Layered Architecture & Project

Course Code: CSC 2210 Course Title: Object Oriented Programming 2

Dept. of Computer Science
Faculty of Science and Technology

Lecturer No:	**12**	**Week No:**	**13**	**Semester:**	
Lecturer:					

Lecture Outline

- Discuss the basics of layered architecture in desktop app.
- Discuss the advantages and disadvantages.
- Working policies.
- Discuss about the project proposal.
- Discuss about the project demonstration.

Layered Architecture

Layered Architecture

- Layered architecture is an approach to breaking down large applications into manageable areas.
- This improves maintainability of the software by isolating functionality, improving testability, allowing code to be reused, and changes to be made to a layer without impacting another in a significant way.
- When the various components in a system are organized systematically we call it a system architecture.
- Each layer having responsibility for a major part of the system and with as little direct influence on other layers.
- **Tiers and Layers are used interchangeably**.
- Layers can be applied to the logical structure of the application and Tiers can be applied to the physical structure of the system architecture or infrastructure.

Layered Architecture

Tiers

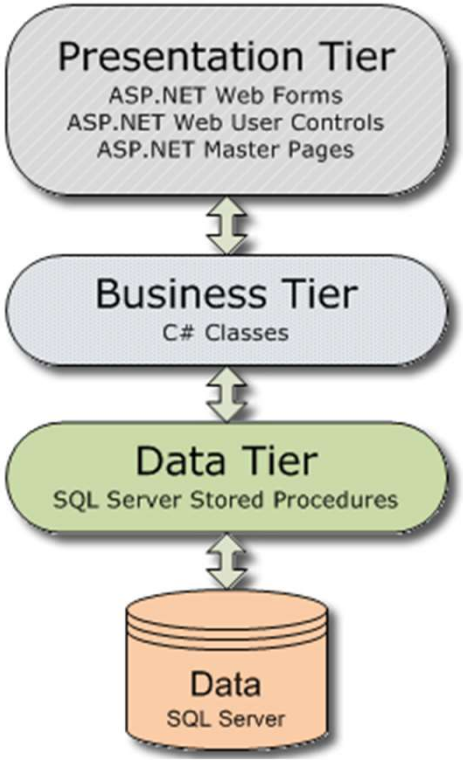
- The terms tier and layer are often used interchangeably. This isn't wrong when the context is correct, but it is important to make and understand the distinction that they are not the same in software architecture. A tier refers to the physical location that your code base runs. A common case would be a client-server application that has two tiers, the physical client, and physical server. Tiers are distinguished by they fact that they are separated by physical boundaries.

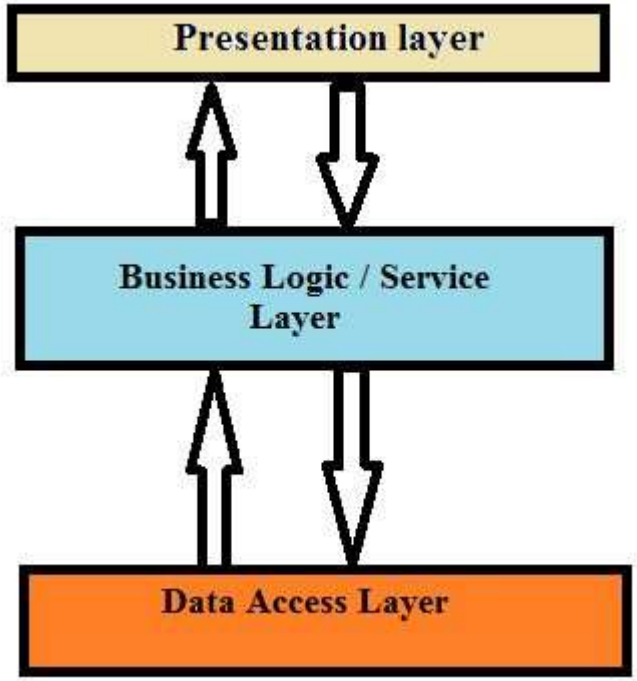
Layered Architecture

Layers

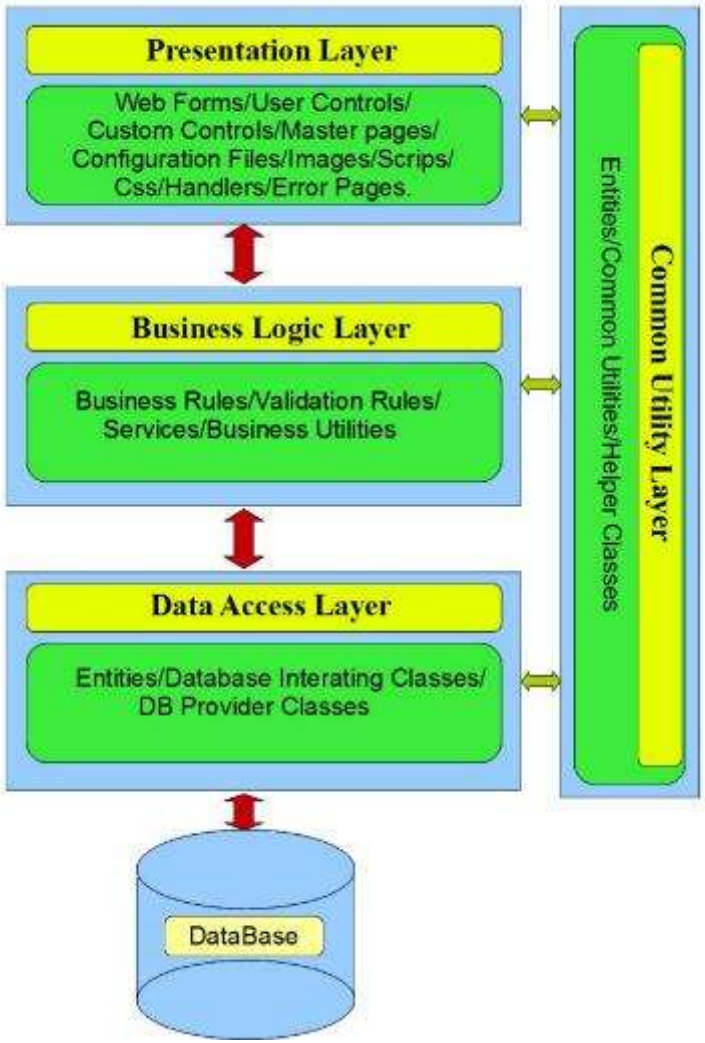
- Layers on the other hand, are logical separations of concerns in your application architecture. You might design your architecture to run a layer of the software on a specific tier, or share a layer across tiers. These decisions usually account for requirements such as security, performance, or management of the infrastructure. The most common and accepted layers of software are listed below:
  - *Data Layer (also combination of Entity Layer)*
  - *Business Layer*
  - *Service Layer*
  - *Application Layer*

Layered Diagram





Layered Diagram



Layered Architecture

Data Layer

- The data layer is also referred to as the DAL, or data abstraction layer.
- This particular area of the software is responsible for encapsulating all access to application data.
- Application data will be stored in a database like SQL Server or MySQL, but the DAL can also encapsulate data stored in other formats, such as xml.
- Sometimes this layer also incorporates '*Entity layer'* which often considered as separate layer.
- ***Entity layer*** includes the classes which maps the DB table with the system.

Layered Architecture

Business Layer

- The business layer is also referred to as business abstraction layer.
- Sometimes it is also referred to as the domain layer.
- The business layer is responsible for encapsulating all business logic for a given problem domain.

Layered Architecture

Service Layer

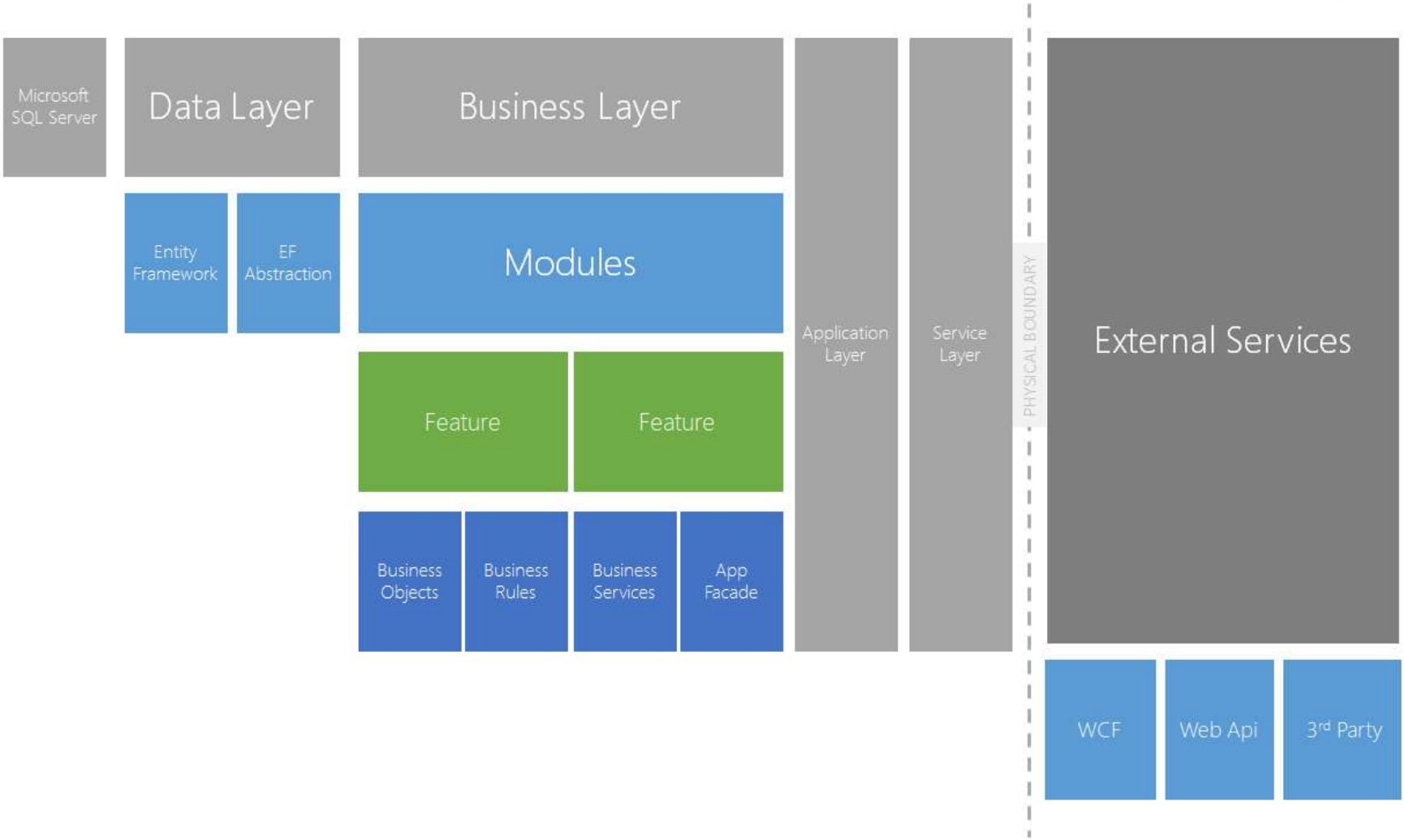
- An application architecture might contain one or more service layers.
- Typically, a service layer is used to abstract communication between a set of services.
- Services could be web services, third party API, or other subsystems of a product.
- It's primary goal is to provide a consistent mechanism for sending data, receiving data, and handling errors.
- This layer is also known as framework layer.

Layered Architecture

Application Layer

- The application layer is also called the presentation layer
- It is the layer of the software responsible for the user interface, and ultimately the specific environment hosting the software.
- In a web application, your app layer will be the web application and the web server, while in a desktop application, it would be the desktop client and the OS.

Layered Diagram



Layered Architecture

Which Architecture to use

- The right architecture largely depends on the application you will build. Designer have to ask some question which are:
  - *How many tiers will your architecture have?*
  - *Is your application heavily data driven?*
  - *Do you have complex domain requirements or business rules?*
- Small applications might not require a layered architecture at all.
- The important part is you architect a solution that fits your needs.

Layered Architecture

Advantages

- Improve productivity via Reuse (The entire Data Access Layer module is reusable).
- Improve Productivity via Team Segment.
- Improve Maintainability.
- If a change is made with Database it will only effect only DAL. If change is BL it will only effect BL.
- Looser coupling.
- More physical deployment. Adding this feature can affect the performance.

Layered Architecture

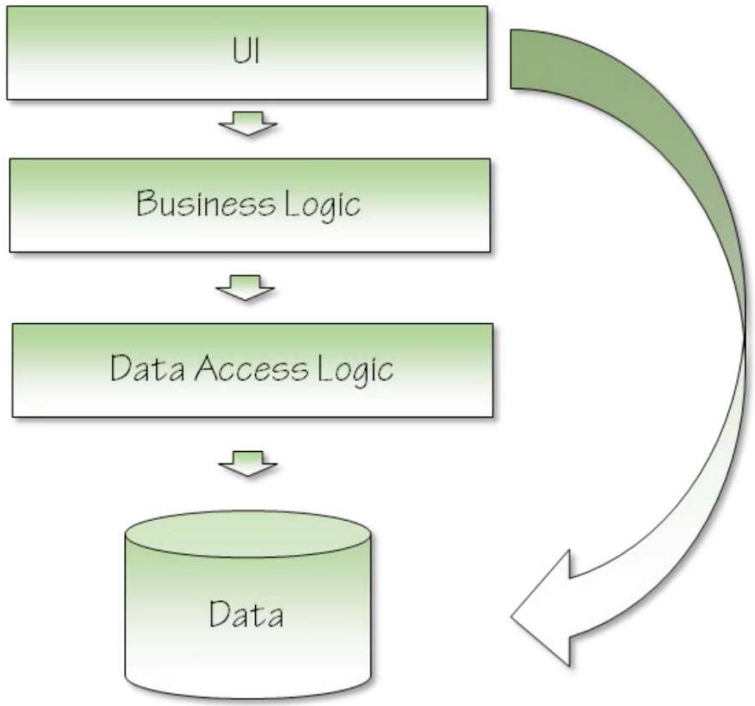
Disadvantages

- Adding additional servers to our application to increase its performance and scalability. But in reality is that the fastest an application can perform is when all its processes is in-process in a single machine.
- More complex designs.
- Complex and good amount of skills and experience required for the team.
- More complex deployment.
- Loose coupling allow connected modules to change independently of one another.

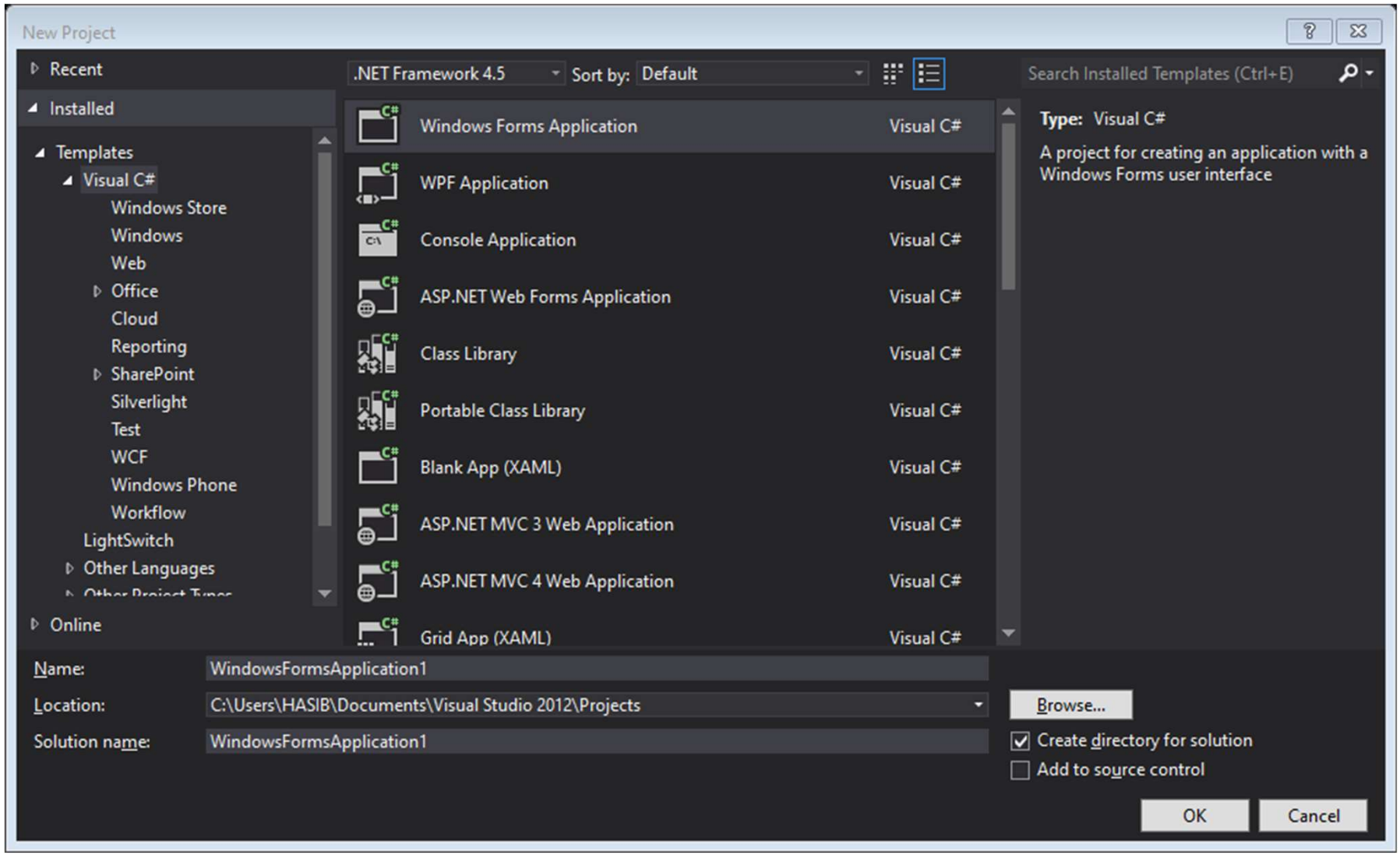
Layered Architecture

Monolithic Application

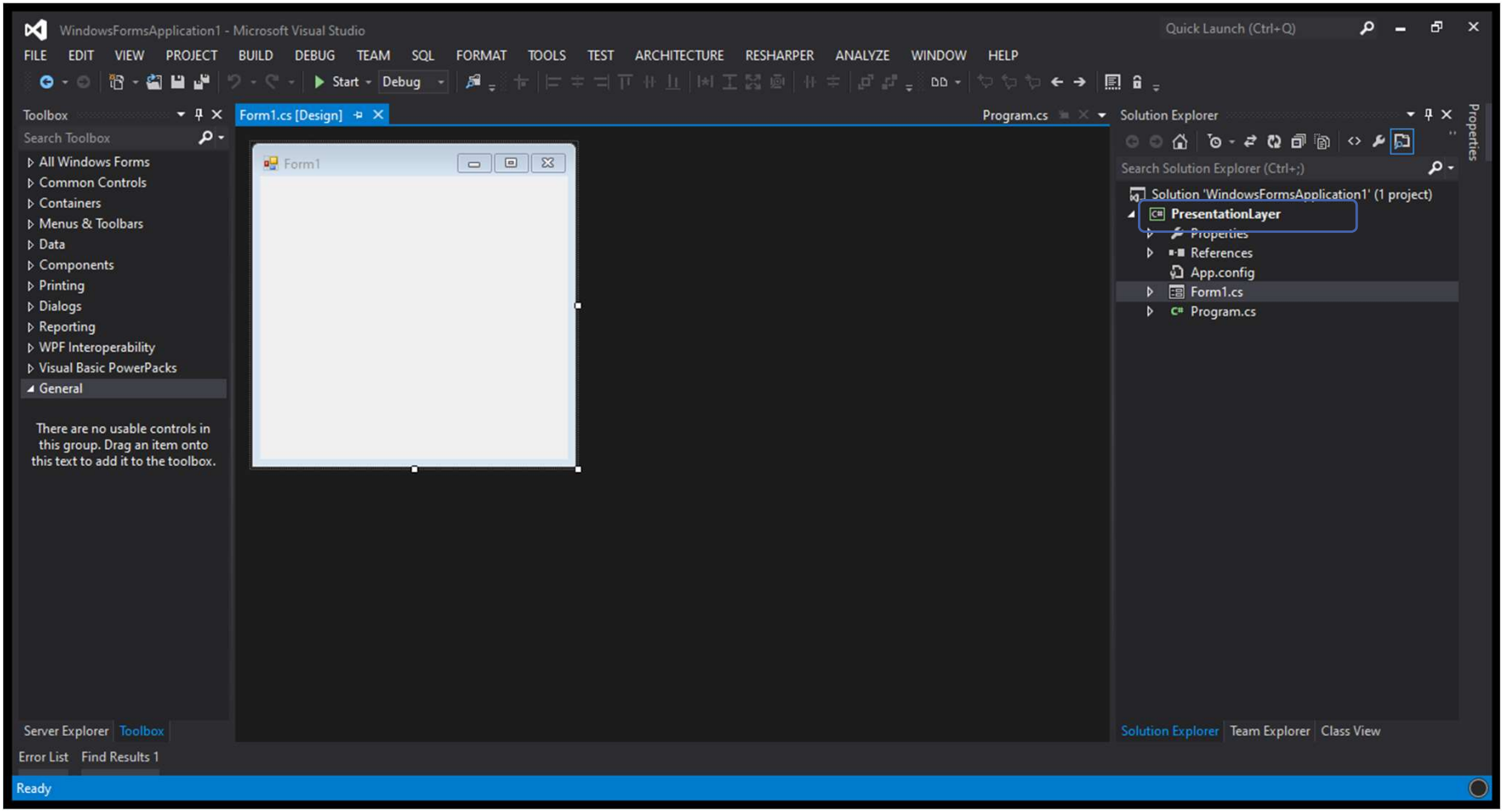
- N-tier application where N= 1, Single unit of deployment.
- No logical or physical separation of concerns.
- All parts of the application are included in one assembly. For example: Microsoft Access Database



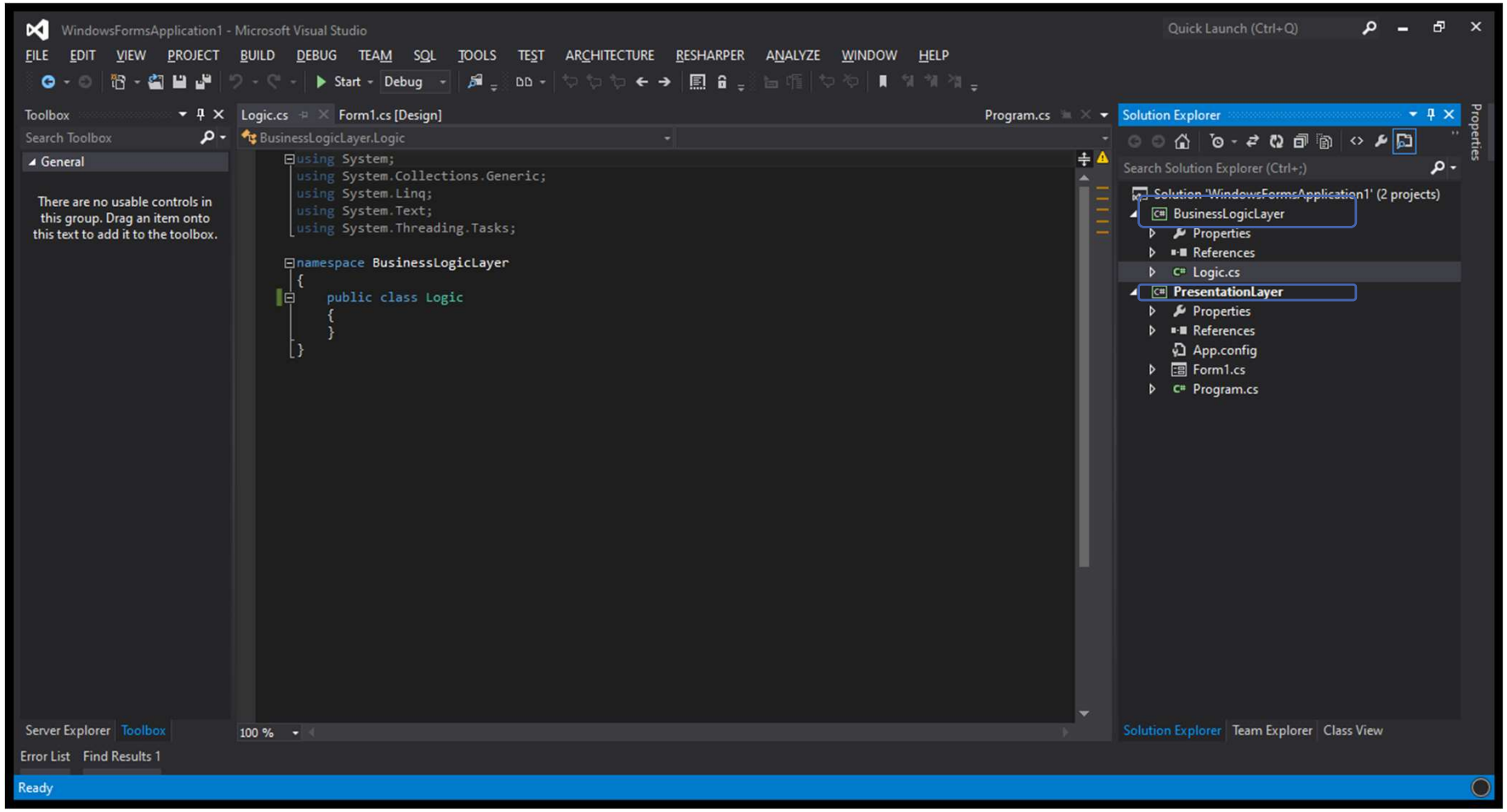
Add a Windows Form Application Project (Presentation Layer)



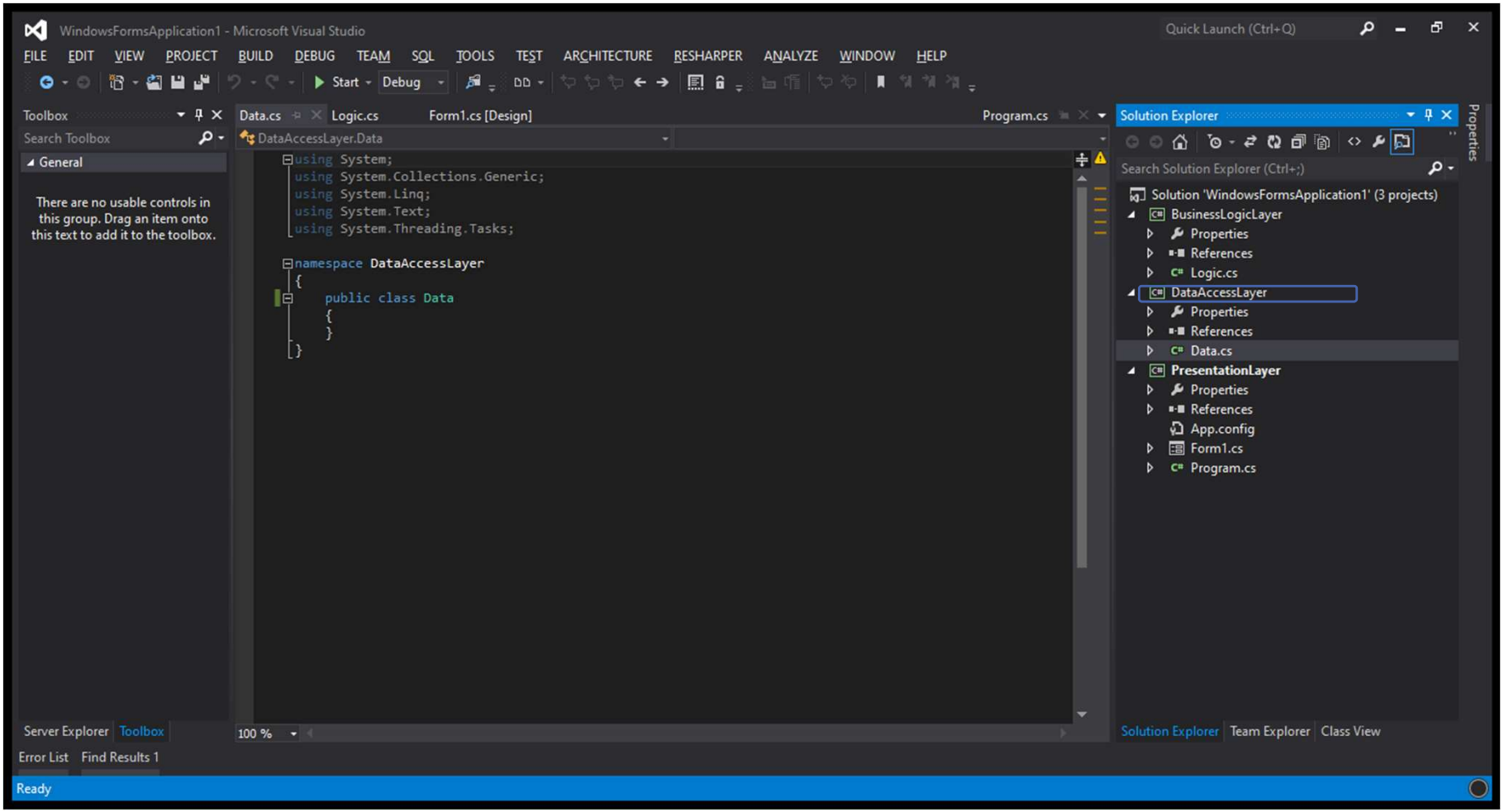
Let's set up our next projects and define them in separate layers.Add 3 projects to your solution named BusinessLogicLayer, DataAccessLayer and ServiceLayer.



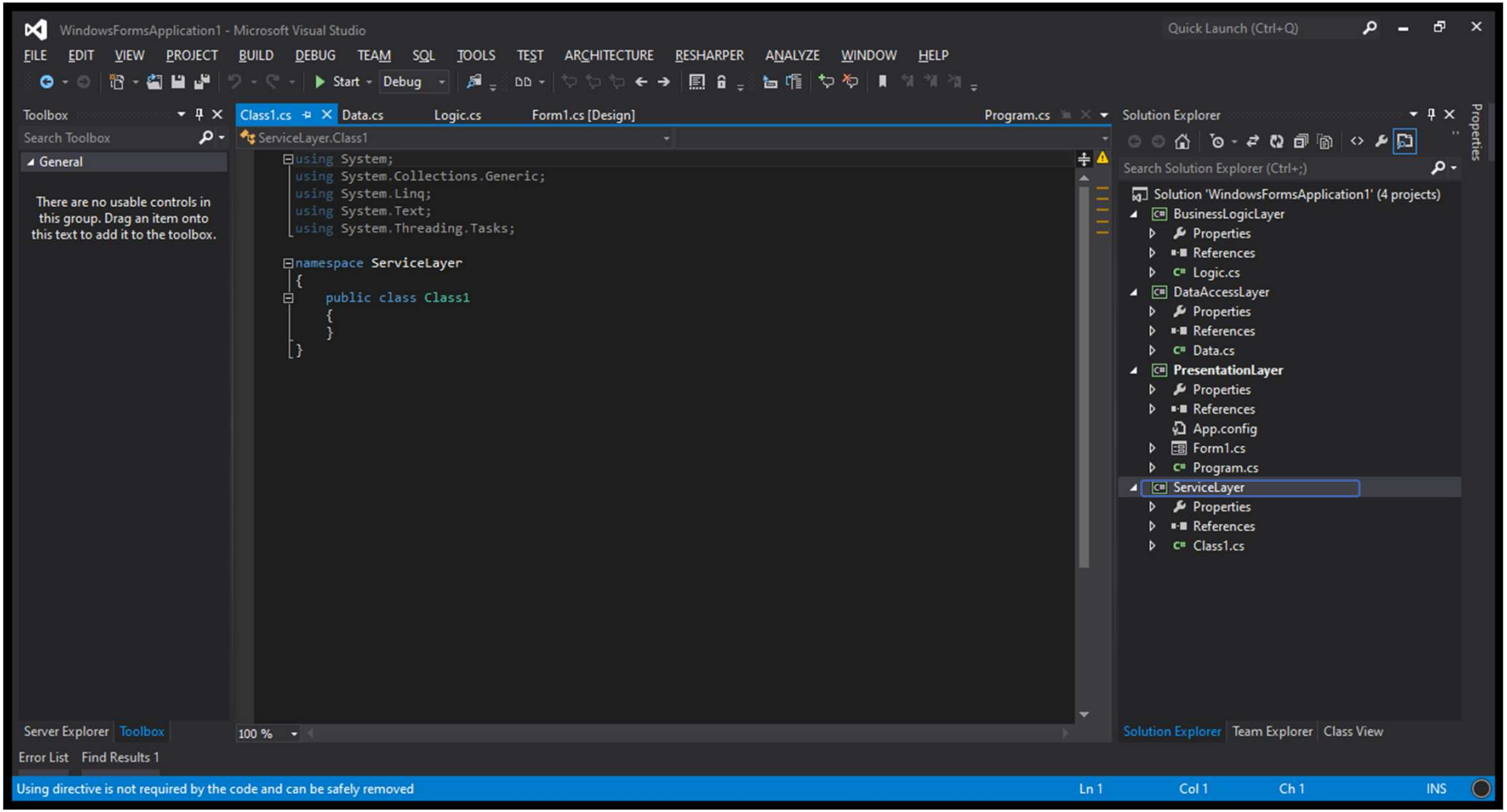
Creating Business Logic Layer



Creating Data Access Layer



Creating Service Layer



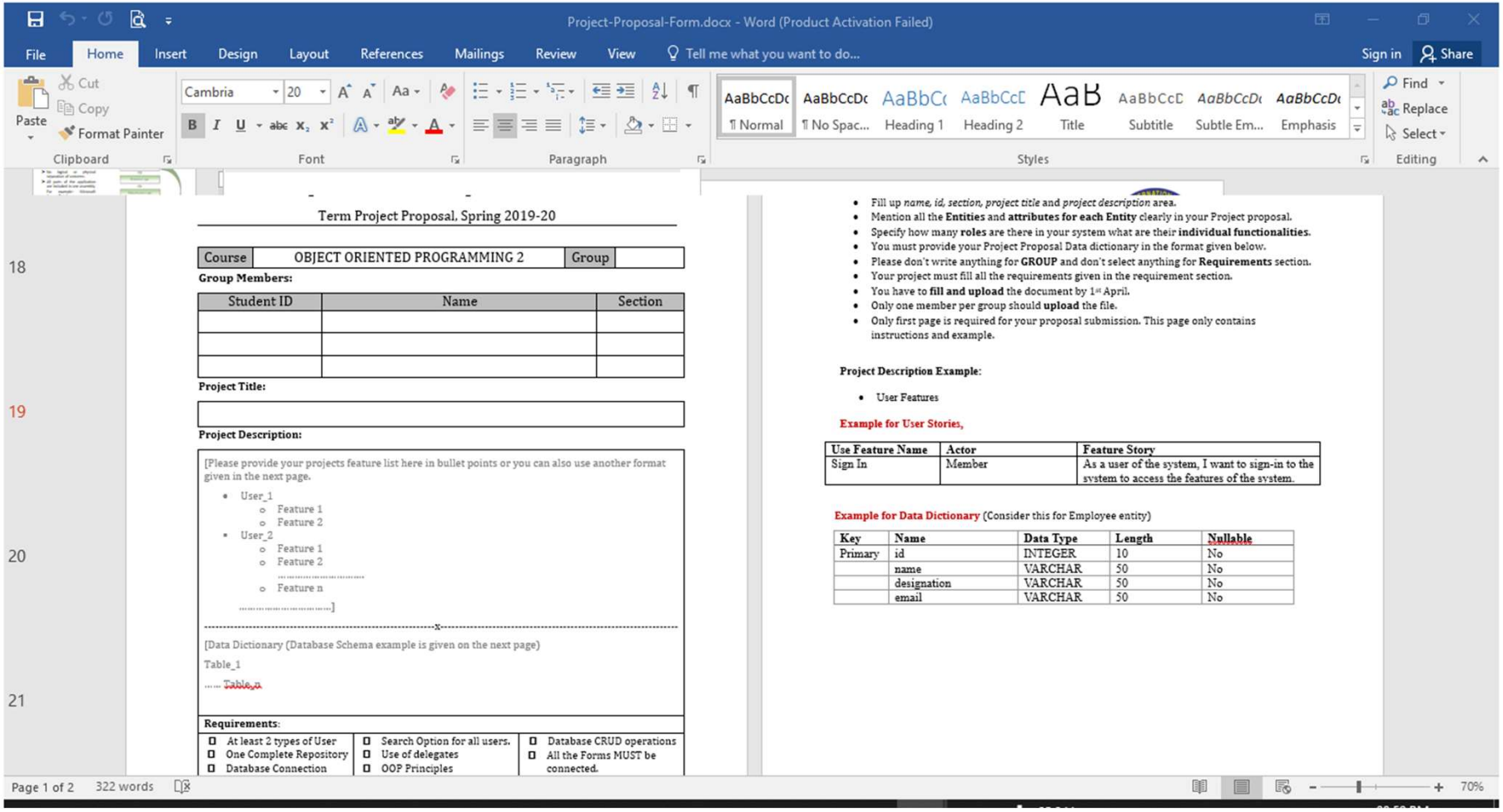
Layered Architecture

Common Rules to practice

- The Data Access Layer Project Contains the entities classes and objects to communicate with the database.
- Service layer contains properties and helper classes to communicate with all.
- It also acts as a mode of communication among the layers.
- Presentation Layer needs no direct interact with the database.
- Add a reference of the Business Logic Layer to Presentation Layer and Data Access Layer to the Business Logic Layer and Service Layer to all other layers.

Project Discussion

Project Proposal Form



Project-Proposal-Form.docx - Word (Product Activation Failed)

File Home Insert Design Layout References Mailings Review View Tell me what you want to do... Sign in Share

Cut Copy Format Painter Paste — Clipboard

Cambria 20 — Font

Paragraph

Normal | No Spac... | Heading 1 | Heading 2 | Title | Subtitle | Subtle Em... | Emphasis — Styles

Find Replace Select — Editing

Term Project Proposal, Spring 2019-20

Course	OBJECT ORIENTED PROGRAMMING 2	Group	

Group Members:

Student ID	Name	Section

Project Title:

Project Description:

[Please provide your projects feature list here in bullet points or you can also use another format given in the next page.

- User_1
  - Feature 1
  - Feature 2
- User_2
  - Feature 1
  - Feature 2
  - .....
  - Feature n

.....]

----------x----------

[Data Dictionary (Database Schema example is given on the next page)

Table_1

..... Table_n

Requirements:

☐ At least 2 types of User	☐ Search Option for all users.	☐ Database CRUD operations
☐ One Complete Repository	☐ Use of delegates	☐ All the Forms MUST be connected.
☐ Database Connection	☐ OOP Principles	

- Fill up *name, id, section, project title* and *project description* area.
- Mention all the **Entities** and **attributes for each Entity** clearly in your Project proposal.
- Specify how many **roles** are there in your system what are their **individual functionalities.**
- You must provide your Project Proposal Data dictionary in the format given below.
- Please don't write anything for **GROUP** and don't select anything for **Requirements** section.
- Your project must fill all the requirements given in the requirement section.
- You have to **fill and upload** the document by 1st April.
- Only one member per group should **upload** the file.
- Only first page is required for your proposal submission. This page only contains instructions and example.

Project Description Example:

- User Features

Example for User Stories,

Use Feature Name	Actor	Feature Story
Sign In	Member	As a user of the system, I want to sign-in to the system to access the features of the system.

Example for Data Dictionary (Consider this for Employee entity)

Key	Name	Data Type	Length	Nullable
Primary	id	INTEGER	10	No
	name	VARCHAR	50	No
	designation	VARCHAR	50	No
	email	VARCHAR	50	No

Page 1 of 2 322 words 70%

Thank You

Books

- C# 4.0 The Complete Reference; Herbert Schildt; McGraw-Hill Osborne Media; 2010
- Head First C# by Andrew Stellman
- Fundamentals of Computer Programming with CSharp – Nakov v2013

References

- https://www.c-sharpcorner.com/UploadFile/1492b1/understanding-multilayered-architecture-in-net/
- https://medium.com/@hidayatarg/n-tier-layer-architecture-in-c-15b8fe97283c
- http://danderson.io/layered-architecture/
- https://github.com/topics/layered-architecture?l=c%23