UNIVERSITETI POLITEKNIK I TIRANES

FAKULTETI I TEKNOLOGJISE SE INFORMACIONIT

Laborator : 10

Dega : Inxhinieri Software

Lenda : Programim I Avancuar

Grupi : Master Profesional

Punoi : Redon Agushi

Pranoi : Dr.Hakik Paci

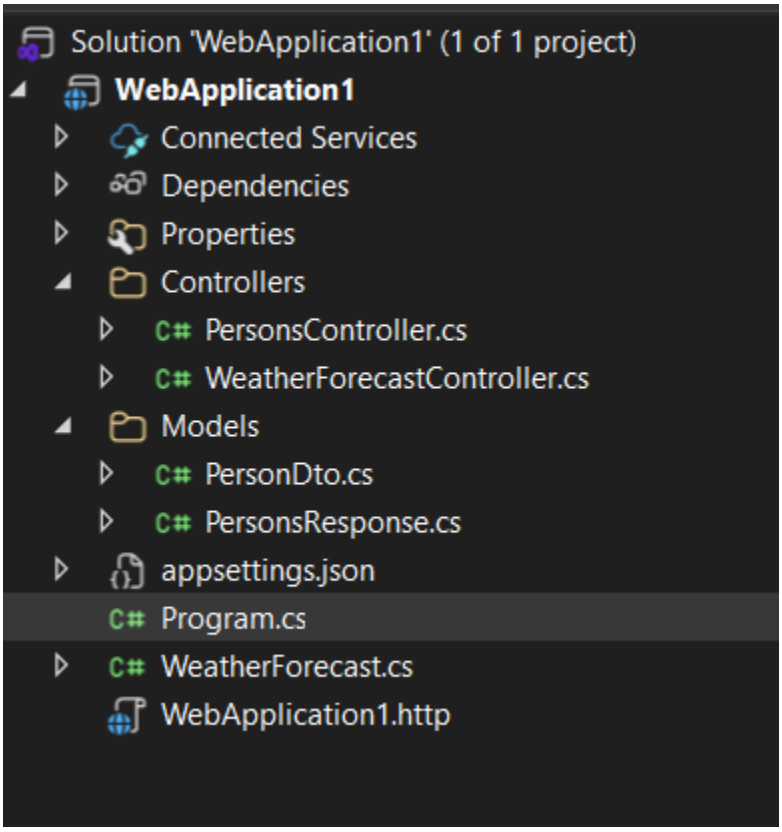
Detyra e Kursit 10:

10- Ndërtoni një WS për aksesimin e të dhënave në një database SQL Server dhe rezultati të jetë në format XML. (Realizimi deri me dt 23.01.2026

Zhvillimi:

Ky projekt realizon një Web Service në gjuhën C# duke përdorur ASP.NET Core Web API.
Web Service lidhet me një database SQL Server dhe lexon të dhëna nga tabela “Personat”.
Të dhënat përfaqësohen me anë të një modeli (DTO) dhe kthehen tek klienti në format XML duke përdorur XML Serializer Formatter.
Shërbimi aksesohet përmes një endpoint-i HTTP GET dhe është testuar me Postman, ku përgjigja kthehet me sukses në format XML.

Kodi I programit:



Kam shtuar 3 klasa :PersonsController.cs,PersonDto.cs,PersonsResponse.cs.

PersonsController.cs

- Është API Controller (Web Service).

- Ka 2 endpoint-e:

`GET /api/persons` → lidhet me SQL Server, lexon `PersonId, Emri, Mbiemri` nga tabela Personat dhe i kthen si XML.

`GET /api/persons/search?emri=...&mbiemri=...` → bën kërkim me `LIKE` (pjesë e emrit/mbiemrit) dhe kthen rezultatet si XML.

- Përdor `IConfiguration` për të marrë connection string nga `appsettings.json`.

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.Data.SqlClient;

[ApiController]
[Route("api/[controller]")]
public class PersonsController : ControllerBase
{
   private readonly IConfiguration _config;

   public PersonsController(IConfiguration config)
   {
      _config = config;
   }

   [HttpGet]
   [Produces("application/xml")]
   public async Task<ActionResult<PersonsResponse>> GetAll()
   {
      var cs = _config.GetConnectionString("Db");
      var result = new PersonsResponse();

      await using var con = new SqlConnection(cs);
      await con.OpenAsync();

      // Ndrysho emrat e kolonave/tabelës sipas DB tënde
      var sql = "SELECT PersonId, Emri, Mbiemri FROM Personat";

      await using var cmd = new SqlCommand(sql, con);
      await using var r = await cmd.ExecuteReaderAsync();

      while (await r.ReadAsync())
      {
         result.Items.Add(new PersonDto
         {
            PersonId = r.GetInt32(0),
            Emri = r.IsDBNull(1) ? null : r.GetString(1),
            Mbiemri = r.IsDBNull(2) ? null : r.GetString(2)
         });
      }
```

```
        // Kthen XML automatikisht (sepse kemi XmlSerializerFormatters)
        return Ok(result);
    }

    // opsionale: kërkim me query ?emri=...&mbiemri=...
    [HttpGet("search")]
    [Produces("application/xml")]
    public async Task<ActionResult<PersonsResponse>> Search([FromQuery] string? emri, [FromQuery] string? mbiemri)
    {
        var cs = _config.GetConnectionString("Db");
        var result = new PersonsResponse();

        await using var con = new SqlConnection(cs);
        await con.OpenAsync();

        var sql = @"
SELECT PersonId, Emri, Mbiemri
FROM Personat
WHERE (@Emri IS NULL OR Emri LIKE @EmriLike)
  AND (@Mbiemri IS NULL OR Mbiemri LIKE @MbiemriLike)";

        await using var cmd = new SqlCommand(sql, con);
        cmd.Parameters.AddWithValue("@Emri", (object?)emri ?? DBNull.Value);
        cmd.Parameters.AddWithValue("@Mbiemri", (object?)mbiemri ?? DBNull.Value);
        cmd.Parameters.AddWithValue("@EmriLike", $"%{emri}%");
        cmd.Parameters.AddWithValue("@MbiemriLike", $"%{mbiemri}%");

        await using var r = await cmd.ExecuteReaderAsync();
        while (await r.ReadAsync())
        {
            result.Items.Add(new PersonDto
            {
                PersonId = r.GetInt32(0),
                Emri = r.IsDBNull(1) ? null : r.GetString(1),
                Mbiemri = r.IsDBNull(2) ? null : r.GetString(2)
            });
        }

        return Ok(result);
    }
}
```

PersonDto.cs

- Është DTO / Model që përfaqëson 1 person.
- Ka fushat: `PersonId`, `Emri`, `Mbiemri`.
- `[XmlRoot("Person")]` bën që në XML çdo objekt të dalë si:

```
using System.Xml.Serialization;
```

```
[XmlRoot("Person")]
public class PersonDto
{
    public int PersonId { get; set; }
    public string? Emri { get; set; }
    public string? Mbiemri { get; set; }
}
```

PersonsResponse.cs.

- Është wrapper për listën e personave (pra përgjigjja e endpoint-it).
- `Items` është lista `List<PersonDto>`.
- `[XmlRoot("Persons")]` bën që e gjithë përgjigjja të jetë:
- `<Persons>...</Persons>`
- `[XmlElement("Person")]` bën që çdo element i listës të jetë tag `<Person>` (jo `<Items>` ose emra të çuditshëm).

```
using System.Xml.Serialization;

[XmlRoot("Persons")]
public class PersonsResponse
{
    [XmlElement("Person")]
    public List<PersonDto> Items { get; set; } = new();
}
```

Rezultati XML pasi e ekzekutojme programin.