Architectural Styles
3 - Tier/Layered Architecture

Software Design Architecture Lab # 12

Muhammad Rehan Baig

3 Tier Architecture

- A 3-tier architecture is a type of software architecture which is composed of three “tiers” or “layers” of logical computing. They are often used in applications as a specific type of client-server system.
- Basically it is a Client Server Architecture.
- It is a type of multilayered architecture where **presentation**, **application processing** and **data management** functions are physically separated.
- architecture provides a model by which developers can create flexible and reusable applications.

3 Tier Architecture – Difference between Tiers and Layers

- Tiers- A Tier is simply the physical separation of your app components.
- Layers- Layers act as more logical separators that exist to separate and organize your actual code. You'll often hear terms like "Business Logic Layer", "Presentation Layer" and others. These are simply ways to organize all of the code within your application.

3 Tier Architecture – Difference between Tiers and Layers (Example)

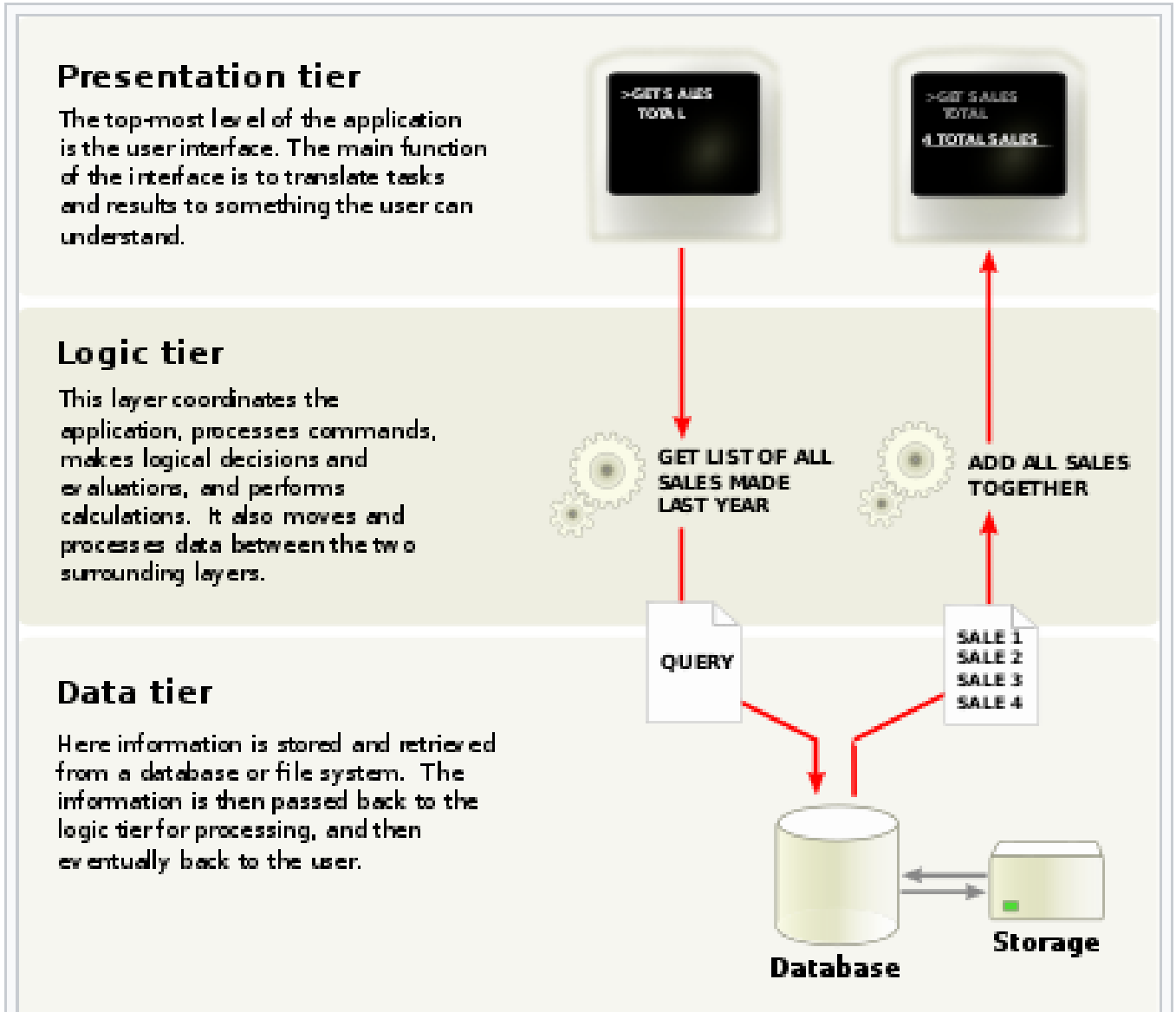
- If you have a web app that contains your Data Access and Business Logic which is running on the same machine/Server, then you will have 3-layered app in 1-Tier.
- Now if your Data Access is hosted on different machine/server, and your Business is also hosted in different machine/server, then you will now have a 3-layered app in 3-Tier.

3 Tier Architecture

- Basically Based on 3 components

1. Presentation Layer/Tier
2. Business Logic Layer/Tier
3. Data Access Layer/Tier

3 Tier Architecture



3 Tier Architecture - Features

- **Secure:** You can secure each of the three tiers separately using different methods.
- **Easy to manage:** You can manage each tier separately, adding or modifying each tier without affecting the other tiers.
- **Scalable:** If you need to add more resources, you can do it per tier, without affecting the other tiers.
- **Flexible:** Apart from isolated scalability, you can also expand each tier in any manner that your requirements dictate

Why and When to Use Tier/Layered Architecture

- For increasing security of an application
- For Easy Modification without effecting other components
- you can adopt new technologies and add more components without having to rewrite the entire application or redesigning your whole software, thus making it easier to scale or maintain
- in terms of security, you can store sensitive or confidential information in the logic tier, keeping it away from the presentation tier, thus making it more secure.

Why and When to Use Tier/Layered Architecture

- **More efficient development.** tier architecture is very friendly for development, as different teams may work on each tier. This way, you can be sure the design and presentation professionals work on the presentation tier and the database experts work on the data tier.
- **Easy to add new features.** If you want to introduce a new feature, you can add it to the appropriate tier without affecting the other tiers.
- **Easy to reuse.** Because the application is divided into independent tiers, you can easily **reuse** each tier for other software projects. For instance, if you want to use the same program, but for a different data set, you can just replicate the logic and presentation tiers and then create a new data tier.

Implementation of Layered/Tier Architecture

- We are using .net framework and c# for implementing Tiers for our application in this Lab
- Steps Included are

1. Create a new project as a asp.net web application (New Solution Created → in this solution Add new class library as a new project)
2. Add Class library as a new Project and Name that project as Business logic Layer
3. Add New Class library to the solution again as a new project and name that project as Data Access layer

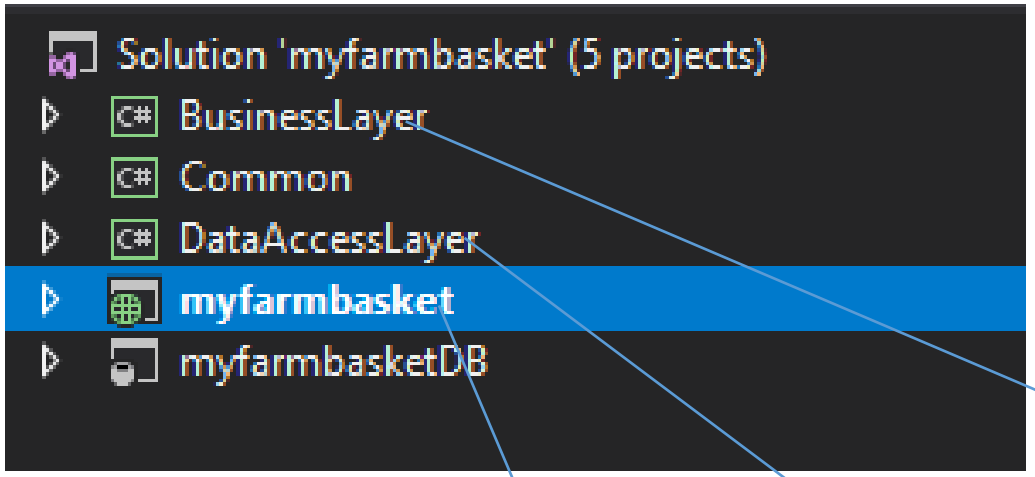
Implementation of Layered/Tier Architecture

1. our first created project in the solution that is not a class library project is our application / or sometimes presentation layer (In this project we specify our UI or we write our web services/ webapis so this layer is exposed to the internet and client can interact with this layer)
2. In our second project in the solution that is class library project that we named as business logic layer we write our managements(Business logic/policies/rules) of our system/project
3. In our third project that is a class library project we named that as data access layer we write database transactions in this project.

Implementation of Layered/Tier Architecture

We can add more layers or tiers in order to increase loose coupling and modularity or in order to add new technologies.

Example Project – Real Life Implementation

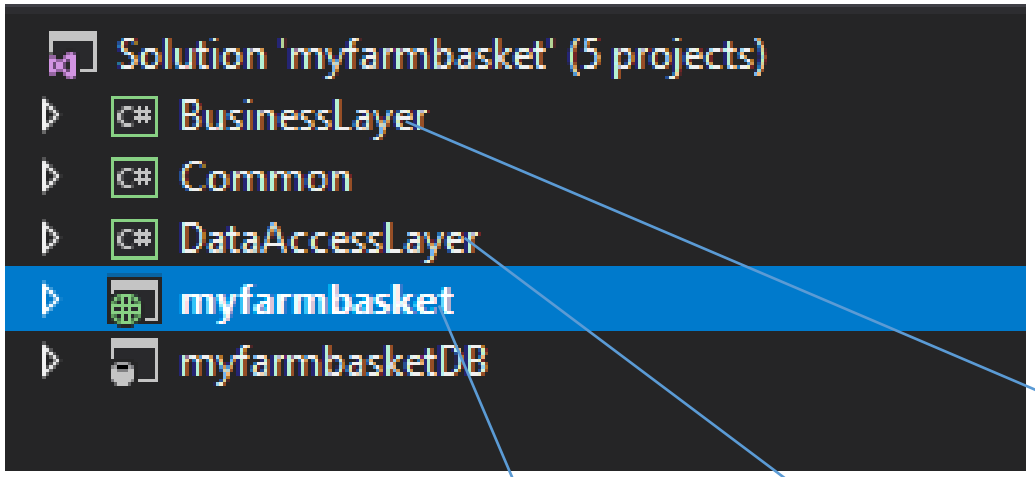


Business Logic Layer

Application/Presentation Layer

Data Access Layer

Example Project – Real Life Implementation

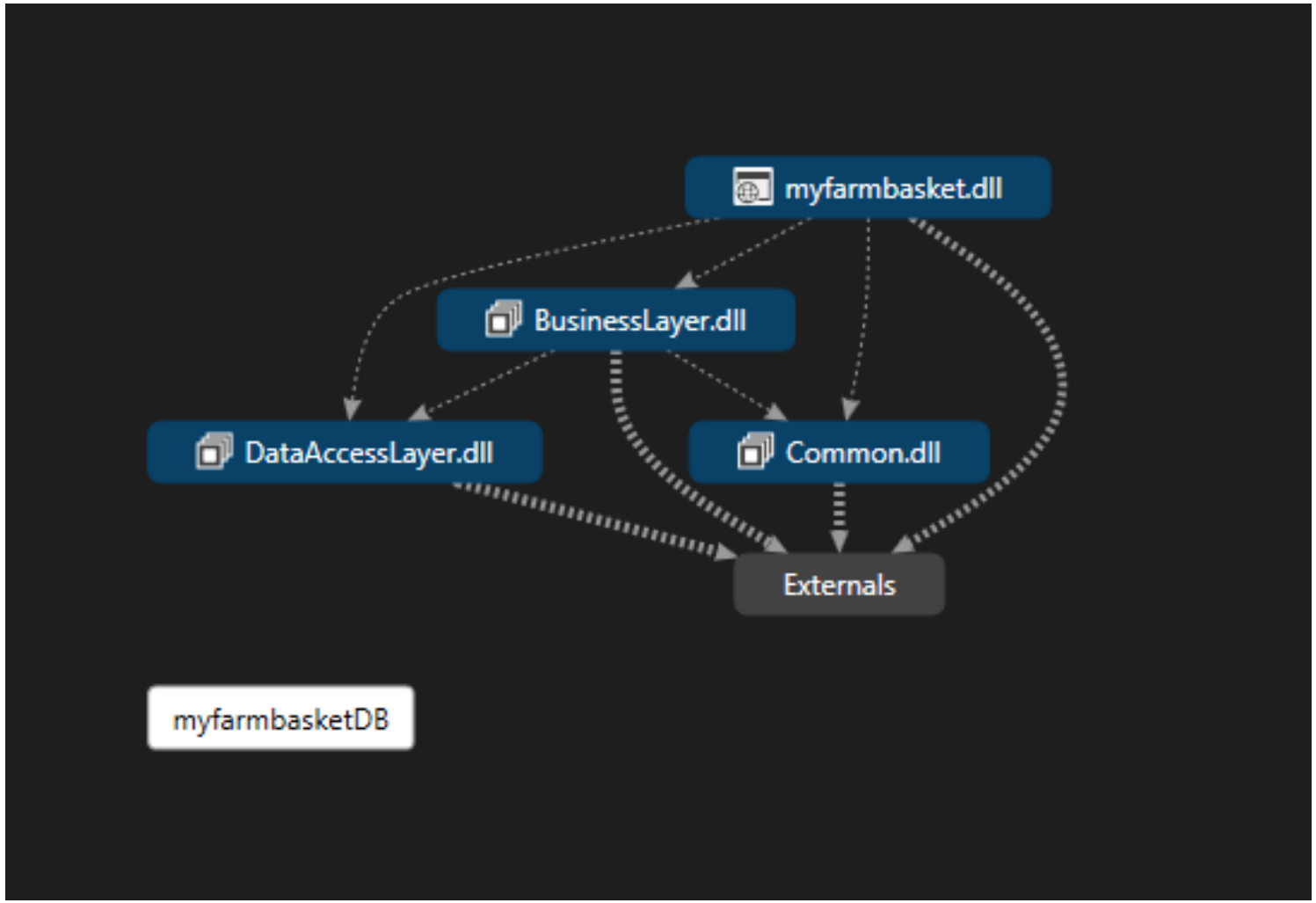


Business Logic Layer

Application/Presentation Layer

Data Access Layer

Example Project – Real Life Implementation Pictorial View



Example Step by Step

Tasks

1. **Create** layered architecture style project for Generating Marksheet of a student.
   The system can be able to add new students → add scores of subjects(you can make 5 subjects of your choice) → Show list of students from database → Generate mark sheet for for particular student when click on it.

Useful link

- https://docs.microsoft.com/en-us/visualstudio/data-tools/walkthrough-creating-an-n-tier-data-application?view=vs-2015&redirectedfrom=MSDN