**REPOSITORY PATTERN**

# Định nghĩa

Là một lớp trừu tượng để truy cập và thao tác dữ liệu. Nó tách biệt logic truy vấn dữ liệu ra khỏi logic nghiệp vụ

* **Lợi ích**
* Đơn giản hóa code: Giảm sự lặp lại của các truy vấn dữ liệu
* Dễ bảo trì: Thay đổi hoặc nâng cấp logic truy cập dữ liệu mà không ảnh hưởng đến code nghiệp vụ
* Tăng khả năng kiểm thử (testable): Dễ dàng thay đổi repository bằng mock trong unit test

# Cấu trúc cơ bản

* Thường bao gồm các thành phần sau:
* Entity: Lớp mô hình biểu diễn dữ liệu
* Irepository: interface định nghĩa các phương thức thao tác với dữ liệu
* Generic Repository: Repository chung xử lý CRUD cho mọi thực thể
* Concrete Repository: Repository cụ thể xử lý các thao tác đặc thù của một thực thể

# Triển khai

Ví dụ:

1. **Entity class:**

A screen shot of a computer code

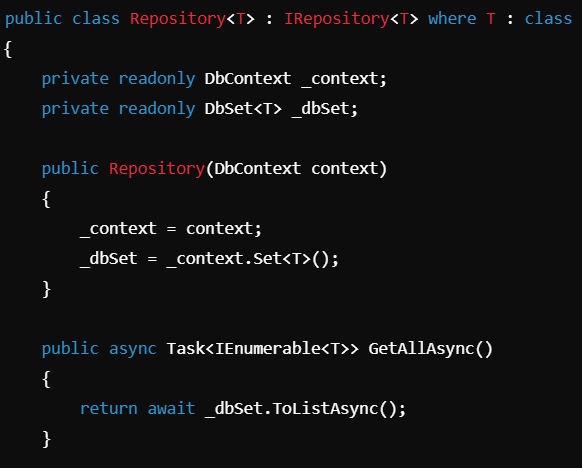
Description automatically generated

1. **IRepository Interface**

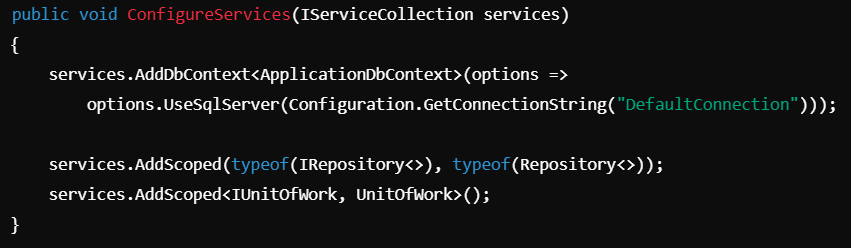
A screen shot of a computer code

Description automatically generated

1. **Generic Repository**



1. **Register Repository Pattern vào DI Container**



1. **Sử dụng Repository trong Controller**

A screen shot of a computer program

Description automatically generated

# Lưu ý khi sử dụng Repository Pattern

* Khi nào sử dụng:
* Ứng dụng phức tạp, có nhiều tầng nghiệp vụ cần xử lý
* Dễ dàng thay đổi nguồn dữ liệu (Database, API, v.v)
* Khi nào không nên:
* Ứng dụng nhỏ với logic đơn giản. Repository có thể tạo ra sự dư thừa

# Repository Pattern và CQRS

Repository Pattern: Tập trung vào CRUD

CQRS (Command Query Responsibility Segregation): Tách rời phần đọc (query) và ghi (Command), Thường được sử dụng với các dự án phức tạp