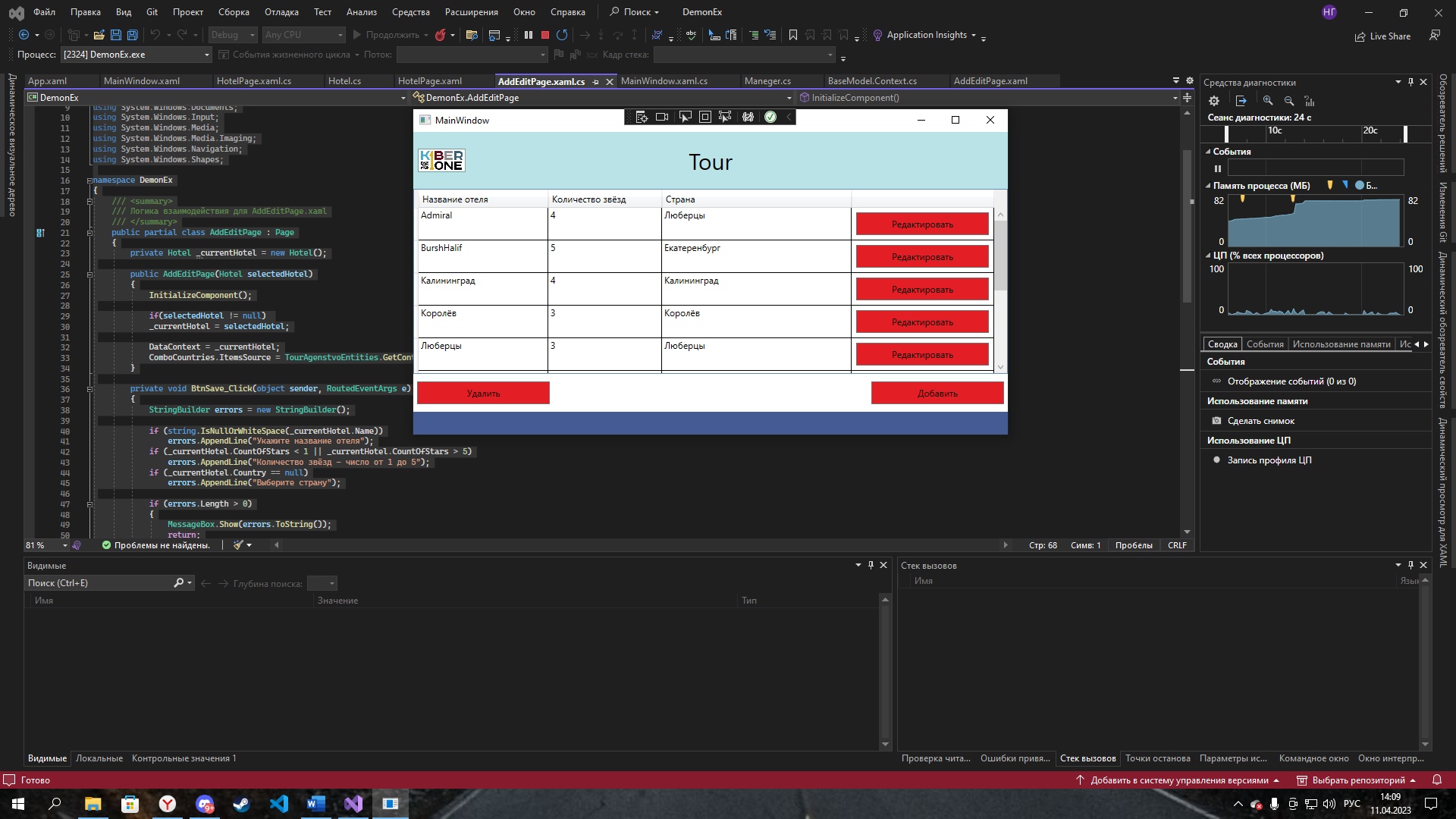
MessageBox.Show(ex.Message.ToString());

}

}

}

}





**Заключение**

В процессе выполнения данной курсовой работы поставленные задачи и цель были выполнены. Представлены общие сведения о технологии .

Было осуществлено создание каркаса и дизайн приложения, так же было осуществлено подключение к базе данных посредством Microsoft SQL Server Database File.Wpf приложение позволяет добавлять и удалять информацию об пропусках сотрудников, причине их пропуска и даты прибывания их на больничном.