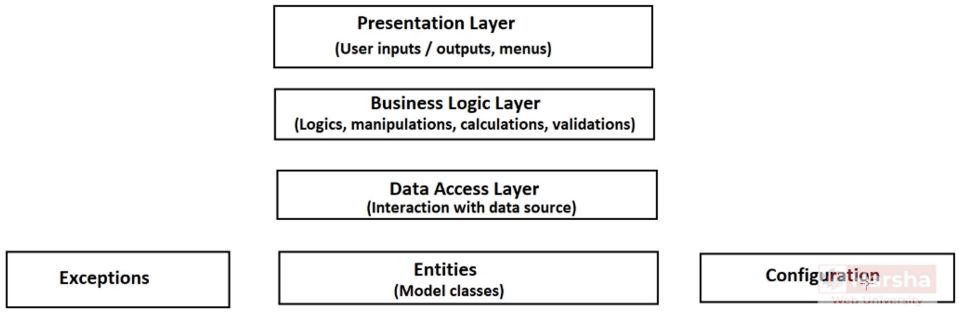
Section 29: Bank Project - Adding Functionality

06 August 2024 22:43




In this lecture I will demonstrate how to create the complete architecture of the application includes all the layers here. Layer is a class library that contains a
meaningful piece of code that is responsible to perform specific task within the project

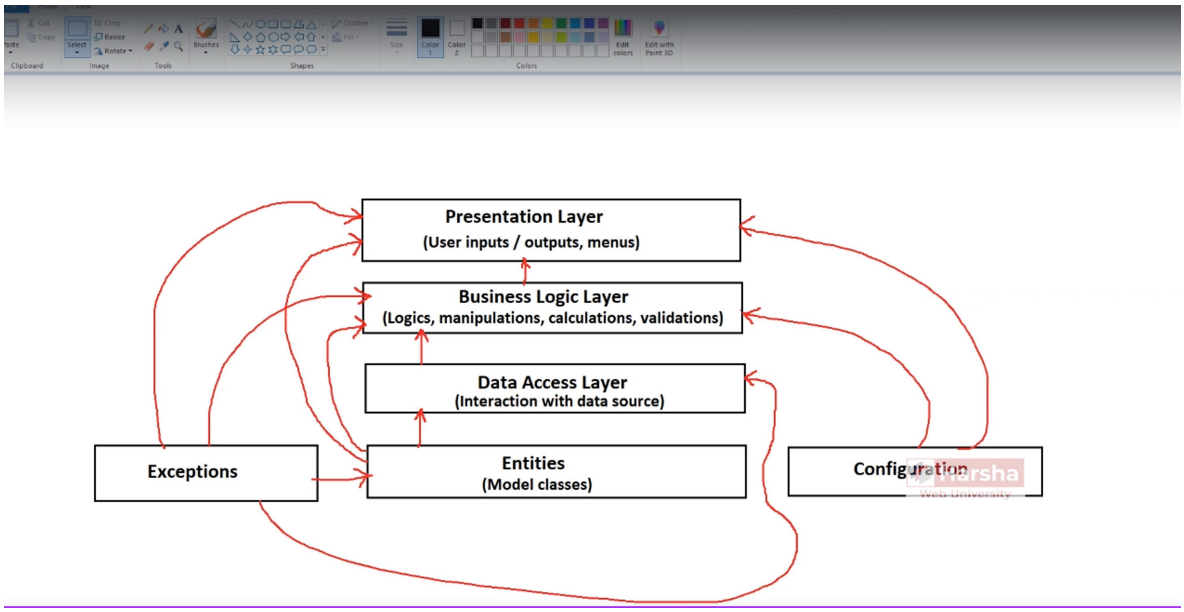
For example we have already created the presentation layer which contains the code for reading the inputs from the user and providing the outputs to the user and also showing the user menus. So the presentation layer is completely responsible to
interact with the user but

The presentation layer is not responsible to validate anything or perform the logical part such as calculations or manipulations or making addition whether the data should be stored in the database table or not whereas the business logic layer is another class library which contains the logical part of the application which includes the manipulations calculations and also validation of data indicating whether the data should be stored or retrieved from the data store the data access layer is another class library which contains the code for interacting with the data source
but it doesn't include any interaction with the user or performing validations

The bottom layer of the data access layer is entities that contains the model classes that represents each and every data structure such as customer bank account etc
additionally you are creating other class libraries such as exceptions or configuration

The Exception layer contains the custom exception classes. For example for customers entity you can create customers exception any kind of validation error related to
customers will be represented as customers exception that means instead of representing all runtime errors and validation errors as the base class called system.exception

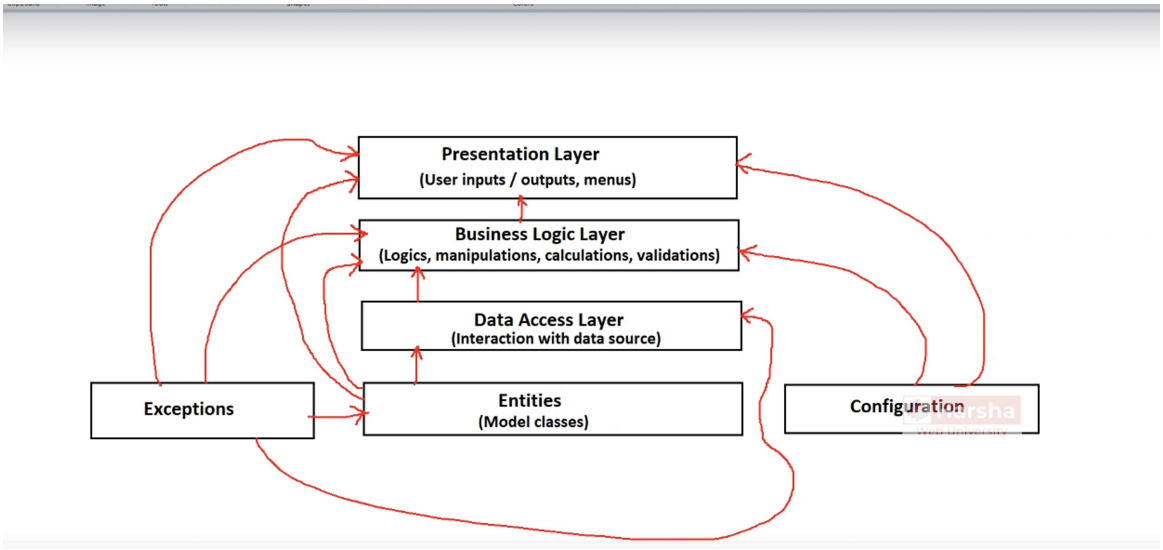
The Configuration layer we can create the static classes in order to contain the application level settings such as base values for automatic generation of customer number these are the layers that we have in the current application.




These are the references that we have added in our current solution.

*it's a good practice of programming to create interfaces for each and every entity because in real world applications the other teams don't want to wait for the completion of entity classes they want to continue writing the code based on the interface so first interface should be created based on the interface the other teams can continue writing the code for data access layer or business logic layer or even the presentation layer. Generally we call interfaces as Contracts.*

## Creating Customer Data Access Layer




Business Logic Layer will call the Data Access Layer, and Data Access Layer will access the Entities.

In our Configuration Project,

***In this lecture we are going to add configuration settings file in the current project so in the real projects there may be so many things that you need to configure globally such as base numbers for automatic generated ids***

***the database server name the client name the location etc so.***

so the business logic layer is a class that contains the programmatical logic
that means the decision making before doing something for example before adding a customer it has to check whether the customer details are correct or not

that means all the values are as per the expected format or not. In case if the customer values are invalid. The customer details should not be inserted into the collection
like this the business logic layer is responsible for decision making validations manipulations calculations all the logical part of the application

But business logic should not interact with the data source directly the business logic should call the data access layer in order to interact with the data source and also business logic layer should never interact with the user directly that means should not show any output to the user by using console.writeline or should not read any inputs from the user by using console.readline

business logic layer is the mediator between the presentation logic and data access logic business logic layer can access the entity classes that are created in the entities class library.

the benefit of creating interface is that the color of the business logic layer that means the presentation logic can create a reference variable for the business logic layer interface and invoke the essential methods without worrying about or without waiting for the completion of development of actual business logic layer code

**The method signature of Business Layer and Data Access Layer will be mostly same.**

in this lecture we are going to add presentation logic layer for the customer's entity that means we are accepting the real inputs from the user by using console.readline and also we are providing the output by using console.writeline

but in the presentation layer we don't include any validation logic that means it will not be responsible to perform manipulations calculations

or any kind of decision making all that logical part is already included in the business logic layer. so overall except in presentation layer in no layer else you can add the io operations with the user that means console.writeline or console.readline

we have already created the presentation layer in the beginning itself