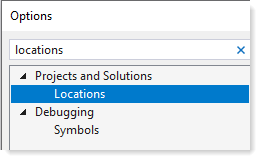
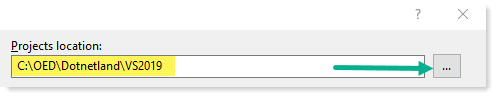
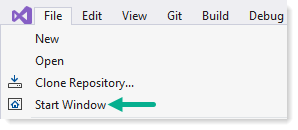
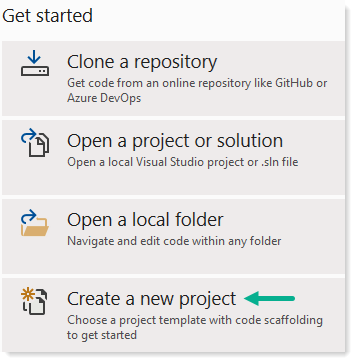
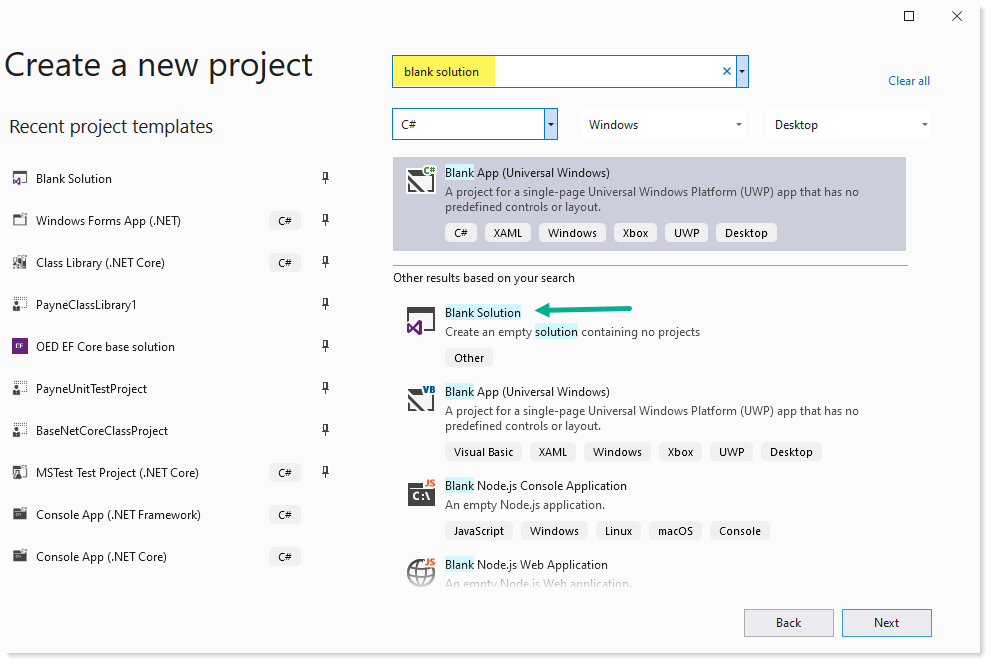
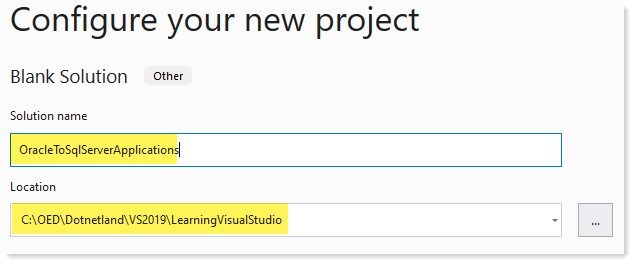
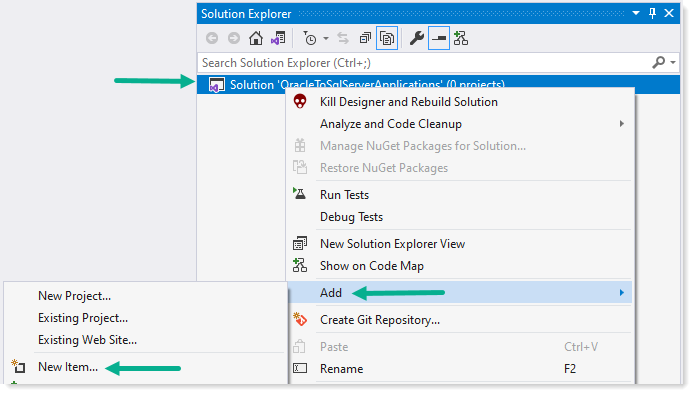
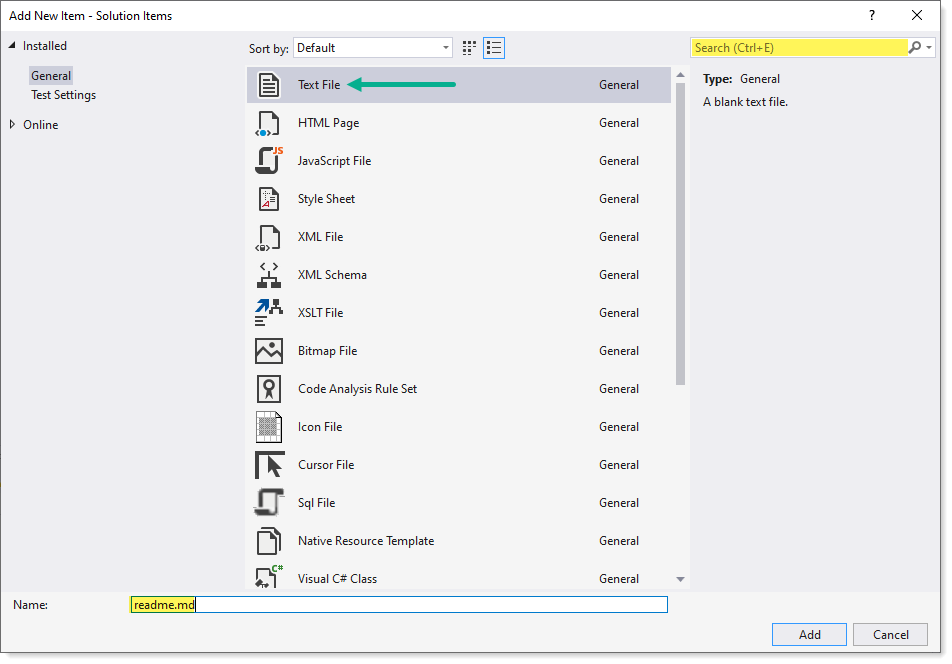
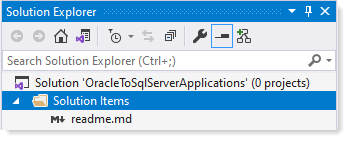
Create new solution

A project is contained within a solution. Despite its name, a solution is not an "answer". It is simply a container for one or more related projects, along with build information, Visual Studio window settings, and any miscellaneous files that aren't associated with a particular project.

1. Open Visual Studio
2. From the menu, select Tools, Options.
3. Top left on the dialog, type in locations and the following appears  
     
   
4. Change “Project location” to the following then click the OK button.  
   
5. From the File menu, select “start window”  
     
   
6. A dialog appears, select create new project  
     
   
7. Select “Blank Solution” then click “Next”  
     
   
8. Provide a meaningful name for the solution, verify the solution is created in the correct folder.  
     
   
9. Click “Create” button
10. Right click on the top node of Solution Explorer, select “Add”, select “new item”  
      
    
11. Select “Text file”, name it readme.md. Every solution should have a [readme.md](https://docs.github.com/en/github/creating-cloning-and-archiving-repositories/about-readmes) file. A readme.md file tells others what the solution is for using Git markdown.   
      
    If your installation of Visual Studio does not have the extension to work with these files (and Karen specified this) and you have Visual Code installed we can work through Visual Studio Code. For GitHub repositories, these files are natively readable.   
      
    Additional readme files with different names may be needed also, for instance, important revision details. Markdown files can be link together and also anchors are supported. For additional details, ask Karen.   
      
    [Basic writing and formatting syntax markdown](https://docs.github.com/en/github/writing-on-github/basic-writing-and-formatting-syntax)  
      
    



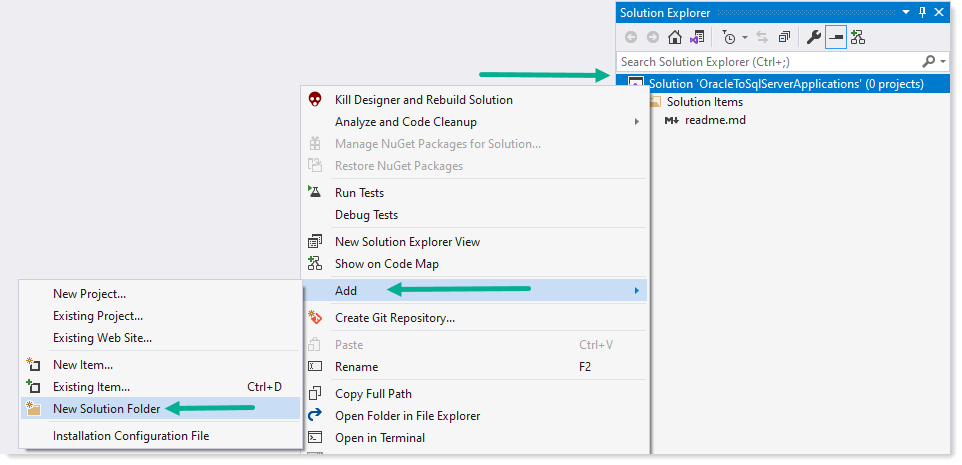
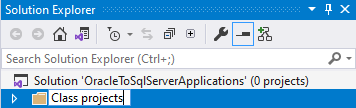
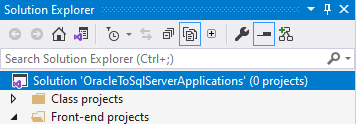
**Solution folders**

|  |
| --- |
| A "solution folder" is a virtual folder that is only in Solution Explorer, where you can use it to group projects in a solution. |

There should be solution folder for at least

* Class projects
* Frontend projects

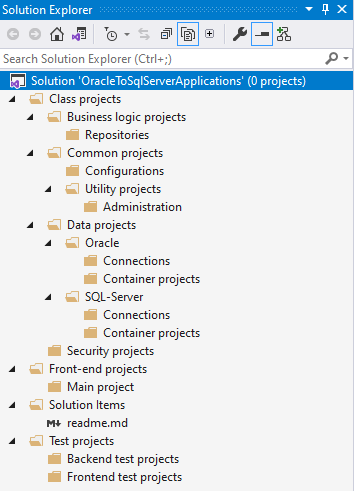
Let’s create two solution folders.

1. From Solution Explorer, right click on the solution node
2. Left click on “Add”
3. Left click on “New solution folder”  
     
   
4. Dialog goes away, type “Class projects”  
     
   
5. Repeat above to add a new solution folder for “Front-end projects”  
   

**Notes**

* When creating a new class project to the solution, right click on the “Class projects” folder to add a new project.
* If by accident you right click on the top node of solution folder to create a new project the new project will not be in any solution folder. To remedy this, single click the new project and drag it to the appropriate solution folder.

An example of what might be done for a solution



|  |
| --- |
| Suggest   * Each project should have a readme.md file which explains the project functions. * Classes, methods etc. should be documented using XML notation (will go over this shortly) |

**Template**

Rather than repeat the above steps, consider creating a template for projects. Karen Payne will do a walkthrough in a later session.

**See also**

[Open a project from a GitHub repo](https://docs.microsoft.com/en-us/visualstudio/get-started/tutorial-open-project-from-repo-visual-studio-2019?view=vs-2019&tabs=vs168later)

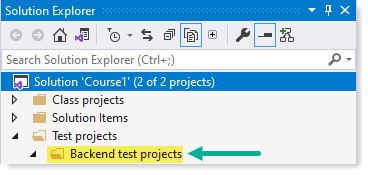
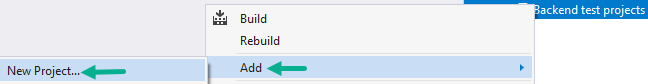
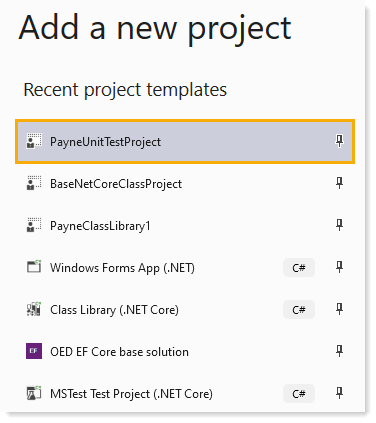
# Rename solution

With Solution Explorer showing

1. Select the first node
2. From Visual Studio’s menu
3. Select File.
4. Save OracleToSqlServerApplications.sln to
5. Enter Course\_1

# Add a unit test project

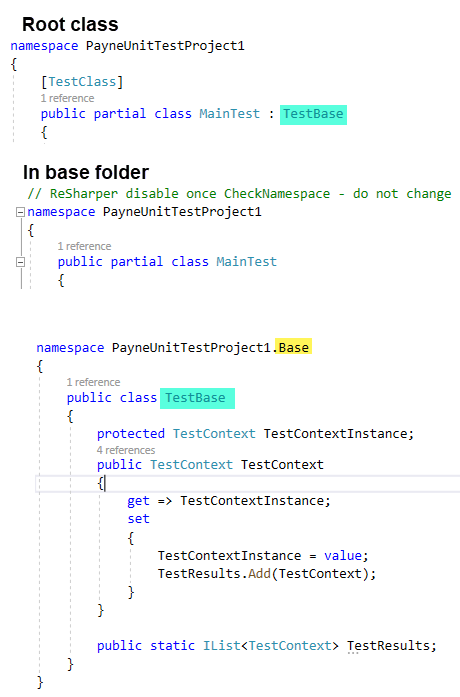
With Solution Explorer showing

1. Select Test projects node, Backend test projects  
   
2. Right click, select Add, New Project.  
   
3. Walk thorough first time using pre-done unit test project PayneUnitTestProject.  
   

# Anatomy of the unit test project

In the case of this custom unit test project

* Base folder contains
  + A main class which all unit test inherit
  + Optionally a partial class for each test class at the root level of the project
  + The class TestTraitsAttribute use to provide trait display in Test Explorer



**Notes**

The field TestResults in TestBase is initialized in MainTest under the base folder. TestResults provides access to TestContext which in turn provides insight into things like what is the name of the current test which can be used before and after a test runs for [TestInitialize] and [TestCleanup] methods.