**IActionResult vs ActionResult**  
ActionResult هو class أساسي بيرجع أنواع نتائج جاهزة زي ViewResult, JsonResult, RedirectResult.  
بينما IActionResult هو interface أوسع وبيسمح ترجع أي نوع نتيجة سواء built-in أو custom زي CsvResult.  
وده بيخلي IActionResult أكتر مرونة وقابل للتوسيع في السيناريوهات المختلفة.

**HttpContext (Request & Response)**  
الـ Request بيتكوّن من: الـ method (GET, POST …)، headers زي User-Agent و Authorization، body (بيانات JSON أو form)، و query string.  
أما الـ Response بيتكوّن من: status code زي (200, 404, 500 …)، headers زي Content-Type، والـ body اللي ممكن يكون HTML أو JSON أو أي نوع بيانات تاني.

**HTTP vs HTTPS**  
HTTP بيبعث البيانات كنص صريح غير مشفر، بينما HTTPS بيستخدم SSL/TLS علشان يشفر البيانات ويحميها من الاختراق.  
على سبيل المثال: إرسال بيانات الفيزا عبر HTTP ممكن يتجسس عليها أي هاكر، لكن مع HTTPS بتكون البيانات محمية.

**Clean URL & URL Mapping**  
الـ Clean URL بيخلي الرابط بسيط ومقروء، زي:  
mysite.com/products/123 بدل mysite.com/products.php?id=123.  
أما URL Mapping فهو اللي بيحوّل الرابط المقروء ده لمسار داخلي يقدر السيرفر يتعامل معاه.  
الميزة الأساسية: SEO أفضل وتجربة مستخدم أسهل.

**Segments & Fragments in URL**  
في الرابط: https://example.com/products/12#reviews  
الـ segments هي products و 12

بينما الـ fragment هو #reviews اللي بيوديك لجزء معين داخل الصفحة.

**Builder & Dependency Injection**  
الـ Builder بيساعد في بناء objects معقدة خطوة بخطوة.  
أما Dependency Injection ففكرته إن الكلاس ياخد الـ dependencies جاهزة بدل ما ينشئها بنفسه، وده بيخلي الكود أنظف وأسهل في الاختبار والتطوير.  
مثال عملي: في ASP.NET Core ممكن تسجل الـ repository في Startup.cs باستخدام

services.AddScoped<IProductRepository, ProductRepository>();

وبعدين أي service محتاجة الـ repository ده هيوصلها تلقائي.

**Razor Pages vs MVC**  
Razor Pages مناسبة أكتر للتطبيقات الصغيرة اللي الصفحة نفسها فيها الـ logic، بينما MVC بيفصل Model وView وController وده بيخليه أنسب للتطبيقات الكبيرة.  
مثال عملي: Blog صغير ممكن يتعمل بـ Razor Pages، لكن E-commerce platform كبيرة يفضل تتعمل بـ MVC.

**Content-Type in Response**  
بيحدد نوع البيانات اللي راجعة في الـ response.  
لو application/json يبقى ده API بيرجع JSON، لو text/html يبقى بيرجع صفحة ويب، ولو application/pdf يبقى الملف PDF.  
وده بيساعد المتصفح أو الـ client يعرف يتعامل مع البيانات بطريقة صحيحة.

**Minification, Bundling, Webpack, Lazy Loading**  
Minification معناها إزالة المسافات والتعليقات من الكود لتصغير الحجم.  
Bundling معناه دمج ملفات CSS أو JS مع بعض علشان يقلل عدد الطلبات.  
Webpack أداة قوية بتعمل bundling, minification وكمان code splitting.  
Lazy Loading معناه تحميل الصور أو الملفات لما تحتاجها بس مش من البداية.  
كل الحاجات دي هدفها الأساسي تحسين سرعة الموقع وأداء الشبكة.