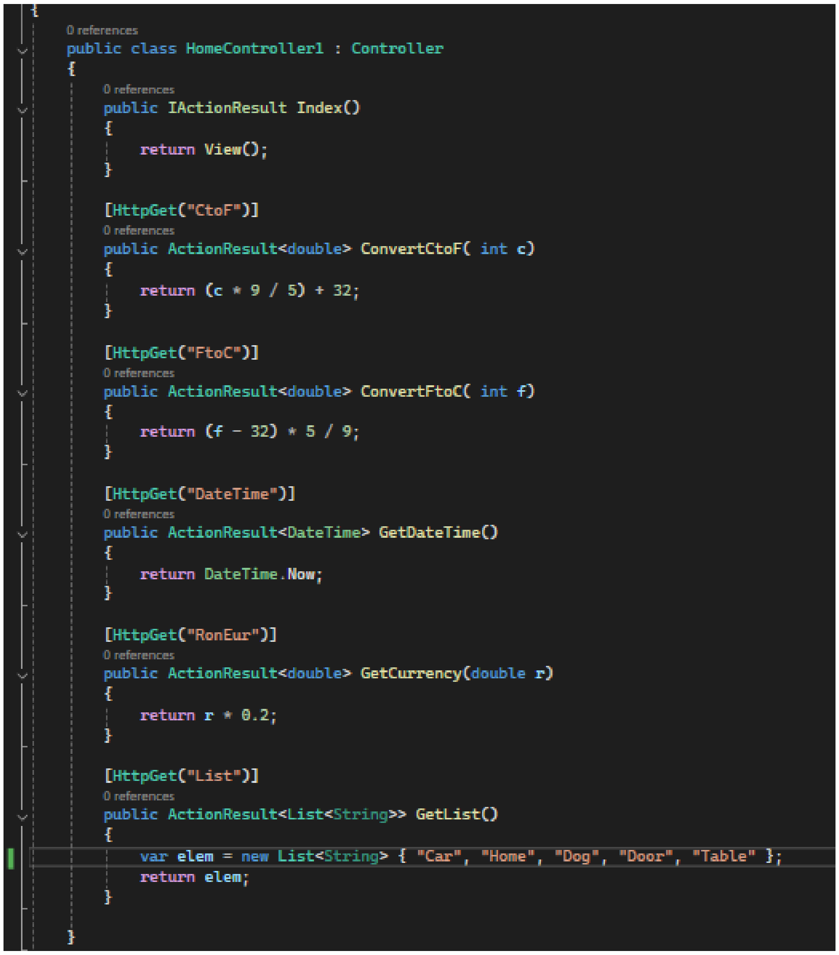
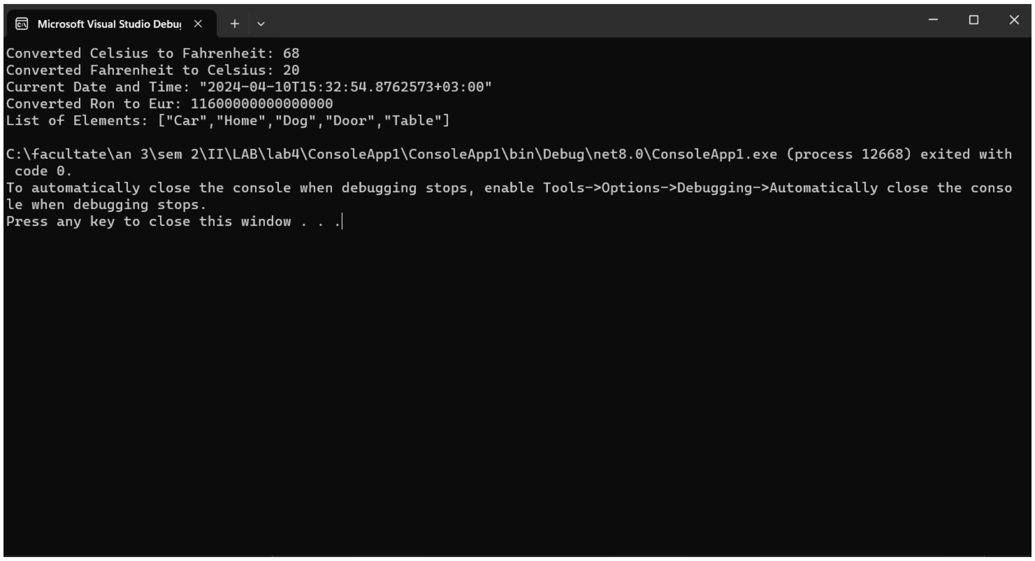
All code available on GitHub at:

Server-side controller:

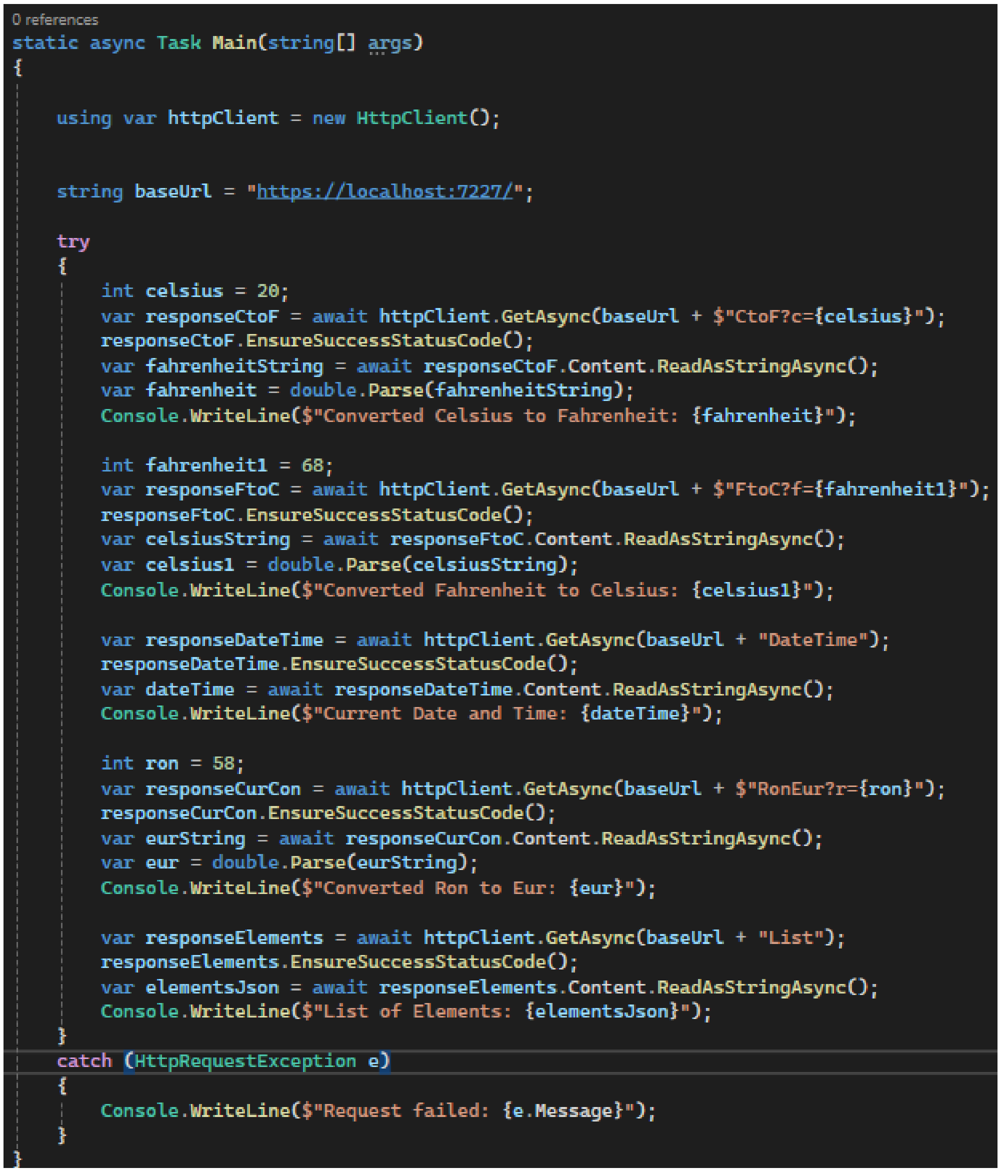


```
{
    0 references
    public class HomeController1 : Controller
    {
        0 references
        public IActionResult Index()
        {
            return View();
        }

        [HttpGet("CtoF")]
        0 references
        public ActionResult<double> ConvertCtoF( int c)
        {
            return (c * 9 / 5) + 32;
        }

        [HttpGet("FtoC")]
        0 references
        public ActionResult<double> ConvertFtoC( int f)
        {
            return (f - 32) * 5 / 9;
        }

        [HttpGet("DateTime")]
        0 references
        public ActionResult<DateTime> GetDateTime()
        {
            return DateTime.Now;
        }

        [HttpGet("RonEur")]
        0 references
        public ActionResult<double> GetCurrency(double r)
        {
            return r * 0.2;
        }

        [HttpGet("List")]
        0 references
        public ActionResult<List<String>> GetList()
        {
            var elem = new List<String> { "Car", "Home", "Dog", "Door", "Table" };
            return elem;
        }

    }
```

Console:

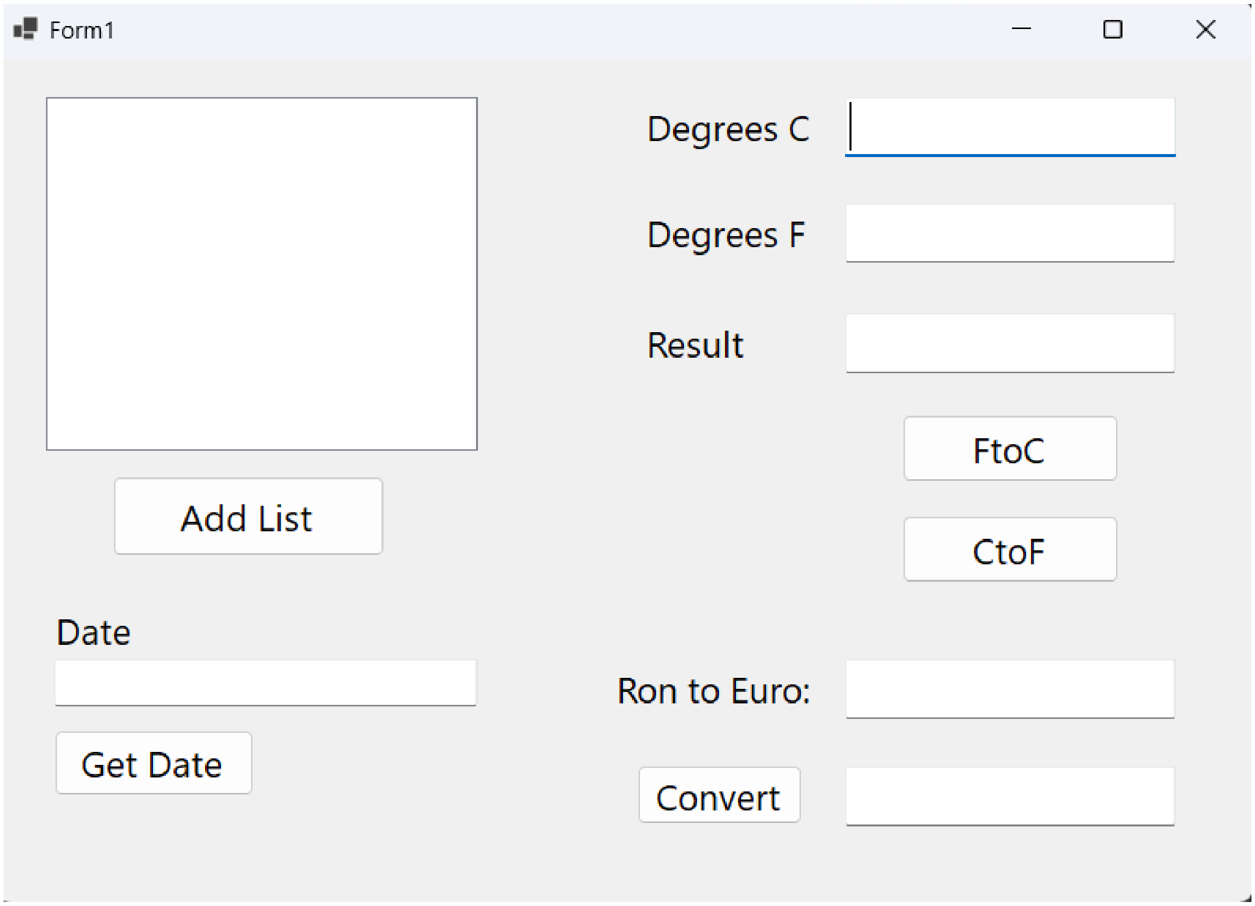


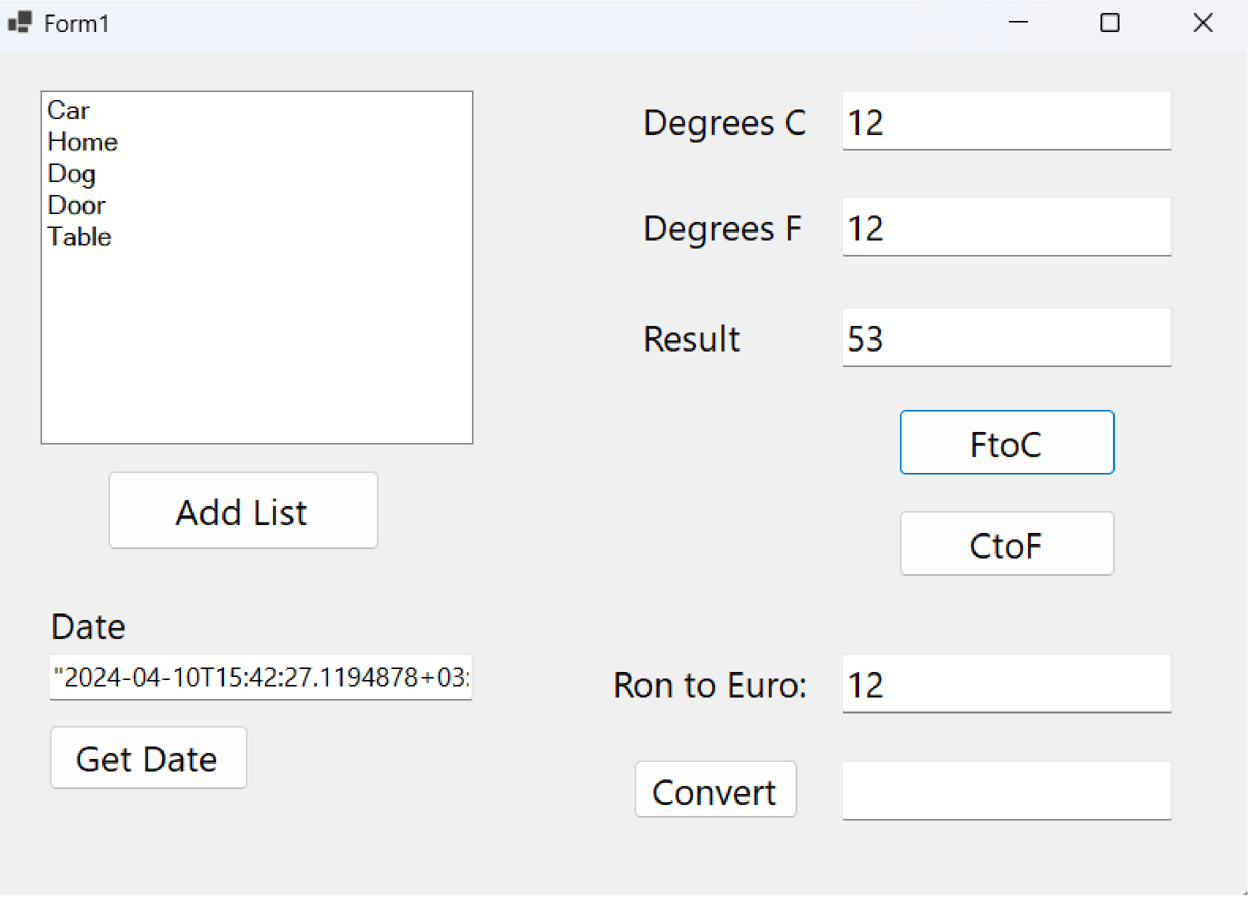
```
Converted Celsius to Fahrenheit: 68
Converted Fahrenheit to Celsius: 20
Current Date and Time: "2024-04-10T15:32:54.8762573+03:00"
Converted Ron to Eur: 116000000000000000
List of Elements: ["Car","Home","Dog","Door","Table"]

C:\facultate\an 3\sem 2\II\LAB\lab4\ConsoleApp1\ConsoleApp1\bin\Debug\net8.0\ConsoleApp1.exe (process 12668) exited with
 code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```



```csharp
static async Task Main(string[] args)
{

    using var httpClient = new HttpClient();


    string baseUrl = "https://localhost:7227/";

    try
    {
        int celsius = 20;
        var responseCtoF = await httpClient.GetAsync(baseUrl + $"CtoF?c={celsius}");
        responseCtoF.EnsureSuccessStatusCode();
        var fahrenheitString = await responseCtoF.Content.ReadAsStringAsync();
        var fahrenheit = double.Parse(fahrenheitString);
        Console.WriteLine($"Converted Celsius to Fahrenheit: {fahrenheit}");

        int fahrenheit1 = 68;
        var responseFtoC = await httpClient.GetAsync(baseUrl + $"FtoC?f={fahrenheit1}");
        responseFtoC.EnsureSuccessStatusCode();
        var celsiusString = await responseFtoC.Content.ReadAsStringAsync();
        var celsius1 = double.Parse(celsiusString);
        Console.WriteLine($"Converted Fahrenheit to Celsius: {celsius1}");

        var responseDateTime = await httpClient.GetAsync(baseUrl + "DateTime");
        responseDateTime.EnsureSuccessStatusCode();
        var dateTime = await responseDateTime.Content.ReadAsStringAsync();
        Console.WriteLine($"Current Date and Time: {dateTime}");

        int ron = 58;
        var responseCurCon = await httpClient.GetAsync(baseUrl + $"RonEur?r={ron}");
        responseCurCon.EnsureSuccessStatusCode();
        var eurString = await responseCurCon.Content.ReadAsStringAsync();
        var eur = double.Parse(eurString);
        Console.WriteLine($"Converted Ron to Eur: {eur}");

        var responseElements = await httpClient.GetAsync(baseUrl + "List");
        responseElements.EnsureSuccessStatusCode();
        var elementsJson = await responseElements.Content.ReadAsStringAsync();
        Console.WriteLine($"List of Elements: {elementsJson}");
    }
    catch (HttpRequestException e)
    {
        Console.WriteLine($"Request failed: {e.Message}");
    }
}
```

Windows Form:


Form1
Degrees C
Degrees F
Result
FtoC
CtoF
Add List
Date
Get Date
Ron to Euro:
Convert



Car
Home
Dog
Door
Table

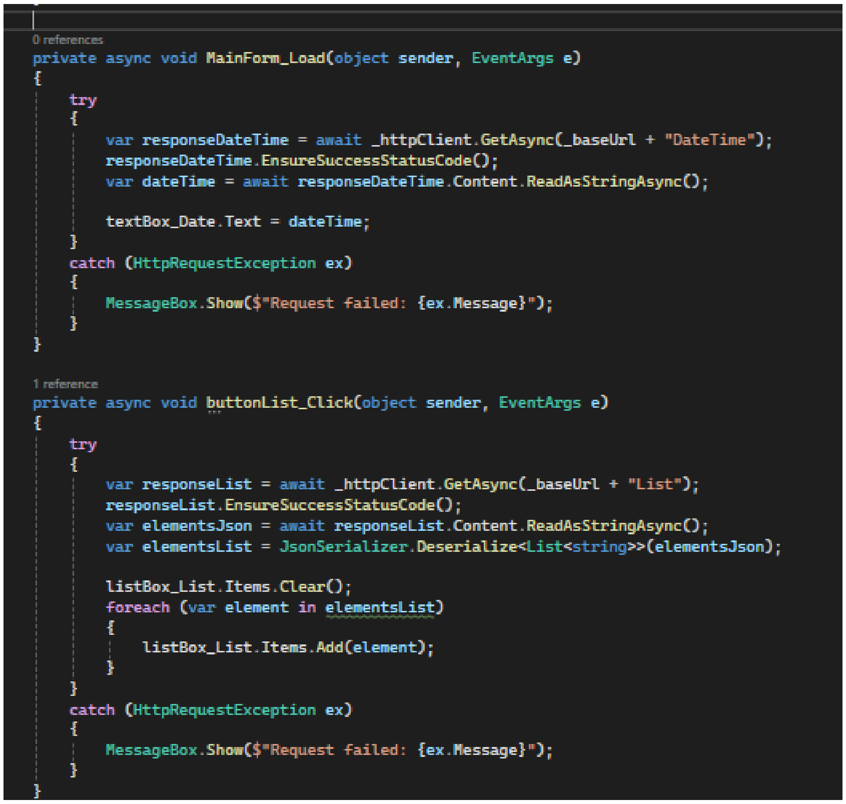
Date: "2024-04-10T15:42:27.1194878+03:

Degrees C: 12

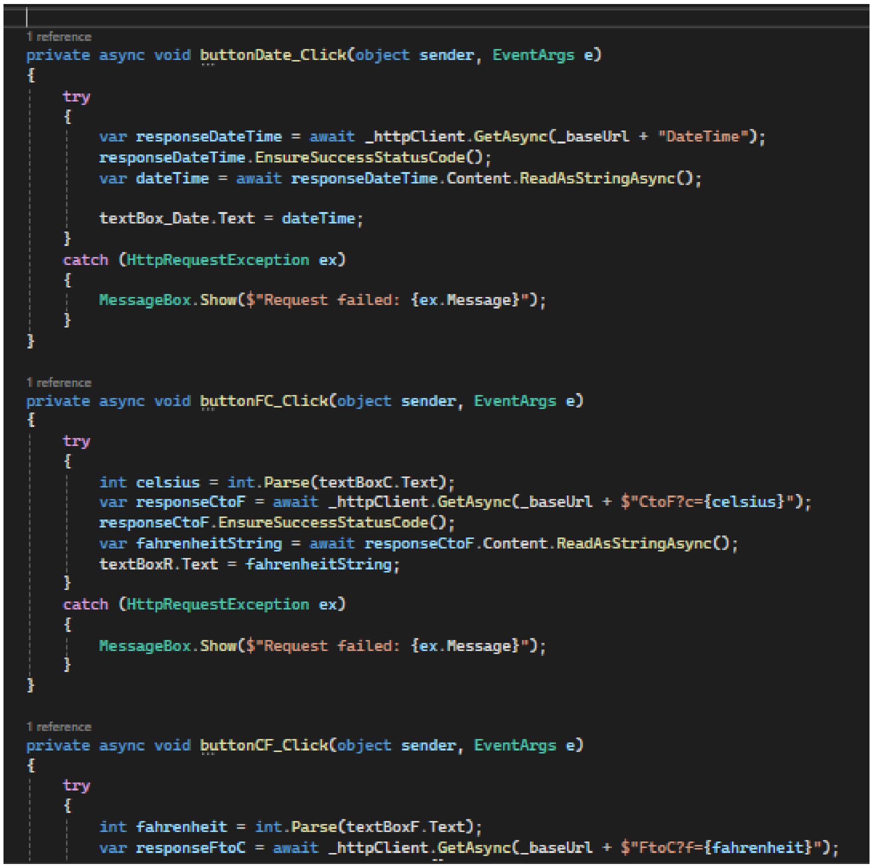
Degrees F: 12

Result: 53

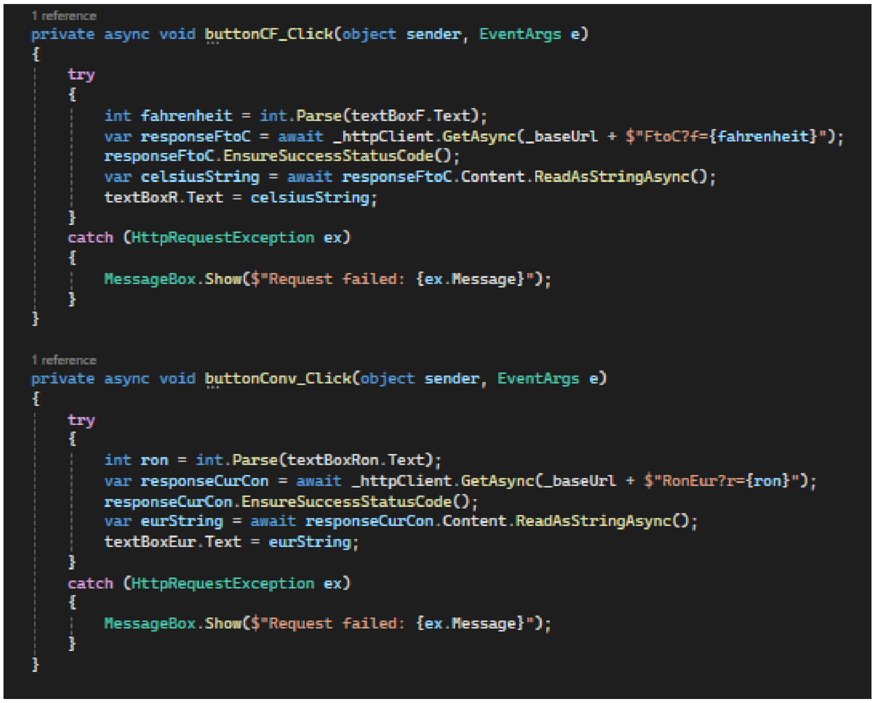
Ron to Euro: 12



```
0 references
private async void MainForm_Load(object sender, EventArgs e)
{
    try
    {
        var responseDateTime = await _httpClient.GetAsync(_baseUrl + "DateTime");
        responseDateTime.EnsureSuccessStatusCode();
        var dateTime = await responseDateTime.Content.ReadAsStringAsync();

        textBox_Date.Text = dateTime;
    }
    catch (HttpRequestException ex)
    {
        MessageBox.Show($"Request failed: {ex.Message}");
    }
}

1 reference
private async void buttonList_Click(object sender, EventArgs e)
{
    try
    {
        var responseList = await _httpClient.GetAsync(_baseUrl + "List");
        responseList.EnsureSuccessStatusCode();
        var elementsJson = await responseList.Content.ReadAsStringAsync();
        var elementsList = JsonSerializer.Deserialize<List<string>>(elementsJson);

        listBox_List.Items.Clear();
        foreach (var element in elementsList)
        {
            listBox_List.Items.Add(element);
        }
    }
    catch (HttpRequestException ex)
    {
        MessageBox.Show($"Request failed: {ex.Message}");
    }
}
```



```
1 reference
private async void buttonDate_Click(object sender, EventArgs e)
{
    try
    {
        var responseDateTime = await _httpClient.GetAsync(_baseUrl + "DateTime");
        responseDateTime.EnsureSuccessStatusCode();
        var dateTime = await responseDateTime.Content.ReadAsStringAsync();

        textBox_Date.Text = dateTime;
    }
    catch (HttpRequestException ex)
    {
        MessageBox.Show($"Request failed: {ex.Message}");
    }
}

1 reference
private async void buttonFC_Click(object sender, EventArgs e)
{
    try
    {
        int celsius = int.Parse(textBoxC.Text);
        var responseCtoF = await _httpClient.GetAsync(_baseUrl + $"CtoF?c={celsius}");
        responseCtoF.EnsureSuccessStatusCode();
        var fahrenheitString = await responseCtoF.Content.ReadAsStringAsync();
        textBoxR.Text = fahrenheitString;
    }
    catch (HttpRequestException ex)
    {
        MessageBox.Show($"Request failed: {ex.Message}");
    }
}

1 reference
private async void buttonCF_Click(object sender, EventArgs e)
{
    try
    {
        int fahrenheit = int.Parse(textBoxF.Text);
        var responseFtoC = await _httpClient.GetAsync(_baseUrl + $"FtoC?f={fahrenheit}");
```



```
1 reference
private async void buttonCF_Click(object sender, EventArgs e)
{
    try
    {
        int fahrenheit = int.Parse(textBoxF.Text);
        var responseFtoC = await _httpClient.GetAsync(_baseUrl + $"FtoC?f={fahrenheit}");
        responseFtoC.EnsureSuccessStatusCode();
        var celsiusString = await responseFtoC.Content.ReadAsStringAsync();
        textBoxR.Text = celsiusString;
    }
    catch (HttpRequestException ex)
    {
        MessageBox.Show($"Request failed: {ex.Message}");
    }
}

1 reference
private async void buttonConv_Click(object sender, EventArgs e)
{
    try
    {
        int ron = int.Parse(textBoxRon.Text);
        var responseCurCon = await _httpClient.GetAsync(_baseUrl + $"RonEur?r={ron}");
        responseCurCon.EnsureSuccessStatusCode();
        var eurString = await responseCurCon.Content.ReadAsStringAsync();
        textBoxEur.Text = eurString;
    }
    catch (HttpRequestException ex)
    {
        MessageBox.Show($"Request failed: {ex.Message}");
    }
}
```