Session 14

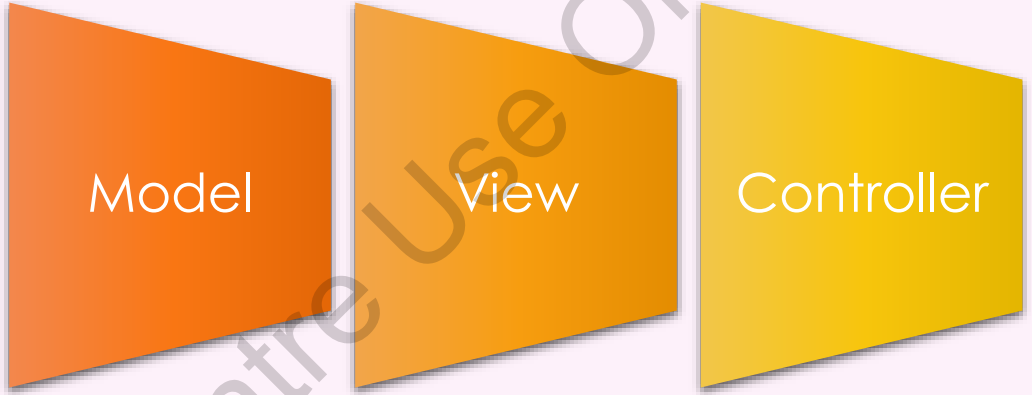
Unit of Work Patterns in ASP.NET Core

Session Overview

- Describe repository pattern
- List advantages of repository pattern
- Explain the difference between repository pattern and repository pattern with unit of work
- List benefits of unit of work

Introduction

Layers used in traditional ASP.NET MVC 5 Web applications:



- View is used for presentation or UI.
- Model is responsible for the database interaction.
- Controller handles all the CRUD operations.

Repository Pattern

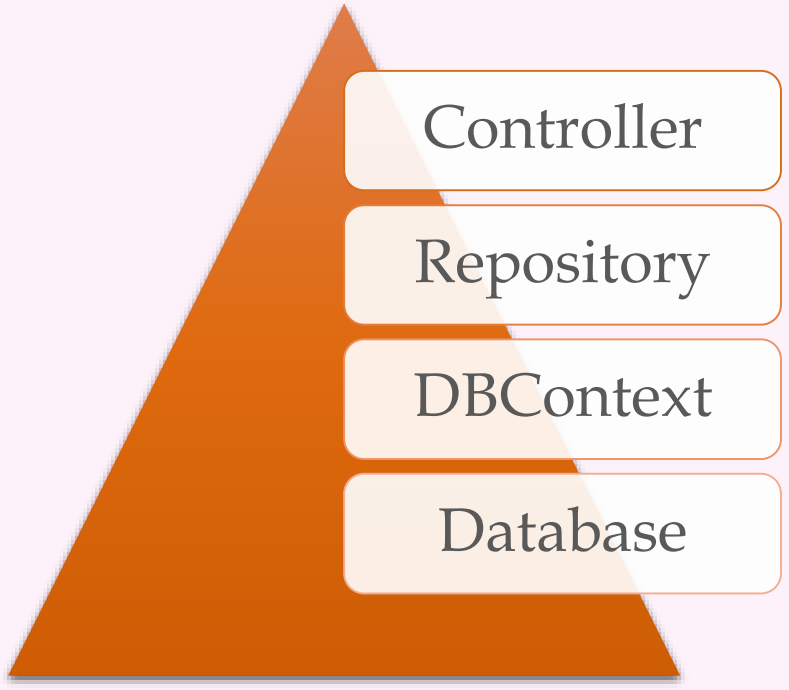


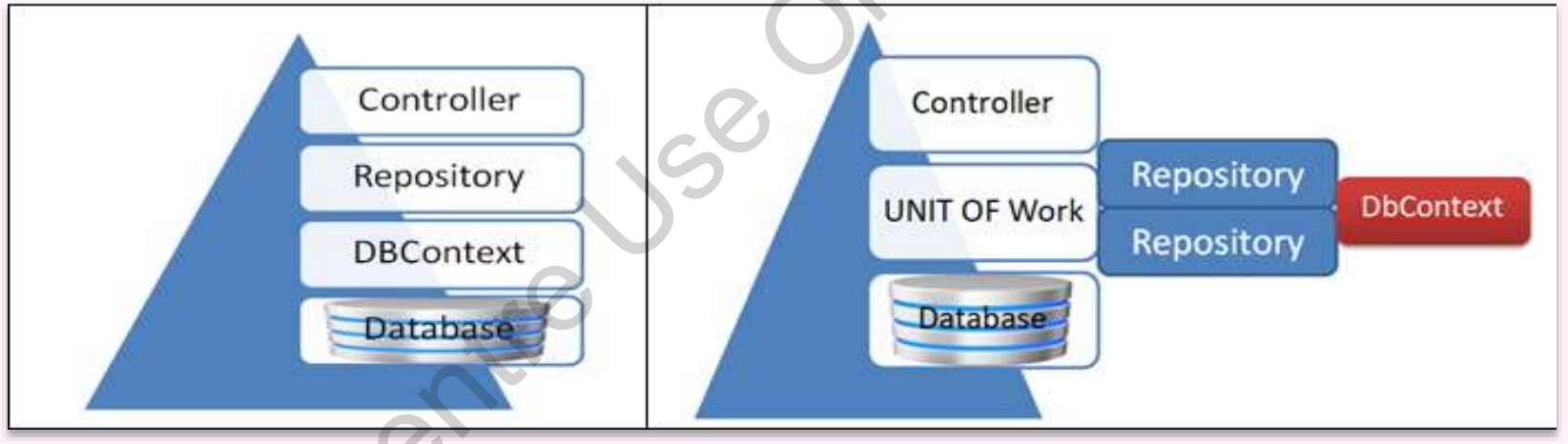
Figure 14.1: Repository Pattern

Easier Testing	Concerns are separated	Loose connection
With separation of layers, testing is easier.	Separate application functionalities depending on function makes the code easier to evolve and maintain.	When switching databases, changes are made only in the data layer but not in the data access layer. The number of changes or adjustments required in code specifically in the data access layer are reduced.

Benefits of Repository Pattern

Unit of Work

Figure 14.2: Repository and Repository with UoW



Benefits of Unit of Work

- Ensures flexibility in architecture.
- Ensures ease of testability.
- Increased abstraction.
- No redundancy in code though number of classes are more.
- Easy management of in-memory database operations.

Summary

- The repository pattern serves as an intermediary layer between an application's data access layer and its business logic layer.
- All database actions relating to a specific entity are defined using the non-generic repository approach within a separate class.
- The generic repository pattern is used to provide common database activities, such as CRUD operations, for all database entities in a single class.
- The UoW combines all CRUD transactions, such as insert/update/delete into one.
- UoW does not block any data table till it commits the changes.