**Why we use IActionResult not ActionResult**

* IActionResult is an **interface** that allows returning **any kind of result** (View, JSON, Redirect, etc.), which gives **flexibility**.
* ActionResult is a **base class** that implements IActionResult.  
  **Scenario:** If you may return different result types (e.g., Ok() or NotFound()), use IActionResult.

**What does the HttpContext request and response consist of?**

* **Request** → Headers, Method (GET/POST), URL, QueryString, Body, Cookies, User.
* **Response** → Status code, Headers, Body (content), Content-Type, Cookies.

**Difference between HTTPS and HTTP**

| **Feature** | **HTTP** | **HTTPS** |
| --- | --- | --- |
| Security | Not encrypted | Encrypted using SSL/TLS |
| Port | 80 | 443 |
| Use Case | Non-sensitive data | Sensitive data (login, payment) |

**Clean URL & URL Mapping**

* **Clean URL:** Readable and SEO-friendly (e.g., /products/details/5 instead of /products?id=5).
* **URL Mapping:** Mapping logical URLs to physical resources via **routing** (in ASP.NET: app.MapControllerRoute()).

**Segments and Fragments in URL**

* **Segment:** Each part between slashes → /products/details/5 → segments = products, details, 5.
* **Fragment:** Part after # used for client-side navigation → /about#team.

**Builder & Dependency Injection**

* **Builder:** Used to configure and build complex objects (e.g., in Program.cs, builder.Services.AddControllers()).
* **Dependency Injection:** Injecting required services into a class (e.g., injecting ILogger into a controller).  
  **Example:**

public class HomeController {

private readonly IEmailService \_email;

public HomeController(IEmailService email) {

\_email = email;

}

}

**Difference between Web Pages (Razor) and MVC**

| **Feature** | **Razor Pages** | **MVC** |
| --- | --- | --- |
| Structure | Page-based | Controller-based |
| Use Case | Simple CRUD apps | Complex apps |
| Example | /Pages/Product.cshtml | /Controllers/ProductController.cs |

**Business cases:**

* Razor Pages → Small admin panel.
* MVC → Large e-commerce site with multiple controllers.

**Content-Type in Response Message**

* Tells browser **what kind of data** is sent.  
  Example:
* text/html → HTML page
* application/json → API response
* Used to let client handle data properly.

**Minification, Web Bundle, Webpack & Lazy Loading**

* **Minification:** Removes spaces/comments from JS/CSS → smaller files.
* **Web Bundle:** Combines multiple files into one → fewer requests.
* **Webpack:** Tool that bundles and optimizes front-end resources.
* **Lazy Loading:** Loads components/images only when needed → faster initial load.  
  **All improve performance by reducing size and network load.**