1. Создание проекта

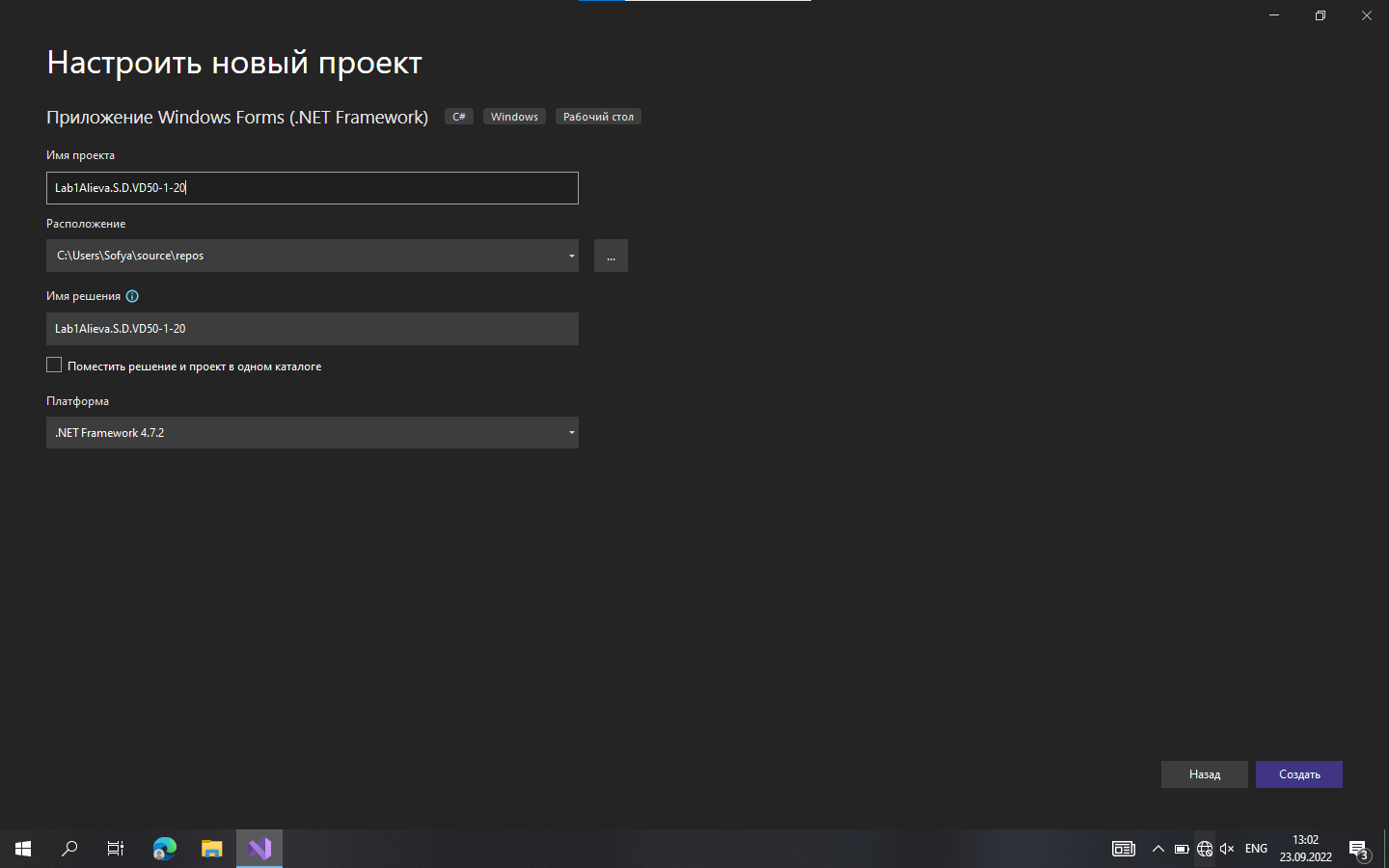


Рисунок 1 - Создание проекта

1. Созданный проект

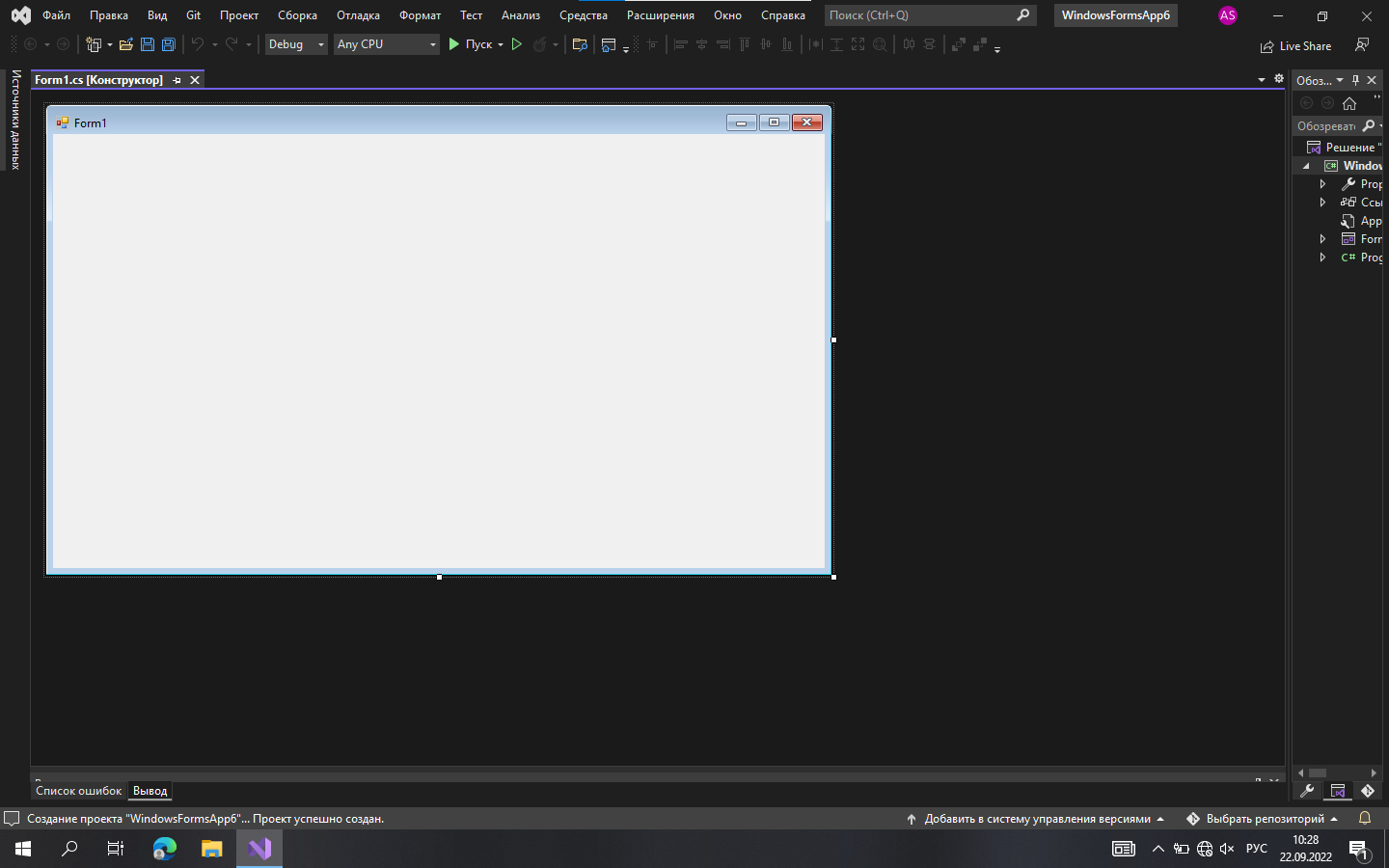


Рисунок 2 - Созданный проект

1. Удаление Form1

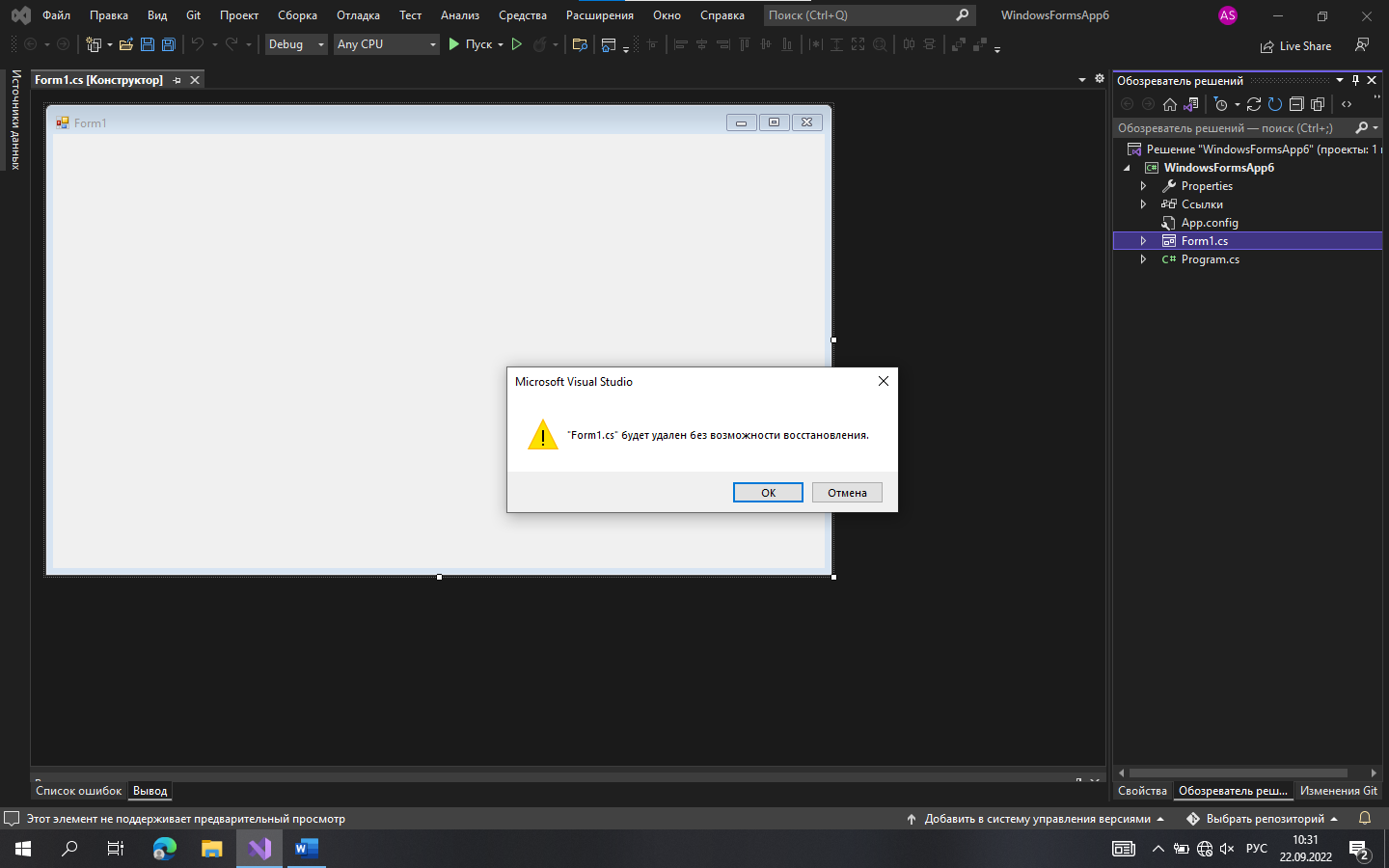


Рисунок 3 - Удаление формы

1. Создание LoginForm

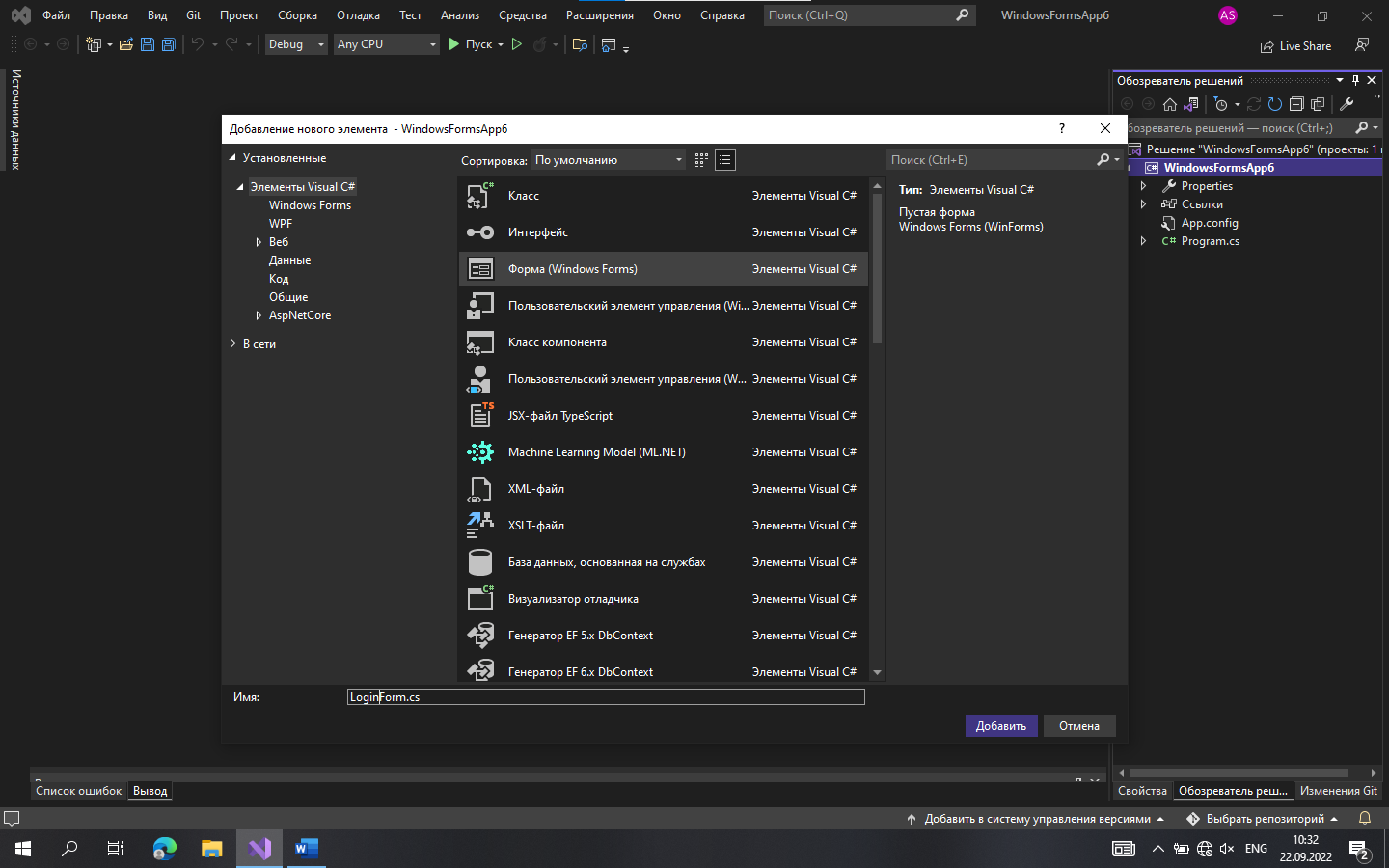


Рисунок 4 - Создание формы

1. Открываем панель элементов

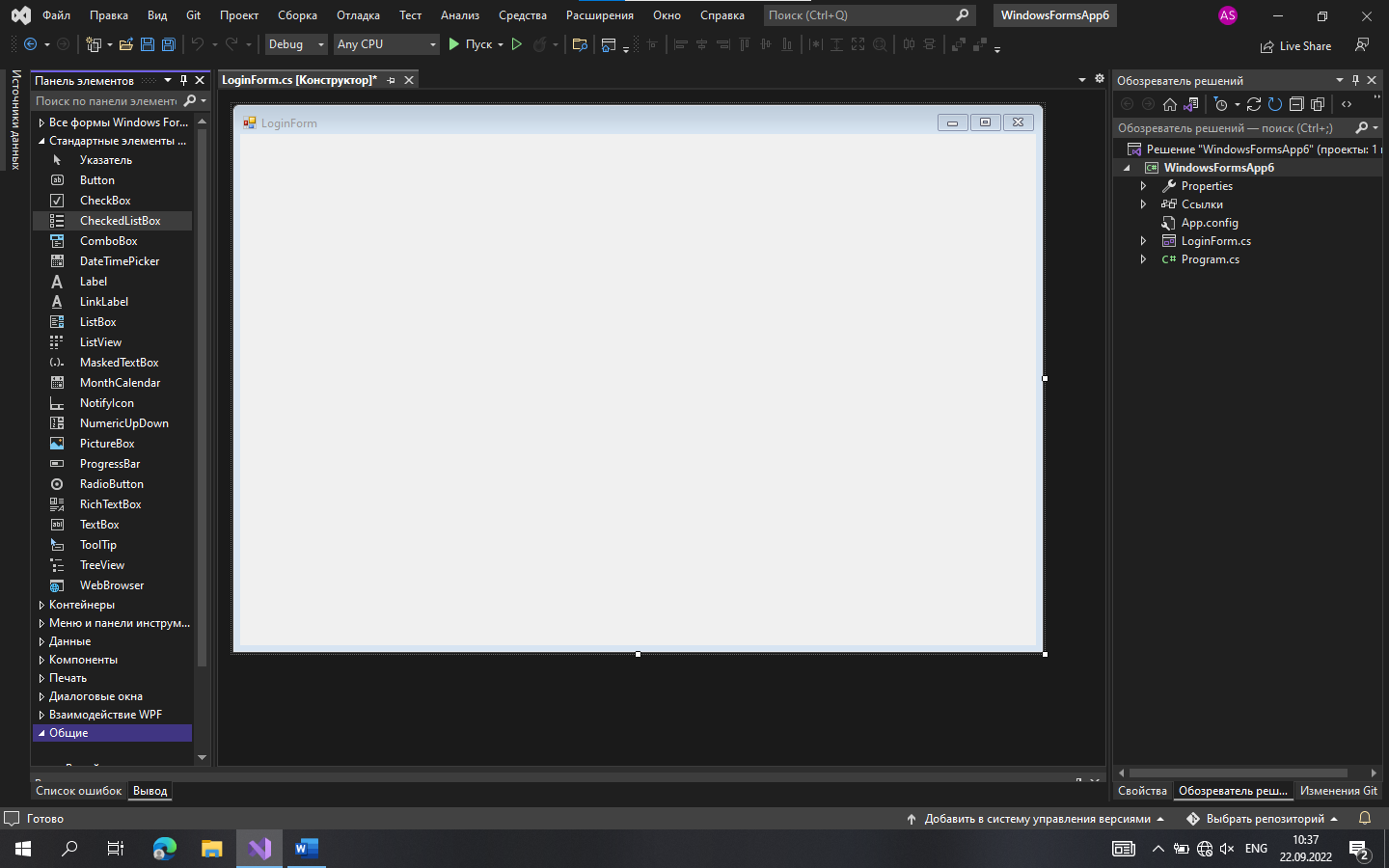


Рисунок 5 - Панель элементов

1. Выбираем в панели элементов элемент panel

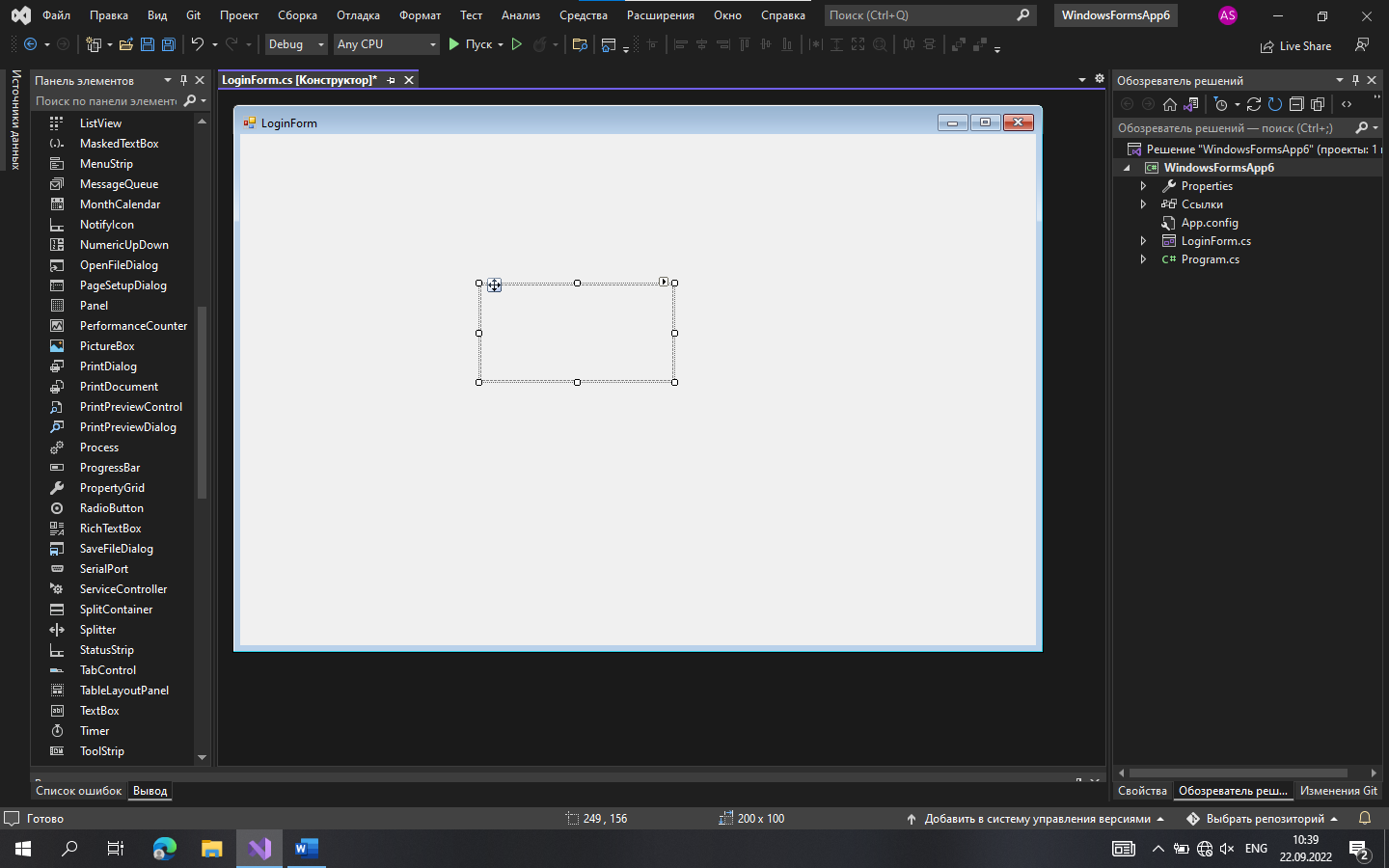


Рисунок 6 - Элемент panel

1. Закрепляем panel в родительском контейнере

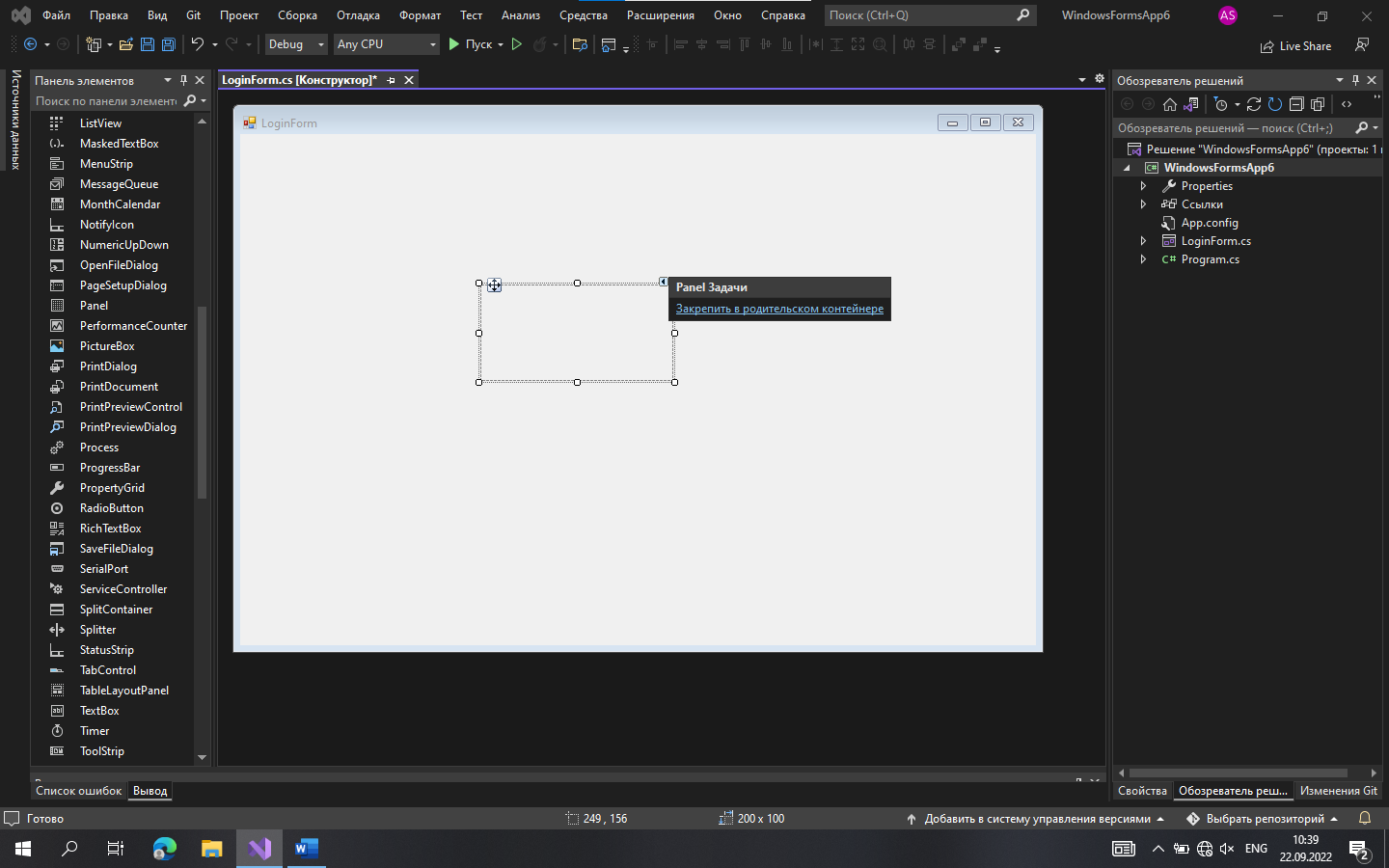


Рисунок 7 – Закрепление panel

1. Меняем значения свойства BackColor элемента Panel на 17;24;34

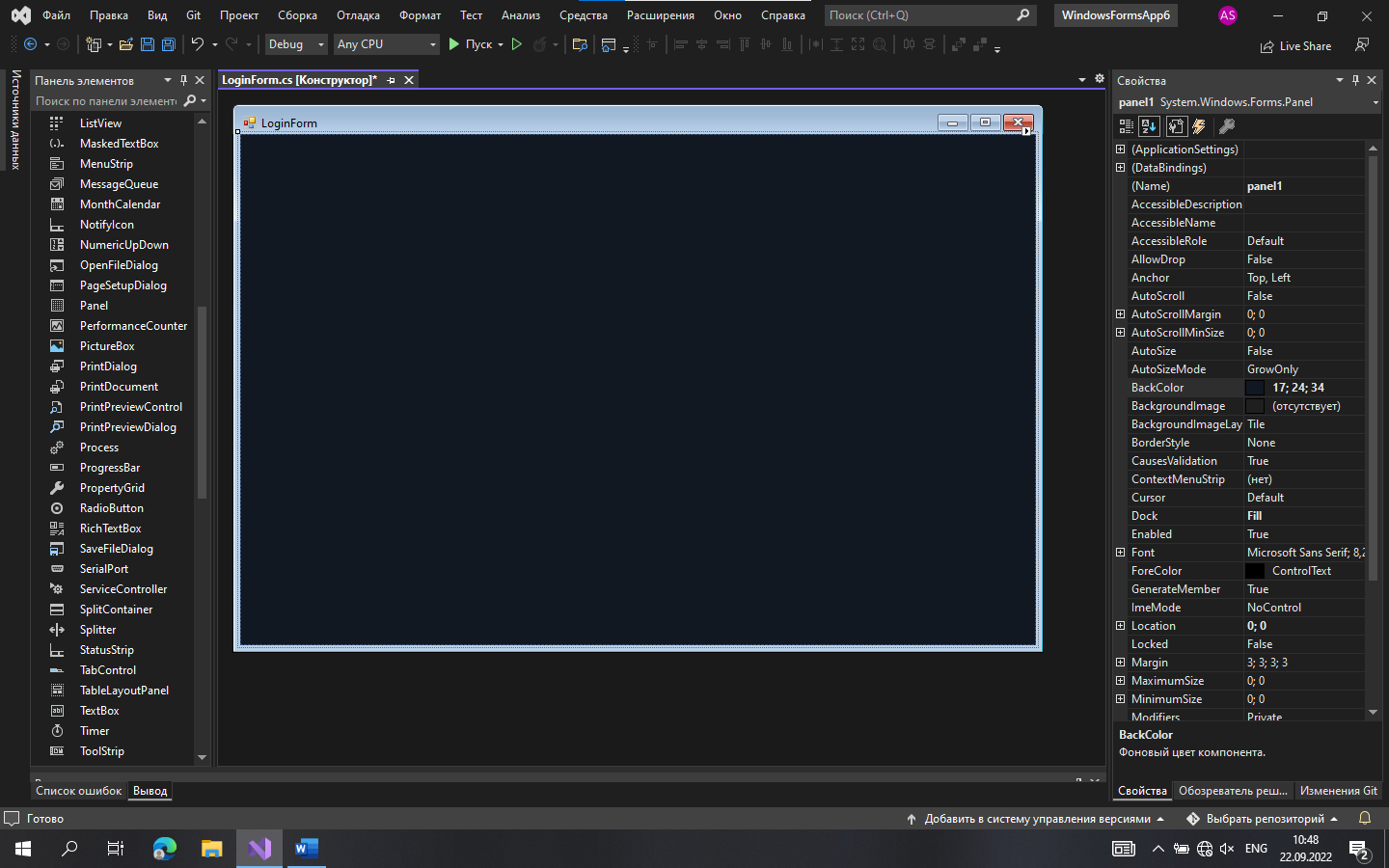


Рисунок 8 - Цвет панели

1. Создаем новую панель и располагаем ее сверху формы по ширине

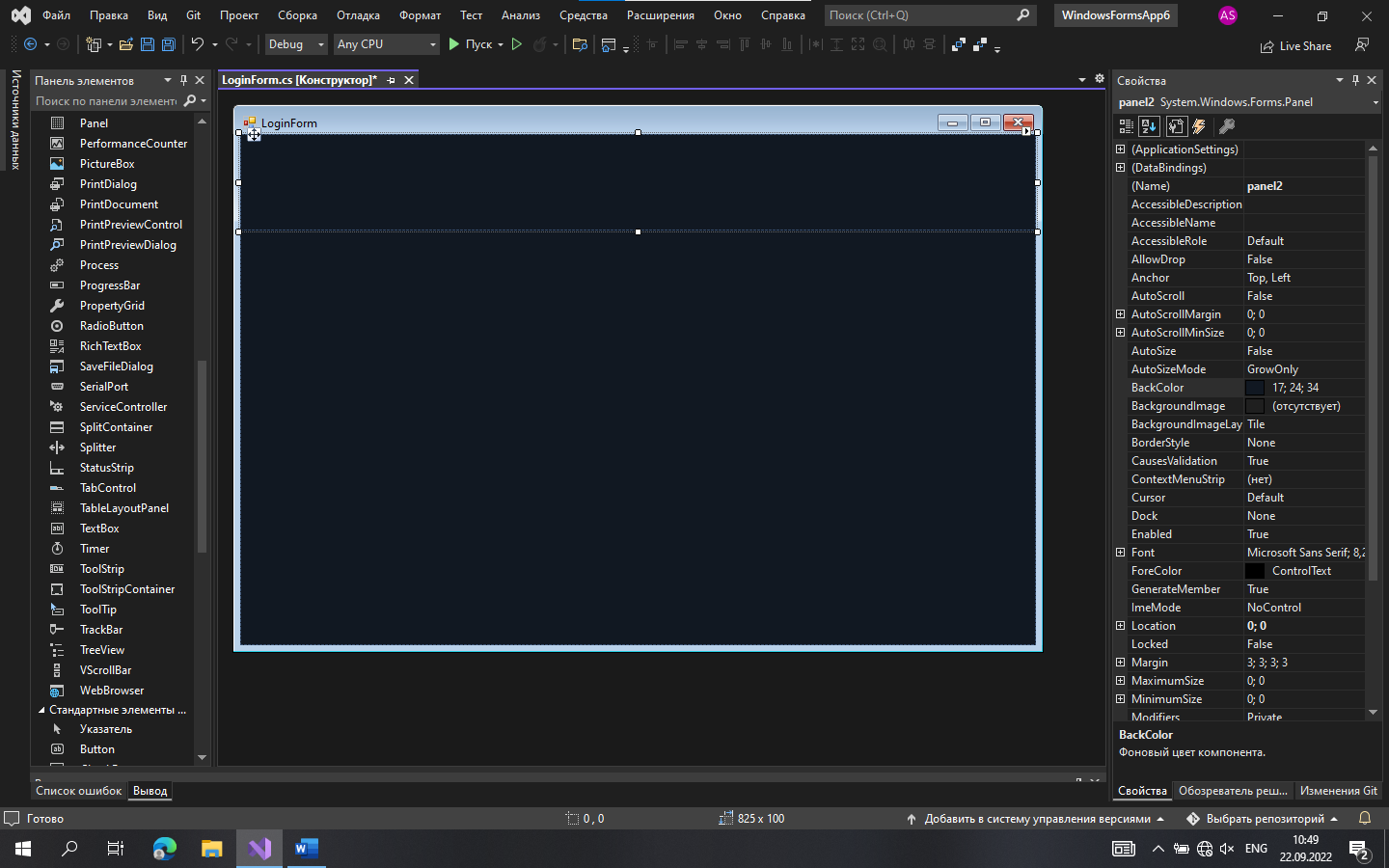


Рисунок 9 - Создание второй панели

1. Меняем значения свойства BackColor элемента Panel на 255;183;19

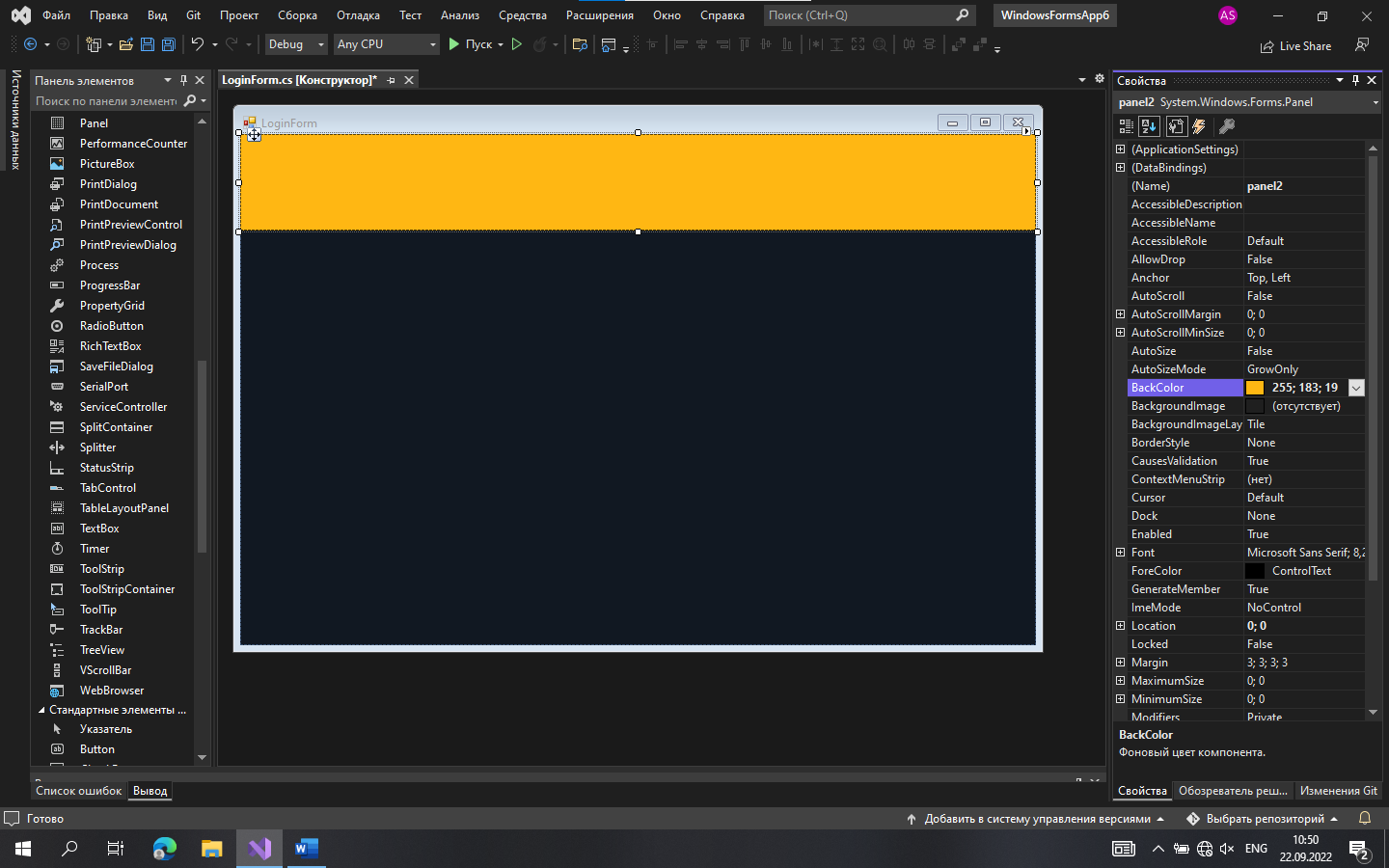


Рисунок 10 - Цвет панели

1. Создаем папку с изображениями под названием image, копируем изображения и вставляем в эту папку.

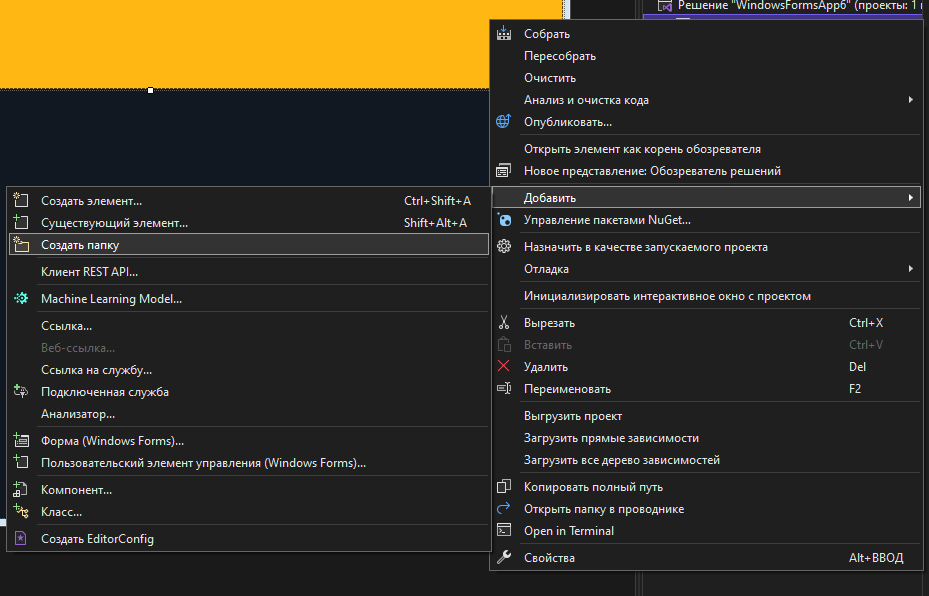


Рисунок 11 - Создание папки

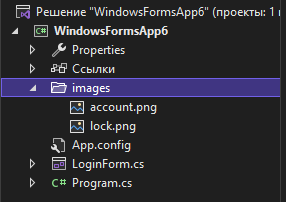


Рисунок 12 - Созданная папка

1. Создаем label и настраиваем шрифт и текст

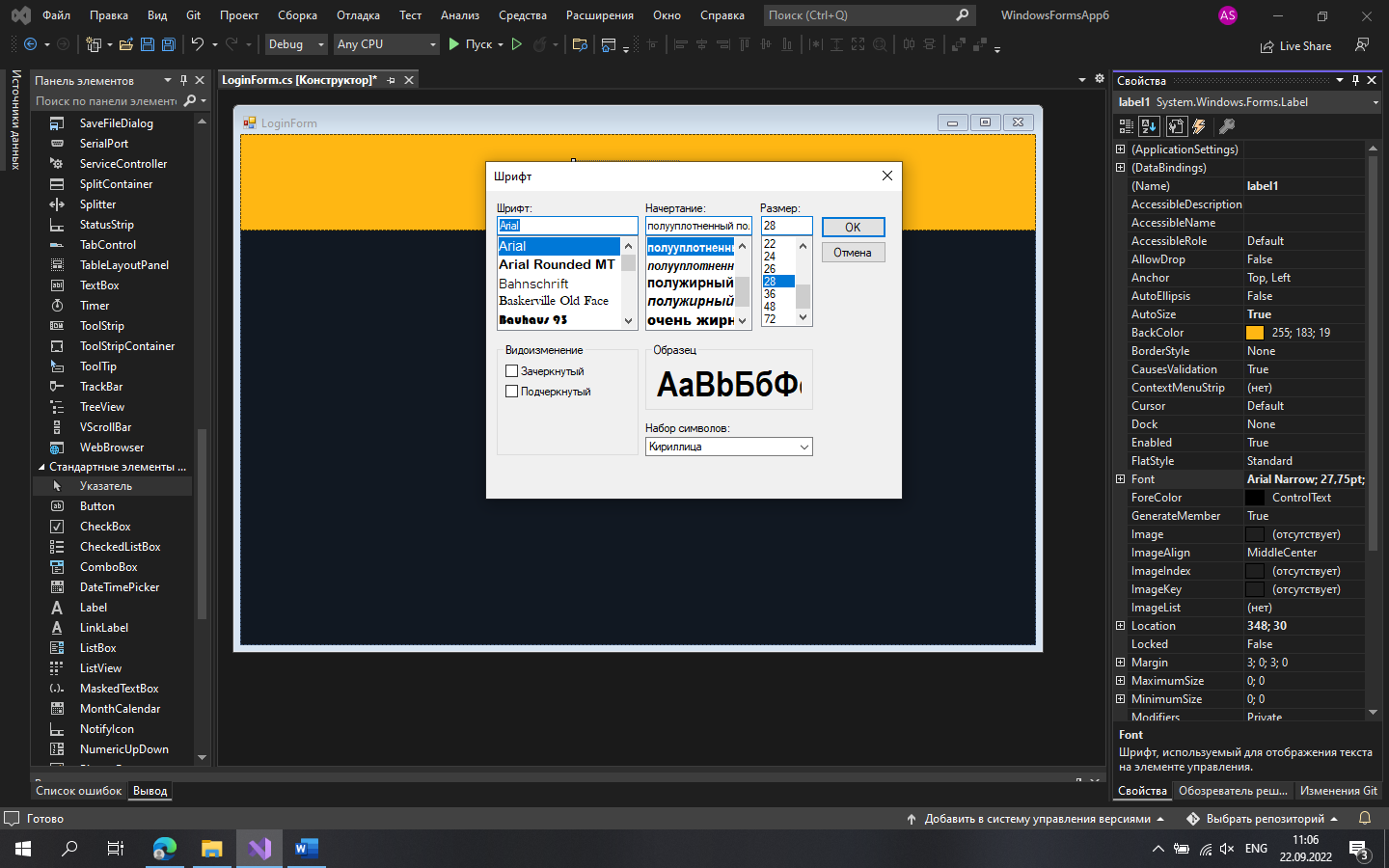


Рисунок 13 - Настройки шрифта

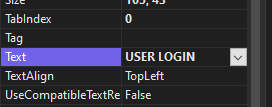


Рисунок 14 - Текст label

1. Меняем свойство dock на значение fill, и textAlign на MiddleCenter

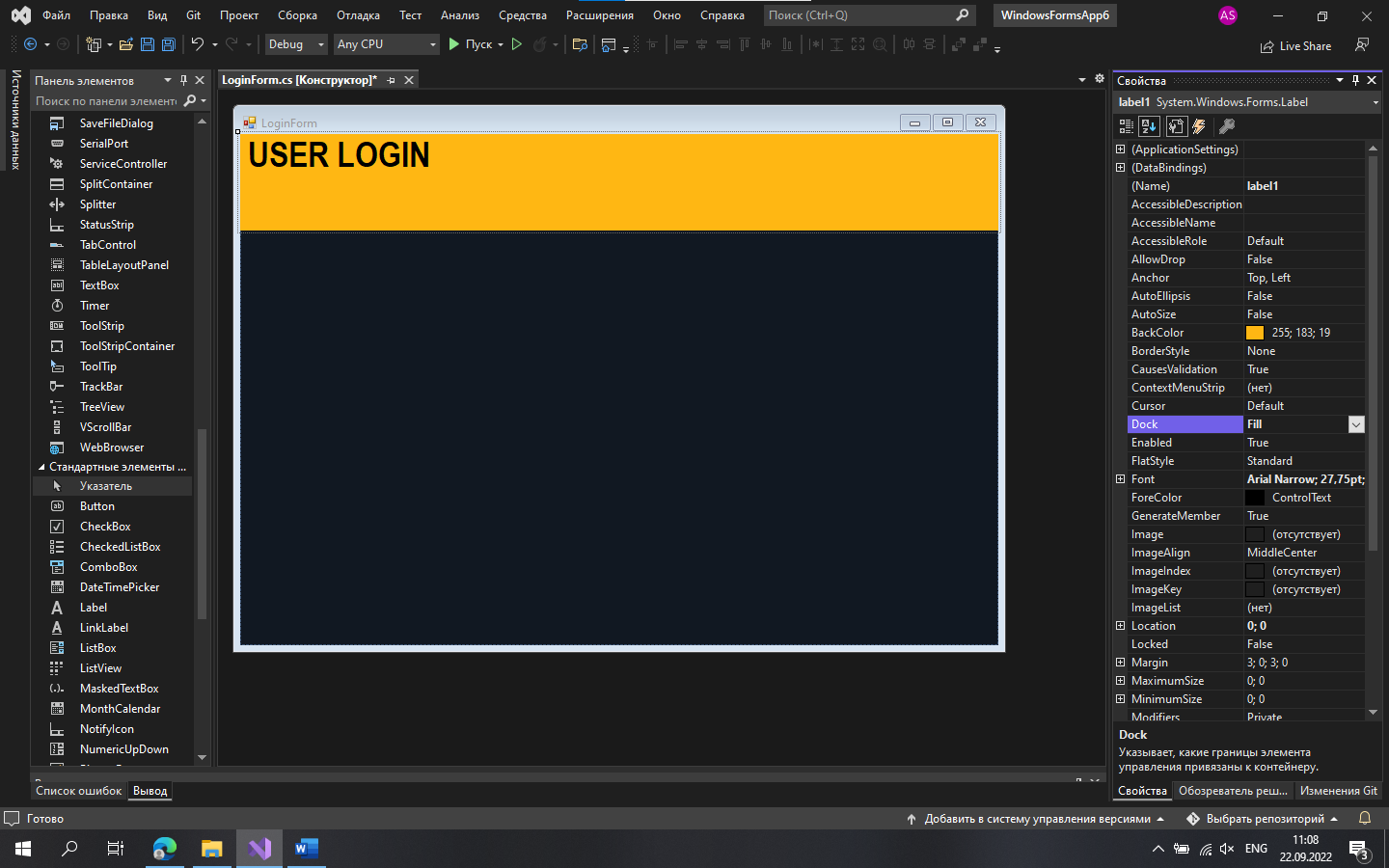


Рисунок 15 - Свойство Dock

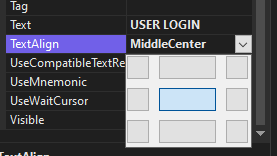


Рисунок 16 - Свойство TextAlign

1. Создаем еще один label, который будет кнопкой закрытия приложения, меняем текст на х и имя на labelClose, а так же настраиваем шрфит

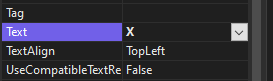


Рисунок 17 - Текст lable



Рисунок 18 - Имя label

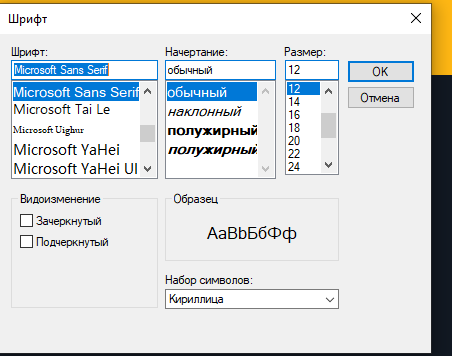


Рисунок 19 - Настройки шрифта

1. Заходим в Program.cs и меняем Form1 на LoginForm, чтобы при включении приложения загружалась LoginForm

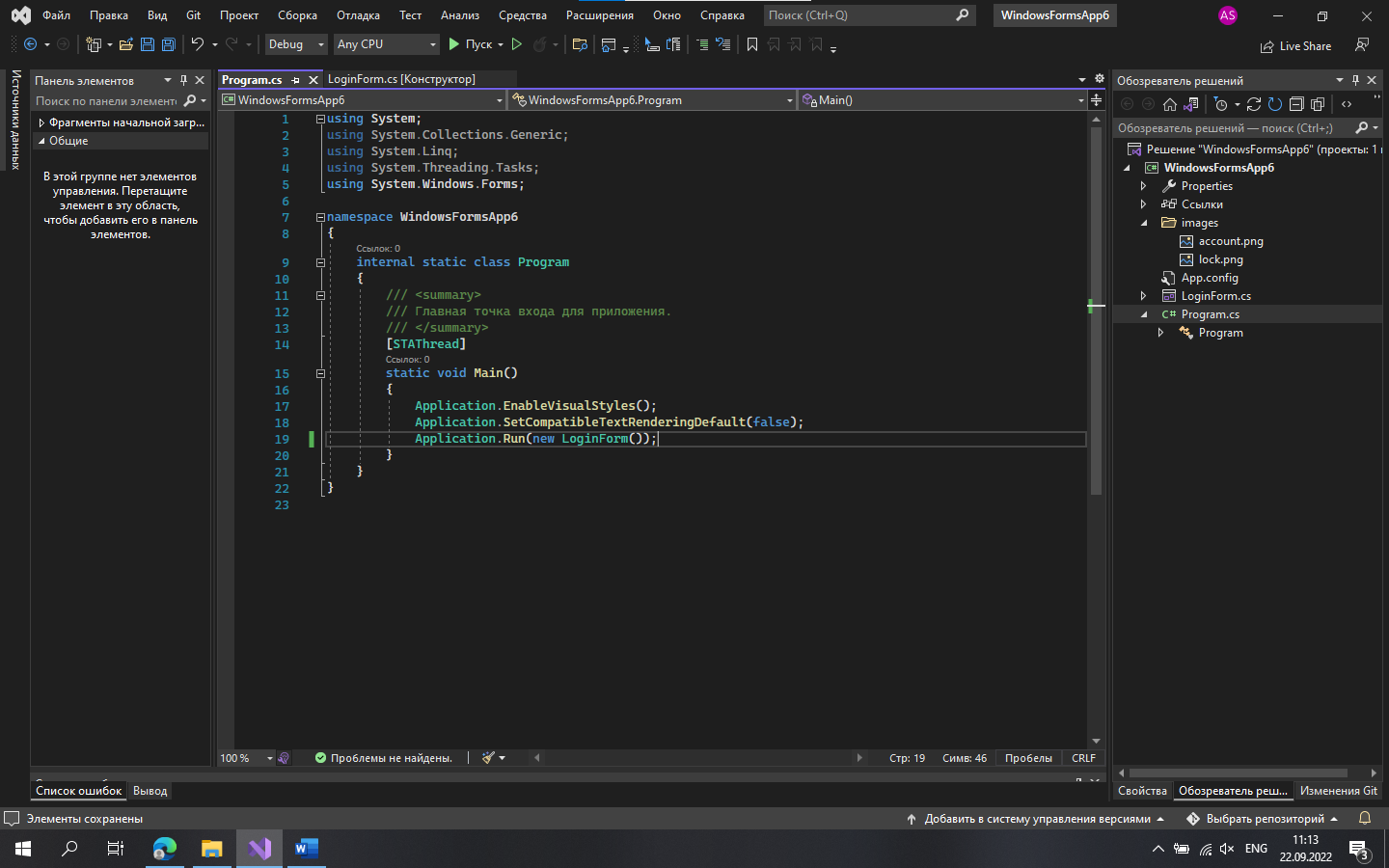


Рисунок 20 - Program.cs

1. Промежуточный результат

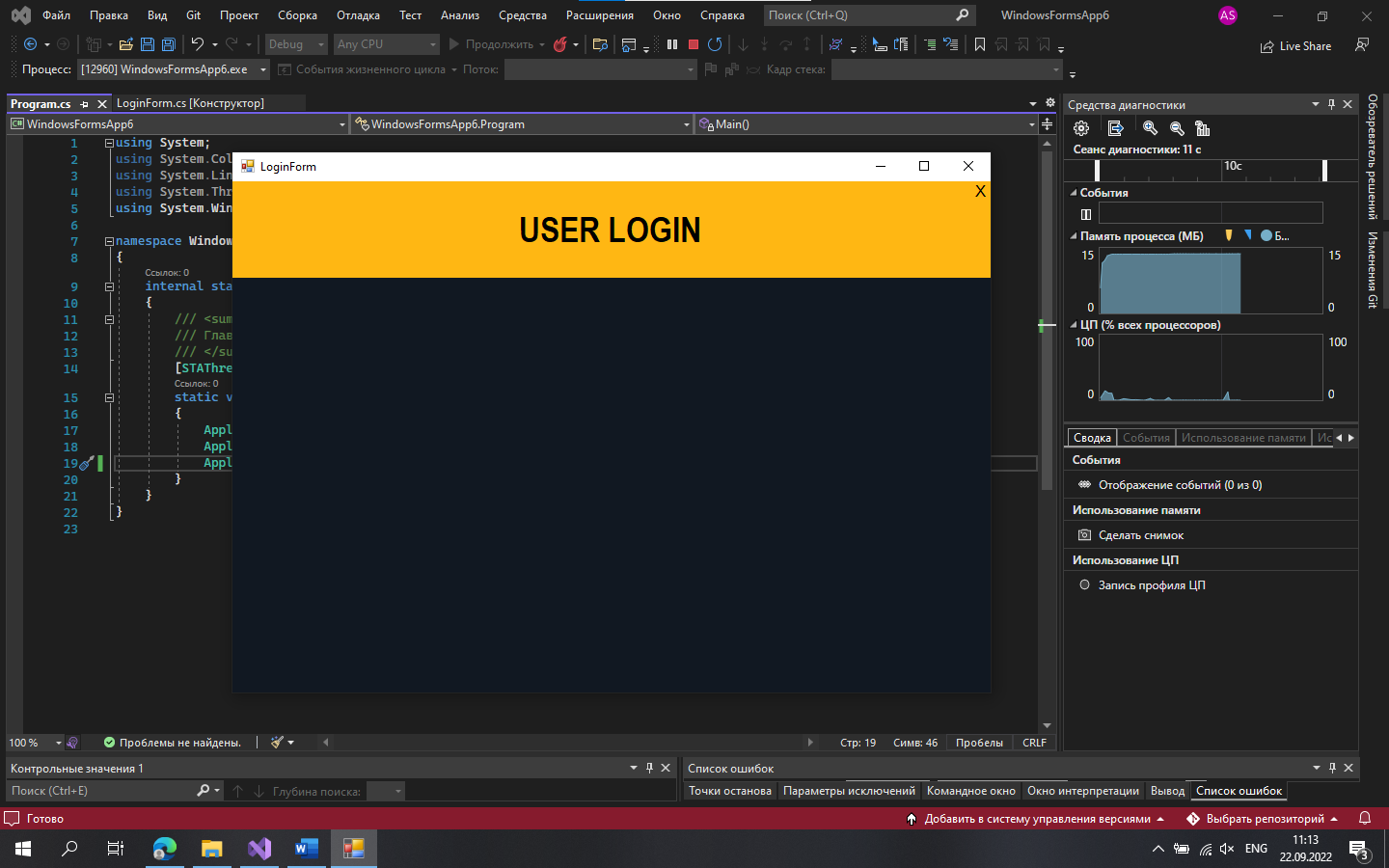


Рисунок 21 - Промежуточный результат

1. Заходим на панель элементов и создаем pictureBox, задаем ему размер и загружаем в него изображение из папки image

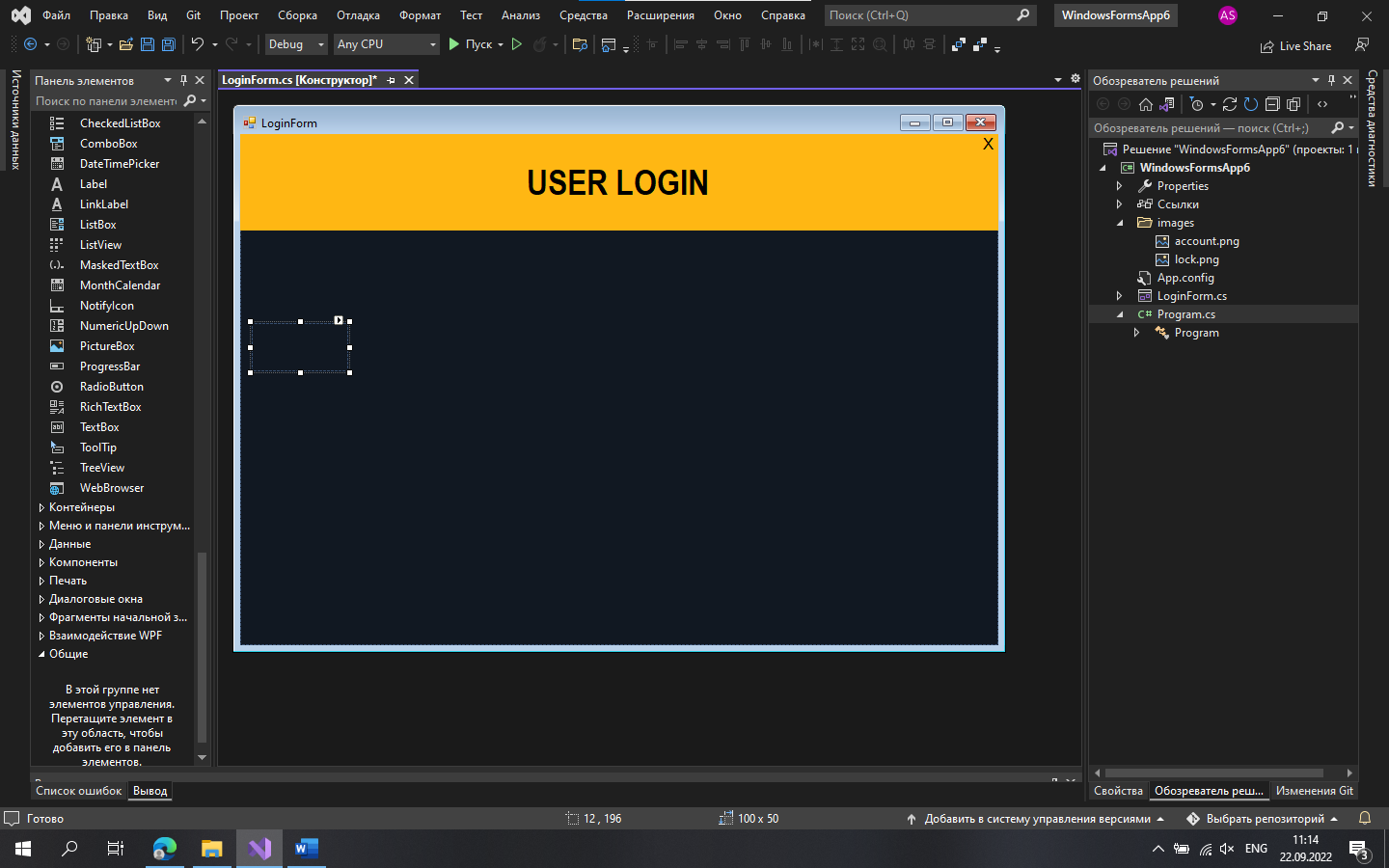


Рисунок 22 - pictureBox



Рисунок 23 - Размер pictureBox

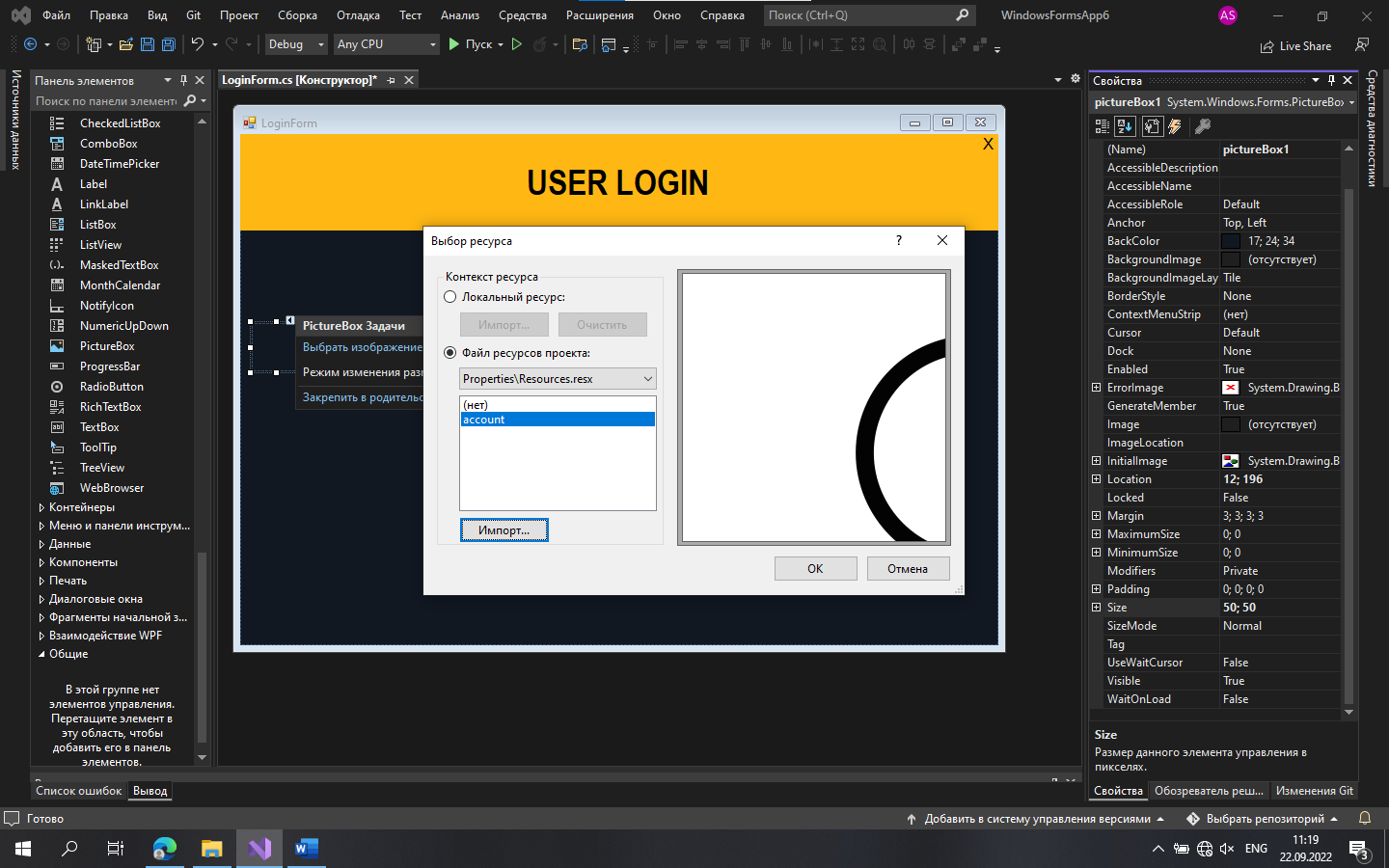


Рисунок 24 - Выбор изображения для pictureBox

1. Готовый pictureBox

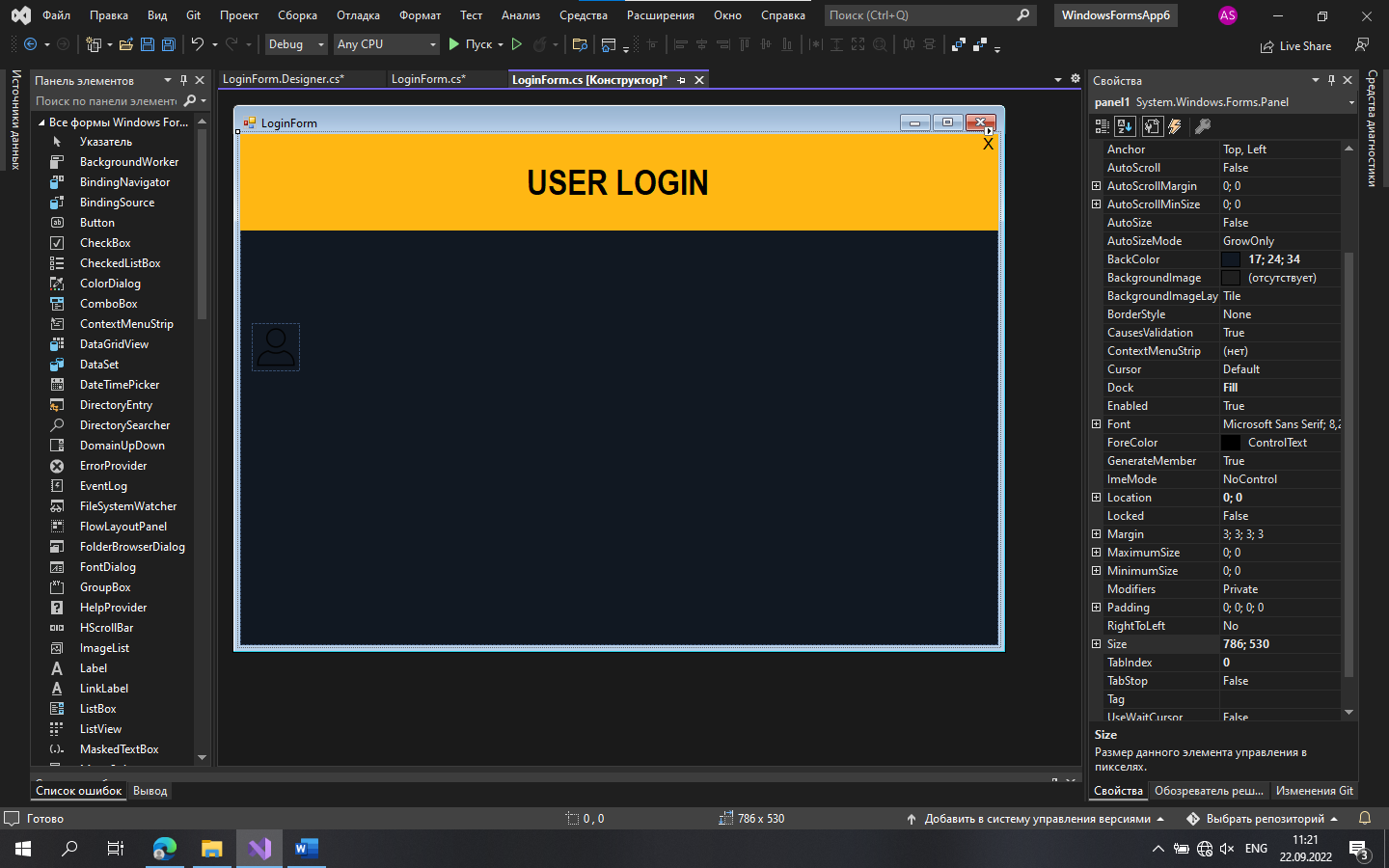


Рисунок 25 - Готовый pictureBox

1. Создаем textbox, в свойстве MultiLine ставим галочку и задаем размеры

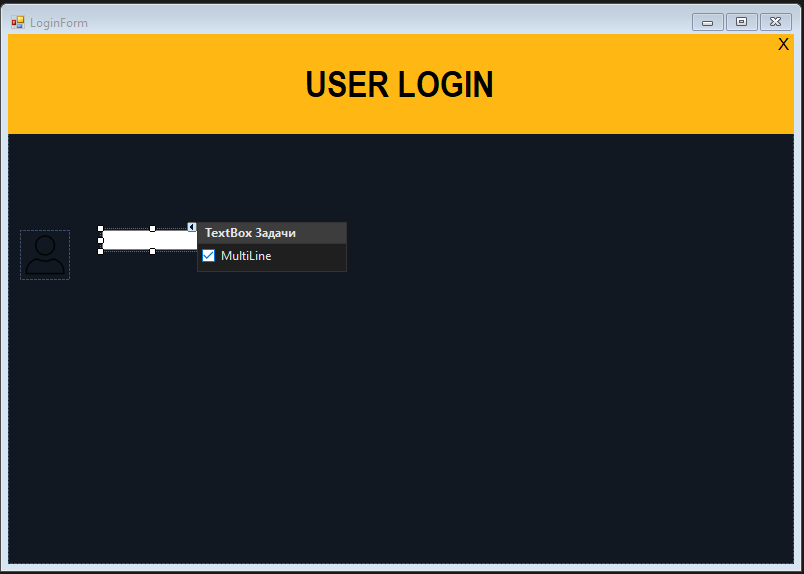


Рисунок 26 - Свойство MultiLine



Рисунок 27 - Размеры textbox

1. Копируем сделанный pictureBox и textbox и меняем изображение в pictureBox на замок

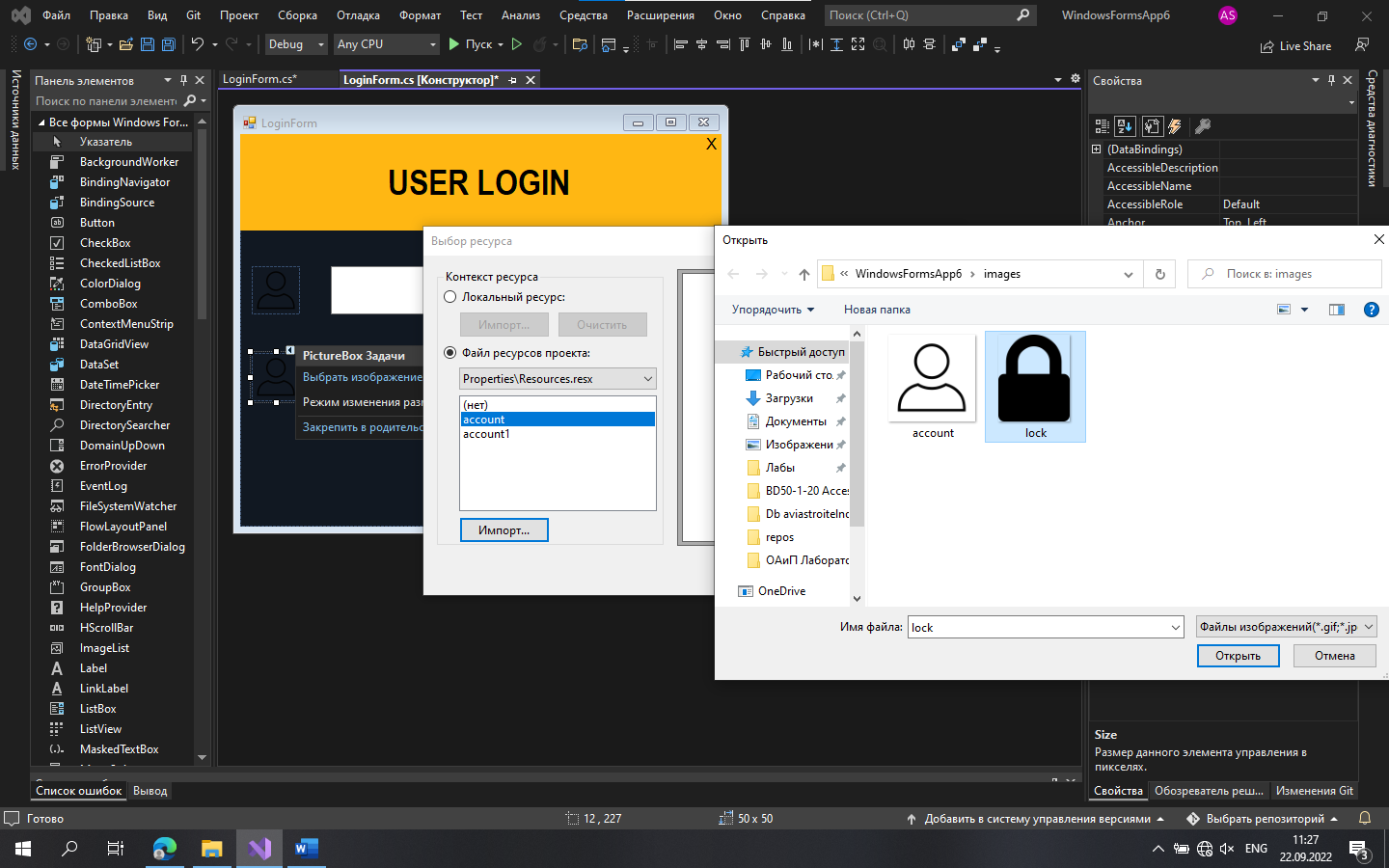


Рисунок 28 - Выбор изображения для pictureBox

1. Промежуточный результат



Рисунок 29 - Промежуточный результат

1. Создаем кнопку и настраиваем следующие свойства: BackColor и FlatStyle

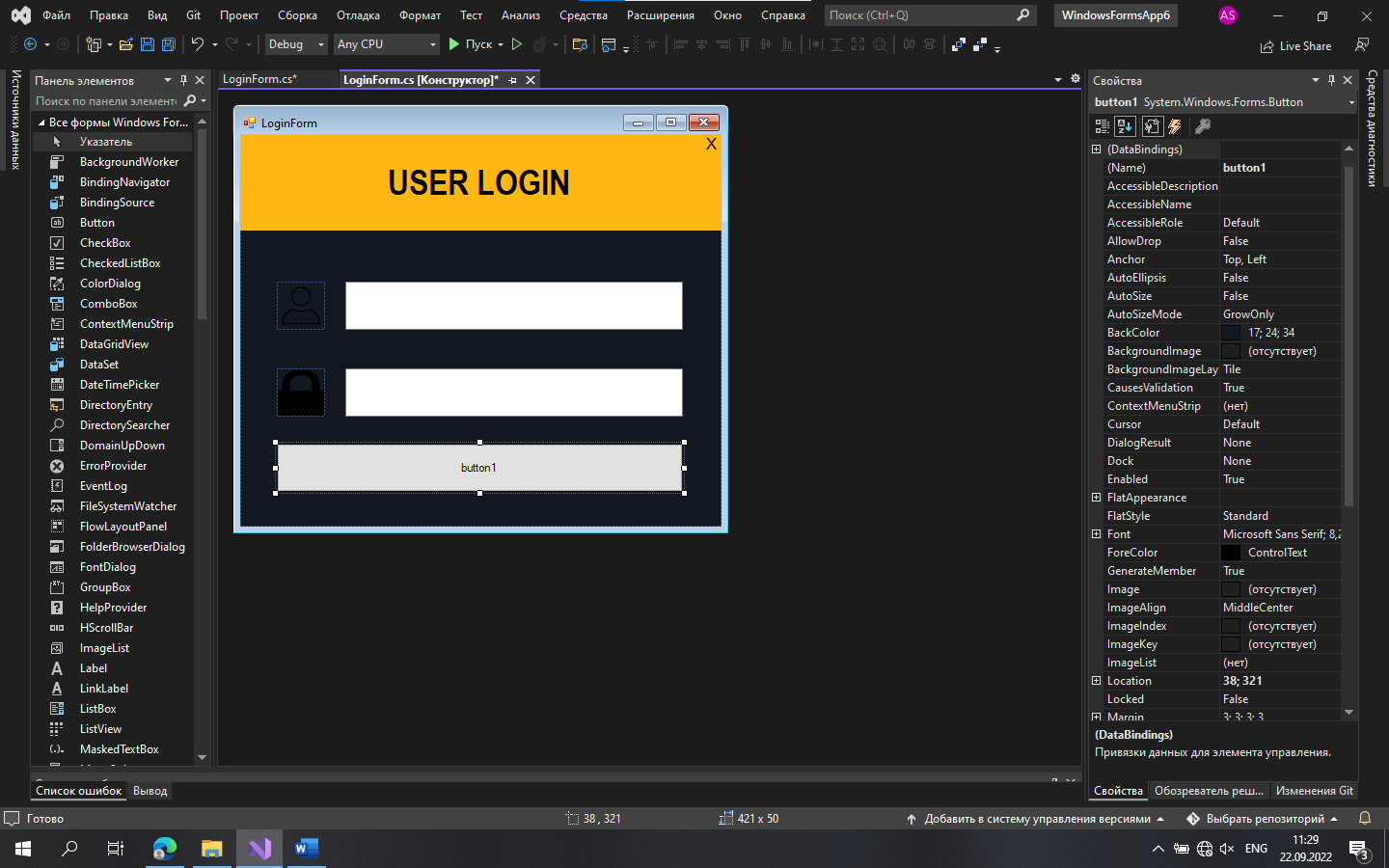


Рисунок 30 - Кнопка

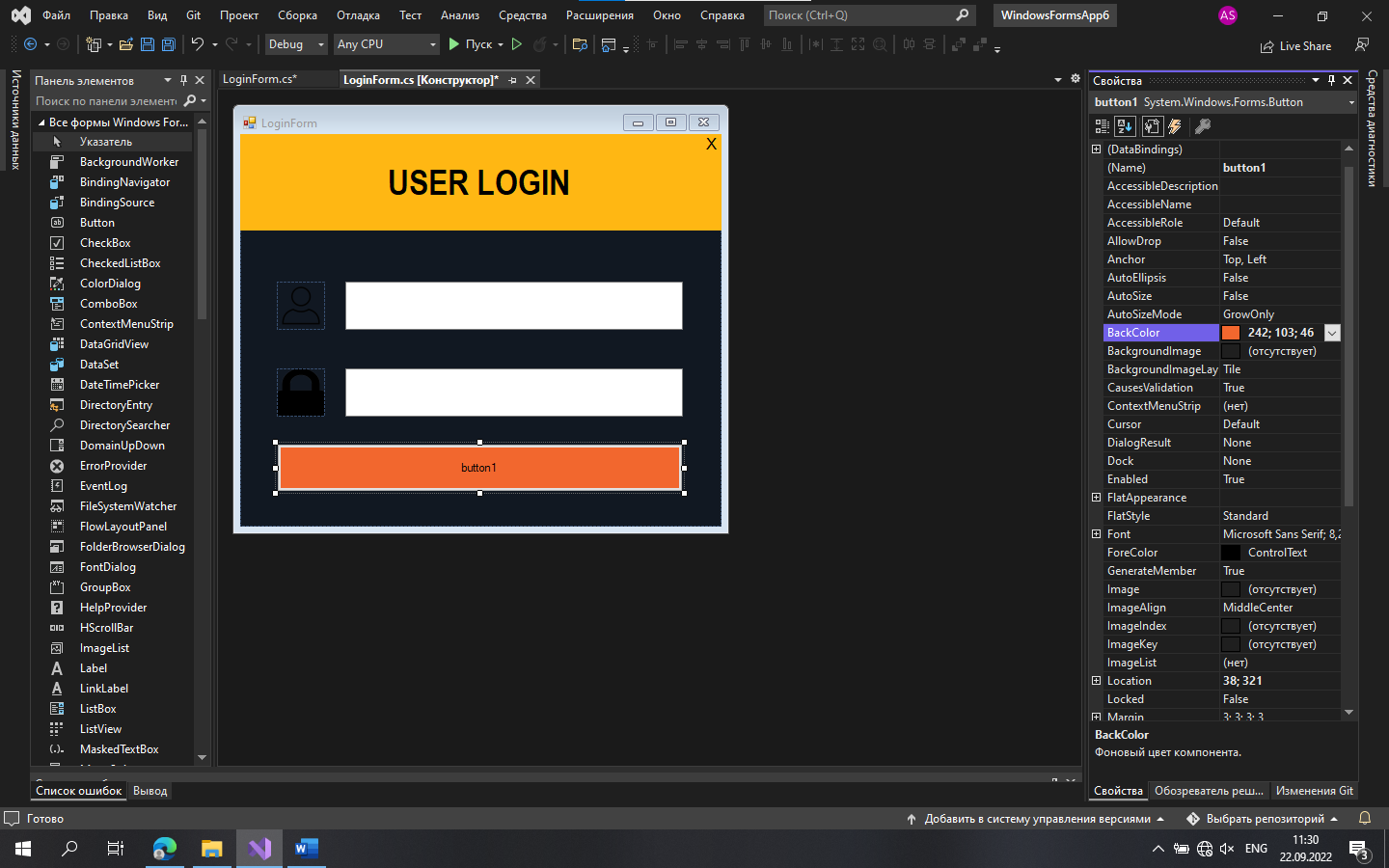


Рисунок 31 - Цвет кнопки



Рисунок 32 - Свойство FlatStyle

1. Настраиваем шрифт кнопки и цвет текста

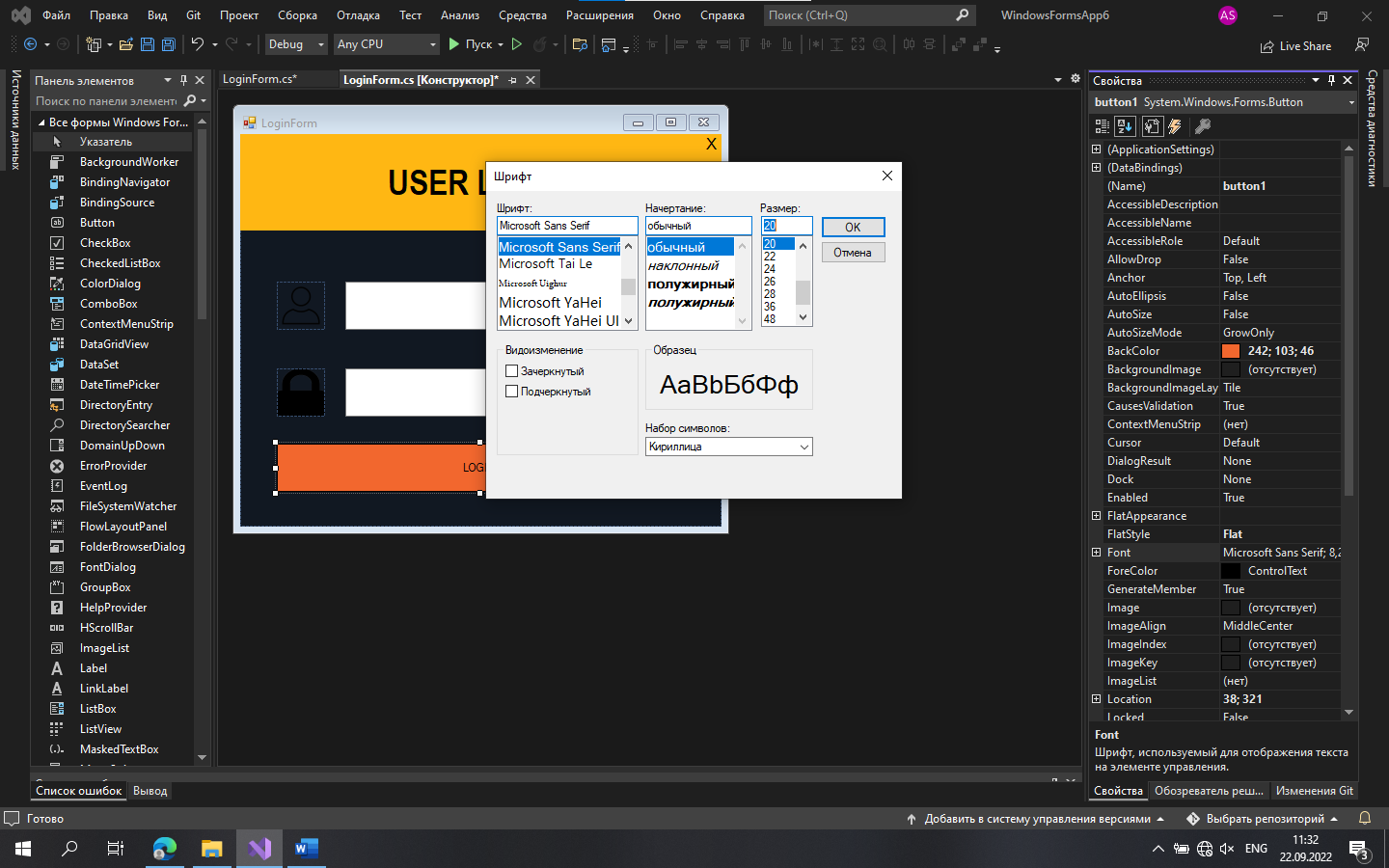


Рисунок 33 - Настройки шрифта



Рисунок 34 - Цвет текта

1. Убираем контур у кнопки

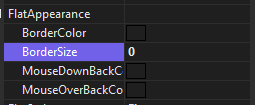


Рисунок 35 - Контур кнопки

1. У текстбоксов настраиваем шрифты

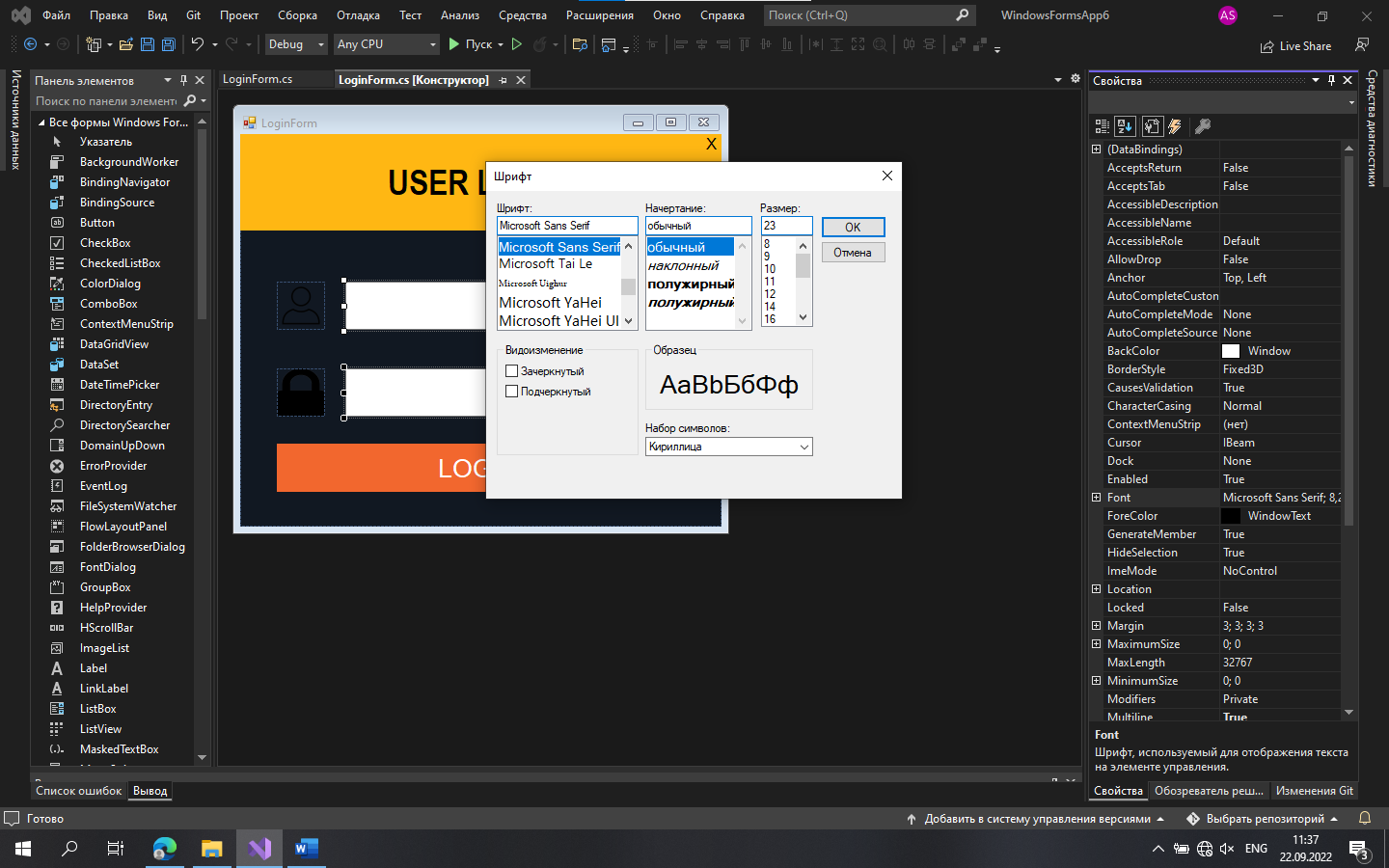


Рисунок 36 - Настройка шрифта

1. Далее переименовываем элементы



Рисунок 37 - Название textboxName



Рисунок 38 - Название textBoxPassword



Рисунок 39 - Название buttonLogin

1. Заходим в настройки textboxPassword и совйство UseSystemPassword меняем на true и в свойстве MultiLine убираем галочку



Рисунок 40 - Свойство UseSystemPassword

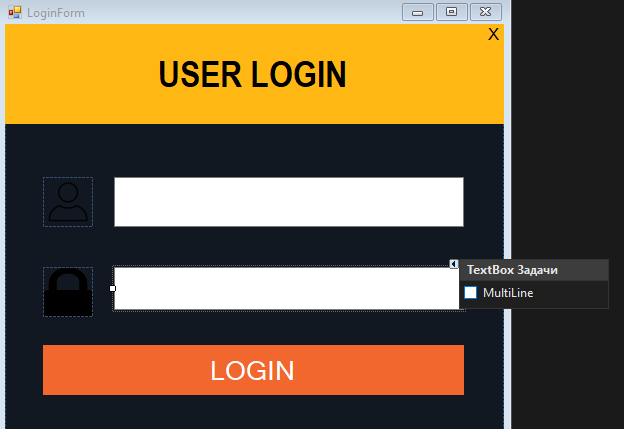


Рисунок 41 - Свойство MultiLine

1. Промежуточный результат

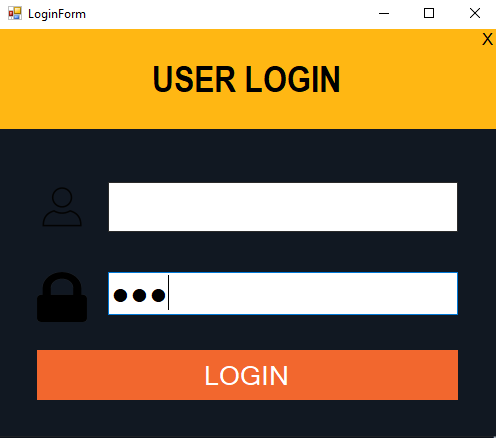


Рисунок 42 - Промежуточный результат

1. Переходим к коду, задаем textBoxPassword размер

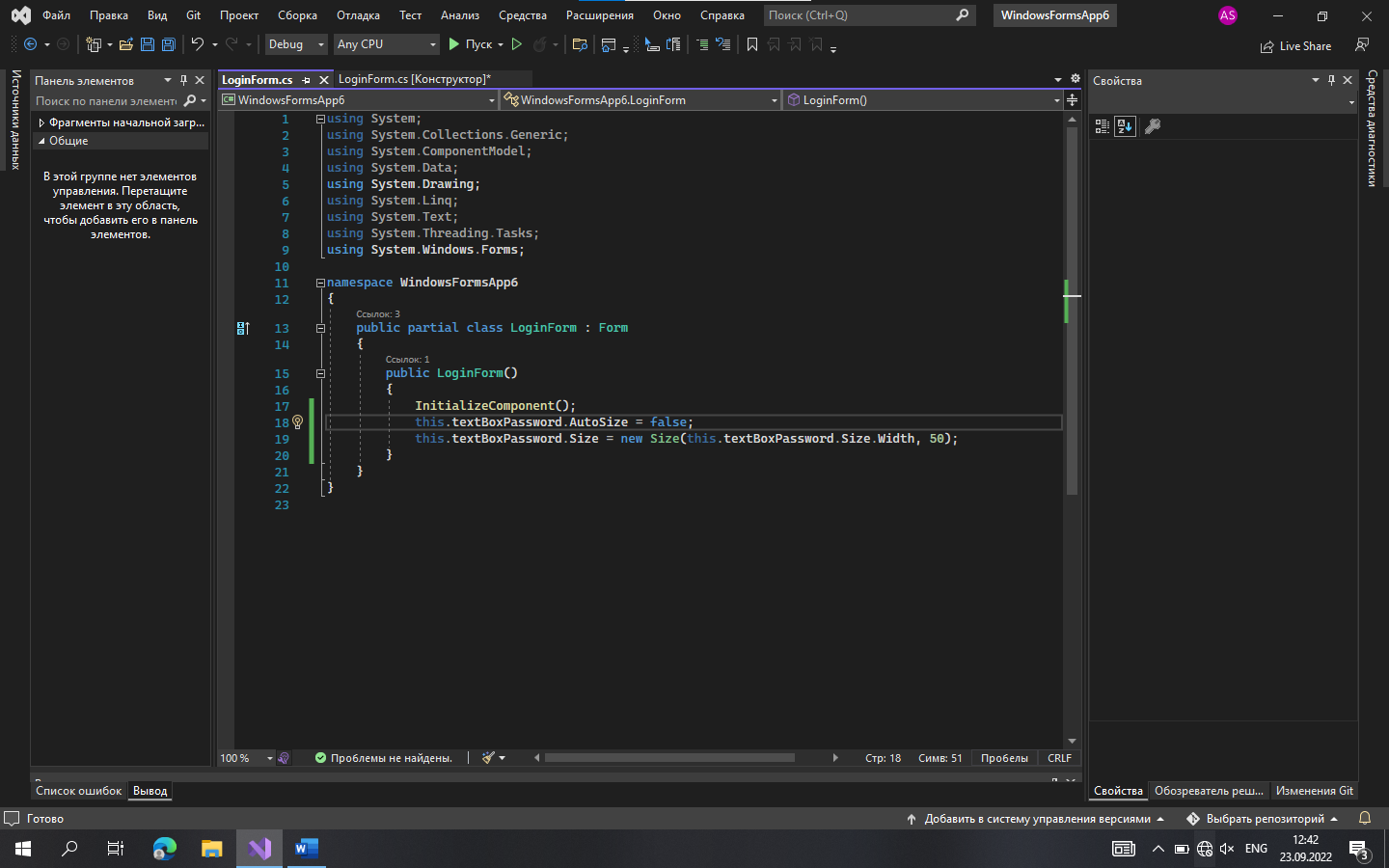


Рисунок 43 - Размер textbox

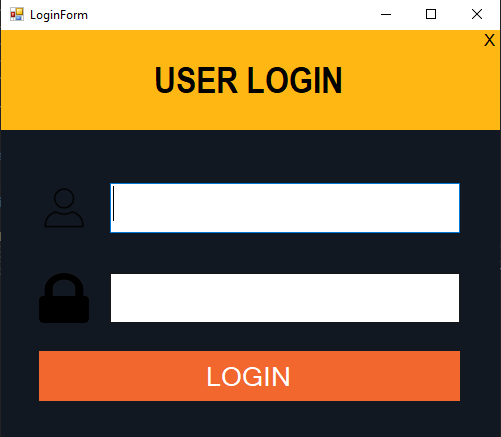


Рисунок 44 - Промежуточный результат

1. Заходим в свойства LoginForm и настраиваем, чтобы форма открывалась по середине экрана и у нее не было конутра



Рисунок 45 - Свойство CenterScreen



Рисунок 46 - Свойство FormBorderStyle

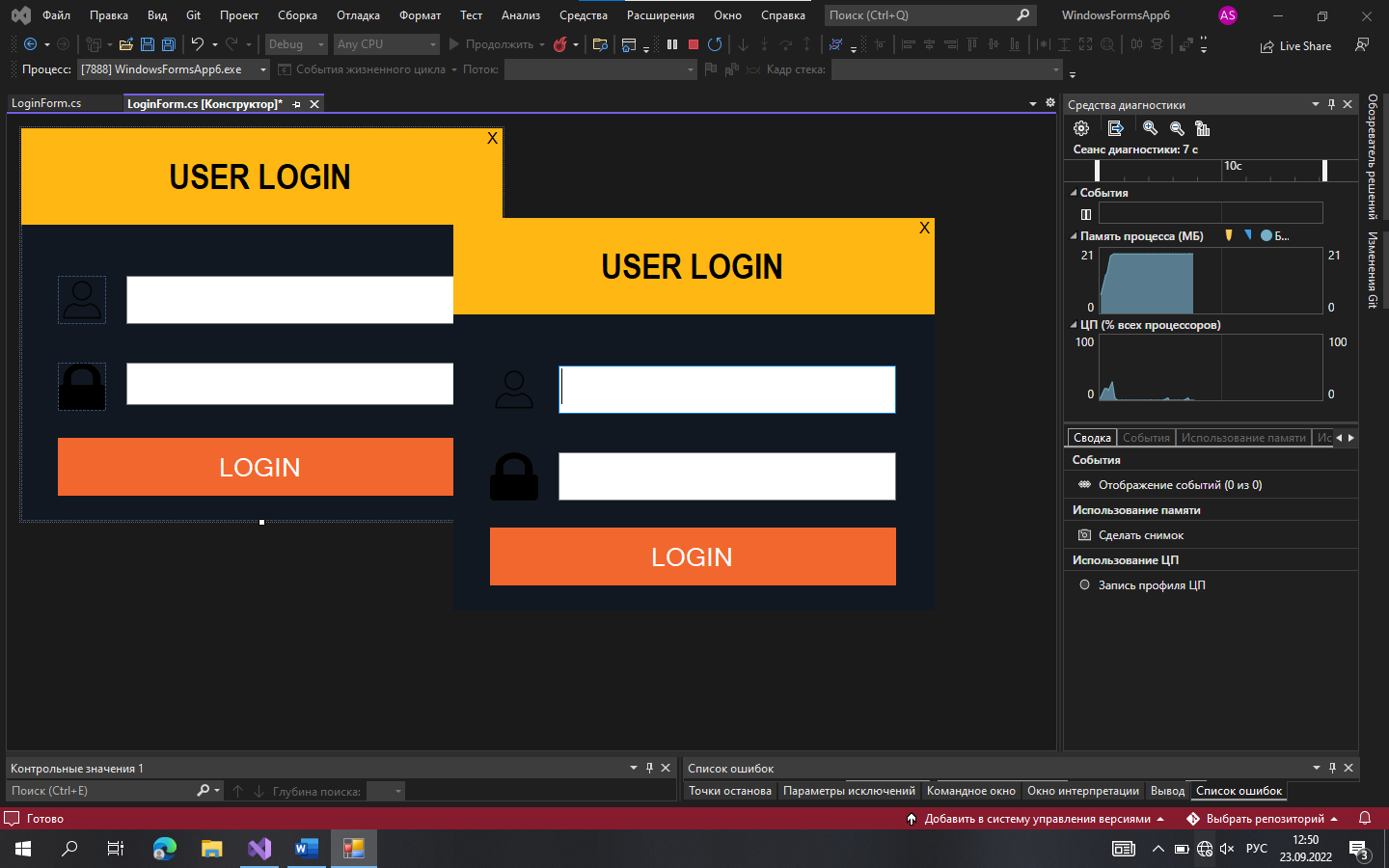


Рисунок 47 - Промежуточный результат

1. Заходим в настройки свойств labelClose и настраиваем цвет и курсор, шрифт



Рисунок 48 - Свойство Cursor



Рисунок 49 - Цвет

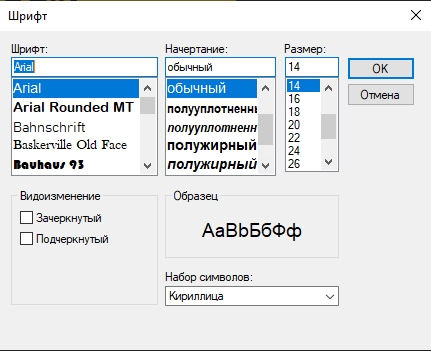


Рисунок 50 - Настройка шрифта

1. Заходим в события labelClose и находим MouseEnter и MouseLeave и задаем цвета, чтобы label был белым, а при наведении курсора черным

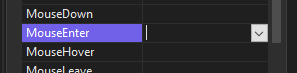


Рисунок 51 - Событие MouseEnter

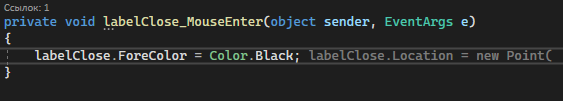


Рисунок 52 - Код события MouseEnter



Рисунок 53 - Событие MouseLeave

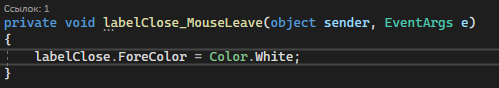


Рисунок 54 – Код события MouseLeave

1. Готовая форма

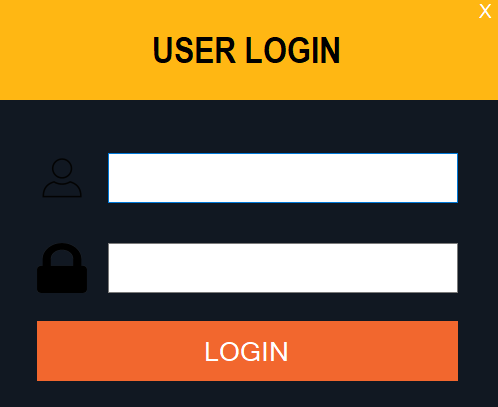


Рисунок 55 -Готовая форма

1. Форма при наведении курсора на кнопку закрыть

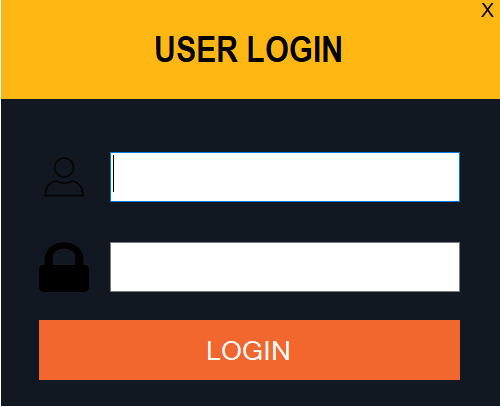


Рисунок 56 - Форма при наведении курсора на кнопку закрыть

1. Заходим в события label, который у нас выполняет роль выхода из приложения (крестик), и пишем событие, чтобы при нажатии на него, приложение закрылось

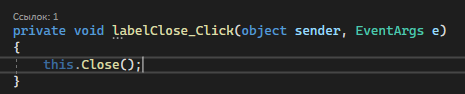


Рисунок 57 - Функция labelClose\_Click

1. Заходим на сайт, представленный на скрине и скачиваем mysql connector

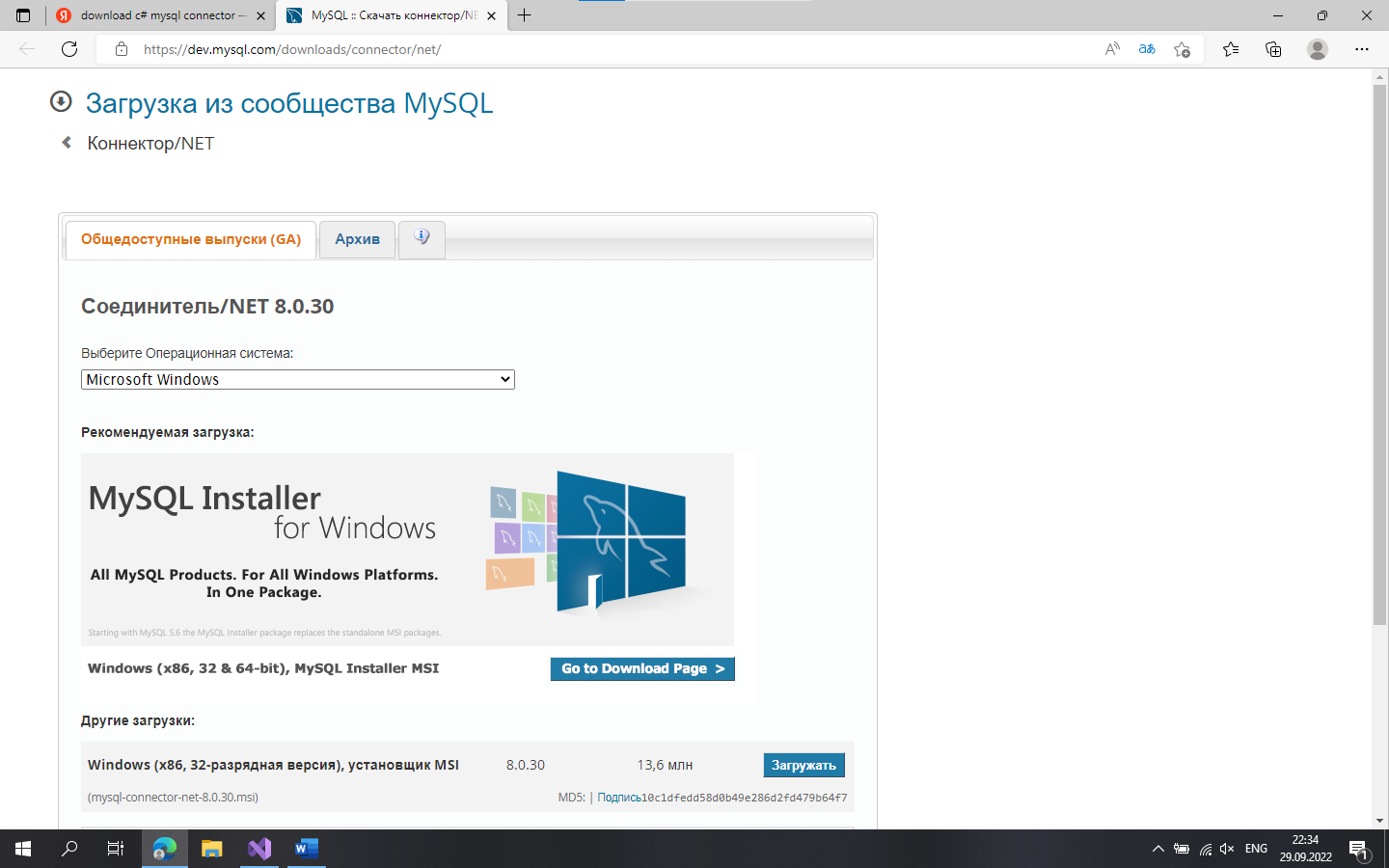


Рисунок 58 - Сайт mysql connector

1. Далее необходимо выполнить установку

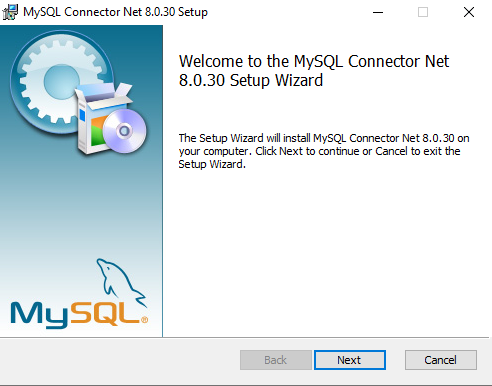


Рисунок 59 - Установка

1. Заходим в менеджер ссылок проекта и добавляем MySql.Data

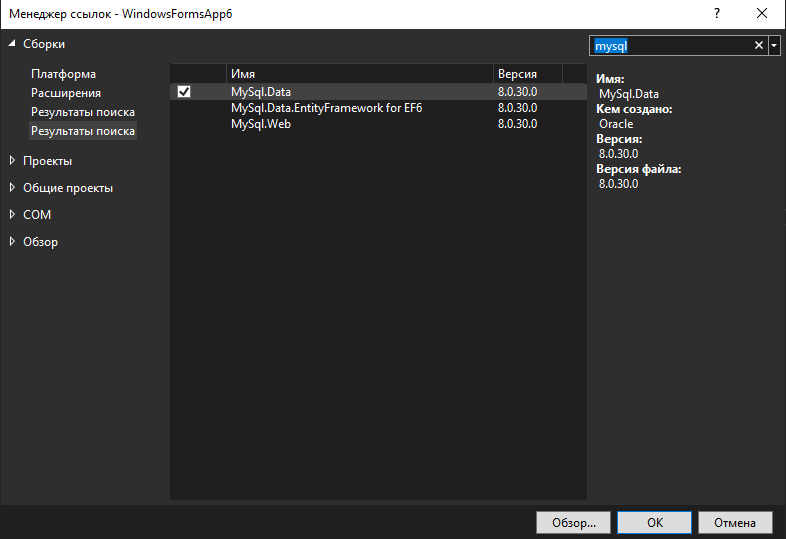


Рисунок 59 - Менеджер ссылок

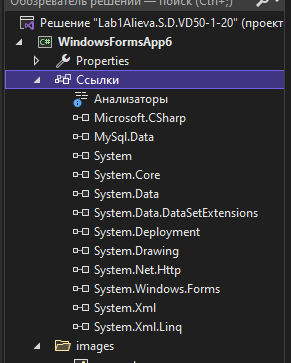


Рисунок 60 - Добавленные ссылки

1. Создаем подключение и в пространство имен подключаем MySql.Data.MySqlClient

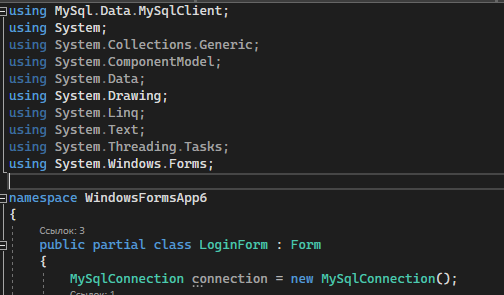


Рисунок 61 - Пространство имен

1. Запускаем сервер и заходим в PhpMyAdmin

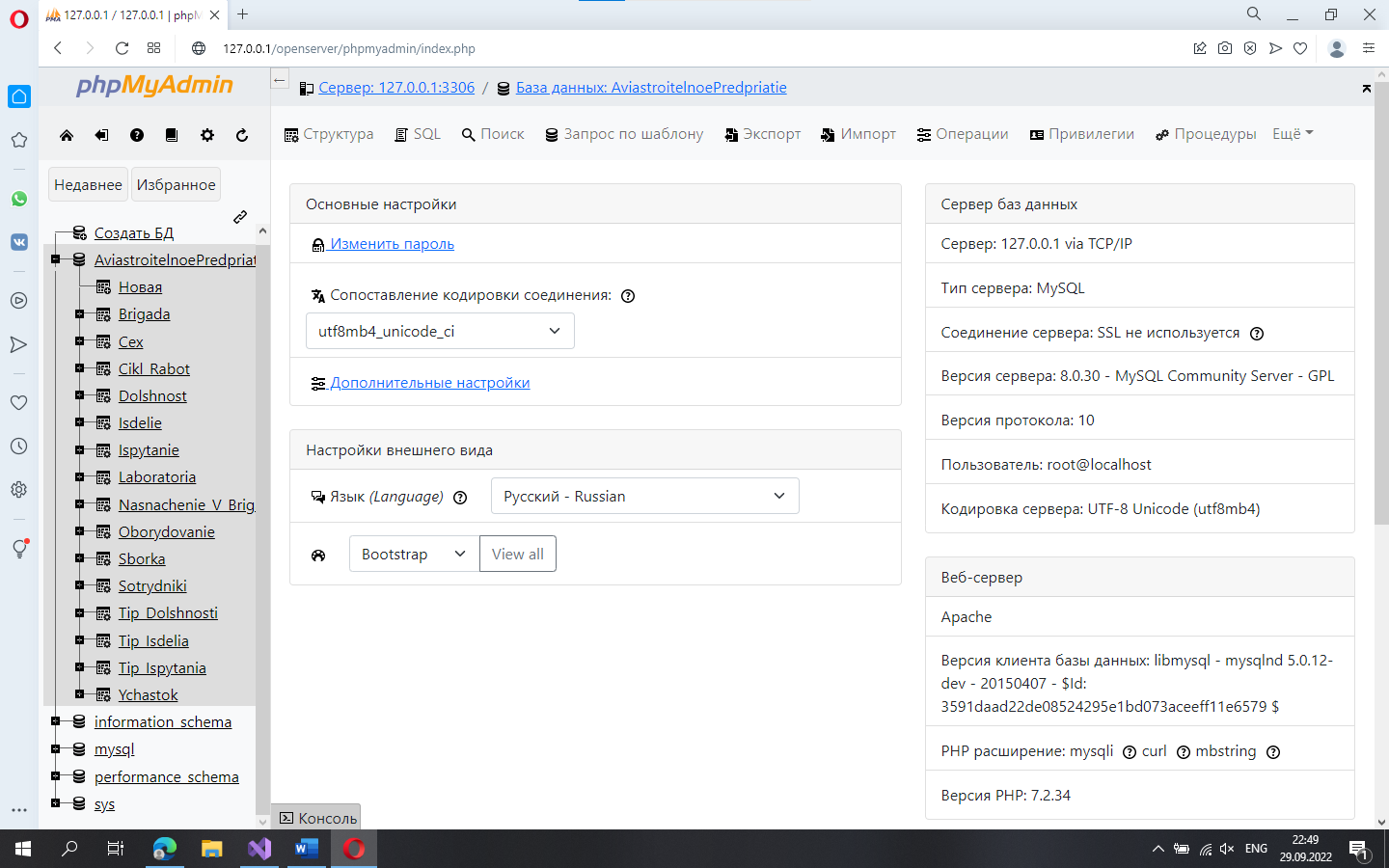


Рисунок 62 - PhpMyAdmin

1. Создаем новую бд под названием csharp\_users\_db

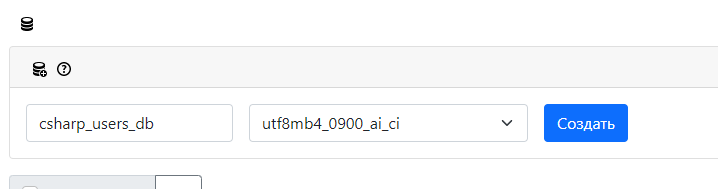


Рисунок 63 - Создание бд

1. Создаем новую таблицу, указывая имя таблицы и количество столбцов

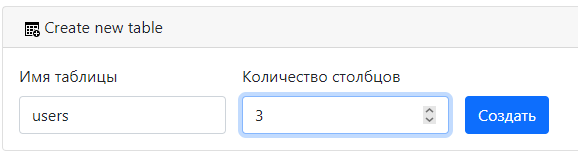


Рисунок 64 - Создание таблицы

1. Заполняем в таблице атрибуты



Рисунок 65 - Заполнение таблицы

1. Создаем подключение к бд, указывая сервер, порт, логин, пароль и название, созданной ранее бд



Рисунок 66 - Подключение бд

1. Создаем новый класс

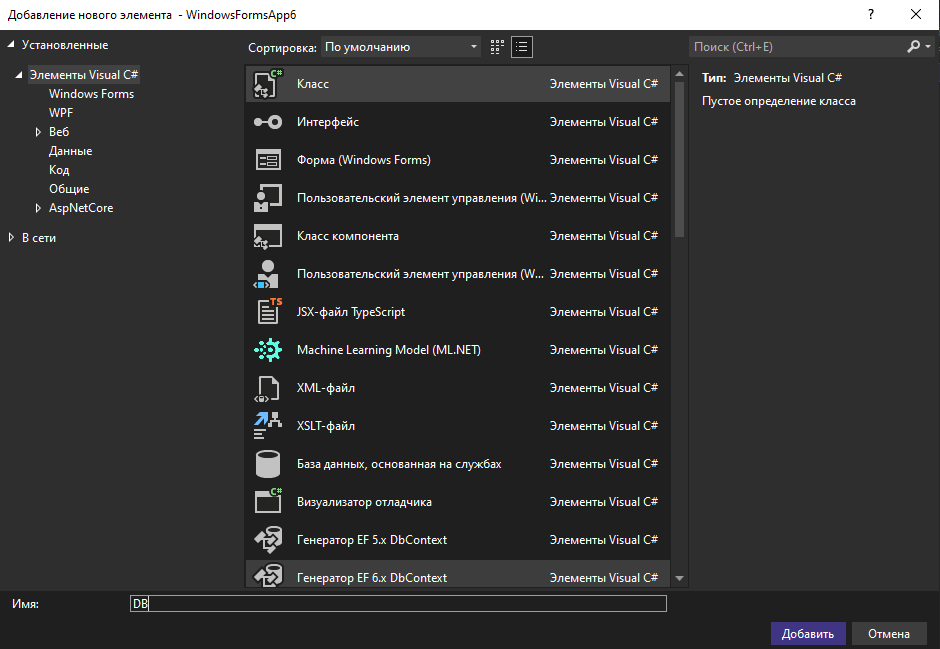


Рисунок 67 - Создание нового класса

1. Копируем наше подключение бд в новый созданный класс

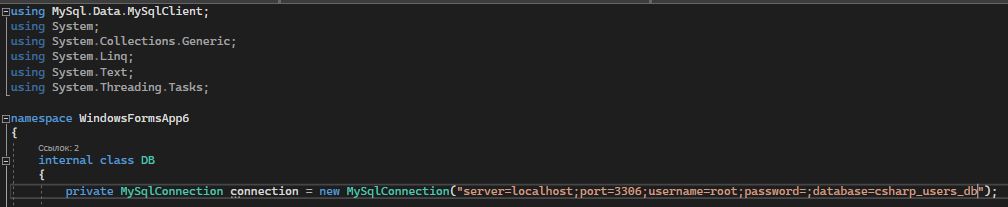


Рисунок 68 - Созданный класс

1. Пишем функцию для открытия соединений

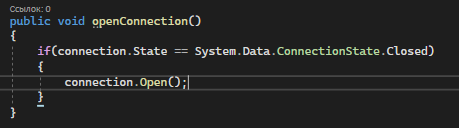


Рисунок 69 - Открытие соединения

1. Создаем функцию для закрытия соединения

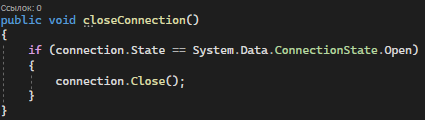


Рисунок 70 - Закрыие соединения

1. Создаем функцию для возврата соединения

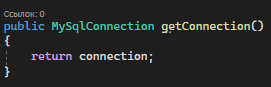


Рисунок 71 - Возврат соединения

1. Пишем событие кнопки login, в котором происходит обращение к бд с данными пользователя для получения пароля и логина, для входа и их проверки

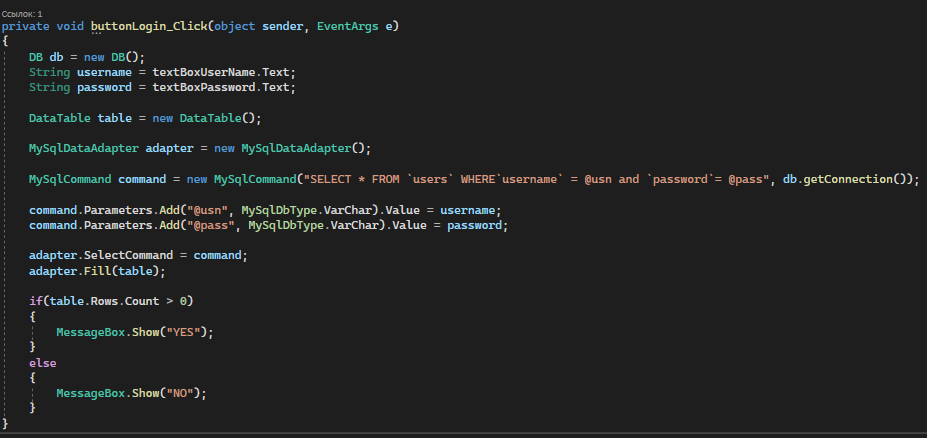


Рисунок 72 - Функция кнопки Login

1. Вносим в нашу бд данные пользователя для проверки авторизации

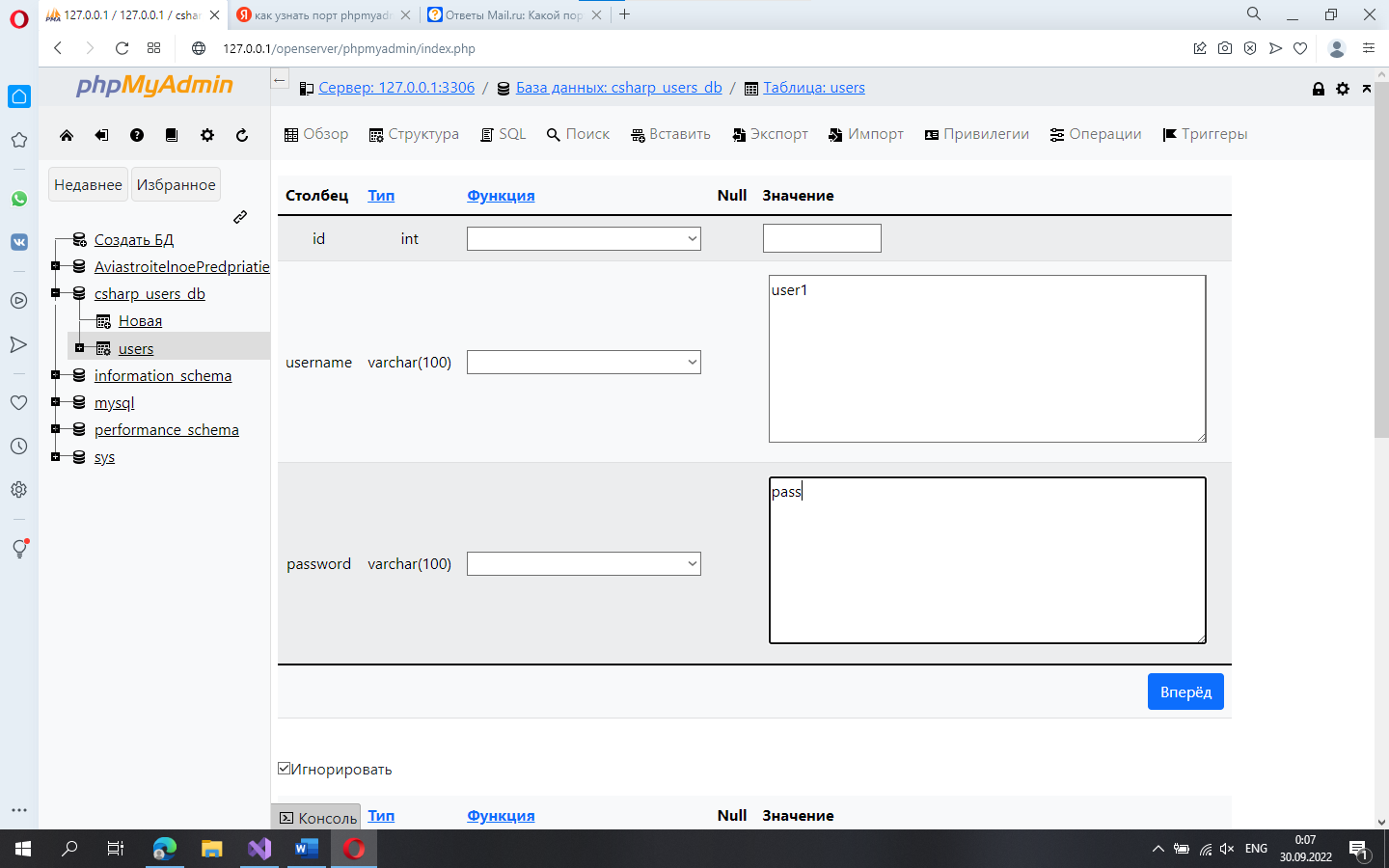


Рисунок 73 - Внесения данных в бд

Тестируем результат

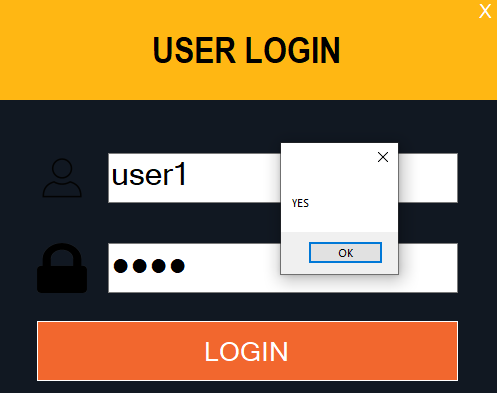


Рисунок 74 - Результат при введении верных логин и пароль

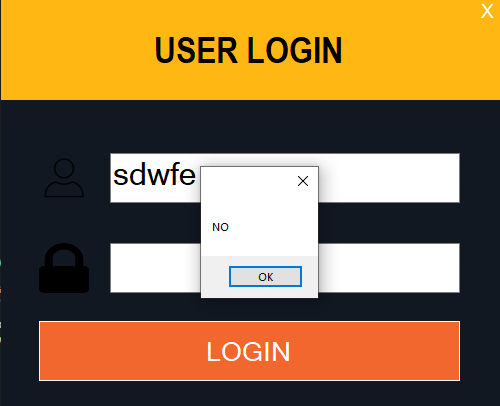


Рисунок 75 - Результат при введении неверных логин и пароль

Добавляем новую форму в проект

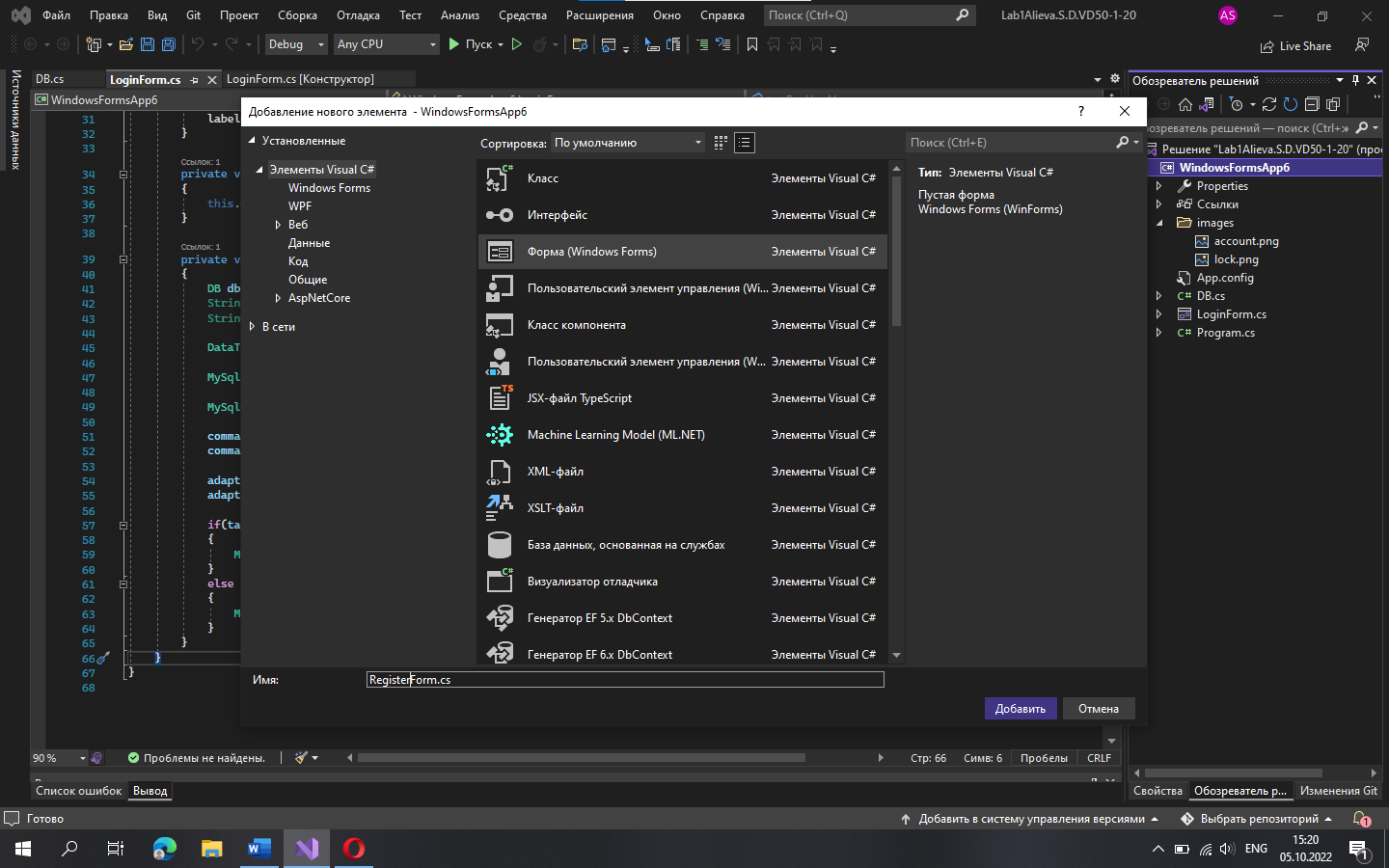


Рисунок 57 - Новая форма

Копируем содержимое LoginForm на новую созданную форму RegisterForm

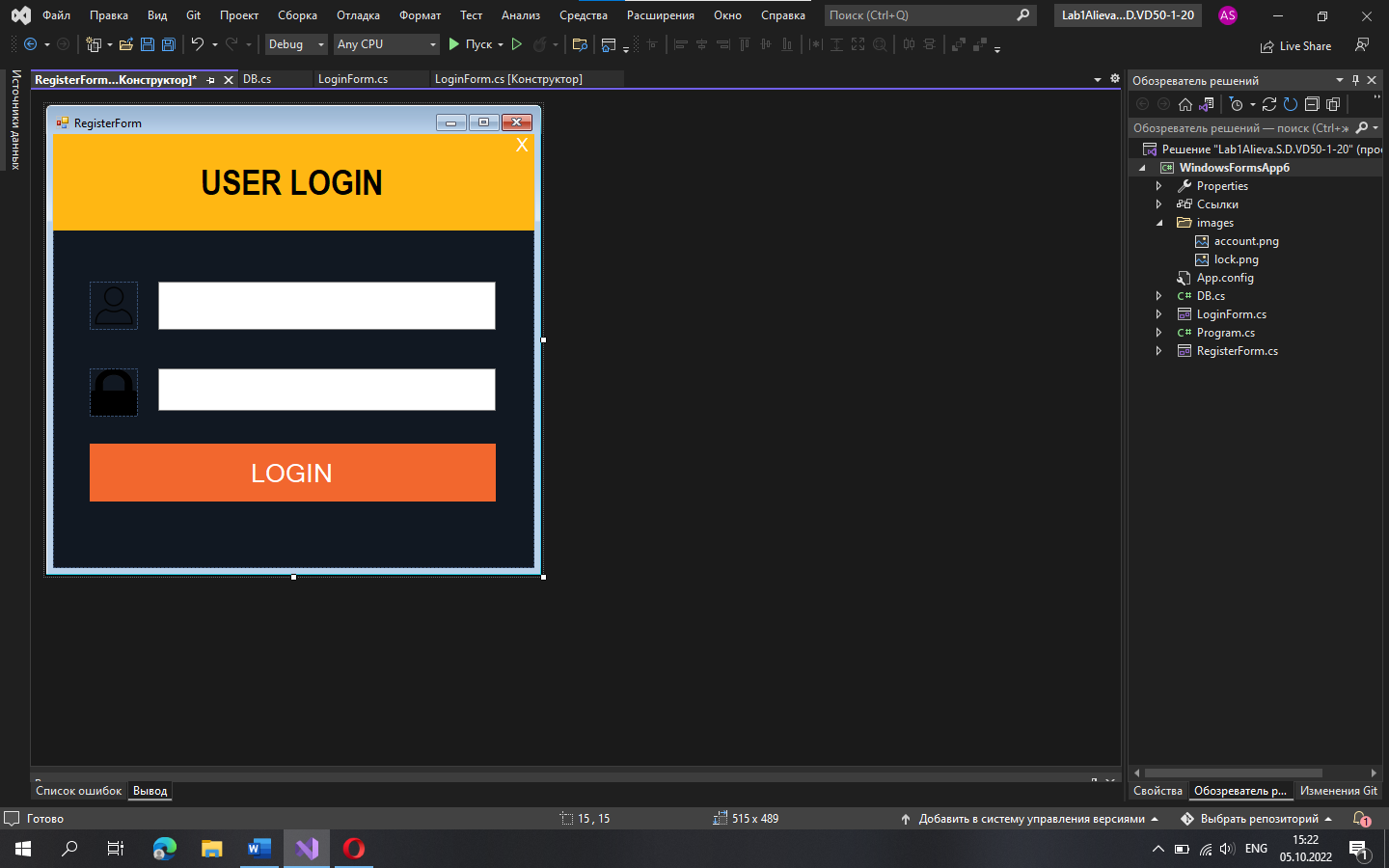


Рисунок 58 - Скопированная LoginForm

Редактируем, добавляем textbox

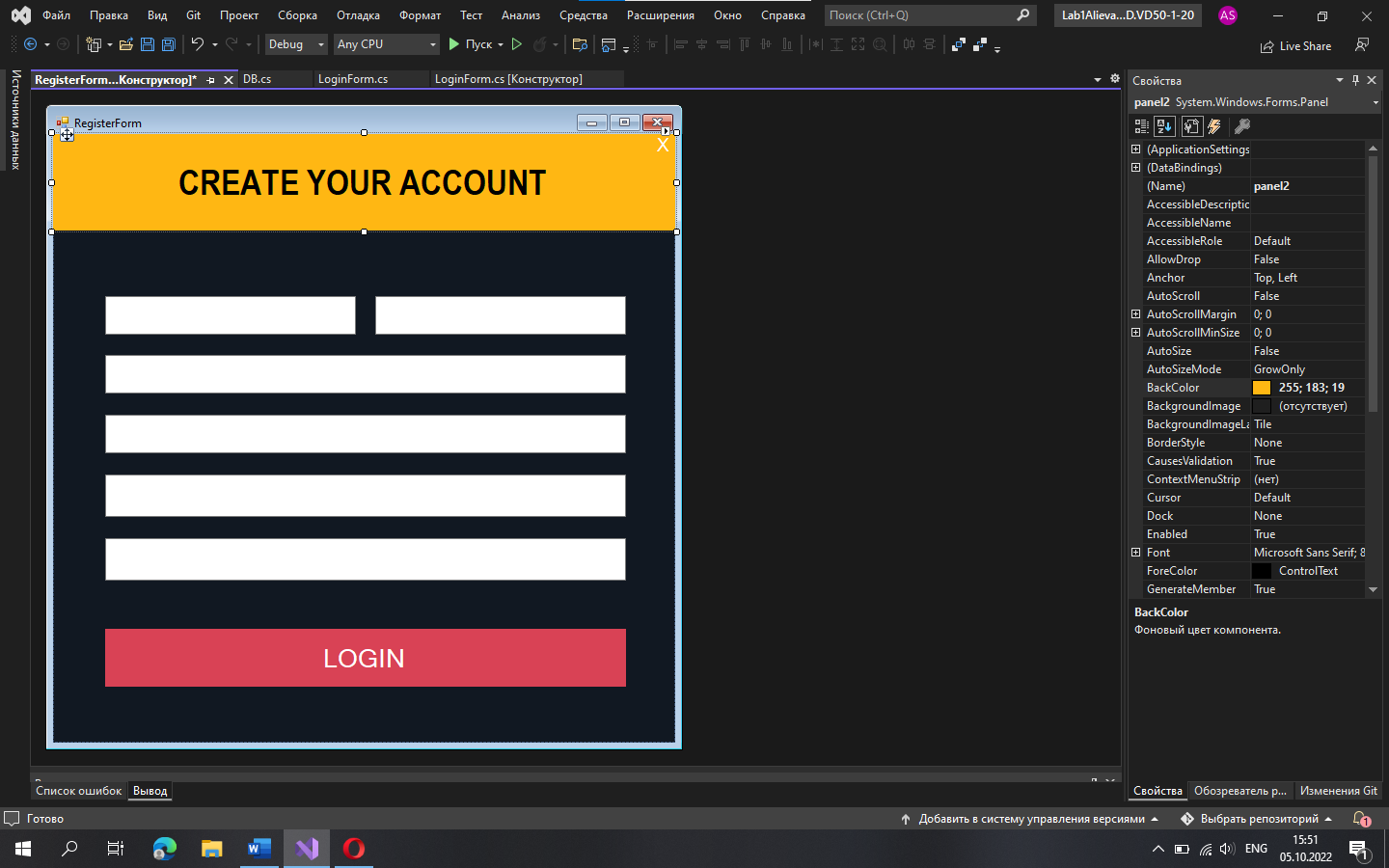


Рисунок 59 - Отредактированная форма RegisterForm

В Program.cs меняем LoginForm на RegisterForm, чтобы при запуске приложения запускалась она

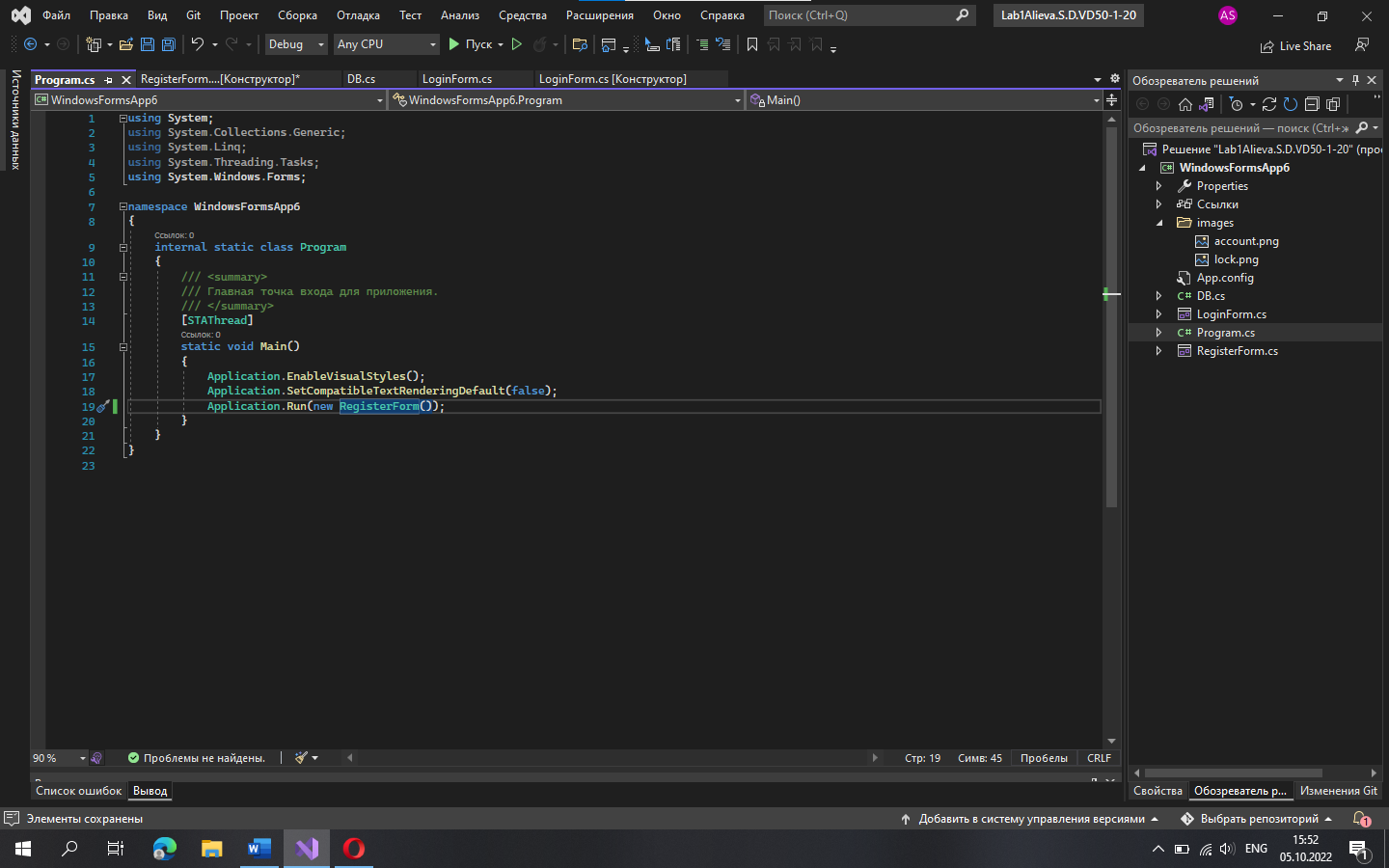


Рисунок 60 - Program.cs

Заходим в свойства формы и задаем, чтобы она запускалась по центру экрана



Рисунок 61 – StartPosition

Переименовываем все textbox



Рисунок 62 - textBoxFirstName



Рисунок 63 - textBoxLastName



Рисунок 64 - textBoxEmail



Рисунок 65 - textBoxUsername



Рисунок 66 - textBoxPasswordConfirm



Рисунок 67 – buttonCreateAccountx

Промежуточный результат

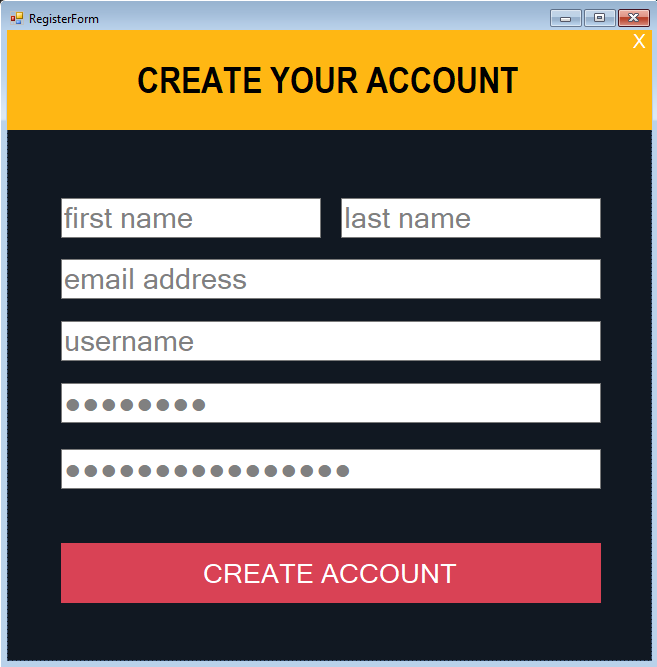


Рисунок 68 - Промежуточный результат

Для textBoxPassword и textBoxPasswordConfirm меняем свойство UseSystemPassword, чтобы в полях было видно, что написано, а не скрыто



Рисунок 69 - UseSystemPassword

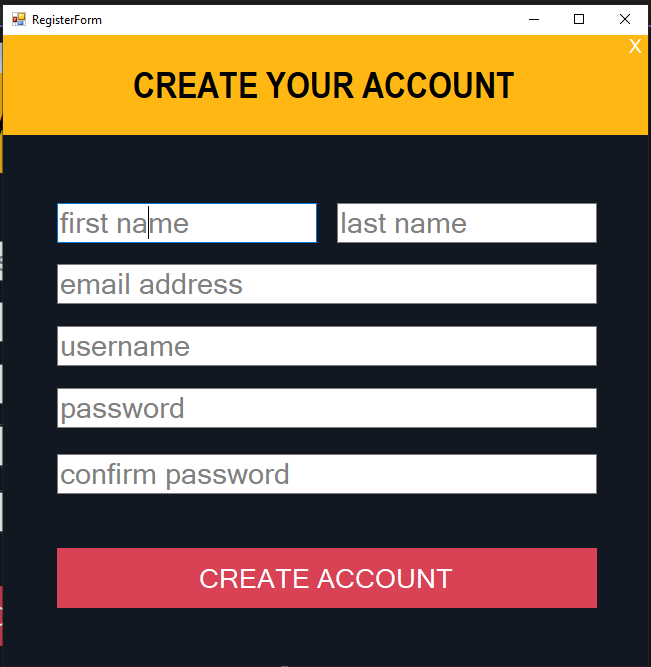


Рисунок 70 – Результат после изменения свойства UseSystemPassword

Для textboxFirstName пишем функции, чтобы при нажатии на данное поле надпись скрывалась, если поле остается пустым, то надпись опять появляется



Рисунок 71 - Функция textBoxFirstName\_Enter



Рисунок 72 - Функция textBoxFirstName\_Leave

Чтобы при запуске текстбоксы не были активными пишем следующую функцию



Рисунок 73 – Активный элемент

Для textboxLastName пишем функции, чтобы при нажатии на данное поле надпись скрывалась, если поле остается пустым, то надпись опять появляется



Рисунок 74 – textboxLastName

Для textboxEmail пишем функции, чтобы при нажатии на данное поле надпись скрывалась, если поле остается пустым, то надпись опять появляется



Рисунок 75 – textboxEmail

Для textboxUsername пишем функции, чтобы при нажатии на данное поле надпись скрывалась, если поле остается пустым, то надпись опять появляется



Рисунок 76 – textboxUsername

Для textboxPassword пишем функции, чтобы при нажатии на данное поле надпись скрывалась, если поле остается пустым, то надпись опять появляется



Рисунок 77 – textboxPassword

Для textboxPasswordConfirm пишем функции, чтобы при нажатии на данное поле надпись скрывалась, если поле остается пустым, то надпись опять появляется



Рисунок 78 – textboxPasswordConfirm

Для label, который выполняет функцию закрытия приложения пишем следующие события, чтобы при нажатии на него оно закрывалось, а при наведении label менял цвет на черный



Рисунок 79 - События кнопки закрытия приложения

Убираем у формы рамки меняя свойство FormBorderStyle



Рисунок 80 - FormBorderStyle

После выполнения всех действий получаем данный результат

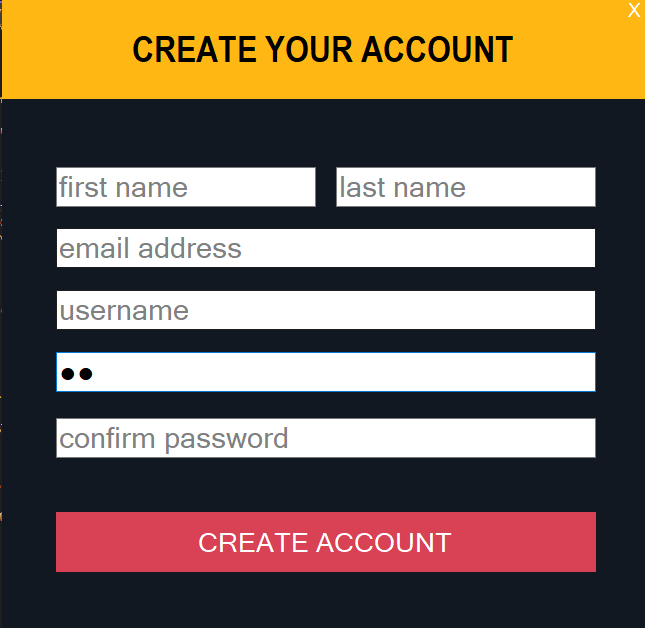


Рисунок 81 - Результат

Заходим в PhpMyAdmin и добавляем в таблице столбцы с атрибутами firstname, lastname и emailaddress

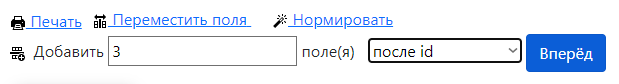


Рисунок 82 - Добавление столбцов в таблицу

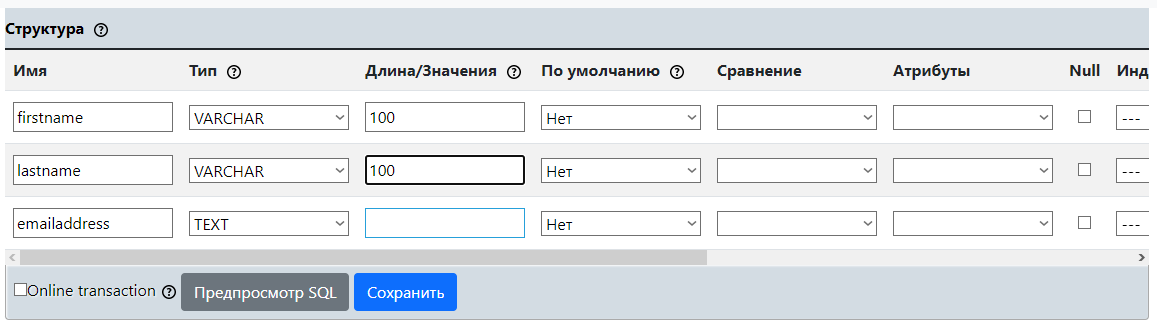


Рисунок 83 - Структура таблицы

В пространство имен подключаем MySql

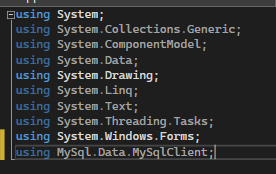


Рисунок 84 - Пространстрво имен

Для кнопки создание аккаунт пишем функцию, в которой данные с текстбоксов считываются и заносятся в бд и выводится сообщение «ACCOUNT CREATED»

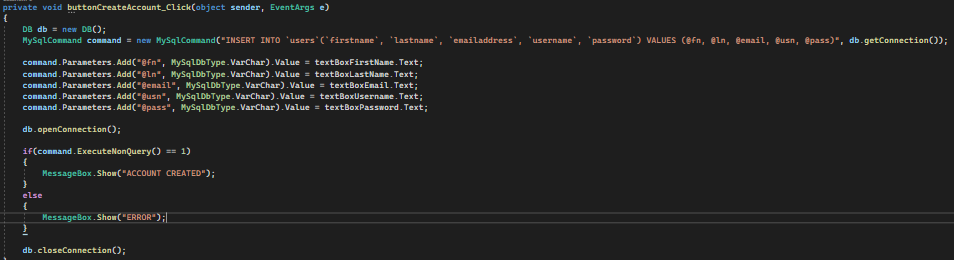


Рисунок 85 - Функция кнопки создания аккаунта

Промежуточный результат



Рисунок 86 - Промежуточный результат

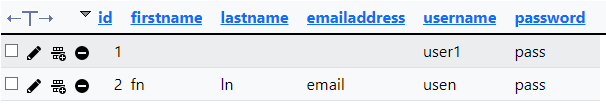


Рисунок 87 - Данные в бд

Пишем событие, в котором будет проверяться username, чтобы невозможно было создать пользователя с таким username, который уже используется

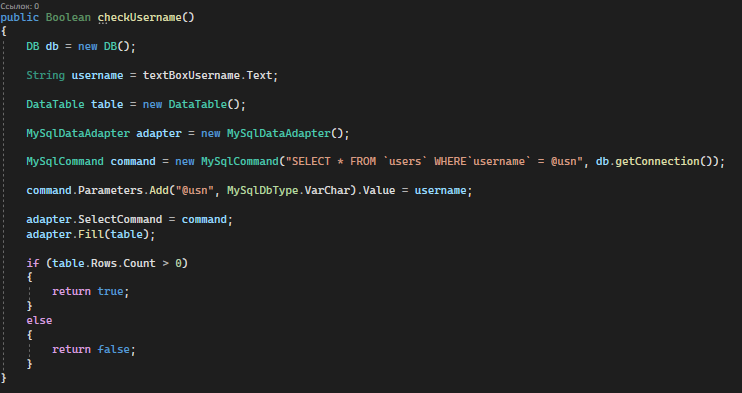


Рисунок 88 - Функция checkUsername

Пишем условие, в котором определяется создан такой пользователь уже или нет, если не создан то выводим месседжбокс, с надписью «account created»



Рисунок 89 - Проверка, существует такой аккаунт или нет

Пишем функцию, в которой проверяются поля, пустые они или заполненные



Рисунок 90 - Проверка, пустые поля регистрации или нет

Проверка создания аккаунта с данными уже существующего пользователя

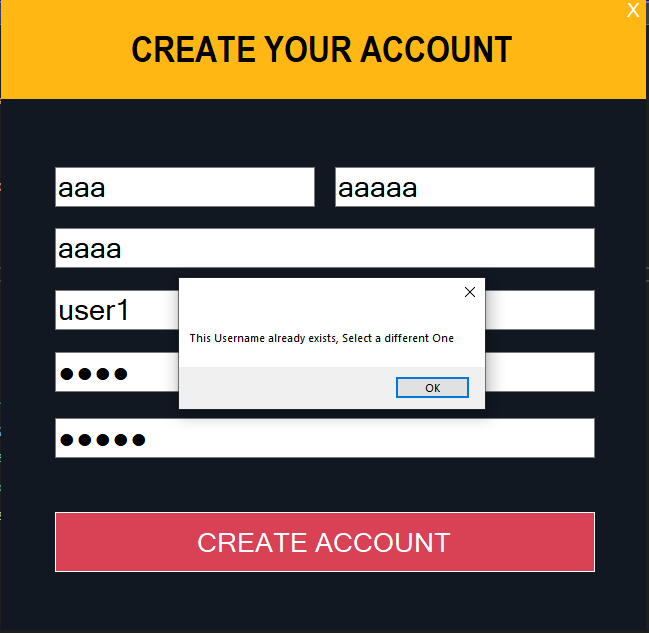


Рисунок 91 - Создание уже существующего пользователя

Проверка создания аккаунт с данными еще существующего пользователя

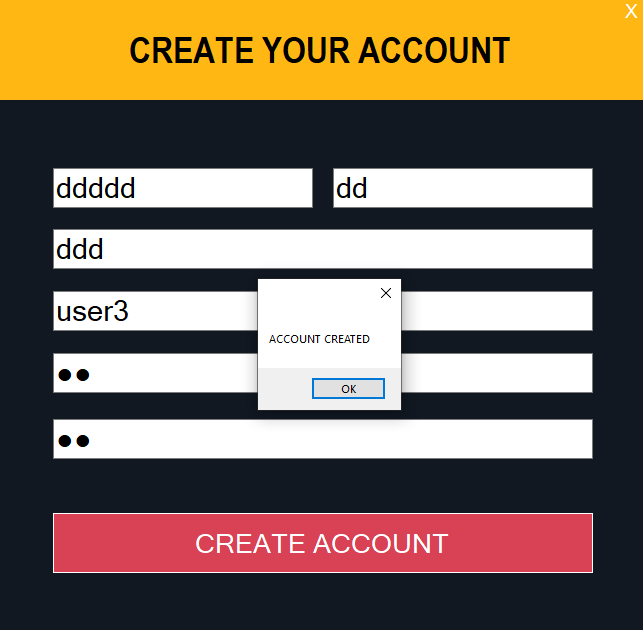


Рисунок 92 - Создание аккаунта

Добавление нового пользователя в базе данных



Рисунок 93 - Новый пользователь в бд

Пишем функцию, в которой проверяется введены какие-то данные пользователем или нет, если нет, то выводится мееседжбокс с сообщением «Enter your information first»



Рисунок 94 - Вывод меседжбокса если поля пустые

Проверка регистрации аккаунта без введения данных



Рисунок 95 - Создание аккаунта с пустыми полями

Пишем условие проверки паролей, чтобы при вводе подтверждения пароля оно совпадало с введенным паролем.



Рисунок 96 - Проверка пароля

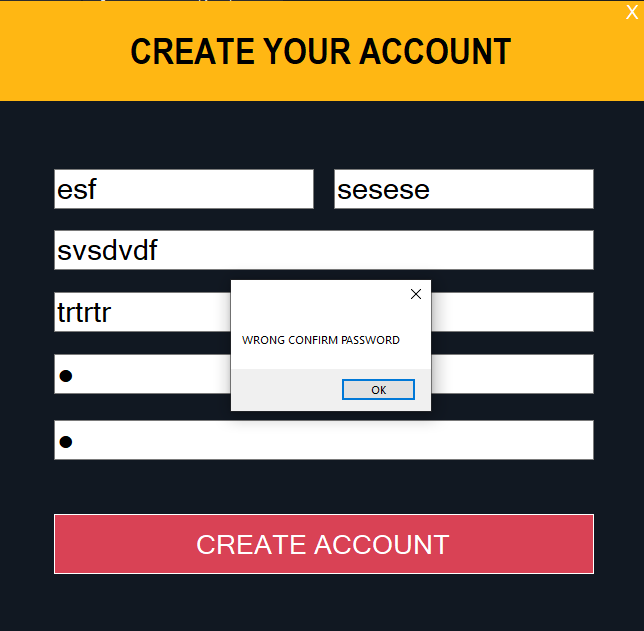


Рисунок 97 – Отрицательный результат проверки пароля

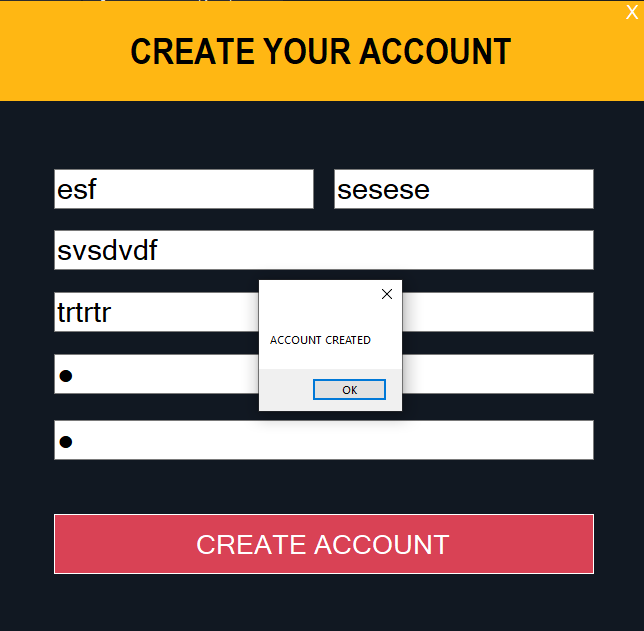


Рисунок 98 – Положительный результат проверки пароля

Добавляем в месседжбокс, который показывается при использовании существующего username кнопки «ОК», «Отмена» и изображение ошибки, а так же в тот, который показывается при создании аккаунта с изображением информации и кнопкой «ОК».



Рисунок 99 - Доработка MessageBox



Рисунок 100 - Готовые MessageBox с ошибкой



Рисунок 101 - Готовые MessageBox без ошибки

Добавляем в месседжбокс, который показывается при вводе в пароль и подтверждения пароля разные данные кнопки «ОК», «Отмена» и изображение ошибки.



Рисунок 102 - Доработка MessageBox



Рисунок 103 – Готовый MessageBox с ошибкой

Добавляем в месседжбокс, который показывается при оставлении всех полей воода пустыми кнопки «ОК», «Отмена» и изображение ошибки.



Рисунок 104 - Доработка MessageBox



Рисунок 105 - Готовый MessageBox с ошибкой

Так же добавляем в мессджбоксы на LoginForm кнопки «ОК» и изображение ошибки.



Рисунок 106 - Доработка MessageBox

Создаем новую форму – MainForm и копируем в нее все элементы с RegisterForm.

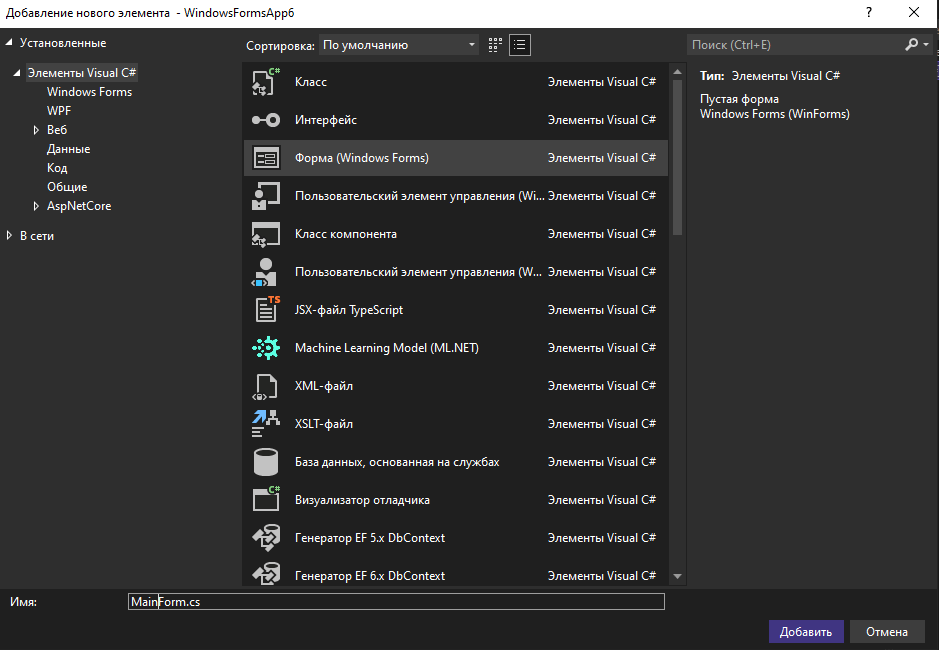


Рисунок 107 - Создание новой формы

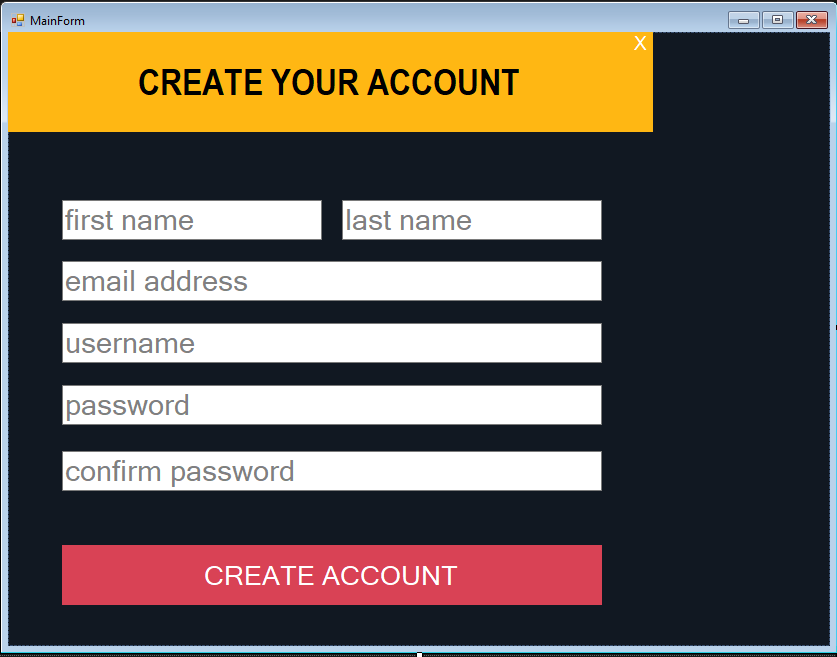


Рисунок 108 - Новая форма

Удаляем с нее некоторые элементы и меняем текст в label1, задаем форме свойства, чтобы она открывалась по середине экрана.



Рисунок 109 - Отредактировання форма



Рисунок 110 - Свойство StartPosition

В форму LoginForm добавляем label с текстом Don’t have account? Sign up.



Рисунок 111 - Текст label

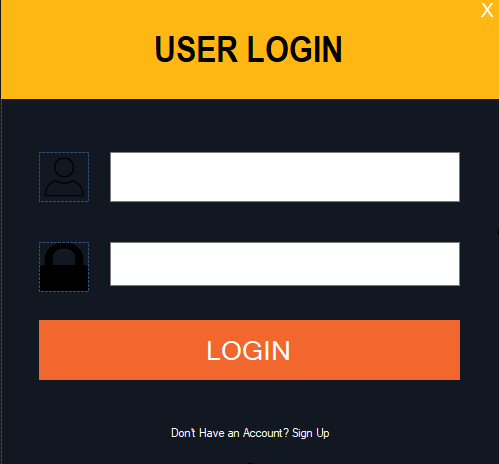


Рисунок 112 - Отредактированная LoginForm

Для созданного label пишем событие на клик, чтобы открывалась форма RegisterForm.



Рисунок 113 - Событие открытия RegisterForm

Заходим в Program.cs и меняем открывающеюся форму на LoginForm.



Рисунок 114 - Program.cs

В Register form на label, который отвечает за закрытие приложения пишем Application.Exit(), чтобы оно действительно закрывалось полностью, а не просто скрывалась форма.



Рисунок 115 - Событие закрытия приложения

На всех формах у кнопок и label, которые выполняют роль кнопки, меняем свойство cursor на hand, чтобы при наведении на них курсор менялся на руку.



Рисунок 116 - Свойтсво Cursor

Пишем события для labelGoSingUp, чтобы в спокойном состоянии тест был белого цвета, а при наведении желтого



Рисунок 117 - Событие labelGoSingUp

В настройках шрифта у labelGoSingUp ставим галочку напротив «Подчерктнутый»



Рисунок 118 - Настройки шрифта

Копируем этот label на форму RegisterForm, меняем у него название и текст

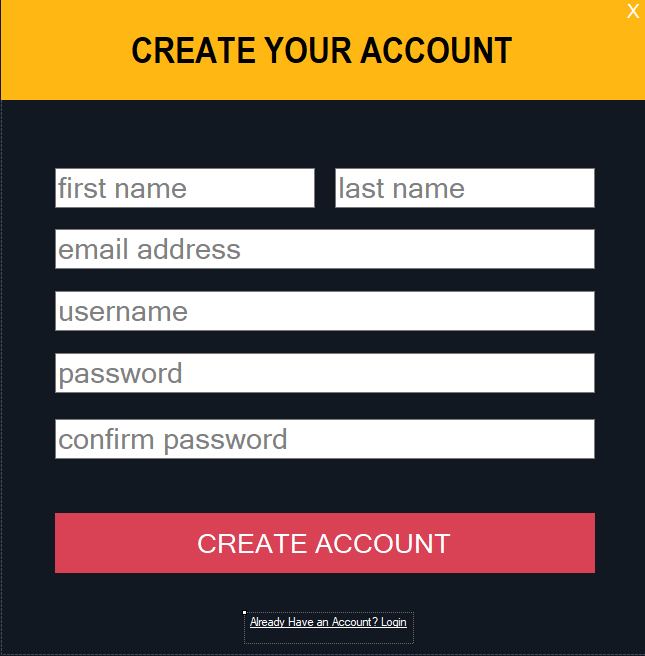


Рисунок 119 - RegisterForm



Рисунок 120 - Название label

Пишем события для него, чтобы в спокойном состоянии тест был белого цвета, а при наведении желтого, а также чтобы при клике на него открывалась форма LoginForm.



Рисунок 121 - Событие смены цвета при наведении на label



Рисунок 122 - Событие открытия LoginForm

В LoginForm на label, который отвечает за закрытие, приложение пишем Application.Exit(), чтобы оно действительно закрывалось полностью, а не просто скрывалась форма.



Рисунок 123 - Событие закрытия приложения

В событие проверки правильности ввода username и password пишем, открытие MainForm.



Рисунок 124 - Открытие MainForm

В MainForm пишем событие закртия приложения.



Рисунок 125 - Событие закрытия приложения

В Login form при проверке введения данных в текстбоксы меняем username на password



Рисунок 126 - Проверка введения данных в текстбоксы

**Определение:**

Отладка программных модулей - это процесс поиска и устранения ошибок в коде, чтобы программа работала корректно.

**Цель:**

Главная цель отладки - сделать программу надежной и безошибочной.

**Задание:**

1. **Воспроизведение ошибки:** Запустите программу и создайте условия, при которых возникает ошибка.
2. **Анализ ошибки:** Определите тип ошибки (синтаксическая, логическая, ошибка времени выполнения) и место ее возникновения.
3. **Исправление ошибки:** Внесите необходимые изменения в код, чтобы устранить ошибку.
4. **Тестирование:** Проверьте, решена ли проблема, и не появились ли новые ошибки.
5. **Повтор:** Повторяйте шаги 1-4, пока не будут найдены и исправлены все ошибки.

**Инструменты:**

Существует множество инструментов для отладки, как встроенных в среды разработки, так и сторонних.

**Распространенные инструменты:**

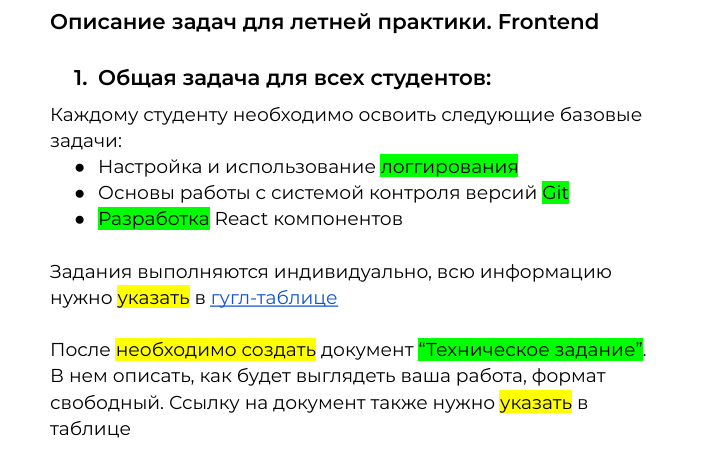
* **Отладчики:** Позволяют пошагово выполнять код, просматривать значения переменных и останавливаться в определенных точках.
* **Анализаторы статического кода:** Выявляют потенциальные проблемы в коде до его компиляции.
* **Тестовые фреймворки:** Автоматизируют процесс тестирования и помогают найти ошибки.

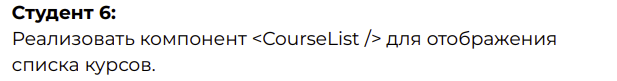
**Методы:**

* **Пошаговая отладка:** Выполнение программы пошагово с остановками в определенных точках.
* **Установка точек останова:** Остановка программы в определенных местах для проверки значений переменных и состояния программы.
* **Использование трассировки:** Запись информации о выполнении программы для последующего анализа.
* **Профилирование:** Измерение производительности программы для выявления узких мест.

**Навыки:**

* **Чтение кода:** Умение понимать код, написанный на языке программирования.
* **Логическое мышление:** Умение анализировать код и находить причины ошибок.
* **Внимательность:** Умение концентрироваться на деталях и не упускать из виду важные моменты.
* **Настойчивость:** Умение не сдаваться и продолжать поиск ошибок, даже если это сложно.





**\*при прохождении на производственной практике произошли на предприятии изменения задания\***

import React, { useState } from 'react';

import './AuthForm.css'; // Подключаем стили

const AuthForm = () => {

const [isLogin, setIsLogin] = useState(true); // Переключатель между входом и регистрацией

const [email, setEmail] = useState('');

const [password, setPassword] = useState('');

const handleSubmit = (e) => {

e.preventDefault();

console.log('Email:', email);

console.log('Password:', password);

// Здесь можно добавить логику для отправки данных на сервер

};

return (

<div className="auth-container">

<h2>{isLogin ? 'Вход' : 'Регистрация'}</h2>

<form onSubmit={handleSubmit}>

<div className="form-group">

<label htmlFor="email">Email</label>

<input

type="email"

id="email"

value={email}

onChange={(e) => setEmail(e.target.value)}

required

/>

</div>

<div className="form-group">

<label htmlFor="password">Пароль</label>

<input

type="password"

id="password"

value={password}

onChange={(e) => setPassword(e.target.value)}

required

/>

</div>

<button type="submit" className="submit-btn">

{isLogin ? 'Войти' : 'Зарегистрироваться'}

</button>

</form>

<div className="auth-links">

{isLogin ? (

<>

<p>

<a href="#forgot-password">Забыли пароль?</a>

</p>

<p>

Еще нет аккаунта?{' '}

<button onClick={() => setIsLogin(false)}>Зарегистрируйтесь</button>

</p>

</>

) : (

<p>

Уже есть аккаунт?{' '}

<button onClick={() => setIsLogin(true)}>Войдите</button>

</p>

)}

</div>

</div>

);

};

export default AuthForm;