**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Российский экономический университет имени Г. В. Плеханова»**

**Московский приборостроительный техникум**

**Отчёт**

**по выполнению практической работы № 4**

По дисциплине / **Основы алгоритмизации и программирования**

МДК:

Тема: **Разработка клиентского приложения. Часть 2.**

**Обеспечение безопасности приложения.**

**Сохранение данных аутентификации**

Специальность: **09.02.07 «Информационные системы и**

**программирование»**

Квалификация: **Разработчик веб и мультимедийных приложений**

Выполнил: Монахов К.М.

Студент группы: ВД50-1-18

Принял: Горшков М. Н.

Дата: «\_\_» \_\_\_\_\_\_\_\_\_ 202\_ г.

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Цель работы:**

Произвести модификацию базы данных на виртуальном веб-сервере

с целью обеспечения безопасной аутентификации в программном обеспечении

для дальнейшей работы внутри информационной системы в клиентском

приложении на универсальной платформе Windows.

**Ход работы:**

1. Пункт

Запускаем сборку виртуального веб-сервера и внутренние процессы

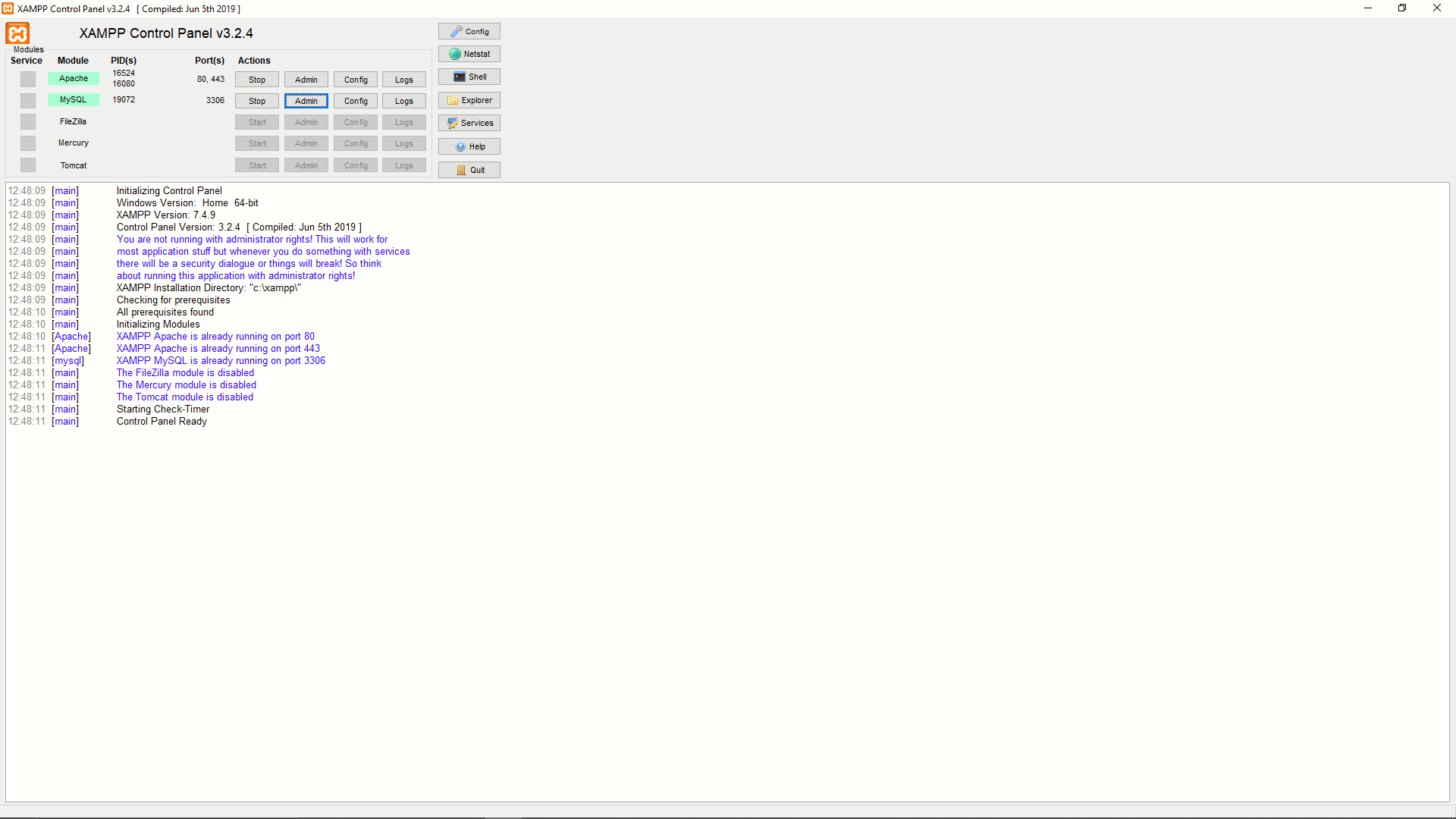
Apache и MySQL.

Рисунок 1 – запуск программ Apache и MySQL.

Переходим в импортированную ранее базу данных в СУБД

phpMyAdmin в меню слева, нажать на кнопку «Новая» под названием и

создаем три новые таблицы

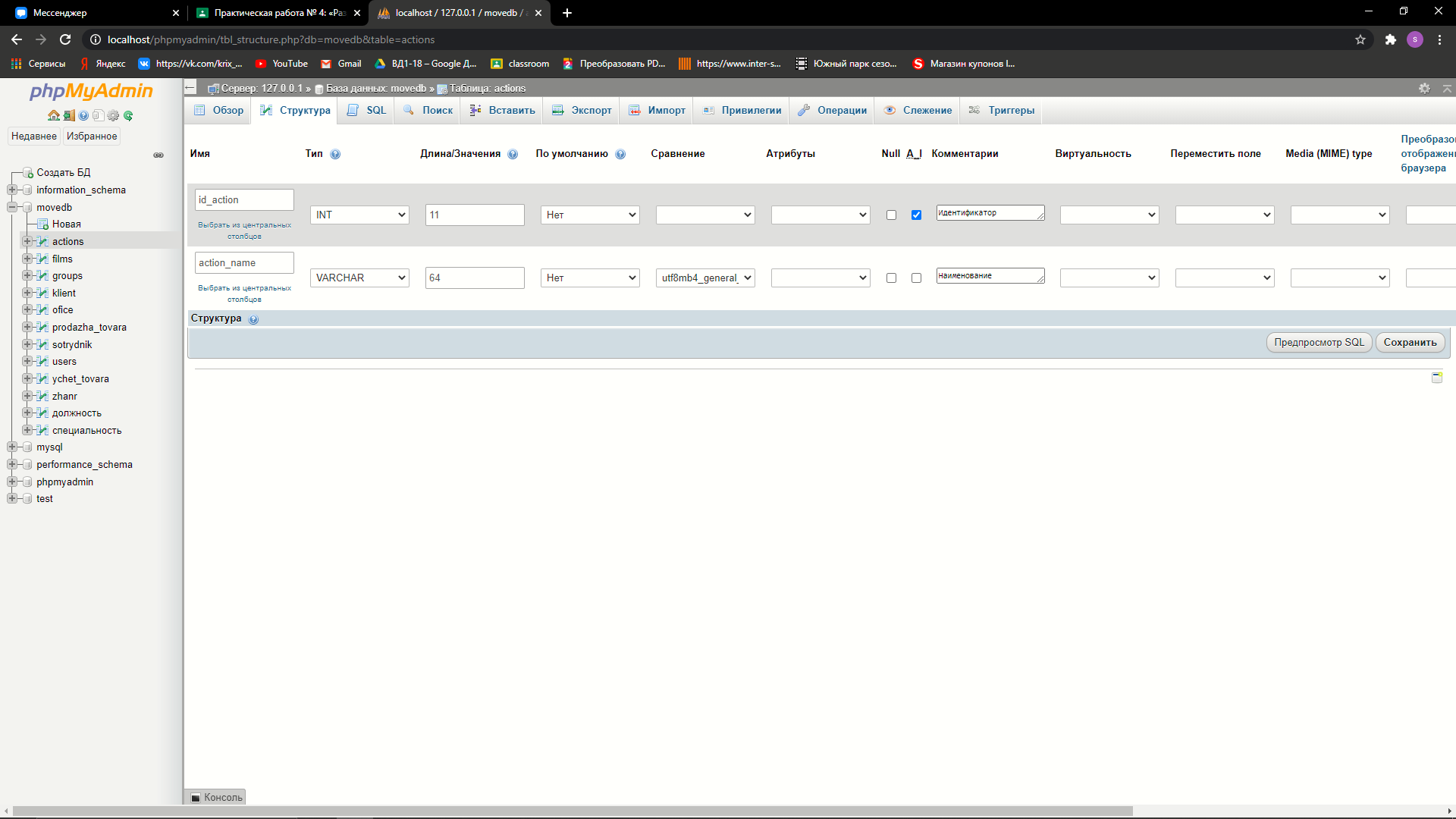


Рисунок 2 – Таблица «Действия»

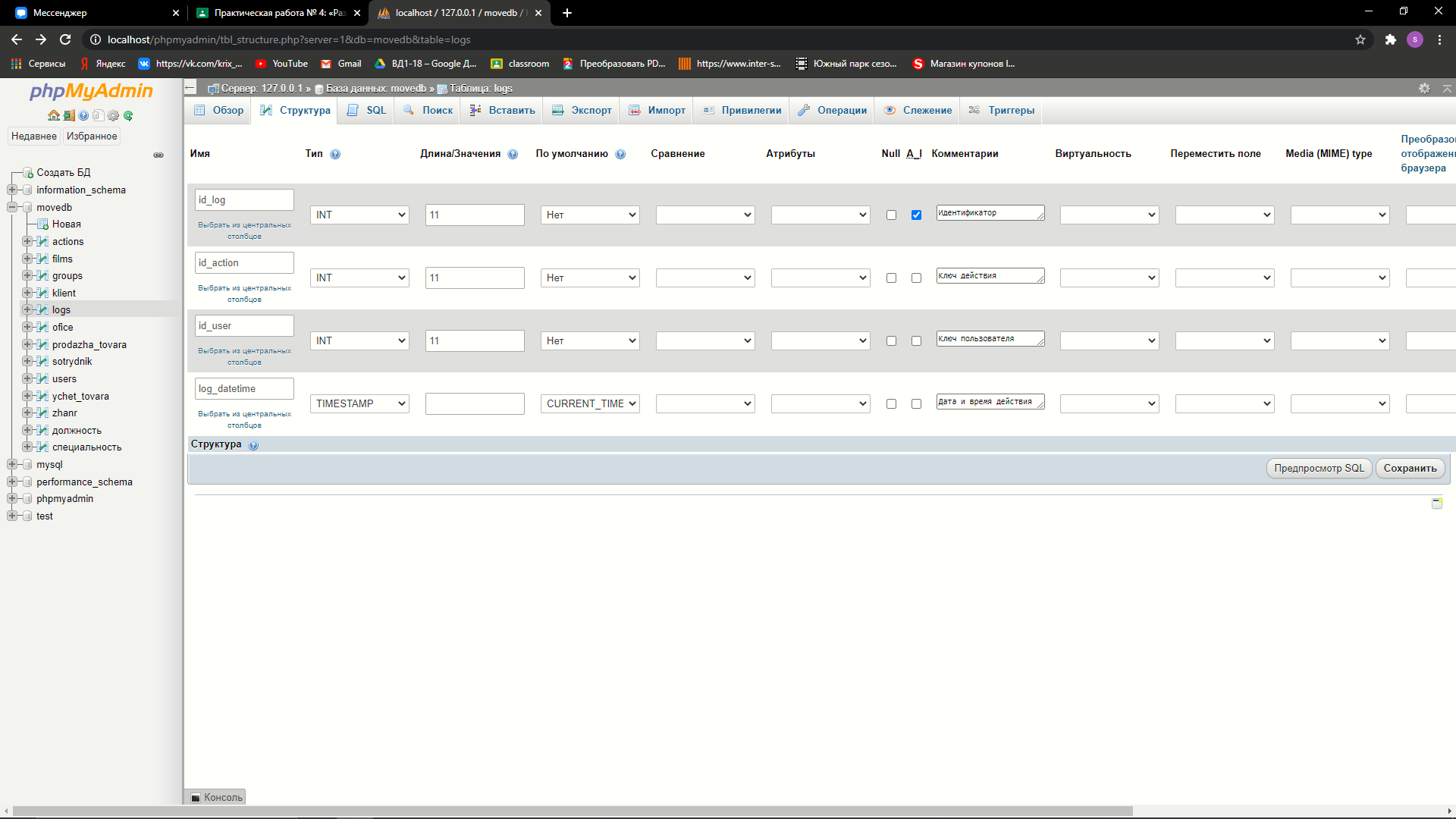


Рисунок 3 – Таблица «Журнал действий»

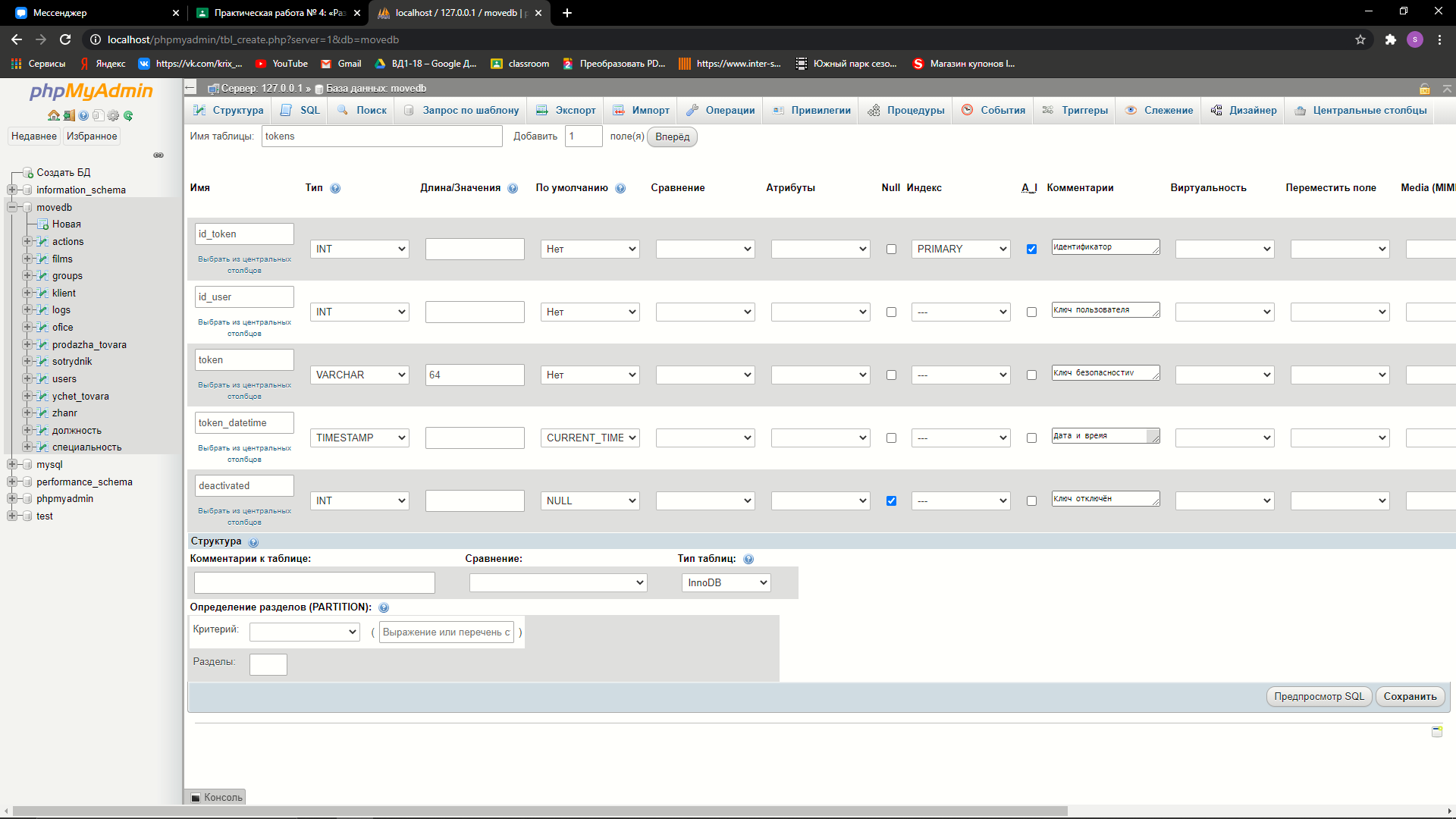


Рисунок 4 – Ключи безопасности

1. Пункт

Убеждаемся в наличии таблицы «Сотрудники», соответствующей правилам нормализации (например, раздельные поля «Фамилия», «Имя» и «Отчество». Последнее поле должно иметь атрибут NULL, что позволяет оставлять его пустым), в ином случае — создать новую или модифицировать существующую. Затем производим модификацию таблицы «Пользователи», добавив в неё ключ сотрудника из таблицы «Сотрудники». Для этого необходимо перейти во вкладку «Структура», добавить одно поле после строки «Ключ группы» и. нажать на кнопку «Вперёд».

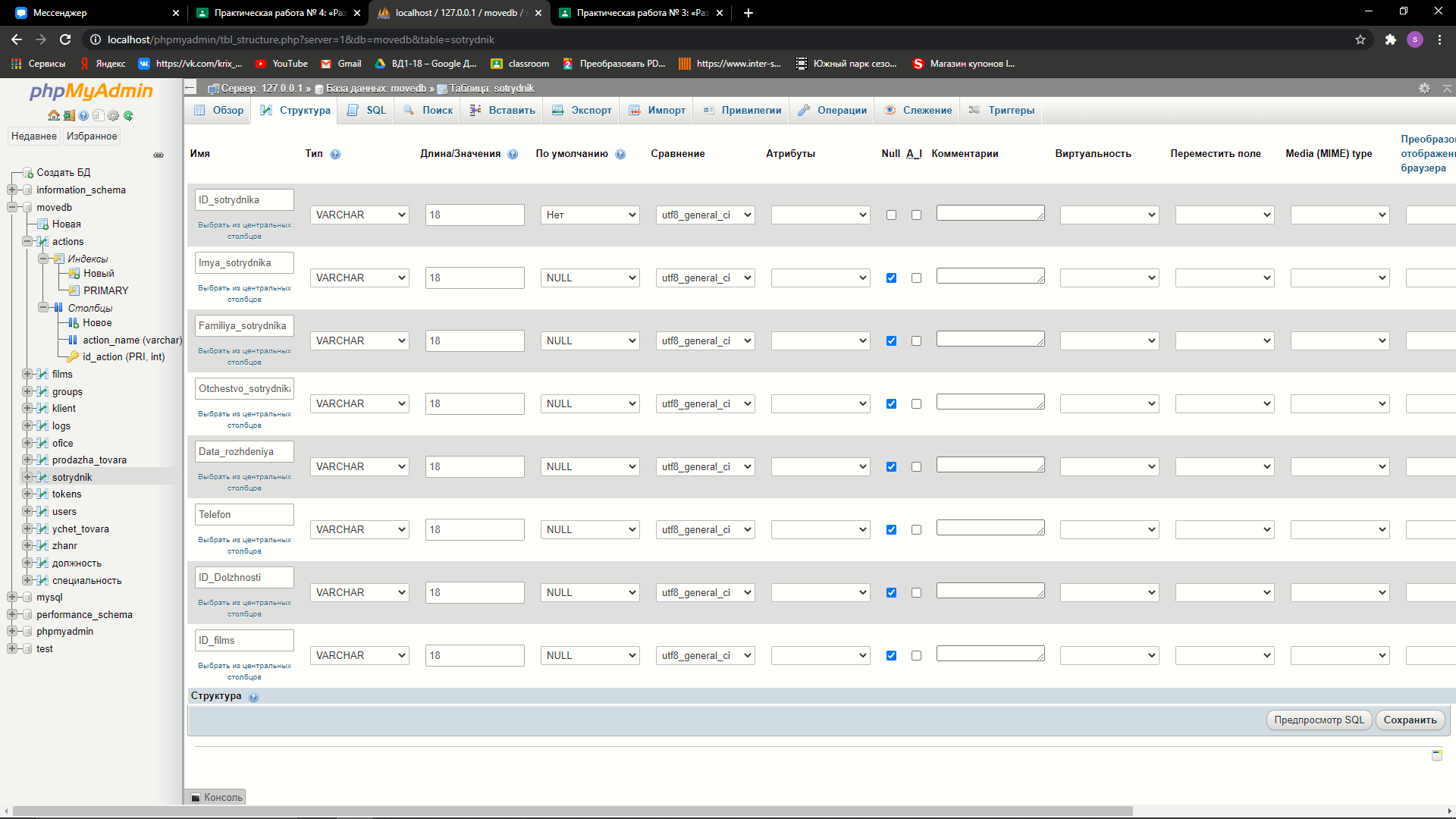
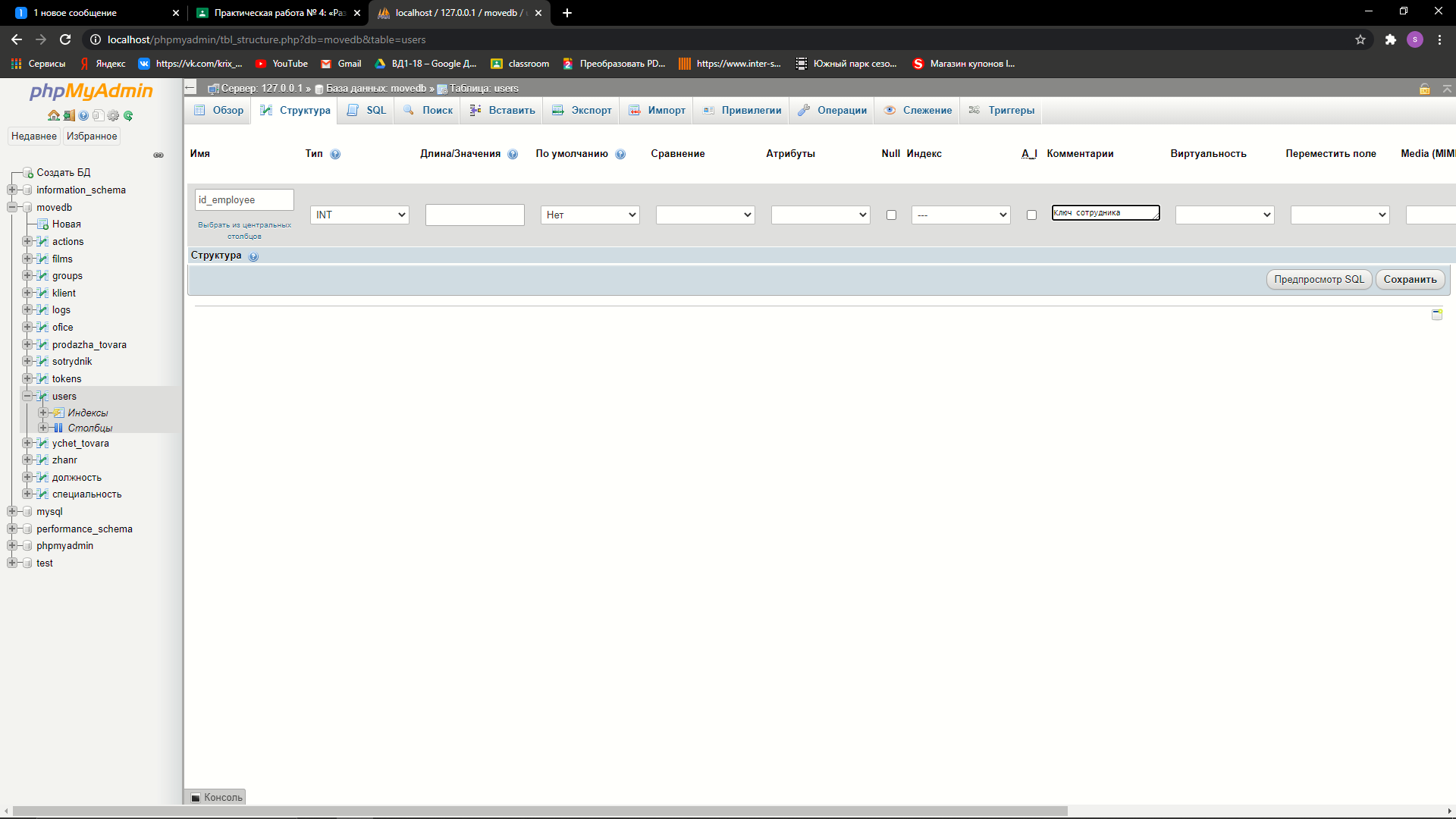
Рисунок 5 – Сотрудники 

Рисунок 6 – Добавление нового поля

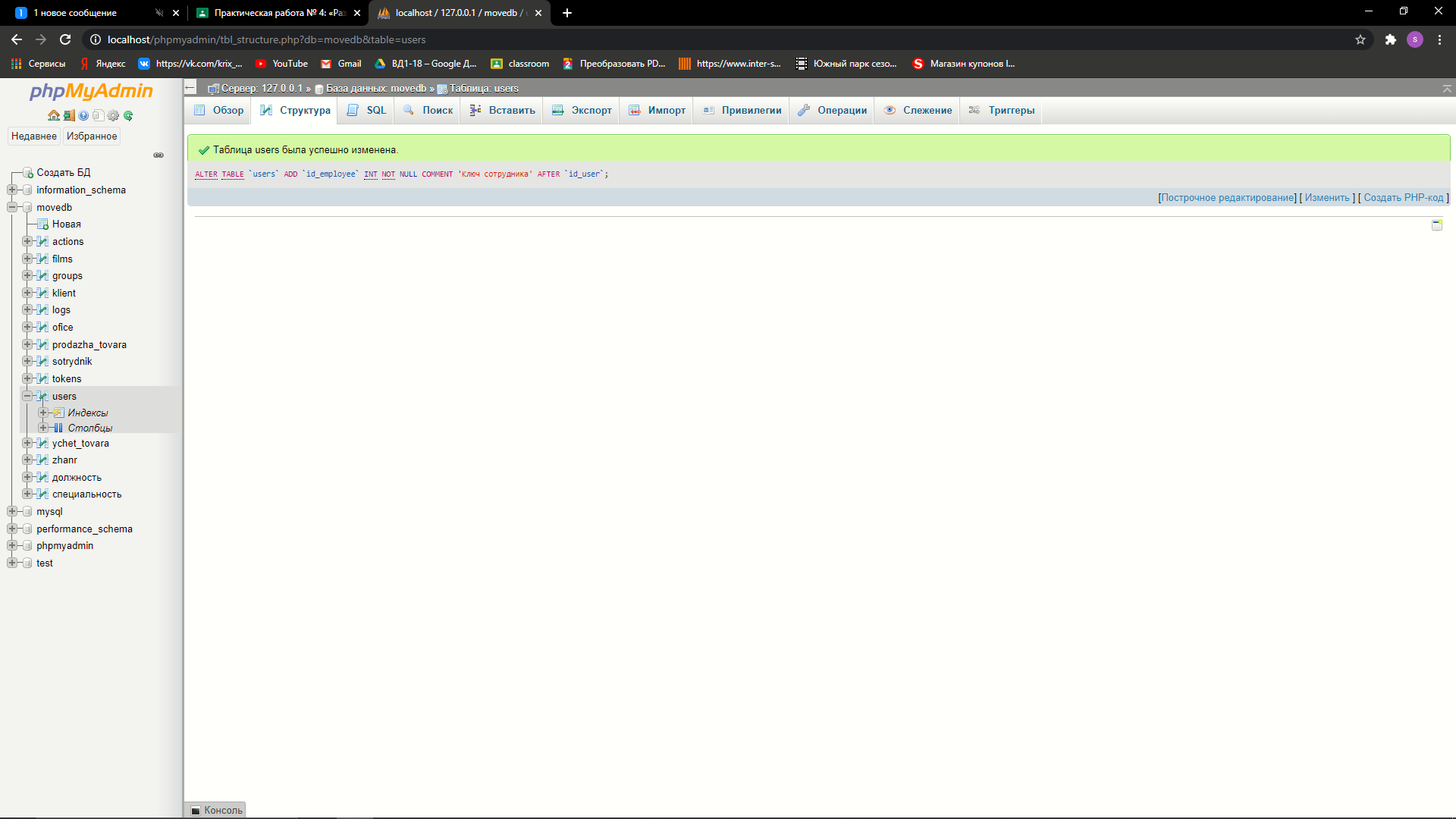


Рисунок 7 – Добавлено

1. Пункт

Связываем созданные и модифицированные таблицы с помощью SQL-

запросов. Затем производим заполнение таблицы «Действия»

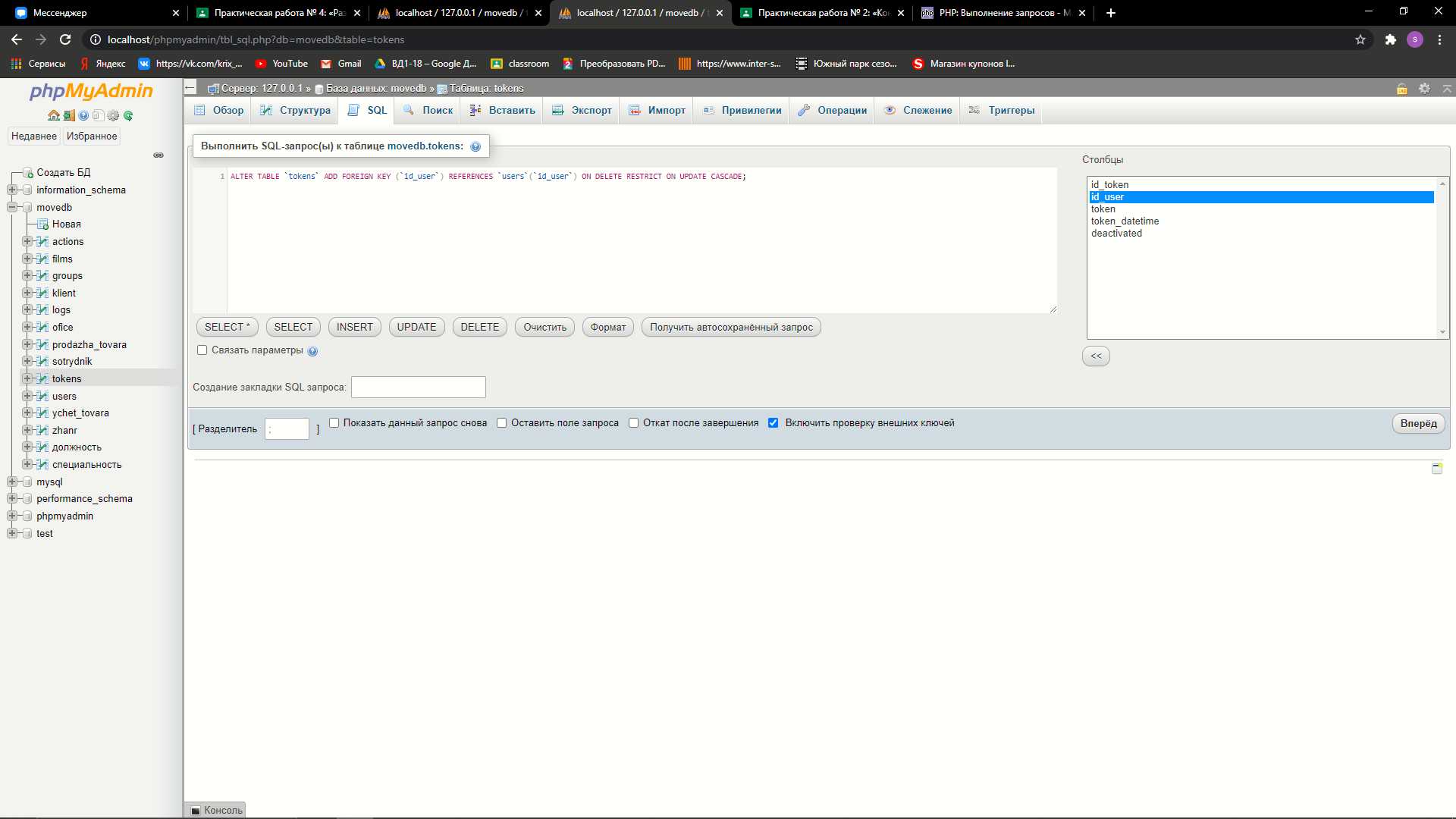


Рисунок 8 – SQL запрос

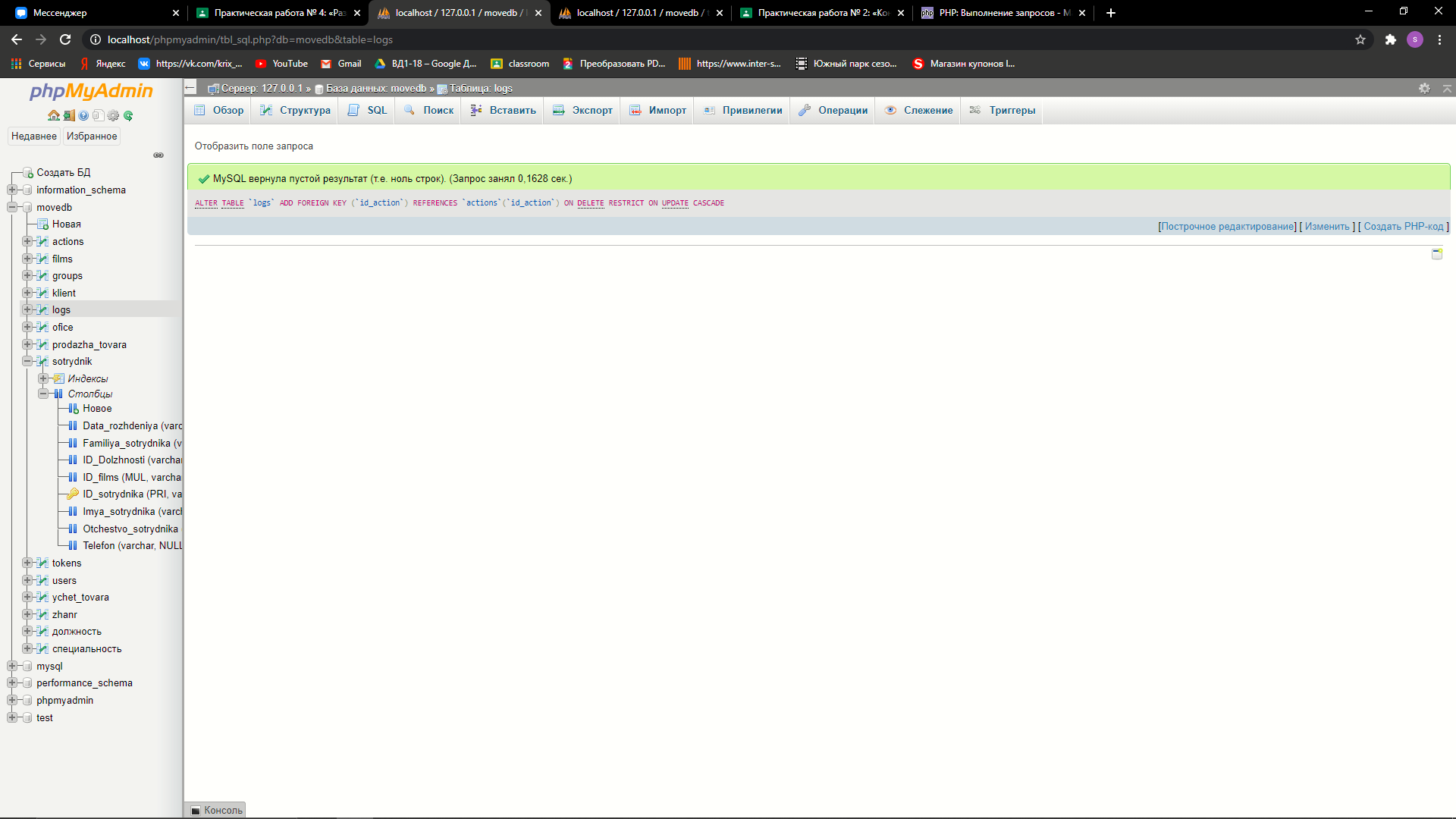


Рисунок 9 – SQL запрос

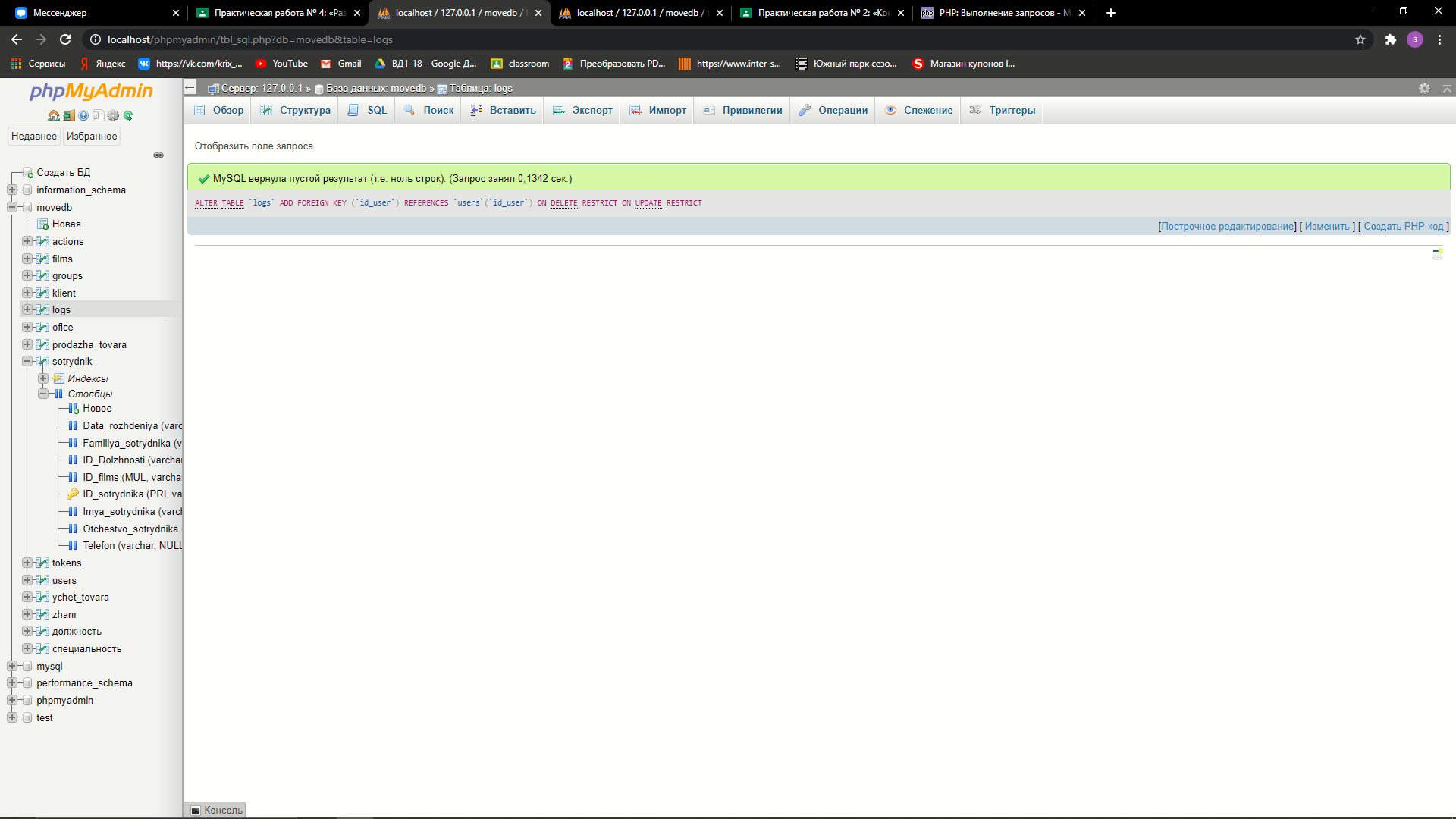


Рисунок 10 – SQL запрос

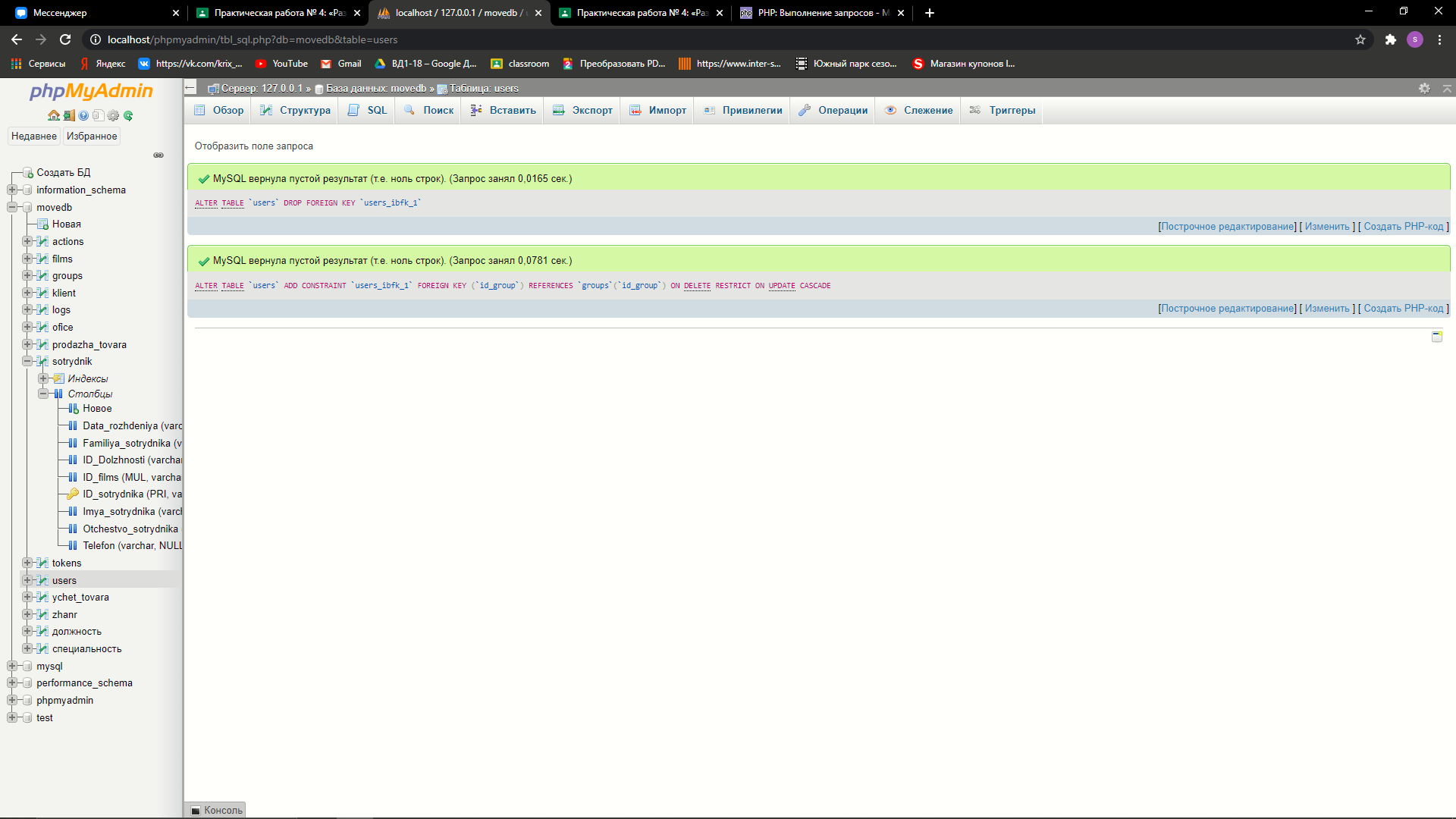


Рисунок 11 – SQL запрос

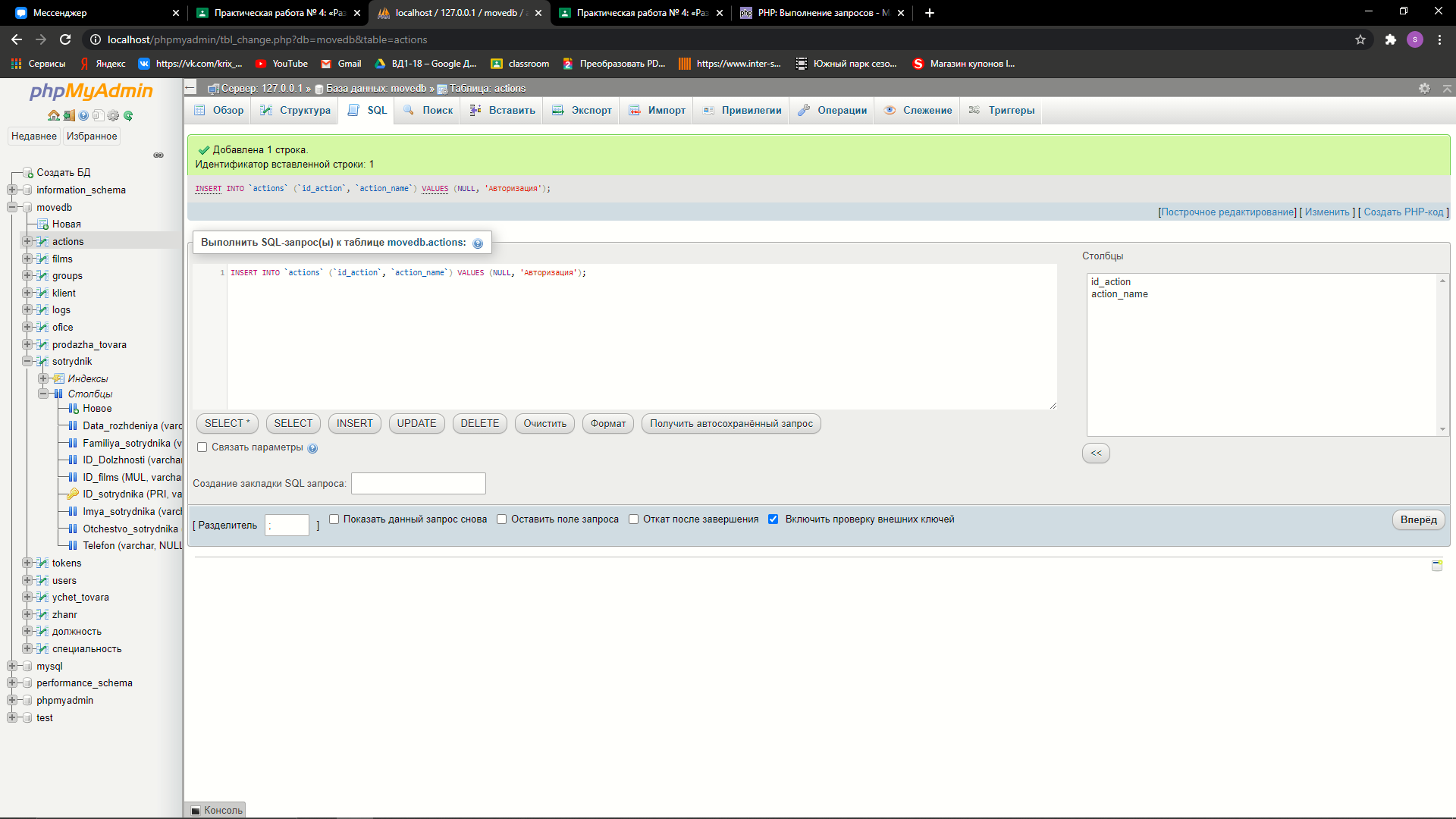


Рисунок 12 – Авторизация

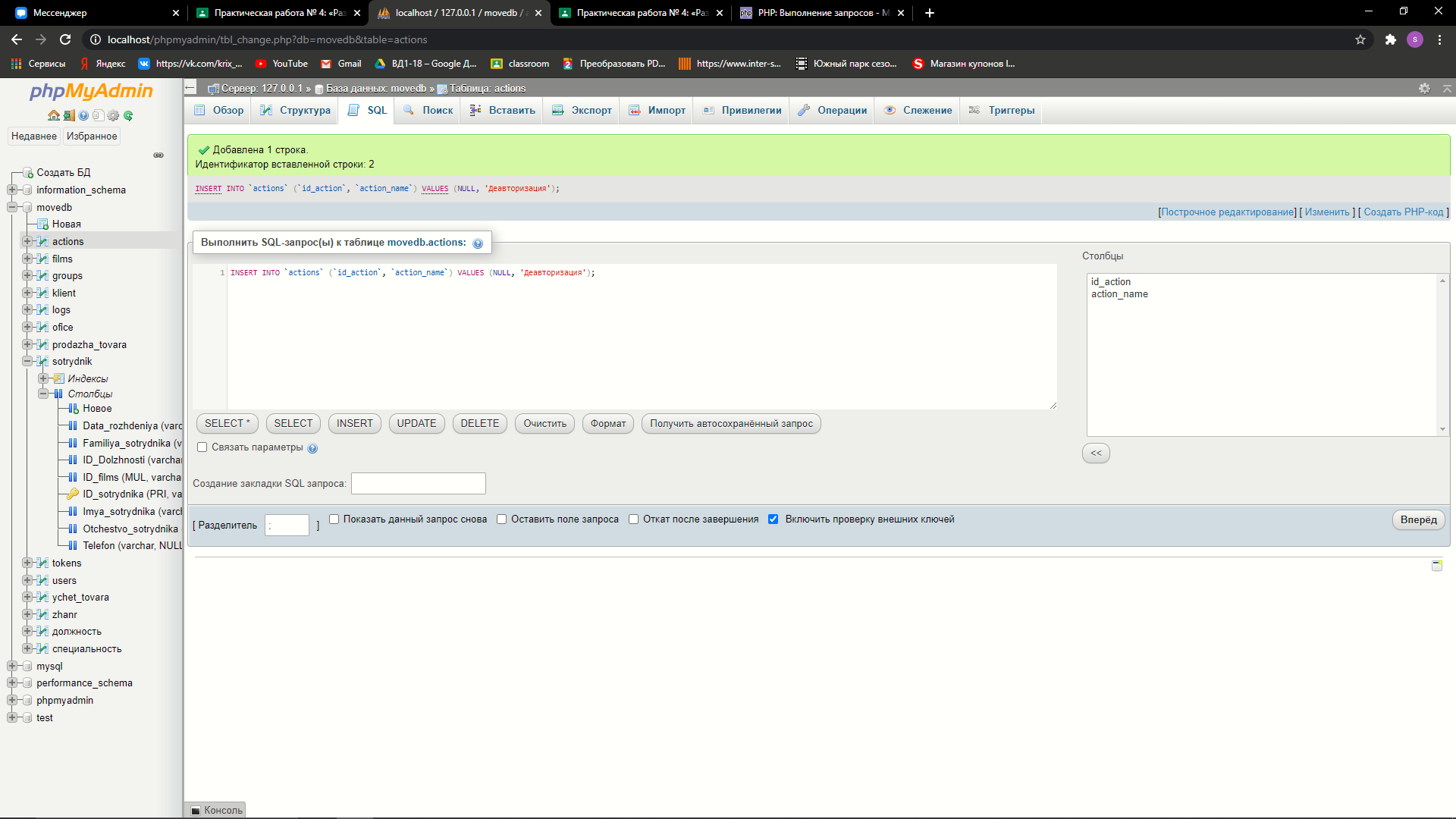


Рисунок 13 – Деавторизация

1. Производим запуск среды разработки Microsoft Visual Studio и запустить созданный ранее проект. В существующем классе Database описываем метод, который позволит добавлять значения в базу данных при вызове

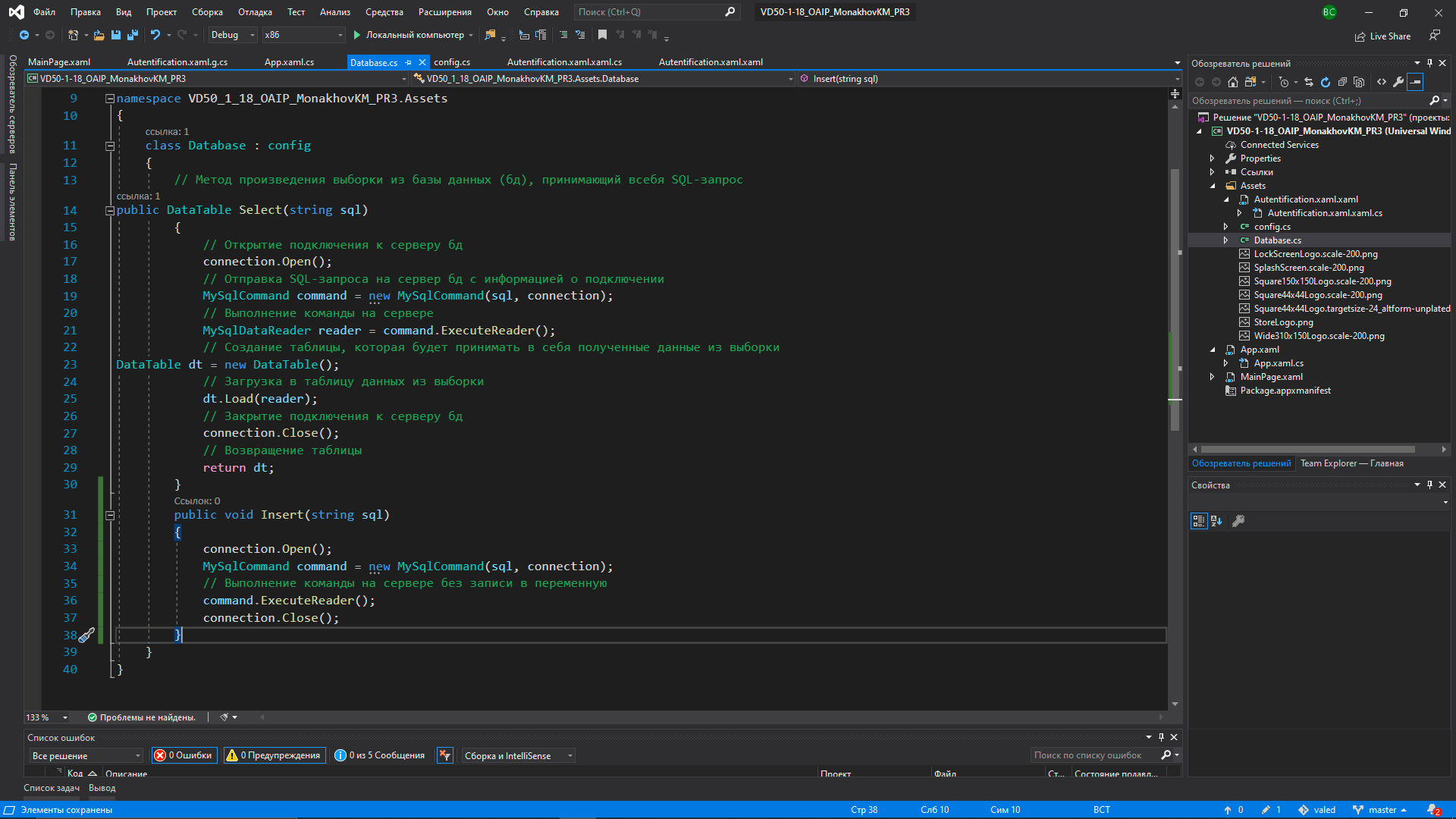


Рисунок 14 – Database

1. Пункт

Нужно объявить в конфигурационном классе две новые переменные, одна из которых использует системные функции для работы с кэшированными данными приложения, а другая для хранения ключа сервера для шифрования передаваемых паролей.

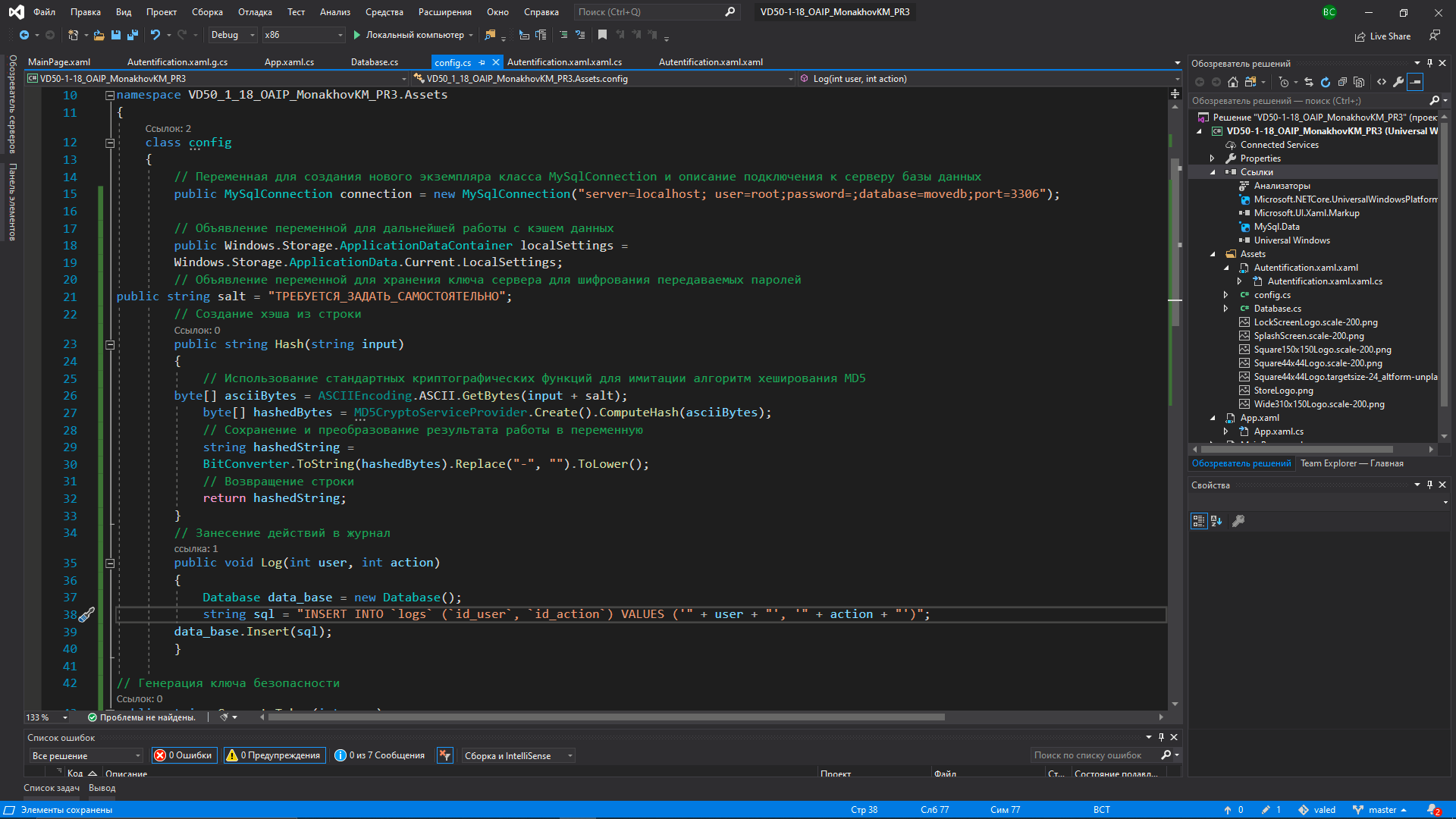


Рисунок 15 – Config.cs

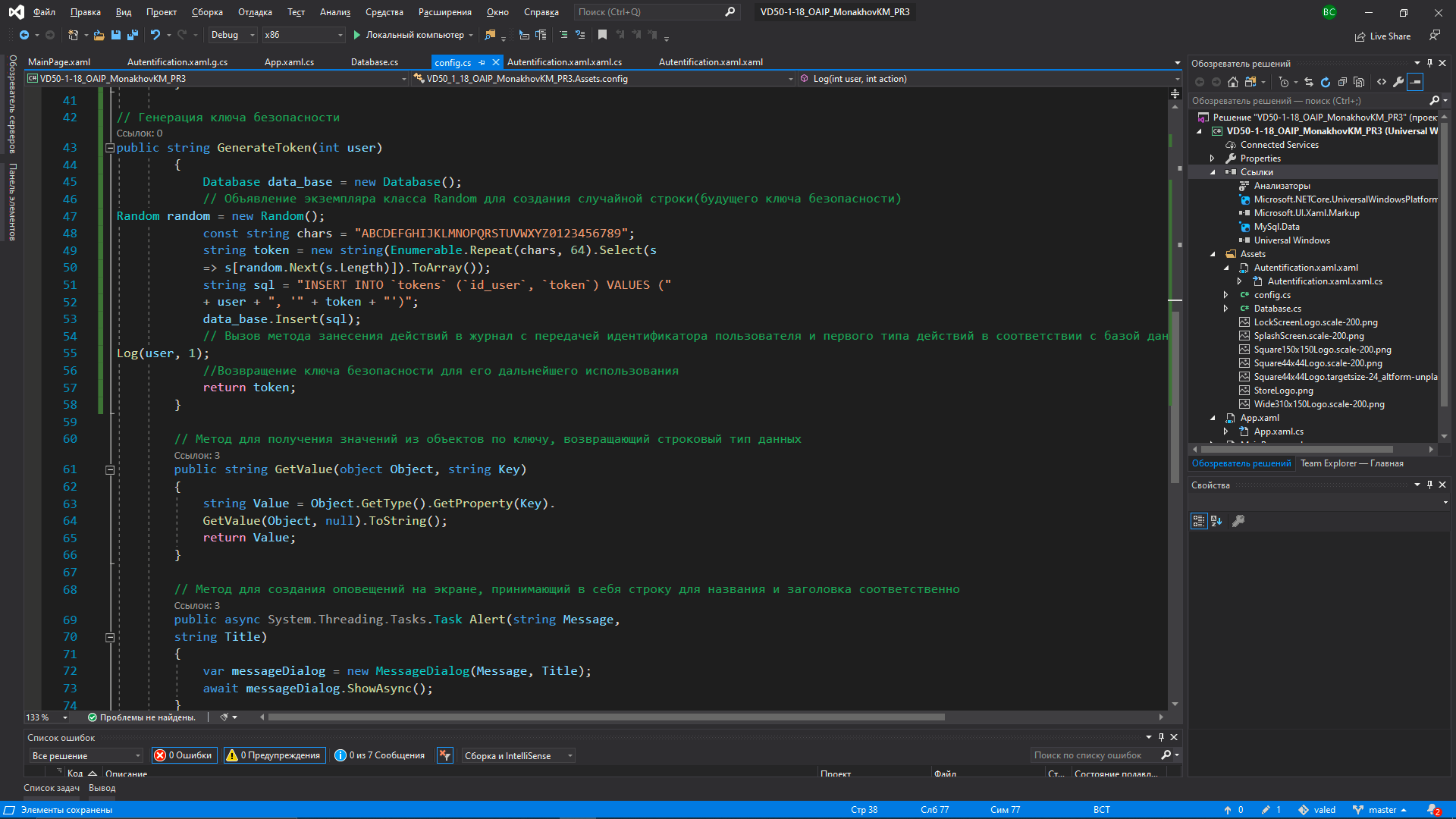


Рисунок 16 – Config.cs

1. Пункт

Далее нам нужно преобразовать созданный ранее класс Data в новый класс

под названием User с наследованием конфигурационного, который будет

храниться в отдельном файле. Произвести создание новых свойств,

соответствующие атрибутам базы данных.

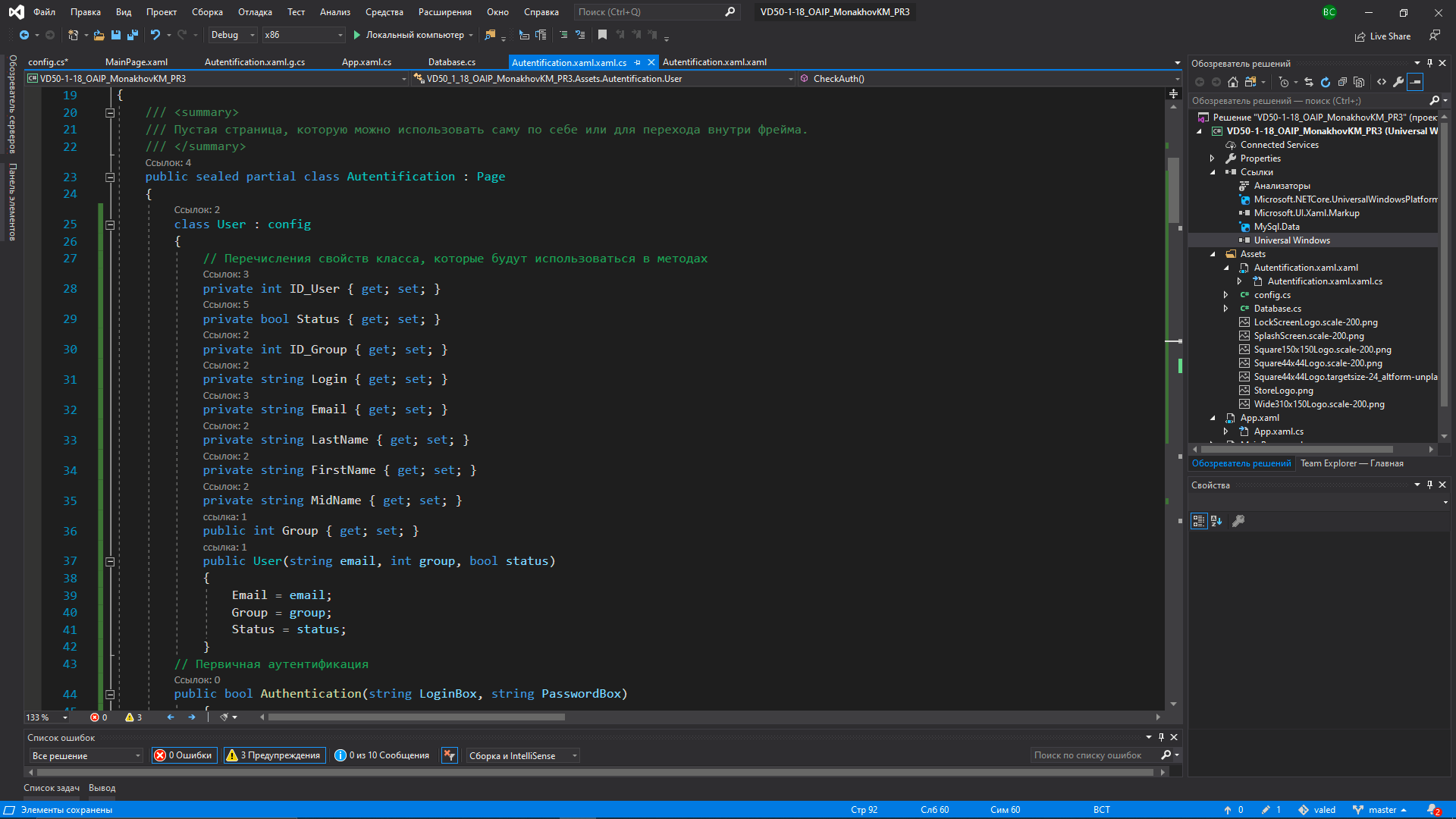


Рисунок 17 – Autentification.xaml.cs

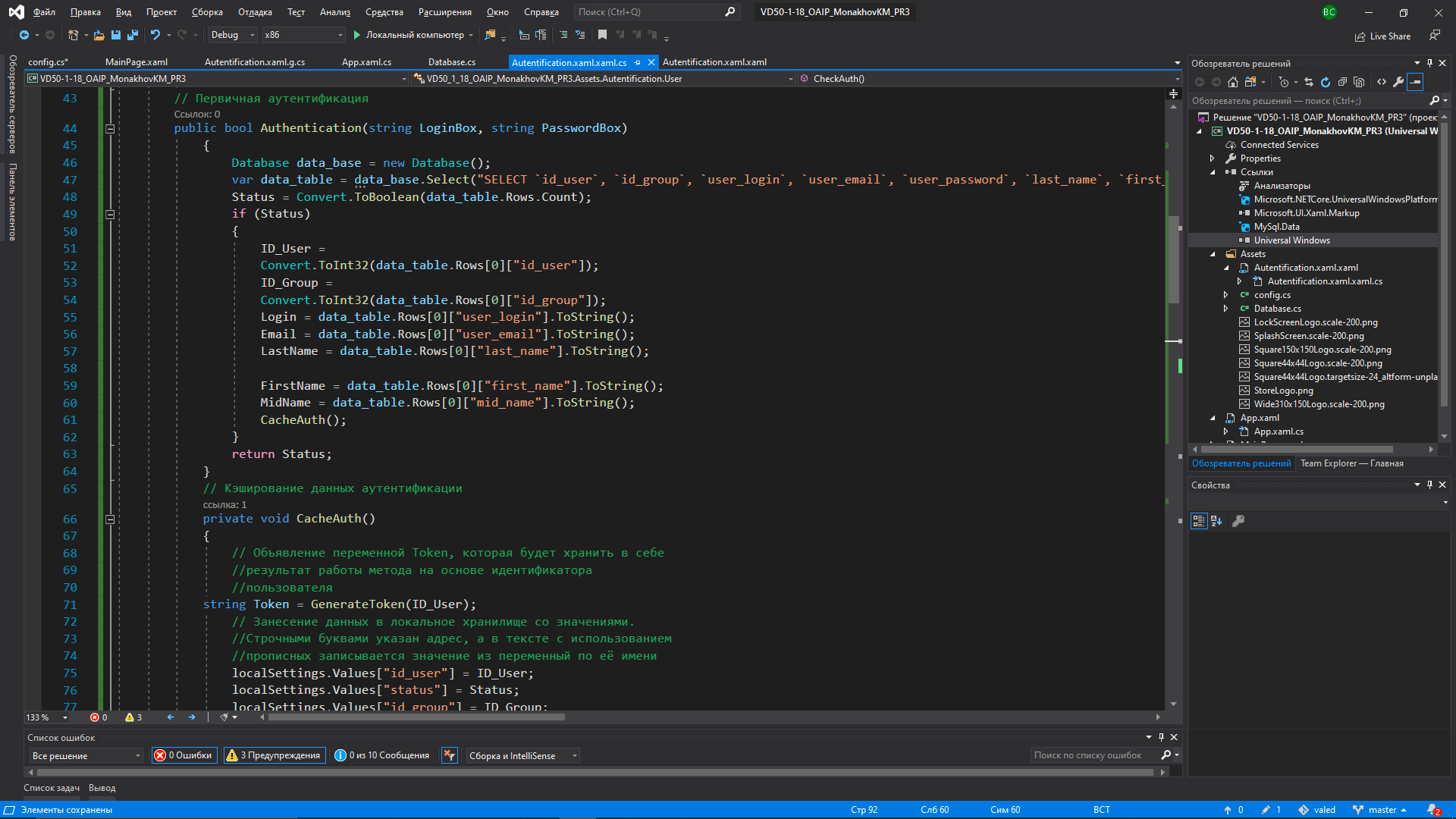


Рисунок 18 – Autentification.xaml.cs

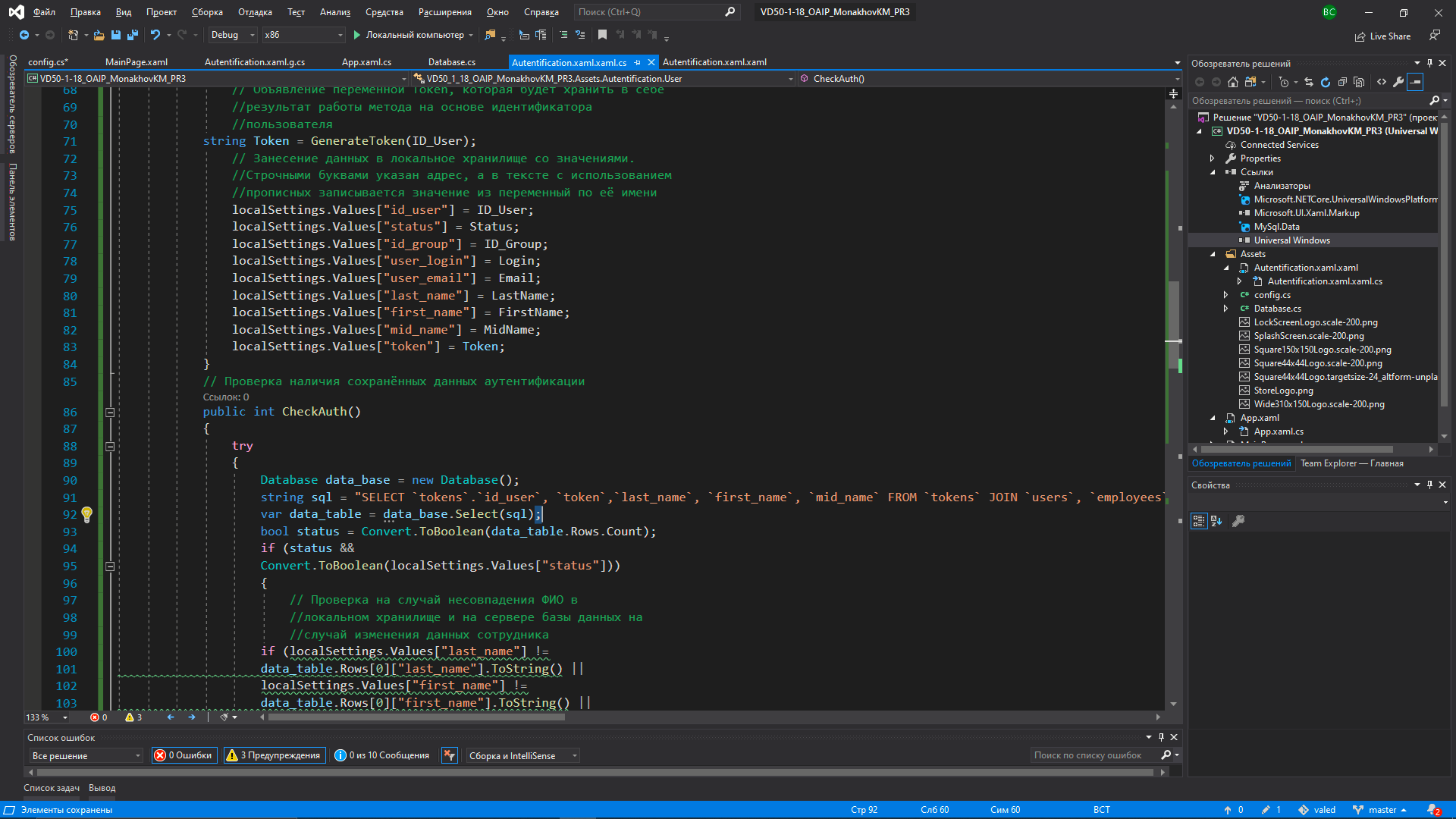


Рисунок 19 – Autentification.xaml.cs

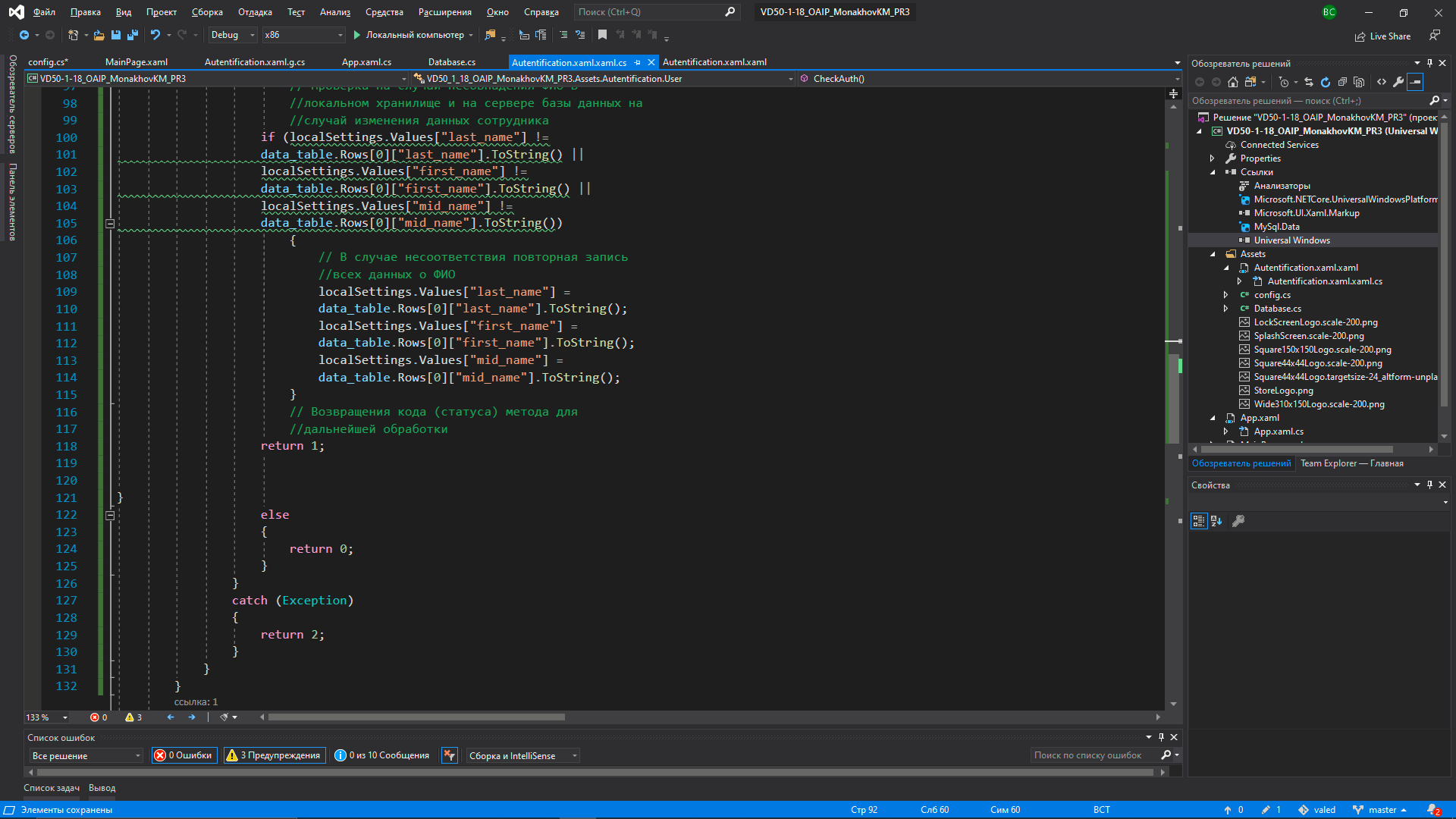


Рисунок 20 – Autentification.xaml.cs



Рисунок 21 – Autentification.xaml.cs

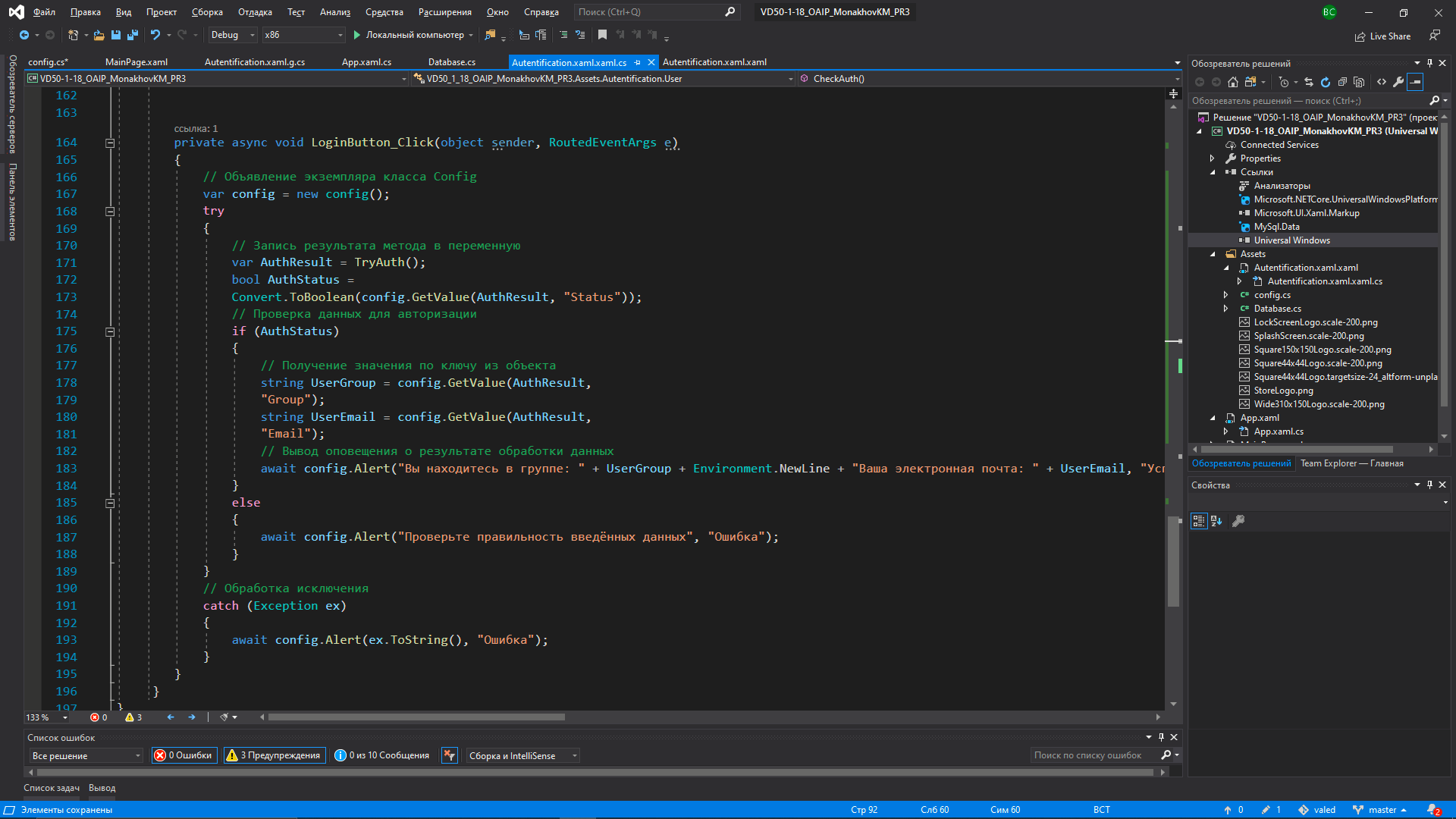


Рисунок 22 – Autentification.xaml.cs

1. Пункт

Требуется создать новую страницу для описания интерфейса пользователя после успешного прохождения процесса аутентификации в приложении под названием PanelFrame.xaml.



Рисунок 23 – PanelFrame.xaml

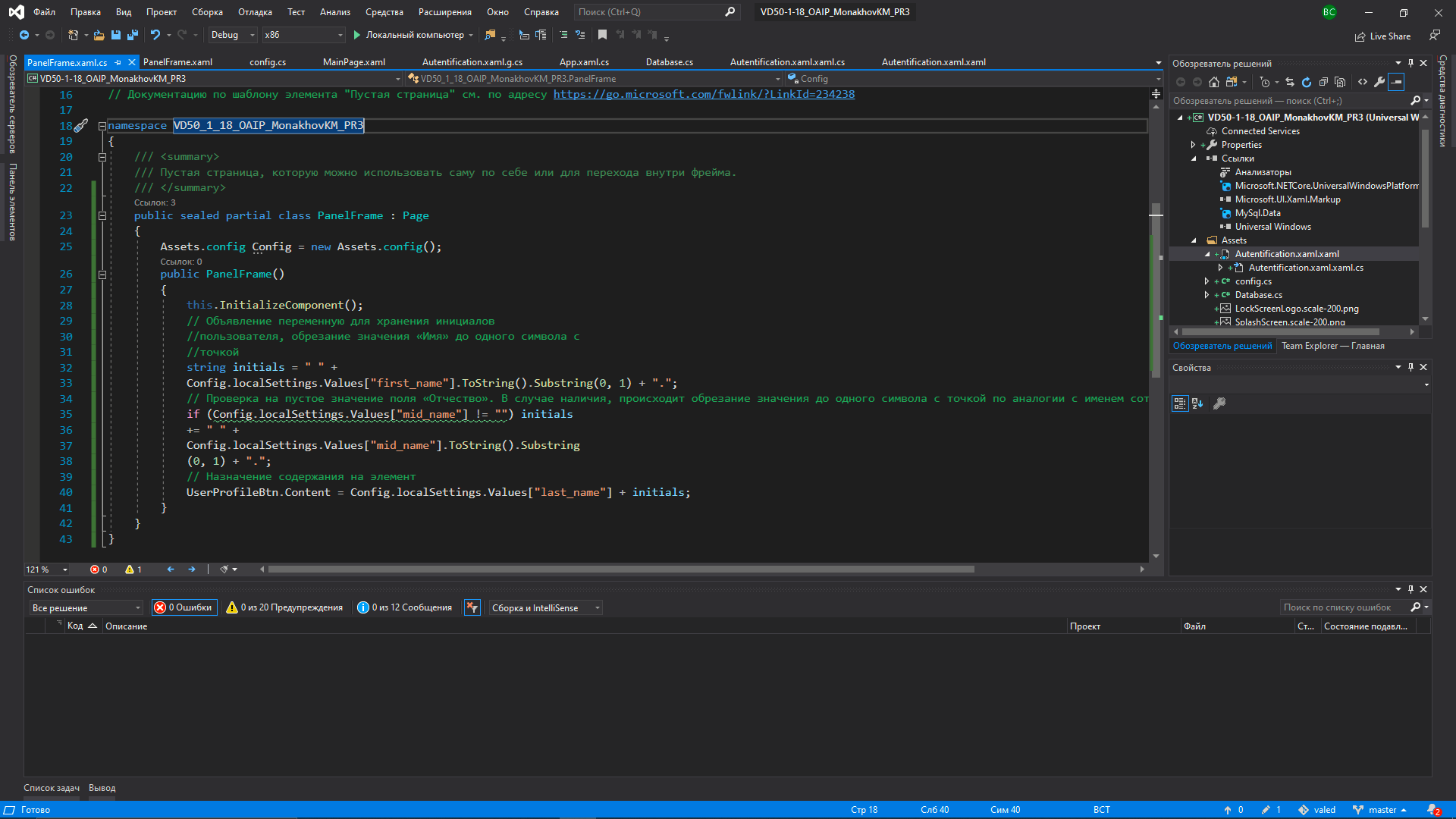


Рисунок 24 – PanelFrame.xaml

8. Преобразовываем программный код страниц Authentication и MainPage в соответствии с новыми методами.

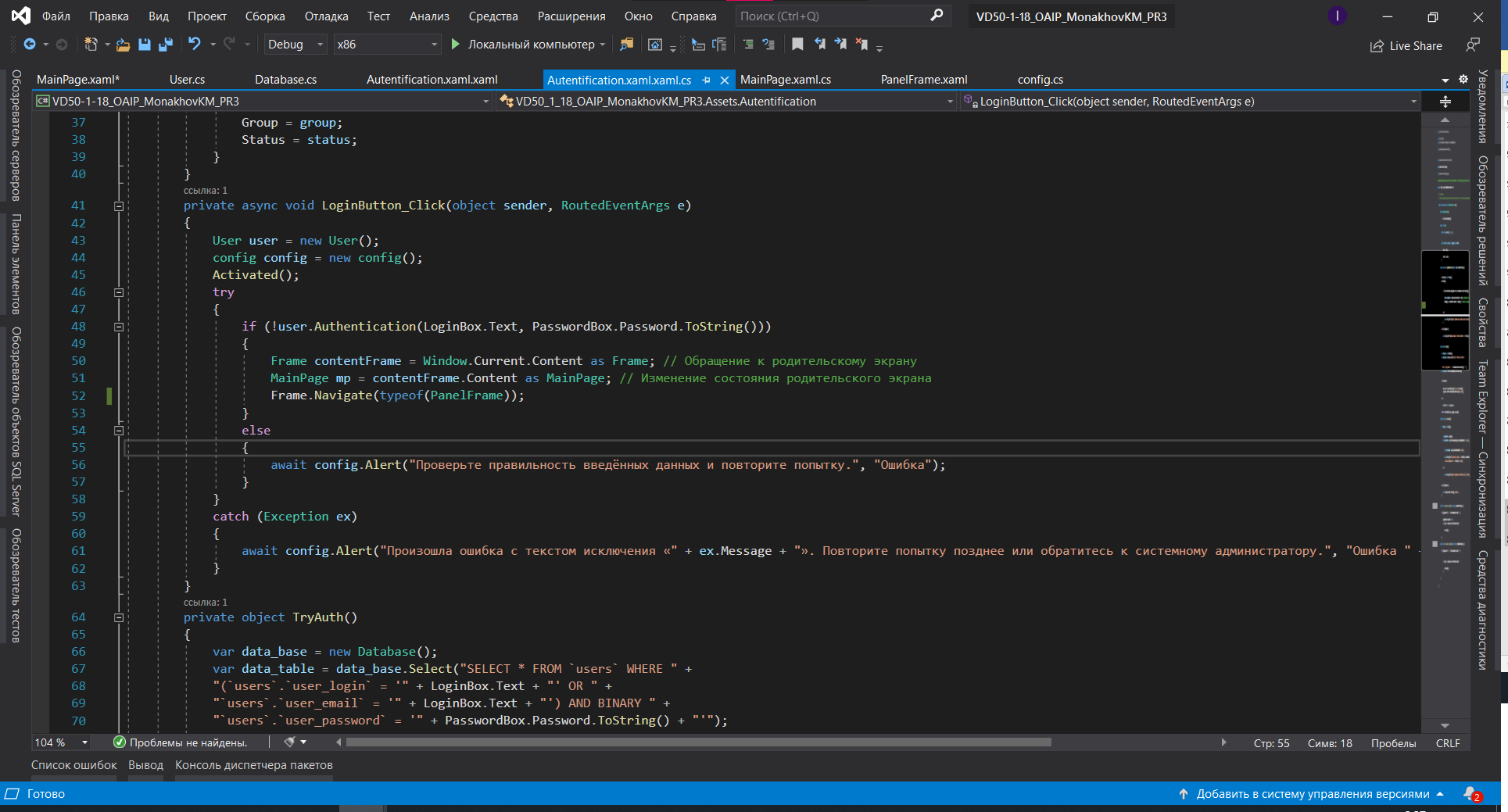


Рисунок 16 - Код Authentication

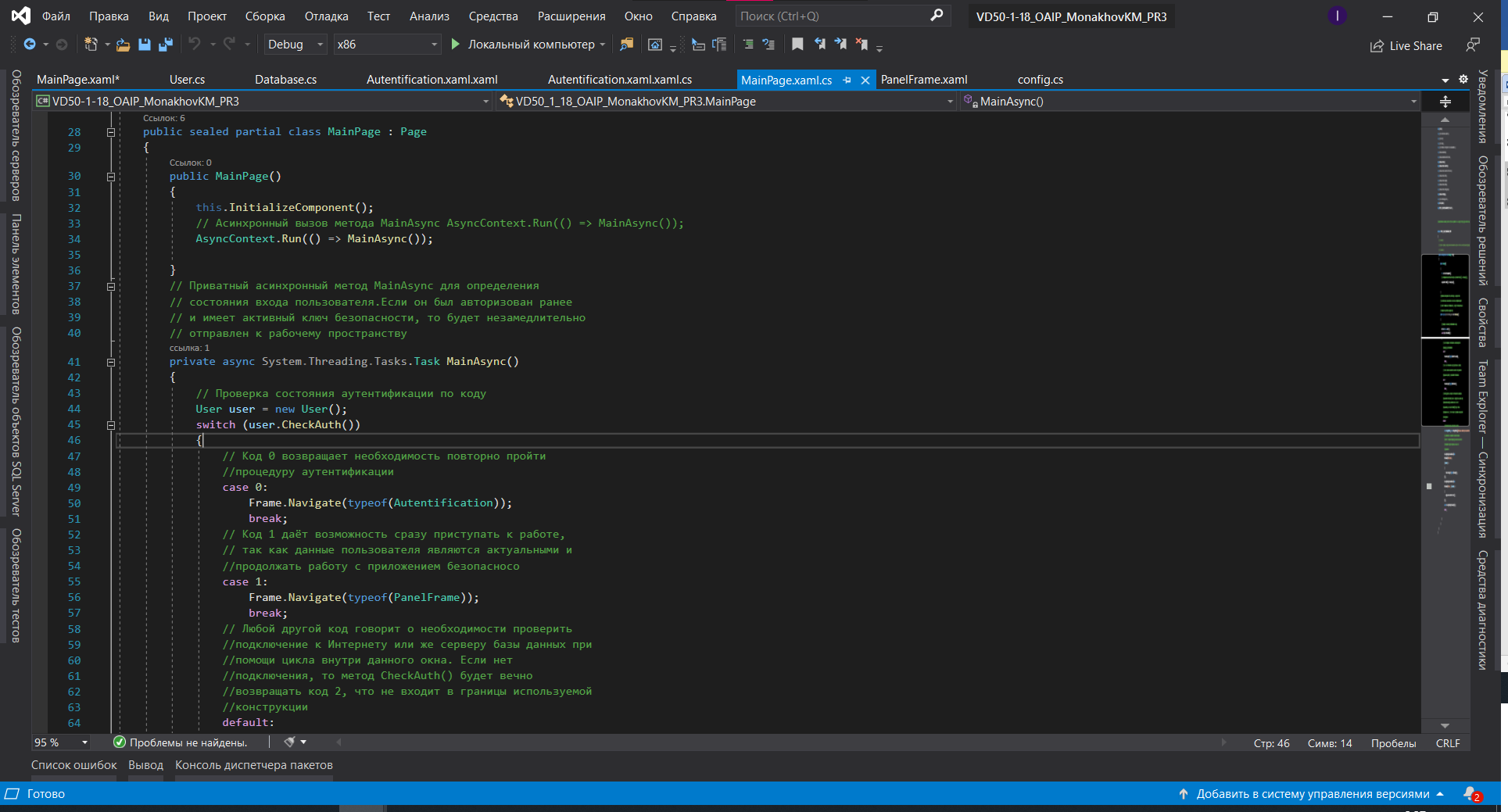


Рисунок 17 - Измененный код MainPage.xaml.cs

9. В папке локального сервера по пути «C:\xampp\htdocs» создаем текстовый файл «hash.php» для локального шифрования паролей пользователя. Затем в интернет-браузере переходим по URL-адресу «http://localhost/hash.php?password=» и после знака равенства вставить текущий пароль пользователя информационной системы. Копируем получившиеся символы и заносим их в поле «Пароль» таблицы «Пользователи».

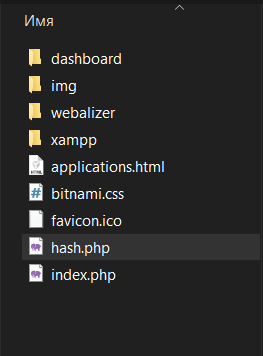


Рисунок 18 - Создание файла php

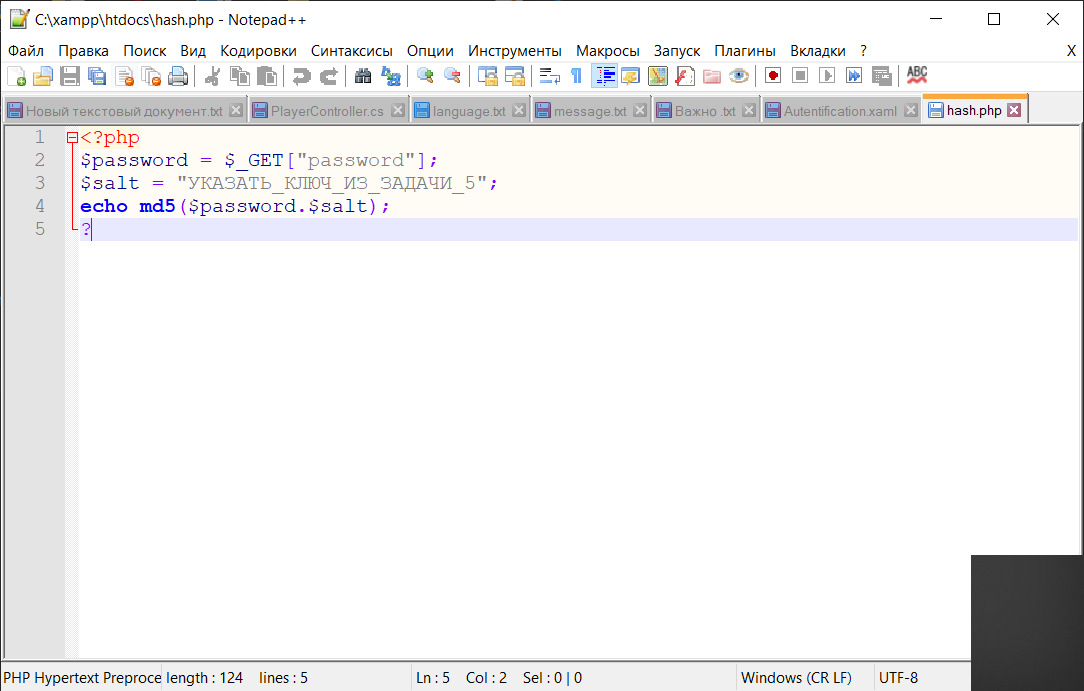


Рисунок 19 - Код файла

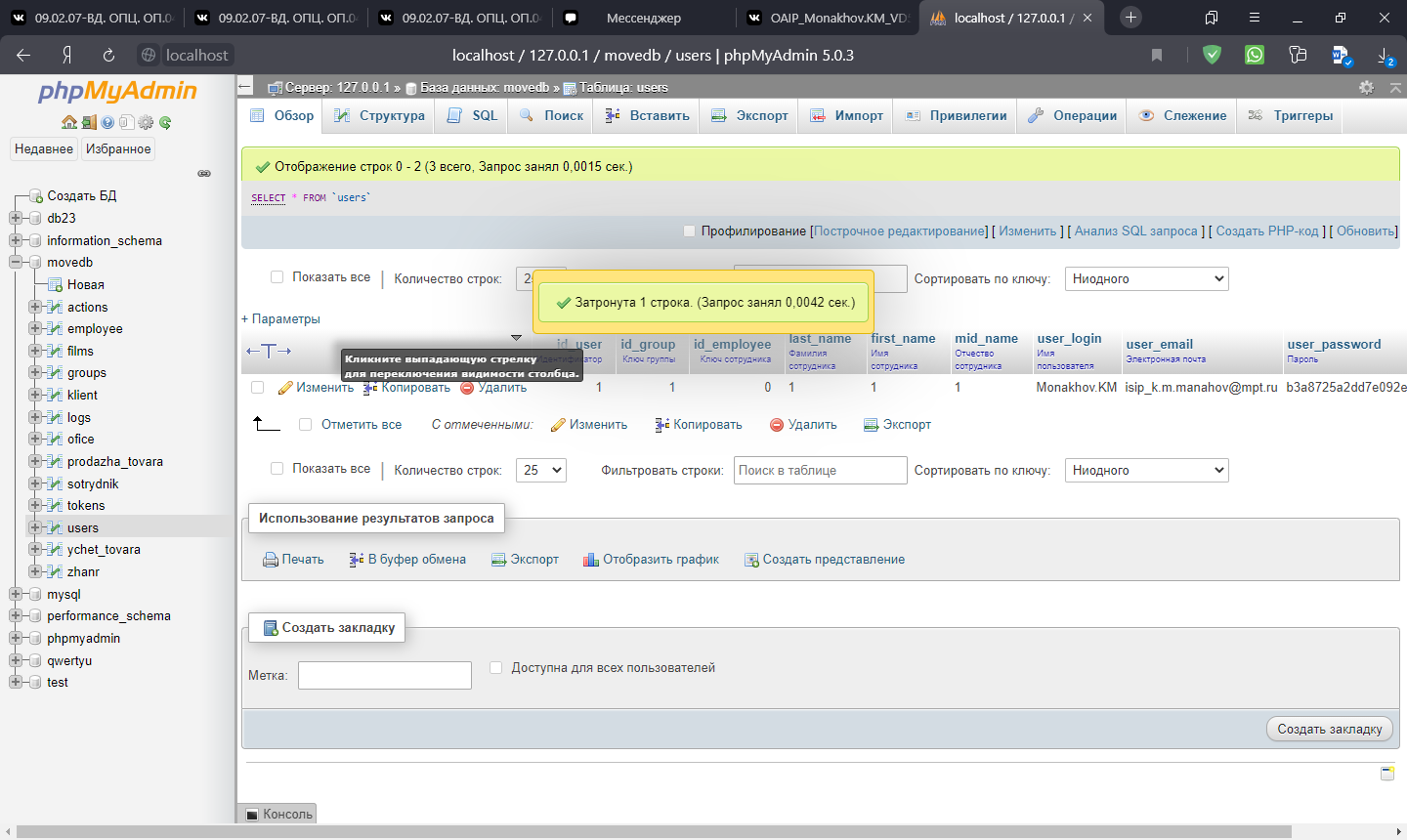


Рисунок 20 - Изменение пароля в таблице

10. Произведем проверку приложения на работоспособность. При запуске приложения не срабатывает строчка кода с ошибкой «Вызов был отклонен».

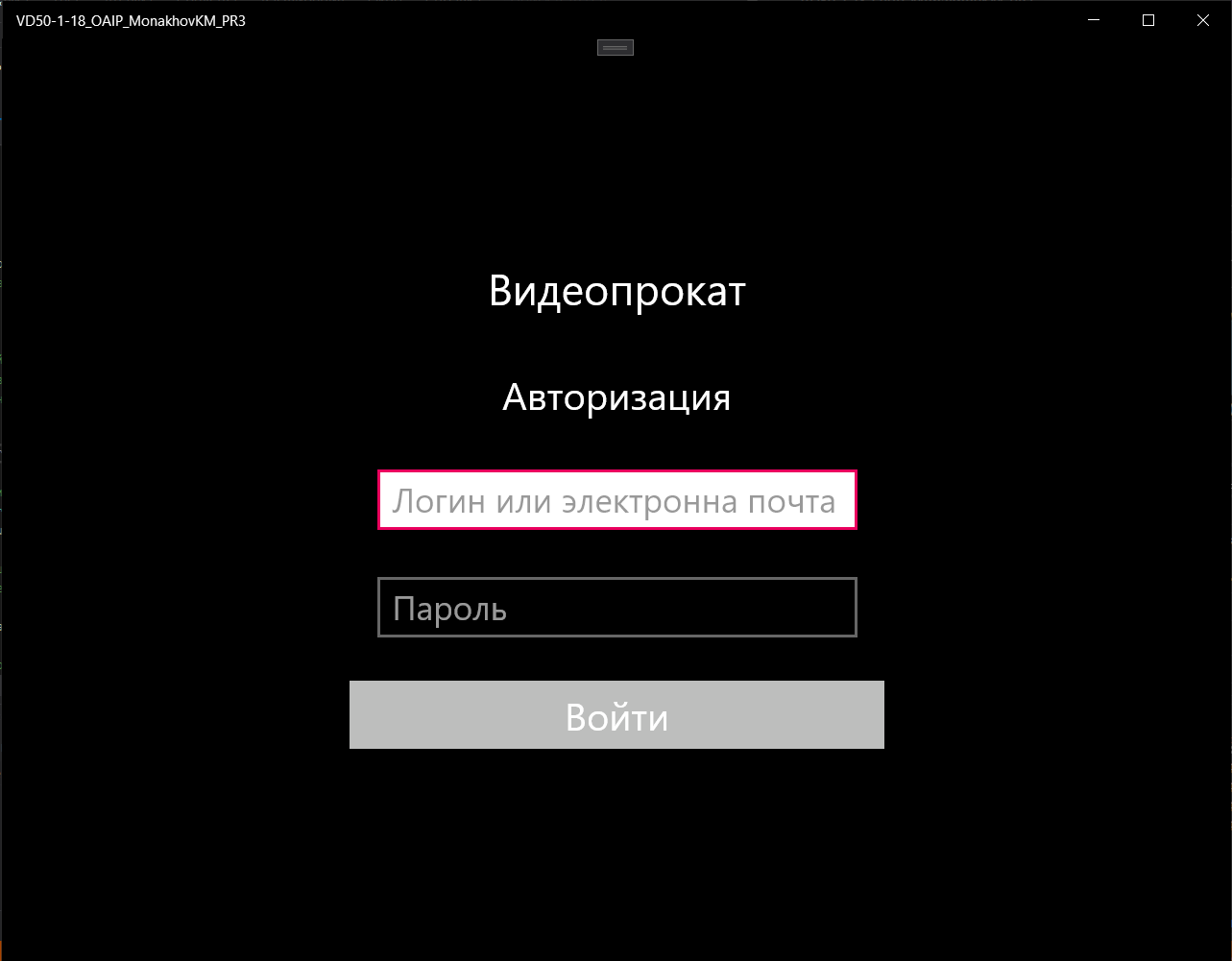


Рисунок 21 – Запущенное приложение

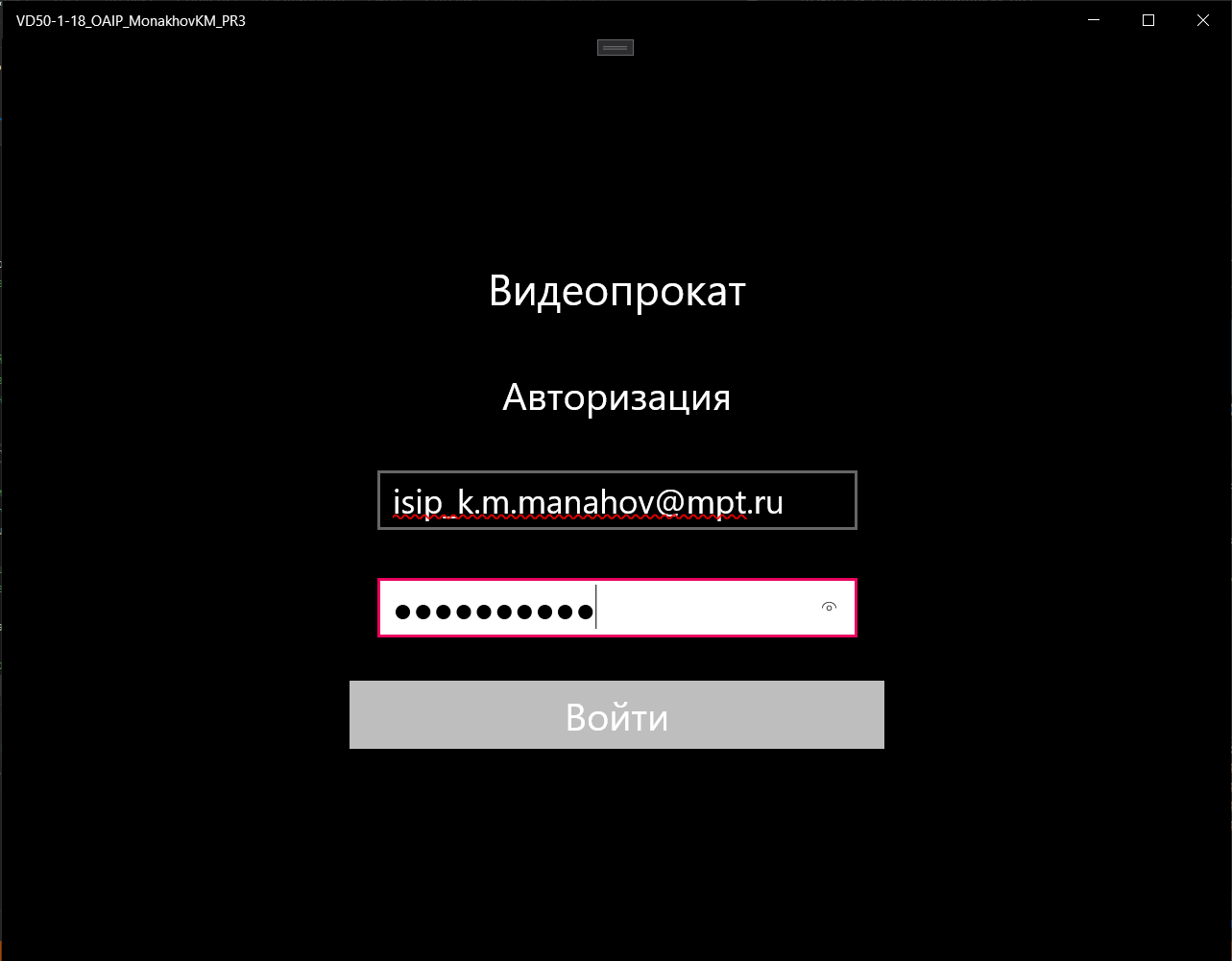


Рисунок 22 – Ввод данных

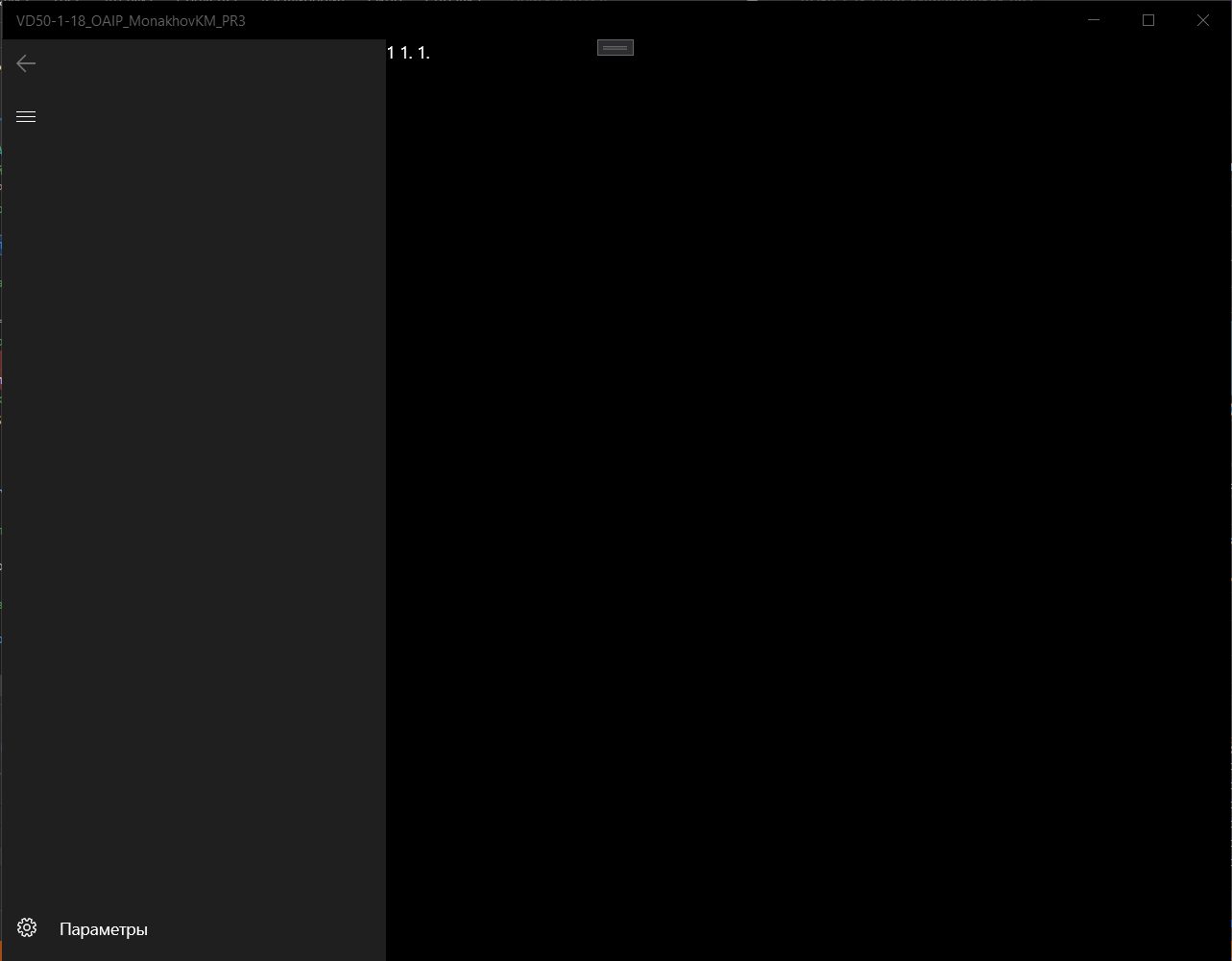


Рисунок 23 – Успешный вход в систему

|  |  |
| --- | --- |
| Autentification.xaml | <Page  x:Class="VD50\_1\_18\_OAIP\_MonakhovKM\_PR3.Assets.Autentification"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  xmlns:local="using:VD50\_1\_18\_OAIP\_MonakhovKM\_PR3.Assets"  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"  mc:Ignorable="d"  Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">  <Grid>  <Grid Padding="20" Width="500px" Height="500px" BorderBrush="Black" BorderThickness="7">  <Grid.RowDefinitions>  <RowDefinition ></RowDefinition>  <RowDefinition></RowDefinition>  <RowDefinition></RowDefinition>  <RowDefinition></RowDefinition>  <RowDefinition></RowDefinition>  </Grid.RowDefinitions>  <TextBlock Text="Видеопрокат" Grid.Row="0" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="36px"></TextBlock>  <TextBlock Text="Авторизация" Grid.Row="1" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="32px"></TextBlock>  <TextBox x:Name="LoginBox" Grid.Row="2" PlaceholderText="Логин или электронна почта" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="28px" Width="400px"></TextBox>  <PasswordBox x:Name="PasswordBox" Grid.Row="3" PlaceholderText="Пароль" FontSize="28px" HorizontalAlignment="Center" VerticalAlignment="Center" Width="400px"></PasswordBox>  <Button x:Name="LoginButton" Grid.Row="4" Content="Войти" Click="LoginButton\_Click" Background="#bdbebd" FontSize="32px" HorizontalAlignment="Center" VerticalAlignment="Center" Width="500px"></Button>  </Grid>  </Grid>  </Page> |
| Autentificationю.cs | using System;  using System.Collections.Generic;  using System.IO;  using System.Linq;  using System.Runtime.InteropServices.WindowsRuntime;  using Windows.Foundation;  using Windows.Foundation.Collections;  using Windows.UI.Xaml;  using Windows.UI.Xaml.Controls;  using Windows.UI.Xaml.Controls.Primitives;  using Windows.UI.Xaml.Data;  using Windows.UI.Xaml.Input;  using Windows.UI.Xaml.Media;  using Windows.UI.Xaml.Navigation;  // Документацию по шаблону элемента "Пустая страница" см. по адресу https://go.microsoft.com/fwlink/?LinkId=234238  namespace VD50\_1\_18\_OAIP\_MonakhovKM\_PR3.Assets  {  /// <summary>  /// Пустая страница, которую можно использовать саму по себе или для перехода внутри фрейма.  /// </summary>  public sealed partial class Autentification : Page  {  public Autentification()  {  this.InitializeComponent();  }  public class Data  {  public string Email { get; set; }  public int Group { get; set; }  public bool Status { get; set; }  public Data(string email, int group, bool status)  {  Email = email;  Group = group;  Status = status;  }  }  private async void LoginButton\_Click(object sender, RoutedEventArgs e)  {  User user = new User();  config config = new config();  Activated();  try  {  if (!user.Authentication(LoginBox.Text, PasswordBox.Password.ToString()))  {  Frame contentFrame = Window.Current.Content as Frame; // Обращение к родительскому экрану  MainPage mp = contentFrame.Content as MainPage; // Изменение состояния родительского экрана  Frame.Navigate(typeof(PanelFrame));  }  else  {  await config.Alert("Проверьте правильность введённых данных и повторите попытку.", "Ошибка");  }  }  catch (Exception ex)  {  await config.Alert("Произошла ошибка с текстом исключения «" + ex.Message + "». Повторите попытку позднее или обратитесь к системному администратору.", "Ошибка " + ex.Source);  }  }  private object TryAuth()  {  var data\_base = new Database();  var data\_table = data\_base.Select("SELECT \* FROM `users` WHERE " +  "(`users`.`user\_login` = '" + LoginBox.Text + "' OR " +  "`users`.`user\_email` = '" + LoginBox.Text + "') AND BINARY " +  "`users`.`user\_password` = '" + PasswordBox.Password.ToString() + "'");  bool row\_count = Convert.ToBoolean(data\_table.Rows.Count);  string user\_email;  int id\_group;  if (row\_count)  {  user\_email = data\_table.Rows[0]["user\_email"].ToString();  id\_group = Convert.ToInt32(data\_table.Rows[0]["id\_group"]);  }  else  {  user\_email = null; id\_group = 0;  }  return new Data(user\_email, id\_group, row\_count);  }  private async void Activated()  {  var config = new config();  try  {  var AuthResult = TryAuth();  bool AuthStatus = Convert.ToBoolean(config.GetValue(AuthResult, "Status"));  if (AuthStatus)  {  string UserGroup = config.GetValue(AuthResult, "Group");  string UserEmail = config.GetValue(AuthResult, "Email");  await config.Alert("Вы находитесь в группе: " + UserGroup + Environment.NewLine  + "Ваша электронная почта: " + UserEmail, "Успешно");  }  else  {  await config.Alert("Проверьте правильность введённых данных", "Ошибка");  }  }  catch (Exception ex)  {  await config.Alert(ex.ToString(), "Ошибка");  }  }  private void LoginBox\_KeyDown(object sender, KeyRoutedEventArgs e)  {  if (LoginBox.Text != "" && PasswordBox.Password != "")  {  LoginButton.IsEnabled = true;  if (e.Key == Windows.System.VirtualKey.Enter)  {  Activated();  }  }  }  private void PasswordBox\_KeyDown(object sender, KeyRoutedEventArgs e)  {  if (LoginBox.Text != "" && PasswordBox.Password != "")  {  LoginButton.IsEnabled = true;  if (e.Key == Windows.System.VirtualKey.Enter)  {  Activated();  }  }  }  }  } |
| Config.cs | using MySql.Data.MySqlClient;  using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Security.Cryptography;  using Windows.UI.Popups;  namespace VD50\_1\_18\_OAIP\_MonakhovKM\_PR3.Assets  {  class config  {  // Переменная для создания нового экземпляра класса MySqlConnection и описание подключения к серверу базы данных  public MySqlConnection connection = new MySqlConnection("server=127.0.0.1; user=root; password=; database=movedb; port=3306");  // Объявление переменной для дальнейшей работы с кэшем данных  public Windows.Storage.ApplicationDataContainer localSettings = Windows.Storage.ApplicationData.Current.LocalSettings;  // Объявление переменной для хранения ключа сервера для шифрования передаваемых паролей  public string salt = "ТРЕБУЕТСЯ\_ЗАДАТЬ\_САМОСТОЯТЕЛЬНО";  // Создание хэша из строки  public string Hash(string input)  {  // Использование стандартных криптографических функций для имитации алгоритм хеширования MD5  byte[] asciiBytes = ASCIIEncoding.ASCII.GetBytes(input + salt);  byte[] hashedBytes = MD5CryptoServiceProvider.Create().ComputeHash(asciiBytes);  // Сохранение и преобразование результата работы в переменную  string hashedString =  BitConverter.ToString(hashedBytes).Replace("-", "").ToLower();  // Возвращение строки  return hashedString;  }  // Занесение действий в журнал  public void Log(int user, int action)  {  Database data\_base = new Database();  string sql = "INSERT INTO `logs` (`id\_user`, `id\_action`) VALUES ('" + user + "', '" + action + "')";  data\_base.Insert(sql);  }  // Генерация ключа безопасности  public string GenerateToken(int user)  {  Database data\_base = new Database();  // Объявление экземпляра класса Random для создания случайной строки(будущего ключа безопасности)  Random random = new Random();  const string chars = "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789";  string token = new string(Enumerable.Repeat(chars, 64).Select(s  => s[random.Next(s.Length)]).ToArray());  string sql = "INSERT INTO `tokens` (`id\_user`, `token`) VALUES ("  + user + ", '" + token + "')";  data\_base.Insert(sql);  // Вызов метода занесения действий в журнал с передачей идентификатора пользователя и первого типа действий в соответствии с базой данных(авторизация)  Log(user, 1);  //Возвращение ключа безопасности для его дальнейшего использования  return token;  }  // Метод для получения значений из объектов по ключу, возвращающий строковый тип данных  public string GetValue(object Object, string Key)  {  string Value = Object.GetType().GetProperty(Key).  GetValue(Object, null).ToString();  return Value;  }  public async System.Threading.Tasks.Task Alert(string Message, string Title)  {  var messageDialog = new MessageDialog(Message, Title);  await messageDialog.ShowAsync();  }  }  } |
| Database.cs | using MySql.Data.MySqlClient;  using System;  using System.Collections.Generic;  using System.Data;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace VD50\_1\_18\_OAIP\_MonakhovKM\_PR3.Assets  {  class Database : config  {  // Метод произведения выборки из базы данных (бд), принимающий всебя SQL-запрос  public DataTable Select(string sql)  {  // Открытие подключения к серверу бд  connection.Open();  // Отправка SQL-запроса на сервер бд с информацией о подключении  MySqlCommand command = new MySqlCommand(sql, connection);  // Выполнение команды на сервере  MySqlDataReader reader = command.ExecuteReader();  // Создание таблицы, которая будет принимать в себя полученные данные из выборки  DataTable dt = new DataTable();  // Загрузка в таблицу данных из выборки  dt.Load(reader);  // Закрытие подключения к серверу бд  connection.Close();  // Возвращение таблицы  return dt;  }  public void Insert(string sql)  {  connection.Open();  MySqlCommand command = new MySqlCommand(sql, connection);  // Выполнение команды на сервере без записи в переменную  command.ExecuteReader();  connection.Close();  }  }  } |
| MainPage.xaml | <Page  x:Class="VD50\_1\_18\_OAIP\_MonakhovKM\_PR3.MainPage"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  xmlns:local="using:VD50\_1\_18\_OAIP\_MonakhovKM\_PR3"  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"  mc:Ignorable="d"  Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">  <Grid>  <NavigationView x:Name="navView" Content="NavigationView"/>  </Grid>  </Page> |
| MainPage.cs | using System;  using System.Collections.Generic;  using System.IO;  using System.Linq;  using System.Runtime.InteropServices.WindowsRuntime;  using Windows.Foundation;  using Windows.Foundation.Collections;  using Windows.UI.Xaml;  using Windows.UI.Xaml.Controls;  using Windows.UI.Xaml.Controls.Primitives;  using Windows.UI.Xaml.Data;  using Windows.UI.Xaml.Input;  using Windows.UI.Xaml.Media;  using Windows.UI.Xaml.Navigation;  using Windows.UI.Popups;  using System.Threading.Tasks;  using Nito.AsyncEx;  using VD50\_1\_18\_OAIP\_MonakhovKM\_PR3.Assets;  // Документацию по шаблону элемента "Пустая страница" см. по адресу https://go.microsoft.com/fwlink/?LinkId=402352&clcid=0x419  namespace VD50\_1\_18\_OAIP\_MonakhovKM\_PR3  {  /// <summary>  /// Пустая страница, которую можно использовать саму по себе или для перехода внутри фрейма.  /// </summary>  public sealed partial class MainPage : Page  {  public MainPage()  {  this.InitializeComponent();  // Асинхронный вызов метода MainAsync AsyncContext.Run(() => MainAsync());  AsyncContext.Run(() => MainAsync());  }  // Приватный асинхронный метод MainAsync для определения  // состояния входа пользователя.Если он был авторизован ранее  // и имеет активный ключ безопасности, то будет незамедлительно  // отправлен к рабочему пространству  private async System.Threading.Tasks.Task MainAsync()  {  // Проверка состояния аутентификации по коду  User user = new User();  switch (user.CheckAuth())  {  // Код 0 возвращает необходимость повторно пройти  //процедуру аутентификации  case 0:  Frame.Navigate(typeof(Autentification));  break;  // Код 1 даёт возможность сразу приступать к работе,  // так как данные пользователя являются актуальными и  //продолжать работу с приложением безопасносо  case 1:  Frame.Navigate(typeof(PanelFrame));  break;  // Любой другой код говорит о необходимости проверить  //подключение к Интернету или же серверу базы данных при  //помощи цикла внутри данного окна. Если нет  //подключения, то метод CheckAuth() будет вечно  //возвращать код 2, что не входит в границы используемой  //конструкции  default:  // Объявление нового сообщения об ошибке  var messageDialog = new MessageDialog("Произошла ошибка при подключении к Интернету.Пожалуйста, повторите попытку.", "Ошибка");  // Добавление в сообщение об ошибке кнопок с  //вместе с передачей процедур по нажатию на них.От  //повторной загрузки экрана до выхода из  //приложения  messageDialog.Commands.Add(new  UICommand("Повторить",  (command) =>  {  Frame.Navigate(typeof(MainPage));  }));  messageDialog.Commands.Add(new  UICommand("Выход", (command) =>  {  Application.Current.Exit();  }));  await messageDialog.ShowAsync();  break;  }  }  }  } |
| PanelFrame.xaml | <Page  x:Class="VD50\_1\_18\_OAIP\_MonakhovKM\_PR3.PanelFrame"  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"  xmlns:local="using:VD50\_1\_18\_OAIP\_MonakhovKM\_PR3"  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"  mc:Ignorable="d"  Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">  <Grid>  <NavigationView x:Name="UserProfileBtn" Content="NavigationView"/>  </Grid>  </Page> |
| PanelFrame.cs | using System;  using System.Collections.Generic;  using System.IO;  using System.Linq;  using System.Runtime.InteropServices.WindowsRuntime;  using VD50\_1\_18\_OAIP\_MonakhovKM\_PR3.Assets;  using Windows.Foundation;  using Windows.Foundation.Collections;  using Windows.UI.Xaml;  using Windows.UI.Xaml.Controls;  using Windows.UI.Xaml.Controls.Primitives;  using Windows.UI.Xaml.Data;  using Windows.UI.Xaml.Input;  using Windows.UI.Xaml.Media;  using Windows.UI.Xaml.Navigation;  // Документацию по шаблону элемента "Пустая страница" см. по адресу https://go.microsoft.com/fwlink/?LinkId=234238  namespace VD50\_1\_18\_OAIP\_MonakhovKM\_PR3  {  /// <summary>  /// Пустая страница, которую можно использовать саму по себе или для перехода внутри фрейма.  /// </summary>  public sealed partial class PanelFrame : Page  {  config Config = new config();  public PanelFrame()  {  this.InitializeComponent();  // Объявление переменную для хранения инициалов  //пользователя, обрезание значения «Имя» до одного символа с  //точкой  string initials = " " +  Config.localSettings.Values["first\_name"].ToString().Substring(0, 1) + ".";  // Проверка на пустое значение поля «Отчество». В случае наличия, происходит обрезание значения до одного символа с точкой по аналогии с именем сотрудника  if (Config.localSettings.Values["mid\_name"] != "") initials  += " " +  Config.localSettings.Values["mid\_name"].ToString().Substring  (0, 1) + ".";  // Назначение содержания на элемент  UserProfileBtn.Content = Config.localSettings.Values["last\_name"] + initials;  }  }  } |
| Users.cs | using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using VD50\_1\_18\_OAIP\_MonakhovKM\_PR3.Assets;  namespace VD50\_1\_18\_OAIP\_MonakhovKM\_PR3  {  class User : config  {  Database data\_base = new Database();  private int ID\_User { get; set; }  private bool Status { get; set; }  private int ID\_Group { get; set; }  private string Login { get; set; }  private string Email { get; set; }  private string LastName { get; set; }  private string FirstName { get; set; }  private string MidName { get; set; }  public bool Authentication(string LoginBox, string PasswordBox)  {  var data\_table = data\_base.Select("SELECT `id\_user`, `id\_group`, `user\_login`, `user\_email`, `user\_password`, `last\_name`, `first\_name`, `mid\_name` FROM `users` JOIN `employee` WHERE(`users`.`user\_login` = '" + LoginBox + "'OR `users`.`user\_email` = '" + LoginBox + "') AND(BINARY `users`.`user\_password` = '" + Hash(PasswordBox) + "') AND(`users`.`id\_user` = `employee`.`id\_employee`)");  if (!Status)  {  ID\_User = Convert.ToInt32(data\_table.Rows[0]["id\_user"]);  ID\_Group = Convert.ToInt32(data\_table.Rows[0]["id\_group"]);  Login = data\_table.Rows[0]["user\_login"].ToString();  Email = data\_table.Rows[0]["user\_email"].ToString();  LastName = data\_table.Rows[0]["last\_name"].ToString();  FirstName = data\_table.Rows[0]["first\_name"].ToString();  MidName = data\_table.Rows[0]["mid\_name"].ToString(); CacheAuth();  }  return Status;  }  private void CacheAuth()  {  string Token = GenerateToken(ID\_User);  localSettings.Values["id\_user"] = ID\_User;  localSettings.Values["status"] = Status;  localSettings.Values["id\_group"] = ID\_Group;  localSettings.Values["user\_login"] = Login;  localSettings.Values["user\_email"] = Email;  localSettings.Values["last\_name"] = LastName;  localSettings.Values["first\_name"] = FirstName;  localSettings.Values["mid\_name"] = MidName;  localSettings.Values["token"] = Token;  }  public int CheckAuth()  {  try  {  Database data\_base = new Database();  string sql = "SELECT `tokens`.`id\_user`, `token`, `last\_name`, `first\_name`, `mid\_name` FROM `tokens` JOIN `users`, `employee` WHERE(`tokens`.`id\_user` ='" + localSettings.Values["id\_user"] + "') AND(`token` = '" + localSettings.Values["token"] + "')AND(`deactivated` = 0) AND(`tokens`.`id\_user` = `users`.`id\_user` AND `users`.`id\_employee` =`employee`.`id\_employee`)";  var data\_table = data\_base.Select(sql);  bool status = Convert.ToBoolean(data\_table.Rows.Count);  if (status && Convert.ToBoolean(localSettings.Values["status"]))  {  if (localSettings.Values["last\_name"] != data\_table.Rows[0]["last\_name"].ToString() || localSettings.Values["first\_name"] != data\_table.Rows[0]["first\_name"].ToString() || localSettings.Values["mid\_name"] != data\_table.Rows[0]["mid\_name"].ToString())  {  localSettings.Values["last\_name"] = data\_table.Rows[0]["last\_name"].ToString(); localSettings.Values["first\_name"] = data\_table.Rows[0]["first\_name"].ToString(); localSettings.Values["mid\_name"] = data\_table.Rows[0]["mid\_name"].ToString();  }  return 1;  }  else  {  return 0;  }  }  catch (Exception)  {  return 2;  }  }  }  } |

Вывод: я произвел модификацию базы данных на виртуальном веб-сервере с целью обеспечения безопасной аутентификации в программном обеспечении для дальнейшей работы внутри информационной системы в клиентском приложении на универсальной платформе Windows. Реализовал механизм для аутентификации в приложении на универсальной платформе Windows и научился делать реализацию проекта