

1 Why use IActionResult not ActionResult?

- ActionResult is a **class** that represents common return types (ViewResult, JsonResult, etc.).
- IActionResult is an **interface**, so it can represent *any* custom result (including ones not derived from ActionResult).


Scenario:

If later you want to return either FileResult (for downloading files) or a custom MyCustomResult, you can only guarantee flexibility if your method returns IActionResult.

2 HttpContext request & response message consist of?

- **Request** has: Request line (method + URL + version), headers, body.
 - **Response** has: Status line (status code + reason), headers, body.
-

3 Difference between HTTP and HTTPS

- **HTTP**: plain text, not encrypted → vulnerable.
 - **HTTPS**: HTTP + SSL/TLS → encrypted, secure.
 Business case: Online banking always uses HTTPS to protect passwords.
-

4 Segments vs Fragments in a URL

- **Segments**: The parts of the path separated by /.
- **Fragment**: The part after #, used only on client side (not sent to server).

Example:

https://site.com/products/phones/iphone#reviews

- Segments = products, phones, iphone
 - Fragment = reviews
-

5 Builder & Dependency Injection (DI)

- **Builder:** Configures and builds services before app runs.
- **DI:** Instead of creating objects with `new`, we let the framework inject them.

Real life example:

Imagine a car factory. Instead of each worker building their own tools (like `new Hammer()`), the factory provides the right tools to each worker (dependency injection).

6 Web Pages (Razor) vs MVC

- **Razor Pages:** Page-focused. Each page has logic + UI. Easier for simple apps.
- **MVC:** Separation of concerns (Model, View, Controller). Better for large apps.

Business cases:

- Razor: Personal blog, portfolio website (small).
 - MVC: E-commerce system with many modules.
-

7 Content-Type in response

- Tells browser how to interpret the body (JSON, HTML, XML, PDF, etc.).
 - Example: API response usually returns `application/json`.
-

8 Minification, Web Bundle, Webpack, Lazy Loading

- **Minification:** Remove spaces/comments → smaller JS/CSS.
- **Bundling:** Combine files into one → fewer requests.
- **Webpack:** Tool that handles bundling, minification, transpiling.
- **Lazy loading:** Load parts of the app only when needed.

💡 Together they reduce file size + number of requests → faster load time.