📋 Project plan: People Data Access

Description	People Data Access Master Documentation
Status	Completed 100% in Production
Team	Role: Developer Wouter Lombard Role: Developer
Related	•

Timeline

Timeline			
Title	**Dates**	**Assigned to**	**Description**
⍰	Jan 8, 2020 - Jan 10, 2020	Wouter L, Developer I	Need to finish up with last code function. Adding Insert People.

Details

We are at the final stages of code design. @Wouter L

https://www.dropbox.com/s/rtikrbttqhda7e5/Project%20People%20Data%20Access.docx?dl=0

```
Project People Data


//Create and custom to each Practice build. and extract this p
art to add to each practice app's custom documentation.
//exc: Groceries list and price
//exc: stock list.
//exc: character List.
All DATA ACCESS SETUP
After creating a new project this is the next step.
```

TakeNote:The DataAccessLibrary specific can fork out and be used with an

other .NET Framework development .ie, WinForms, MVC, Xamarin, etc.

(IMPORTANT PROCESS EVERY TIME WHEN DOING DATA ACCESS)

Right Click on **Solution at main top** of solution explorer

Add **New Project (** Don't want to tie appnameUI directly to Data Access) **very important**

Search for class lib, choose class library type to use .net standard framework(works across more platforms than .net core does. .net standard is best option, if you can. very important).

Name it **DataAccessLibrary**

Delete Class1 always

Create class in DataAccessLibrary name: SqlDataAccess.cs

inside class add public Infront of class SqlDataAccess

in DataAccessLibrary, right click on Dependensies – Manage Nuget Packages

search and installs Dapper

inside SqlDataAccess.cs 1. create constructor (ctor)

```
public class SqlDataAccess
{
            private readonly IConfiguration _config; //comes from 2
            public string ConnectionStringName { get; set; } = "Default"; //  3
            public SqlDataAccess(IConfiguration config) 1 and 2
            {
                        _config = config; //comes from 2
}
```

With IConfiguration –

```
Add using ctrl. install package Microsoft.Extensions.Configura
tion.Abstarctions
after adding config to IConfiguration –
ctrl. Create and initialize field _config


still inside public class SqlDataAccess
public async Task<List<T>> LoadData<T, U>(string sql, U parame
ters)
 (ctrl. using System.Treading.Tasks;)
{
string connectionString = _config.GetConnectionString(Connecti
onStringName);
            using (IDbConnection connection = new SqlConnectio
n(connectionString))
ctrl. using System.Data;
also add after parameter is added. ctrl. install package Syste
m.Data.SqlClient
            {
                        var data = await connection.QueryAsync
<T>(sql, parameters);
                        (ctrl. using Dapper;)
                        return data.ToList();
                        (ctrl. using System.Linq;)
}
         }
         public async Task SaveData<T>(string sql, T parameter
s)
         {
         string connectionString = _config.GetConnectionString
(ConnectionStringName);
         using (IDbConnection connection = new SqlConnection(c
onnectionString)
```

```
         {
         await connection.ExecuteAsync(sql, parameters);
}
}
NEXT STEP
asking for IConfiguration - we going to put it in a dependenci
es injection

 right click top on class SqlDataAccess quick actions and refa
ctoring
for adding, at bottom of list Extract Interface…
( mandatory) adds ISqlDataAccess.cs to DataAccessLibrary
Right click DataAccessLibrary - add class - AnyNameData.cs
public class PeopleData
{
private readonly ISqlDataAccess _db;
(create ctor)
public PeopleData(ISqlDataAccess db)
ctrl. Create and initialize field _db
{
         _db = db;

}
Create Models folder in DataAccessLibrary
Create class PersonModel inside Models Folder
public class PersonModel
{
public string FirstName { get; set; }
public string LastName { get; set; }
public string EmailAddress { get; set; }
}

after PersonModel was created.
add inside public class PeopleData
the additional code. after ISqlDataAccess db
{
}
CONNECT AND OR CREATE DATABASE EX: dbo.People Table
public Task<List<PersonModel>> GetPeople()
ctrl. using System.Threading.Tasks;
ctrl. using DataAccessLibrary.Models;
{
string sql = “select * from dbo.People”;
return _db.LoadData<PersonModel, dynamic>(sql, new { });
}
public Task InsertPerson(PersonModel person)
{
string sql = @”insert into dbo.People (FirstName, LastName, EmailAddress)
values (@FirstName, @LastName, @EmailAddress);”;
         return _db.SaveData(sql, person);
}
RIGHT CLICK ON public class PeopleData
quick actions and refactoring – bottom of list – extract Interface
}
GO TO:  startup.cs
         in public void ConfigureServices(IServiceCollection services)
         Add services.AddTransient<ISqlDataAccess, SqlDataAccess>();
```

```
Ctrl. Add reference to ‘DataAccessLibrary’ using DataAccessLib
rary; nr 1
Add services.AddTransient<IPeopleData, PeopleData>();
//Transient means going to create an instance every time we as
k for one.
//Singleton creates one instance for the entire application.
GO TO: appsettings.json in BlazorUI to add connectionString
// GO TO: Database file DatabasenameDB right click go to prope
rties
// double click on connectionString and copy

under “AllowedHosts”: “*”,
“ConnectionStrings”: {
 “Default”: “ paste database connectionString in here”
}  //remember to add password

CREATE A PAGE TO INSERT PEOPLE
         Under BlazerUI go inside Pages and create another fol
der ConfigDataPages
         //PLEASE NOTE: Razor Pages is .cshtml , with a PageMo
del behind it.
         //Blazor pages is  Razor component.razor.
Right click on folder ConfigDataPages and add new item and cho
ose Razor component
it has a .razor file extension give name Filename.razor (Peopl
e)

GO TO: People.razor

         Add @page “/ConfigDataPages/People” // 1 entry
         @using DataAccessLibrary            // 2 entry
         @using DataAccessLibrary.Models    // 3 entry
         @inject IPeopleData _db            // 4 entry // give
s access to dataAccess


         <h1>People Data Page</h1>


         <h4>Current People</h4>
         @if (people is null)
         // entry 7
         {
                     <p><em>Loading…</em></p>
         }
else
       //entry 8
{
            <table class=”table table-striped”>
            <thread> //* dont add thread if spacing of columns
are needed
                        <tr>
                        <th>First Name</th>
                        <th>Last Name</th>
                        <th>Email Address</th>
</tr>
</thread> //*



<tbody>
@foreach (var person in people) //loop through all people in @
code private list
{
<tr> // each gets a row
<td>@person.FirstName</td>
<td>@person.LastName</td>
<td>@person.EmailAddress</td>
</tr>
}
</tbody>
            </table>
}
         @code
{
     private List<PersonModel> people;            // 5 entry
     protected override async Task OnInitializedAsync() //entr
y 6
              {
                     people = await _db.GetPeople();
     }
}
VERY IMPORTANT
GO TO: Shared folder NavMenu.razor
         in @NavMenuCssClass
 // THIS LINKS TO PEOPLE.RAZOR
         add <li class=”nav-intem px-3”>
<NavLink class=”nav-link” href=”ConfigDataPages/People”>
         <span class=”oi oi -people” aria-hidden=”true”></span
> People //oi is opsouce Lib
         </NavLink>
//play around with settings and oi icons
              </li>
</ul>
CONTINUE AT 58:58
GO TO: AplicationNameUI
```

```
          right click and create new folder
          Models
          Create new class called DisplayPersonModel.cs
          public class DisplayPersonModel
          {
                   [Required] // ctrl. using System.ComponentMo
del.DataAnnotations;
                   [StringLength(15, ErrorMessage = “First Name
is too long.”)]
                   [Minlength(5, ErrorMessage = “First Name is
to short”)]
                    public string FirstName { get; set; }

                      [Required]
                      [StringLength(15, ErrorMessage = “Last Na
me is too long.”)]
                      [MinLength(5, ErrorMessage = “Last Name i
s too short.”)]
                      public string LastName { get; set; }

                      [Required]
                      [EmailAddress]
                      public string EmailAddress { get; set; }
          }

GO TO: People.razor
add using ApplicationNameUI.Models
inside People.razor
@code{
entry: private DisplayPersonModel newPerson = new DisplayPerso
nModel(); // instantiate newPerson right away
```

```
}
Entry under <h1>People Data</h1>
<h4>Insert New Person Data</h4>
//CONTINUE 1:05:30
 <EditForm Model="@newPerson" OnValidSubmit="@InsertPerson">
         <DataAnnotationsValidator />   // This is the code in
side DisplayPersonModel in [Required] etc.
         <ValidationSummary />


         <InputText id="firstName" @bind-Value="newPerson.Firs
tName" />
         <InputText id="lastName" @bind-Value="newPerson.LastN
ame" />
         <InputText id="emailAdddress" @bind-Value="newPerson.
EmailAddress" />


         <button type="submit" class="btn btn-primary">Submit
</button>
</EditForm>


Inside @code create a method
private async Task InsertPerson() //change from private viod I
nsertPerson()
{
//What does form do? This is what form do?


PersonModel p = new PersonModel
{
         FirstName = newPerson.FirstName,
         LastName = newPerson.LastName,
         EmailAddress = newPerson.EmailAddress
};
         await _db.InsertPerson(p);

         people.Add(p); // adding direct because it is not dat
abase specific.
// OR  people = await _db.GetPeople(); // Will update from the
database after you insert the record
         newPerson = new DisplayPersonModel();
}

//Documentation is addiquite for new project.
```

To-dos

- [x] ~~Will Complete by Thursday @Wouter L~~ ~~Thu, Jan 9, 2020~~
- [ ] Testing will be done then. @Developer I Fri, Jan 10, 2020

Bug fixes

```
//Please note that <thread> need to be removed. to fix bug.
<thread>
  <tr>
    <th style="background-color:whitesmoke;color:teal">First N
ame</th>
    <th style="background-color:whitesmoke;color:teal">Last Na
me</th>
    <th style="background-color:whitesmoke;color:teal">Email A
ddress</th>
  </tr>
</thread>
```