Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №6**

По дисциплине «СПП»

за 6-й семестр

Выполнили:

студенты 3 курса

группы ПО-4

Воробей А.П.

Яковчик И.А.

Проверил:

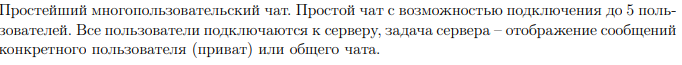
Монтик Н.С.

Брест, 2022

**Цель работы:** освоить приемы разработки клиент-серверного приложения.

**Вариант:** 22

**Задание:**



**Код программы:**

**MainWindow.cs:**

using System;  
using System.Linq;  
using System.Threading.Tasks;  
using System.Windows;  
using Microsoft.AspNetCore.SignalR.Client;  
  
namespace Client  
{  
 public partial class MainWindow : Window  
 {  
 private readonly HubConnection connection;  
  
 public MainWindow()  
 {  
 InitializeComponent();  
  
 PersonBtn.IsEnabled = false;  
  
 connection = new HubConnectionBuilder()  
 .WithUrl("http://localhost:5155/chat")  
 .WithAutomaticReconnect()  
 .Build();  
  
 Connect();  
   
 connection.Closed += async (error) =>  
 {  
 await Task.Delay(new Random().Next(0, 5) \* 1000);  
 await connection.StartAsync();  
 };  
 }  
  
 private async void Connect()  
 {  
 connection.On<string, string>("Send", (user, message) =>  
 {  
 Dispatcher.Invoke(() =>  
 {  
 var newMessage = $"{user}: {message}";  
 MessagesList.Items.Add(newMessage);  
 });  
 });  
   
 connection.On<string, string>("SendToPerson", (user, message) =>  
 {  
 Dispatcher.Invoke(() =>  
 {  
 var newMessage = $"{user}: {message}";  
 MessagesList.Items.Add(newMessage);  
 });  
 });  
  
 try  
 {  
 await connection.StartAsync();  
 MessagesList.Items.Add("Connection started");  
 }  
 catch (Exception) { }  
 }  
  
 private async void Send\_Click(object sender, RoutedEventArgs e)  
 {  
 try  
 {  
 await connection.InvokeAsync("Send", connection.ConnectionId, MessageBox.Text);  
 MessageBox.Clear();  
 }  
 catch (Exception ex) { }  
 }  
   
 private async void Send\_ClickPrivate(object sender, RoutedEventArgs e)  
 {  
 try  
 {  
 await connection.InvokeAsync("SendToPerson", UserBox.Text ,connection.ConnectionId, MessageBox.Text);  
 MessageBox.Clear();  
 }  
 catch (Exception ex) { }  
 }  
  
 private void MessagesList\_SelectionChanged(object sender, System.Windows.Controls.SelectionChangedEventArgs e)  
 {  
 UserBox.Clear();  
 var id = e.AddedItems[0].ToString().Split(":").First();  
 UserBox.Text = id;  
 PersonBtn.IsEnabled = true;  
 }  
 }  
}

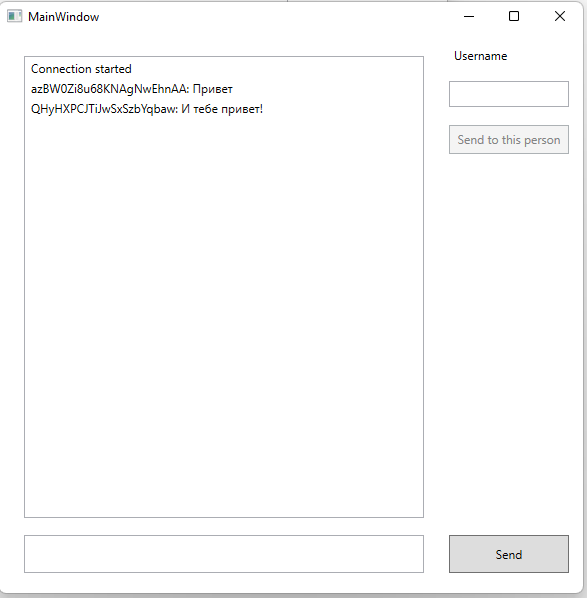
**ChatHub.cs:**

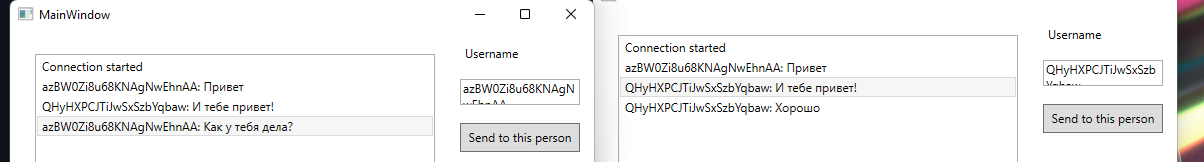
using Microsoft.AspNetCore.SignalR;  
  
namespace Server;  
  
public class ChatHub : Hub  
{  
 public async Task Send(string user, string message)  
 {  
 await Clients.All.SendAsync("Send", user, message);  
 }  
   
 public async Task SendToPerson(string connectionId,string user ,string message)  
 {  
 await Clients.Client(connectionId).SendAsync("SendToPerson", user, message);  
 }  
}

**Program.cs:**

using Server;  
  
var builder = WebApplication.CreateBuilder(args);  
  
*// Add services to the container.*builder.Services.AddRazorPages();  
builder.Services.AddSignalR(options =>  
{  
   
});  
  
var app = builder.Build();  
  
*// Configure the HTTP request pipeline.*if (!app.Environment.IsDevelopment())  
{  
 app.UseExceptionHandler("/Error");  
 *// The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.* app.UseHsts();  
}  
  
app.UseRouting();  
app.UseEndpoints(endpoints =>  
{  
 endpoints.MapHub<ChatHub>("/chat");  
});  
  
app.UseHttpsRedirection();  
app.UseStaticFiles();  
  
app.UseRouting();  
  
app.UseAuthorization();  
  
app.MapRazorPages();  
  
app.Run();

**Результаты работы:**

****

****

**Выводы:** в ходе выполнения лабораторной работы были освоены приемы разработки клиен-серверного приложения.