Single Tier—Three Layer Model

In this model, we logically break BL and DAL in different namespaces, introducing cleaner code separation.

ASP.NET Web Project that has logical separation between presentation, business logic and data access code:

- All presentation code will be under the `MyApp.Web` namespace (Layer 1).
- Furthermore, the single project can have two folders:
    - Business (Layer 2): for business logic code, with namespace `MyApp.Code.Business`
    - DAL (Layer 3): for data access code, with namespace `MyApp.Code.DAL`

Note that it is not necessary to have different folders for the logical separation of code; using different namespaces in different code files will also work fine. We can use this model for a medium-to-large web application where we know that many users won't log in simultaneously. For handling a large number of users, the application needs to be scalable, and to do this we might need to separate BL and DAL code into their own physical assemblies.

Two Tier Model

Here we create two projects, one normal web project for UI code, and another class library project for the BL and DAL code. This will ensure that even if we change the BL or DAL code, we don't need to recompile the web project as we have separate physical assemblies. This setup is more scalable and maintainable than all previous options. Separating code into different assemblies will involve a slight performance hit, but that is negligible considering the flexibility and maintainability benefits we get by having two tiers.

The solution will have:

- ASP.NET Web Project having GUI and presentation code (Tier 1)
- A class library project having business logic and data access coding under a single namespace, `MyApp.Code`; no separate namespaces for business logic and data access code (Tier 2)

In this case, we still have the BL and DAL code under one namespace, but we can logically separate them further, as shown below.

Two Tier—Two Layer Model

We can further separate the BL and DAL code into their own separate namespaces and class files, so that different developers can work on BL and DAL simultaneously, under a multiteam set up.

The solution will have:

- ASP.NET Web Project having Presentation Layer coding in ASPX and ASCX files, under the namespace, `MyApp.Web` (Tier 1)
- A class library project having two folders (Tier 2):
    - Business: for business logic code, with namespace `MyApp.Code.Business` (Layer 1)
    - DAL: for data access code, with namespace `MyApp.Code.DAL` (Layer 2)

Three Tier Model

If the project is large, with a lot of complicated business logic, then it's more useful to separate the BL and DAL into in their own assemblies so that we can change the BL code without changing the DAL assembly. This makes our application more flexible and loosely-coupled as we can use a different DAL assembly for a different database with the same BL assembly.

The solution will have:

- ASP.NET Web Project having Presentation Layer coding in ASPX and ASCX files, under namespace `MyApp.Web` (Tier 1)
- A class library project having business logic code, with namespace, `MyApp.Code.Business` (Tier 2)
- A class library project DAL for data access code, with namespace, `MyApp.Code.DAL` (Tier 3)

Once again, if we also bring the Presentation and Database to be a part of the entire application here, the above 3-tier model would become a 5-tier model!