1. تفاوت AddScoped, AddTransient, AddSingleton چیست؟
   1. AddSingleton: یکبار در طول برنامه ساخته میشه و بین تمام درخواست ها همون یه instance داره استفاده میشه. زمانی که میخوایم یه چیز ثابت بین همه داشته باشیم یا یه سرویسی که stateless باشه یا یه State ثابت داشته باشه و یا اینکه ساخت اون سرویس (instance ساختن) هزینه بر باشه. نسبت به اون دوتای دیگه حافظه ی کمتری توی مموری اشغال میکنه (چون فقط یه بار ساخته میشه). باید در زمان ساخت این سرویس، حواسمون باشه که thread safe باشه.
   2. AddScoped: به ازای هر client request ساخته میشه. مناسب برای زمانیه که میخوایم توی یه درخواست state ها رو نگه داریم (و اون state ها مختص همون درخواست از سمت کلاینت، هستن). مثلا برای DbContext که میخوایم تغییرات رو track کنیم.
   3. AddTransient: به ازای هر بار که این سرویس درخواست بشه یه نمونه ازش ساخته میشه. ممکنه توی مراحل مختلف پردازش یه درخواست، بیشتر از یه بار یه سرویسی رو بخوایم استفاده کنیم اگه اون سرویس transient باشه، به ازای هر بار درخواست اون سرویس، ازش یه instance جدید ساخته میشه. برای سرویس های سبک و Stateless مناسبه که لازمه در هر بار استفاده یه instance جدید ساخته بشه. باید توجه داشت که همین هر بار ساختن از روی سرویس، میتونه باعث performance cost بشه.  
      نکته: توی scope کوچیکتر نمیشه از scope بزرگتر instance داشت. مثلا توی یه سرویس که transient هست، نمیشه سرویس scoped داشت.
2. Middleware ها چیستند و تفاوت آنها با Pipeline چیست؟
   1. Middleware ها قسمت های تشکیل دهنده یک pipeline هستند. در واقع زمانی که یک درخواست از طرف کلاینت به سرور می آید، از middleware ها باید گذر کند. هر middleware وظیفه دارد یک درخواست را به صورت ورودی گرفته و عملیاتی را روی آن درخواست (متناسب با وظیفه تعریف شده برای آن middleware) انجام دهد. پس از انجام عملیات اگر که middleware دیگری بعد از آن وجود داشت، نتیجه را به آن ارسال می کند. در غیر اینصورت نتیجه را به middleware قبلی که از آن درخواست را دریافت کرده بود ارسال می کند و middleware ای که در ابتدا درخواست را دریافت کرده بود، نتیجه نهایی را به کلاینت ارسال می کند. در هر کدام از middleware ها اگر مشکلی به وجود بیاید، ادامه پردازش متوقف شده و روند بازگشت به ابتدا انجام می شود.
   2. تفاوت دیگر اینکه pipeline ها معمولا یک روند خطی را دنبال می کنند یعنی مانند middleware ها روند بازگشت به ابتدا وجود ندارد.
3. Option pattern در configuration یک پروژه دات نت؟
   1. یک پترن برای کانفیگ قسمت های مختلف در پروژه است. زمانی که یک سرویس را میخواهیم به پروژه اضافه کنیم، احتمال اینکه بخواهیم یک کانفیگ custom روی آن پیاده سازی کنیم وجود دارد. این پترن اجازه می دهد که با یک ساختار مشترک در کد نویسی، کانفیگ های لازم را روی سرویس مشخص کنیم.
4. تفاوت بین TempData, ViewData, ViewBag؟
   1. ViewData: این یه نوع دیکشنریه و scope اش منحصر به یه request میشه. برای casting دیتای داخلش باید explicit عمل کنیم و بدرد پاس دادن دیتا به view میخوره.
   2. ViewBag: این یه حالت dynamic از همون ViewData هست و مثل این میمونه که یه class model رو داریم پاس میدیم (یه جورایی type-safe شده قبلی هست). Scope اش مثل قبلیه اما casting دیگه نداره.
   3. TempData: استفاده اصلی این واسه نگه داشتن دیتا در زمانی هست که میخوایم به request های دیگه ای هم بریم (مثلا redirection). اینم از نوع دیکشنریه و مثل اولی casting لازم داره.  
      مشکل دوتای اول اینه که compile error توشون نداریم. در نتیجه به راحتی ممکنه ارور پیش بیاد و حل کردنش هم زمانبر میشه. پس بهتره به جاشون از View model یا DTO استفاده کنیم.
5. تفاوت DTO و ViewModel؟
   1. DTO ها برای جابجایی دیتا به کار می روند اما ViewModel ها برای نمایش دیتا به کلاینت.
   2. ViewModel یک View که قرار است به کلاینت نمایش داده شود را هم شامل می شود اما DTO تنها شامل دیتا است.
6. مزایا و معایب استفاده از auto mapper؟
   1. مزایا:
      1. ساده سازی مپینگ به خاطر قابلیت object to object mapping.
      2. Type safe mapping به دلیل compile time checking.
      3. Config کردن و گذاشتن برخی قوانین برای مپینگ های پیچیده
   2. معایب:
      1. در زمان دیباگ، ممکن است پیدا کردن مشکل سخت باشد
      2. اگر به درستی config نشود، احتمال خطا بالا می رود.
      3. در custom mapping کنترل بیشتری روی مپینگ وجود دارد.
      4. Thread safe نیست (در واقع متد CreateMap آن است که این قابلیت را ندارد اما خود متد Map این مشکل را ندارد)
7. تفاوت بین SingleOrDefault و FirstOrDefault در Entity framework؟
   1. Single: انتظار داره فقط یدونه نتیجه بدست بیاد. صفر یا بیشتر از یکی اگه باشه توی اون Collection، ارور میده.
   2. SingleOrDefault: این میگه هیچی یا یدونه اگه داشته باشه قبوله. غیر از اون ارور میده.
   3. First: اولین موردی که پیدا کنه رو بر میگردونه اگه چیزی پیدا نکنه ارور میده.
   4. FirstOrDefault: اولین موردی که پیدا کنه رو برمیگردونه و اگه چیزی پیدا نکنه default رو بر میگردونه (این ارور نمیده)
8. تفاوت بین Find و First در Entity framework؟
   1. First یه متد LINQ هست که برای همه IEnumerable ها استفاده میشه اما Find یه متد مختص List هست.
   2. First در صورتی که چیزی پیدا نکنه ارور میده اما Find اگه پیدا نکنه null برمیگردونه و ارور نمیده.
   3. First داره از foreach و در نتیجه یه iterator استفاده میکنه ولی Find داره روی همون آرایه iterate میکنه. پس احتمالا Find یه مقدار سریعتر باشه.
9. تفاوت بین All و TrueForAll در EntityFramework (LINQ)؟
   1. هر دوتا برای چک کردن یه شرط برای کل element های یه collection استفاده میشن. اگه اون شرط روی همه شون صدق کنه true و در غیر اینصورت false میده. اما تفاوت این دوتا توی اینه که اولی داره از iterator و foreach استفاده میکنه اما TrueForAll داره مستقیما خود آرایه رو iterate میکنه و اجرا شدن اش هم هیچ memory allocation ای نداره در حالی که All یه مقدار کمی memory allocation داره.
10. تفاوت بین Code-first و Database-first در Entity framework؟
    1. در Code-first، اول مدل ها در کد برنامه ساخته میشه و بعدش از روی اونها جداول و دیتابیس ساخته میشه اما در database-first برعکسه و دیتابیس با جداولش اول ساخته میشه (یا از قبل ساخته شدن) و بعد Entity framework از روی اون جدول مدل ها رو می سازه.
11. Navigation property ها در entity framework چیستند؟
    1. بین دو جدول در دیتابیس ممکن است رابطه برقرار باشد (یک به یک، یک به چند، چند به چند). توی مدل های این جداول در کد، ما میتونیم یه property هایی تعریف کنیم که نماینده این رابطه هستن. مثلا یه کالا داریم و یه دسته بندی، توی مدل کالا، میتونیم یه پراپرتی از نوع Collection<Category> داشته باشیم که دسته بندی های اون کالا رو تعیین میکنه و نشون میده که کالا با مشتری یه رابطه یک به چند داره و نماینده این رابطه در کد، این پراپرتی هست. وقت هایی که کوئری به دیتابیس زده میشه که یه کالایی رو با دسته بندی هاش بهمون بده، توی این پراپرتی میاد حاصل دسته بندی های اون کالا در کوئری (که درواقع با یه join بدست میاد) رو میریزه و در دسترس ما در کد قرار میده.
12. کاربرد include در Entity framework؟
    1. استفاده از include مثل join زدن توی دیتابیسه. زمانی که ما یه class model داریم که داخلش یه پراپرتی از نوع class model دیگه ای رو داره (relation دارن به هم دیگه) و میخوایم زمان کوئری زدن، بگیم که اون پراپرتی دیگه (که در واقع میشه یه navigation property) رو هم واکشی کنه و برامون بیاره، از این تابع استفاده میکنیم.
13. Fluent api در EF چیه؟
14. راه های بالا بردن پرفورمنس در API؟
    1. Pagination
    2. Async logging
    3. Caching
    4. Payload compression
    5. Connection pool