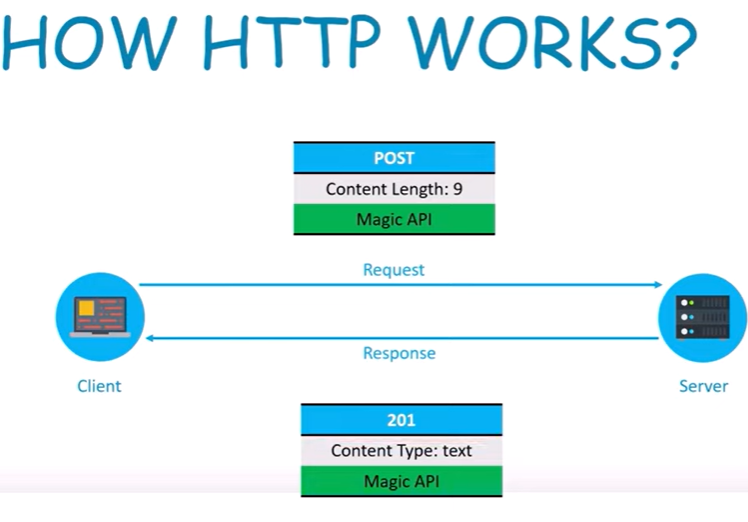
RESTFUL Web API

From - [RESTful API with .NET Core (.NET 7) - Full Course for Beginners (youtube.com)](https://www.youtube.com/watch?v=_uZYOgzYheU)

Way to transfer data

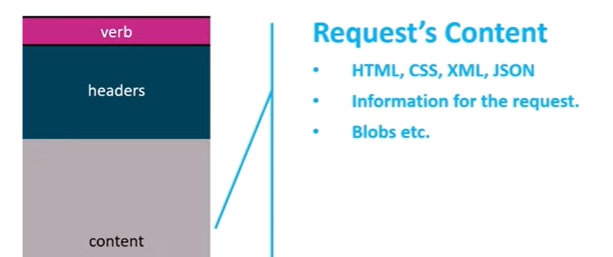
Way for application to communicate with each other

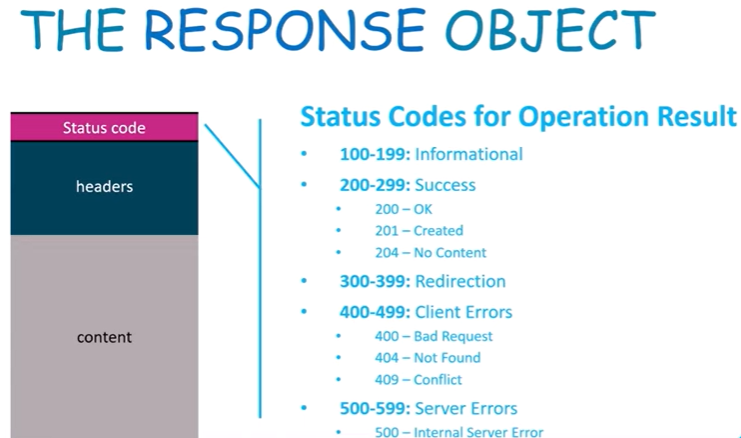


Stateless – will not remember the request

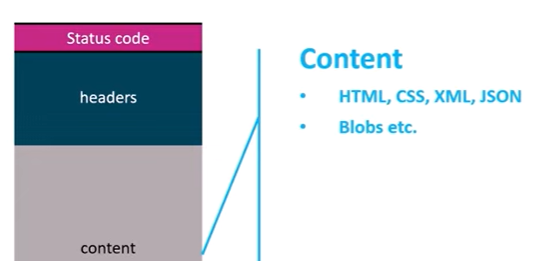












**Folder Structure**

Dependencies

Properties

Appsettings.json

Program.cs

**Controller**

A call with controller name and Derived from controllerBase

And has a attribute [ApiController] above the class

**Models**

Class file in Model folder

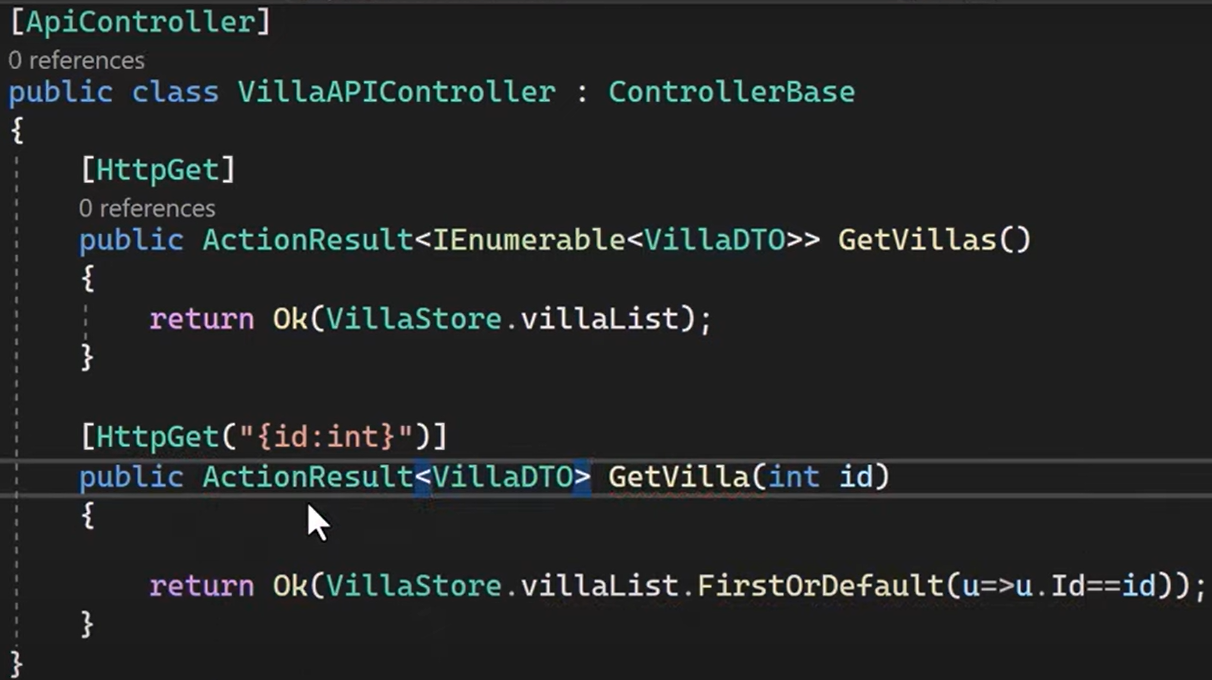
Has properties

**Route**

Above controller

[Route{“api/APIName”]

Or [Route{“api/[controller]”]



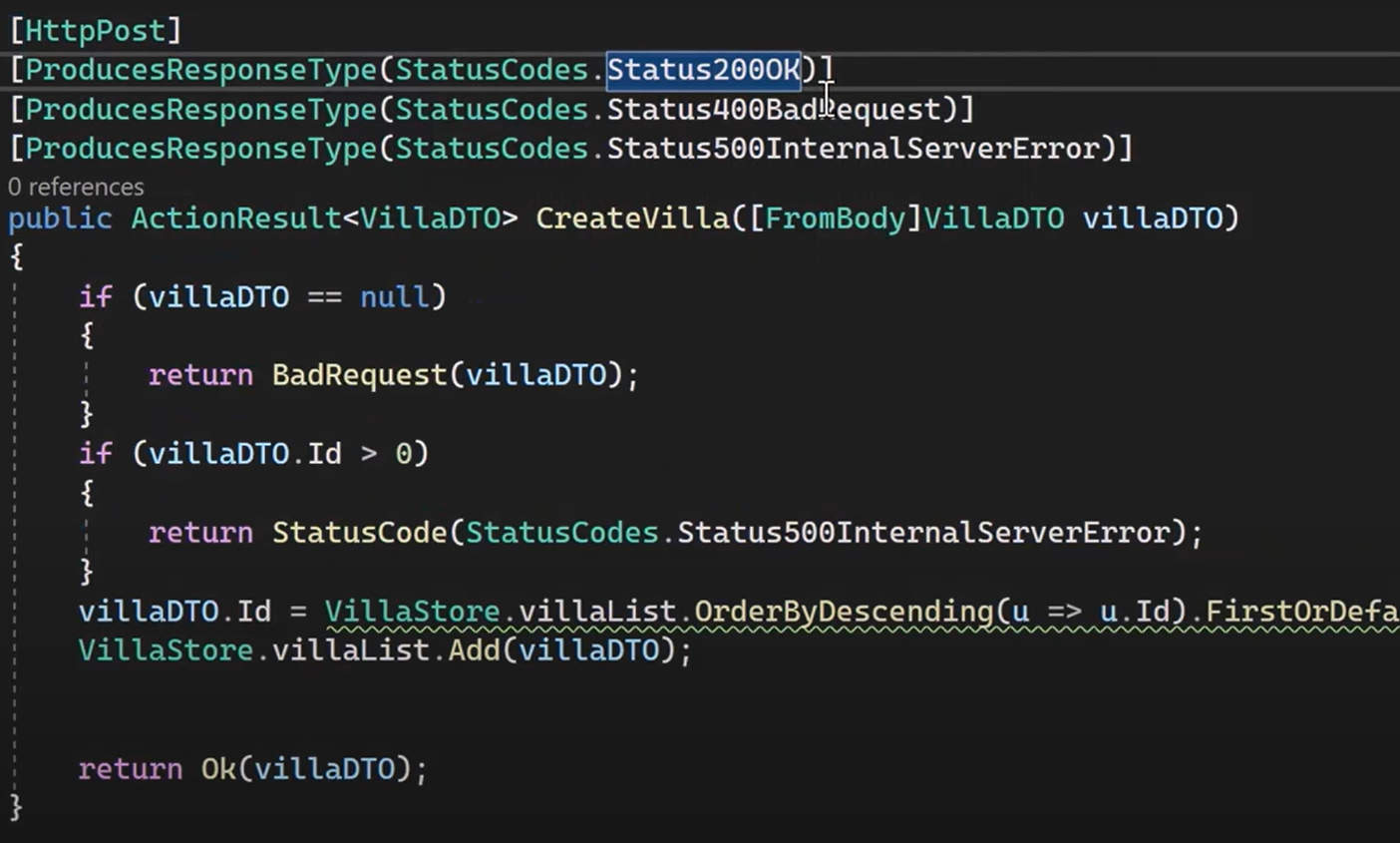
**DTO** – Data transfer Objects – what we need to expose to client. Like model has created date but DTO won’t

To **document** possible outcome of a request-

About action method

[ProducesResposeType(200, Type=typeof(Model))]

**HttpPost**



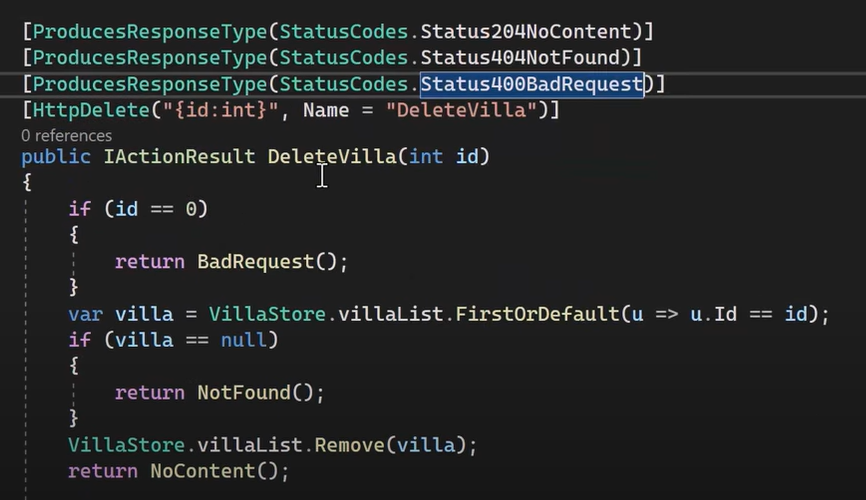
CreatedAtRoute method will give 201

Data Annotation to validate the records or ModelState.Valid

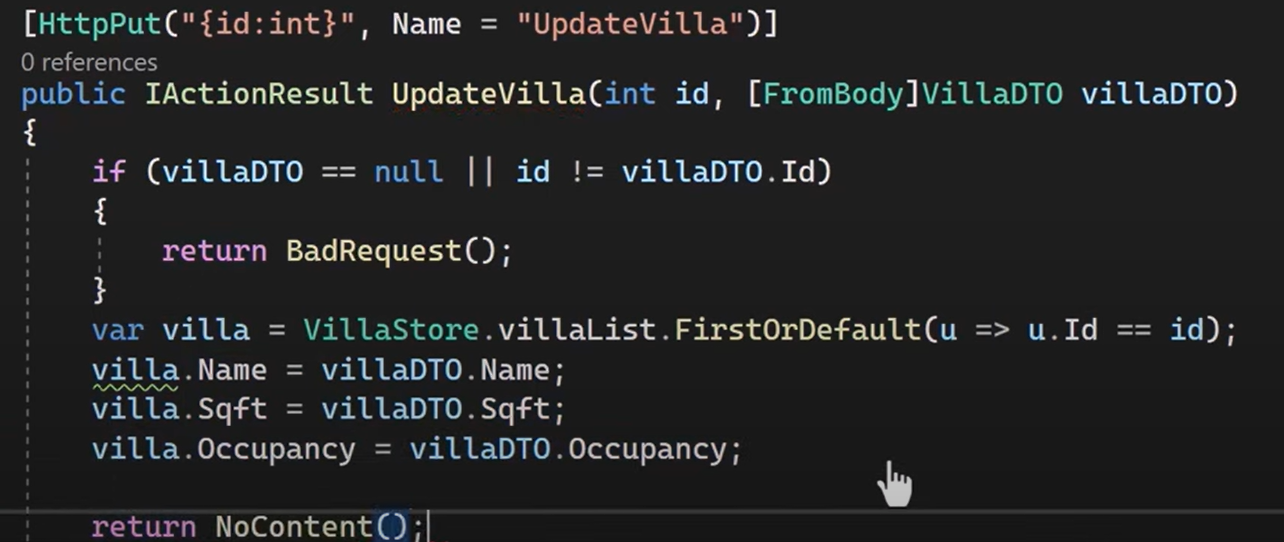
Custom Validation error – ModelState.AddModelError(Name, Message)

Return badReq(ModelRequest)

**HttpDelete**

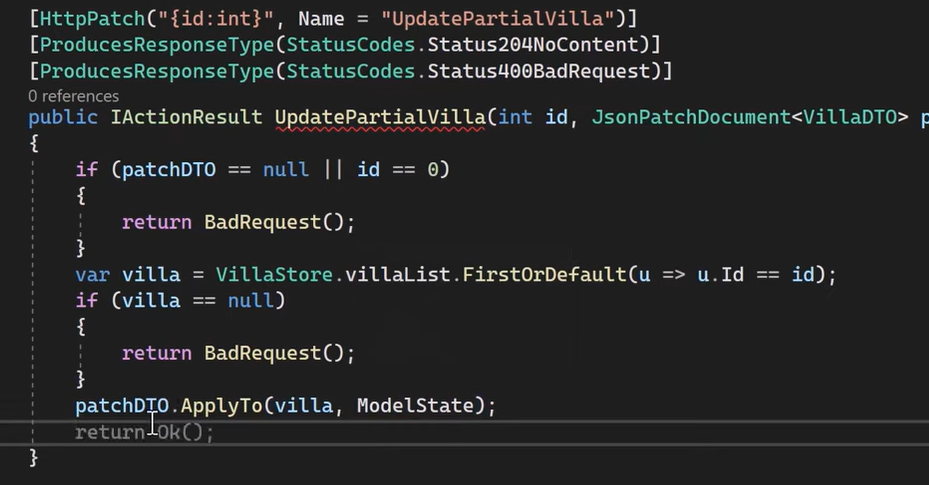


**HttpPut**



**HttpPatch**

Package – JSONPatch, NewtonsoftJSON



**Dependency Injection**

Constructor injection

Create a class and use it in constructor

And in program.cs

Builder.services.addScoped<Ilogging, logging>();

addSingleton – longest lifetime. created when app start and will be used everytime application request implementation

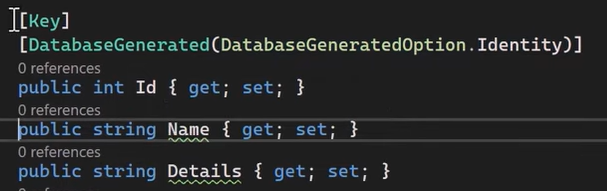
addScoped -for every request

add transient – every time object is accessed. Within a single request accessed multiple times

**Entity Framework Core –** Code First approach

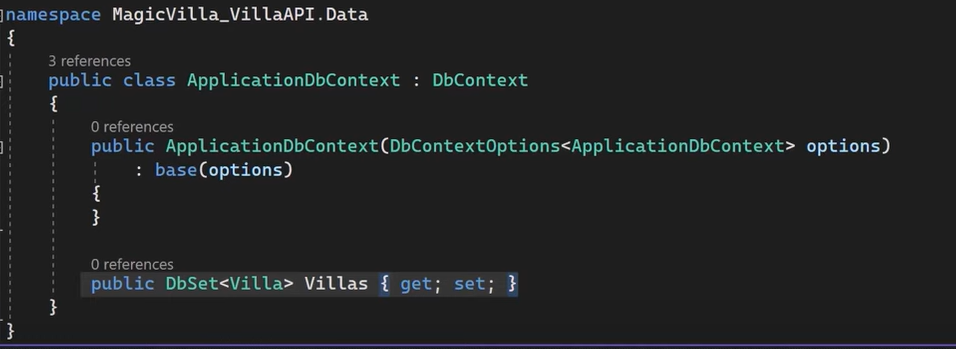
LINQ will be changed to SQL queries

Create a model class

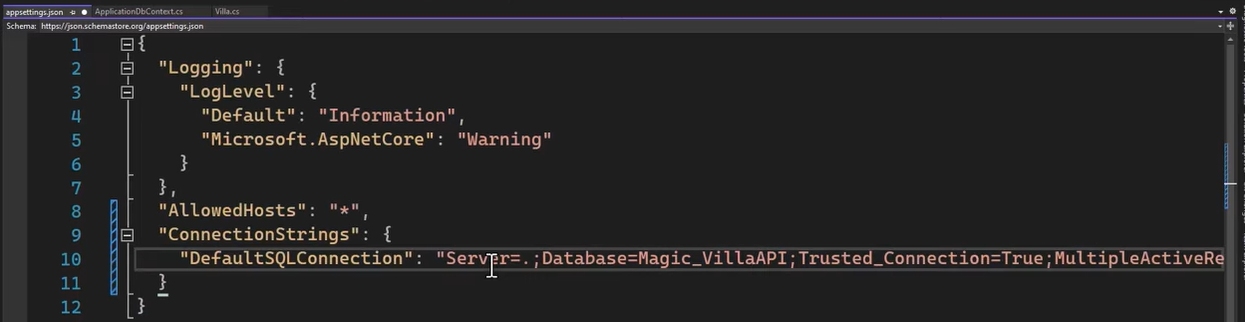


Add EFCore, EFcore.Tools sqlserver packages

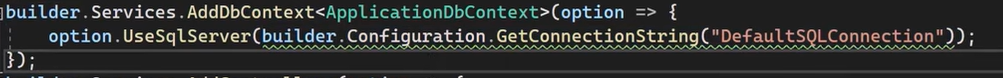
Create data folder and create a DBContext class which inherits from DBContext and create DBsets



Add connection string in appsettings.JSON



Register DB Context class and provide connection string

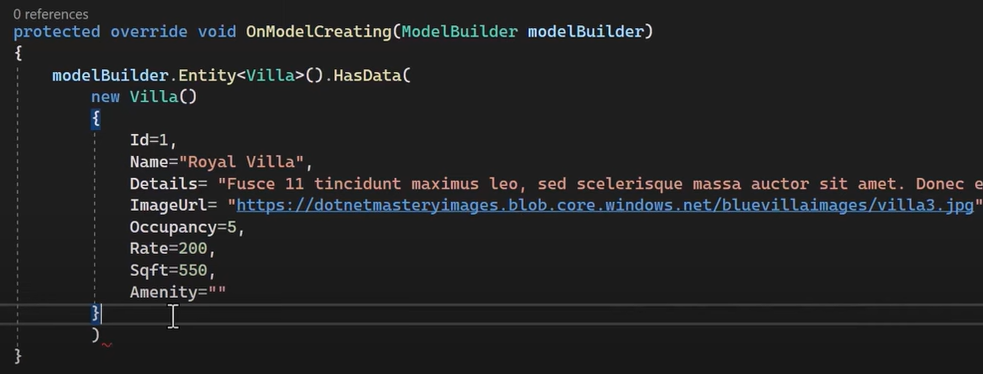


In PM Console –

add-migration name

Update database

Seeding data in DBContext



**To use DB Context in a class**

Create aprivate readonly variable \_var

In Ctor DI \_db=db;

\_db.DBSet.Remove(), .Update(), FirstOrDefault()

\_ db.Savechanges();

Not to track a record –

\_db.DbSet.AsNoTracking()