

# Using Entity Framework Core in our Controllers

---



**KEVIN DOCKX**

ARCHITECT

@KevinDockx <https://www.kevindockx.com>



# Coming Up



## Introducing the Repository Pattern

### Reading, Creating, Updating and Deleting Resources via Entity Framework Core

### Using AutoMapper



# Introducing the Repository Pattern

## No Repository Pattern

Code duplication

More error-prone code

Harder to test the consuming class

## Repository Pattern



# The Repository Pattern

An abstraction that reduces complexity and aims to make the code, safe for the repository implementation, persistence ignorant



# Introducing the Repository Pattern

## No Repository Pattern

Code duplication

More error-prone code

Harder to test the consuming class

## Repository Pattern

No duplication

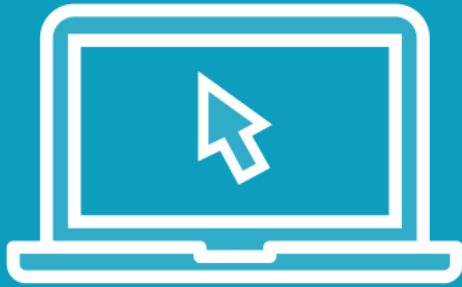
Less error-prone code

Better testability of the consuming class

Persistence ignorant: switching out the persistence technology is not the main purpose. Choosing the best one for each repository method is.



# Demo



## Introducing the Repository Pattern



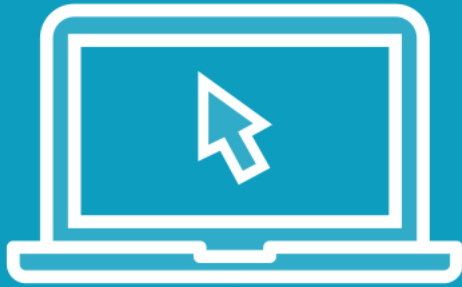
# Demo



**Returning Data from the Repository  
when Requesting Resources**



# Demo

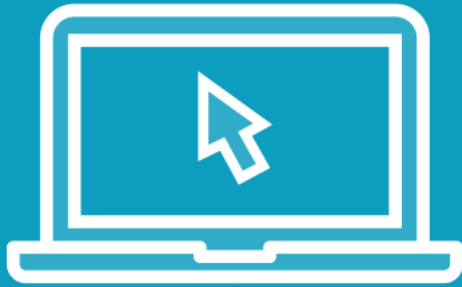


Using AutoMapper to map Between  
Entities and DTOs





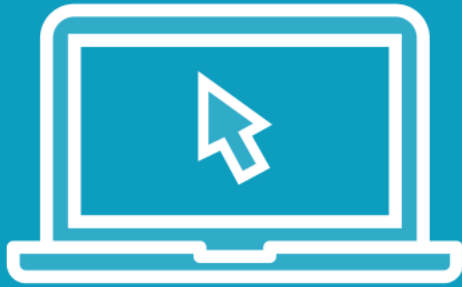
# Demo



## Creating a Resource



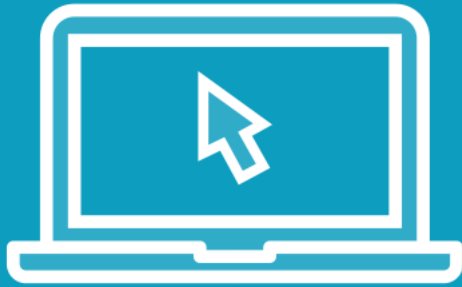
# Demo



## Updating a Resource



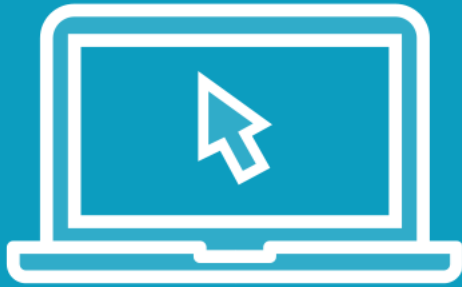
# Demo



## Partially Updating a Resource



# Demo



## Deleting a Resource



# Summary



The repository pattern is an abstraction that reduces complexity and aims to make the code, safe for the repository implementation, persistence ignorant

Using AutoMapper greatly reduces error-prone mapping code

You're ready to be AWESOME



@KevinDockx

