PE_PRN212_SU25_TrialTest - Note
SPRING 2025
Subject: PRN212
Duration: 85 minutes

INSTRUCTIONS

Please read the instructions carefully before doing the questions.

- You are **NOT allowed** to use any other materials. You are **NOT allowed** to use any device to share data with others.
- You must use IDE as **Visual Studio 2019 or later, MSSQL Server 2016 or later database** for your development tools.

IMPORTANT – Before you start doing your solution, MUST do the following steps:

1. To do your program, you must use **Windows Presentation Foundation (WPF), apply 3-Layer architecture**, there are at least 2 Projects for the Solution. *The database connection string must get from appsettings.json file.*

   In the case your program connects directly to database from WPF Windows/Pages or you hardcode connection string, you will get 0 point.
2. Create Solution in Visual Studio 2019/2022 named **PE_PRN212_SU25_TrialTest_StudentName**. Inside the Solution, Project WPF named: **ResearchProjectManagement_StudentCode.**
3. Create your MS SQL database named **SU25ResearchDB** by running code in script **SU25ResearchDB.sql.**
4. Set the default user interface for your project as **Login** window/page.
5. ***If there are syntax errors or compilation errors in your PE program, you will not pass the PE requirements, the point will be 0.***
6. ***Your work will be considered invalid (0 point) if your code inserts stuff that is unrelated to the test.***

REFERENCES ***(This session is just for reference; students can use the other approach to do the Practical Exam.)***

Working with DB connection string from JSON file.

a. In the Presentation layer (WPF Project), you create *appsettings.json* and add ConnectionStrings same as the bellow to config the connection string to SQL Server Database.

```
{
        "ConnectionStrings": {
            "DefaultConnection": "server=(local); database=SU25ResearchDB; uid=sa; pwd=1234567890;
TrustServerCertificate=True;"
        }
}
```

You can change **server**, **uid** and **pwd** to suitable information with your local machine.

b. Set property "Copy to Output Directory" of *appsettings.json* file to "Copy always"

c. Using Manage Nuget packages to install packages

Package using for .NET:

	Microsoft.EntityFrameworkCore.SqlServer version	*Microsoft.Extensions.Configuration, Microsoft.Extensions.Configuration.Json version*
.NET 8	*8.0.12*	*8.0.1*
.NET 9	*9.0.2*	*9.0.2*

- Install package using Tools NuGet Package Manager Package Manager Console

Install-Package Microsoft.EntityFrameworkCore.SqlServer -Version 9.0.2
Install-Package Microsoft.EntityFrameworkCore.Design -Version 9.0.2
Install-Package Microsoft. EntityFrameworkCore.Tools -Version 9.0.2
Install-Package Microsoft.Extensions.Configuration -Version 9.0.2
Install-Package Microsoft.Extensions.Configuration.Json -Version 9.0.2

- Install package using CLI or Power Shell

dotnet add package Microsoft.EntityFrameworkCore.SqlServer --version 9.0.2
dotnet add package Microsoft.EntityFrameworkCore.Design --version 9.0.2
dotnet add package Microsoft.EntityFrameworkCore.Tools --version 9.0.2
dotnet add package Microsoft.Extensions.Configuration --version 9.0.2
dotnet add package Microsoft.Extensions.Configuration.Json --version 9.0.2

d. Using *ConfigurationBuilder* to init Configuration object for reading *appsettings.json* file same as this code:

```
private string GetConnectionString()
```

```
{
        IConfiguration config = new ConfigurationBuilder()
                .SetBasePath(Appcontext.BaseDirectory)
                .AddJsonFile("appsettings.json",true,true)
                .Build();
        var strConn = config["ConnectionStrings:DefaultConnection"];

        return strConn;
}
```

e. After that, durring development, student can bypass the ConnectionString (which read from *appsettings.json*) to Data access layer by constructor or ***OnConfiguring*** method

```
protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
{
        optionsBuilder.UseSqlServer(GetConnectionString());
}
```

Entity Framework Core
- *Install dotnet-ef for CLI*
dotnet tool install --global dotnet-ef --version 9.0.2

- *Use Entity Framework Core to generate Object Model from existing database – CLI*
dotnet ef dbcontext scaffold
"Server=(local);uid=**sa**;pwd=**1234567890**;database=**SU25ResearchDB**;TrustServerCertificate=True;"
Microsoft.EntityFrameworkCore.SqlServer --output-dir ./

- *Generate database from domain classes – CLI.*
dotnet ef migrations add "InitialDB"
dotnet ef database update

Entity Framework Core
- *Use Entity Framework Core to generate Object Model from existing database – Package Manager Console*
Scaffold-DbContext
"Server=(local);uid=**sa**;pwd=**1234567890**;database=**SU25ResearchDB**;TrustServerCertificate=True;"
Microsoft.EntityFrameworkCore.SqlServer -OutputDir ./

- *Generate database from domain classes – Package Manager Console*
Add-Migration "InitialDB"

```
Update-Database -verbose
```